Universidad Eafit

Escuela de Ingeniería
Maestría en Ingeniería
Medellín, Colombia

Tesis de Maestría

AgileFM: Modelo de desarrollo ágil formal basado en la ISO/IEC 29110 para las micro, pequeñas y medianas empresas

Estudiante: Ing. Juan David Yepes González
Director: PhD. César Jesús Pardo Calvache

AgileFM: Modelo de desarrollo ágil formal basado en la ISO/IEC 29110 para las micro, pequeñas y medianas empresas

Juan David Yepes González

Tesis o trabajo de investigación presentada(o) como requisito parcial para optar al título de:

Magister en Ingeniería con Énfasis en Desarrollo de Software

Director (a):

Ph.D. César Jesús Pardo Calvache

Línea de Profundización:

Ingeniería de software

Universidad Eafit

Escuela de Ingeniería

Medellín, Colombia

2016

Nota de aceptación

_____

_____

_____

_____

Presidente del jurado

_____

Jurado

_____

Jurado

_____

Medellín, 18 de octubre de 2016

A mi familia que gracias a su apoyo incondicional y a su constante motivación logré la terminación de este trabajo.

Agradecimientos

Como muchas cosas en la vida, un trabajo de tesis no es una actividad que se realice sin la ayuda e intervención de otras personas. Estos individuos participan en distintos niveles, que gracias a su apoyo y colaboración se logra finalizar con éxito el objetivo trazado.

En primer lugar, agradezco a mi esposa: Luisa Viviana Calderón por su gran apoyo durante toda esta etapa de crecimiento tanto como personal y profesionalmente. Al estar siempre a mi lado pendiente de mis necesidades y progreso fue más que suficiente para llenarme de motivación y continuar con esta meta.

También debo agradecer a mi asesor: César Jesús Pardo Calvache, por su guía durante todo este proceso, ya que, sin él, difícilmente se hubiera podido alcanzar esta meta de forma tan satisfactoria.

A mis jefes: Jaime Walter Correa y Juan Carlos García que me ofrecieron los espacios y medios para poder alcanzar las metas logradas, además de su alto interés en mi proceso de crecimiento profesional y personal, disparan mi motivación a ser siempre mejor y a ayudar al crecimiento de la empresa.

Por último, pero no menos importante, a HMV Ingenieros Ltda. que deposito su fe en mi crecimiento profesional, otorgándome una beca de estudio y la disponibilidad para cumplir mis objetivos y ambiciones. Este hecho me motiva enormemente como parte del equipo de la empresa a seguir creciendo con ellos y a ayudar a la organización a conseguir sus objetivos.

Juan David Yepes, Medellín, Antioquia, Colombia.

Resumen

Actualmente, la gestión de proyectos de desarrollo de software en las micro, pequeñas y medianas empresas está atravesando un proceso de cambio gracias a la introducción de nuevos enfoques y modelos ágiles que se integran a los procesos existentes, permitiendo mejorar los tiempos, calidad, productividad y costos asociados. Sin embargo, la falta de comprensión y de una solución que soporte la integración adecuada con los procesos existentes de una organización, trae consigo una serie de obstáculos que entorpecen su exitosa implementación. Con el objetivo de ayudar a soportar el trabajo en la implementación de la norma ISO/IEC 29110 desde un enfoque aplicable a las micro, pequeñas y medianas empresas, se llevó a cabo la definición de un modelo de desarrollo ágil basado tanto en las prácticas definidas en la norma así como también en los principios del manifiesto ágil, en propuestas ágiles como eXtreme Programming, SCRUM y algunas técnicas empleadas actualmente en el desarrollo de software, por ejemplo: captura de requisitos, definición y documentación de arquitecturas de software, automatización de pruebas, entre otros. Esta tesis también muestra un estado del arte de los trabajos relacionados, el cual fue llevado a cabo a través de una revisión sistemática de la literatura, y que tuvo por objetivo conocer lo que se había realizado y logrado en micro, pequeñas y medianas empresas respecto a las metodologías ágiles principalmente y otros modelos desde el punto de vista de: tendencias, propuestas, experiencias, factores de éxito, entre otros. El modelo propuesto ha sido aplicado y validado a través de un estudio de caso realizado en la compañía HMV Ingenieros Ltda.

Palabras clave: metodologías ágiles; revisión sistemática; SCRUM; eXtreme Programming; ISO/IEC 29110; CMMI; micro pequeñas y medianas empresas (MiPyMEs).

Abstract

Currently, the project management of software development in smalls and medium enterprises (SMEs) is in a process of change through the introduction of new approaches and agile models which are integrated to the existing processes, enabling to improve: time, quality, productivity and associated costs. However, the lack of understanding and a solution that allows organizations to carry out their implementation in an integrated way with legacy processes brings a set of obstacles which obstruct their successful implementation. In order to help support the work in implementing the ISO/IEC 29110 standard from an applicable m to small and medium enterprises, carried out the definition of a model agile development based on both the practices defined in standard as well as the principles of the agile manifesto, agile proposals as eXtreme Programming, SCRUM and some techniques currently employed in the development of software, for example capture requirements definition and documentation of software architectures, test automation , among others. This thesis also shows the state of art related work, which was carried out through a systematic review of the literature, and that was to identify what has been had done and achieved in small and medium enterprises mainly regarding agile methodologies and other models from the point of view of: trends, proposals, experiences, success factors, among others. The proposed model has been applied and validated through a case study in the company HMV Ingenieros Ltda.

Keywords: agile methodologies; systematic review; SCRUM; eXtreme Programming; ISO/IEC 29110; CMMI; Small and medium enterprises (SMEs).

Contenido

Pág.

Lista de figuras…. 17

Lista de tablas…. 19

Glosario……. 21

Capítulo 1. Introducción 23

1. Introducción……. 25
1.1 Planteamiento del problema y motivación 25
1.2 Hipótesis y objetivos 28
1.2.1 Objetivo general 28
1.2.2 Objetivos específicos 29
1.2.3 Estructura del documento 29

Capítulo 2. Método de Investigación 31

2. Método de Investigación 33
2.1 Métodos de investigación en ciencias de la computación 33
2.2 Solución de problemas integrado y método de investigación en ingeniería de sistemas 34
2.3 Marco de trabajo Action-Research-Evaluation 36
2.4 Estudios de caso 37
2.5 Estrategia de investigación en esta tesis 38

Capítulo 3. Estado del Arte 41

3. Marco Teórico y Estado del Arte 43
3.1 Marco teórico: metodologías y modelos para el desarrollo de software 43
3.1.1 Capability maturiy model integration (CMMI) 43
3.1.2 Estándar ISO/IEC 29110 43
3.1.3 Marco de referencia SCRUM 44
3.1.4 Metodología eXtreme Programming (XP) 47
3.2 Estado del arte 47
3.2.1 Formulación de la pregunta 48
3.2.2 Selección de las fuentes 48
3.2.3 Selección de los estudios y extracción de la información 49
3.3 Resultados y discusión 52

3.3.1 Tendencia de las publicaciones....52
3.3.2 Países de estudio....53
3.3.3 Clasificación por tipo de industria....54
3.3.4 Clasificación por tamaño de empresa....55
3.3.5 Clasificación por tamaño de los equipos....57
3.3.6 Modelos utilizados....57
3.3.7 Software safety....59
3.3.8 Factores de éxito extraídos de los estudios analizados....60
3.4 Conclusiones....61

Capítulo 4. AgileFM: Modelo de gestión de proyectos ágiles formal basado en el estándar ISO/IEC 29110....65

4. AgileFM: Modelo de gestión de proyectos ágiles formal basado en el estándar ISO/IEC 29110….....67
4.1 Elementos del modelo propuesto....67
4.2 Roles del modelo....69
4.2.1 Product Owner....69
4.2.2 Stakeholder claves....69
4.2.3 SCRUM Master....70
4.2.4 Equipo de desarrollo....70
4.3 Herramientas....71
4.3.1 Herramientas de modelado....72
4.3.2 Herramientas de desarrollo....72
4.3.3 Herramientas de gestión....73
4.3.4 Herramientas de mantenimiento y de integración....74
4.3.5 Herramientas de pruebas....74
4.3.6 Herramientas de aprendizaje....75
4.4 Artefactos....76
4.4.1 Impact Mapping....76
4.4.2 User Story Mapping....78
4.4.3 Plan de negocio....79
4.4.4 Historias de usuario....81
4.4.5 Casos de prueba....83
4.4.6 Product Backlog....83
4.4.7 Sprint Backlog....84
4.4.8 Repositorio documentación de la solución....84
4.4.9 Release Planning....84
4.4.10 Burndown....86
4.4.11 Repositorio de gestión de cambios....87
4.4.12 Repositorio de lecciones aprendidas....87
4.4.13 Repositorio de resultado de pruebas....88
4.4.14 Tableros Kanban....89
4.5 Ceremonias....90
4.5.1 Reuniones de iniciación....90
4.5.2 Sprint Planning....91
4.5.3 Daily meeting....92
4.5.4 Sprint Review....93
4.5.5 Sprint Retrospective....93
4.5.6 Release Review....94
4.5.7 Release Retrospective....95

4.5.8 Reuniones de entregables 95
4.5.9 Reuniones de cambios urgentes 95
4.6 Técnicas y procesos 96
4.6.1 TDD / ATDD 96
4.6.2 Programación en pares 97
4.6.3 Planning Poker 98
4.6.4 Refactoring 100
4.6.5 Mad, Sad, Glad 100
4.7 Ejecución del proceso 101
4.8 Comparativo 104
4.8.1 Comparación de actividades 104
4.8.2 Comparación de documentos requeridos 106

Capítulo 5. Estudio de Caso 109

5. Estudio de caso 111
5.1 Pregunta de investigación 111
5.2 Selección del Estudio de caso 112
5.3 Diseño de la actividad académica 112
5.4 Unidades de Análisis 113
5.5 Procedimiento de campo y recolección de información 113
5.6 Intervención en el estudio de caso 114
5.6.1 Ejecución del primer ciclo 114
5.6.2 Análisis de resultados del primer ciclo 115
5.6.3 Retroalimentación en el segundo ciclo 118
5.6.4 Plan de acción para el segundo ciclo 119
5.6.5 Ejecución del segundo ciclo 119
5.6.6 Análisis de resultados del segundo ciclo 119
5.6.7 Contexto con el estándar ISO/IEC 29110 121
5.7 Plan de validación y limitaciones del estudio de caso 124

Capítulo 6. Conclusiones y recomendaciones 127

6. Conclusiones y Recomendaciones 129
6.1 Conclusiones 129
6.2 Recomendaciones 131
6.3 Trabajo Futuro 132

Bibliografía…….. 135
Anexo A. Encuesta Implementación y Adopción del Método Ágil Propuesto 141
Anexo B. Encuesta de Satisfacción para el Product Owner 143

Lista de figuras

Pág.

Figura 2-1: Enfoque para solución de problemas. 34
Figura 2-2: Propuesta de método de investigación. 35
Figura 2-3: Propuesta Action-Research-Evaluation Framework 36
Figura 2-4: Enfoque empleado en la investigación 38
Figura 3-1: Tendencia de publicaciones 52
Figura 3-2: Distribución de los estudios por país 54
Figura 3-3: Clasificación por tipo de industria 55
Figura 3-4: Clasificación por tamaño de la empresa 56
Figura 3-5: Clasificación por tamaño de los equipos 58
Figura 3-6: Modelos utilizados 58
Figura 4-1: Modelo de gestión de proyectos de software propuesto 68
Figura 4-2: Plataforma de Desarrollo de Software 71
Figura 4-3: Artefactos Propuestos 76
Figura 4-4: Estructura Impact Mapping 77
Figura 4-5: User Story Mapping 78
Figura 4-6: Ejemplo Burndown 86
Figura 4-7: Ciclo de desarrollo de TDD 96
Figura 4-8: Ejemplo planning poker 99
Figura 4-9: Proceso desarrollo de software 102
Figura 4-10: Relación entre la Gestión de Proyecto y la Implementación del Software. 105
Figura 4-11: Proceso de Gestión de Proyectos [38] 105
Figura 4-12: Proceso de Implementación del Software [38] 105
Figura 5-1: Resultados pregunta 1 anexo A 116
Figura 5-2: Resultados pregunta 2 anexo A 116

Lista de tablas

		Pág.
Tabla 3-1:	Cadena de búsqueda básica	48
Tabla 3-2:	Ejemplo de la plantilla utilizada para almacenar la información	50
Tabla 3-3:	Clasificación por tipo de estudio	53
Tabla 3-4:	Factores de éxito	60
Tabla 4-1:	Estructura plan de negocio	79
Tabla 4-2:	Atributos de calidad de software	80
Tabla 4-3:	Plantilla atributos de calidad	81
Tabla 4-4:	Estructura Historia de Usuario	81
Tabla 4-5:	Estructura casos de prueba	83
Tabla 4-6:	Estructura simple de un Release Planning	85
Tabla 4-7:	Procesos ISO/IEC 29110 vs AgileFM	106
Tabla 4-8:	Documentación ISO/IEC 29110 vs AgileFM	106
Tabla 5-1:	Plan de acción inicial	112
Tabla 5-2:	Artefactos y herramientas establecidas	115
Tabla 5-3:	Asignación de roles en el proyecto	115
Tabla 5-4:	Nuevas actividades adicionadas al plan de acción	119
Tabla 5-5:	Consumo de tiempo y costo del proyecto	120
Tabla 5-6:	Consumo de tiempo y costo del proyecto de gestión de proveedores	121
Tabla 5-7:	Procesos del estándar ISO/IEC 29110 y AgileFM en HMV	122
Tabla 5-8:	Documentación del estándar ISO/IEC 29110 y AgileFM en HMV	122

Glosario

ATDD: Acceptance Test-Driven Development.

BPMN: Business Process Model and Notation

DMS: Document Management System.

CACIED: Congreso Andino de Computación, Informática y Educación.

CMMI: Capability Maturity Model Integration.

CMMI-ACQ: Capability Maturity Model Integration for Acquisition.

CMMI-DEV: Capability Maturity Model Integration for Development.

CMMI-SVC: Capability Maturity Model Integration for Services.

CMS: Content Management System.

EPC: Engineering Procurement and Construction Management.

ERP: Enterprise Resource Planning.

ESPOL: Escuela Superior Politécnica del Litoral.

IDE: Integrated Development Enviroment

IEC: International Electrotechnical Commission

ISO: International Organization for Standardization.

MiPyME: Micro, pequeña y mediana empresa.

PCH: Pequeña Central Hidroeléctrica.

PM: Project Management.

PMI: Project Management Institute.

SP: Software Implementation.

SPEM: Software & Systems Process Engineering MetaModel.

SPLE: Software Product Line Engineering.

TDD: Test-Driven Development.

UML: Unifield Modeling Language.

XP: eXtreme Programming.

Capítulo 1. Introducción

"La verdadera motivación se consigue con el logro, el desarrollo personal, la satisfacción con el trabajo y el reconocimiento",
(Frederick Irving Herzberg, renombrado psicólogo que fue de las personas más influyentes en la gestión administrativa de empresas, 1923-2000)

Este capítulo presenta una descripción y motivación para la realización de la tesis, además, detalla la hipótesis, los objetivos de investigación y la estructura de este trabajo.

1. Introducción

1.1 Planteamiento del problema y motivación

En las últimas décadas, las micro, pequeñas y medianas empresas (MiPyMEs) han emergido, crecido y evolucionado impactando la economía de todos los países a nivel internacional. Las MiPyMEs desarrolladoras de software por su parte, han permitido extender el mercado de los países en nuevas áreas económicas completamente nuevas y de alto impacto para otros sectores, convirtiéndose en una de las principales industrias para muchos países, por ejemplo: India, Irlanda, Uruguay, Estados Unidos, Brasil, México, Colombia, entre otros. Según el reporte del 2013 de la Comisión Europea, el sector de las tecnologías de la información experimentó un crecimiento del 10% en el 2011, el cual ha seguido aumentando año tras año de forma positiva [1].

De acuerdo a [2] y [3], un gran porcentaje de empresas de software son MiPyMEs que desarrollan productos y servicios de gran valor para distintas economías, por ejemplo: el sector industrial, agroindustrial, salud, comunicaciones, bancario, gobierno, militar, entre otros. Por otra parte, dado que la exigencia de la calidad ha aumentado de forma paralela a la evolución de las tecnologías, los productos de software requieren de buenas prácticas que permitan no sólo garantizar su calidad, sino también que sean acorde y adaptadas a las características del tipo de industria que las desarrolla, por ejemplo: tamaño y tipo de industria.

En los últimos años, las metodologías ágiles han cobrado especial interés en el sector académico y profesional, permitiendo principalmente dotar a las MiPyMEs de prácticas que favorecen su quehacer y la gestión de sus recursos desde enfoques ágiles y sin mayores formalismos que las metodologías tradicionales. Algunos de los enfoques ágiles más utilizados son eXtreme Programming (XP) [4], SCRUM [5], Lean [6] y Kanban [7]. La adopción de estos modelos ha permitido la gestión de proyectos de desarrollo de

software, el control de requisitos variables, la gestión efectiva y eficaz de los grupos de trabajo y el involucramiento y empoderamiento del cliente dentro del proyecto. Sin embargo, las empresas de desarrollo de software, en especial las MiPyMEs no siempre están preparadas para afrontar la implementación y adopción rápida y eficiente de un enfoque o metodología ágil, debido en primer lugar a hábitos obtenidos de sus métodos tradicionales que hacen más difícil el cambio de mentalidad, y segundo, por la falta de un verdadero entendimiento de los valores, principios y procesos en los cuales se basan las metodologías ágiles. Además, los métodos o marcos de trabajo ágiles generalmente se centran en el proceso de desarrollo del producto software e ignoran o dejan a la definición de las organizaciones aspectos importantes en la gestión de un proyecto como el relacionamiento con: (i) los clientes y proveedores, (ii) el manejo de la documentación, (iii) los procesos que apoyan la mejora continua, (iv) los sistemas de métricas o de lecciones aprendidas y (v) el manejo de la trazabilidad de todo el proyecto.

Además, también ha surgido un nuevo estándar pensado para MiPyMEs apalancado por la International Organization for Standardization (ISO) que busca ayudar a estas organizaciones a implementar un proceso maduro y organizado que permita soportar el desarrollo de proyectos de software. Este estándar es el ISO/IEC 29110, pero que al igual que los métodos mencionados anteriormente, las organizaciones no cuentan con la capacidad de asimilar este estándar, muchas veces debido a lo extenso y complejo de su documentación para el tipo de industria que caracteriza a las micro empresas, lo cual requiere gran cantidad de tiempo por parte de la empresa y de sus equipos de trabajo para su conceptualización, implementación y adopción, siendo un contexto utópico para la mayoría de los casos, debido a que en un medio tan competitivo, este tiempo representa dinero y oportunidad en el mercado de la organización que no se puede dar el lujo de perder. Por otra parte, están los paquetes de despliegue (deployment packages) que se dan para este estándar, con el objetivo de facilitar a las empresas la implementación de un proceso pre-definido para minimizar costos asociados a la implementación, sin embargo, aunque estos paquetes permiten cumplir con los requerimientos establecidos por la norma, no permiten que las empresas sean flexibles en la definición de sus procesos de acuerdo a sus preferencias, características y necesidades, lo que resulta en muchos casos la implementación de un proceso desprovisto del alma empresarial.

Es debido a estos factores que las MiPyMEs prefieren en muchos casos adoptar procesos tradicionales, ya conocidos o en algunas ocasiones prefieren diseñar procesos informales con el fin de responder rápidamente a la demanda del mercado y convivir con los inconvenientes que se les presenta en el día a día.

En este sentido, y teniendo en cuenta lo anterior, en este trabajo se presenta un modelo ágil de desarrollo de software formal denominado AgileFM, pensado para las MiPyMEs, en donde se enlazan conceptos de varias disciplinas ágiles, haciendo énfasis en un proceso de iteraciones cortas que ofrece flexibilidad a la organización para responder rápidamente a los cambios constantes, la utilización de herramientas y ceremonias (eventos o reuniones) que promuevan la comunicación entre todos los participantes del proyecto, la utilización de técnicas y herramientas de desarrollo que soportan el diseño arquitectónico, la codificación y la realización y automatización de pruebas, y el uso de herramientas de trabajo colaborativo entre el equipo de trabajo, el cliente y de más interesados, esto con el fin de dar mayor transparencia al proceso, facilidad en la toma de decisiones y una visión global compartida del alcance del proyecto, enfocada en la gestión del conocimiento para que permita apalancar el aprendizaje y la mejora continua de los individuos y los procesos dentro de la organización. Esto, a través de un mapeo con el estándar ISO/IEC 29110, con el fin de construir un modelo que ayude y guíe a las organizaciones a cumplir con satisfacción sus proyectos de software de manera más ágil, y que, además, puedan durante el proceso, mejorar su rendimiento, desempeño, competitividad, poder llevar de forma más fácil y óptima la documentación del proyecto, incentivando a la vez la gestión del conocimiento dentro de la empresa y posteriormente ayude a un proceso de certificación.

Teniendo en cuenta los aspectos mencionados en la presente introducción, el objetivo de esta tesis es presentar un modelo de proceso de gestión de proyectos de software formal con un enfoque ágil denominado AgileFM, pensada para equipos de trabajo pequeños y para MiPyMEs de software. La propuesta reúne elementos importantes de enfoques populares como son eXtreme Programming y SCRUM, además de varias herramientas y técnicas empleadas en el desarrollo de software y los divide en seis componente: i) Los roles, que definen el papel que va a desempeñar cada integrante del proyecto, ii) los artefactos, que son instrumentos o procedimientos que darán soporte a toda la ejecución del proyecto, desde la etapa inicial de captura de requisitos, hasta el despliegue final de

la solución, iii) ceremonias o eventos, que ofrecen los espacios y mecanismos necesarios para promover la comunicación entre todos los participantes, ya sean para definir requisitos o aclarar dudas de los mismos, como para realizar retroalimentaciones del trabajo realizado, iv) las técnicas y procesos, que dan soporte a la estimación, codificación y retroalimentación dentro del proyecto, v) las herramientas de software, que ayudan al trabajo colaborativo, a la automatización de pruebas, a la gestión de conocimiento y al control de calidad del producto final y vi) el proceso de gestión de desarrollo de software, que basa su esencia en un trabajo iterativo, centrado en la gestión del conocimiento y mejoramiento continuo de los individuos y los procesos corporativos. Lo anterior, con el objetivo de facilitar la integración del estándar ISO/IEC 29110 con un enfoque ágil que pueda ser aplicado más fácilmente en una MiPyME de software trayendo consigo beneficios como: i) un proceso de desarrollo de software más formal, ii) flexibilidad en la aplicación de otras prácticas, iii) agilidad por la incorporación de prácticas menos complejas, iv) competitividad para participar de mercados nacionales e internacionales y v) posibilidad de conducir y/o alcanzar una certificación.

1.2 Hipótesis y objetivos

La hipótesis principal de esta tesis de maestría es:

> Es posible implementar un enfoque de gestión de proyectos de software ágil basado en la norma ISO/IEC 29110 desde un enfoque ágil que permita a la MiPyMEs mejorar el desempeño de su proceso de desarrollo, ser más competitivos en el mercado y lograr una certificación.

De acuerdo a la hipótesis anterior, el objetivo principal y objetivos específicos se presentan a continuación.

1.2.1 Objetivo general

Definir un modelo de gestión de proyectos de software ágil que cumpla con los lineamientos del estándar ISO/IEC 29110 y que permita a las MiPyMEs de software ser más eficientes y a la vez puedan certificar su proceso.

1.2.2 Objetivos específicos

- Realizar una revisión sistemática de la literatura, para conocer el estado actual de los enfoques ágiles en la industria y su nivel de implementación y adopción.
- Proponer un modelo de gestión de proyectos ágiles que ayude a las organizaciones a formalizar su proceso de desarrollo y ser más eficientes y competitivos en el mercado.
- Validar el enfoque propuesto en un estudio de caso de la región.

1.2.3 Estructura del documento

Este documento está estructurado en seis capítulos:

- Capítulo 1: consiste de la introducción, en la cual se describen las razones para la realización de esta tesis de maestría, además, se presenta la hipótesis, el objetivo principal, los objetivos específicos y la estructura del documento.
- Capítulo 2: presenta el método de investigación empleado durante el desarrollo de este trabajo, el cual ha utilizado métodos diseñados para la investigación en áreas de las tecnologías de la información y afines.
- Capítulo 3: presenta una revisión del estado del arte, en el cual se realiza una revisión sistemática de la literatura acerca del estado actual de las metodologías ágiles y su implementación junto a otros modelos en las MiPyMEs. De esta investigación se obtiene una síntesis de los trabajos y un análisis estadístico del comportamiento de los enfoques ágiles más utilizados, tanto en la academia como en la industria, los cuales finalmente son traducidos en factores claves a tener en cuenta a la hora de implementar y adoptar un método ágil en una organización. Durante la realización del estado del arte, se llevó a cabo la publicación de dos artículos:
    - En la Revista Tecnológica ESPOL [8] como edición especial de los mejores artículos presentados en el II Congreso Andino de Computación, Informática y Educación (CACIED'15) celebrado en Guayaquil, Ecuador [9].
    - En la XVI Convención y Feria Internacional – INFORMÁTICA 2016 celebrado en la Habana, Cuba [10].
- Capítulo 4: presenta la propuesta a partir de la investigación previamente realizada de un modelo de gestión de proyectos de software dividida en seis categorías.

- Capítulo 5: en este capítulo se desarrolla el estudio de caso de caso implementando el modelo planteado en el capítulo 4.
- Capítulo 6: presenta las conclusiones y recomendaciones adquiridas durante la investigación y construcción de este trabajo.

Capítulo 2. Método de Investigación

"Cada día sabemos más y entendemos menos", (Albert Einstein, Físico teórico alemán, 1897-1955)

Este capítulo presenta el método de investigación el cual fue usado para lograr los objetivos definidos de la tesis. El método aplicado fue una mezcla entre la propuesta del método de investigación y solución de problemas en Ingeniería de Sistemas y el marco de trabajo Action-Research-Evaluation basado en Action-Research propuesto por [11] [12] para llevar a cabo la etapa de investigación del estado del arte y para el estudio de caso [13].

2. Método de Investigación

2.1 Métodos de investigación en ciencias de la computación

A nivel académico existen varios métodos de investigación que ofrecen marcos de trabajo detallados que ayudan a dirigir una investigación en particular, ya sea a través de la experimentación, teoría fundamental o el uso de encuestas y entrevistas, todos buscan dar solución a un planteamiento o problema inicial; pero tanto [11] como [12] expresan la importancia de proponer nuevos enfoques investigativos para las ciencias de la computación o la ingeniería de software que se adapten a las necesidades de estas.

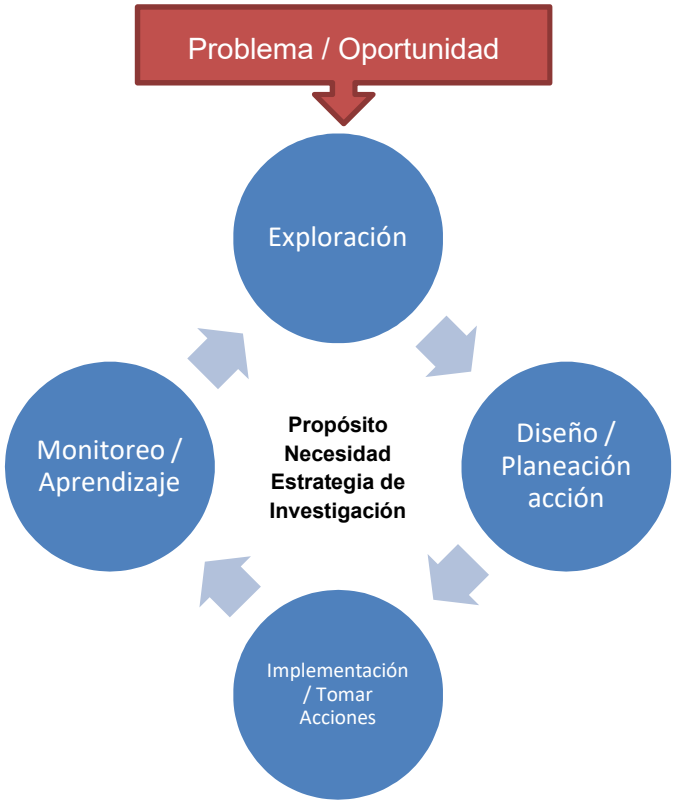
Esto se debe a la naturaleza cambiante de estas disciplinas, las cuales exigen flexibilidad en la investigación al surgimiento de nuevas variables, requisitos, restricciones o estudios que afecten la investigación en curso. Además, los autores hacen especial referencia a la relación que hay entre la industria y la academia, debido a que esta primera impulsa y patrocina la investigación y desarrollo enfocados en sistemas de información y de esta forma surge en países como el Reino Unido el Doctorado en Ingeniería (EngD) como alternativa al PhD y con nuevos retos a nivel de solución de problemas y de investigación.

Con base en lo anterior, se toma como referencia dos propuestas de investigación enfocadas a ciencias de la computación que habilitan el trabajo colaborativo y resultados iterativos, además de haber sido probados en estudios de caso con muy buenos resultados.

2.2 Solución de problemas integrado y método de investigación en ingeniería de sistemas

Acorde a lo propuesto por [12] y que se puede apreciar en la Figura 2-1, se propone un enfoque iterativo-incremental que parte de una oportunidad de estudio o problema dado, enfocado en cuatro etapas y centrado en una estrategia de investigación, esto con el fin de construir una solución a un problema dado.

Figura 2-1: Enfoque para solución de problemas [12].



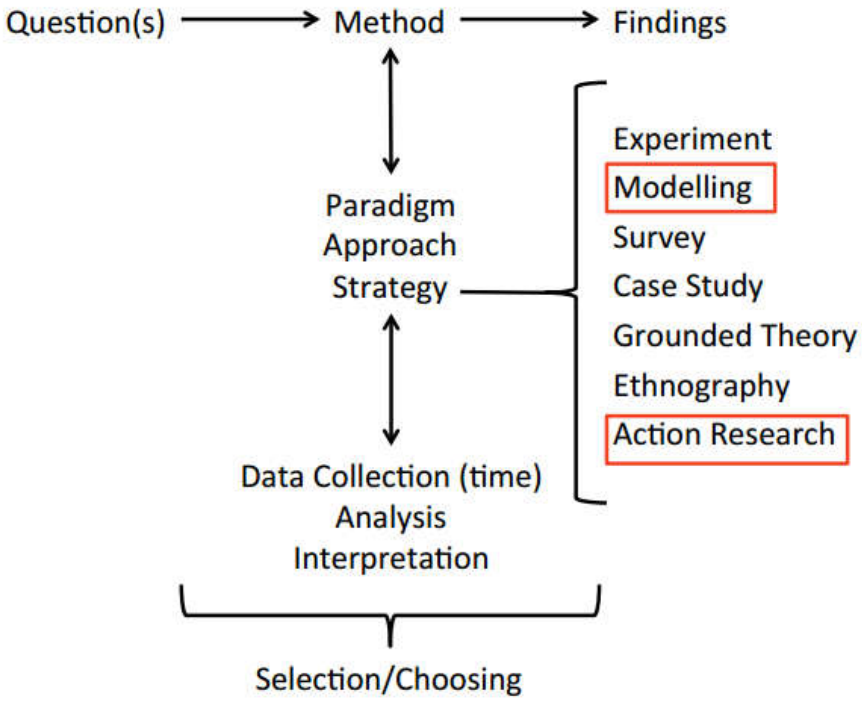
Cada etapa posee un conjunto de procesos recomendados para su ejecución, alguno de estos puede ser:

- **Exploración:** Identificación de stakeholders, exploración de sistemas, remover barreras o impedimentos, interpretación desde distintos puntos de vista del problema u oportunidad, entre otros.
- **Diseño/Planeación acción:** Desarrollo de opciones, modelado, pruebas, estudios de factibilidad, estudios socio-culturales (examinar el cambio), entre otros.

- **Implementación:** Crear o construir prototipos, gestión del cambio, gestión de proyectos, comunicaciones, entre otros.
- **Monitoreo y aprendizaje:** Recolección de datos, plan de métricas, análisis de datos y resultados, entre otros.

Como estrategia de investigación el autor propone un proceso que se puede apreciar en la Figura 2-2, la cual parte de una pregunta de investigación y se realiza una selección del enfoque o método de investigación de acuerdo a la necesidad o naturaleza del problema.

Figura 2-2: Propuesta de método de investigación [12].

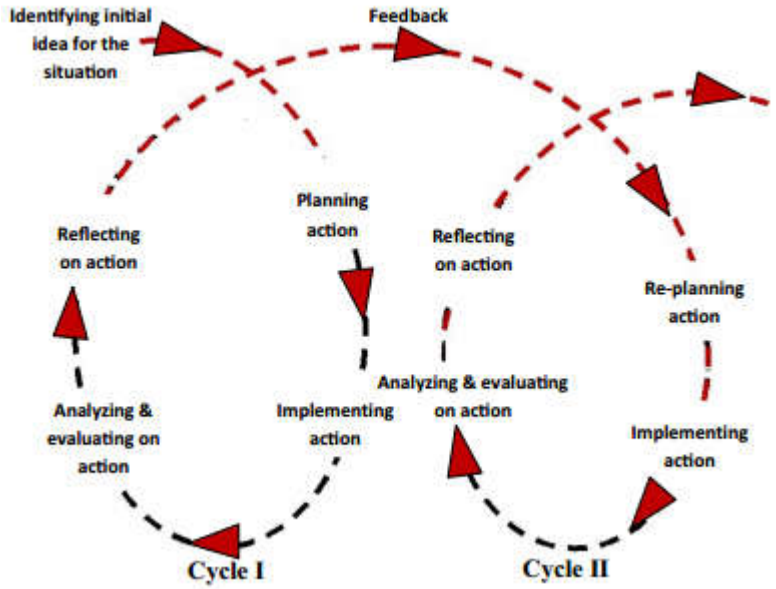


Esta metodología centra su recomendación de aplicación en proyectos de investigación en la ingeniería de software del modelamiento y Action-Research. El primero lo sustenta con base en que los datos son obtenidos de los resultados de las simulaciones que se construyan, lo que conlleva a una investigación altamente válida y el segundo enfoque lo respalda a la capacidad de ejecutar varios ciclos que ayuden a responder las preguntas de investigación que vayan surgiendo a medida que corren las iteraciones.

2.3 Marco de trabajo Action-Research-Evaluation

Por otro lado [11] propone un enfoque basado en Action-Research adaptado a investigaciones en las áreas de los sistemas de información. La propuesta se observa en la Figura 2-3 y está dividida en dos ciclos compuestos por cinco fases cada uno.

Figura 2-3: Propuesta Action-Research-Evaluation Framework [11]



- **Fase I. Identificación de la idea inicial para la solución:** en esta fase se busca determinar las necesidades y requisitos del estudio de caso.
- **Fase II. Planeación de la acción:** esta fase se centra en revisar los objetivos y diseños del proceso, con el fin de identificar participantes que deban ser incluidos, además de formar tareas de grupo y cronogramas de trabajo.
- **Fase III. Implementación de la acción:** como su nombre lo indica, el objetivo de esta fase es implementar el plan trazado en la fase anterior.
- **Fase IV. Analizando y evaluando una acción**: esta fase inicialmente, se centra en analizar las acciones tomadas, para luego evaluar los resultados obtenidos. Se recomienda evaluaciones observativas, formativas y sumativas para una investigación más eficaz.
- **Fase V. Reflejando la acción:** por último, esta fase provee una retroalimentación de la fase de evaluación, además, ayuda a la toma de decisiones para el siguiente ciclo. Los resultados de esta fase son usados en el plan de acción del ciclo dos.

2.4 Estudios de caso

Un estudio de caso, es un enfoque de investigación utilizado para estudiar fenómenos contemporáneos en escenarios reales, enfocado en explorar y describir el comportamiento de un evento en profundidad. Este se presenta en dos tipos:

- Estudios de caso fuertes, enfocados en una filosofía de investigación positivista.
- Estudios de caso suaves, que tienen un enfoque más interpretativo.

Un estudio de caso, es usado particularmente cuando no se tiene claro los límites y es utilizado para cualquier propósito de estudio siendo guiado por tres caminos: i) Exploratorio, ii) descriptivo y iii) explicativo. Un estudio de caso no genera los mismos resultados, esto se debe por lo general a factores sociales, geográficos, culturales y del mismo ambiente laboral, que determinan los resultados finales. Debido a esto, los estudios de caso son criticados por no ofrecer alto valor en sus resultados e imposibilitando la generalización de los mismos y algunas veces clasificados como un método difícil de aplicar rigurosamente [14].

Debido a lo anterior, [13] propone un proceso de estudio de caso conformado por cinco actividades:

- **Diseño y planeación del estudio de caso:** Enfocado en resolver el “Cómo”, “Por qué” y “Qué puede ser aprendido”, ya sea de forma exploratoria o para resolver una hipótesis de investigación.
- **Preparación, procedimientos y protocolos de la recolección de datos:** Enfocado en las habilidades que se requieren para dirigir un estudio de caso y se define el protocolo, procedimientos y reglas a seguir para llevar la investigación.
- **Recolección de evidencia:** Donde se origina la evidencia recolectada, esta puede ser de documentación existente, de archivos de registros como los logs de eventos, entrevistas, observación directa, observación participativa o artefactos físicos o culturales.
- **Análisis de los datos:** Busca examinar, categorizar, tabular, probar o recombinar los datos, tanto cualitativa como cuantitativamente, orientado al propósito del estudio.

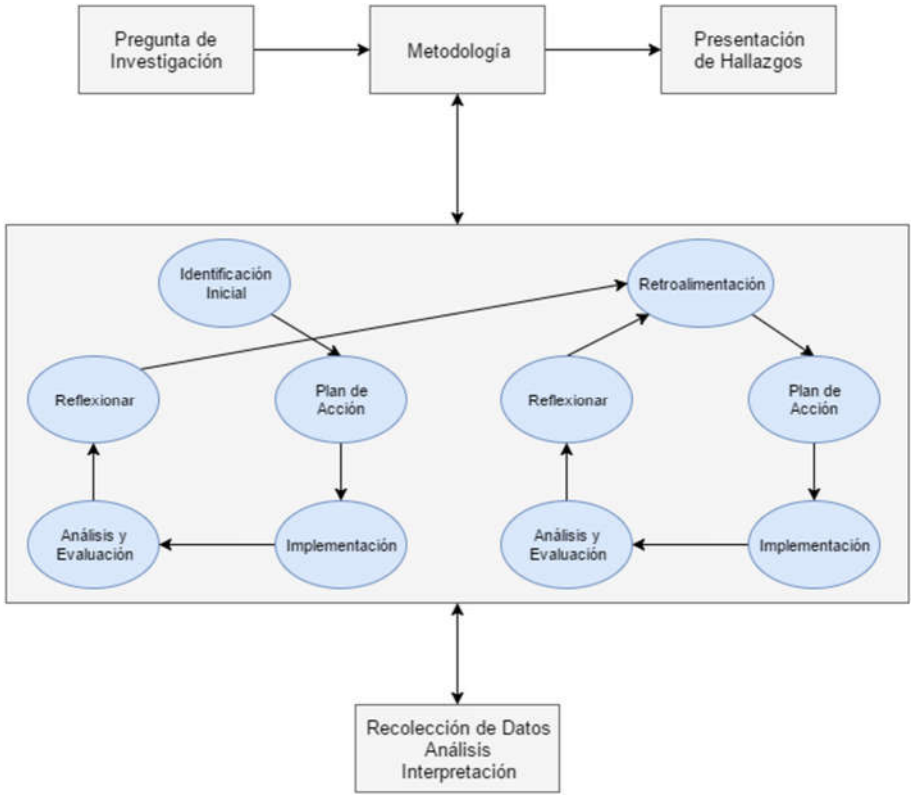
- **Reporte:** Es la presentación de los resultados obtenidos del análisis, identificando la audiencia de destino, diseñando la estructura de visualización y siguiendo ciertos procedimientos definidos con anterioridad.

2.5 Estrategia de investigación en esta tesis

Las metodologías que guían la investigación en esta tesis son: i) solución de problemas Integrado/Método de Investigación en Ingeniería de Sistemas, ii) marco de trabajo Action-Research-Evaluation y iii) método de estudios de caso. Esta sección define la estrategia definida en términos de procesos realizados durante la construcción de la tesis.

La Figura 2-4 detalla los procesos seguidos durante la elaboración de la tesis, centrándose en los enfoques mencionados anteriormente.

Figura 2-4: Enfoque empleado en la investigación



- **Pregunta de investigación:** Este proceso centra su esfuerzo en determinar el objetivo y alcance de la investigación, lo cual determinará el alcance del trabajo.

- **Metodología:** A través del marco de trabajo Action-Research-Evaluation, se busca dar respuesta a la pregunta de investigación pasando por las fases del modelo. Para este trabajo las fases se abordaron de la siguiente forma:
    - **Identificación inicial:** Se tomó la pregunta de investigación y se detalla su objetivo y alcance principal dentro del trabajo.
    - **Plan de acción:** Con el alcance ya identificado, se construyó un plan de trabajo compuesto por una serie de actividades y cronograma que ayuden a dar solución al problema planteado. Para el estado del arte el plan determinó seguir una revisión sistemática de diversas fuentes relacionadas con el problema de investigación. Para el estudio de caso, el plan determinó una serie de actividades que parten desde la conceptualización del modelo hasta la implementación y puesta en marcha.
    - **Implementación:** Se realiza la ejecución del plan trazado, realizando la revisión sistemática de múltiples fuentes de investigación y el plan de acción definido para el estudio de caso.
    - **Análisis y evaluación:** Con los resultados obtenidos, se realizan análisis estadísticos para encontrar correlación entre las distintas fuentes de investigación y empezar a construir una base de hechos y teorías que dan solución a la pregunta de investigación planteada. Este análisis se realiza tanto para la revisión sistemática como para el caso de estudio.
    - **Reflexionar:** Con base en los resultados obtenidos de la fase anterior se analiza la solución planteada, se realiza una retroalimentación del desempeño de las distintas fases, se presenta un plan de mejora y nuevamente se empieza el ciclo con base en las lecciones aprendidas del primer ciclo.
- **Recolección de datos, análisis e interpretación:** Los resultados obtenidos de cada uno de los ciclos y fases, es almacenado para un posterior análisis e interpretación. La revisión sistemática es guardada siguiendo una estructura formal que ayuda a un mejor entendimiento de los datos, para luego ser interpretados en fases más maduras de los ciclos. En el estudio de caso los datos recolectados se realizaron a través de entrevistas informales semi-estructuradas para posteriormente ser analizadas.
- **Presentación de hallazgos:** Se realiza en dos momentos:

- Presentación de la información obtenida, a través de análisis de muestreo, comparación porcentual y análisis de tendencias de los datos recolectados y posteriormente la construcción de unos hechos que dan respuesta a la pregunta de investigación.
- Se propone un enfoque que dé respuesta a los hechos presentados, fortaleciendo los aspectos positivos y mitigando los negativos, encontrados durante la etapa de investigación y análisis de los datos recolectados.

Para el estudio de caso se realiza un análisis de los datos obtenidos y se presentan los hallazgos con los cuales se realiza una retroalimentación tanto al proceso, como al equipo de trabajo del estudio de caso y se obtiene unas conclusiones y recomendaciones de la experiencia.

Los capítulos siguientes muestran los resultados obtenidos al implementar los tres enfoques mencionados al inicio de este capítulo. El capítulo tres se centra en el estado del arte, se presentan tanto el proceso que se realizó, como la presentación de los resultados obtenidos. El capítulo cuatro presenta una propuesta basándose en los resultados y análisis realizado durante la revisión sistemática y el capítulo cinco presenta el estudio de caso que pone en verificación la propuesta presentada.

Capítulo 3. Estado del Arte

"Todos somos muy ignorantes, lo que ocurre es que no todos ignoramos las mismas cosas", (Albert Einstein, Físico teórico alemán, 1879-1955)

En este capítulo se recopila información de conocimiento y apoyo para el contexto general de la propuesta durante su construcción. Asimismo, se presenta un análisis de los trabajos relacionados el cual fue realizado a través de una revisión sistemática de la literatura. La revisión sistemática permitió sintetizar y enfocar mejor el alcance en esta área de conocimiento, asimismo, identificar factores importantes a tener en cuenta para definir la propuesta de investigación presentada en esta tesis de maestría.

3. Marco Teórico y Estado del Arte

3.1 Marco teórico: metodologías y modelos para el desarrollo de software

Para entrar en contexto se ofrece una introducción a Capability Maturity Model Integration (CMMI), la ISO/IEC 29110, SCRUM y eXtreme Programming (XP).

3.1.1 Capability maturiy model integration (CMMI)

CMMI es un modelo que ofrece unos lineamientos para el mejoramiento de procesos en organizaciones de desarrollo de software, además es usado para evaluar el nivel de madures de los procesos de una empresa. CMMI actualmente está orientado a tres áreas de interés [15]:

- Desarrollo de productos y servicios. (CMMI for Development – CMMI-DEV) [16].
- Establecimiento de servicios, Gestión. (CMMI for Services – CMMI-SVC) [17].
- Adquisición de Productos y Servicios. (CMMI for Acquisition – CMMI ACQ) [18].

3.1.2 Estándar ISO/IEC 29110

La ISO/IEC 29110 es un estándar internacional y reporte técnico que ofrece unos perfiles de ciclo de vida y lineamientos para el desarrollo de sistemas de información orientado a muy pequeñas empresas de hasta 25 empleados [19] [20] [21] [22] [23] [24].

La ISO/IEC 29110 es una serie de estándares que buscan el mejoramiento de la calidad en el desarrollo de productos y servicios y no impide el uso metodologías de desarrollo de software tradicional como Cascada o Espiral o ágil como SCRUM y XP.

La ISO/IEC 29110-5-1-2:2011 Management and engineering guide, propone dos procesos principales: Project Management (PM) el cual se enfoca en la iniciación y construcción de una planeación del proyecto, la ejecución de dicho plan, el control y valoración del proyecto y de la clausura al final del mismo. Software Implementation (SP) el cual se enfoca en el análisis de los requerimientos, diseño de la arquitectura global y detallada, la construcción del software, diseño e implementación de los pruebas, integración continua y entregables.

3.1.3 Marco de referencia SCRUM

SCRUM es un macro para gestión de proyectos de desarrollo de software, el cual se basa en la definición de unos roles, eventos, artefactos y reglas para la utilización conjunta de estos elementos [5].

Los roles en SCRUM son tres y están representados por [25]:

- **El Product Owner:** Es el responsable y canal directo del cliente, encargado de dirigir la solución de acuerdo a las necesidades del cliente. Es el rol más importante porque el producto final será un reflejo de lo que esta persona determine y detalle. Esta persona debe asistir a todas las ceremonias con excepción a los Daily meetings y a las reuniones de retrospectiva, a los cuales su presencia puede ser opcional o esporádica.

  Es importante que esta persona sea seleccionada del lado del cliente, debido a que se necesita que tenga un conocimiento amplio de los procesos del negocio, además de habilidades sociales y de comunicación que ayuden a detallar muy bien los requisitos. En algunos enfoques o en charlas sobre agilismo proponen un Product Owner satélite cuando del lado del cliente no proporcional a la persona, es decir, una persona del equipo de desarrollo debe cumplir este rol, lastimosamente esta práctica no genera buenos resultados al final porque esta persona primero debe pasar por una etapa prolongada de conocimiento de la empresa como si se tratase de un empleado más y aun así su entendimiento del cliente no serán los mismos que los de un verdadero empleado de la empresa, además el equipo de trabajo pierde un miembro de trabajo que puede ser determinante en la velocidad del equipo.

De entrada, se debe presentar este requisito al cliente realizando una explicación detallada de los beneficios futuros, también se pueden proponer talleres para demostrar las bondades de este rol en la construcción de la solución.

Un buen acercamiento se puede realizar a través del Spec Writing Game [26]; este es un juego con el cual se puede mostrar las bondades del agilismo, de la participación del cliente en la construcción de los requisitos de la solución y de las iteraciones cortas frente al modelo tradicional. El juego consiste en dividir el grupo participante en pequeños grupos, cada uno tendrá rol de cliente y rol de proveedor y, además, se les entregará un par de dibujos geométricos a los clientes. La idea es que cada dibujo sea transmitido por el cliente en sus palabras al proveedor y cada una de las metodologías, es decir, un dibujo la captura de los requisitos y la posterior solución será realizada de forma tradicional y para el otro dibujo se realiza de forma ágil. Cada ejercicio debe tener un límite de tiempo de 12 minutos y dentro del enfoque ágil se deben presentar las iteraciones para mostrar avances y corregir defectos. Al final se comparan los resultados obtenidos.

Este tipo de juegos ayudan a sensibilizar al cliente sobre la metodología y a convertirlo en parte fundamental del proceso de desarrollo de la solución.

- **El SCRUM master:** Es un cambio de paradigma con respecto al antiguo Líder de Proyecto, el cual era el responsable de la toma de decisiones y de delegar tareas a los miembros de su equipo, también era el encargado de realizar la estimación y planeación del proyecto de acuerdo a su comprensión de las necesidades del cliente. En la mayoría de las veces este personaje es peligroso porque realiza compromisos inalcanzables que pueden terminar inclusive en la quiebra de una compañía.

  Ahora, el rol busca ser un facilitador a todos los miembros del equipo y al proyecto en general, esta persona debe poseer muy buenas habilidades sociales y de comunicación, debido a que muchas veces servirá de apoyo psicológico. Es importante que esta persona tenga una comprensión muy alta del modelo o enfoque que se está aplicando porque debe velar que se cumplan las etapas y que se realicen o se utilicen correctamente las ceremonias y artefactos. Debe ser una persona muy proactiva y estar pendiente del estado de salud del proyecto, para identificar o

predecir obstáculos que puedan entorpecer el desarrollo de la solución y buscar mecanismos apoyado en su equipo y el Product Owner para buscar soluciones rápidas y eficaces. Además, es el responsable de reeducar a los clientes en el enfoque empleado generando empoderamiento de ellos por la futura solución, haciendo que se sientan involucrados permanentemente a todo el proceso.

- **El equipo de desarrollo:** Son las personas encargadas de construir la solución de software, debe ser un grupo de alto nivel técnico y de múltiples disciplinas (cross-functional), es decir, que dentro del equipo de trabajo debe haber ingenieros de requisitos, arquitectos, testers, programadores y expertos en experiencia de usuario.

Entre los eventos o reuniones más importantes encontramos los siguientes [25]:

- **Los Sprints:** que son interacciones de 1 a 4 semanas dependiendo de la complejidad de los requerimientos, que buscan tener un entregable que de valor al final.
- **Los Sprint Planning**: que son reuniones que ocurren al comienzo de cada sprint en el cual se busca definir el alcance, los compromisos y el incremento del sprint.
- **Los Daily SCRUM:** que son reuniones diarias no superiores a 15 minutos en donde el equipo expone lo que se hizo el día anterior, qué se va a realizar y cuáles impedimentos se han presentado que requieran atención.
- **El Sprint Review:** es una reunión al final del sprint con el Product Owner en la cual se realiza una presentación del trabajo realizado, es decir, del entregable que ofrezca valor y de aquello que no se pudo completar.
- **El Sprint Retrospective:** es una reunión al final del sprint que busca identificar el estado del proceso respondiendo tres preguntas: i) ¿Qué se hizo bien y se debe seguir haciendo?, ii) ¿Qué se debe mejorar? Y iii) ¿Qué se hizo mal y no se debe volver a hacer?

Los artefactos están compuestos por [25]:

- **El Product Backlog:** el cual es una lista ordenada por prioridad e impacto de todos los requerimientos del proyecto, llamados historias de usuario y la cual es mantenida por el Product Owner en colaboración con el equipo de desarrollo.

- **El Sprint Backlog:** es una lista con las historias de usuario que se van a trabajar durante el Sprint.
- **El SCRUM Task Board:** es un tablero dividido en secciones relacionadas con las actividades que se van realizando con las historias de usuario; cada equipo SCRUM personaliza el tablero de acuerdo a la forma de trabajo.
- **El Sprint Burn-Down Chart:** es una gráfica pública que muestra el avance del sprint, esta debe ser actualizada cada día.
- **El Release Burn-Down chart:** es una gráfica pública que muestra el avance del proyecto, esta debe ser actualizada al final de cada sprint.

3.1.4 Metodología eXtreme Programming (XP)

XP es una metodología de desarrollo de software que busca mejorar la calidad del software y responder rápidamente a los cambios requeridos por el cliente. Esto lo hace proponiendo entregas frecuentes en cortos ciclos de desarrollo con la intención de obtener retroalimentación oportuna frente a buenas necesidades del negocio [4]. Incorpora el concepto de programación en pares en donde dos personas trabajan juntas en una misma máquina, en donde uno toma el rol de driver quien es el que escribe el código y el observer o navigator quien está revisando el código en busca de errores o dando alternativas de codificación. La pareja frecuentemente debe intercambiar de roles. XP, además propone que las parejas también varíen durante el transcurso del proyecto con el fin de incentivar que el conocimiento sea compartido con todos los miembros de equipo de trabajo.

3.2 Estado del arte

Para el análisis de los trabajos relacionados, se desarrolló una revisión sistemática de la literatura o estado del arte la cual ha sido publicada en [10] [8]. La revisión sistemática se llevó a cabo sobre los trabajos relacionados a las metodologías ágiles y otros modelos en micro, pequeñas y medianas empresas (MiPyMEs), la cual siguió la plantilla de protocolo utilizada en [27] [28]. El protocolo para la revisión consta de cinco partes generales: (i) formulación de la pregunta, (ii) selección de las fuentes, (iii) selección de los estudios, (iv) extracción de la información y (v) resumen de los resultados.

3.2.1 Formulación de la pregunta

La pregunta de investigación que se planteó para desarrollar esta revisión fue: ¿Qué procesos alrededor de las metodologías ágiles para micro, pequeñas y medianas empresas se han desarrollado o abordado para el desarrollo de software? A continuación, se presenta el listado de términos utilizados para diseñar la pregunta de investigación:

Agile Project Management, Agile Software Development, Agile Methods, SCRUM, Extreme Programming, XP, Feature Driven Development, FDD, Pair Programming, ISO 209110, ISO/IEC 29110, SMES, SMBS, Small and Medium Enterprises, Small and Medium-Sized Businesses, PYMES, Small Sized Companies, Startup, Startup Company, Small Organizations, Small Enterprise, Small Entity, Medium Sized Company, Medium Organizations, Medium Enterprise, Medium Entity, Small Projects, Small Teams, Software, SPI, Software Process Improvement.

La población que se observó fueron las publicaciones relacionadas con la implementación de metodologías ágiles en MiPyMEs, existentes en las fuentes seleccionadas para la revisión.

3.2.2 Selección de las fuentes

Con la combinación del listado de palabras identificadas se utilizaron conectores lógicos “AND”, “OR” y “NOT” y se obtuvo una cadena general básica de búsqueda (Tabla 3-1).

Tabla 3-1: Cadena de búsqueda básica

(("Agile Project Manament" OR "Agile Software Development" OR "Agile methods" OR SCRUM OR "extreme programming" OR XP OR "feature driven development" OR FDD OR "pair programming" OR "ISO 29110" OR "ISO/IEC 29110") AND (smes OR smbs OR "small and medium enterprises" OR "small and medium-sized businesses" OR pymes OR "Small Sized Companies" OR startup OR "Startup Company" OR "small organizations" OR "small enterprise" OR "small entity" OR "Medium Sized Companies" OR "medium organizations" OR "medium Enterprise" OR "medium entity" OR "Small Projects" OR "small teams") AND software AND NOT SPI AND NOT “software process improvement”)

La lista de fuentes empleada para llevar a cabo la revisión sistemática fue:

- ScienceDirect en el tema de Computer Science,
- Wiley Online Library en el tema de Computer Science,

- ACM Digital Library,
- Scopus,
- El estándar ISO/IEC 29110,
- Los sitios oficiales del Manifiesto Ágil, SCRUM, eXtreme Programming (XP) y
- Otros artículos relacionados con el tema se revisaron como literatura gris.

Al momento de utilizar la cadena y ejecutar la búsqueda, la cadena tuvo que ser adaptada a las características de cada uno de los motores de búsqueda de las fuentes escogidas.

3.2.3 Selección de los estudios y extracción de la información

El procedimiento de selección de estudios para esta revisión está basado en el modelo de desarrollo software Iterativo Incremental. El proceso iterativo se realizó en la búsqueda, extracción y visualización de los resultados en cada una de las fuentes de búsqueda seleccionadas. Por su parte, el proceso incremental se desarrolla porque el procedimiento de selección de los estudios se ejecuta sucesiva o iterativamente en cada una de las fuentes, de manera que el informe de la revisión fue creciendo y evolucionando cada vez más hasta completar y obtener el reporte final de la revisión el cual ha sido resumido en este artículo.

El criterio de inclusión de los estudios primarios obtenidos, se basó en el análisis del título, resumen y palabras claves. Algunos artículos no eran lo suficientemente claros en estos atributos, así que se decidió leer la introducción para tener una idea clara de la investigación. De esta forma se pudo determinar si los artículos tenían algún tipo de relación con la implementación de las metodologías ágiles en la industria y el tipo de enfoque con el cual abordan el tema.

Como criterio de exclusión, en algunos casos y para tener un mejor entendimiento del estudio, se realizó una lectura del resto del documento que ayudara a comprender los lineamientos de la investigación. Con base en esto, se pudo pasar a una lectura y análisis más detallada de los estudios, definir la relación con los objetivos de la investigación en curso y la revisión sistemática y seleccionarlo como estudio primario.

La información de las publicaciones que fueron consideradas como primarias se almacenó en una plantilla (Tabla 3-2), donde se observa un ejemplo de cómo se utilizó la plantilla con el análisis de uno de los estudios seleccionados.

Tabla 3-2: Ejemplo de la plantilla utilizada para almacenar la información

Título	Understanding post-adoptive agile usage: An exploratory cross-case analysis.
Publicación	The Journal of Systems and Software 85 (2012) 1255–1268.
Año	2012.
Autores	Mali Senapathi, Ananth Srinivasan.
Resumen	Los estudios que se encuentras en la literatura están enfocados principalmente en estudiar o proponer métodos o enfoques para la implementación de una metodología ágil en la empresa para hacer frente a los proyectos de ingeniería de software, debido a esto hay una falta en identificar qué factores se presentan después de la post adopción, los retos que deben enfrentar las organizaciones para mantener el uso de estos métodos y la efectividad de los mismos. Este estudio busca dar un entendimiento de todos estos aspectos que se presentan en el día a día en la utilización de algún método ágil y la aceptación del mismo dentro de la organización.
Descripción	
Metodología ágil propuesta	Un framework para lograr un entendimiento de los factores que ayudan a una mejor adaptación de las metodologías ágiles.
Nivel de abstracción para el análisis	El estudio primordialmente ha fusionado otros métodos y propuestas para lograr entender el nivel de adaptación de una metodología ágil dentro de la organización y su efecto en un framework, que es aplicado a dos empresas que llevan tiempo empleando estos enfoques ágiles. Con esto se logra extraer unos factores claves de éxito en la adopción y expansión de uso del método en la organización.
Tipo de evaluación empírica	Estudio empírico sobre dos organizaciones que han implementado prácticas ágiles a sus procesos de desarrollo de software, utilizando entrevistas semi-estructuradas a miembros claves del equipo de trabajo.
Marcos analizados	SCRUM con Kanban, SCRUM con Waterfall/RUP
Relevancia	Alto
Aspectos a destacar	

- En el proceso de adopción de nuevas tecnologías o enfoques, se pueden encontrar propuestas como:
  - Teoría de la difusión de la innovación (DOI) que está basada en 5 fases de proceso secuencial:
    - Ventajas: Ofrece ventajas especificas con tu predecesor.
    - Compatibilidad: Es compatible con valores, practicas, procesos existentes.
    - Complejidad: Fácil de usar.
    - Testeabilidad: puede ser testeable
    - Observabilidad: Es observable.
  - Modelo de Implementación de Sistemas de Información que ofrece 6 fases de modelo de implementación y buscan detallar la profundidad de penetración de la innovación en la organización.

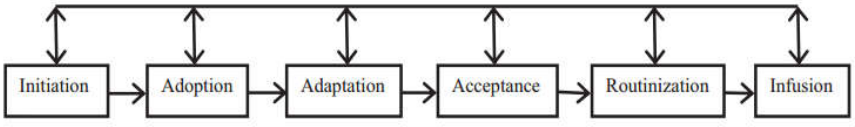


Fig. 1. IS implementation model.
Adapted from Cooper and Zmud (1990) and Kwon and Zmud (1987).

- Iniciación: Se identifica una necesidad de cambio, una coincidencia se identifica entre una innovación y la aplicación en la organización.
- Adopción: La decisión es tomada de adoptar una innovación.
- Adaptación: La adaptación trae unas necesidades contextuales.

  - Reutilización: Incremento en la extensión e intensidad de uso.
  - Infusión: Aumento en el uso de una forma más amplia y resultados de manera integrada en la mayor eficacia de los sistemas de desarrollos.
- Proponen un modelo integrando varios modelos obtenidos de la investigación para evaluar e identificar los factores que afectan la adopción y uso de una metodología ágil:

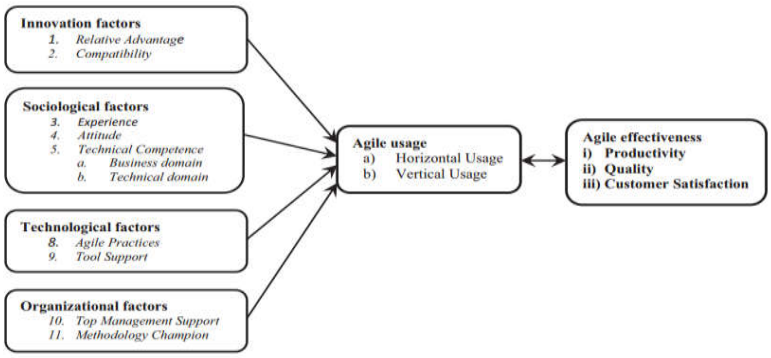


Fig. 3. The research model–agile usage model.

- Factores de Innovación: Una organización se puede mover satisfactoriamente a una fase de post adaptativa solo cuando la innovación ofrece un mejoramiento específico consistente en comparación a su predecesor.
- Factores Sociales: Relacionado con el nivel de conocimiento técnico de los equipos involucrados en la innovación, esto contribuye con el incremento de la productividad. Un alto nivel de habilidad puedo no ser asociado a la curva de aprendizaje de un domino no familiar o habilidad en un lenguaje.
- Factores Tecnológicos: Incluye prácticas de desarrollo ágil y herramientas de soporte, como herramientas de testing automáticos. Registrar las prácticas ágiles pueden proveer ideas de que las prácticas híbridas pueden conducir a más resultados efectivos.
- Factores Organizacionales: Incluye a TMS el cual se refiere a la puesta en marcha, soporte continuo y estimulo en la alta administración ejecutiva en la adopción e implementación de la innovación y MC se refiere a que el éxito en la adopción de metodologías ágiles implica cambios en la cultura organizacional, en los procesos y en la forma de hacer las cosas y juega un papel importante en fomentar y facilitar el uso continuo de las prácticas ágiles.
- Uso de ágil: Es una medida clave en el éxito de implementación de una metodología de desarrollo de software y hace uso de diferentes terminologías como aceptación, uso/reutilización, infusión e incorporación.
  - El uso horizontal es el uso de la innovación a través de la organización como el porcentaje de proyectos y número de equipos.
  - El uso vertical es la máxima intensidad de uso, es decir la profundidad de uso de una práctica ágil en específico.
- Efectividad del uso de ágil: Busca examinar si hay un mejoramiento específico sobre todo el proceso de desarrollo de software como resultado de emplear prácticas ágiles y como esta es percibida por la organización. Esto a través de tres criterios: Mejoramiento de la productividad, mejoramiento de la calidad y mejoramiento de la satisfacción del cliente.
- Del estudio se puede concluir que los factores que ayudan a la adopción de las metodologías ágiles pueden ser:

Table 1
Summary of findings.

	BBCW	Stats NZ
Methodology	Kanban	Scrum
Predecessor	Scrum	Waterfall/RUP
Innovation factors		
Relative advantage	Limiting work in progress Reducing batch size Granularity of visualization	Better time management in requirements gathering Improved delivery schedules Higher team morale
Compatibility	Not mentioned	Resistance from business analysts Incompatibility with resourcing model
Sociological factors		
Experience	High	Moderate
Attitude	Positive attitude, willingness to learn, and change, highly motivated	Mixed attitudes toward change
Technical competence	Technical – high Business domain – moderate	Technical – moderate Business domain – high
Technological factors		
Agile practices	Deeper use of all practices	Adaptations of Scrum practices
Tool support	Cucumber for automated testing, TeamCity for continuous integration	Rally project management tool, in-house release management systems, and Visual Studio Team Foundation Server
Organizational factors		
Top management support and methodology champion	Played a critical role in influencing agile usage	Played a critical role in influencing agile usage
Agile usage	Significant increase in both horizontal (80% of projects using Kanban – spread referred to as *Kanban flu*) and vertical usage (all core properties of Kanban)	Significant increase in both horizontal (almost all projects using Scrum) and vertical usage (derived adaptations of Scrum properties)
Agile usage effectiveness	Specific improvements recorded in improved quality and productivity	Improvements in both quality and productivity Customer engagement and satisfaction seen as having a direct impact on the success of the project

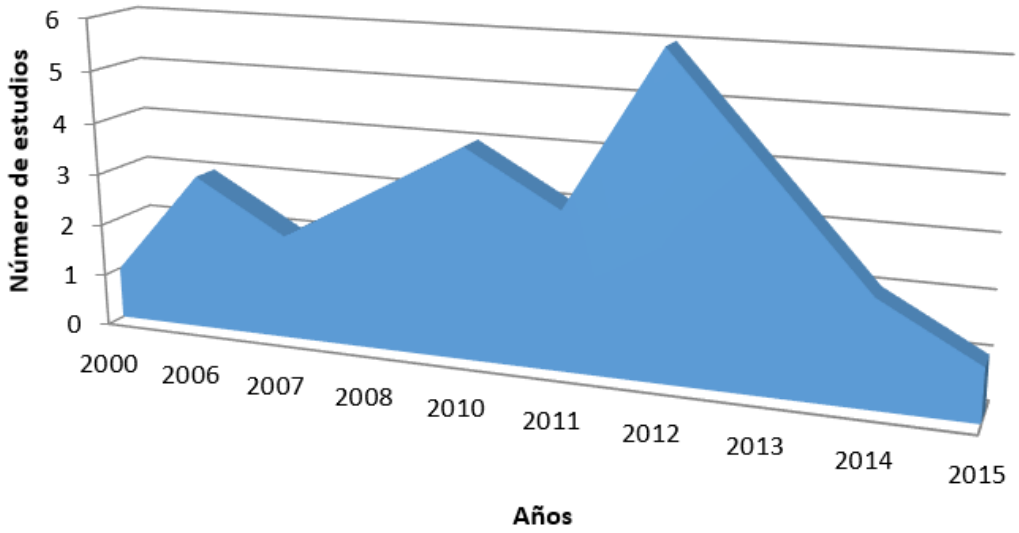
3.3 Resultados y discusión

Sobre los estudios seleccionados a los cuales se les aplicó la extracción de información, se realizó un análisis estadístico desde diferentes perspectivas, entre ellas: (i) tendencias de las publicaciones, (ii) países de estudio, (iii) tipo de Industria, para conocer qué sectores o clientes objetivo están interesados en dichos temas, (iv) tipo de empresa con relación al tamaño de la misma y (v) tamaño de los equipos de trabajo.

3.3.1 Tendencia de las publicaciones

Después de aplicar el procedimiento para la obtención de los estudios primarios, se encontraron 51 estudios, de los cuales 29 se clasificaron como estudios primarios. En la Figura 3-1 se puede visualizar la tendencia de publicaciones de los estudios desde el año 2000 al 2015. Como se puede apreciar, desde el 2012 hubo un creciente interés por parte de la comunidad académica y profesionales del desarrollo de software en el estudio de las metodologías ágiles, a partir del 2013 se observa un descenso en la cantidad de investigaciones, para lo cual se puede asumir que parte de los estudios realizados en el 2014 no han sido todavía agregados e indexados en las bases de datos de las fuentes consultadas. Por su parte, los estudios relevantes para el 2015 son aún muy pocos, esto debido a que apenas está avanzando la primera mitad de este periodo.

Figura 3-1: Tendencia de publicaciones



Con el análisis realizado sobre los estudios primarios, fue posible clasificarlos en tres categorías como se muestra en la Tabla 3-3 y donde es posible observar que: el 21.89% de los estudios se pueden clasificar como estudios que se enfocan en analizar el estado de arte sobre otros estudios propuestos sobre metodologías ágiles (categoría 1), el 25% son estudios que se enfocan en proponer una nueva metodología ágil (categoría 2) y el 53.1% son estudios que se enfocan en analizar estudios de caso desarrollados en la industria de software.

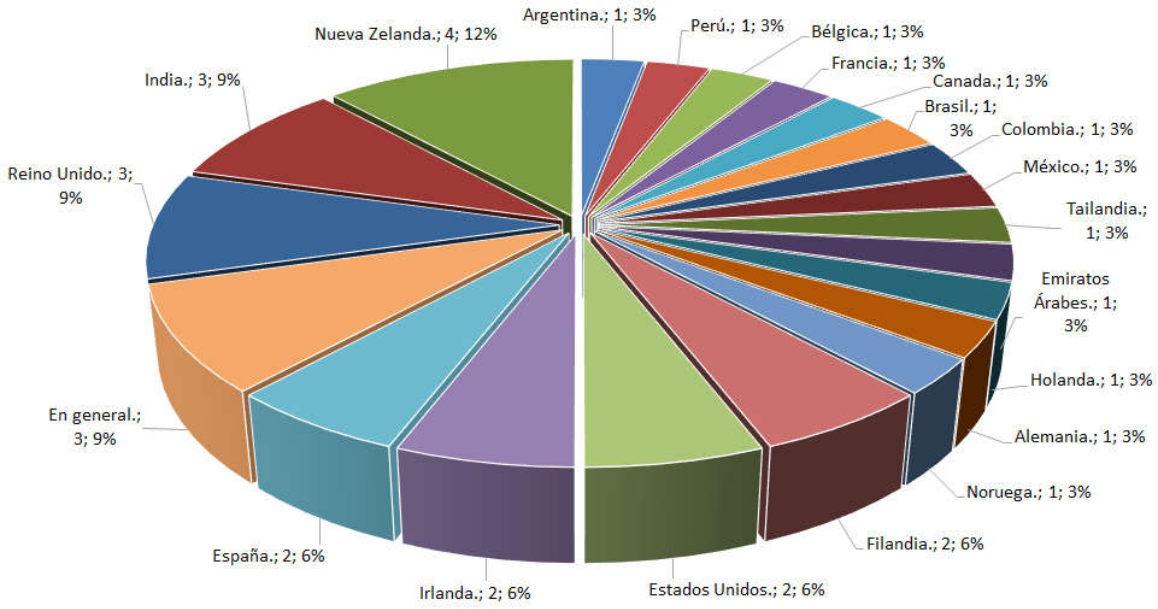
Tabla 3-3: Clasificación por tipo de estudio

Cat.	Tipo de Estudio	Estudios	No. de Estudios	%
1	Estudios que se enfocaron en analizar el estado de arte (otros estudios) sobre metodologías ágiles.	[29] [30] [31] [32] [33] [34] [35]	7	21.9 %
2	Estudios que se enfocaron en proponer una nueva metodología ágil.	[31] [36] [37] [38] [39] [40] [41] [42] [43]	8	25.0 %
3	Estudios que se enfocaron en analizar estudios de caso desarrollados en la industria de software.	[29] [31] [35] [37] [42] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54]	17	53.1 %

3.3.2 Países de estudio

En la Figura 3-2 se pueden observar los países donde se han llevado a cabo las investigaciones de los estudios primarios seleccionados. La mayor contribución de estudios se centra en los países europeos con un porcentaje del 54%, esto, debido a que en estos países la implementación de metodologías ágiles lleva mucho más tiempo. Con relación a los países asiáticos, estos alcanzan un 15%, donde: India, Emiratos Árabes y Tailandia están a la cabeza. Los países latinoamericanos como: Argentina, Brasil, Colombia, México y Perú conforman un 15%, este porcentaje es un poco bajo en comparación a otros continentes, sin embargo, es necesario tener en cuenta que la implementación de los enfoques ágiles es muy joven al igual que su integración con otros enfoques como CMMI, estándares ISO y otros modelos.

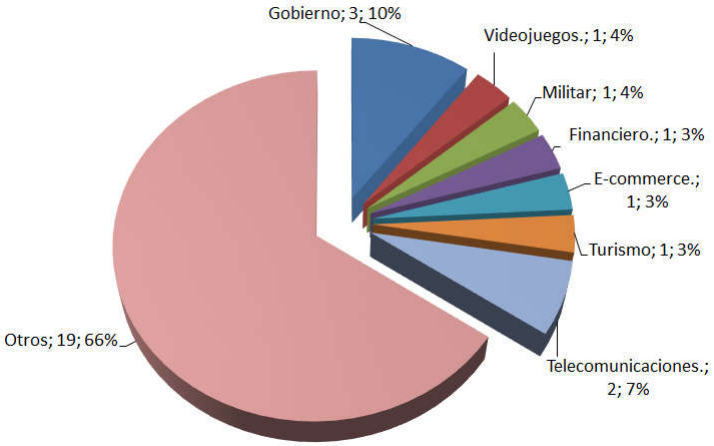
Figura 3-2: Distribución de los estudios por país



3.3.3 Clasificación por tipo de industria

La Figura 3-3 muestra los sectores en los cuales las empresas de estudio de casos identificados se han enfocado, entre ellos: al sector financiero, telecomunicaciones, educación, turismo, gobierno, sector militar, entre otros. En la clasificación “entre otros” se han agrupado los estudios donde las empresas de desarrollo de software realizan trabajos a la medida sin importar el sector. Asimismo, en este grupo se clasificaron los estudios donde no se detalla el tipo de proyecto o sector. Con relación a la distribución de acuerdo a los sectores identificados, es posible observar que el sector gobierno (10%) posee un interés en el aprendizaje de los enfoques ágiles y su implementación en los procesos y proyectos de software. Otro sector interesado es el de las telecomunicaciones (7%); esto debido al fuerte crecimiento de este tipo de organizaciones a nivel mundial y la necesidad de apalancar las nuevas tecnologías emergentes alrededor de la movilidad y otras áreas de Internet, así como el Internet de las cosas (IoT). Otros sectores como el de los videojuegos (4%), el sector militar (4%), financiero (3%), comercio electrónico o e-commerce (3%) y turismo (3%) tienen una menor representación.

Figura 3-3: Clasificación por tipo de industria



Al realizar un análisis más detallado por país, ha sido posible identificar que Noruega enfoca sus estudios en el sector financiero, e-commerce, turismo y telecomunicaciones. Estos sectores en particular han sido los de mayor crecimiento y por consiguiente los que demandan y requieren soluciones de software. Otros países como, por ejemplo, Tailandia (3%), Emiratos Árabes (3%) y Nueva Zelanda (12%) se orientan más al sector del e-commerce (3%). A partir de la Figura 3-3, es posible observar que el mayor porcentaje de los estudios realizados se centra en la industria del desarrollo de software a la medida en vez de un sector en particular, la principal razón para entender y aplicar los métodos ágiles responde a la necesidad de afrontar cualquier tipo de proyecto independientemente del sector en el que las empresas están trabajando. Además, el incremento de startups con base tecnológica y en especial el software ha incrementado en todos los países hace algunos años.

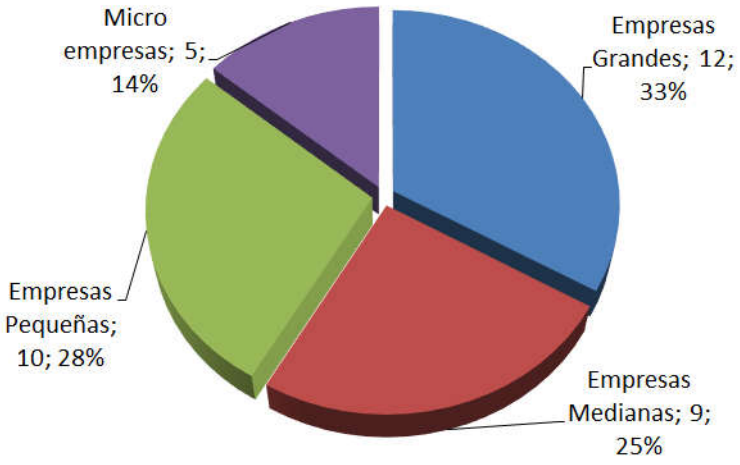
3.3.4 Clasificación por tamaño de empresa

De acuerdo al Ministerio de Industria y Comercio en Colombia, las empresas se pueden clasificar de la siguiente forma de acuerdo al número de sus empleados: micro empresas no mayor a diez (10 empleados), pequeña empresa (entre 11 y 50 empleados), mediana empresa (entre 51 y 200 empleados) y grandes empresas (mayor a 201 empleados) [1].

Como se muestra en la Figura 3-4, los esfuerzos de aprendizaje, implementación y adopción de las distintas metodologías ágiles, está distribuido de forma casi equitativa entre los distintos tipos de organizaciones. Sin embargo, es importante resaltar que las MiPyMEs representan el 67% de los estudios con una diferencia, en total del 34% con

relación a las grandes empresas. En este sentido, es posible decir que el grupo empresarial más grande (por cantidad de empresas) y enfocado en la implementación de prácticas y principios ágiles ha sido el de las MiPyMEs. Asimismo, hemos podido identificar que las MiPyMEs han decidido implementar prácticas ágiles dado que se adaptan mejor a sus características empresariales relacionadas a la cantidad de empleados, recursos económicos y tipo de proyectos con requerimientos cambiantes y/o desarrollos a la medida. Por otra parte, las grandes empresas buscan implementar un enfoque ágil para alivianar sus procesos y disminuir la complejidad y costos asociados a la implementación de enfoques con mayor formalismo, por ejemplo: una empresa certificada en CMMI que ha decidido implementar SCRUM.

Figura 3-4: Clasificación por tamaño de la empresa



Al realizar un análisis más detallado de los modelos más utilizados, ha sido posible observar que las MiPyMEs centran su interés de estudio en el estándar ISO/IEC 29110 y su integración con otros modelos ágiles así como SCRUM y XP, esto debido al interés en implementar la ISO/IEC 29110 con un enfoque ágil y lograr la certificación en un modelo que se adapte mejor a sus características, entre las que se pueden destacar principalmente: el costo asociado a la implementación de modelos como ISO/IEC 29110 es menor que con CMMI, el mantenimiento y complejidad de un modelo como CMMI es mucho mayor y requiere de un equipo demasiado grande para lograrlo. De forma inesperada, ha sido posible también observar un escenario bastante interesante con las grandes empresas, en donde el objetivo ha estado enfocado a agilizar sus procesos con la adaptación de sus prácticas, rediseñando por completo los procesos existentes, pero

conservando lo necesario para mantener la certificación. En este sentido, se han desarrollado enfoques híbridos que permiten soportar la implementación de un modelo como CMMI a partir de un modelo de gestión ágil como SCRUM.

3.3.5 Clasificación por tamaño de los equipos

En la Figura 3-5 el 55% de los trabajos analizados tienen en común el haber conformado equipos de trabajo medianos; característica alineada con los principios ágiles y que tiene un fuerte impacto en el éxito de los proyectos ágiles [45]. El problema se presenta con los equipos de gran tamaño, a los cuales la implementación y adopción de una metodología ágil se convierte en un gran desafío y lleno de obstáculos que impactan directamente en: el manejo de los equipos, cumplimiento de los sprints o reuniones de entrega y de lograr un entendimiento global del proyecto por parte del equipo, por ejemplo, para estos equipos los artefactos físicos como el muro de tareas (task board) y las historias de usuario, se convierten en una tarea compleja de mantener, además impide que el equipo logre tener una visión general del proyecto, sobre todo cuando estos equipos están dispersos geográficamente. Aunque se han desarrollado soluciones electrónicas de estos artefactos, al final no ofrecen las mismas ventajas en términos de cooperación y coordinación que en las reuniones físicas [46]. Otro ejemplo es que los equipos grandes están acostumbrados a trabajar con métodos más tradicionales como cascada o espiral y son muy reacios a cambiar su esquema de trabajo, también se ve cómo los equipos grandes crean falsas expectativas en su primer intento ágil, piensan que el nuevo enfoque puede solucionar todos los obstáculos y, además, encaran un proyecto grande o de fuerte impacto sin estar realmente preparados [38]. Otro de los escenarios analizados ha sido el de los video juegos, los cuales por lo general tienen equipos de trabajo muy grandes y los estándares y modelos existentes proponen procesos que no son naturales para ellos y no les ofrece la flexibilidad y adaptabilidad que requieren [55].

3.3.6 Modelos utilizados

La Figura 3-6 muestra los estándares, modelos y metodologías utilizadas en los estudios seleccionados. Como se puede observar, de los 29 estudios 7 utilizaron ISO/IEC 29110, 8 CMMI, 11 SCRUM y 9 XP, asimismo, algunos estudios han utilizado enfoques tradicionales como el modelo en cascada y espiral [52]. Por otro lado, el 34.5% de los estudios analizados proponen nuevas propuestas que permiten integrar metodologías

tradicionales, estándares y modelos de facto con enfoques ágiles, por ejemplo, CMMI e ISO/IEC 29110. Asimismo, se puede observar una tendencia de los trabajos realizados con relación al uso de modelos como SCRUM y XP con el 23.9% y 19.6% respectivamente, en este sentido, de acuerdo al análisis, éstos son los modelos ágiles más populares y de mayor implementación en la industria de software hasta el momento.

Figura 3-5: Clasificación por tamaño de los equipos

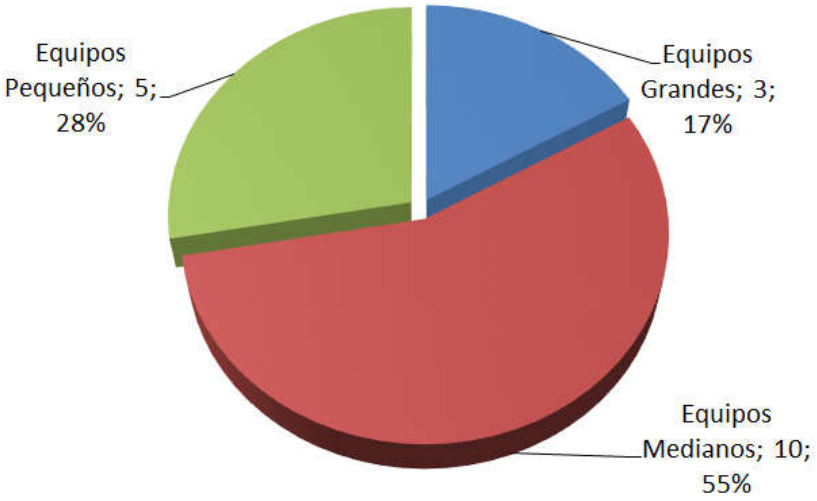
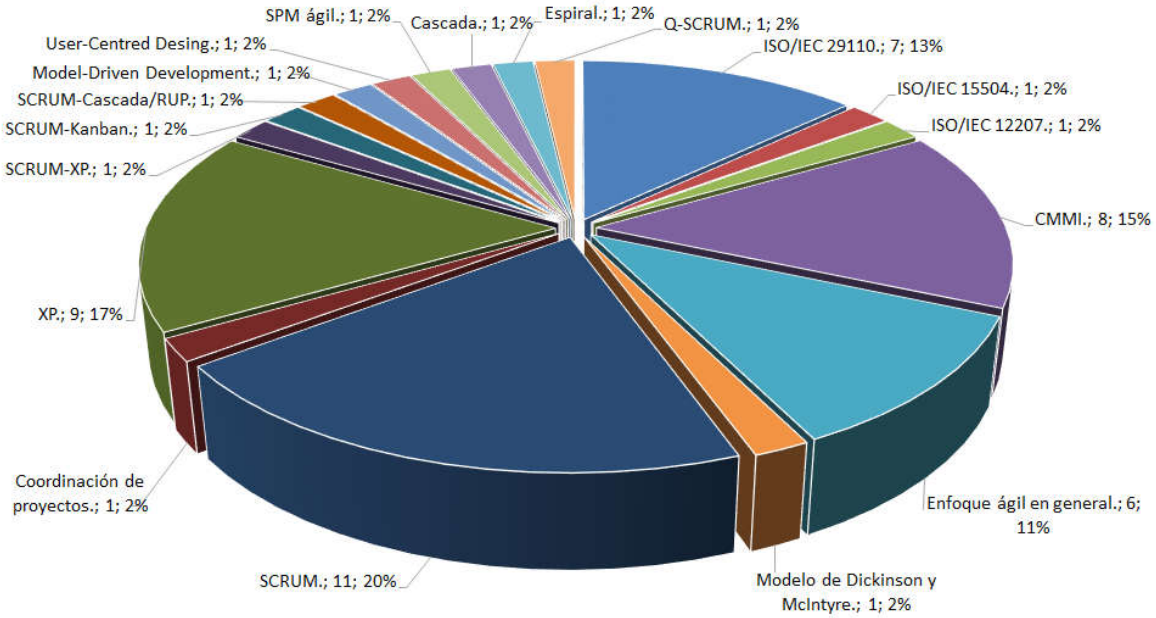


Figura 3-6: Modelos utilizados



3.3.7 Software safety

Durante la investigación realizada, se encontró con un tipo especial de software en el cual los métodos tradicionales y ágiles no son propicios para su ejecución y control, y este es el caso del software safety o software seguro [56].

Un software seguro o software para sistemas seguros, es aquel que requiere altos niveles de calidad y seguridad, en el cual la falla de un sistema de estos puede ocasionar la pérdida de grandes cantidades de dinero, el cierre definitivo de una compañía o atentar contra la vida de seres humanos. Este tipo de software hace uso de métodos y herramientas específicos y muy especializados para la definición de requisitos y gestión del proyecto a lo largo de su ciclo de vida.

Además, Se identifican 6 áreas de trabajo claves para el desarrollo de software para sistemas seguros:

1. **Análisis de peligro:** Esta es un área importante que busca determinar todos los posibles daños y niveles de daño en un determinado sistema crítico seguro.
2. **Especificación y análisis de software seguro:** Basada en métodos formales para la gestión de requisitos seguros, lo cual ayuda en el mejoramiento de la calidad del producto final y ofrece los lineamientos y restricciones de los casos de pruebas.
3. **Diseñando para la seguridad:** Propone mecanismos para la detección de peligros, las pruebas automáticas, el manejo de excepciones y reconfiguración, con el fin de mitigar las fallas en el sistema.
4. **Pruebas:** Una óptima construcción de pruebas pueden ayudar a demostrar que una solución funciona adecuadamente y a prevenir o anticipar situaciones anómalas. Además, se centra en eliminar supuestos acerca del ambiente, supuestos acerca de los usuarios y supuestos acerca de la operación.
5. **Certificación y estándares:** La certificación en sistemas de información cada día toma más importancia, pero la falta de estándares de la industria en muchas disciplinas de la ciencia que hacen uso de software seguro es determinante. Se debe empezar a construir estándares que ayuden a estos sistemas obtener una certificación que los avalen como software de sistemas críticos seguros.

6. **Recursos:** Existe una gran cantidad de referencias bibliográficas que describen técnicas usadas en ingeniería de software seguro. Se debe empezar a realizar una investigación más detallada de estas técnicas y empezar a indagar en otras áreas de las ciencias de la computación que ayuden a la construcción de este tipo de soluciones.

3.3.8 Factores de éxito extraídos de los estudios analizados

Con el análisis de los estudios primarios, ha sido posible identificar y extraer algunos factores a tener en cuenta para soportar la implementación exitosa de una metodología ágil (Tabla 3-4).

Tabla 3-4: Factores de éxito

No.	Nombre del factor	Descripción
1	**Tener equipos convencidos e involucrados.**	El equipo debe estar convencido que sus proyectos necesitan un cambio y que adoptar e integrar otro enfoque será determinante para su realización. Todo el equipo debe involucrarse.
2	**Implementar e institucionalizar un proyecto de mejora.**	La implementación e integración de nuevas prácticas debe gestionarse como un proyecto de mejora de procesos el cual debe estar apoyado por la alta gerencia, tener asignados recursos y gestionado por personal competente y experto.
3	**Identificar el valor agregado.**	El retorno de la inversión debe expresarse desde el valor agregado que la empresa desea alcanzar. El valor agregado puede definirse desde el punto de vista de la ausencia de un conjunto de atributos que la organización quiere lograr, por ejemplo: mayor formalismo y control, documentación, agilidad, retroalimentación permanente con el cliente, aumento de la satisfacción, entre otros. La empresa debe identificar cual es el valor agregado que espera tener en su proceso.
4	**Obtener el apoyo de la alta gerencia.**	El apoyo de la alta gerencia para implementar e integrar nuevas prácticas a las existentes es fundamental. La asignación adecuada de recursos, capacitaciones, personal, tiempo y esfuerzo es de suma importancia.
5	**Establecer objetivos alcanzables y medibles.**	Establecer objetivos alcanzables en el tiempo y que permitan medir el progreso y el retorno de la inversión a corto, mediano y largo plazo. La implementación de un nuevo enfoque o modelo debería impactar positivamente a la organización, nunca entorpecer y sobrecargar sus procesos existentes.
6	**Conseguir resultados rápidos y continuos.**	Es importante que, si se integran nuevas prácticas a un proceso de desarrollo, éstas deben lograr conseguir resultados rápidos que permitan mantener la motivación continua y permanente a medida que se implementan cambios sobre proyectos en ejecución. Este factor debe permitir mostrar que el camino hacia el retorno de la inversión identificado se está logrando.
7	**Definir una estrategia para lograr la confianza y la colaboración.**	Una estrategia adicional al proyecto de mejora debe ser definida, en especial porque el proyecto de mejora por sí mismo no establece los métodos o prácticas para que los participantes estén altamente motivados, confiados y

		colaborativos. Adicional, los mecanismos de comunicación deben soportar las tareas relacionadas a la implementación de nuevas prácticas, documentación de lecciones aprendidas, problemas, soluciones, entre otros.
8	**Medir, supervisar y monitorear el valor ganado.**	Es importante poder controlar cada una de las tareas, actividades y procesos que se desarrollen en la implementación de nuevas prácticas. Asimismo, es importante poder medir el valor ganado a partir de los objetivos establecidos y el valor agregado identificado que la empresa desea alcanzar.
9	**Establecer procedimientos concretos para la combinación de múltiples enfoques**	Identificar los procedimientos concretos a seguir y que permitan llegar a cabo la integración de múltiples enfoques sin mayores traumatismos. La solución a seguir debe establecer los elementos necesarios para dependiendo las características de la empresa, por ejemplo: tiempo, recursos, sencillez, entre otros.
10	**Comunicación cara a cara**	El uso de herramientas tecnológicas no debe minimizar el valor de la comunicación CARA A CARA entre los participantes y miembros de equipos distribuidos geográficamente. En proyectos y compañías donde el desarrollo de software global es una de las formas principales de trabajo, es aconsejable al menos identificar una forma de comunicación en la cual las personas se puedan verse cara a cara.

3.4 Conclusiones

En este capítulo se ha presentado una revisión sistemática sobre las metodologías ágiles definidas e implementadas en las MiPyMEs desarrolladoras de software. La revisión ha seguido un formalismo el cual ha permitido tener una visión completa de la situación actual. Asimismo, de validar los resultados a partir del protocolo establecido para llevar a cabo la revisión sistemática.

De acuerdo a los resultados obtenidos, ha sido posible observar un creciente interés sobre las metodologías ágiles, tanto en el aumento de su aplicación e implementación en empresas desarrolladoras de software, así como en la cantidad de estudios relacionados a su adopción. Por otra parte, ha sido interesante poder identificar cómo el interés por integrar practicas ágiles con modelos tradicionales, convencionales, de facto y estándares internacionales ha aumentado considerablemente. Esto, quizás a necesidades particulares y también generalizadas, las cuales buscan solucionar problemas relacionados a: procesos complejos, retardos en las entregas, insatisfacción del cliente, costos, entre otros.

Para el fortalecimiento de la industria de software, es importante comprender que es imposible que un solo modelo o estándar solucione todas las necesidades de una

empresa, en este sentido, es necesario que los gobiernos y proyectos al interior de los distintos países tengan una visión incluyente y orientada a escenarios multimodelo, la cual permita que las MiPyMEs y grandes empresas puedan implementar e institucionalizar múltiples prácticas a partir de múltiples enfoques, permitiendo obviamente solucionar múltiples necesidades y que además sea acorde a sus características.

Un aspecto a resaltar, es la popularidad de algunos enfoques ágiles, así como SCRUM y XP, los cuales han sido ampliamente implementados e integrados por MiPyMEs y grandes empresas junto a otros modelos, así como CMMI, ISO/IEC 29110, modelo en espiral, cascada, ISO/IEC 15504, Kanban, entre otros. Por otra parte, también es importante resaltar que la implementación de modelos más formales en MiPyMEs con una base de gestión ágil, no siempre termina en la certificación de dichos modelos, la principal causa identificada es el costoso relacionado a su implementación y certificación, o simplemente porque no necesitan acreditar la certificación en un modelo en particular y sólo buscan darles mayor formalismo a sus procesos actuales. Lo mismo ocurre con las grandes empresas, aunque en su mayoría están certificadas en un modelo un gran porcentaje de estos ha decidido implementar e integrar un enfoque más ágil que permita reducir los costos asociados. El objetivo ha sido mantener la certificación del modelo formal, pero desde un enfoque ágil.

Con relación a las propuestas que integran enfoques formales con prácticas ágiles, es necesario destacar que se han desarrollado diversas soluciones que soportan su integración. Sin embargo, la gran mayoría toma como base SCRUM y XP como modelos (base) para llevar a cabo la integración de otros enfoques como CMMI, ISO 29110, Kanban, entre otros. Esto, quizá porque las MiPyMEs son el grupo más representativo en comparación al número que conforman las grandes empresas. Sin embargo, esta situación no ha disminuido el diseño de propuestas que permitan soportar la implementación de prácticas ágiles en empresas certificadas previamente en modelos con mayor formalismo.

En el estudio realizado también resalta los aspectos claves por el cual un enfoque ágil puede fallar dentro de una organización, de hecho, los principales factores están relacionados con una falta de entendimiento y de empoderamiento del enfoque, de sus

prácticas y artefactos. Comúnmente los nuevos equipos ágiles afrontan sus proyectos con poca orientación en el tema y muchas veces miembros del equipo pueden estar enceguecidos por las bondades del agilismo, olvidando o dejando de un lado todos los procesos técnicos necesarios para desarrollar una solución de software, o en el otro extremo, pueden estar totalmente en desacuerdo con el nuevo enfoque, generando más obstáculos al equipo y por ende al desarrollo de la solución. Otro factor clave que se presenta es la falta de involucramiento de los clientes en todo el proceso, lo cual esta desalineado con uno de los principios del agilismo y obteniendo como resultado proyectos con muchas incertidumbre e incumplimientos.

En base a los factores de éxito extraídos de la investigación y teniendo en cuenta los desafíos a los que se enfrentan las empresas con estos modelos de desarrollo, en el capítulo 4 se presenta detalladamente un modelo de desarrollo de software para MiPyMEs que integra tres enfoques claves. El primero, se toma las prácticas de un modelo estándar y certificable como la ISO/IEC 29110. El segundo, se agregan elementos de gestión ágil de proyectos de software. Y el tercero, se enmarcado en un proceso de gestión de conocimiento que apoye el crecimiento de la organización, de sus recursos humanos y de sus clientes. Con este modelo, se espera poder soportar la gestión ágil de proyectos de software además de facilitar la certificación de sus procesos de las empresas.

Capítulo 4. AgileFM: Modelo de gestión de proyectos ágiles formal basado en el estándar ISO/IEC 29110

"La ciencia es una ecuación diferencial. La religión es una condición de frontera", (Alan Turing, Matemático Británico, 1912-1954)

Con base en la investigación y análisis realizado a través de la revisión sistemática presentada en el capítulo anterior, en este capítulo se presenta AgileFM, un modelo para la gestión de proyectos de software para MiPyMEs tomando elementos de enfoques ágiles populares como SCRUM y XP, junto con elementos del estándar ISO/IEC 29110 que apoyen un proceso de certificación. La primera sección del capítulo se presenta un bosquejo general del modelo propuesto y en las siguientes secciones se detalla cada componente y elemento. El contenido de este capítulo es complementado por el capítulo 5, el cual presenta la aplicación de la propuesta en un estudio de caso.

4. AgileFM: Modelo de gestión de proyectos ágiles formal basado en el estándar ISO/IEC 29110

4.1 Elementos del modelo propuesto

Teniendo en cuenta el contexto y las necesidades presentadas en el capítulo anterior, se ha desarrollado un modelo ágil para el desarrollo de software basado en el estándar ISO/IEC 29110 denominado AgileFM. La Figura 4-1 presenta el esquema de los elementos que constituyen AgileFM. El modelo propuesto es iterativo incremental, toma prácticas y procesos de algunos enfoques ágiles como XP, SCRUM y principalmente ISO/IEC 29110. Asimismo, está orientado a entregas tempranas, continuas y en cortos plazos, a la realización de ceremonias[1], al involucramiento constante del cliente y a un trabajo colaborativo. Dado que el modelo incluye elementos claves del estándar ISO/IEC 29110, éste facilita la documentación adecuada de los proyectos y artefactos presentes en un proyecto, gestión del conocimiento generado para facilitar la toma de decisiones en proyectos futuros y la posibilidad de llevar a cabo la certificación para una MiPyME.

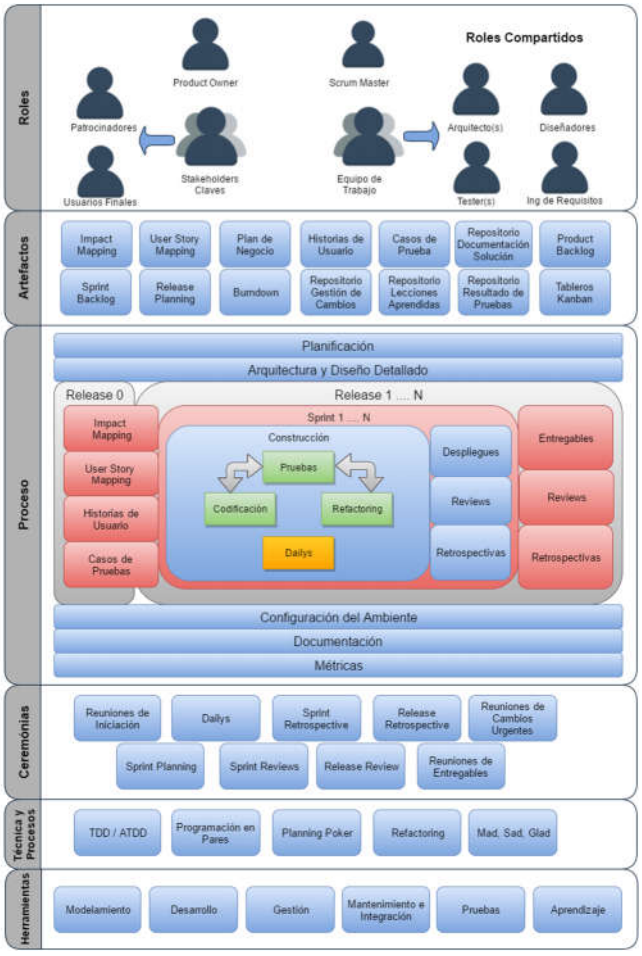
Como se puede ver en la Figura 4-1, el modelo define un conjunto de seis elementos esenciales:

- **Roles:** Determina el papel de cada individuo o grupo dentro del proyecto.

[1] Reuniones presenciales que buscan motivar la comunicación y entendimiento del problema por parte de todos los interesados. Son un mecanismo eficaz para construir confianza, compromiso en los individuos y clarificar dudas existentes.

- **Artefactos:** Son los elementos físicos o digitales que apoyan al proyecto y al equipo de trabajo. Los artefactos son considerados también como herramientas que ayudan a tener un lenguaje global por parte de todos los involucrados en el proyecto.
- **Ceremonias:** Reuniones presenciales que buscan motivar la comunicación y entendimiento del problema por parte de todos los interesados. Son un mecanismo eficaz para construir confianza, compromiso en los individuos y clarificar dudas existentes.
- **Técnicas y Procesos:** Son buenas prácticas utilizadas en la estimación, en la codificación y en las ceremonias.
- **Herramientas:** Herramientas tecnológicas que ayudan en la gestión de un proyecto de software, orientadas al trabajo colaborativo y a la automatización.
- **Proceso Principal:** Describe todo el ciclo de vida del producto a lo largo del proyecto.

Figura 4-1: Modelo de gestión de proyectos de software propuesto



En los siguientes apartados se describen de manera más detallada cada uno de los componentes que propone el modelo.

4.2 Roles del modelo

Cuando se empieza con un proyecto de desarrollo de software es importante determinar los actores que están involucrados y el papel que juegan dentro del mismo, esto ayuda a generar empoderamiento en las personas. En AgileFM se toman los roles propuestos por SCRUM que se han descrito en el capítulo 3, sección 3.1.3. A continuación, se presenta un resumen de estos con algunos elementos diferenciadores en algunos de ellos:

4.2.1 Product Owner

Es el representante del cliente y responsable que el producto ofrezca valor para el negocio. Además, define, aclara y prioriza los requerimientos conforme a las necesidades del negocio. Realiza un papel de gestor de Stakeholder acercándose a todos los actores que de forma directa o indirecta serán influenciados por el producto. La responsabilidad del producto final no debe recaer sobre el Product Owner, se debe considerar un trabajo de equipo.

4.2.2 Stakeholder claves

SCRUM como marco de trabajo solo hace especial énfasis en el Product Owner y su papel como puente entre el equipo de desarrollo y los Stakeholder, pero es importante tener presente a todos los involucrados del proyecto, ya sea que tengan una participación o se vean afectados directa o indirectamente. Este grupo de personas está conformado por todos los patrocinadores y usuarios finales de la solución, en muchas ocasiones algún miembro de este grupo puede participar en las ceremonias, previo acuerdo entre el Product Owner y el Equipo de Desarrollo, esto con el fin de dar un mejor entendimiento o esclarecer las incertidumbres presentadas en requisitos específicos.

También es importante tenerlos en cuenta a todos los Stakeholders, sobre todo a los usuarios finales en la presentación de los prototipos donde se detallen características de usabilidad, con el fin de que revelen características, comportamientos o carencias que no se hayan tenido en cuenta.

Es importante también realizar una clasificación de aquellos miembros que estén a favor, en contra o neutrales a la construcción de la solución y realizar trabajos especiales con cada subgrupo individualmente. Aquellos que están a favor se convierten en socios estratégicos del proyecto y en lo posible se deben tener muy contentos con los entregables realizados, ya que ellos ayudarán a difundir las bondades de la solución al resto de la organización. Los que están en contra deben ser escuchados cuidadosamente, muchas veces estas personas poseen una perspectiva diferente y ayudan a visualizar situaciones que no han sido detalladas por la empresa que puedan provocar el fracaso del proyecto, se deben hacer acercamientos a las personas de este grupo para determinar cuáles son los factores que provocan el rechazo al proyecto y trabajar sobre estos factores para generar confianza y seguridad a estas personas. Muchas veces, estos factores están relacionados con el miedo de perder el empleo, en estos casos es importante trabajar de la mano con el área de desarrollo humano para generar estrategias que den seguridad a sus empleados.

4.2.3 SCRUM Master

Cumple su papel como motivador, fomentando: la comunicación, auto-gestión y auto-organización, y como facilitador: ofrece los espacios y mecanismos para la comunicación y construcción de la solución, protege al equipo removiendo los impedimentos que se puedan presentar durante el desarrollo del proyecto, comparte y aprende constantemente sobre la gestión de proyectos, sobre agilismo y sobre habilidades humanas y de control que permitan vigilar que los lineamientos del método o enfoque empleado si se cumplen.

4.2.4 Equipo de desarrollo

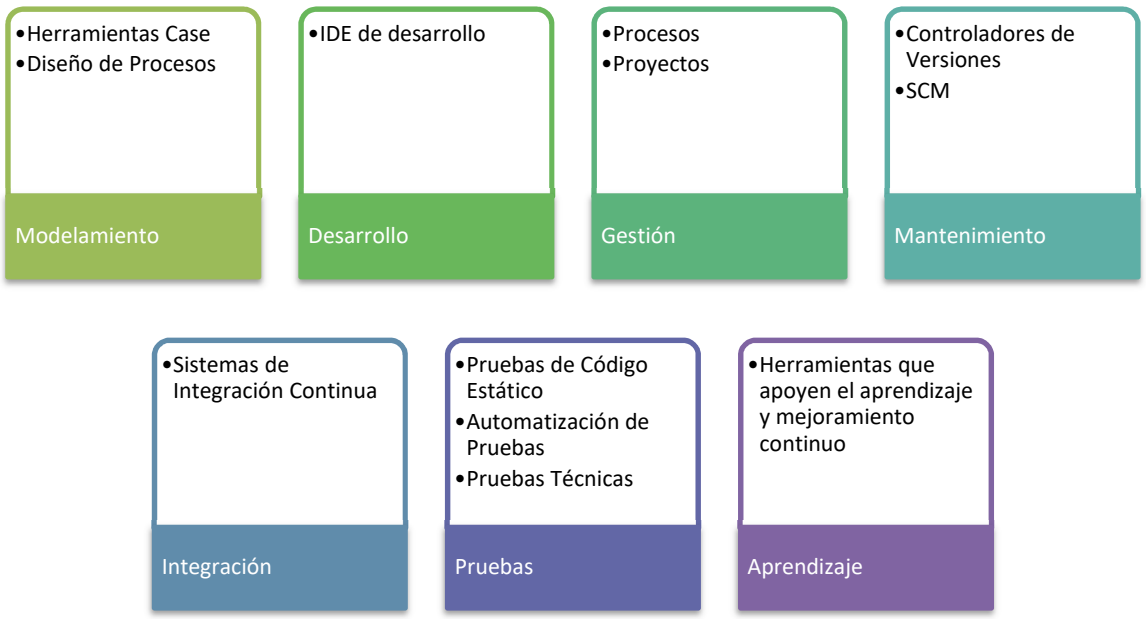
Son los responsables de la construcción de la solución y de realizar entregas que den valor en cada interacción. Adicionalmente, AgileFM busca que el equipo pueda compartir los roles, para desarrollar las habilidades de cada individuado en las distintas disciplinas que se requieren en el desarrollo de software y así poder mitigar las dependencias a personas en específico. Además, se busca eliminar la figura de Arquitecto de Software que realiza propuestas en las etapas iniciales, pero que no participa en la construcción e implementación de la solución, también se busca suprimir los roles de solo Testers que solo se centran en realizar pruebas y que carecen de habilidades en desarrollo de software.

Eliminando los roles específicos se remueven los futuros problemas o conflictos entre áreas, ocasionadas por las responsabilidades estrictas de cada rol, además de que habilita que todos los miembros del equipo se involucren en todas las etapas del proyecto y participen activamente.

4.3 Herramientas

Este componente toma elementos propuestos del estándar ISO/IEC 29110 enfocados a la gestión de proyectos, así como aspectos de gestión documental y de conocimiento. Adicionalmente, se incluye elementos de colaboración entre todos los participantes, tanto del lado del cliente como del equipo de desarrollo, esto, con el objetivo de diseñar una plataforma de desarrollo de software común para todos los proyectos y de estandarizar los procesos de la organización. Estas herramientas están clasificadas en siete grupos que detallaremos a continuación y que se puede apreciar en la Figura 4-2.

Figura 4-2: Plataforma de Desarrollo de Software



4.3.1 Herramientas de modelado

Estas herramientas están orientadas a apoyar la etapa de requerimientos para lograr entender el negocio o la necesidad del cliente. Algunas de las herramientas de modelado pueden ser:

- **Herramientas Case:** Enfocadas a respaldar el análisis de los requisitos del cliente y diseño de la arquitectura de la solución, ayudan a tener un lenguaje común entre todos los miembros del equipo de desarrollo. Asimismo, están enfocadas en lenguajes de modelamiento estandarizados como Unifield Modeling Language (UML). Algunas herramientas conocidas son: Enterprise Achitect, Smart Draw, StarUML, entre otros.
- **Diseño de Procesos:** Basados en los estándares Software & Systems Process Engineering Metamodel (SPEM) y Business Process Model and Notation (BPMN), el primero enfocado a procesos de software y el segundo a procesos de negocio. Estos metamodelos ofrecen las herramientas para plasmar los procesos corporativos de forma más entendible y gráfica para todos los participantes [57] [58].

4.3.2 Herramientas de desarrollo

Actualmente, la industria ofrece una gran cantidad de lenguajes de programación orientados a muchos propósitos y donde cada uno ofrece sus propias ventajas y desventajas. De acuerdo a lo anterior, es importante seleccionar y utilizar entornos de desarrollo integrados (IDE – Integrated Development Environment) que ofrezcan la versatilidad de emplear más de un lenguaje de programación y que se puedan acoplar fácilmente a Sistemas de Integración Continua.

Es importante que los equipos de desarrollo no se casen con un solo lenguaje de programación y que siempre tengan presente que dependiendo de la necesidad del proyecto se debe seleccionar la herramienta más adecuada. Por eso la importancia de individuos con excelentes habilidades técnicas e interesados en un proceso de constante aprendizaje.

El crecimiento de estos entornos ha sido enorme en las últimas décadas, entre los más relevantes nos podemos encontrar con Microsoft Visual Studio, Oracle Netbeans y Eclipse.

4.3.3 Herramientas de gestión

Su objetivo es apoyar la gestión del día a día del proyecto de software, enfocado en el trabajo colaborativo y que todos los miembros del proyecto tengan una visión global y compartida de los alcances y del estado del mismo. Es importante seleccionar herramientas en los dos siguientes grupos:

- **Gestión de Procesos:** Por lo general se hace uso de software especializado como Content Management System (CMS) y Document Management System (DMS) o aplicaciones híbridas como Drupal, Alfresco o SharePoint. Una buena elección de esta herramienta es esencial, porque se busca que el proyecto sea altamente colaborativo entre todos los participantes y que permita gestionar fácilmente el versionado y trazabilidad de toda la documentación generada. Estas plataformas ofrecen el espacio para construir sitios personalizados, blogs, foros y wikis con los cuales construir bases de lecciones aprendidas, reporte de bugs, actas de ceremonias, repositorios multimedia para fotos y videos, entre muchas características que dan soporte a la gestión del proyecto.
- **Gestión de Proyectos:** Son herramientas orientadas a la planeación y gestión integrada de los proyectos. De acuerdo a la robustez de la herramienta seleccionada se puede solo centrar en la construcción de los cronogramas, en ese caso se puede llegar a necesitar herramientas específicas de Gestión de Requisitos como IBM Rational Requierements Composer, Modelio Requirement Analyst, entre otros. Gracias al surgimiento de los enfoques ágiles se puede encontrar en el mercado herramientas especializadas como Jira, Team Foundation, Visual Studio Online, Trello o EasyBackLog que ayudan a llevar un control detallado de los requisitos, estado del proyecto, bugs encontrados, entre otros.

Con las últimas herramientas de gestión de procesos y proyectos ya mencionadas es suficiente para llevar un control del cronograma, de los requisitos, casos de pruebas y

bugs del proyecto de desarrollo de software, además de que se cumple las necesidades de gestión de proyectos y documental que se plantea en el estándar ISO/IEC 29110.

4.3.4 Herramientas de mantenimiento y de integración

Hoy en día es indispensable que los equipos de trabajo cuenten con sistemas de integración continua que habiliten al equipo a llevar un control del código fuente, control de versionado y cambios de los diferentes artefactos, trabajo colaborativo y mantenimiento de una versión estable para ser distribuida y versiones de desarrollo y pruebas. Es importante que la herramienta seleccionada tenga integración directa con el entorno de desarrollo IDE, lo cual permite que el trabajo por el equipo sea más fluido.

El mercado ofrece excelentes herramientas para realizar esta tarea como pueden ser Microsoft Team Foundation, Jenkins o GitHub, los cuales ofrecen integración con los IDE más populares y soportan gran cantidad de proyectos distintos.

Es importante definir desde un inicio esta plataforma, debido a que prácticamente es un matrimonio entre la empresa desarrolladora y la herramienta, y el divorcio en un momento inoportuno pueden atrasar e impactar negativamente el desarrollo de un proyecto en marcha.

4.3.5 Herramientas de pruebas

La calidad en los productos de software es un atributo altamente demandado por los clientes y que ya hace parte integral de la solución, es por eso, que en las negociaciones la calidad ya no es un valor agregado al producto sino una característica inherente a él.

Las herramientas de pruebas buscan almacenar la información de cómo se deben realizar, las actividades relacionadas y sus resultados, todo esto orientado al aseguramiento de la calidad del producto final. Durante el desarrollo de un proyecto se deben utilizar y realizar pruebas como:

- **Pruebas de código estático:** Se realizan sin ejecutar el código y buscan evaluar la calidad del mismo detectando posibles bugs, complejidad, permiten verificar que se

cumplan las reglas de codificación, entre otros aspectos. Una herramienta ampliamente utilizada es SonarQube.

- **Automatización de pruebas:** Se construyen programas que realizan pruebas que por lo general se ejecutarían de forma manual, ofreciendo la opción de ejecutarlas cuando se necesiten, lo cual permite reducir los tiempos de ejecución. Existen un gran número de aplicativos que realizan este tipo de actividades, los propios IDE's ofrecen herramientas para construir pruebas unitarias que pueden ser ejecutadas en cualquier momento, también aplicaciones más especializadas como Selenium que ofrecen las herramientas para programar el comportamiento de un usuario final sobre un ambiente Web o de Escritorio.
- **Pruebas técnicas:** Orientadas a medir el rendimiento de una solución de software a través de pruebas de carga, estrés y de rendimiento, con el fin de cumplir con los atributos de calidad que se hayan determinado para la solución y ofrecer productos tolerantes a fallos y altas concurrencias.

Es importante dentro del plan de trabajo incluir todo el tema relacionado con las pruebas, por eso la importancia de los roles compartidos y recursos de alta habilidad técnica en el cual todos los miembros del equipo de trabajo se involucren en la construcción de un producto de alta calidad.

4.3.6 Herramientas de aprendizaje

A través de la plataforma de gestión de contenido se busca construir bases de datos de conocimiento que apoyen el creciendo profesional de los miembros del equipo, esto se puede lograr a través de repositorios de lecciones aprendidas, bases de datos de incidentes y problemas conocidos, repositorios de video tutoriales enfocados en las herramientas empleadas, nuevas técnicas o mejores prácticas a implementar.

También es importante gestionar un plan de capacitaciones anuales relacionados con el proceso de desarrollo de software y afines a varios miembros del equipo, que luego se transformen en cursos internos para los demás empleados.

También se pueden construir charlas técnicas semanales o mensuales impartidas por algún empleado al resto del equipo sobre nuevas tecnologías, técnicas, entre otro conocimiento que ayude a la organización a creer.

4.4 Artefactos

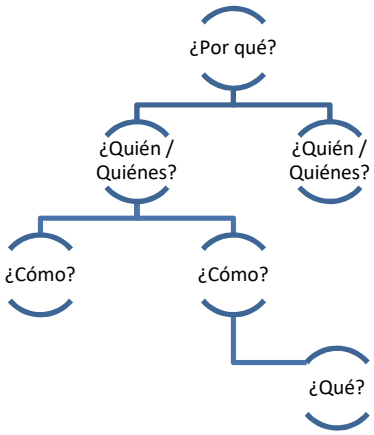
Los artefactos son instrumentos físicas o virtuales que ayudan a mantener organizado nuestros proyectos, a planificar cada una de las iteraciones, al trabajo colaborativo y a recopilar un conjunto de información vital para el desarrollo del mismo. La Figura 4-3 muestra un listado de artefactos propuestos desde metodologías ágiles y otros contextos para llevar a cabo su integración y aplicación junto con la norma ISO/IEC 29110. A continuación, se describen cada uno de ellos.

Figura 4-3: Artefactos Propuestos

4.4.1 Impact Mapping

Un Impact Mapping es una técnica de planificación y estrategia detallada en [59] la cual ayuda a visualizar el alcance y los lineamientos de una necesidad, su construcción debe realizarse de forma colaborativa por los dueños del negocio y miembros del equipo de trabajo. Su construcción es un mapa mental que va creciendo durante el proyecto y durante las reuniones que se presenten. La Figura 4-4 muestra los cuatro niveles que buscan responder un conjunto de preguntas claves.

Figura 4-4: Estructura Impact Mapping



- **¿Por qué?:** Es el punto de partida del mapa de impacto y busca responder la más importante pregunta: ¿Por qué estamos haciendo esto? Esta es la meta u objetivo que se desea lograr.
- **¿Quién / Quiénes?:** Es el primer nivel del mapa de impacto con el cual se busca identificar los distintos actores que puedan influenciar en el resultado del proyecto, a través de responder preguntas como: ¿El comportamiento de quién queremos impactar?, ¿Quién puede producir el efecto deseado?, ¿Quién puede obstruir los resultados?, ¿Quiénes son los consumidores o usuarios del producto?, ¿Quién va ser impactado por el producto?
- **¿Cómo?:** Segundo nivel del mapa, en el cual se busca identificar desde la perspectiva de cada actor cómo se obtiene la meta del negocio, cómo se realizan las actividades actualmente y/o cómo se desean en el futuro. A través de estas preguntas se puede obtener la información: ¿Cómo debe cambiar el comportamiento de nuestros actores?, ¿Cómo pueden ayudarnos a lograr el objetivo?, ¿Cómo pueden obstruir o impedir el éxito?
- **¿Qué?**: Es el último nivel del mapa y busca identificar los entregables de la solución por cada uno de los “Cómo” identificados. Se puede hacer uso de una pregunta como: ¿Qué podemos hacer, como una organización o equipo de trabajo, para soportar los impactos requeridos?

El mapa de impacto es de gran importancia para la realización del proyecto, esto debido a que sirve de insumo para otros artefactos que iremos viendo más adelante, además, es un artefacto que va evolucionado a medida que el proyecto avanza y por tal motivo debe

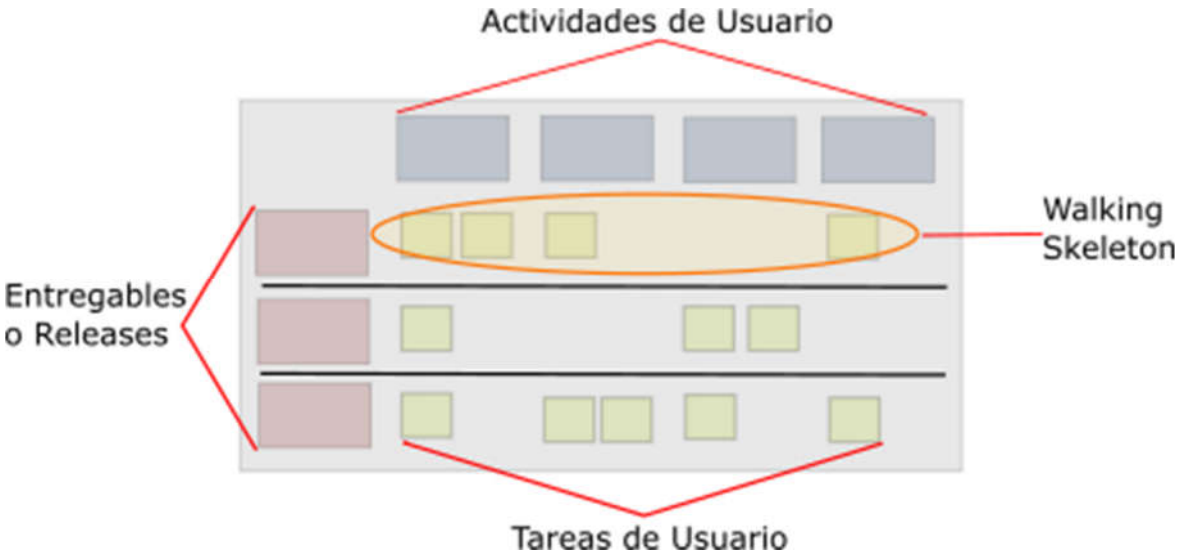
tener un control de versionado y trazabilidad a través de herramientas que se detallaron en la sección 4.3.3.

4.4.2 User Story Mapping

Es una técnica detallada por Jeff Patton [60] que permite generar una representación visual del sistema completo, ofreciendo una vista general de todas las funcionalidades que lo componen de punta a punta. Su construcción parte de la información obtenida del Impact Mapping y permite identificar las actividades de usuario que conformarán el proyecto, ofreciendo una visión del Product Backlog en dos dimensiones: Releases (entregables) y Funcionalidades (actividades de usuario).

En la Figura 4-5 se ve una representación visual de la construcción del User Story Mapping. Las tareas de usuario se representan horizontalmente según la prioridad que se pacte con el cliente, construyendo primero aquellas que más generen valor a la solución. Estas tares posteriormente se transformarán en historias de usuario que serán el insumo del equipo de desarrollo para construir la solución.

Figura 4-5: User Story Mapping



El mapa ira evolucionando durante el proyecto vertical y posiblemente horizontalmente a medida que se incluyan nuevas funcionalidades y se vayan definiendo los entregables de las tareas de menos prioridad.

4.4.3 Plan de negocio

Es un documento que se va construyendo con la información del Impact Mapping y del User Story Mapping, donde se realiza un análisis detallado de la necesidad planteada por el cliente y se detalla la arquitectura para la solución. Al igual que los anteriores artefactos es evolutivo durante toda la ejecución del proyecto, por lo cual es importante el control de versionado y trazabilidad del mismo.

La Tabla 4-1 detalla la estructura propuesta para el documento, esta puede variar de acuerdo al tipo de proyecto y/o a la madurez del equipo de trabajo.

Tabla 4-1: Estructura plan de negocio

Estructura Propuesta	
Sección	**Objetivo**
Introducción	Párrafo introductorio que especifique el objetivo y alcance del documento.
1. Análisis y Plan de Trabajo	
1.1. Enfoque	Párrafo que detalle la metodología de trabajo que se va a emplear, con sus ventajas y compromisos por parte del equipo de trabajo. Este punto es importante debido a que introduce al cliente en la forma de trabajo del equipo.
1.2. Entregables	Se detallan los entregables que va a recibir el cliente, que pueden ser documentos, diagramas, registros fotográficos, entre otros. Diferentes a la solución final.
1.3. Estimación	Estimación tanto en tiempo como en dinero del proyecto, que sirva de base para la toma de decisiones por parte del cliente. Es importante darle a entender que son tiempos y costos estimados que van a variar de acuerdo a los entregables y la priorización de las funcionalidades.
1.4. Matriz de Interesados	Se detallan todos los actores involucrados en el proyecto, previamente identificados en el Impact Mapping, detallando cuáles son sus intereses en el proyecto, su nivel de interés y de poder en la toma de decisiones con una escala de Alto, Medio y Bajo. Su importancia radica en poder identificar aquellos actores de gran poder que puedan obstruir el desarrollo del mismo, ya que es importante realizar un trabajo especial con estas personas para poder garantizar el éxito del proyecto.
1.5. Plan de Gestión de Requisitos	Se detalla que herramientas y mecanismos se emplearan para controlar los requisitos, detallando las bondades del mismo.
1.6. Plan de Comunicación	Es un punto muy importante, debido a que detalla cómo será la comunicación entre el equipo de trabajo y el cliente. Como serán las reuniones, el lugar, la intensidad, si serán presenciales o a través de herramientas de telepresencia. Esto será acordado entre el equipo de trabajo con el cliente, de acuerdo al enfoque propuesto. Tener en cuenta aspectos culturales y geográficos cuando son proyectos con personas ubicados en distintas ciudades, países o continentes.
1.7. Plan de Gestión de la Trazabilidad	Se detalla que herramientas, mecanismos y nomenclatura que se va a emplear para controlar la trazabilidad del documento y que todos los involucrados puedan consultar en cualquier momento la

	historia y ruta de un artefacto.
1.8. Plan de Gestión de Cambios y Control de Versiones de Requisitos	Se detalla que herramientas y mecanismos se van a emplear para realizar el control de cambios y versiones en los requisitos o funcionalidades del proyecto. Esto con el fin de llevar un registro histórico de los cambios presentados en el proyecto y que todos los involucrados puedan acceder a esta información.
1.9. Plan de Priorización	Detalla el procedimiento a utilizar para priorizar las funcionalidades de la solución, de acuerdo al enfoque detallado en el punto 1.1.
2. Arquitectura de la solución	
2.1. Enfoque de la solución	Párrafo donde se detalla el tipo de solución que se le va a ofrecer al cliente. Si es web, móvil, cliente servidor, entre otros. Además, inicialmente se puede detallar bajo que plataforma va a operar, si es en la nube, que tipo de repositorio va a emplear, entre otros aspectos importantes que se deben tener en cuenta y evaluar.
2.2. Beneficios de la Solución	Una lista de beneficios cuantitativos y cualitativos que la solución va a ofrecer al negocio y al cliente. Como reducción de tiempos en procesos, automatización, reducción de costos, entre otros que se puedan identificar en conjunto con el cliente.
2.3. Riesgos Identificados	Una lista de riesgos que puedan poner el éxito del proyecto junto con posibles soluciones para prevenirlos o mitigarlos. No se espera un proceso de gestión de riesgo detallada como se plantea en el Project Management Institute (PMI), debido a que esto genera mucho esfuerzo y tiempo.
2.4. Atributos de Calidad	De acuerdo a [61] y trabajando en conjunto con el cliente, se debe detallar los atributos de calidad más relevantes para la solución. Esto varía de acuerdo al tipo de proyecto y a las mismas necesidades del cliente. En la Tabla 4-2 se detalla una lista con los atributos de calidad y en la Tabla 4-3 se presenta la plantilla para su especificación.
2.5. Restricciones	Lista donde se detalle las restricciones de carácter técnico y del negocio que se puedan presentar en la solución. Como limitaciones de alguna tecnología en particular, limitaciones en tiempo o por leyes.
2.6. Visión General de Arquitectura	Diagrama general que la solución propuesta con una descripción detallada de los elementos más importantes.
2.7. Decisiones de Arquitectura	Por cada atributo de calidad identificado, se detalla las tácticas y patrones de arquitectura a ser utilizados para lograr la solución.
2.8. Vistas del Sistema	Utilizando el modelo de 4+1 vistas de Kruchten [62], se busca detallar los componentes más impactantes de la solución en donde se tiene: • Vista lógica: Representa la funcionalidad que el sistema proporcionará al usuario final. • Vista de despliegue: Muestra el sistema desde la perspectiva de un desarrollador. • Vista de procesos: Muestra los procesos que existen en un sistema y como estos se comunican entre sí. • Vista física: Muestra el sistema desde la perspectiva del arquitecto. • Vista de escenarios: Representación en casos de uso donde se unen y relacionan las demás vistas.

Tabla 4-2: Atributos de calidad de software

Atributo	Descripción
Disponiblidad	Relacionado con una falla del sistema y la duración de la misma.
Modificabilidad	Relacionado con el costo de un cambio, tanto en tiempo como en dinero.
Desempeño	Relacionado con la oportunidad. Cuando ocurre un evento el sistema debe responder de manera oportuna.
Seguridad	Habilidad de un sistema de prevenir o resistir a accesos no autorizados.

Testeabilidad	Relacionado con lo fácil que un software puede demostrar sus fallas o la falta de ellas.
Usabilidad	Que tan fácil es un sistema para un usuario para realizar sus tareas.

Tabla 4-3: Plantilla atributos de calidad

Ítem	Descripción
Identificación	Para controlar la trazabilidad. Por ejemplo, QAS001 o ACS001.
Descripción	Párrafo breve que detalla el comportamiento o necesidad que requiere la solución.
Afecta	Atributo de calidad de acuerdo a [61].
Origen del Estímulo	Quien genera el estímulo en la solución: Una persona, un sistema, entre otros.
Estímulo	Cuál es el evento o condición que necesita ser considerada.
Entorno	Cuáles son las condiciones cuando el estímulo ocurre.
Artefacto	Cuales elementos del sistema son afectados.
Respuesta	Actividad que debe ocurrir después de que llega el estímulo.
Medida de la Respuesta	Cuando ocurre la respuesta, esta debe ser medible ppr lo que el requisito se puede probar.
Justificación	Por qué este atributo es importante para la solución.
Importancia	Nivel dada en una escala de Alto, Medio o Bajo acordado con el cliente.
Dificultad Técnica	Nivel dado en una escala de Alto, Medio o Bajo que represente la dificultad de implementar el atributo en la solución. Tener en cuentas aspectos tecnológicos, curvas de aprendizajes de nuevas tecnologías o limitaciones del medio.

La idea es que este documento sea trabajado en una plataforma colaborativa, en donde tanto el equipo de trabajo como el cliente puedan interactuar y tener la última versión del mismo. Se pueden utilizar herramientas como Wikis o plataformas de gestión de contenido y de sitios como los mencionados en la sección 4.3.3.

4.4.4 Historias de usuario

Partiendo de la construcción del User Story Mapping, se empieza a tomar cada tarea de usuario para convertirla en una o varias historias de usuario. Las historias de usuario son descripciones cortas de una necesidad del cliente, los cuales son muy utilizadas bajo eXtreme Programming y SCRUM. Su desarrollo es en conjunto con el cliente, en este caso el Producto Owner, que le dará forma y sentido.

La Tabla 4-4 muestra la estructura propuesta para la historia de usuario.

Tabla 4-4: Estructura Historia de Usuario

Ítem	Descripción
Identificación	Para controlar la trazabilidad. Por ejemplo, HU001.
Título	Nombre corto que describa la historia de usuario.
Descripción	Contexto de la historia, donde se detalle el rol involucrado, la funcionalidad y el resultado esperado: Yo como <*rol*>,

	deseo <*funcionalidad*>, para <*resultado*>.
Criterios de Aceptación	Criterios que dan por aceptada la implementación de la historia, no debe ser mayor a 4, ya que esto es un índice de que la historia es muy grande. Debe incluir el contexto, el evento y el comportamiento esperado ante este evento.
Escenarios	Describe como el sistema debe comportarse desde el punto de vista de un usuario. Los escenarios pueden servir para validar el comportamiento, construir los casos de prueba y documentar el sistema: Dado <*contexto*> y <*más contexto*>, cuando <*evento*>, entonces <*resultado*> y <*otro resultado*>.
Tareas	Listado de tareas que se pueden realizar en uno o dos días que ayuden a llevar el avance de la historia durante la iteración o sprint. Son definidas por el equipo de desarrollo desde la perspectiva técnica.
Prioridad	De acuerdo a las necesidades del negocio, el cliente le da una prioridad a cada historia, el cual indica el orden a ser implementadas. De esta forma se logra que lo que más genere valor es lo primero que se le entrega al cliente.
Estimación	Puede ser dada por varias formas: ▪ Deal & Slide. ▪ Tallaje de camisetas ▪ Planning Poker Su objetivo es medir el nivel de esfuerzo que se requiere para su implementación y sirve de insumo para la construcción del Release Planning.

Las historias de usuario deben cumplir con la característica de INVEST [63]:

- **Independiente:** La historia de usuario no debe depender de otras historias.
- **Negociable:** No son contratos, sino promesas de comunicación.
- **Valor:** Valiosa para el cliente.
- **Estimable:** Debe indicar el esfuerzo que requiere para su implementación.
- **Small (Pequeña):** Debe ser pequeña y concisa. Cuando una historia no puede ser desarrollada en un sprint o iteración de denomina que es épica, y debe ser divida en varias historias de usuario.
- **Testeable:** Debe permitir escribir un caso de prueba que valide su funcionamiento.

Por lo general, en muchas ocasiones se presentan elementos no funcionales que no son considerados un entregable al usuario y por consiguiente no le generan un valor palpable. Estos elementos son conocidos como historias técnicas que requieren de su especificación e implementación para poder construir las historias de usuario. Al igual que las historias de usuario, deben tener una identificación, un título, una descripción y una priorización. La descripción no presenta el mismo formato empleado en la historia de usuario, esta se detalla como un requisito:

El sistema debe <*funcionalidad*>,
para <*resultado*> o Implementar o instalar <*funcionalidad, necesidad*>,
para <*resultado*>.

4.4.5 Casos de prueba

Es un conjunto de condiciones o variables bajo las cuáles se puede determinar si una aplicación o solución es satisfactoria o no. El insumo para su construcción está dado por los escenarios identificados en las historias de usuario.

Los casos de prueba son muy importantes, debido a que, con esto el equipo de desarrollo puede codificar las pruebas para las distintas historias de usuario, además, pueden fácilmente realizar prácticas como TDD y ATDD, en donde el desarrollo de una solución de software está basado en las pruebas, con el objetivo de ofrecer un producto de mayor calidad.

En la Tabla 4-5 se propone es una estructura para su definición.

Tabla 4-5: Estructura casos de prueba

Ítem	Descripción
Identificación	Para el control de su trazabilidad.
Historia de Usuario	Para indicar a que historia de usuario pertenece.
Escenario	Para indicar a cuál escenario dentro de la historia de usuario pertenece.
Precondiciones	Todo proceso o información previa a la ejecución que se debe tener en cuenta.
Valores de entrada	Conjunto de variables con sus valores que se requieren para el inicio de la prueba. Cada variable puede tener un rango de valores de acuerdo al escenario en el que se esté trabajando.
Resultados esperados	Descripción de los resultados que se esperan tras a ver completado la prueba.
Dependencias	Para indicar que otros subsistemas están involucrados y en qué grado.

4.4.6 Product Backlog

Es la lista de historias de usuario definidas y priorizadas con el Product Owner según el valor que genere al negocio. Esta lista ayuda a tener una perspectiva global por parte de todos los miembros del proyecto de qué se debe hacer y tener claras las prioridades del cliente [25]. El product backlog permite que el equipo de trabajo sea auto-disciplinado y auto-organizado y le permite al cliente redefinir las prioridades de acuerdo a las necesidades cambiantes del negocio.

También permite manejar la incertidumbre del proyecto, obligando al equipo de trabajo y al cliente a detallar más las historias más prioritarias y postergar las de menos prioridad, que en este caso estarían al final de la lista. De esta forma, se van detallando alrededor de las primeras 20 historias de la lista definiendo sus criterios de aceptación, escenarios y tareas, con el fin de que el equipo de desarrollo en el próximo sprint la pueda implementar.

4.4.7 Sprint Backlog

Es la lista de historias de usuario que se van a desarrollar en un determinado sprint, previamente definidas en una reunión de iniciación de sprint del equipo de trabajo y el Product Owner y la priorización dada a cada historia [25].

4.4.8 Repositorio documentación de la solución

Es un espacio colaborativo implementado en la herramienta de gestión seleccionada en el cual toda la información que genere el proyecto es respalda y compartida con todos los interesados, ofreciendo transparencia y una visión global y unificada por todos.

Esta documentación puede ser modelos de proceso de negocio, políticas corporativas, plantillas, macros, leyes que apliquen, manuales o instructivos finales del producto, entre otros documentos que sean importantes para el desarrollo del proyecto.

4.4.9 Release Planning

Es una estimación que se realiza sobre la información que se tiene de las historias de usuario, está dada por el esfuerzo de cada una de las historias de usuario y además está dividida en releases y sprints en un periodo de tiempo. Es importante hacerle entender al cliente que la estimación no implica una promesa, debido a que, gracias a la naturaleza cambiante de los negocios, las prioridades pueden cambiar, la necesidad de incluir nuevas funcionalidades, restricciones de negocio o técnicas puede cambiar en cualquier momento, estas y otras variables pueden afectar la estimación [25].

La Tabla 4-6 muestra una estructura simple para construir un Release Planning.

Tabla 4-6: Estructura simple de un Release Planning

	Esfuerzo	Rango de fecha por día o por semana desde la iniciación del proyecto						
Release 1								
Sprint 1								
HU001	1							
HU005	5							
HU006	½							
Sprint 2								
HU002	5							
HU003	3							
Release 2								
Sprint 3								
HU007	3							
HU008	5							

El rango de fecha se obtiene calculando en base a la cantidad de esfuerzo de todo el product backlog, la velocidad del equipo y la duración en semanas de un sprint. De esta forma calculamos esto con base en la siguiente formula:

$$Semanas = (\Sigma\, esfuerzo\ product\ backlog \div velocidad\ equipo) \times semanas\ sprint$$

Por ejemplo, si tenemos un product backlog con un valor de 1000 puntos, la velocidad del equipo es de 100 puntos y se tiene que cada sprint dura 2 semanas, entonces aplicando la fórmula tenemos:

$$Semanas = (1000 \div 100) \times 2$$
$$Semanas = 20$$

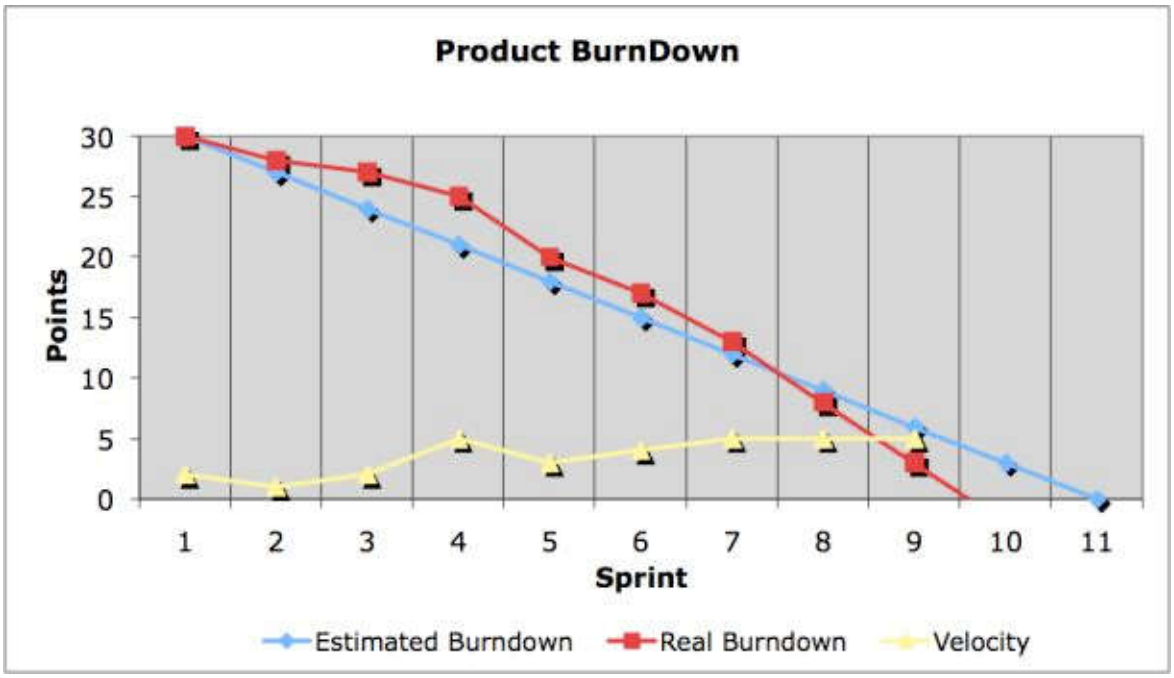
Los colores al frente de cada historia representa el esfuerzo de su implementación dependiendo de la velocidad del equipo. Esta velocidad que se va obteniendo con la experiencia del equipo de trabajo sobre la metodología de trabajo y el aumento de las habilidades técnicas. La velocidad representa la cantidad de historias de usuario dependiendo de su esfuerzo en puntos que un equipo de trabajo es capaz de implementar durante un sprint.

4.4.10 Burndown

El Burndown es una representación gráfica del trabajo que hay por hacer versus el tiempo. Las tareas pendientes del backlog se representan en el eje vertical, con el tiempo a lo largo de la horizontal. En pocas palabras es un gráfico de ejecución de las tareas pendientes que ayuda a predecir cuando el proyecto o la solución se completará.

La Figura 4-6 muestra un ejemplo de la representación del gráfico. El eje Y representa los puntos del proyecto, mientras el eje X visualiza los sprint. La primera gráfica es la estimación de los puntos a desarrollar por sprint, la segunda gráfica son los puntos reales que se han implementado por sprint; estas dos primeras gráficas son un acumulado que va disminuyendo durante cada sprint. La tercera gráfica representa la velocidad del equipo durante cada sprint, representado por los puntos específicos implementados en cada sprint.

Figura 4-6: Ejemplo Burndown



Para los equipos de trabajo inmaduros esta gráfica es un gran aliado para poder conocer su verdadera velocidad y poder ajustar la estimación del proyecto a algo más cercano a la realidad. No es de extrañar que en las primeras iteraciones hallan desfases y no se logre cumplir con la estimación.

Cuando la segunda gráfica (lo real) está por encima de la primera (lo planeado), quiere decir que no se está cumpliendo con lo comprometido en el sprint. En este punto es importante ajuntar la velocidad del equipo para la próxima iteración y analizar junto con el Product Owner las historias de usuario tanto del sprint actual como del próximo, ya que pueden estar mal estimadas, no estar bien claras o estar muy épicas y ser necesario descomponerlas aún más. Todos estos indicadores o análisis se logran gracias a este simple artefacto que ayudan a direccionar el proyecto oportunamente hacia el éxito.

4.4.11 Repositorio de gestión de cambios

El objetivo de este artefacto es llevar una bitácora de los cambios que surgen durante el proyecto, debido al ambiente cambiante del entorno. Estos cambios pueden deberse a muchos factores como modificación de políticas organizaciones que afectan los procesos de la empresa, cambios en leyes o normativas gubernamentales que afectan al negocio, muy común en sectores como el financiero y salud.

Haciendo uso de la herramienta de gestión de contenido se puede construir un repositorio a través de un Blog (registro más informal) o una lista más estructurada (registro más formal), donde se describa cual es el cambio que se requiere y la justificación de este y además a que elementos tanto del Impact Mapping, del User Story Mapping, del plan de negocio, de las historias de usuario y los casos de prueba son afectados y deber ser redefinidos en conjunto con el Product Owner y con los stakeholders puntuales del caso.

Este repositorio busca que haya transparencia en las necesidades cambiantes del negocio y de la solución, busca que todos los participantes tengan el mismo entendimiento del proyecto y apoya el trabajo colaborativo en donde todos los participantes pueden aportar ideas que apunte a una óptima solución al cambio presentado.

4.4.12 Repositorio de lecciones aprendidas

Este repositorio busca que el equipo de trabajo construya una base de conocimiento que trascienda más allá del proyecto actual. Su objetivo es construir un espacio, ya sea un blog o una wiki, con ayuda de la herramienta de gestión de contenido en donde el equipo

de trabajo pueda plasmar el conocimiento nuevo adquirido durante la ejecución de un proyecto; este conocimiento puede estar dada sobre la utilización de una buena tecnología, sobre algún nuevo patrón arquitectónico o de diseño que dan solución a un problema en concreto, también soluciones técnicas para resolver problemas que se presentan en la utilización de una herramienta tecnológica.

Pero no todo es de carácter técnico, también se puede plasmar conocimiento orientado a la gestión del proyecto, sobre mejores prácticas en la utilización de los artefactos, buenas y malas experiencias en las ceremonias (reuniones) con el cliente y como se pueden mejorar estas experiencias.

Todos los miembros del equipo deben participar en la construcción de este repositorio, debido a que todos son responsables del proyecto, además promueve el trabajo colaborativo y el empoderamiento de cada persona sobre el proyecto y sobre la empresa.

Este repositorio debe ser consultado permanentemente, al inicio del proyecto para identificar patrones de proyectos anteriores que se pueden aplicar al actual, durante su ejecución para buscar soluciones a problemas que se presentan y para registrar nuevos sucesos y al final para registrar las experiencias al realizar la entrega en donde se describa el sentir del equipo de trabajo y del cliente, tanto a nivel técnico y de gestión del proyecto.

4.4.13 Repositorio de resultado de pruebas

Se busca construir un repositorio bajo la herramienta de gestión de contenido que se llene automáticamente con los resultados de las pruebas que se realicen sobre la solución, tanto las pruebas estáticas, automatizadas y las técnicas. Este repositorio debe tener campos claves como el tipo de prueba, la fecha de ejecución y la descripción del resultado de la prueba.

Este repositorio ofrece las evidencias necesarias a todos los participantes de las pruebas realizadas sobre toda la solución, de la calidad de la misma y del estado final de las pruebas del producto final.

4.4.14 Tableros Kanban

Estos tableros son artefactos que apoyan a la productividad, transparencia y trabajo colaborativo, ofrecen una vista global unificada para todos los participantes [7].

Para esta propuesta se utilizan tableros Kanban en el User Story Mapping para desagregar la solución en actividades o módulos claves dividido en tareas y en entregables, lo cual ayuda a visualizar las funcionalidades más prioritarias para el cliente.

Para el Product Backlog en donde todas las historias de usuario se introducen y van pasando por tres columnas mínimas, Por hacer (To Do), Haciendo (Doing) y Hecho (Done). Durante la ejecución de un sprint y el paso entre sprint se debe ver movimiento en este tablero, lo cual indica que se está avanzando en el proyecto, de lo contrario el SCRUM Master debe tomar medidas e identificar lo más pronto posible el problema que está ocurriendo en el desarrollo de la solución.

Para el Sprint Backlog es en donde se llevan las tareas que componen cada historia de usuario y al igual que el tablero anterior, mínimamente debe estar dividido en: Por hacer (To Do), Haciendo (Doing) y Hecho (Done). El seguimiento de este tablero debe hacer al inicio y al final de cada día para evaluar su evolución, esto debido a que las tareas deben poder realizarse en un lapso no mayor de 2 días. Si este tablero no presenta cambios durante el día el SCRUM Master debe intervenir para determinar la causa y ayudar a eliminar los posibles obstáculos que se estén presentado.

Los equipos de trabajo más maduros personalizan sus tableros de acuerdo a sus necesidades y experiencia, de acuerdo a particularidades del proyecto o bien, de acuerdo a mejores prácticas que se pueden encontrar en la literatura. Lo que es importante tener en cuenta es que el tablero debe buscar ayudar a todos los participantes del proyecto y a ser más productivos, si esto no se está logrando se debe reevaluar la división de los tableros, ya que posiblemente este ocasionando más reprocesos y confusión en su utilización.

También es bueno tener en cuenta que los tableros físicos ofrecen mejores resultados que los tableros online o digitales, sobretodo en equipo de trabajo con poca experiencia,

esto debido a que el tablero ofrece una visual permanente del estado del proyecto y obliga al equipo a estar pendiente del estado de salud del proyecto y ser más proactivos a los posibles problemas. Los tableros online ofrecen una funcionalidad igual, pero dependen de notificaciones en aplicaciones o en dispositivos móviles que fácilmente pueden ser ignoradas por las personas. Los tableros online son más recomendados para equipos muy maduros los cuales ya se han empoderado de la metodología y ven este artefacto como una herramienta vital para su trabajo.

4.5 Ceremonias

Las ceremonias son reuniones preferiblemente presenciales entre el equipo de trabajo, el SCRUM Master, el Product Owner y los Stakeholders claves, en donde se busca dar claridad de las necesidades del negocio y construir en conjunto una propuesta de solución que pueda ser implementada. En las siguientes secciones de detallan las ceremonias mínimas que debe tener un enfoque ágil tomadas de las ya propuestas por SCRUM.

4.5.1 Reuniones de iniciación

Son un conjunto de reuniones que se deben realizar al inicio del proyecto junto con los actores claves del cliente, esta primera etapa o iteración se conoce como Release 0 y buscar dar a conocer el equipo de desarrollo al cliente, identificar los actores claves del proyecto que conformaran los Stakeholders y se define el Product Owner. No hay un número mágico de reuniones que se deben hacer, ya que esto está relacionado con la complejidad del proyecto, aspectos sociales, geográficos y hasta políticos.

Las primerias reuniones deber ser muy exploratorias, preferiblemente en las instalaciones del cliente, con el fin de poder conocer el negocio y sus procesos y empezar a lograr un entendimiento de la necesidad. Se debe realizar charlas con los actores claves para conocer como realizan sus actividades diarias, identificar aplicaciones legadas o procesos manuales que realicen. También se debe empezar a identificar y asignar el Product Owner del proyecto, el cual es muy importante para el resto de las ceremonias del proyecto y representa la piedra angular del éxito del mismo.

En las reuniones subsiguientes se empieza a construir el Impact Mapping el cual es el insumo principal del proyecto para la construcción de los demás artefactos y de la implementación final de la solución. Con este artefacto ya claro se puede empezar a construir el User Story Mapping en conjunto con el Product Owner y otros stakeholders claves que el considere importantes, además el equipo también puede empezar a crear el Plan de Negocio donde se detalla las necesidades del cliente, la propuesta y la arquitectura inicial planteada. Recordar que todos estos artefactos son evolutivos y sufrirán cambios conformen vaya avanzando el proyecto.

Con una definición inicial del User Story Mapping se puede empezar a construir las historias de usuario y los casos de prueba que son más importantes en valor para el cliente, las cuales serán insumo para las próximas iteraciones y la implementación de la solución.

Al final de estas ceremonias se debe tener las versiones iniciales del Impact Mapping, del User Story Mapping, Historias de Usuario, Casos de Prueba y Plan de Negocio; estos artefactos ofrecen a todo los involucrados la misma visión y entendimiento del proyecto y su construcción apoya al trabajo colaborativo y auto-organizado.

Estas reuniones pueden tener una duración máxima de cuatro horas para evitar fatigas entre los participantes y se pueden realizar entre dos y tres a la semana, dependiendo de la necesidad, complejidad y tiempos del proyecto.

4.5.2 Sprint Planning

Son reuniones que se llevan a cabo al inicio de cada Sprint, en éstas se discute con el Product Owner que historias de usuarios van a ingresar para ser implementadas, teniendo en cuenta la priorización dada por el cliente y por la velocidad del equipo, de esta forma se trabaja en lo más importante y no se asigna más trabajo del que el equipo es capaz de realizar.

En estas reuniones también se resuelven inquietudes sobre los requisitos que el equipo pueda tener y se empiezan a auto-organizar para formar pequeños sub-equipos de trabajo y seleccionar una historia a solucionar.

Lo ideal es que esta reunión no sobrepase las cuatro horas, si es necesario se puede extender un poco más en consenso con el resto de los participantes. No se recomienda que se parta la reunión en varios días, ya que esto afecta los tiempos de proyecto y la continuidad de las discusiones

4.5.3 Daily meeting

Reuniones cortas diarias donde solo participan el equipo de trabajo y el SCRUM Master, en determinados momentos, previamente acordado puede participar el Product Owner, con el fin de dar claridad a inquietudes del equipo. Su duración esta entre diez y quince minutos y se realiza de pie para evitar que los participantes se salgan del tema de la reunión.

La idea es que cada miembro del equipo de trabajo exponga brevemente que logró hacer el día anterior, que piensa realizar el día actual y que problemas o situaciones se le están presentado que puedan obstaculizar su trabajo. El SCRUM Master debe tomar nota de estos problemas para buscar soluciones o guiar al equipo para que puedan realizar su trabajo.

Una buena práctica es que durante el día los miembros del equipo vayan anotando en post-it aquellos obstáculos que se presenten y depositarlos en algún buzón o urna, que luego el SCRUM Master pueda ir leyendo y analizando previamente; de esta forma durante los daily’s meeting se ahorra tiempo y se puede discutir sobre las posibles acciones o soluciones que vayan a realizar.

Estas reuniones ayudan a formar más unión y confianza entre el equipo o que todos conozcan el estado del proyecto y a dar soluciones a problemas que se presenten, ofrece un sentir de auto-organización y colaboración que es importante en un enfoque ágil y ayuda a mitigar poco a poco las barreras jerárquicas que imponen las organizaciones tradicionales.

Cabe destacar que este tipo de reuniones son ideales para equipo pequeños no mayores de nuevo o diez personas. En equipos más números, esta práctica representa un indicador de ineficiencia por la gran cantidad de tiempo que se emplea en su ejecución. En estos casos el equipo de trabajo debe ser segmentado en equipos más pequeños,

cada uno con su propio SCRUM Master y llevar sus propios daily's meeting individualmente y posteriormente los SCRUM Masters pueden llevar pequeñas reuniones donde se pongan al día con los temas de cada equipo de desarrollo.

4.5.4 Sprint Review

Al finalizar cada Sprint se realiza una reunión con el Producto Owner y los Stakeholder que el considere necesarios para revisar los avances del proyecto, pudiendo ver funcionalidades palpables que le den valor y que pueda determinar si cumple o no con la historia de usuario asociada.

Dado el caso que los criterios de aceptación no se cumplan en su totalidad se debe marcar la historia para ser analizada y detallada posteriormente al finalizar esta reunión o al inicio de un Sprint Planning. Las historias que cumplen con los criterios de aceptación y sus casos de prueba pueden pasar a Hechas (Done) en el tablero kanban del Product Backlog.

La reunión debe estar en un rango de cuatro a seis horas y se debe centrar en la presentación de las historias de usuarios asignadas e implementadas en el Sprint. Si al finalizar esta tarea sobra buen tiempo, se puede empezar a analizar las historias que no pasaron, para determinar cuáles son los problemas, las soluciones, actualizar la historia de usuario y asignarla en primer lugar en el Product Backlog. Si es necesario se realiza una entrada en el repositorio de gestión de cambios y en el de lecciones aprendidas.

4.5.5 Sprint Retrospective

Es una reunión en la cual se discute a través de técnicas de retrospectivas el sentir del equipo de trabajo y del Product Owner sobre el proyecto y el proceso que se está llevando, sirven para conocer como están las cosas e iniciar procesos de mejoramiento a aquellas actividades u acciones que no están ofreciendo el resultado esperado.

En [64] y [65] se detallan los pasos y aspectos a seguir en este tipo de reuniones, recalcan la diferencia entre una reunión de retrospectiva a una de lecciones aprendidas y enfatizan que las retrospectivas deben ser positivas y actuar como un catalizador para el cambio y no un analizador de problemas y responsables.

Estas reuniones se centran en plasmar visualmente las cosas que se hicieron bien durante la iteración y que se deben seguir haciendo durante el proyecto, las cosas que no se hicieron bien y no dieron los resultados esperados y que se deben mejorar y cosas que no se hicieron bien, dieron malos resultados y no se deben volver a realizar.

El factor que normalmente afecta esta práctica está dado por la falta de confianza entre los miembros del equipo y el miedo a ser juzgados. En este punto el SCRUM Master juego un rol importante creando un espacio de tolerancia y libre de juicios, imponiendo orden a los participantes que no acaten las reglas. También se pueden usar mecanismos para mejorar la confianza en la herramienta paulatinamente. Por ejemplo, el SCRUM Master puede disponer de una runa cerrada en la cual los miembros del equipo pueden ir introduciendo Post-It relacionados con los tres aspectos mencionados en el párrafo anterior, al inicio pueden ser notas impresas para evitar que reconozcan la letra de algún integrante y conforme va pasando el tiempo estos pueden ser escritos a mano a medida que se gane confianza.

Con los temas ya tratados se empiezan a agrupar y proponer procesos de mejora que el SCRUM Master estará pendiente de su ejecución y resultados del mismo.

4.5.6 Release Review

Esta reunión cumple una función simular al Sprint Review, se diferencia en que en el Sprint Review solo se muestran las historias ya implementadas y funcionando y no se le entregan al cliente para que empiece a operar, mientras que en el Release Review se le entrega al cliente módulos ya operando y desplegados que ofrecen valor y que pueden empezar a utilizar.

Tener en cuenta que un Release puede estar conformado por uno o más Sprint y al final del último Sprint se realiza un Release Review en vez del Sprint Review. Además, ya desde el primer Release Review se pueden empezar a planear campañas de capacitación y de gestión del cambio a los actores claves.

4.5.7 Release Retrospective

Cumple la misma funcionalidad que el Sprint Retrospective y se centra en analizar el trabajo realizado durante todo el release, además el SCRUM Master puede analizar si las acciones de mejora si están funcionando o si situaciones mencionadas en anteriores Sprint Retrospective se siguen manifestando.

Al final del último sprint del release se realiza el Release Retrospective en vez del Sprint Retrospective.

4.5.8 Reuniones de entregables

Después de haber realizado el Release Review y de ajustar aquellas cosas de bajo impacto en el desarrollo, se realiza un despliegue de las funcionalidades listas al cliente y se inicia con el mismo un proceso de capacitación y de gestión de cambio.

Este despliegue puede ser sobre la infraestructura final del cliente o sobre una provisional que sea fácil después migrar a la definitiva. Además, es importante hacer seguimiento diario con el cliente sobre la operación del sistema, esto ayuda a que el cliente se sienta acompañado y a identificar posibles fallas de manera oportuna.

Es una buena práctica llevar unas actas de entrega en donde se especifique que módulos se están entregando y a que personas se están capacitando, esto con el fin de llevar un mejor control y trasparencia en el proceso.

4.5.9 Reuniones de cambios urgentes

Son reuniones programadas por el Product Owner en el momento que cree conveniente, debido a cambios que se presenten en la solución por fuerza mayor, ya sea porque hubo cambios en las políticas corporativas, cambios en leyes, normas, entre otros.

En estas reuniones se discute los cambios que requiere la solución y se empieza hacer un análisis del impacto del mismo sobre las historias ya implementadas. Toda esta información debe ser registrada en el repositorio de gestión de cambio y se debe iniciar la actualización de los artefactos afectados, desde el Impact Mapping hasta el Release Planning.

4.6 Técnicas y procesos

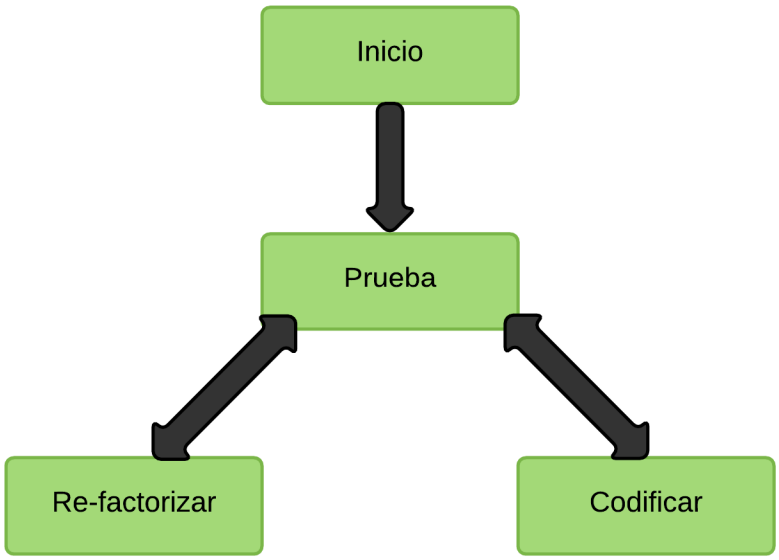
Construir una nueva solución de software requiere además de un equipo de altas habilidades técnicas, de aplicar algunas técnicas y/o procesos que son recomendadas como buenas prácticas por la comunidad académica y la industria, debido a los nuevos resultados que ofrecen en el producto final y en la satisfacción del cliente.

A continuación, se presentan algunas de estas técnicas y procesos a tener en cuenta:

4.6.1 TDD / ATDD

Test-driven development (TDD) [66] es una técnica que se basa en desarrollar primero las pruebas unitarias, luego se desarrolla el código que resuelve la prueba y por último se re-factoriza el código construido. La Figura 4-7 detalla el ciclo de desarrollo que empieza con construir la prueba, realizar la codificación, nuevamente correr la prueba, si esta es aceptada se pasa a realizar la re-factorización y finalmente probar de nuevo.

Figura 4-7: Ciclo de desarrollo de TDD



Por otro lado, tenemos acceptance test-driven development (ATDD) [67] que es un enfoque en donde hay una fuerte participación el Producto Owner, de los stakeholder involucrados y del equipo de trabajo donde se analizan las historias de usuario, sus

criterios de aceptación para poder construir los escenarios que describan el comportamiento de la necesidad, con esta información se convierten los escenarios en pruebas automatizadas haciendo uso de alguna herramienta de las mencionadas en la sección 4.3.5.

Ambos conceptos se complementan, debido a que a través de ATDD todos los participantes del proyecto obtienen claridad de las necesidades y se obtiene las pruebas que servirán de insumo para TDD, en el cual ya pueden pasar a la codificación y re-factorización mientras se va probando lo construido.

Estos enfoques ofrecen varias ventajas como las siguientes:

- Ayuda a realizar un análisis previo y detallado de las necesidades y de los diversos escenarios, además de que da claridad y el mismo entendimiento a todos los involucrados en el proyecto.
- Evita que se obvie pruebas o escenarios, lo cual puede pasar al realizar las pruebas al final.
- Evita la construcción de código innecesario, debido a que la codificación se centra en pasar su prueba.
- La realización de una buena etapa de re-factorización hace que el código sea más legible, óptimo y fácil de entender.

4.6.2 Programación en pares

La programación en pares o pair programming [4] [68] es una técnica en la cual dos programadores unen esfuerzo para construir una historia y en la cual cada uno toma un rol diferente. Uno de ellos se convierte en el controlador y se encarga de realizar la codificación y el otro, toma el rol de navegador que dirige la situación, está pendiente de posibles errores y va pensando en maneras de mejorar el código.

Se sugiere que los roles se estén intercambiando durante el día, en la literatura se sugiere cambios cada media hora o cuando se complete la prueba de una actividad o tarea de la historia. Algunos estudios sugieren partir el día entre tres o cuatro cambios de rol y no hacerlo demasiado seguido para evitar la pérdida de concentración o el impulso

del que lleva el rol de controlador en el momento. También es recomendable variar las parejas entre sprint, esto ayuda a mejorar la confianza, camarería y trabajo en equipo de todos los miembros y ayuda a que todos tengan la misma visual de la solución que se está construyendo y se elimina las dependencias de componentes construidos a personas específicas [46] [29].

Esta técnica ofrece muchas ventajas como las mencionadas a continuación:

- Más disciplina, se hace lo que se debe hacer.
- Mejor código, al realizar el trabajo en pareja se reduce el riesgo de construir malos diseños, lo que se traduce en mayor calidad del producto final.
- Un flujo constante de trabajo, debido a que el trabajo en pareja ofrece una recuperación más rápida, es decir, cuando se finaliza el día, el próximo se inicia más rápido.
- Es una técnica resistente a las interrupciones, debido a que uno de los miembros atiende la interrupción mientras el otro continúa codificando.
- Las rotaciones de parejas ofrecen una perspectiva global y compartida de la solución, lo que ayuda a superar problemas que se presenten, debido a que todos los miembros del equipo de desarrollo conocen el código y pueden ofrecer soluciones.
- Ayuda al proceso de capacitación de los desarrolladores novatos, al emparejarlos con los más expertos, lo que incentiva el compartir conocimiento y mejora la moral.

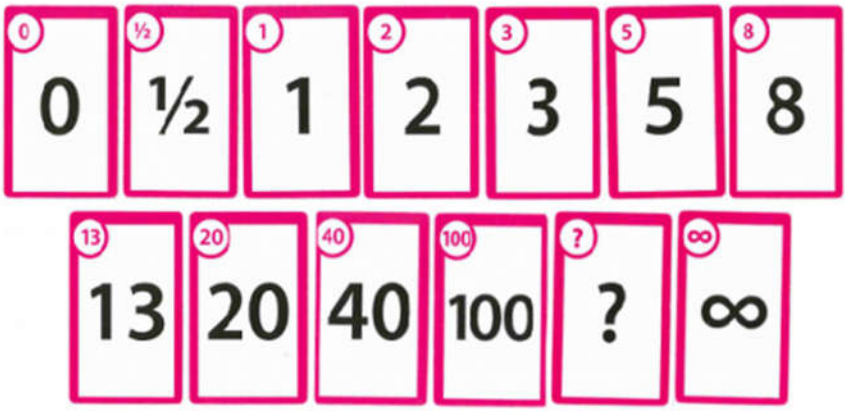
4.6.3 Planning Poker

El Planning Poker es una técnica que ayuda a estimar en consenso la cantidad de esfuerzo que requiere una historia de usuario para su construcción. Esta técnica hace uso de una baraja que cada persona del equipo de desarrollo debe tener, esta baraja está compuesta por cartas con una secuencia Fibonacci que incluye un cero, un interrogante y un infinito, estas dos últimas para representar incertidumbre o desconocimiento de la característica. La Figura 4-8 muestra un ejemplo de la baraja [69].

El proceso de estimación es bastante simple y se prosigue de la siguiente manera:

- Un moderador, que no jugará, preside la reunión. En este caso el SCRUM Master y asesorado por el Product Owner que conoce las necesidades y puede ayudar a eliminar incertidumbres.
- Del Product Backlog se van tomando las historias de acuerdo a su priorización.
- El Product Owner y algún miembro del equipo con más claridad sobre la historia dan una breve introducción. Los demás miembros del equipo pueden ir haciendo preguntas para dar más claridad de la historia, de sus criterios de aceptación y de sus escenarios.
- Cada persona del equipo de trabajo pone boca abajo una carta que represente la estimación.
- Todos muestran sus cartas de forma simultánea.
- A las personas con las estimaciones más altas y más bajas se les da tiempo para sustentar su estimación y se realiza una discusión al respecto supervisada por el SCRUM Master.
- Se repite el proceso de estimación hasta que se lleve a un consenso y se asigne al desarrollador o pareja de desarrollo que se va a encargar se su implementación.
- Se puede utilizar un reloj de arena o cronometro para ayudar a que el debate sea estructurado.
- Las historias de usuario que terminen con valores muy altos de estimación debes ser analizadas en una posterior reunión con el Product Owner porque pueden ser historias épicas que deben ser divididas, de igual manera con aquellas que quedaron con incertidumbre o no claras. Esto porque posiblemente el Product Owner deba invitar a un Stakeholder clave que ayude a darle entendimiento a la historia.

Figura 4-8: Ejemplo planning poker [69]



Esta técnica se alinea con los principios ágiles de trabajo en equipo y colaborativo, debido a que la estimación del proyecto se realiza en conjunto con todos los miembros del equipo y no cae en la responsabilidad de un gerente de proyecto, además, estas estimaciones tienen a ser menos optimistas y más precisas que al hacerlas de una forma tradicional basándose en hora/hombre.

Hay otras técnicas además del Planning Poker que se pueden usar como reemplazo o complemento a esta técnica, algunas de ellas pueden ser:

- Por talla de camisas usando las letras de XL hasta XS.
- Valoración en masa relativa, en donde se agrupan historias que estén relacionadas entre sí y se les asigna una misma estimación en esfuerzo.

4.6.4 Refactoring

La re-factorización o refactoring [70] es un proceso que ayuda a clarificar y simplificar el diseño de código existente, sin cambiar su comportamiento, comúnmente llamado limpiar código. Como vimos en la sección 4.6.1 este debe ser parte del proceso de desarrollo de software, en donde se debe hacer refactorización al código que ya haya cumplido con su prueba unitaria y nuevamente realizar la prueba para comprobar que el comportamiento no se haya alterado.

Su objetivo no es arreglar errores, sino el de mejorar la facilidad de comprensión del código por parte de todos los miembros del equipo, eliminar código muerto o sobrante y facilitar su mantenimiento futuro.

4.6.5 Mad, Sad, Glad

Mad, Sad, Glad es una técnica de retrospectiva [64] [71] que alienta a los miembros del equipo de trabajo a pensar sobre sus emociones relacionadas con el proyecto en curso. Es una técnica fácil de aprender, de explicar y ejecutar.

Está técnica por lo general se emplea para destacar los aspectos positivos (glad) como los negativos (mad y sad), haciendo que el equipo piense y exprese como se siente con los hechos que ocurrieron durante un sprint o iteración.

La técnica se centra en un tablero Kanban divido en tres secciones: Mad para expresar las situaciones que te hicieron sentir mal o enojado y que no deberían volver a suceder, Sad para expresar las situaciones que te hicieron sentir triste, situaciones que se deben mejorar porque no dieron los resultados esperados y Glad para aquellas situaciones que se hicieron sentir muy bien, muy contento y que se deben seguir haciendo.

Estos sentimientos son expresados en notas y ubicadas en una de las tres columnas del tablero, al final entre todos y con ayuda del SCRUM Master se analizan las notas, se agrupan en aquellas que estén relacionadas y se diseñan planes de mejoramiento para minimizar los aspectos negativos y maximizar los positivos.

A veces los equipos que no son muy maduros en enfoques ágiles y en estas técnicas, presentan poca confianza entre sus integrantes, dificultando que las personas expresen libremente sus ideas. Debido a eso existe una variante en donde el SCRUM Master posee una urna cerrada en donde los miembros del equipo de trabajo van depositando sus sentimientos durante el sprint e indicando si es un Mad, un Sad o un Glad y al final del sprint durante la reunión de retrospectiva se abre la urna y se empieza a ubicar cada nota en su columna correspondiente del tablero. Al principio las notas pueden ser impresas para proporcionar mayor anonimato y mientras el equipo adquiere mayor madurez y confianza.

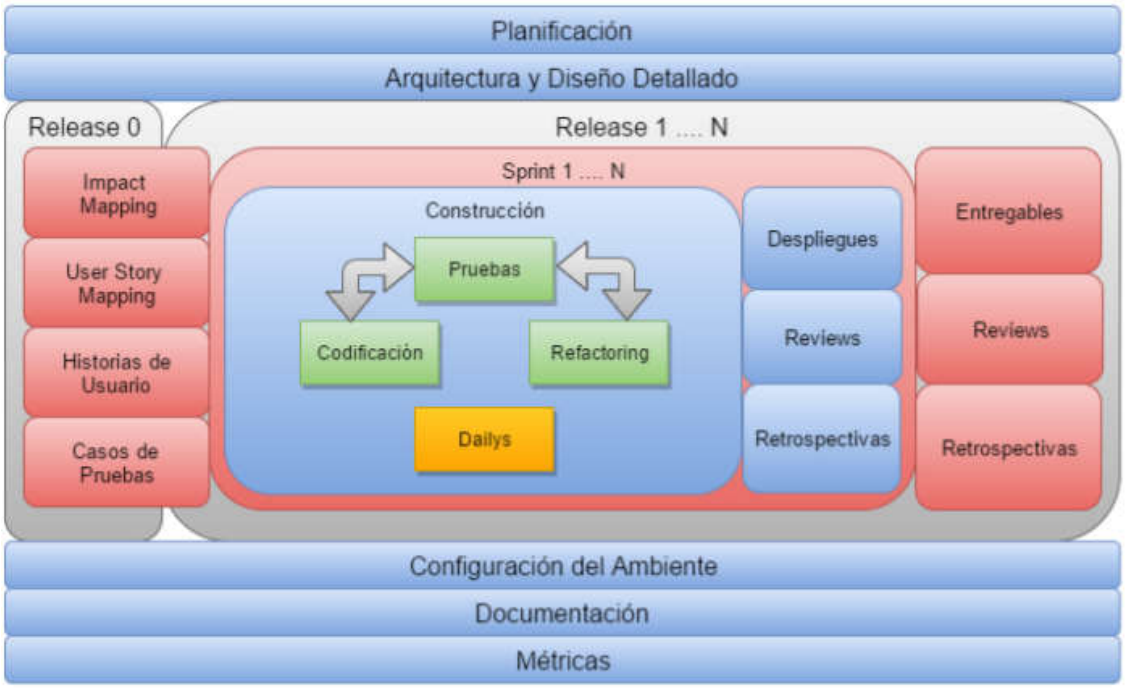
Dependiendo del equipo de trabajo se puede aplicar otro tipo de técnicas acorde a la forma de trabajar, madurez, nivel de confianza u otros aspectos que puedan influenciar. Algunas técnicas conocidas pueden ser The Wheel, The Sail Boat, entre otras. Todas estas técnicas se centran en identificar los mismos aspectos de Mad, Sad, Glad.

4.7 Ejecución del proceso

Al iniciar un nuevo proyecto de desarrollo de software usando AgileFM, se propone seguir un proceso bastante simple orientado en lo propuesto por los enfoques ágiles, técnicas y artefactos descritos anteriormente, en donde hay constante retroalimentación entre el cliente y el equipo de trabajo, con entregables tempranos que dan valor a los interesados y que sea flexible ante el cambio. En la Figura 4-9 se presenta un extracto de AgileFM donde presenta de manera detallada el proceso propuesto que lo compone.

Todo el proceso empieza con un *Release 0* en donde se busca tener un mejor entendimiento y tener una documentación inicial de las necesidades del cliente, realizando varias reuniones en donde esté involucrado el cliente, el equipo de desarrollo y el SCRUM Master. Durante las primeras reuniones se debe seleccionar el Product Owner quien será que guie el proyecto a una finalización satisfactoria.

Figura 4-9: Proceso desarrollo de software



El proceso empieza con el release cero, en donde comienza la construcción del Impact Mapping, del User Story Mapping, de las historias de usuario y los casos de prueba, se empieza a construir el Plan de Negocio, a diseñar la arquitectura propuesta y a construir o seleccionar un plan de métricas que ayuden a monitorear la salud del proyecto con respecto al desempeño y cumplimiento de lo estimado. Además, se debe acceder a la documentación de proyectos anteriores y de lecciones aprendidas, con el fin de direccionar correctamente el nuevo proyecto.

En cuanto al plan de métricas, se puede seguir un enfoque como GQM [72] (Goal, Question, Metric), en el cuál se define un modelo de medición en tres niveles: i) Goal o nivel conceptual, en donde se definen los objetivos, ii) question o nivel operacional, el

cual está compuesto por un conjunto de preguntas enfocadas a lograr los objetivos y iii) metric o nivel cuantitativo, el cual está formado por un conjunto de métricas asociadas a cada pregunta que buscan dar respuesta a la preguntas.

Todos estos artefactos irán evolucionando a medida que el proyecto avance, es decir, no son documentos o información estática y son flexibles a los cambios que se presenten por la naturaleza del proyecto, es por eso que el gráfico todos estos procesos son transversales a todos los releases y ciclo de vida del proyecto.

Los entregables de este release son de gran importancia, debido a que son la materia prima para la construcción del producto y representan una visión unificada de la solución para todos los involucrados (cliente, Product Owner, Equipo de Desarrollo, SCRUM Master, entre otros). Es importante durante las reuniones iniciales determinar que todos los involucrados tengan el mismo entendimiento de las necesidades, esto es vital para evitar reprocesos innecesarios y obtener entregables que cumplan con las expectativas del cliente.

Al finalizar este release, además de la documentación inicial de la necesidad y solución propuesta, se obtiene el cronograma y estimación de trabajo, separando el proyecto en releases y estos a su vez en sprints o iteraciones. Estos sprints se recomiendan que tengan una duración mínima de una semana y máxima de 3 semanas; sprints más largos ocasiona una separación entre el equipo de trabajo y el cliente, lo que se puede traducir en una pérdida de visión global del proyecto y se pierde flexibilidad en la adopción de los cambios. Los releases pueden tener tantos sprints como sea necesario, lo recomendable es que no sea más de 4 sprints, ya que en ese caso tal vez sea necesario separar el release. Al final de un sprint se realiza la entrega de funcionalidades de la solución que ofrecen valor al cliente, pero no están desplegadas para su uso en producción si no en modo de prueba o beta, mientras que los entregables de un release, es la sumatorio de los entregables de sus sprints que dan valor al cliente, pero en este caso se puede desplegar para producción.

Durante cada sprint, es importante destacar el proceso de desarrollo o construcción de la solución, en donde se propone seguir un proceso TDD o ATDD, partiendo de los casos de pruebas, construyendo primero las pruebas unitarias y luego se codifica la solución

que cumpla con la prueba, cuando esta es satisfactoria se pasa a hacer un refactoring para mejorar la visibilidad y optimización del código y nuevamente se garantiza que cumpla con la prueba. Este proceso de desarrollo va encaminado a dar calidad a la solución final como algo inherente al producto y no como un valor agregado como en algunos casos se ofrece en el mercado.

También es importante resaltar la realización de los daily's meeting, con el fin de que todos los miembros del equipo de trabajo este al día con el estado del proyecto y su salud, con el fin de detectar posibles problemas y buscar soluciones oportunamente.

4.8 Comparativo

Como se ha detallado a lo largo de este capítulo, AgileFM es una propuesta de gestión de proyectos de software que toma elementos de los métodos ágiles y del estándar ISO/IEC 29110. AgileFM se compone de seis elementos: Roles, herramientas, artefactos, ceremonias, técnicas y procesos y el proceso principal. Su objetivo, es cumplir con los principios agiles, a la vez que cumple con el formalismo y requisitos de gestión documental del estándar ISO/IEC 29110.

Las subsecciones siguientes realizan dos comparaciones entre AgileFM y el estándar ISO/IEC 29110. El primero de ellos está relacionado con las actividades que sigue el estándar y como estos están soportados por AgileFM. El segundo comparativo, se centra en la documentación requerida por el estándar ISO/IEC 29110 y como AgileFM propone dar solución a cada requisito documental.

4.8.1 Comparación de actividades

El estándar ISO/IEC 29110 propone dos procesos principales: Gestión de proyectos o Project Management (PM) el cual se enfoca en la iniciación y construcción de una planeación del proyecto, la ejecución de dicho plan, el control y valoración del proyecto y de la clausura al final del mismo. La Implementación de Software o también conocido como Software Implementation (SP) se enfoca en el análisis de los requerimientos, diseño de la arquitectura global y detallada, la construcción del software, diseño e implementación de los pruebas, integración continua y entregables. La Figura 4-10

muestra como es la relación entre estos dos procesos, mientras que la Figura 4-11 y la Figura 4-12 muestran como están compuestos estos procesos.

Figura 4-10: Relación entre la Gestión de Proyecto y la Implementación del Software [38]

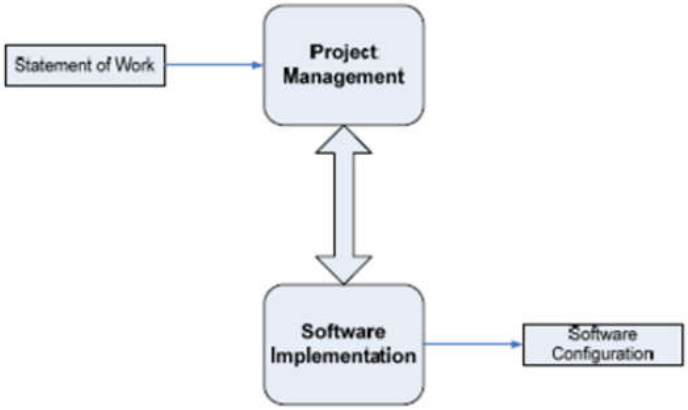


Figura 4-11: Proceso de Gestión de Proyectos [38]

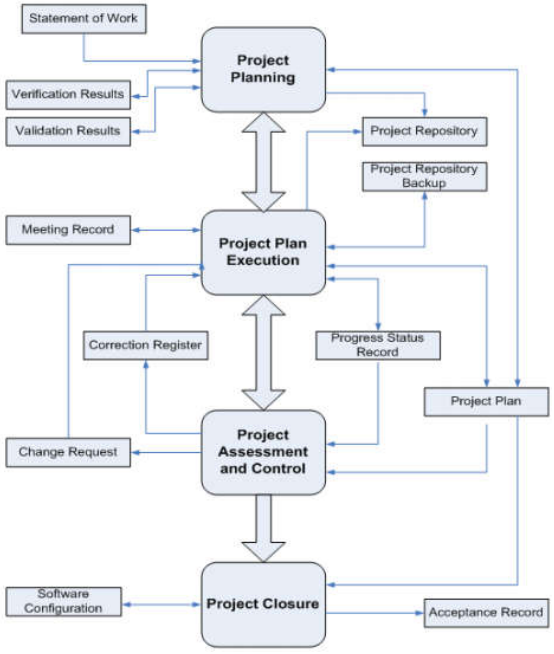
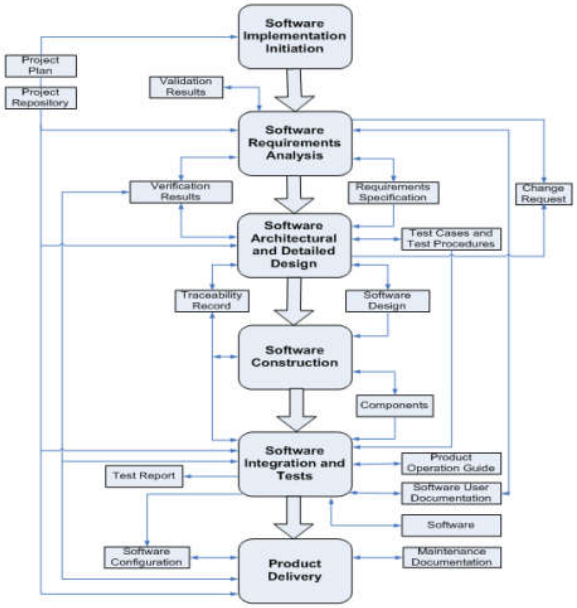


Figura 4-12: Proceso de Implementación del Software [38]



Teniendo en cuante los procesos y actividades descritos por el estándar ISO/IEC 29110 y AgileFM, es posible llevar a cabo una comparación que permita relacionar cómo los

procesos del estándar pueden ser soportados por los procesos y actividades que se presentan en el elemento de técnicas y procesos descripto en la sección 4.6 y en el proceso principal detallado en la sección 4.7 de AgileFM. De esta forma, la Tabla 4-7 realiza una comparación a alto nivel entre estos procesos y el planteado en AgileFM.

Tabla 4-7: Procesos ISO/IEC 29110 vs AgileFM

ISO/IEC 29110	AgileFM
Project Management	
Project Planning	• Planificación • Documentación
Project Plan Execution	
Project Assesment Control	• Planificación • Durante los procesos de reviews de los releases y sprints. • Métricas
Project Closure	• Actividades de Entregables, Review y Retrospectiva del último release.
Software Implementation	
Software Implementation Initiation	• Actividades de Impact Mapping, User Story Mapping, Historias de Usuario y Casos de Pruebas al inicio de cada release • Configuración del Ambiente
Software Requirement Analysis	
Software Arquitectural and Detailed Desing	Arquitectura y Diseño Detallado en cual es un capítulo del plan de negocio
Software Construction	Actividad de construcción que incluye: • Pruebas • Codificación • Refactoring
Software Integration and Test	
Product Delivery	• Actividad de despliegue en cada sprint • Actividad de entregable en cada release

De acuerdo a lo anterior, AgileFM logra cumplir a un alto nivel con la totalidad de los procesos que se presentan en el estándar ISO/IEC 29110, pudiendo llevar un proceso ágil formal compatible con el estándar.

4.8.2 Comparación de documentos requeridos

Por otra parte, de acuerdo al estándar ISO/IEC 29110, hay un conjunto de documentos que se deben crear y mantener durante la ejecución de un proyecto de software. En la Tabla 4-8 se detallan estos documentos con su contraparte en la propuesta dada.

Tabla 4-8: Documentación ISO/IEC 29110 vs AgileFM

ISO/IEC 29110	AgileFM
Registro de Aceptación	Repositorio de Gestión de Cambios
Solicitud de Cambios	
Registro de Corrección	
Documento de Mantenimiento	Repositorio Documentación Solución, Herramientas de Mantenimiento e Integración

Registro de Reuniones	Repositorio Documentación Solución • Registro fotográfico de las ceremonias. • Actualización del User Story Mapping e Historias de Usuario.
Guía de Operación del producto	Repositorio Documentación Solución
Registro de Estado del Progreso	Product Backlog, Tableros Kanban.
Plan del Proyecto	Plan de Negocio, Release Planning
Repositorio del Proyecto	Herramientas de Mantenimiento e Integración
Respaldo del Repositorio del Proyecto	
Especificación de Requisitos	Impact Mapping, User Story Mapping, Historias de Usuario
Software	Plan de Negocio, Software que se construye, ATDD/TDD
Componente de Software	
Configuración del Software	
Diseño de Software	Plan de Negocio
Documentación del Usuario del Software	Repositorio Documentación Solución
Declaración de Trabajo	Impact Mapping, Plan de Negocio, Historias de Usuario
Casos de Prueba y Procedimientos de Prueba	Historias de Usuario, Casos de Prueba
Reporte de Pruebas	Repositorio de Resultados de Pruebas
Registro de Trazabilidad	Herramientas de Gestión, Herramientas de Mantenimiento e Integración
Resultados Verificación	Repositorio de Resultados de Pruebas
Resultados Validación	

Teniendo en cuenta lo anterior, con AgileFM es posible llevar la totalidad de la documentación requerida por el estándar ISO/IEC 29110 de una forma más ágil y colaborativa, sin perder la formalidad y gestión requerida. Además, esto unido a la comparación realizada en la sección 4.8.1, permite que una MiPyME que adopte AgileFM como proceso para la gestión de proyectos de software, obtenga los siguientes beneficios:

- Formalizar su proceso de desarrollo.
- Cumplir con los lineamientos del estándar ISO/IEC 29110.
- Obtener una certificación del proceso de desarrollo y ser más competitivo en el mercado.
- Obtener un proceso más colaborativo entre todos los participantes de un proyecto.
- Promover la gestión del conocimiento dentro de la organización.

Capítulo 5. Estudio de Caso

"El experimentador que no sabe lo que está buscando, no comprenderá lo que encuentra", (Claude Bernard, Fisiólogo francés, 1813-1878)

El modelo propuesto ha sido validado mediante un estudio de caso para llevar a cabo la aplicación de AgileFM en un proyecto de desarrollo de software en el *área de desarrollo de software* de HMV Ingenieros Ltda., haciendo uso del método propuesto en la sección 2.4 del capítulo 2, el cual está basado en las propuestas de [11] [12] [13].

5. Estudio de caso

Se ha utilizado una mezcla de las metodologías Solución de Problemas Integrado/Método de Investigación en Ingeniería de Sistemas, marco de trabajo Action-Research-Evaluation y estudios de caso para evaluar el desempeño e impacto de AgileFM dentro de un equipo pequeño de desarrollo, de esta manera la investigación pasará por dos ciclos, el cual busca mejorar actividades del proceso orientadas a obtener mejores resultados en la utilización de AgileFM.

Las siguientes secciones describen el estudio de caso en términos la formulación de la pregunta de investigación, la descripción del estudio de caso, el diseño de la actividad académica, las unidades de análisis, el procedimiento de campo, la recolección de información, la ejecución de las actividades académicas, la intervención en el estudio de caso, el análisis de los resultados y el plan de validación y limitaciones.

5.1 Pregunta de investigación

Tomando como pregunta de investigación, la hipótesis de la sección 1.2 del capítulo 1:

Es posible implementar un modelo de gestión de proyectos de software ágil basado en la norma ISO/IEC 29110 desde un enfoque ágil que permita a la MiPyMEs mejorar el desempeño de su proceso de desarrollo, ser más competitivos en el mercado y lograr una certificación.

Se tiene como objetivo:

Demostrar los beneficios en tiempo, costo y satisfacción del cliente al aplicar AgileFM en un equipo de desarrollo pequeño o en una MiPyME de desarrollo de software.

El alcance de este estudio de caso se centra en la conceptualización y capacitación del equipo de trabajo en la utilización del modelo ágil propuesto, su implementación en el

proyecto seleccionado y analizar los resultados y experiencias obtenidas durante su ejecución, con el fin de realizar mejoramiento al proceso.

5.2 Selección del Estudio de caso

HMV Ingenieros Ltda. es una compañía con más de 55 años de experiencia y más de 1000 empleados, que ofrece servicios de ingeniería, proyectos EPC (Engineering, procurement, and construction management) de subestaciones, líneas de transmisión y de PCHs (Pequeñas Centrales Hidroeléctricas). La organización cuenta con un área de Gestión Tecnológica conformada por 9 personas, el área es la encargada principalmente del desarrollo de soluciones de software que apoyen los procesos misionales y de apoyo de la compañía. El área no cuenta con un método de gestión de proyectos formal que apoye la ejecución.

A finales de 2015, el área financiera de la organización presentó una necesidad con respecto a la gestión de los documentos de pago (facturas, anticipos, cuentas de cobro, entre otros), en la cual se requiere hacer un seguimiento de estos documentos, desde que llegan a las oficinas del *Centro de Documentación*, pasando por el personal autorizado de su aprobación, luego pasando al personal de contabilidad para causar el documento en la plataforma de planeación de recursos empresarial (ERP - Enterprise Resource Planning) y finalizando en tesorería con su pago y archivado final. Este nuevo proyecto fue tomado para poder a prueba a AgileFM.

5.3 Diseño de la actividad académica

Con el fin de implementar AgileFM, se desarrolló el siguiente plan de acción que se detalla en la Tabla 5-1:

Tabla 5-1: Plan de acción inicial

Actividad	Responsable	Dirigido a	Fechas Tentativas
Conceptualización del método al equipo de desarrollo a través de capacitación. Dos secciones por semana de dos horas y media.	Juan David Yepes González	Equipo de Desarrollo	Del 2016-01-04 al 2016-01-29
Definición y selección de los artefactos y herramientas de trabajo.	Juan David Yepes González y Equipo de Desarrollo	Equipo de Desarrollo	Del 2016-01-19 al 2016-01-29

Definición de mecanismos para la recolección y análisis de datos	Juan David Yepes González	No hubo necesidad de asignar rol	Del 2016-01-04 al 2016-01-08
Iniciación del proyecto e implementación del método propuesto	Juan David Yepes González y Equipo de Desarrollo	No hubo necesidad de asignar rol	Del 2016-02-01 al 2016-04-09
Primera etapa de análisis e interpretación de resultados y ajuste del plan de acción	Juan David Yepes González	No hubo necesidad de asignar rol	Del 2016-03-14 al 2016-03-18
Segunda etapa de análisis e interpretación de resultados	Juan David Yepes González	No hubo necesidad de asignar rol	Del 2016-05-02 al 2016-05-06

Las medidas usadas para guiar la pregunta de investigación en el estudio de caso son:

- El esfuerzo en tiempo y costo que tomó el seguir los procesos propuestos en AgileFM.
- El nivel de satisfacción del Product Owner con el proceso realizado.
- El nivel de satisfacción de los usuarios finales con el producto entregado.

5.4 Unidades de Análisis

Las unidades de análisis en el estudio de caso fueron las prácticas del modelo AgileFM.

5.5 Procedimiento de campo y recolección de información

El procedimiento de campo y la recolección de información del estudio de caso estuvo relacionado con los roles, herramientas, artefactos, ceremonias, técnicas, procesos y actividades propuestas en AgileFM descritos en el capítulo 4. Además, se construyó un plan de capacitaciones orientadas a las prácticas ágiles y a los elementos propuestos en AgileFM, con el fin soportar la implementación y adopción del modelo dentro del estudio de caso.

Para poder medir el nivel de adopción de AgileFM y el desempeño del proceso en la ejecución del proyecto y satisfacción del Product Owner, se diseñaron dos encuestas semi-estructuradas para ser diligenciadas durante cada ciclo por los miembros del equipo de desarrollo y el cliente. La encuesta Implementación y Adopción del Método Ágil Propuesto (ver anexo A) detalla la serie de preguntas que se le realizaron al equipo de

desarrollo, mientras que la encuesta de Satisfacción para el Product Owner (ver anexo B) fue entrega al Product Owner con el fin de evaluar los resultados finales del proyecto. Además, se consultó la información de consumo de tiempo y costos relacionados con el proyecto y se registró la información resultante de las retrospectivas. Toda esta información fue almacenada en listas de Excel para su posterior análisis a través de tablas dinámicas.

5.6 Intervención en el estudio de caso

5.6.1 Ejecución del primer ciclo

La ejecución del estudio de caso inició con el proceso de capacitación al equipo de desarrollo sobre el agilismo y sobre AgileFM. Se realizaron un total de 8 secciones con un total de 24 horas de capacitación, en los cuales se abordaron los siguientes temas:

- Introducción al agilismo.
- Roles del enfoque.
- Artefactos y herramientas.
- Técnicas y prácticas en el desarrollo de software.
- Introducción al proceso de AgileFM.
- Definición y seguimiento de métricas.

Las capacitaciones siguieron un proceso dinámico y participativo por parte del equipo de desarrollo, haciendo uso de actividades prácticas que llevan a interiorizar mejor los conceptos y a identificar las bondades de los enfoques ágiles. De igual forma, durante la ejecución del proyecto, personalmente realicé el papel de consultor ágil, con el fin de orientar al equipo de trabajo sobre dudas que se fueron presentando.

Antes de iniciar el proyecto, fue necesario primero definir que artefactos y herramientas que se emplearon, teniendo en cuenta que el área de desarrollo de la compañía ya contaba con varias herramientas que lo apoyan en su día a día. Al final de esta actividad se obtuvo un listado de artefactos y herramientas que se presentan en la Tabla 5-2:

Tabla 5-2: Artefactos y herramientas establecidas

Elemento Requerido	Elemento Seleccionado
Herramienta de Modelamiento	▪ StarUML ▪ Yaoqiang para modelos BPMN
Herramienta de Desarrollo	▪ Visual Studio 2015 Professional
Herramienta de Gestión	▪ Sitio colaborativo en SharePoint
Herramienta de Mantenimiento y de Integración	▪ Microsoft Team Foundation
Herramienta de Pruebas	▪ Vissual Studio 2015 Professional ▪ SonarQube
Herramienta de Aprendizaje	▪ Wikis y Blogs sobre SharePoint
Impact Mapping	▪ XMind
User Story Mapping	▪ Se realiza de forma física usando un tablero y Post-it
Historias de Usuario	▪ Se diseñó un sitio ágil sobre SharePoint
Casos de Prueba	▪ Se diseñó un sitio ágil sobre SharePoint
Product Backlog	▪ Se diseñó un sitio ágil sobre SharePoint
Sprint Backlog	▪ Se diseñó un sitio ágil sobre SharePoint
Release Planning	▪ Documento en Excel que se almacena en el sitio ágil
Burndown	▪ Documento en Excel que se almacena en el sitio ágil
Repositorio de Gestión de Cambios	▪ Sitio colaborativo en SharePoint
Repositorio de Lecciones Aprendidas	▪ Sitio colaborativo en SharePoint en donde se plasman todas las lecciones aprendidas de la empresa.
Repositorio de Resultados de Pruebas	▪ Sitio colaborativo en SharePoint
Tableros Kanban	▪ Se hacen uso de tableros físicos y virtuales a través de Trello

Posteriormente, se definieron las personas que iban a desempeñar los roles específicos propuestos en AgileFM. La Tabla 5-3 presenta el listado de personas y el rol a desempeñar:

Tabla 5-3: Asignación de roles en el proyecto

Empleado o Grupo	Cargo	Rol
Jorge Luis Restrepo Vélez	Coordinador Financiero	Product Owner
Jaime Walter Correo Cosme	Ingeniero Senior	SCRUM Master
Juan David Yepes González	Ingeniero de Diseño	Equipo de Desarrollo – Arquitecto
Jamyth Agudelo Ariza	Auxiliar de Desarrollo	Equipo de Desarrollo – Desarrollador y UX
Francy Catalina Bernal Quiroz	Auxiliar de Desarrollo	Equipo de Desarrollo – Desarrollador
Cristian Felipe Jaramillo Medina	Auxiliar de Desarrollo	Equipo de Desarrollo – Desarrollador
Andrés Mauricio Ramírez Ramírez	Analista de Sistemas	Equipo de Desarrollo – Tester
Personal de Contabilidad	Auxiliares Contables	Stakeholder Clave
Personal de Tesorería	Analistas de Tesorería	Stakeholder Clave
Personal de Compras	Coordinadores de Suministros	Stakeholder Clave

5.6.2 Análisis de resultados del primer ciclo

En la mitad del proyecto se aplicaron las respectivas encuestas a los involucrados del proyecto y se juntó con la información obtenida de las retrospectivas, además, se realizó una reunión extraordinaria con el equipo de desarrollo para hablar acerca del desarrollo

de esta primera etapa. La Figura 5-1 y Figura 5-2 muestran los resultados obtenidos al realizar la encuesta al equipo de desarrollo.

Figura 5-1: Resultados pregunta 1 anexo A

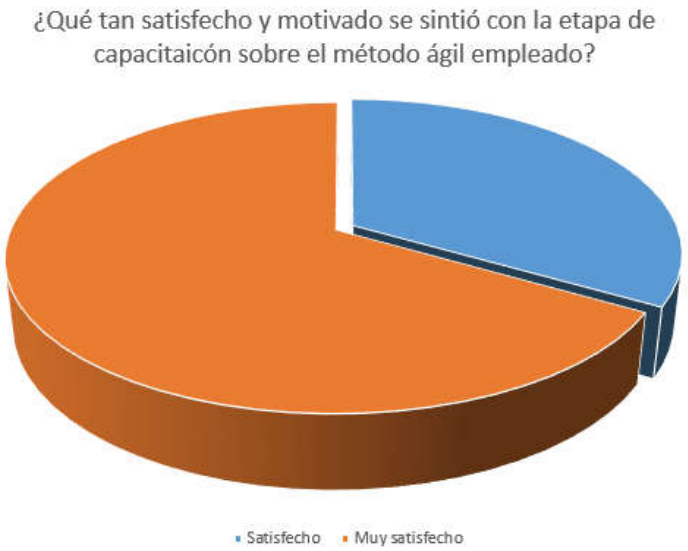
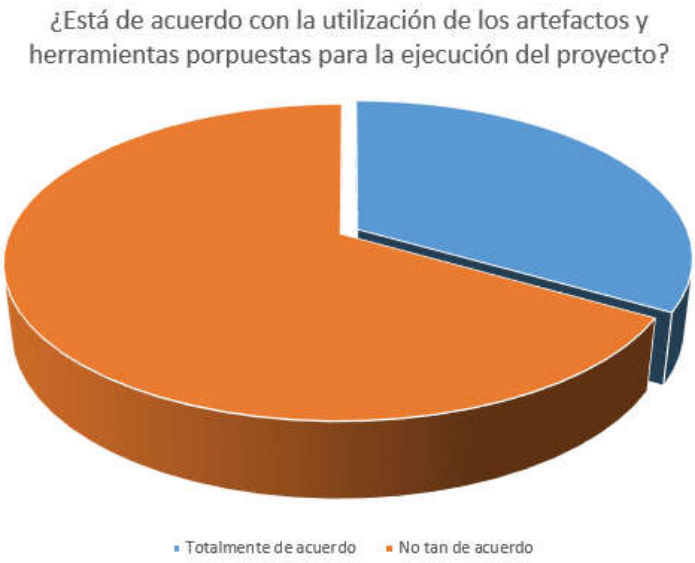


Figura 5-2: Resultados pregunta 2 anexo A



Del análisis de esta información se obtuvieron los siguientes resultados:

- El equipo de trabajo quedo muy satisfecho con las secciones de capacitación, ellos manifestaron que este tipo de actividades generan más empoderamiento hacia la empresa y aumenta la motivación en las personas. Además, se generó una cultura de

realizar capacitaciones sobre diversos temas relacionados con el desarrollo de software y la gestión ágil, donde el responsable de la charla varíe dentro del equipo.

- Con relación a la utilización de artefactos y herramientas de apoyo, se presentaron varios inconvenientes desde el inicio, los cuales fueron detectados tanto en los daily's meeting como en las reuniones de retrospectivas. Estos inconvenientes están relacionados con la falta de hábito en la utilización de los tableros kanban para el seguimiento y en la definición de los casos de prueba.
- Se pudo observar que se presentaron algunos problemas para entender la estimación basada en puntos de esfuerzo y la manera en la cual se determina la velocidad del equipo, esto, quizá es debido a la falta de métricas e información de proyectos pasados que ayudarán a determinar y utilizar fácilmente la utilización de estas técnicas. Además, el hecho de estar dentro de una compañía de ingeniería tradicional, ha condicionado el área de desarrollo a una estimación por horas/hombre y al estar fuertemente ligados a un cronograma inicial poco flexible. Por otro lado, la experiencia de realizar un *User Story Mapping* junto con el Product Owner, tuvo muy buena acogida y el sentimiento general fue que la identificación y definición de requisitos fue mucho más fácil.
- La utilización de prácticas como TDD/ATDD durante este primer ciclo fue bastante difícil, la falta de experiencia y la costumbre de codificar de forma tradicional sin preocuparse por las pruebas, son determinantes en el cambio cultural y en la curva de aprendizaje. Pero la utilización del refactoring fue asimilado rápidamente, ellos manifestaron que mejora la visualización del código, agiliza notablemente el entendimiento de otros segmentos del código realizado por otra persona y de esta forma se puede intervenir más rápidamente sobre a cambios o un problema, además se evita la necesidad de que él responsable del código deba explicar su funcionamiento.
- Con relación a las ceremonias, se presentó un rechazo inicial tanto por parte del equipo de desarrollo como por el Product Owner, debido principalmente a que veían que eran muchas reuniones y que esto iba a afectar los tiempos de entrega, pero conforme se iban realizando, empezaron a notar de manera positiva que las ceremonias estaban ayudando a agilizar el desarrollo del proyecto y a eliminar las dudas e inconvenientes que se iban presentando. A través de las daily's meetings, el SCRUM Master tuvo los mecanismos para estar al tanto del proyecto y de la situación

del equipo de desarrollo, de esta forma, pudo ayudar más activamente mejorando la velocidad del equipo y su concentración en el proyecto.

- Otro aspecto en el cual el equipo de desarrollo debió integrar en su proceso, fue el de llevar una documentación más activa de los sucesos del proyecto que permitiera mejorar la documentación de lecciones aprendidas, el registro del resultado de las pruebas y el registro de los cambios solicitados por el cliente, todo lo anterior con el objetivo de llevar una gestión del conocimiento en el área más activa. La mayoría de los integrantes del equipo de desarrollo, no estaba acostumbrado a llevar un registro de estos eventos, sin embargo, la expectativa de poder usar esta información a futuro para agilizar el desarrollo de otros productos de software y así evitar problemas recurrentes fue mayor que su rechazo. Por otro lado, el Product Owner vio con muy buenos ojos que esta documentación estuviera disponible y fuera de fácil acceso, esto para generar un contexto de transparencia en el desarrollo del proyecto.
- Para el Product Owner fue la primera vez que jugada un rol tan importante en un proyecto de desarrollo de software, el hecho de ser el responsable de definir y direccionar los objetivos y alcance del proyecto, ayudaron a que sintiera que realmente tenía el control del proyecto y que se estaba desarrollando bajo los parámetros definidos.
- En general el equipo de desarrollo se encuentra muy motivado con el nuevo modelo y manifestaron la intensión de continuar utilizándolo en el futuro.

5.6.3 Retroalimentación en el segundo ciclo

Con base en las encuestas, entrevistas, retrospectivas y análisis anterior, se plantearon los siguientes objetivos específicos para el segundo ciclo de investigación del proyecto:

- Realizar charlas semanales durante el resto del proyecto orientadas a reforzar los conceptos alrededor de AgileFM.
- Realizar un taller de programación a comienzos del segundo ciclo del proyecto, orientado al uso de TDD/ATDD con el fin de fortalecer estas habilidades.

5.6.4 Plan de acción para el segundo ciclo

El plan de acción inicial no tuvo alteraciones en sus actividades existentes, pero se adicionaron dos actividades nuevas que buscaron reforzar el trabajo con el modelo ágil propuesto. La Tabla 5-4 presenta estas actividades:

Tabla 5-4: Nuevas actividades adicionadas al plan de acción

Actividad	Responsable	Dirigido a	Fechas Tentativas
Capacitaciones semanales sobre agilismo	Juan David Yepes González	Equipo de Desarrollo	Viernes de cada semana
Taller sobre técnicas de desarrollo – TDD/ATDD	Juan David Yepes	Equipo de Desarrollo	2016-03-27

5.6.5 Ejecución del segundo ciclo

Se siguió con la ejecución del plan de acción incluyendo las dos nuevas actividades, durante las capacitaciones semanales se propusieron varios ejercicios que ayudaron a interiorizar las técnicas y buenas practicas ágiles, además, los miembros del equipo de desarrollo trajeron propuestas de ejercicios que ellos mismos habían consultado.

El taller de desarrollo se realizó bajo el concepto de Coding Dojo, el cual se basa en reunir un grupo de desarrolladores y trabajar en un reto de programación durante un tiempo determinado. Esto con el fin de adquirir de forma más divertida y colaborativa, distintas técnicas de programación que el equipo requiere [73].

5.6.6 Análisis de resultados del segundo ciclo

Nuevamente, el equipo de desarrollo y el Product Owner diligenciaron las encuestas con sus apreciaciones de este segundo ciclo, además se llevaron a cabo una serie de entrevistas informales para determinar el sentir individual de las personas del proyecto. Luego, esta información de unió con las experiencias obtenidas de las retrospectivas y se obtuvo el siguiente análisis:

- El equipo de desarrollo se muestra optimista con adoptar completamente AgileFM dentro del proceso de desarrollo del área de Gestión Tecnológica. Ellos reconocen que deben seguir estudiando y mejorando en la utilización del proceso, pero

empiezan a reconocer los beneficios en rendimiento, organización y satisfacción en comparación al que les trae el cliente o impone su propio modelo.

- El equipo sigue estudiando sobre las distintas técnicas de desarrollo de software y se han propuesto continuar con los Coding Dojo una vez por mes y participar de los eventos de desarrolladores que se llevan en la ciudad a través de la corporación Ruta N.
- El Product Owner manifestó que el área financiera tiene otros proyectos de mejora de procesos en el cual están involucrados el desarrollo de un software específico al proceso y desea que se siga la misma metodología de trabajo, esto gracias a los buenos resultados obtenidos.

Adicionalmente, se consultó la información relacionada con consumo de tiempo y costos del proyecto y se comparó con un proyecto anterior de complejidad similar, para tener una idea inicial de los beneficios de AgileFM.

La Tabla 5-5 presenta los consumos de tiempo y de costo del proyecto del estudio de caso. El proyecto se logró cumplir en el tiempo estimado y se obtuvo una buena respuesta de aceptación del producto final por parte del Product Owner y de los usuarios finales.

Tabla 5-5: Consumo de tiempo y costo del proyecto

Año	Mes	Horas	Costos
2016	Enero	318	$ 4,862,837
2016	Febrero	115	$ 1,758,573
2016	Marzo	318.5	$ 4,870,483
2016	Abril	150	$ 2,293,791
Total		**901.5**	**$ 13,785,685**

Por otra parte, la Tabla 5-6 presenta el mismo informe para un proyecto de construcción de un módulo de gestión de proveedores llevado a cabo durante el 2015. Este proceso presento varios reprocesos y cambios de alcance. Esto se evidencia en los consumos de mayo y junio de 2015. Posiblemente, el haber utilizado AgileFM como modelo para la gestión del proyecto, hubiera permitido ser más proactivos y flexibles al cambio en etapas

tempranas del proyecto y no a finales, cuando se presentaron los cambios por parte del cliente.

Tabla 5-6: Consumo de tiempo y costo del proyecto de gestión de proveedores

Año	Mes	Horas	Costos
2015	Febrero	208.5	$ 1,953,209
2015	Marzo	208	$ 3,842,598
2015	Abril	144.5	$ 586,062
2015	Mayo	275.5	$ 5,942,034
2015	Junio	201	$ 3,226,428
Total		**1038**	**$ 15,550,331**

Teniendo en cuenta todo lo anterior, se puede decir que AgileFM cumple con mejorar el desempeño de un equipo de desarrollo con poca experiencia en agilismo, obteniendo beneficios en tiempo y costo, además de mejorar la comunicación con el cliente y tenerlo satisfecho. También, se gana beneficios a nivel de gestión de conocimiento, en mejoramiento del trabajo en equipo, en el incremento de la confianza entre los miembros del equipo y mejora la imagen del área frente a la organización.

5.6.7 Contexto con el estándar ISO/IEC 29110

Cabe destacar que hasta este punto, lo que se ha realizado es una implementación de AgileFM dentro del equipo de desarrollo para el proyecto del estudio de caso y no se ha implementado ni abordado ningún aspecto relacionado con el estándar ISO/IEC 29110, tampoco se ha planteado realizar una certificación al proceso a corto plazo. Sin embargo, la empresa ha manifestado su interés en el largo plazo de llevar a cabo la certificación de su proceso. Para dejar constancia de lo implementado desde el modelo AgileFM y su relación con el estándar ISO/IEC 29110.

La Tabla 5-7 presenta las actividades de los procesos principales del estándar que se lograron abordar aplicando AgileFM al proyecto y que dan soporte al cumplimiento de algunas prácticas según la norma ISO/IEC 29110. La Tabla 5-7 presenta como pueden ser soportadas actualmente en la empresa HMV Ingenieros con la implementación de las practicas sugeridas desde AgileFM. Sin embargo, se debe tener en cuenta, que este es el primer acercamiento a AgileFM y que la ejecución de las distintas actividades del

modelo no se cumple en totalidad. La actividad de métricas no se realizó debido en parte al desconocimiento del equipo de trabajo de cómo llevar un proceso de medición a un proyecto más allá del uso de artefactos como el Release Planning y el Burndown. La actividad de la construcción de la solución no se realizó en su totalidad basado en la técnica de TDD/ATDD, ya que esta requiere un nivel mayor de madurez y destreza técnica, que se irán consiguiendo con los Coding Dojo y trabajar en nuevos proyectos.

Tabla 5-7: Procesos del estándar ISO/IEC 29110 y AgileFM en HMV

ISO/IEC 29110	AgileFM en HMV Ingenieros
Project Management	
Project Planning	▪ Se realizaron las actividades de planificación y documentación, aunque todavía se debe mejorar en la estimación y construcción del release planning
Project Plan Execution	
Project Assesment Control	▪ Planificación ▪ Durante los procesos de reviews de los releases y sprints.
Project Closure	▪ Actividades de Entregables, Review y Retrospectiva del último release.
Software Implementation	
Software Implementation Initiation	▪ Se realizaron las actividades de Impact Mapping, User Story Mapping, Historias de Usuario, pero la actividad para definir los casos de prueba no se realizó en su totalidad. ▪ La actividad de configuración del ambiente se obvió debido a que ya contaba con el ambiente disponible.
Software Requirement Analysis	
Software Arquitectural and Detailed Desing	▪ La actividad de arquitectura y diseño detallado se realizó con la estructura propuesta de AgileFM, pero se debe mejorar las habilidades de diagramación para obtener diseños más entendibles por el equipo de trabajo.
Software Construction	Actividad de construcción: • Pruebas • Codificación • Refactoring
Software Integration and Test	
Product Delivery	• Actividad de despliegue en cada sprint • Actividad de entregable en cada release

Por otro lado, de acuerdo a lo propuesto por AgileFM, se creó un sitio web sobre la plataforma SharePoint 2013, este sitio alberga una serie de bibliotecas de documentos, listas de registros, blogs, wikis y gestión de tareas, que ayudaron a llevar la documentación del proyecto de forma más ágil, colaborativa y organizada. La Tabla 5-8 detalla los documentos requeridos por el estándar ISO/IEC 29110 que fueron soportados por el sitio web en SharePoint 2013.

Tabla 5-8: Documentación del estándar ISO/IEC 29110 y AgileFM en HMV

ISO/IEC 29110	AgileFM en HMV Ingenieros
Registro de Aceptación	Lista de registro en el sitio web de SharePoint para realizar la gestión de cambios al proyecto
Solicitud de Cambios	
Registro de Corrección	
Documento de Mantenimiento	Wiki en el sitio web de SharePoint donde se construyó la

	documentación tanto a nivel de componentes de software, como el mismo manual de usuario final. Esta se denominó Wiki de documentación del producto
Registro de Reuniones	Sobre una biblioteca de documentos del sitio web de SharePoint, se almacenó los documentos de las actas de reuniones basados en una plantilla corporativa.
Guía de Operación del producto	Wiki de documentación del producto
Registro de Estado del Progreso	Los sitios de SharePoint cuentan con listas de gestión de tareas y actividades, la cual se adaptó para que funcionara como un product backlog y tablero kanban. Aunque todavía se tiene que mejorar en esta adaptación
Plan del Proyecto	Se creó una wiki para documentar todo el plan de negocio del proyecto y se usó la lista de tareas para el Release Planning, aprovechando que ésta lista cuenta con vistas en calendario y en diagrama de Gantt.
Repositorio del Proyecto	El área de gestión tecnológica cuenta con Team Foundation Server para llevar el control de versionado y trazabilidad del código realizado.
Respaldo del Repositorio del Proyecto	
Especificación de Requisitos	El Impact Mapping se realizó con XMind y se almacenó en la biblioteca de documentos del sitio en SharePoint. El User Story Mapping se realizó de forma física y se realizaron capturas fotográficas que se almacenaron en la biblioteca de documentos. Para las historias de usuario se creó una lista de registro en el sitio en base a la estructura propuesta por AgileFM.
Software	Wiki del Plan de Negocio
Componente de Software	
Configuración del Software	
Diseño de Software	Wiki del Plan de Negocio
Documentación del Usuario del Software	Wiki de documentación del producto
Declaración de Trabajo	Wiki del Plan de Negocio, y lista de registro de historias de usuario
Casos de Prueba y Procedimientos de Prueba	Aunque en la estructura de las historias de usuario se propone la definición de los escenarios para la construcción de los casos de prueba, esta información no fue construida en su totalidad, y solo se abordó alrededor de un 10% de las historias de usuario.
Reporte de Pruebas	Se contó con el registro de las pruebas unitarios que genera el IDE, pero por el momento no se integraron con el sitio web en SharePoint, además. Tampoco se llevó un registro de otras pruebas realizadas como de usabilidad y de rendimiento.
Registro de Trazabilidad	Team Foundation Server provee manejo de versionado sobre el código fuente y a través del sitio web en SharePoint se lleva el control del versionado y la trazabilidad de todas las bibliotecas de documentos y listas de registros.
Resultados Verificación	Al igual que con el reporte de pruebas, no se realizó un registro de los resultados obtenidos al realizar la validación y verificación de los componentes y entregables del producto.
Resultados Validación	

En la Tabla 5-8 se puede apreciar, que gran parte de la documentación requerida por el estándar ISO/IEC 29110 está soportada de alguna manera con la implementación de AgileFM en el estudio de caso, pero todavía hay mucho margen de mejora y algunos documentos que quedaron por fuera en este primer acercamiento, como fueron los registros de los resultados relacionados con las pruebas, validación y verificación del producto. También, hay falencias en el registro de los casos de prueba, en parte por la falta de costumbre en la realización de esta actividad, pero que poco a poco con la

ejecución de nuevos proyectos bajo AgileFM se espera que este hábito se adquiera y a la vez apalanque la utilización de TDD/ATDD como técnica en la construcción de software. Si se quiere como organización y como equipo de desarrollo alcanzar una certificación sobre el estándar ISO/IEC 29110, se deben seguir haciendo mejoras continuas en la implementación y adopción de AgileFM, esto a través de capacitaciones regulares sobre el modelo y su utilización en los próximos proyectos que se presenten.

5.7 Plan de validación y limitaciones del estudio de caso

Con el objetivo de direccionar el estudio de caso lo mejor posible, se consideraron varios factores descritos a continuación:

- Para la construcción de la validación se usaron varias fuentes de evidencia como participación observativa, encuestas semi-estructuradas y abiertas, entrevistas con los involucrados en el proyecto, información recolectada de las retrospectivas e información de consumos de tiempo y costos asociados al proyecto.
- Se ha podido determinar que la implementación de AgileFM para el estudio de caso ha permitido satisfacer el objetivo y alcance propuesto.
- En cuanto a la validez interna, ha sido posible determinar que la aplicación de las encuestas en el estudio de casos nos ha permitido satisfacer las necesidades identificadas. También se ha tenido en cuenta los beneficios reportados por el estudio de caso.
- Para la validez externa, el estudio de caso fue apoyado por un asesor. Las observaciones y lecciones aprendidas fueron recogidas con el objetivo de perfeccionar el protocolo y procedimiento de campo, esto con el fin de ser capaces de realizar una replicación en estudios de casos futuros.

Las limitaciones consideradas para el estudio de caso son:

- No se cuenta con métricas de proyectos pasados para realizar una mejor comparación entre la implementación de AgileFM y el método utilizado en proyectos anteriores.

- No se planean acercamientos al estándar ISO/IEC 29110 a corto plazo, y aunque existe un interés a largo plazo de perseguir una certificación, se establece una relación con los elementos implementados hasta el momento descritos en AgileFM.
- Dado que la población fue muy pequeña, éste no es una muestra representativa y de alguna manera afecta el poder generalizar los resultados. Por lo tanto, se hace necesario replicar el estudio de caso en un número de estudios de casos más grande que permita asegurar un análisis adecuado y subsecuentemente llevar a cabo la generalización de los resultados.
- El sesgo en el estudio de caso con relación a: (i) la falta de conocimientos en el uso de metodologías ágiles y el modelo propuesto y (ii) la realización de las actividades se pudo ver afectada por la observación y monitoreo continuo. La observación de los participantes puede resultar ventajoso cuando se trata de obtener cierta información que sería imposible de obtener de otra manera. Sin embargo, el principal problema con esto es la posibilidad de sesgo producido.

Capítulo 6. Conclusiones y recomendaciones

"La felicidad total del hombre consiste en disfrutar de la estimación de los demás", (Blaise Pascal, Científico, filósofo y escritor francés, 1623-1662)

La elaboración de este trabajo ha levantado una serie de experiencias y de lecciones aprendidas que serán plasmadas en las subsecciones siguientes a través de conclusiones y recomendaciones, tanto a nivel del proceso de desarrollo de software como a nivel de investigación.

6. Conclusiones y Recomendaciones

6.1 Conclusiones

Cuando se está llevando a cabo un proyecto de investigación en el área relacionada a la ingeniería de software, es realmente importante darle un vistazo a los enfoques y metodologías diseñadas para facilitar y cumplir dicho propósito. Teniendo en cuenta que la ingeniería de software es un área de conocimiento relativamente joven con relación a las ciencias tradicionales, no siempre los enfoques investigativos existentes ofrecen la flexibilidad que se requiere en el mundo de la tecnología. Gracias a este trabajo, se han podido conocer nuevos paradigmas que apoyan los procesos de investigación que ofrecen buenos resultados. Asimismo, fue posible conocer la formalidad que demandan.

Gracias a la revisión sistemática realizada al estado actual del agilismo en la academia y la industria, se pudo identificar algunos factores relacionados con el éxito y/o fracaso de un proyecto de software, estos factores afectan de manera considerable el día a día de una empresa de desarrollo de software.

Entre los factores más determinantes identificados, se encontró el relacionado con el primer acercamiento al agilismo de un equipo de trabajo o empresa desarrolladora de software. Por lo general, los equipos de trabajo quedan altamente estimulados por las supuestas ventajas que ofrecen estos nuevos enfoques, en parte, por culpa de estos nuevos equipos o empresas que se encargan de difundir los enfoques ágiles como la solución definitiva para las empresas desarrolladoras, muchas veces se mal interpretan los valores y principios en los que se basan estos métodos, ocasionando experiencias negativas en los equipos de trabajo.

Por otra parte, es común observar que las empresas que ofrecen servicios de coaching, no tengan una alta experiencia real en la construcción de software, esto se puede

visualizar en la forma en que realizan los acompañamientos y entrenamientos de los equipos de desarrollo, se enfocan principalmente en los procesos de gestión y obvian procesos tan importantes y críticos como por ejemplo: las etapas de iniciación para la toma de requisitos y diseño arquitectónico, e incluso obvian procesos y estrategias importantes en la etapa de construcción.

A lo largo de la investigación también se reveló que los enfoques ágiles no son aplicables a todos los proyectos de desarrollo de software, esto se debe a la misma naturaleza del proyecto y del producto final. Para ponerlo en contexto, existen proyecto de muy baja complejidad e impacto, los cuales fácilmente pueden ser abordados con un método en cascada, ya que no amerita generar iteraciones y entregables continuos. Otro caso puede ser el desarrollo de software muy especializado que requiere altos niveles de calidad y seguridad (Software Safety), en los cuales la falla de un sistema puede acarrear la pérdida de cuantiosas sumas de dinero, el cierre de empresas o atentar en contra de la vida de seres humanos; estos proyectos requieren enfoques mucho más estrictos con métodos de especificación de requisitos, de documentación y de pruebas mucho más formales; en donde un enfoque ágil puede que no ofrezca los mecanismos adecuados para su ejecución.

Para proyectos de desarrollo de software de una complejidad moderada o alta, no relacionado con proyectos de software safety, los enfoques ágiles son una muy buena alternativa para abordar estos proyectos; pero es importante recordar que como toda nueva metodología requiere una curva de aprendizaje y de adopción, al igual que compromiso por parte de las personas involucradas, esto con el objetivo a mediano y largo plazo de obtener buenos resultados en los proyectos de software. De igual forma, es importante aclarar que el agilismo no tiene nombres propios, es decir, no es solo SCRUM, o eXtreme Programming; el agilismo es un conjunto de valores, principios, buenas prácticas, artefactos, herramientas, personas y procesos, trabajando en armonía de acuerdo a las necesidades del equipo de trabajo, de la empresa desarrolladora, del cliente y del producto final.

La investigación realizada, logra identificar los elementos claves de éxito que conforman los métodos ágiles existentes en su momento de implementación y ejecución, al igual de aquellas, situaciones y/o prácticas que entorpecen la adopción de un método ágil dentro

de una organización. Por otro lado, se estudió a fondo la propuesta que presenta el estándar ISO/IEC 29110 como una alternativa a la gestión de proyectos de software, pero que no es ajena a trabajar de la mano con otros enfoques, por lo tanto, se identifican sus procesos y actividades propuestos y sus requisitos a nivel de documentación y de trazabilidad de un proyecto. De esta forma, se construye una propuesta denominada AgileFM, en la cual se integra una serie de buenas prácticas, artefactos, técnicas y procesos agiles, alineados a una gestión documental y del conocimiento, donde se promueve el trabajo colaborativo entre todos los actores involucrados de un proyecto, en la transparencia de la información y en el mejoramiento continuo de los procesos y personas.

6.2 Recomendaciones

Antes de iniciar cualquier proceso de investigación, es aconsejable realizar una revisión de los métodos investigativos que se estén usando, con el fin de evaluar cuál es el mejor camino a tomar de acuerdo al área de conocimiento que se está abordando. Realizar esta actividad inicialmente ofrecerá la línea base de un proceso de investigación, además de poder iniciar un plan de trabajo que ayude a obtener buenos resultados.

Para los equipos de trabajo o empresas desarrolladoras de software que desean adentrarse en el agilismo por primera vez, es importante que recuerden que este es un proceso largo, el cual requiere mucha dedicación y esfuerzo por parte de todos los miembros del equipo, además de estar muy seguros de que requieren el cambio para mejorar sus procesos internos. Antes que nada, es importante realizar un levantamiento de los procesos actuales y medir que tan eficientes son en cada una de las actividades que se realizan; esto ofrece una línea base con la cual se puede comparar a futuro con el nuevo proceso ágil implementado. Con el levantamiento de los procesos realizado, se debe realizar una comparación con el nuevo enfoque a institucionalizar, viendo que se cubra la totalidad de actividades, aunque también se puede dar el caso en que se encuentren actividades que no aportan valor y que se puedan omitir en el nuevo proceso. Esta actividad le dará vida al nuevo proceso de desarrollo y es aconsejable realizado en comunión con el equipo de trabajo.

Una buena práctica cuando se realiza un cambio tan drástico en los procesos de una empresa, es realizar por lo menos una vez al mes charlas o capacitaciones relacionadas con la nueva metodología, estas pueden ser impartidas por terceros con experiencia real, o por algún miembro del equipo de trabajo que tenga mayor experiencia. El objetivo es mantener al equipo motivado, actualizado e incluido en el proceso. Estas charlas deben centrarse en el trabajo en equipo, en la creatividad y en el crecimiento tanto personal como profesional, promoviendo el crecimiento técnico y social de los equipos de trabajo.

Es importante nuevamente recordar que el agilismo no es solo un enfoque con nombre propio, es un conjunto de buenas prácticas, técnicas, valores y principios adaptados al entorno de la empresa o del equipo de trabajo. Este trabajo propone un modelo que reúne varias herramientas, artefactos y conceptos empleados en el desarrollo de proyectos de software bajo un enfoque ágil. AgileFM es una propuesta genérica que puede ser adaptada a las necesidades del negocio o del proyecto en curso. Es importante que los equipos u organizaciones ágiles evolucionen constantemente, mejorando su proceso, mejorando técnicamente, mejorando como equipo e individuos y eliminado impedimentos; esto a la final marcara la diferencia entre el éxito o fracaso en un proyecto de software y AgileFM muestra ser una propuesta que tiene los elementos para ayudar a un equipo de trabajo y a una MiPyME lograr sus objetivos trazados en sus proyectos, a través de un proceso orientado a la gestión documental y del conocimiento, que motiva el trabajo colaborativo, tanto entre el equipo de desarrollo y el cliente y que además, apalanca el crecimiento de sus individuos, tanto social como profesionalmente.

6.3 Trabajo Futuro

En el mediano plazo, se pretende gestionar nuevos proyectos de software dentro de HMV Ingenieros Ltda. haciendo uso de AgileFM como modelo, de esta forma poder seguir obteniendo una constante retroalimentación de los elementos a mejorar dentro de la propuesta y, además, a largo plazo poder optar por una certificación sobre el estándar ISO/IEC 29110.

Asimismo, se espera poder realizar un acercamiento a MiPyMEs de desarrollo de software en el Valle de Aburrá que deseen realizar un salto a la gestión de proyectos agiles y estén dispuestos a adoptar AgileFM como una alternativa, esto ayudaría

enormemente en la validación y verificación del modelo para su mejoramiento continuo y apoyarlos en un proceso de certificación en el estándar ISO/IEC 29110 para aquellos interesados.

Por otra parte, existe un tema muy recurrente actualmente en el desarrollo software y es la Ingeniería de Líneas de Producto de Software (SPLE – Software Product Line Engineering), el cual cuenta con diversas investigaciones a nivel académico y marcos de trabajo propuestos. Por lo tanto, se puede guiar una investigación en la cual se pueda adaptar AgileFM a un marco de trabajo de SPLE y ganar elementos de gestión documental y del conocimiento, además del trabajo colaborativo dentro de un proyecto de líneas de producto.

Bibliografía

[1] I. y. T. Ministerio de Comercio, «Definición Tamaño Empresarial Micro, Pequeña, Mediana o Grande,» [En línea]. Available: http://www.mipymes.gov.co/publicaciones.php?id=2761.

[2] E. Commission, «Annual Report on European SMES 2012/2013,» 2013. [En línea]. Available: http://ec.europa.eu/enterprise/policies/sme/facts-figures-analysis/performance-review/files/supporting-documents/2013/annual-report-smes-2013_en.pdf. [Último acceso: 2015].

[3] Laitinen y R. Ward, «Software Engineering in the Small,» *Communications of the ACM,* vol. 43, nº 3, pp. 115-118, 2000.

[4] E. Programming, «Extreme Programming: A gentle introduction. Online:,» [En línea]. Available: http://www.extremeprogramming.org/.

[5] Scrum, «Scrum – Improving the Profession of Software Development,» [En línea]. Available: https://www.scrum.org/.

[6] Lean Enterprise Institute, Inc, «Lean Enterprise Institute,» 2000. [En línea]. Available: http://www.lean.org/. [Último acceso: Octubre 2015].

[7] Lean Kit, «Lean Kit,» [En línea]. Available: https://leankit.com/learn/kanban/kanban-board/. [Último acceso: Enero 2016].

[8] J. D. Yepes González, C. J. Pardo Calvache y O. S. Gómez Gómez, «Revisión Sistemática Acerca de la Implementación de Metodologías Ágiles y Otros Modelos en Micro, Pequeñas y Medianas Empresas,» *Revista Tecnológica ESPOL - RTE,* vol. 28, nº 5, pp. 464-479, 2015.

[9] C. Monsalve, M. Villavicencio y S. Coque, «Revista Tecnológica ESPOL,» 2015. [En línea]. Available: http://www.rte.espol.edu.ec/index.php/tecnologica/article/view/461. [Último acceso: 2015].

[10] J. D. Yepes González, C. J. Pardo Calvache y O. S. Gómez, «Estado del Arte de la Utilización de Metodologías Ágiles y Otros Modelos en Pymes de Software,» *Informática - XVI Convensión y Feria Internacional 206,* 2016.

[11] M. A. Abdel-Fattah, «Grounded theory and action research as pillars for interpretive information systems research: A comparative study,» *Egyptian Informatics Journal,* nº 16, pp. 309 - 327, 2015.

[12] M. Yearworth, G. Edwards, J. Davis, K. Burger y A. Terry, «Integrating Problem Solving and Research Methods Teaching for Systems Practice in Engineering,» *Procedia Computer Science,* nº 16, pp. 1072 - 1081, 2013.

[13] R. K. Yin, «Case Study Research. Design and Methods,» *Sage Publications,* vol. 4, p. 240, 20019.

[14] P. Runeson y M. Höst, «Guidelines for conducting and reporting case study research,» *Empirical Software Engineering,* vol. 14, nº 2, pp. 131-164, 2009.

[15] CMMI Institute, «CMMI Institute,» 2016. [En línea]. Available: http://cmmiinstitute.com/. [Último acceso: Enero 2016].

[16] CMMI Product Team, CMMI for Development, Version 1.3, Hanscom AFB: Carnegie Mellon, 2010.

[17] CMMI Product Team, CMMI for Services, Version 1.3, Hanscom AFB: Carnegie Mellon, 2010.

[18] CMMI Product Team, CMMI for Acquisition, Version 1.3, Hanscom AFB: Carnegie Mellon, 2010.

[19] ISO, ISO/IEC TR 29110-1:2011. Software engineering - Lifecycle profiles for VSEs - Part 1: Overview, Ginebra: ISO, 2011.

[20] ISO, ISO/IEC 29110-2:2011. Software engineering - Lifecycle profiles for VSEs - Part 2: Framework and taxonomy, Ginebra: ISO, 2011.

[21] ISO, ISO/IEC TR 29110-3:2011. Software engineering - Lifecycle profiles for VSEs - Part 3: Assessment guide, Ginebra: ISO, 2011.

[22] ISO, ISO/IEC 29110-4-1:2011. Software engineering - Lifecycle profiles for VSEs - Part 4-1: Profile specifications, Ginebra: ISO, 2011.

[23] ISO, ISO/IEC TR 29110-5-6-2:2014. Software engineering - Lifecycle profiles for VSEs - Part 5-6-2: Management and engineering guide, Ginebra: ISO, 2014.

[24] ISO, ISO/IEC TR 29110-5-1-1:2012. Software engineering - Lifecycle profiles for VSEs - Part 5-1-1: Management and engineering guide, Ginebra: ISO, 2012.

[25] Scrum.Org and ScrumInc, «Scrum Guide,» [En línea]. Available: http://www.scrumguides.org/scrum-guide.html. [Último acceso: Marzo 2016].

[26] SolutionsIQ, «SolutionsIQ,» [En línea]. Available: http://www.solutionsiq.com/spec-writing-game/. [Último acceso: Octubre 2015].

[27] B. Kitchenham, «Procedures for Performing Systematic Reviews,» *Joint Technical Report,* vol. 33, pp. 1-26, 2014.

[28] C. Pardo, «Revisión Sistemática de la Armonización de Marcos en la Mejora de los Procesos Software,» *Departamento de Tecnologías y Sistemas de Información,* 2009.

[29] T. Bipp, A. Lepper y D. Schmedding, «Pair programming in software development teams – An empirical study of its benefits,» *Information and Software Technology,* vol. 50, pp. 231-240, 2008.

[30] P. Clarke y R. V. O'Connor, «The situational factors that affect the software development process: Towards a comprehensive reference framework,» *Information and Software Technology,* vol. 54, pp. 433-447, 2012.

[31] W. Huang, R. Li, H.-j. Yang, C. Maple, D. Foskett y V. Cleaver, «A novel lifecycle model for Web-based application development in small and medium enterprises,» *International Journal of Automation and Computing,* vol. 7, pp. 389-398, 2010.

[32] K. Nageswara Rao, G. Kavita Naidu y P. Chakka, «A Study of the Agile Software Development Methods, Applicability and Implications in Industry,» *International Journal of Software Engineering and Its Applications,* vol. 5, 2011.

[33] M. Fritzsche y P. Keil, «Agile Methods and CMMI: Compatibility or Conflict?,» *e-Informatic a Software Engineering Journal,* vol. 1, 2007.

[34] A. S. C. Marçal, B. C. C. de Freitas, F. S. Furtado Soares y A. D. Belchior, «Mapping CMMI Project Management Process Areas to SCRUM Practices,» *Software Engineering Workshop,* pp. 13-22, 2007.

[35] R. V. O'Connor y C. Y. Laporte, «Software Project Management in Very Small Entities with ISO/IEC 29110,» *Systems, Software and Services Process Improvement,* vol. 301, pp. 330-341, 2013.

[36] B. Losada, M. Urretavizcaya y I. Fernández Castro, «A guide to agile development of interactive software with a “User Objectives”-driven methodology,» *Science of Computer Programming,* vol. 78, pp. 2268-2281, 2013.

[37] N. B. Moe, T. Dingsøyr y T. Dybå, «A teamwork model for understanding an agile team: A case study of a Scrum project,» *Information and Software Technology,* vol. 52, pp. 480-491, 2010.

[38] V. Siddoo, N. Wongsai y R. Wetprasit, «An Implementation Approach of ISO/IEC 29110 for Government Organizations,» *Lecture Notes in Computer Science,* vol. 7983, pp. 5-19, 2013.

[39] S. Galván Cruz, M. Mora, R. O'Connor, F. Acosta Escalante y F. Alvarez, «On project management process in agile systems development methodologies and the ISO/IEC 29110 standard (entry profile),» *International Conference on Informatics and Computing (CNCIIC-ANIEI),* 2014.

[40] F. Navarrete, P. Botella y X. Franch, «An Approach to Reconcile the Agile and CMMI Contexts in Product Line Development,» *APLE '06: First International Workshop on Agile Product Line Engineering (in conjunction with SPLC),* 2006.

[41] M. Pikkarainen y A. Mantyniemi, «An approach Using CMMI in Agile Software Development Assessments: Experiences from Three Case Studies,» 2006.

[42] J. A. Hurtado Alegría y M. C. Bastarrica, «Implementing CMMI using a Combination of Agile Methods,» *CLEI ELECTRONIC JOURNAL,* vol. 9, nº 1, 2006.

[43] A. C. Pasini, S. Esponda, M. Boracchia y P. Pesado, «Q-Scrum: una fusión de Scrum y el estándar ISO/IEC 29110,» *XVIII Congreso Argentino de Ciencias de la Computación,* 2013.

[44] S. L. H. B. H. S. L. Diane E. Storde, «Coordination in co-located agile software development projects,» *Journal of Systems and Software,* vol. 85, pp. 1222-1238, 2012.

[45] H. Sharp y H. Robinson, «Collaboration and co-ordination in mature eXtreme programming teams,» *International Journal of Human-Computer Studies,* vol. 66, pp. 506-518, 2008.

[46] S. Wood, G. Michaelides y C. Thomson, «Successful extreme programming: Fidelity to the methodology or good teamworking?,» *Information and Software Technology,* vol. 55, pp. 660-672, 2013.

[47] J. N. S. M. Rashina Hoda, «The impact of inadequate customer collaboration on self-organizing Agile teams,» *Information and Software Technology,* vol. 53, pp. 521-534,

2011.

[48] M. Senapathi y A. Srinivasan, «Understanding post-adoptive agile usage: An exploratory cross-case analysis,» *Journal of Systems and Software,* vol. 85, pp. 1255-1268, 2012.

[49] K. Vlaanderen, S. Jansen, S. Brinkkemper y E. Jaspers, «The agile requirements refinery: Applying SCRUM principles to software product management,» *Information and Software Technology,* vol. 53, pp. 58-70, 2011.

[50] N. J. Linda Rising, «The Scrum software development process for small teams,» *IEEE Software,* vol. 17, pp. 26-32, 2000.

[51] H. Hajjdiab, A. S. Taleb y J. Ali, «An Industrial Case Study for Scrum Adoption,» *Journal of Software,* vol. 7.

[52] R. Hoda, J. Noble y S. Marshall, «Organizing Self-Organizing Teams,» *Proceedings – International Conference on Software Engineering,* vol. 1, pp. 285-294, 2010.

[53] J. Kasurinen, R. Laine y K. Smolander, «How applicable is ISO/IEC 29110 in Game Software Development?,» *Product-Focused Software Process Improvement,* pp. 5-19, 2013.

[54] M. Pikkarainen y A. Mantyniemi, «An approach Using CMMI in Agile Software Development Assessments: Experiences from Three Case Studies,» 2006.

[55] H. Glazer, J. Dalton, D. Anderson, D. J. Anderson, M. Konrad y S. Shrum, «CMMI or Agile: Why Not Embrace Both!,» Software Engineering Process Management, Pittsburgh, 2008.

[56] R. R. Lutz, «Software Engineering for Safety: A Roadmap,» *Proceeding ICSE '00 Proceedings of the Conference on The Future of Software Engineering,* pp. 213-226 , 2000.

[57] Object Management Group, «Object Management Group,» [En línea]. Available: http://www.omg.org/spec/SPEM/2.0/. [Último acceso: Febrero 2016].

[58] Object Management Group, «Object Management Group - Business Process Model and Notation,» [En línea]. Available: http://www.bpmn.org/. [Último acceso: Enero 2016].

[59] G. Adzic, Impact Mapping: Making a big impact with software products and projects, Surrey: Provoking Thoughts Limited, 2012.

[60] J. Patton, User Story Mapping: Discover the Whole Story, Build the Right Product, Beijing ; Sebastopol, CA: O'Reilly Media, 2014.

[61] M. R. Barbacci, R. J. Ellison , C. B. Weinstock y W. G. Wood, «Software Engineering Institute - Carnegie Mellon University,» Enero 2000. [En línea]. Available: http://www.sei.cmu.edu/reports/00sr001.pdf. [Último acceso: Mayo 2015].

[62] P. Kruchten, «Architectural Blueprints—The “4+1” View Model of Software Architecture,» *IEEE Software ,* vol. 12, nº 6, 1995.

[63] Agile Alliance, «Agile Alliance - Invest,» [En línea]. Available: https://www.agilealliance.org/glossary/invest/. [Último acceso: Noviembre 2015].

[64] N. L. Kerth, Project Retrospectives: A Handbook for Team Reviews, New York: Dorset House Publishing Co Inc.,U.S., 2001.

[65] M. P. Kua, The Retrospective Handbook: A guide for agile teams, S.l.: CreateSpace Independent Publishing Platform, 2013.

[66] K. Beck, Test Driven Development: By Example, Boston: Addison-Wesley Professional, 2002.

[67] M. Gärtner, ATDD by Example: A Practical Guide to Acceptance Test-Driven Development, Boston: Addison-Wesley Professional, 2012.

[68] K. Beck, Extreme Programming Explained: Embrace Change, Boston: Addison-Wesley, 2004.

[69] «planningpoker.com,» [En línea]. Available: https://www.planningpoker.com/. [Último acceso: Noviembre 2015].

[70] M. Fowler, K. Beck, J. Brant, W. Opdyke y D. Roberts, Refactoring: Improving the Design of Existing Code, Massachussetts: Addison-Wesley Professional, 1999.

[71] W. Grant, «The Upward Spiral: Learning new stuff,» 1 Abril 2012. [En línea]. Available: https://waynedgrant.wordpress.com/2012/04/01/sprint-retrospective-techniques/. [Último acceso: Octubre 2015].

[72] Universidad Eafit, «Modelo GQM (Goal Question Metric),» Medellín, 2014.

[73] Coder Dojo, «Coder Dojo,» [En línea]. Available: https://coderdojo.com/. [Último acceso: Marzo 2016].

Anexo A. Encuesta Implementación y Adopción del Método Ágil Propuesto

Implementación y Adopción Método Ágil Propuesto

Esta encuesta esta dirigida al equipo de desarrollo de HMV Ingenieros y tiene como objetivo evaluar el nivel de adopción y éxito, la utilización del método ágil propuesto para el proyecto de gestión de documentos de pago de la organización.

¿Qué tan satisfecho y motivado se sintió con la etapa de capacitación sobre el método ágil empleado?

- ○ Muy satisfecho
- ○ Satisfecho
- ○ No tan satisfecho
- ○ Para nada satisfecho

Por favor, expanda con sus palabras la respuesta anterior

Tu respuesta

¿Está de acuerdo con la utilización de los artefactos y herramientas propuestas para la ejecución del proyecto?

- ○ Totalmente de acuerdo
- ○ No tan de acuerdo
- ○ Para nada de acuerdo

Por favor, expanda con sus palabras la respuesta anterior

Tu respuesta

►¿Cómo se sintió con la utilización de las técnicas y buenas prácticas de desarrollo, sintió que ayudaron a mejorar el proceso de desarrollo?

Tu respuesta

►¿Cómo se sintió con la realización de las ceremonias, cree usted que realmente ayudan a mejorar la gestión del proyecto?

Tu respuesta

¿Cree que la documentación del proyecto y registro de los sucesos, como las lecciones aprendidas, solicitudes de cambio y resultados de pruebas, ayudaron en la gestión del proyecto y por qué?

Tu respuesta

¿Cree que la documentación del proyecto y registro de los sucesos, como las lecciones aprendidas, solicitudes de cambio y resultados de pruebas, le pueden ayudar en la ejecución de otro proyecto que se presente a futuro y por qué?

Tu respuesta

¿Estaría dispuesto a seguir usando el método de gestión ágil en futuros proyectos y por qué?

Tu respuesta

Anexo B. Encuesta de Satisfacción para el Product Owner

Encuesta de Satisfacción para el Product Owner

Esta encuesta esta dirigida al Product Owner del proyecto de gestión de documentos de pago de la organización y tiene como objetivo, medir el nivel de satisfacción del cliente con el producto final y su experiencia al participar en las distintas actividades que proponen el método ágil

¿Ha quedado satisfecho con el producto obtenido al final del proyecto?

- ◯ Si
- ◯ No

¿Qué opinión tiene acerca del método que se llevo para la gestión del proyecto?

Tu respuesta

¿Cree que las distintas reuniones que se llevaron para a lo largo del proyecto, ayudaron a mejorar la calidad del producto final y por qué?

Tu respuesta

¿Cree que las distintas reuniones que se llevaron para a lo largo del proyecto, ayudaron a que todos los participantes entendieran las necesidades y alcance del proyecto y por qué?

Tu respuesta

¿Que opinión tiene acerca de haber desempeñado el rol de dueño del producto (product owner) dentro del proyecto?

Tu respuesta

¿Participaría nuevamente en otro proyecto que utilice el método ágil propuesto en este proyecto?

Tu respuesta