



Universidad Nacional Mayor de San Marcos

Universidad del Perú. Decana de América

Dirección General de Estudios de Posgrado
Facultad de Ingeniería de Sistemas e Informática
Unidad de Posgrado

**Contribuciones para el aseguramiento de la calidad en
el proceso de desarrollo de software ágil**

TESIS

Para optar el Grado Académico de Doctor en Ingeniería de
Sistemas e Informática

AUTOR

Gloria de Lourdes ARCOS MEDINA

ASESOR

Dr. David Santos MAURICIO SÁNCHEZ

Lima, Perú

2023



Reconocimiento - No Comercial - Compartir Igual - Sin restricciones adicionales

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

Usted puede distribuir, remezclar, retocar, y crear a partir del documento original de modo no comercial, siempre y cuando se dé crédito al autor del documento y se licencien las nuevas creaciones bajo las mismas condiciones. No se permite aplicar términos legales o medidas tecnológicas que restrinjan legalmente a otros a hacer cualquier cosa que permita esta licencia.

Referencia bibliográfica

Arcos, G. (2023). *Contribuciones para el aseguramiento de la calidad en el proceso de desarrollo de software ágil*. [Tesis de doctorado, Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería de Sistemas e Informática, Unidad de Posgrado]. Repositorio institucional Cybertesis UNMSM.

Metadatos complementarios

Datos de autor	
Nombres y apellidos	Gloria de Lourdes Arcos Medina
Tipo de documento de identidad	Cédula de identidad
Número de documento de identidad	EC / 1801738764
URL de ORCID	https://orcid.org/0000-0003-4601-0768
Datos de asesor	
Nombres y apellidos	David Santos Mauricio Sánchez
Tipo de documento de identidad	DNI
Número de documento de identidad	06445495
URL de ORCID	https://orcid.org/0000-0001-9262-626X
Datos del jurado	
Presidente del jurado	
Nombres y apellidos	Carlos Edmundo Navarro Depaz
Tipo de documento	DNI
Número de documento de identidad	08482690
Miembro del jurado 1	
Nombres y apellidos	Cayo Víctor León Fernández
Tipo de documento	DNI
Número de documento de identidad	07001405
Miembro del jurado 2	
Nombres y apellidos	Lenis Rossi Wong Portillo
Tipo de documento	DNI
Número de documento de identidad	10438282
Datos de investigación	
Línea de investigación	C.0.3.22. Ingeniería de software
Grupo de investigación	Inteligencia artificial

Agencia de financiamiento	Sin financiamiento
Ubicación geográfica de la investigación	País: Ecuador Provincia: Chimborazo Ciudad: Riobamba Latitud: -1.67435 Longitud: -78.6483
Año o rango de años en que se realizó la investigación	2017 - 2022
URL de disciplinas OCDE	Ingeniería de sistemas y comunicaciones https://purl.org/pe-repo/ocde/ford#2.02.04



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
Universidad del Perú. Decana de América
Facultad de Ingeniería de Sistemas e Informática
Vicedecanato de Investigación y Posgrado
Unidad de Posgrado

**ACTA DE SUSTENTACIÓN VIRTUAL DE TESIS PARA OPTAR EL GRADO
ACADÉMICO DE DOCTOR EN INGENIERÍA DE SISTEMAS E INFORMÁTICA**

Al primer (01) día del mes de marzo de 2023, siendo las 5:00 pm., se reunieron en la sala virtual <https://meet.google.com/iir-ytso-jni> el Jurado de Tesis conformado por los siguientes docentes:

Dr. Carlos Edmundo Navarro Depaz (Presidente)
Dr. Cayo Víctor León Fernández (Miembro)
Dra. Lenis Rossi Wong Portillo (Miembro)
Dr. David Santos Mauricio Sánchez (Miembro asesor)

Se inició la Sustentación invitando a la candidata a Doctor **GLORIA DE LOURDES ARCOS MEDINA**, para que realice la exposición oral y virtual de la tesis para optar el Grado Académico de Doctor en Ingeniería de Sistemas e Informática, siendo la Tesis intitulada:

**“CONTRIBUCIONES PARA EL ASEGURAMIENTO DE LA CALIDAD EN EL
PROCESO DE DESARROLLO DE SOFTWARE ÁGIL”**

Concluida la exposición, los miembros del Jurado de Tesis procedieron a formular sus preguntas que fueron absueltas por la graduanda; acto seguido se procedió a la evaluación correspondiente, habiendo obtenido la siguiente calificación:

..... **DIECINUEVE (19) - EXCELENTE**

Por tanto, el presidente del Jurado, de acuerdo con el Reglamento General de Estudios de Posgrado, otorga a la Magíster **GLORIA DE LOURDES ARCOS MEDINA** el Grado Académico de Doctor en Ingeniería de Sistemas e Informática.

Siendo las 18:26 horas, el presidente del Jurado de Tesis, da por concluido el acto académico de Sustentación de Tesis.

Dr. Carlos Edmundo Navarro Depaz.
(Presidente)

Dr. Cayo Víctor León Fernández
(Miembro)

Dra. Lenis Rossi Wong Portillo
(Miembro)

Dr. David Santos Mauricio Sánchez
(Miembro Asesor)



Lima, 21 de Noviembre del 2022

INFORME N° 000013-2022-UPG-VDIP-FISI/UNMSM

INFORME DE EVALUACIÓN DE ORIGINALIDAD

1. Autoridad Académica que emite el Informe de Originalidad:	Director de la Unidad de Posgrado
2. Apellidos y Nombres de la autoridad académica:	Dr. Cayo Víctor León Fernández
3. Operador del programa informático de similitudes:	Dr. Cayo Víctor León Fernández
4. Documento evaluado:	Tesis para Posgrado Título: "Contribuciones para el aseguramiento de la calidad en el proceso de desarrollo de software ágil"
5. Autor del documento:	GLORIA DE LOURDES ARCOS MEDINA
6. Fecha de recepción de documento	18/11/2022
7. Fecha de aplicación del programa detector de similitudes:	18/11/2022
8. Software utilizado:	Turnitin
9. Configuración del programa detector de similitudes:	i. Excluye textos entrecomillados: SI ii. Excluye biografías: SI iii. Excluye cadenas menores a 40 palabras: SI iv. Otro criterio (especificar): NO
10. Porcentaje de similitudes según programa detector de similitudes	Diez por ciento (10%)
11. Fuentes originales de las similitudes encontradas	Se adjuntan en cinco (05) fojas al presente informe
12. Observaciones:	Ninguna
13. Calificación de originalidad i. Documento cumple criterios de originalidad, sin observaciones. ii. Documento cumple criterio de originalidad, con observaciones. iii. Documento no cumple criterios de originalidad.	Documento cumple criterio de originalidad, sin observaciones.
14. Fecha del Informe:	18/11/2022

DR. CAYO VICTOR LEON FERNANDEZ
DIRECTOR DE LA UNIDAD DE POSGRADO(e)

CLF/rda



DEDICATORIA

A mi madre, por impulsarme cada día a ser una mejor persona.

A mi esposo Freddy, por su paciencia y tolerancia durante todo el tiempo consumido en este difícil reto.

A mis hijos Freddy, Andrés y Lelia por su motivación constante y apoyo incondicional.

A Andresito y Marah Sofía, que llegaron a iluminar mi vida.

AGRADECIMIENTOS

Quiero expresar un especial agradecimiento al Dr. David Mauricio Sánchez, Asesor de este trabajo de investigación, mi admiración por su calidad humana y por su excelencia profesional que guiaron el desarrollo de este trabajo. Gracias por su paciencia y muchas horas de dedicación revisando las publicaciones y resultados de esta investigación.

A la UNMSM, a su planta docente y administrativa, que abrió sus puertas para que estudiantes ecuatorianos tengamos la oportunidad de obtener el grado de doctores en Sistemas e Informática.

A mis amigos Danilo, Alejandra e Ivonne, por compartir sus conocimientos que contribuyeron a mejorar esta investigación y por sus palabras de aliento en los momentos difíciles.

A la Escuela Superior Politécnica de Chimborazo, por brindarme todas las facilidades y el financiamiento para cumplir esta meta.

A Pedro Castañeda, nuestro primer amigo peruano, con quien compartimos muchas experiencias, por ser nuestro guía cultural y turístico de la hermosa ciudad de Lima y por su apoyo incondicional.

¡Mil gracias a todos!

ÍNDICE GENERAL

<i>CAPÍTULO 1: INTRODUCCIÓN</i>	1
1.1 Situación Problemática	1
1.2 Formulación del problema	3
1.2.1 Problema general	3
1.2.2 Problemas específicos	4
1.3 Justificación teórica	4
1.4 Justificación práctica	5
1.5 Objetivos de la investigación	9
1.5.1 Objetivo general	9
1.5.2 Objetivos específicos	10
1.6 Propuesta de la investigación	10
1.7 Matriz de consistencia	11
1.8 Organización de la tesis	17
<i>CAPÍTULO 2: MARCO TEÓRICO</i>	19
2.1 Antecedentes del problema	19
2.2 Calidad del software	22
2.2.1 Estándares de calidad del software	23
2.2.2 Tipos de calidad de software	26
2.3 Metodologías y principios ágiles	27
2.4 Factores críticos de éxito	31
<i>CAPÍTULO 3: REVISIÓN SISTEMÁTICA DE LITERATURA</i>	34

3.1	Introducción	34
3.2	Metodología para la revisión sistemática	34
3.3	Planificación de la revision	34
3.4	Conducción de la revisión	36
3.5	Evaluación de la calidad de los estudios primarios	37
3.6	Reporte de la revisión.....	39
3.6.1	Evolución de las publicaciones	39
3.6.2	Fuentes de datos.....	40
3.6.3	Clasificación de los estudios	40
3.7	Análisis.....	41
3.7.1	RQ1: ¿Cuáles son los factores que influyen en el éxito de un proyecto de desarrollo ágil? .	41
3.7.2	RQ2: ¿Cuáles son las métricas utilizadas para evaluar la calidad del software?	42
3.7.3	RQ3: ¿Cuáles son las prácticas ágiles que influyen en la calidad de software?	44
3.7.4	RQ4: ¿Cuáles son los modelos de calidad orientados proceso de desarrollo ágil?	45
3.7.5	RQ5: ¿Cuáles son los principios ágiles usados por los modelos de calidad encontrados?	73
3.7.6	RQ6: ¿Cuáles son las características de calidad que están contemplados en los modelos analizados?	74
3.7.7	RQ7: ¿Cuáles son los aspectos de calidad que cumplen los modelos encontrados?	75
3.8	Principales hallazgos	78
3.9	Conclusiones del capítulo	81
 <i>CAPÍTULO 4: MODELO CONCEPTUAL de INFLUENCIA DE LOS FACTORES CRÍTICOS DE ÉXITO Y PRÁCTICAS ÁGILES SOBRE LA CALIDAD DEL SOFTWARE</i>		
.....		83
4.1	Metodología para estructurar el modelo conceptual.....	83

4.2	Estructura del modelo conceptual.....	86
4.2.1	Componente 1: Factores críticos de éxito	88
4.2.2	Componente 2: Prácticas ágiles.....	92
4.2.3	Componente 3: Características de calidad	94
4.2.4	Componente 4: Hipótesis	95
4.3	Conclusiones del capítulo	102
 <i>CAPÍTULO 5: INFLUENCIA DE LOS FACTORES CRÍTICOS DE ÉXITO SOBRE LAS PRÁCTICAS ÁGILES DE DESARROLLO DE SOFTWARE.....</i>		
<i>103</i>		
5.1	Introducción	103
5.2	Motivación.....	104
5.3	Modelo conceptual.....	106
5.4	Metodología.....	106
5.4.1	Recopilación de datos	107
5.4.2	Método utilizado para el análisis de resultados	110
5.4.3	Validez de los resultados.....	112
5.5	Resultados de la influencia de los factores críticos de éxito sobre las prácticas ágiles de desarrollo de software.....	112
5.5.1	Estadística descriptiva.....	113
5.5.2	Resultados de la influencia de factores sobre las prácticas ágiles	114
5.5.3	Fiabilidad y viabilidad del test	115
5.5.4	Análisis de residuos ajustados	116
5.6	Discusión de resultados	117
5.6.1	Aplicación de metodologías ágiles	117
5.7	Prueba de Hipótesis	118
5.7.1	Hipótesis de la influencia de prácticas ágiles y características de calidad del software.....	118

5.8	Conclusiones del capítulo	121
 <i>CAPÍTULO 6: INFLUENCIA ENTRE PRÁCTICAS ÁGILES Y CARACTERÍSTICAS DE CALIDAD DEL SOFTWARE</i>		
<i>123</i>		
6.1	Introducción	123
6.2	Motivación.....	124
6.3	Metodología.....	125
6.4	Resultados de la influencia de las prácticas ágiles sobre las características de calidad del software	126
6.4.1	Estadística descriptiva.....	127
6.4.2	Fiabilidad y test de validez.....	128
6.4.3	Análisis de residuos ajustados	128
6.5	Prueba de hipótesis.....	129
6.5.1	Hipótesis de la influencia de prácticas ágiles y características de calidad del software.....	129
6.5.2	Influencia de las prácticas ágiles en la calidad del software	131
6.6	Relación entre los factores críticos de éxito y características de calidad del software.....	132
6.7	Conclusiones del capítulo	134
 <i>CAPÍTULO 7: CONCLUSIONES Y TRABAJOS FUTUROS</i>		
<i>135</i>		
7.1	Conclusiones.....	135
7.1.1	Conclusión General.....	135
7.1.2	Conclusiones específicas.....	135
7.2	Limitaciones	136
7.3	Trabajos futuros	137

LISTA DE TABLAS

Tabla 1-1 Matriz de consistencia	13
Tabla 2-1. Modelos ágiles.....	28
Tabla 2-2. Principios ágiles.....	30
Tabla 3-1 Criterios de selección y exclusión	36
Tabla 3-2 Estudios potenciales y estudios seleccionados	36
Tabla 3-3 Criterio de clasificación de la relevancia de los papers analizados.....	38
Tabla 3-4 Criterio de clasificación del número de citas de los papers analizados.	38
Tabla 3-5 Fuentes de las publicaciones sobre calidad en metodologías ágiles.....	40
Tabla 3-6 Clasificación de los estudios encontrados en RSL acerca de la aplicación de la calidad en el desarrollo de proyectos ágiles.....	41
Tabla 3-7 Factores críticos de éxito de un proyecto de desarrollo ágil	42
Tabla 3-8 Métricas aplicadas en los estudios seleccionados	43
Tabla 3-9 Prácticas ágiles aplicadas en los estudios encontrados.....	44
Tabla 3-10 Modelos de calidad aplicados al desarrollo ágil.....	45
Tabla 3-11 Principios ágiles que cumplen los modelos de calidad estudiados.....	73
Tabla 3-12 Número de atributos usados por los modelos de calidad y estudios primarios que aplican desarrollo de software ágil.	74
Tabla 3-13 Aspectos considerados para el análisis de los modelos.....	75
Tabla 3-14 Cumplimiento de los aspectos de calidad de los modelos estudiados.....	75
Tabla 3-15 Resumen del cumplimiento de los aspectos de calidad en los modelos estudiados.	76
Tabla 3-16 Características de calidad basados en otros estándares incluidas en los modelos estudiados.....	77
Tabla 4-1 Factores críticos de éxito propuestos.....	89
Tabla 4-2 Prácticas ágiles	93
Tabla 4-3 Características de calidad del software.....	94
Tabla 4-4 Hipótesis derivadas del Grupo 1.....	99
Tabla 4-5 Hipótesis derivadas del Grupo 2.....	101
Tabla 5-1 Ámbitos cubiertos por las prácticas ágiles y factores críticos de éxito	105
Tabla 5-2 Resumen de estadísticas	114
Tabla 5-3 Estadísticas de fiabilidad para las 24 variables analizadas.....	116
Tabla 5-4 Ranking de metodologías ágiles más usadas.....	117
Tabla 5-5 Resultados de las pruebas de hipótesis	118
Tabla 6-1 Contribuciones realizadas sobre las prácticas ágiles y las características de calidad.	124
Tabla 6-2 Resumen de Estadísticas.....	127
Tabla 6-3 Estadísticas de fiabilidad para las 32 variables analizadas.....	128
Tabla 6-4 Resultados de las pruebas de hipótesis	130

LISTA DE FIGURAS

Figura 1-1	Reporte de proyectos exitosos, fallidos y discutibles del 2011 al 2020.....	6
Figura 1-2	Proyectos exitosos, fallidos y discutibles usando modelos ágiles y modelo en cascada.....	7
Figura 1-3	Proyectos exitosos, fallidos y discutibles en proyectos pequeños, medianos y grandes.....	8
Figura 1-4	Esquema general de la propuesta de investigación	11
Figura 1-5	Proceso de lectura de la tesis.....	18
Figura 2-1	Estructura SQuaRE	26
Figura 3-1	Proceso aplicado para la RSL	37
Figura 3-2	Evolución de publicaciones acerca de la calidad aplicado al desarrollo ágil.....	39
Figura 3-3	Arquitectura AQAM	49
Figura 3-4	Proceso AQAM.....	49
Figura 3-5	Proceso de desarrollo ACRUM.....	51
Figura 3-6	Proceso AQUA.....	52
Figura 3-7	Ejemplo de la Tabla RAM	52
Figura 3-8	Proceso VAQ	53
Figura 3-9	Modelo de Sinch	55
Figura 3-10	Enfoque de Aseguramiento de la Calidad de Darwish.....	58
Figura 3-11	Integración Continua.....	60
Figura 3-12	Medición Continua.....	60
Figura 3-13	Mejora Continua.....	61
Figura 3-14	Esquema adquirido de métricas en el proceso iterativo	63
Figura 3-15	Sumarización de métricas	64
Figura 3-16	Estabilidad de código basada en artefactos de re-evaluación	65
Figura 3-17	Métricas de calidad típicas para pruebas de procesos.....	65
Figura 3-18	Metodología ágil centrada en el cliente	70
Figura 3-19	Modelo Conceptual de entrega a tiempo en proyectos ágiles	72
Figura 3-20	Características y subcaracterísticas del estándar ISO/IEC 25010.....	74
Figura 4-1	Metodología para la construcción del modelo conceptual.....	85
Figura 4-2	Modelo Conceptual.....	87
Figura 5-1	Modelo parcial 1: Relaciones entre factores críticos de éxito y prácticas ágiles	106
Figura 5-2	Estrategia metodológica aplicada.....	107
Figura 5-3	Escala Stanines definida descriptivamente.	112
Figura 5-4	Metodologías ágiles utilizadas por los encuestados.....	113
Figura 5-5	Perfil de los encuestados	114
Figura 5-6	Mapa de calor de la media y la desviación estándar de las relaciones entre factores y prácticas ágiles	115
Figura 5-7	Mapa de calor de los residuos ajustados entre factores críticos de éxito y prácticas ágiles	116
Figura 5-8	Resultado de la influencia de los factores críticos de éxito en las prácticas ágiles de desarrollo de software	119
Figura 6-1	Modelo Conceptual para determinar el impacto de la aplicación de prácticas ágiles	

sobre las características de calidad del software	126
Figura 6-2 Mapa de calor de la media y desviación estándar de la relación entre prácticas ágiles y características de calidad.....	127
Figura 6-3 Mapa de calor de los residuos ajustados entre las prácticas ágiles y las características de calidad.	129
Figura 6-4 Resultados de la Influencia de categorías de prácticas ágiles en las características de calidad, con base en el estándar ISO/IEC 25010	130
Figura 6-5 Mapa de calor de residuos ajustados entre factores y características de calidad	134

LISTA DE ANEXOS

- Anexo A. Principales estándares de calidad de Software
- Anexo B. Estudios incluidos en la RSL
- Anexo C. Evaluación de la Calidad de los papers analizados
- Anexo D. Lista de Factores Críticos de Éxito encontrados en la RSL
- Anexo E. Lista de Métricas encontradas en la RSL
- Anexo F. Lista de Prácticas ágiles encontradas en la RSL
- Anexo G. Lista de características de calidad del software encontradas en la RSL
- Anexo H. Proceso de Selección de la hipótesis a probar. Relación factores críticos de éxito – Prácticas ágiles
- Anexo I. Proceso de Selección de la hipótesis a probar. Relación Prácticas ágiles – características de calidad
- Anexo J. Encuesta aplicada
- Anexo K. Resumen obtenido de los datos de la encuesta. Relación Factor – Prácticas Ágiles
- Anexo L. Resumen obtenido de los datos de la encuesta. Influencia Prácticas Ágiles - Características de Calidad
- Anexo M: Publicaciones realizadas

ACRÓNIMOS Y ABREVIATURAS

Acrónimo / Abreviatura	Significado
ACM	Association for Computing Machinery
ACOMP	Atributo de Compatibilidad
AFUN	Atributo de Funcionalidad
ACRUM	Attribute-driven sCRUM
AMAIN	Atributo de Mantenibilidad
APEF	Atributo de Eficiencia
APORT	Atributo de Portabilidad
AQAM	Agile Quality Assurance Model
AQUA	Analysis of Quality Attributes
ARELI	Atributo de Fiabilidad
ASD	Agile Software Development / Adaptative Software Development
ASEC	Atributo de Seguridad
AUP	Agile Unified Process
AUSAB	Atributo de Usabilidad
CI	Integración continua
CK	Chidamber & Kemerer
CMMI	Capability Maturity Model Integrated
DRE	Eficiencia en la eliminación de defectos
DSDM	Dynamic Systems Development Method
FCM	Factor Medios de Comunicación usados
FCO	Factor Confianza
FDD	Feature-Driven Development
FIAI	Factor Integridad y Disponibilidad de la Información y Experiencias Aprendidas
FWQL	Factor Calidad de Vida Laboral
FSE	Factor Autoeficacia Percibida
FSTR	Factor Estrés
GQM	Goal – Question – Metric

Acrónimo / Abreviatura	Significado
IDE	Entorno de Desarrollo Integrado
IEC	International Electrotechnical Commsision
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
LD	Lean Development
LOC	Lines of Code
MOOD	Metrics for Object Oriented Design
PENG	Práctica de Ingeniería
PMD	Programming Mistake Detection
PMP	Práctica de gestión del Proyecto
PTEST	Práctica de Testing
PWT	Práctica Equipo de Trabajo
QMOOD	Quality Metrics for Object Oriented Design
RAM	Relation Association Matrix
RQ	Research Question
RSL	Revisión Sistemática de Literatura
SQuaRE	Software Quality Requirements and Evaluation
TMMI	Testing Maturity Model Integrated
UAT	Prueba de Aceptación de Usuario
VAQ	Validation of Quality Attribute
VCS	Sistema de Control de Versiones
XP	EXtreme Programming

RESUMEN

El propósito de esta investigación es contribuir en el aseguramiento de la calidad del software a través de la determinación de nuevos factores críticos de éxito y prácticas ágiles que influyen en las características de calidad de software. A pesar de que existen estándares y modelos de calidad para asegurar la calidad en proyectos de desarrollo ágil, su complejidad en su aplicación es contraria a los principios propuestos por el agilismo, lo que impide tener un software con un adecuado nivel de calidad. Para cumplir con el objetivo propuesto se ha aplicado un diseño de investigación cualitativa, los resultados se obtuvieron mediante el análisis de residuos estandarizados a 146 cuestionarios dirigidos a personas involucradas en el proceso de desarrollo de software. Esta investigación contiene una revisión sistemática de la literatura que permitió catalogar 148 factores críticos de éxito, 137 prácticas ágiles, 165 métricas y 71 atributos de calidad relacionados con el desarrollo ágil de software. Además, se presenta un modelo compuesto por cuatro componentes: 1) seis factores críticos de éxito obtenidos de las teorías del comportamiento humano y administrativo, 2) cuatro categorías de prácticas ágiles, 3) ocho características de calidad basadas en el estándar ISO/IEC 25010, y 4) hipótesis que determinan la influencia entre los componentes del modelo propuesto. Las contribuciones realizadas como resultado de este estudio permitirán asegurar la calidad en el proceso de desarrollo de software ágil y tomar acciones para mitigar la influencia negativa de algunos factores críticos y contribuir a la implementación de proyectos exitosos.

PALABRAS CLAVE: Ingeniería de Software, ASD, Desarrollo de Software Ágil, Calidad de Software, Prácticas Ágiles, Factores de Críticos de éxito.

ABSTRACT

The purpose of this research is to contribute to software quality assurance through the determination of new critical success factors and agile practices that influence software quality characteristics. Although there are quality standards and models to ensure quality in agile development projects, their complexity in their application is contrary to the principles proposed by agile, which prevents having software with an adequate level of quality. To meet the proposed objective, a qualitative research design has been applied, the results were obtained through the analysis of standardized residues to 146 questionnaires addressed to people involved in the software development process. This research contains a systematic review of the literature that allowed us to catalog 148 critical success factors, 137 agile practices, 165 metrics, and 71 quality attributes related to agile software development. In addition, a model composed of four components is presented: 1) six critical success factors obtained from the theories of human and administrative behavior, 2) four categories of agile practices, 3) eight quality characteristics based on the ISO standard. /IEC 25010, and 4) hypotheses that determine the influence between the components of the proposed model. The contributions made as a result of this study will make it possible to ensure quality in the agile software development process and take actions to mitigate the negative influence of some critical factors and contribute to the implementation of successful projects.

KEYWORDS: Software Engineering, ASD, Agile Software Development, Software Quality, Agile Practices, Success Critical Factors.

CAPÍTULO 1: INTRODUCCIÓN

1.1 Situación Problemática

Desde los últimos años, muchos productos nunca llegan a el mercado, y para aquellos que lo hacen, el éxito no está garantizado. La razón detrás de su fracaso es el proceso de ingeniería de software que se utiliza para desarrollar estos productos. Muchas veces el proceso utilizado se enfoca en la entrega oportuna del producto, los procesos de ingeniería de software se enfocan más en la especificación de requisitos, refactorización, tiempo de entrega, costo de desarrollo y minimización de esfuerzos y muchos más factores de los que depende el desarrollo de software (Jain et al., 2018). Sin embargo, a más de definir un proceso de desarrollo de software que permita crear un producto de calidad.

El aseguramiento de la calidad del software (SQA por sus siglas en inglés Software Quality Assurance) es un método para ayudar a lograr la calidad. SQA puede monitorear el proceso de ingeniería de software y los métodos utilizados para desarrollar la calidad del software (Kassie & Singh, 2020). El aseguramiento de la calidad en el proceso de desarrollo de software mediante el uso de metodologías ágiles se determina mediante el conjunto de actividades planificadas y sistemáticas necesarias para aportar la confianza que el software satisface los requisitos dados de calidad para conseguir un producto sostenible en el tiempo y que sea fácil de operar.

Los métodos ágiles e híbridos (combinación de modelos ágiles y modelos tradicionales), y lo que es más importante, una mentalidad ágil, están formando parte de algunas funciones organizacionales (Digital.ai, 2021), por lo que se hace necesario adaptar el pensamiento organizacional a la cultura ágil.

Si bien esta realidad dinámica y el cambio continuo que promueven las metodologías ágiles, podría percibirse como una amenaza para la calidad, en cambio presenta una oportunidad para reconsiderar la calidad dentro de un nuevo contexto operativo (Cheriyann et al., 2020). Por esta razón se puede afirmar que el aseguramiento de la calidad de software también está cambiando (Nataraj et al., 2016), (Fuggetta & Di Nitto, 2014),

no porque la calidad haya cambiado sino porque el mundo está cambiando irreversiblemente y las necesidades del cliente van acorde con este cambio (Liubchenko, 2016).

La calidad del producto de software está directamente relacionada con la calidad del proceso empleado para su desarrollo (SEI, 2010), a pesar de que los modelos ágiles tienen muchas ventajas, su utilización no garantiza un proceso de calidad; tampoco se puede afirmar que ágil es la solución a todos los problemas en la entrega de software de alta calidad (Nader-Rezvani, 2019b); importante que todo el conjunto de actividades que se lleva a cabo para construir, desplegar y evolucionar el producto de software desde su concepción hasta su retiro garantice un software de calidad, sostenible en el tiempo.

A pesar de los esfuerzos realizados por las metodologías de software ágiles y propuestas realizadas, el desarrollo de software no es perfecto y se ve influenciado por factores humanos y organizacionales, y prácticas ágiles que inciden en la calidad de software; por lo que es importante definir cómo influyen estos aspectos en la calidad del software de tal forma que se proporcione la agilidad y dinamismo que un modelo ágil exige, y a su vez que no compliquen el proceso de desarrollo del software. También es importante que los desarrolladores mejoren la medición de la calidad y la incluyan en sus procesos de toma de decisiones (López et al., 2022).

Según (Jharko, 2021), los principales factores para la insatisfacción de la calidad son los siguientes:

- Errores funcionales e inconsistencias con los requisitos.
- Errores no funcionales, como violación de las reglas del lenguaje de programación, uso de funciones de biblioteca y componentes de terceros, etc.
- Errores durante el control, entre los que se pueden incluir, costos laborales evaluados incorrectamente, plan de trabajo elaborado incorrectamente.
- Transparencia insuficiente del proceso de desarrollo ya que en un momento dado, es difícil evaluar el estado del proyecto y el porcentaje de finalización.
- Falta de garantía de calidad, debido a la probabilidad de errores a lo largo de todo el ciclo de vida de un sistema digital.

Al analizar la literatura y estándares existentes, se ha determinado que existe una gran cantidad de modelos para asegurar la calidad del software, a nivel del producto, y a nivel del proceso de desarrollo; sin embargo, muy pocas investigaciones han propuesto métodos aplicables al desarrollo ágil, la mayoría de ellas no son compatibles con el dinamismo y flexibilidad de los modelos ágiles; por lo que se podría decir que calidad y modelos ágiles son una contradicción. Por otra parte, los modelos existentes, no han incorporado factores críticos de éxito y prácticas ágiles que influyen en la calidad de software.

A pesar de que existen varios modelos de calidad propuestos, tanto para metodologías ágiles como tradicionales, estos modelos presentan las siguientes debilidades:

- La aplicación en casos de estudio es limitada.
- Su complejidad está contrapuesta con los principios del desarrollo ágil de aplicaciones tomando en cuenta que la comunicación del equipo de trabajo con el cliente y el tiempo de entrega es de vital importancia en estos modelos de desarrollo.
- Se constituyen en esfuerzos aislados orientados a la calidad, que no se han integrado un modelo formal integral que cubran todas las etapas del proceso de desarrollo basados en factores críticos de éxito y prácticas ágiles que inciden en la calidad del software.

Por lo tanto, es necesario determinar los factores críticos de éxito y las prácticas ágiles que influyen en la calidad del software, de tal manera que se evite el fracaso de los proyectos de software.

1.2 Formulación del problema

1.2.1 Problema general

PG Inadecuado nivel de calidad de software, debido a la poca implementación de

factores críticos de éxito y prácticas ágiles que influyan los sobre las características de calidad durante el proceso de desarrollo de software, lo que dificulta su operación y mantenimiento futuro.

1.2.2 Problemas específicos

El problema general ha sido sistematizado en los siguientes problemas específicos:

PE1 Poco conocimiento de los factores críticos de éxito, prácticas ágiles y atributos de calidad que se aplican en un proyecto de desarrollo de software utilizando metodologías ágiles.

PE2 Desconocimiento del nivel de influencia de los factores críticos de éxito en la utilización de prácticas ágiles.

PE3 Falta de conocimiento de la influencia de las prácticas ágiles en las características de calidad de un software.

PE4 Conocimiento limitado de la influencia de los factores críticos de éxito en las características de calidad de un software.

1.3 Justificación teórica

Hoy en día surge la necesidad de las empresas e instituciones de contar con software de calidad, que cumpla las expectativas y exigencias del cliente; esto puede ser posible aplicando un continuo control de la calidad del software en base a estándares, modelos y métricas que se adapten al proceso dinámico de las metodologías ágiles.

Desde el inicio del presente milenio, las metodologías ágiles de desarrollo de software han sido adoptadas por muchas empresas; éstas se caracterizan por ser incrementales, colaborativas, sencillas y adaptativas (Abrahamsson et al., 2017). Hoy en día, los beneficios de los modelos ágiles han sido reconocidos por gestores y desarrolladores de proyectos de software. Las metodologías tradicionales no se adaptan fácilmente a los nuevos requerimientos de los clientes, se basan en una planificación

rígida previamente definida, no son flexibles ante las nuevas necesidades que tienen los usuarios hoy en día, estos aspectos marcan una gran contraste con las metodologías ágiles que permite el desarrollo del software desde la recopilación de requisitos hasta la prueba y entrega del producto y espera la retroalimentación del usuario (Abdalhamid et al., 2019).

El propósito de esta investigación es contribuir con nuevos aportes que complementen el conocimiento ya existente sobre la aplicación de prácticas ágiles y fortalecimiento de factores críticos de éxito como aspectos que contribuirán a la mejora de las características de calidad de software. Los resultados de esta investigación podrán sistematizarse en una serie de estrategias que pueden ser incorporadas al conocimiento ya existente con respecto a la mejora de la calidad en proyectos que utilizan metodologías ágiles en su proceso de desarrollo y contribuir en la toma de decisiones mediante la aplicación de prácticas ágiles y la medición de la calidad.

1.4 Justificación práctica

Existen varios autores que motivan su investigación en una serie de encuestas realizadas desde el año de 1998, en las que se determinan los principales problemas del fracaso de proyectos de software alrededor del mundo; por ejemplo, (N. P. Singh & Soni, 2011) en su investigación recoge una serie de resultados que determinan que mientras más grande es el proyecto, más probable es su fracaso. (*Statistics over IT Failure Rate*, 2001), así mismo menciona que en el Reino Unido, únicamente el 13% de los proyectos de desarrollo de software no fracasan y la utilización del modelo en cascada fue el principal factor que contribuyó al fracaso, siendo citado en el 82% de los proyectos como el problema número uno.

Investigaciones más recientes como la realizada por (Standish Group, 2020), el 19% de los proyectos de software fracasan, el 50% no cumplen con presupuestos, planificación y características solicitadas por los usuarios solamente el 31% de los proyectos ha sido exitosos. Si se considera conjuntamente el porcentaje de proyectos fallidos y el porcentaje de proyectos que no pueden categorizarse como exitosos ni fallidos, dan un total del 69%, considerado muy alto en comparación con los proyectos exitosos. En la Figura 1-1, se representan los porcentajes de proyectos exitosos, fallidos

y discutibles desde el año 2011, de acuerdo con estos datos, los picos altos y bajos indican que no existe una tendencia al incremento de proyectos exitosos, ni a la reducción de proyectos fracasados. De esto se puede deducir que las metodologías, estándares y buenas prácticas de gestión y desarrollo de software, no han sido suficientes para mejorar este escenario. El Chaos Report 2020, incorpora nuevos factores que garantizan el éxito de un proyecto, como: un buen patrocinador, un buen equipo y un buen lugar, por lo que estos aspectos se convierten en una prioridad para que la empresa y mejorar el rendimiento de un proyecto.

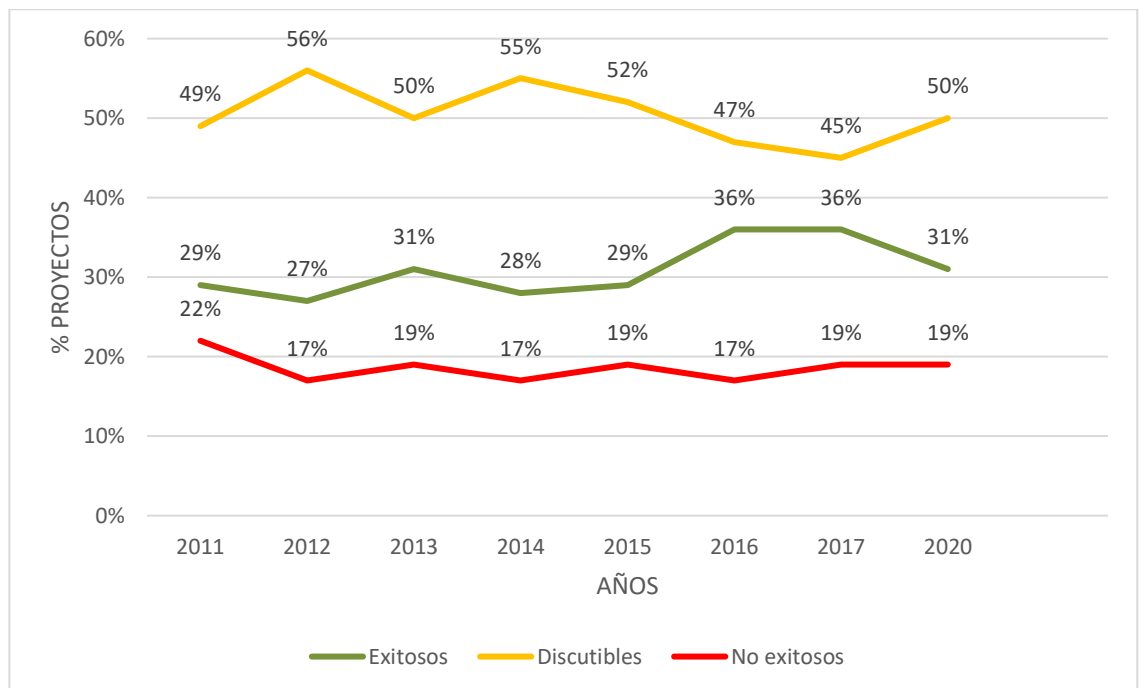


Figura 1-1 Reporte de proyectos exitosos, fallidos y discutibles del 2011 al 2020

Fuente: Chaos Reports

Por otro lado, es importante recalcar que los datos de la Figura 1-1, no consideran la metodología aplicada en el desarrollo de software, el mismo Chaos Manifiesto 2020, hace una relación entre el éxito, fracaso y cambios de los proyectos considerando el modelo de desarrollo aplicado; este estudio considera la utilización de modelos ágiles y el modelo en cascada como principal representante de los modelos tradicionales; como lo indica la Figura 1-2 en el año 2020 el 11% de los proyectos que usaron modelos ágiles no han tenido éxito, mientras que al analizar los proyectos que utilizaron el modelo en cascada, el 28% fracasaron; si a este porcentaje se incrementa los proyectos discutibles

se llega a un 58% en ágil 87% en cascada, que se puede considerar un porcentaje alto. Tomando en cuenta que los modelos ágiles fueron difundidos a partir del año 2001 para contribuir a la mejora de la calidad del software y dar contestación más fácilmente a las nuevas expectativas de los usuarios, los resultados demuestran que este fin aún no se ha logrado.



Figura 1-2 Proyectos exitosos, fallidos y discutibles usando modelos ágiles y modelo en cascada

Fuente: Standish Group Report 2020

Cuando se considera el tamaño de los proyectos, se determina que el 59% de los proyectos pequeños que usan metodologías ágiles son exitosos; mientras que cuando se usa la metodología en cascada el 45% de los proyectos no son exitosos. Cuando se analizan los datos de los proyectos grandes la situación es más crítica, solamente el 19% de los proyectos que usan metodologías ágiles son exitosos, frente al 8% de los proyectos que usan la metodología en cascada, ver Figura 1-3 Proyectos exitosos, fallidos y discutibles en proyectos pequeños, medianos y grandes.

Por otro lado, las pérdidas económicas de un proyecto están alrededor de los 81 billones de dólares, debido a proyectos que han sido cancelados y un exceso de presupuesto por mala planificación del proyecto de 59 billones de dólares (Ichu & Nemani, 2011).

El desarrollo de software no es perfecto, a pesar de los esfuerzos realizados por las metodologías tanto ágiles como tradicionales, el proceso de desarrollo de software aún no es satisfactorio, lo que generalmente ha provocado retrasos, fallas, proyectos abandonados o rechazados (Larman, 2004); las fallas de un proyecto son causadas por la falta de definición de las características de calidad, consideradas como requisitos no funcionales ya que los usuarios y desarrolladores dan mayor énfasis en la funcionalidad del software (Jeon et al., 2011).

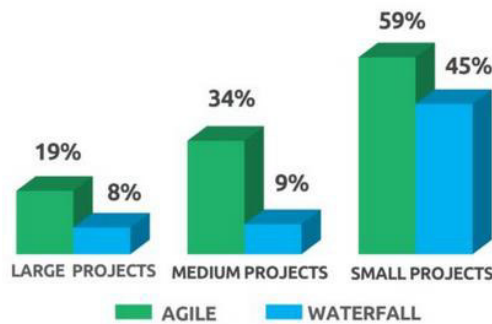


Figura 1-3 Proyectos exitosos, fallidos y discutibles en proyectos pequeños, medianos y grandes

Fuente: : Standish Group Report 2020

Muchos sistemas desarrollados tienen diseños no flexibles, difíciles de extender, código complicado, documentación escasa o inexistente, casos de prueba y resultados que no han sido archivados, gestión de cambio no existente; es decir, el software que no cumple los estándares mínimos de calidad por lo que causan problemas al momento de su actualización.

Los datos de Standish Group también segmentaron el éxito y el fracaso según el tamaño del proyecto para proyectos que usan metodologías ágiles y en cascada. Como puede imaginarse, los proyectos más pequeños tienen más éxito que los proyectos más grandes. Los proyectos más pequeños tienen menos riesgo y ciclos de retroalimentación más cortos. Los datos destacan que cuánto más pequeños son los proyectos el nivel de éxito aumenta. Para los proyectos ágiles, los proyectos pequeños tienen 3 veces más probabilidades de éxito que los grandes, mientras que para los proyectos en cascada, las tasas de éxito de los proyectos pequeños aumentan hasta 6 veces más que las de los grandes.

Muchos modelos de calidad han sido propuestos a fin de mejorar de manera continua los procesos de desarrollo de software, a su vez, permiten medir los resultados obtenidos, tanto del proceso como del producto de software, reduce costos de retrabajo. Las ventajas de la utilización de modelos de calidad se ven reflejadas en la satisfacción del cliente al ofrecerles productos de mayor calidad, cumpliendo los tiempos previstos para su entrega.

Hasta el momento, como se detalló anteriormente, existe una gran cantidad de literatura y estándares para evaluar y asegurar la calidad del software; sin embargo, muy

pocas investigaciones han propuesto métodos aplicables al desarrollo ágil tomando en cuenta que la comunicación del equipo de trabajo con el cliente y el tiempo de entrega es de vital importancia en estos modelos de desarrollo; el modelo de calidad que se aplique en este tipo de proceso de desarrollo también debe soportar el dinamismo y la flexibilidad que el manifiesto ágil propone.

De acuerdo con (Janus et al., 2012) y (Liubchenko, 2016), es necesario realizar conjuntamente actividades de gestión y aseguramiento de la calidad de tal forma que las acciones de garantía de calidad tienen que ser convertidos en prácticas ágiles o las prácticas ágiles tienen que adaptarse para responder al cambio.

Las metodologías ágiles como XP y Scrum plantean procesos de revisión del código; sin embargo, métodos como el PSP (Personal Process Software) o TSP (Personal Process Software), han logrado que los errores se reduzcan entre un 60 y 80%. La programación en parejas, una práctica ágil de XP, tiene resultados del 20 al 40% de reducción de errores, que no es mucho frente al 10 y el 25% de un programador (Sanabria & Acebey, 2008).

La cultura del aseguramiento de la calidad en el desarrollo de software, aportará resultados positivos en la generación de aplicaciones exportables y capaces de competir a nivel internacional, apoyando de esta manera al sector de producción de software de un país.

Esta investigación contribuye con aspectos de calidad que promueva la mejora continua, establezca procesos de desarrollo resultados medibles y promueva la eficiencia en estos procesos; los aspectos planteados, conllevan a explicar las relaciones que existen entre las características de calidad de un producto software y su desarrollo.

1.5 Objetivos de la investigación

1.5.1 Objetivo general

OG Contribuir en el aseguramiento de la calidad del software a través de la determinación de nuevos factores críticos de éxito y prácticas ágiles que influyen

en las características de calidad de software, de forma a mejorar su operación y mantenimiento futuro, que considere aspectos como baja complejidad en su implementación y estrategias para implementarlos.

1.5.2 Objetivos específicos

- OE1** Determinar los factores críticos de éxito, prácticas ágiles y atributos de calidad que han sido utilizados en el proceso de desarrollo de software.
- OE2** Analizar los estándares y modelos de aseguramiento de calidad de software existentes e identificar las limitaciones de los métodos existentes para asegurar la calidad del software en métodos ágiles de desarrollo de software.
- OE3** Establecer el nivel de influencia de los factores críticos de éxito en las prácticas ágiles en el contexto de la calidad del software.
- OE4** Determinar la influencia de las prácticas ágiles y las características de calidad del software.
- OE5** Establecer el nivel de influencia de los factores críticos de éxito en las características de calidad del software.

1.6 Propuesta de la investigación

La propuesta planteada es un punto de referencia que puede ser utilizado como parte de la gestión de la calidad incluida como parte del proceso de desarrollo ágil de proyectos de software, de tal forma que permitan adaptarse a los cambios y evolucionar acorde con los requerimientos de los usuarios. Además, incluye factores, prácticas ágiles, características y estrategias que deben aplicarse para cumplir los requisitos de calidad o no funcionales del software.

La propuesta tiene como fin asegurar la calidad de un proyecto de desarrollo de software que aplique modelos ágiles para generar la confianza de que se cumplan los requisitos de calidad durante el proceso de desarrollo de software, de tal forma que permita identificar los problemas de calidad en las primeras etapas de desarrollo; de esta manera, el desarrollador puede corregirlos de forma inmediata y poder continuar con el desarrollo del proyecto con rapidez; a fin de obtener un producto de calidad.

El aporte consiste en un modelo que representa como determinar la relación de los factores críticos de éxito y prácticas ágiles con las características de calidad del software y está compuesto por los siguientes elementos: **Estructura:** conformada por: factores, prácticas ágiles y características de calidad. **Actores/agentes**, es decir las personas que involucradas en el proceso de desarrollo, como: desarrolladores, jefes de desarrollo, clientes, etc.

Para llegar a la propuesta principal, se han planteado los siguientes aportes secundarios (Ver Figura 1-4):

- Relaciones entre factores de éxito y desarrollo ágil (prácticas ágiles)
- Relaciones entre prácticas ágiles características de calidad.
- Relaciones entre factores críticos de éxito y características de calidad del software.

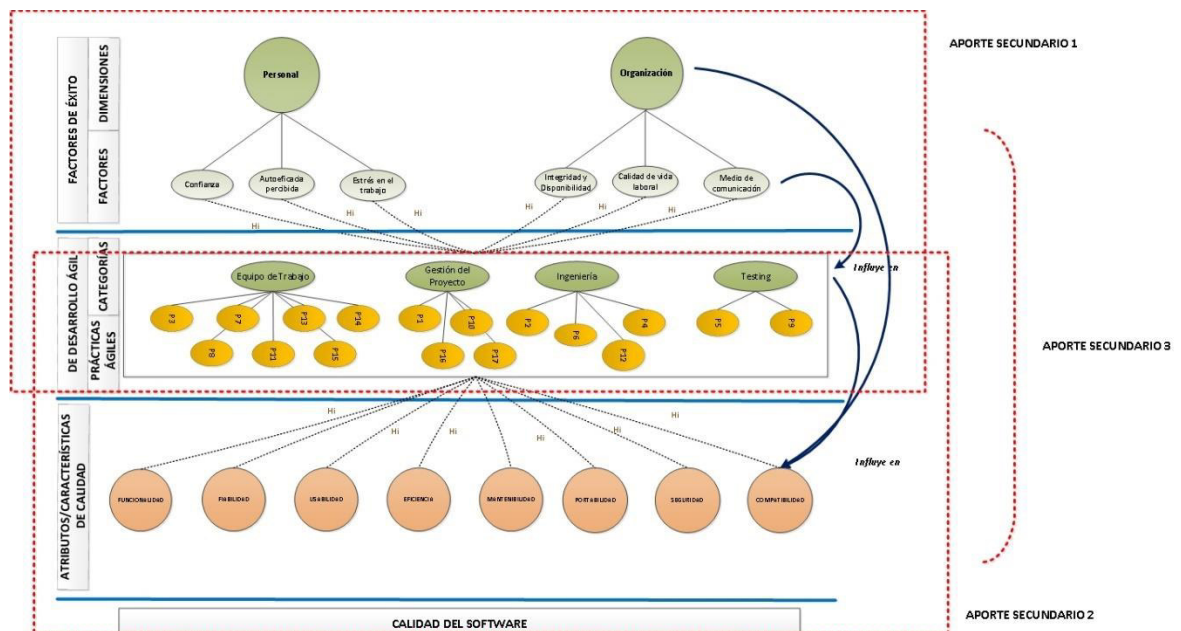


Figura 1-4 Esquema general de la propuesta de investigación

1.7 Matriz de consistencia

La matriz de consistencia que se presenta en la Tabla 1-1, resume en forma general y sucinta los elementos básicos de la investigación, permite presentar una visión de las variables, indicadores y su coherencia con el objetivo y el problema de la investigación.

Esta matriz permite visualizar los aspectos metodológicos del estudio que permitan cumplir los objetivos planteados con un rigor científico adecuado.

Tabla 1-1 Matriz de consistencia

Problema general	Objetivos	Hipótesis	Variables e Indicadores	Métodos/ Técnicas
Problema principal	Objetivo general	Hipótesis principal		
PG: Inadecuado nivel de calidad de software, debido a la poca implementación de factores críticos de éxito y prácticas ágiles que influyen sobre las características de calidad durante el proceso de desarrollo de software, lo que dificulta su operación y mantenimiento futuro.	OG: Contribuir en el aseguramiento de la calidad del software a través de la determinación de nuevos factores críticos de éxito y prácticas ágiles que influyen en las características de calidad de software, de forma a mejorar su operación y mantenimiento futuro, que considere aspectos como baja complejidad en su implementación y estrategias para implementarlos.	Hi: Se puede mejorar aseguramiento de la calidad del software a través de la implementación de nuevos factores críticos de éxito y prácticas ágiles que influyen en las características de calidad de software.	Variables Factores Críticos de éxito Prácticas ágiles Indicadores Nivel de implementación de factores Nivel de agilidad	• Método inductivo • Encuestas • Test de Cronbach • Análisis de residuos estandarizados
Problemas secundarios	Objetivos específicos	Hipótesis secundarias		
PE1: Poco conocimiento de los factores críticos de éxito, prácticas ágiles y atributos de calidad que se aplican en un proyecto de desarrollo de software utilizando metodologías ágiles.	OE1: Determinar los factores críticos de éxito, prácticas ágiles y características de calidad que han sido utilizados en el proceso de desarrollo de software. OE2: Analizar los estándares y modelos de aseguramiento de calidad de software existentes e identificar las limitaciones de los métodos existentes para asegurar la calidad del software en métodos ágiles de desarrollo de software.	H1: Los estándares y modelos de aseguramiento de calidad de software existentes no aplican los aspectos de calidad que requiere un proyecto de desarrollo de software ágil.	Variables Factores críticos de éxito Prácticas ágiles Métricas de calidad Modelos de calidad Indicadores Frecuencia de factores críticos existentes estudiados. Frecuencia de prácticas ágiles existentes estudiadas Frecuencia de métricas de calidad existentes estudiadas No. de modelos de calidad existentes	• Revisión Sistemática de Literatura • Estadística descriptiva

Problema general	Objetivos	Hipótesis	Variables e Indicadores	Métodos/ Técnicas
Problema principal	Objetivo general	Hipótesis principal		
			No. de estándares de calidad existentes % del cumplimiento de los modelos en los aspectos de calidad. % de aplicación de los principios de calidad	
PE2: Desconocimiento del nivel de influencia de los factores críticos de éxito en la utilización de prácticas ágiles.	OE3: Establecer el nivel de influencia de los factores críticos de éxito en las prácticas ágiles en el contexto de la calidad del software.	H2: Existe una influencia significativa de los factores críticos de éxito sobre la aplicación de las prácticas ágiles durante el proceso de desarrollo de software.	Variables Factores críticos de éxito Prácticas ágiles Indicadores Nivel de influencia FCO-PWT Nivel de influencia FCO-PENG Nivel de influencia FCO-PTEST Nivel de influencia FSE-PWT Nivel de influencia FSE-PTEST Nivel de influencia FSTR-PMP Nivel de influencia FSTR - PENG Nivel de influencia FSTR – PTEST Nivel de influencia FIAI-PWT Nivel de influencia FIAI-PTEST Nivel de influencia FQWL-PTEST Nivel de influencia FCM-PWT Nivel de influencia FCM-PMP	• Método inductivo • Encuestas • Test de Cronbach • Análisis de residuos estandarizados
PE3 Falta de conocimiento de la influencia de las prácticas ágiles en las características de calidad de un software.	OE4: Determinar la influencia de las prácticas ágiles y las características de calidad del software.	H3: Existe una influencia significativa de la aplicación de las prácticas ágiles durante el proceso de desarrollo de software sobre las	Variables Prácticas ágiles Características de calidad de SW. Indicadores	• Método inductivo • Encuestas • Test de Cronbach

Problema general	Objetivos	Hipótesis	Variables e Indicadores	Métodos/ Técnicas
Problema principal	Objetivo general	Hipótesis principal		
		características de calidad del software	Nivel de influencia PWT-AFUN Nivel de influencia PMP-AFUN Nivel de influencia PWT-AFUN Nivel de influencia PMP-AFUN Nivel de influencia PENG-AFUN Nivel de influencia PENG-AMAIN Nivel de influencia PENG-ARELI Nivel de influencia PENG-AUSAB Nivel de influencia PENG-APEF Nivel de influencia PENG-ASEC Nivel de influencia PENG-ACOMP Nivel de influencia PEST-AFUN Nivel de influencia PEST-ARELI Nivel de influencia PEST-APEF Nivel de influencia PEST-APORT	Análisis de residuos estandarizados
PE4 Conocimiento limitado de la influencia de los factores críticos de éxito en las características de calidad de un software.	OE5: Establecer el nivel de influencia de los factores críticos de éxito en las características de calidad del software.	H3: Existe una influencia significativa de los factores críticos de éxito sobre las características de calidad del software	Variables Factores críticos de éxito Características de calidad de SW. Indicadores Nivel de influencia ACOMP-	• Método inductivo • Análisis de residuos estandarizados

Problema general	Objetivos	Hipótesis	Variables e Indicadores	Métodos/ Técnicas
Problema principal	Objetivo general	Hipótesis principal		
			FSTR Nivel de influencia APEF-FCM Nivel de influencia ARELI-FSTR Nivel de influencia ARELI-FCM Nivel de influencia AFUNC-FSE Nivel de influencia AFUNC-FIAI Nivel de influencia AFUNC-FCM Nivel de influencia AMAIN-FSTR Nivel de influencia APORT-FQL Nivel de influencia APORT-FSTR Nivel de influencia ASEC-FSTR Nivel de influencia AUSAB-FSTR Nivel de influencia PEST-ARELI	

1.8 Organización de la tesis

A más del presente capítulo introductorio, la tesis contiene seis capítulos, que son los siguientes:

El Capítulo 2 describe el Marco Teórico de la tesis, el mismo que ha sido organizado en tres categorías conceptuales: Metodologías y Prácticas ágiles, Factores Críticos de Éxito y Calidad de Software.

El Capítulo 3 contiene una Revisión Sistemática de Literatura sobre los aspectos de calidad, en el que se detalla los diferentes modelos de calidad, las características de calidad, prácticas, métricas aplicadas en el desarrollo ágil, que han aparecido desde la declaración del Manifiesto ágil. Además, este capítulo contiene un análisis de los elementos más utilizados en los estudios realizados, así como un estudio comparativo de los modelos encontrados.

En el Capítulo 4 se describe el aporte de esta investigación, aquí se describe el Modelo Conceptual que permite determinar la influencia de los Factores Críticos de Éxito y Prácticas Ágiles sobre la calidad del software. En este capítulo se detallan los cuatro componentes del modelo y su respectiva fundamentación.

El Capítulo 5 muestra un estudio secundario relacionado con la demostración de la primera parte del modelo propuesto. Aquí se determina la influencia de seis factores críticos de éxito en la aplicación de cuatro categorías de prácticas ágiles.

En el Capítulo 6 se determina la influencia de la aplicación de prácticas ágiles y factores críticos de éxito en las características de calidad de software, basado en el estándar ISO/IEC 25010.

Por último, el Capítulo 7, detalla las conclusiones finales, limitaciones del trabajo y trabajos futuros derivados de la presente tesis.

Los Capítulos 3, 4, 5 y 6 corresponden los aportes secundarios de esta investigación, cada uno de ellos describe su metodología de desarrollo y discusión de los resultados de

cada uno de ellos.

Para posibilitar una fácil lectura de este trabajo, se sugiere guiarse a través de la secuencia presentada en la Figura 1-5. El lector puede iniciar la lectura con el Resumen de la tesis y continuar de acuerdo al esquema presentado.

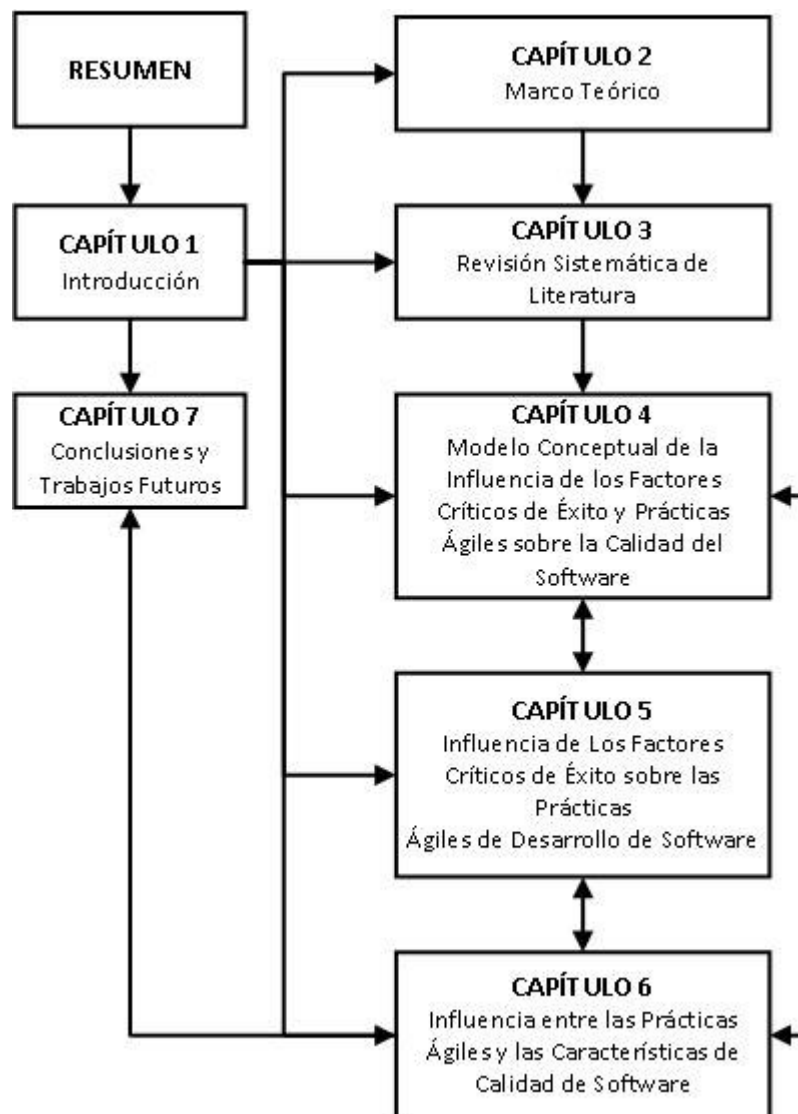


Figura 1-5 Proceso de lectura de la tesis

CAPÍTULO 2: MARCO TEÓRICO

2.1 Antecedentes del problema

“La calidad es un término que ha adquirido gran relevancia con el paso del tiempo, ya que es considerada como uno de los principales activos con los que cuenta un país para mejorar su posición competitiva global” (Ivancevich et al., 1997).

La calidad del software es “La concordancia con los requerimientos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente” (Pressman, 2010).

Varios autores han propuesto conceptos de calidad dependiendo de la perspectiva del usuario, como por ejemplo: Barry Bohem; Phil Crosby, Tom McCabe, John Musa de Laboratorios Bell (Bawane M. & C.V. Srikrishna, 2010). Según Bohem, existen dos enfoques pueden ser seguidos para asegurar la calidad del software; uno se centra en la especificación directa y evaluación de la calidad del producto de software, mientras que la otra se centra en asegurar la calidad de proceso por el cual se desarrolla el producto de software.

Para definir la calidad del software es importante diferenciar entre la calidad del producto y la calidad del proceso de desarrollo de éste (calidad de diseño y desarrollo). Sin embargo, los requerimientos no funcionales que generalmente se refieren a la calidad del producto van a guiar el cumplimiento de los objetivos del proyecto de desarrollo. La calidad del producto va a depender, entre otros aspectos, de su proceso de construcción. “*Sin un buen proceso de desarrollo es casi imposible obtener un buen producto*” (Mendoza et al., 2005).

La medición de la calidad es un elemento clave en cualquier proceso de ingeniería. Estas métricas son utilizadas para medir la calidad de los productos de la ingeniería de software y entender de mejor manera las características de los modelos creados. Debido a las características

propias del software, las medidas y métricas son en gran parte indirectas, es decir, que se calculan con base en otras medidas directas o indirectas y, por lo tanto, expuestas a contextos específicos de medición. Una métrica está estructurada en dos partes: a) definición de un método de cálculo y b) una escala asociada.

Se afirma que: “...en el software lo que se mide son atributos propios del mismo, se descompone un atributo general en otros más simples de medir, se mide con datos estadísticos no avalados, es imposible decir que la medición se hace en forma correcta” (Ferreira et al., 2006). De acuerdo con esta afirmación, para conseguir calidad del software es esencial establecer una planificación de las medidas a tomar con respecto a los proveedores o desarrolladores. Es también importante utilizar los modelos y métodos apropiados para controlar el proceso de desarrollo.

Existen diferentes formas para evaluar y certificar la calidad de productos de software, entre ellas tenemos dos tipos de estándares y normas: las que se orientan a *procesos* y las que se orientan a *productos*.

Por otra parte, en febrero de 2001, aparece el término “ágil” aplicado al desarrollo de software como una contribución de diecisiete especialistas de la industria del software, entre los que participaron varios promotores y creadores de metodologías de software. El objetivo de esta reunión de expertos fue “ *proponer los valores y principios que permitan a los equipos de trabajo desarrollar software rápidamente y respondiendo a los cambios que puedan surgir a lo largo del proyecto*” (Agile Alliance, 2001).

Además, la intención de este grupo de expertos fue brindar una alternativa diferente a los procesos de desarrollo conocidos como “tradicionales”, o como “modelos basados en un plan” que se caracterizan por ser rígidos y priorizan la documentación que se genera en cada una de las etapas propias de cada metodología. Otro hito en la evolución del concepto “ágil” fue la creación de The Agile Alliance, que promueve los conceptos, valores y principios del desarrollo de software ágil, además apoya a las organizaciones en la adopción de estos conceptos. El inicio fue el Manifiesto Ágil, conocido como el principal documento que sintetiza la filosofía ágil.

A pesar de la existencia del Manifiesto Ágil, varios autores han propuesto metodologías que tienen características propias y dan énfasis a valores y principios ágiles más específicos. Las metodologías promueven la utilización de prácticas probadas para incrementar la calidad de los procesos de desarrollo. Sus principales ventajas son: la adaptación al cambio, entregas frecuentes, documentación indispensable, simplicidad en el diseño, controles diarios y desarrollo evolutivos entre otras.

A partir de la publicación del Manifiesto Ágil, se han realizado algunos estudios relacionados con la calidad del software aplicando modelos ágiles, entre ellos se puede destacar:

- a) (Bawane M. & C.V. Srikrishna, 2010), propone un modelo de calidad basado en los componentes básicos que han sido elegidos en base a los cinco atributos de calidad: fiabilidad, facilidad de uso, de eficiencia, mantenibilidad y portabilidad.
- b) (Hashmi & Baik, 2007), presentan una comparación entre el modelo XP y el Ciclo de Vida en Espiral, en esta investigación se concluye que la tasa de fallos para las iteraciones XP es más pequeña que los de espiral, esto se debe a que una de las razones de la mayor proporción de fallo en la versión del proyecto de XP es el menor uso de la práctica de la programación en parejas propio de los modelos ágiles.
- c) (Jeon et al., 2011), propone el método ACRUM, la principal característica de la solución propuesta es que se deriva de los valores y prácticas de SCRUM para ser compatible con el proceso de Scrum y para mantener intacta su agilidad. Los resultados mostraron que ACRUM es más eficiente que el proceso de desarrollo ágil aplicando únicamente.
- d) (Hongying & Cheng, 2011), proponen el Modelo AQAM (Agile Quality Assurance Model), el cual se aplica al desarrollo de software de proyectos pequeños y de mediano tamaño; AQAM tiene 20 áreas clave del proceso; cada área clave tiene: guías, beneficios, procesos, mejores prácticas, plantillas, personalización y niveles de madurez.
- e) (N. P. Singh & Soni, 2011) proponen un nuevo modelo que combina el concepto ágil con la Metodología en Cascada que considera la documentación, retroalimentación del cliente y desarrollo iterativo de software como sus componentes. Este modelo será capaz de responder las preguntas de administración, aseguramiento de la calidad, garantizando la

correcta dirección del proyecto.

- f) El enfoque 3C para el aseguramiento de la calidad ágil (Janus et al., 2012), se basa en la Integración Continua una de las prácticas para aseguramiento de la calidad en el desarrollo ágil, a esta técnica se ha añadido la Medición Continua y la Mejora Continua, este modelo establece una base de Métricas para un aseguramiento de la calidad ágil. Este modelo fue adaptado por (Jhons et al., 2014) para que se facilite su aplicación en la metodología Scrum.

2.2 Calidad del software

La calidad del software puede ser definida como “la totalidad de características de un producto de software que tienen como habilidad satisfacer necesidades explícitas o implícitas” (ISO/IEC 9126, 2004).

Según el estándar IEEE 610.12, el aseguramiento de la calidad de software es el “*conjunto planificado y sistemático de actividades que aseguran que el proceso del ciclo de vida de desarrollo de software y sus productos se ajusten a los requisitos, normas, y procedimientos*” (IEEE SA 610.2, 1990). Este concepto, permite clasificar la calidad del software en dos grandes grupos: la calidad del producto y la calidad del proceso de desarrollo de software (Muñoz et al., 2010).

A fin de contar con un software de calidad, es necesario considerar lo siguiente:

- Los requisitos del software como base para determinar las métricas de calidad, determinados específicamente en los requisitos no funcionales del software.
- Los estándares específicos relacionados con el ciclo de vida del software, que guían la aplicación de la ingeniería del software durante su proceso de desarrollo.
- Los conjuntos de requisitos implícitos que frecuentemente no son especificados por el usuario. Es importante resaltar que si el software se ajusta a sus requisitos explícitos, pero no logra cumplir los requisitos implícitos, puede obtenerse un producto de mala calidad.
- Impacto estratégico. Oportunidad de ventaja competitiva.
- Planificación, fijación de objetivos, coordinación, formación, adaptación de toda la

organización.

2.2.1 Estándares de calidad del software

Un estándar se puede conceptualizar como:

“Una especificación normativa formal establecida en base a los resultados consolidados de la ciencia, la tecnología y la experiencia por consenso y aprobada por un organismo reconocido, que provee para uso común y repetido, reglas, lineamientos o características para las actividades o sus resultados” (ISO, 2018).

Uno de los principales objetivos de control de calidad en la ingeniería de software es verificar el cumplimiento de los estándares a través de un monitoreo permanente de los proyectos de software (Opiyo et al., 2002).

Los estándares surgen de actividades de organizaciones de estandarización oficiales o por fuerza de la práctica (estándares de facto) (Rada & Craparo, 2000); establecen procedimientos operativos para ofrecer productos y servicios de calidad.

Las ventajas de la aplicación de estándares para determinar la calidad del software son:

- Influyen positivamente en el proceso de desarrollo.
- Contribuyen a que el software se desarrolle sin ambigüedades.
- Se disminuyen las interpretaciones erróneas de cualquier aspecto del desarrollo de software.
- Proporcionan procedimientos que se estructuran en un sistema de calidad para asegurar la calidad de los requisitos, diseño, implementación, pruebas, instalación y servicio.
- Los procedimientos operativos que se incluyen en el estándar pueden ser personalizados de acuerdo a los requerimientos de la industria o requerimientos individuales a través de incremento o eliminación de determinadas secciones del estándar, según sea el caso.
- Existe una gran cantidad de estándares relacionados con la calidad del software.

En resumen, si la industria adopta estándares de calidad de software, éstos pueden mejorar la calidad del proceso y los productos de software (Opiyo et al., 2002).

Por otra parte, la Organización Internacional de Normalización (ISO) conjuntamente con el IEC, el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) a través de Computer Society, y el Instituto de Ingeniería de Software (SEI), participan activamente en la generación de estándares utilizados por los desarrolladores de software de todo el mundo. Otras numerosas organizaciones internacionales y locales y gobiernos de todo el mundo redactan varios estándares de ingeniería de software.

El Anexo A presenta los principales estándares relacionados con la calidad del software, donde se listan 30 estándares que abordan distintos aspectos de calidad, tales como:

- Procesos de aseguramiento de la calidad como: verificación y validación, revisiones y auditorías, planes de aseguramiento de la calidad y solución a problemas.
- Características de calidad como: seguridad, fiabilidad, integridad, pruebas
- Modelos de calidad
- Métricas de calidad
- Gestión de la calidad

El primer modelo de calidad de software fue propuesto por Mac Call en 1977, el cual establece tres capacidades que debe poseer un software y una serie de factores para cada capacidad; además establece métricas que permiten evaluar los factores establecidos por el modelo. Sin embargo se considera que, *“Uno de los principales estándares de calidad de producto software, es el estándar ISO 9126 en el año 1991; luego, en el año 2001, este estándar fue remplazado por los estándares: ISO/IEC 9126, que especifica características y métricas de la calidad del software; y el estándar ISO/IEC 14598, que presenta el proceso de la evaluación de los productos de software”* (Estayno et al., 2009).

El estándar ISO/IEC 9126 está compuesto por cuatro partes:

- 1) *“Modelo de calidad* (ISO/IEC 9126 -1), que considera seis características de calidad del software: funcionalidad, fiabilidad, usabilidad, eficiencia, mantenibilidad y portabilidad; cada una de ellas está subdividida en subcaracterísticas.
- 2) *Métricas externas* (ISO/IEC 9126 -2), que evalúa la calidad externa del software que se mide y se evalúa por medio de características dinámicas del código ejecutable en un software, esto es, cuando un módulo o la aplicación completa es ejecutada en un entorno real o en un ambiente simulado lo más cercano a un ambiente real. Estas métricas se obtienen en las fases finales del ciclo de vida del software (testing u operación de software) con un producto ejecutable. Estas métricas con usadas para especificar la calidad del producto.
- 3) *Métricas internas* (ISO/IEC 9126 -3), que permite evaluar la calidad externa del software, es decir, los atributos estáticos del software (especificación de requerimientos, arquitectura, diseño, código fuente, etc.), que generalmente son obtenidos en etapas tempranas del ciclo de vida del desarrollo de software. Las métricas internas permiten medir la calidad de las entregas o productos intermedios del proceso de desarrollo de software a fin de identificar problemas de calidad e iniciar acciones correctivas de manera oportuna”.
- 4) *Métricas para la calidad en uso* (ISO/IEC 9126 - 4), las métricas de calidad en el uso miden *“el grado en el que un producto cumple con las necesidades de los usuarios especificados para conseguir las metas especificadas de eficacia, productividad, seguridad y satisfacción”*.

El modelo más actual está representado por las normas “ISO 25000:2005, conocidas con el nombre de SQuaRE (Software Quality Requirements and Evaluation), basada en ISO/IEC 9126 y en ISO 14598, se divide en 5 tópicos que se presentan en la Figura 2-1.

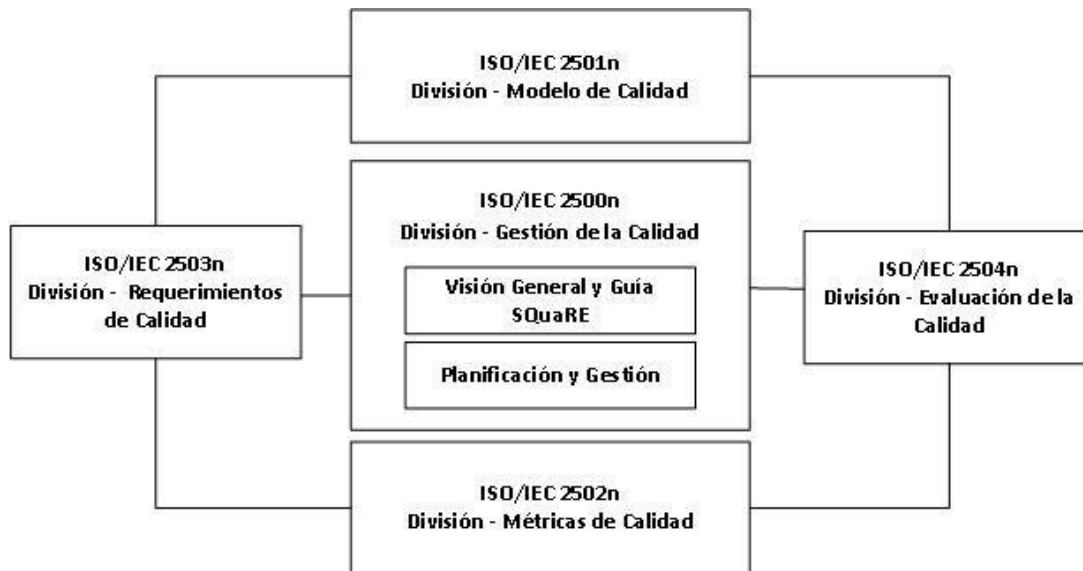


Figura 2-1 Estructura SQuaRE
Fuente: (ISO/IEC 25000:2005)

Desde 1995 aparece CMMI, Integración de modelos de madurez de capacidades o Capability Maturity Model Integration, un modelo para la mejora y evaluación de procesos para el desarrollo, mantenimiento y operación de sistemas de software (SEI, 2010).

2.2.2 Tipos de calidad de software

La calidad del software, puede ser analizada desde cuatro enfoques diferentes:

Calidad a nivel de producto: Cuando se analiza la calidad producto final del desarrollo de software, determinando el nivel de cumplimiento de las características de calidad del software, Por ejemplo, usabilidad, funcionalidad, compatibilidad, etc

Calidad a nivel de proyecto: Permite controlar todas las etapas de gestión de proyecto (planificación, monitoreo, evaluación, etc), a través de la aplicación de metodologías, instrumentos y mejores prácticas. De acuerdo con (Standish Group, 2020), “un proyecto exitoso es aquel que cumple con su alcance, presupuesto y tiempo planificado”, que son aspectos relacionados con la calidad de un proyecto.

Calidad a nivel de proceso: Cuando se gestionan las áreas de proceso del desarrollo de software que se ejecuta en una organización, de tal forma que puedan controlarse y mejorarse y consecuentemente aumentar la calidad de los productos relacionados con estos procesos. Además, durante el proceso de desarrollo de software se realizan pruebas y verificación de requerimientos en cada etapa, independientemente del modelo de proceso o metodología que se utilice para su desarrollo, a fin de detectar y corregir los posibles defectos que aparecen durante su construcción y tomar acciones correctivas de manera oportuna.

Calidad de uso: “Intenta medir las percepciones y reacciones de los usuarios, interactuando con el producto en escenarios específicos de uso” (ISO/IEC, 2005); es decir, el usuario determina la calidad del producto cuando lo ha utilizado en un entorno real y consiguen las metas esperadas en el cumplimiento de sus actividades, como por ejemplo, efectividad, productividad, satisfacción, etc.

Calidad del equipo /personas: Las personas que forman parte del equipo de desarrollo son un componente clave que determina el éxito o fracaso de un proyecto. Existen muchas propuestas para la mejora de los equipos de trabajo, que van desde el coaching a la aplicación de principios ágiles a fin de lograr la auto-organización, motivación, madurez emocional, etc.

2.3 Metodologías y principios ágiles

En la actualidad existen varios modelos de desarrollo de software, los mismos que han sido categorizados en dos grandes grupos:

Los modelos tradicionales o guiados por un plan; que se centran en un control riguroso del proceso, establecen las actividades a desarrollarse, los artefactos que se deben producir, y las herramientas, estándares y notaciones que serán utilizados.

Los modelos ágiles que “dan mayor valor al individuo, al desarrollo incremental del software con iteraciones muy cortas y a la colaboración con el cliente; este enfoque está mostrando efectividad en proyectos con requisitos que cambian frecuentemente y cuando se exige reducir

el tiempo de desarrollo, manteniendo una alta calidad” (Canós et al., 2003).

El término “ágil” aplicado al desarrollo de software surgió en 2001, en una reunión que contó con la participación de 17 especialistas en el desarrollo de software en la industria, su principal objetivo fue “establecer los principios y valores que permitan a los equipos desarrollar software de una forma rápida y que respondan a los cambios que surgen durante el ciclo de vida de un proyecto de manera rápida y oportuna” (Muñoz et al., 2010). Como resultado de esta reunión se creó The Agile Alliance, que en la actualidad promueve la aplicación de conceptos relacionados con el desarrollo de software ágil conocido como ASD por sus siglas en inglés (Agile Software Development).

Un hito importante del desarrollo ágil fue la emisión del Manifiesto Ágil (Agile Alliance, 2001), un documento que contiene los principios y valores de la filosofía ágil. Aunque los creadores e impulsores de las metodologías ágiles coinciden con sus principios, cada modelo ágil tiene sus características propias.

Los modelos ágiles constituyen un tema que está acaparando gran interés, en la actualidad existen una gran cantidad de modelos ágiles que están siendo aplicados por los desarrolladores de software; (Santos, 2011).

De acuerdo con (Letelier & Penadés, 2006), (García-Mireles et al., 2015) entre las metodologías ágiles más representativas se encuentran: Extrem Programming, Scrum, Crystal Methodologies, Dynamic Systems Development Method (DSDM), Adaptive Software Development (ASD), Adaptive Software Development (ASD), Feature-Driven Development (FDD), Lean Software Development (LSD), éstos y otros modelos ágiles se presentan en la Tabla 2-1.

Tabla 2-1. Modelos ágiles

Modelo	Descripción
XP (eXtrem Programming)	Propone un cuatro valores que se deben aplicar en un proyecto: la simplicidad, la comunicación, la retroalimentación y el coraje.

Modelo	Descripción
Scrum	Es un enfoque iterativo e incremental para la gestión de los proyectos de software en un entorno cambiante
Kanban	Método usado en el desarrollo de software ágil para visualizar el avance de las tareas que se realizan durante la producción del software.
Scrumban	Combina las mejores prácticas Scrum y Kanban
Crystal	Se centran en el desarrollo incremental, que puede ser en paralelo. Cada incremento puede tener varias iteraciones. Las metodologías Crystal son una familia de metodologías ágiles, donde cada una de ellas está adecuada para un tipo de proyecto.
FDD (Feature Driven Development)	Está basado en iteraciones cortas para el desarrollo de software. Prioriza las fases de diseño y construcción
DSDM (Dynamic System Development Method)	Proporciona un framework que apoya el desarrollo de software rápido, interactivo y colaborativo para la producción de soluciones de sistemas de información empresarial de alta calidad
ASD (Adaptative Software Development)	Ofrece un enfoque ágil y adaptable a los proyectos de alta velocidad y software de alto cambio
AUP (Agile Unified Process)	Está basado en el Modelo RUP, aplica técnicas ágiles que incluyen: desarrollo orientado a pruebas, modelado ágil, gestión de cambios ágil y refactorización de bases de datos para mejorar la productividad del software.
Lean Software Development	Es un modelo ágil que adapta los siete principios de Lean al desarrollo de software

El Manifiesto Ágil (Agile Alliance, 2001) estableció un conjunto de valores y principios para las metodologías ágiles. El manifiesto detalla cuatro valores importantes para conseguir equipos eficientes y productivos: “ 1) Los individuos e interacciones sobre procesos y herramientas, 2) Entregar software que funciona sobre documentación comprensiva, 3) Colaboración del cliente sobre negociación de contratos, y 4) Responder al cambio sobre desarrollar un plan”

Los 12 principios en los que se basa el desarrollo ágil se presentan en la Tabla 2-2.

Tabla 2-2. Principios ágiles

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

Fuente: (*Manifiesto for Agile Software Development*, 2001)

Las metodologías ágiles basan su proceso en la aplicación de prácticas ágiles, es decir, actividades o acciones orientadas al desarrollo de proyectos altamente productivos. Scrum y Scrumban, son las más utilizadas en el desarrollo de software ágil (Versionone, 2020). Scrum, Scrumban y una metodología híbrida entre XP y Scrum son aplicadas por el 76% de los desarrolladores.

Las metodologías utilizan varias prácticas ágiles, por ejemplo (Xu, 2009), cita 12 prácticas ágiles para XP: "planning game, pequeñas versiones, metáforas, diseño simple, testing, refactoring, pair programming, propiedad colectiva del código, integración continua, 40-horas

por semana, Cliente on-site, estándares de codificación”. Las principales prácticas ágiles utilizadas por Scrum (Schwaber & y Jeff Sutherland, 2013) son: el desarrollo iterativo e incremental, planificación del proyecto, team empowerment, el control del progreso del proyecto orientado a tareas, gestión de cambios, las retrospectivas y los análisis “post mortem” al final de cada iteración, uso de timeboxing para todas las actividades de Scrum.

2.4 Factores críticos de éxito

Se puede identificar una amplia variedad de interpretaciones para el significado del éxito en el desarrollo de software (Mirza et al., 2013), que conducen a una diversidad de variables descriptivas para este fenómeno. El éxito en el desarrollo de software ha sido evaluado en función del tiempo, el alcance, el costo y la calidad (Wit, 1988), (Agarwal & Rathod, 2006), (Chow & Cao, 2008), (Kerzner, 2017).

Los autores (Chow & Cao, 2008) mencionan como factores críticos de éxito, a los aspectos que deben estar presentes para que un proyecto ágil sea exitoso. Por otra parte, (Boghossian, 2002) divide estos factores en dimensiones que consideran las etapas de desarrollo de software, la planificación de proyectos a nivel estratégico y sus respectivos recursos, la estimación y gestión de proyectos.

En la literatura existente se han identificado diversos trabajos que presentan varios factores que influyen en el éxito de un proyecto de desarrollo de software basados en metodologías ágiles, entre los cuales se puede mencionar: Chow & Cao recopilan 36 factores de éxito reportados en la literatura ágil y realiza un análisis factorial y consolida 12 factores en cinco diferentes categorías: organización, personas, procesos, técnicas, y proyecto.

El Chaos Manifiesto 2015, realiza un estudio de los 10 principales factores de éxito en proyectos de desarrollo de software; en él se consideran los siguientes factores, organizados por su importancia; a) Soporte de los Directores ejecutivos, b) Involucramiento del usuario, c) Optimización, d) Recursos especializados, e) Experticia en la gestión de proyectos, f) Proceso ágil, g) Objetivos del negocio claros, h) Madurez emocional, i) Ejecución, j) Herramientas e

infraestructura; según este estudio, los primeros cinco factores de éxito se centran en la ejecución de pequeñas habilidades de proyectos, y proporcionan el mayor beneficio para el éxito.

(Rico, 2008), propone un framework que se basa en cuatro prácticas principales de los métodos ágiles identificados a partir de su análisis: a) desarrollo iterativo, b) la retroalimentación del cliente, c) equipos bien estructurados, y d) flexibilidad; la propuesta está basada en el análisis de nueve principales métodos ágiles.

(Bermejo et al., 2014), realizan un análisis del uso de los principios ágiles que contribuyen al éxito de proyectos de software en organizaciones brasileñas, enfocados en 6 factores principales: a) Capacidad del equipo, b) éxito en el desarrollo de software, c) relación con los proveedores externos, d) cultura, e) Comunicación con el cliente, y f) Configuración del ambiente.

(Clarke & O'Connor, 2012), propone un framework de 8 clasificadores y 44 factores y 170 subfactores de éxito; los resultados de esta investigación pretenden ser un referente para planificar procesos de desarrollo de software. Los clasificadores considerados por este autor son: organización, negocio, aplicación, gestión, requisitos, tecnología, personal y operación.

(Subramanyam & Prasad, 2013), establece algunas prácticas organizacionales que afectan a la calidad del software, estas prácticas están controladas por métricas y herramientas utilizadas en el desarrollo de software. Estas prácticas son las siguientes: Asignación de los recursos humanos, mantenimiento de documentos y otros recursos, dependencia con el equipo principal y cambio continuo de los jefes de proyecto.

(Acuña et al., 2009), basado en *The Social Psychology of Innovation in Groups* de (Michael A. West & James L. Farr, 1990), analizan el clima de trabajo dentro de un equipo de desarrollo de software y su impacto en la calidad de software; los factores humanos identificados por la autora son los siguientes: “seguridad participativa: la confianza que los miembros del equipo que participan sienten que hay dentro del grupo al explicar sus opiniones e ideas; visión del

equipo, apoyo a la innovación y orientación a la tarea”.

El estudio, (Acuña et al., 2008) está basado en factores humanos que contribuyen a la calidad de software, indica que la personalidad, cohesión y conflictos del equipo, interdependencia y autonomía, se relacionan con la satisfacción laboral y la calidad de software.

(Fernández-Sanz & Misra, 2011) afirma que los factores humanos, han llamado poco la atención a pesar de que el desarrollo de software depende de la mano de obra y productividad de los profesionales influyen de gran manera; el autor indica que los principales factores humanos que deben considerarse en el desarrollo de software son: a) experiencia del desarrollador, b) Problemas de experiencia del jefe de equipo, c) oficina, d) no pago de horas extras, e) moral, f) organización, g) capacitación anual, g) trabajo a presión.

(Gorla & Lin, 2010) proponen 3 tipos de factores, individuales, tecnológicos, organizacionales y 24 factores distribuidos en cada tipo.

(Santos, 2011), indica que las metodologías ágiles tienen valores y prácticas similares, las mismas que son aplicadas a similares contextos durante las fases de un proyecto de software; ellos consolidan las prácticas ágiles aplicadas en cada etapa de desarrollo de un proyecto de software ágil. En este mismo estudio, los autores evalúan 42 prácticas ágiles en función de la calidad, basada en el criterio de percepción de los involucrados en el proceso.

CAPÍTULO 3: REVISIÓN SISTEMÁTICA DE LITERATURA

3.1 Introducción

En este capítulo se detalla la Revisión Sistemática de Literatura (RSL), con respecto a los principales aspectos de calidad relacionados con la aplicación de las metodologías ágiles de desarrollo de software, como son: factores críticos de éxito que influyen en el éxito de los proyectos ágiles, prácticas ágiles, modelos de calidad ágiles métricas de calidad y atributos de calidad en proyectos ágiles. La RSL fue desarrollada mediante la adaptación de la metodología de Bárbara Kitchenham y dio como resultado un catálogo de los aspectos analizados.

3.2 Metodología para la revisión sistemática

La revisión sistemática de literatura se realizó con base en el procedimiento presentado por (Kitchenham & Charters, 2007), que consta de tres etapas; las mismas que han sido adaptadas para realizar esta RSL. Estas etapas son:

- a) **Planificación de la revisión:** en esta etapa se plantean las preguntas de investigación y se determina el protocolo que se aplicará en la revisión.
- b) **Conducción de la revisión:** en esta etapa se ejecutan las acciones planificadas y se seleccionan los estudios primarios que cumplen con los criterios de inclusión y exclusión establecidos, además, se evalúa la calidad de los estudios seleccionados.
- c) **Reporte de la revisión:** muestra los resultados del análisis de la revisión que son presentados en las secciones 3.3 y 4 respectivamente.

3.3 Planificación de la revision

Esta RSL nos permitirá contestar siete preguntas de investigación con base en la literatura existente. Estas preguntas contribuirán a determinar cuáles son los aspectos de calidad que han sido desarrollados para el aseguramiento de la calidad en el proceso de desarrollo de software

con metodologías ágiles.

RQ1: ¿Cuáles son los factores que influyen en el éxito de un proyecto de desarrollo ágil?

RQ2: ¿Cuáles son las métricas utilizadas para evaluar la calidad del software?

RQ3: ¿Cuáles son las prácticas ágiles que influyen en la calidad de software?

RQ4: ¿Cuáles son los modelos de calidad orientados proceso de desarrollo ágil ?

RQ5: ¿Cuáles son los principios ágiles que utilizan los modelos encontrados?

RQ6: ¿Cuáles son las características de calidad que se utilizan en los modelos encontrados?

RQ7: ¿Cuáles son los aspectos de calidad que están contemplados en los modelos analizados?

La búsqueda de la información que permitió responder las preguntas planteadas se realizaron en bases de datos como: IEEE Xplore, ACM Digital Library, Springer, Science Direct y DOAJ (Directory of Open Access Journal); la investigación cubre el período comprendido desde el año 2001 hasta el año 2022, se ha escogido el 2001 porque a partir de este año se establece la filosofía ágil para el desarrollo de software.

Los estudios que han sido considerados en esta investigación, responden al siguiente criterio de búsqueda:

(("Software Quality" OR CMMI OR "Software Assurance Quality" OR "Quality Assessment Software" OR "Quality Evaluation Software") AND (Factors OR Determinants OR Practices OR Principles OR Metrics OR Measure OR Attributes OR Models OR Frameworks OR Approach OR Patterns) AND (Agile OR "Agile Software Development")).

Esta cadena de búsqueda ha sido aplicada al título, abstract y keywords de cada paper potencial; además, estos términos fueron adaptados para que respondan a las preguntas de investigación y las especificaciones establecidas en cada uno de los motores de búsqueda de las bases de datos analizadas. Una vez ubicados los estudios primarios se consideraron los criterios de selección y exclusión que se presentan en la Tabla 3-1.

Las fuentes de búsqueda consideradas fueron: journals, proceedings y workshops, los

mismos que fueron sujetos a una evaluación de calidad.

Tabla 3-1 Criterios de selección y exclusión

Criterios de Selección	Criterios de Exclusión
• Presentan modelos o enfoques orientados a la calidad de software para el desarrollo ágil. • Proponen características y/o factores de calidad para el desarrollo ágil. • Proponen métricas para la medición de la calidad en el desarrollo ágil. • Respondan directamente las preguntas de investigación. • Estudios empíricos y teóricos.	• Estudios que no cumplen con los objetivos de este trabajo. • Papers que no contribuyen con aspectos analizados en este estudio. • Papers que no tienen un contexto ágil • Papers que mencionan la calidad pero no detallan sus componentes. • Posters, editoriales y tesis. • Reportes de opinión y experiencias.

3.4 Conducción de la revisión

Los estudios primarios potenciales encontrados aplicando el proceso de búsqueda establecido en la etapa de planificación fueron analizados con base en los criterios de inclusión y exclusión indicados anteriormente. Para una correcta selección se realizó una revisión preliminar al contenido de cada estudio a fin de determinar si están acorde con las preguntas de investigación de este estudio y sobre todo para identificar si estos estudios se aplican a los modelos de desarrollo ágiles. El proceso aplicado para la selección de los estudios y los resultados obtenidos en cada paso del proceso, se representan en la Figura 3-1. Posteriormente se procedió a analizar el contenido de cada artículo para dar contestación a las preguntas de investigación.

Luego del proceso de ejecución de la RSL se encontraron 1.208 estudios primarios, de los cuales 111 estudios fueron seleccionados porque responden a las preguntas de investigación planteadas, así como también, están relacionados con los aspectos de calidad que son aplicados a las metodologías ágiles de desarrollo de software (ver Tabla 3-2). Los estudios incluidos en el estudio se presentan en el Anexo B, y han sido identificados como [S1]... [S111].

Tabla 3-2 Estudios potenciales y estudios seleccionados

Fuentes	Estudios potenciales	Estudios seleccionados
ACM Digital Library	340	20
IEEE Xplore	954	56
Science Direct	221	13
Springer	374	22
Total	1889	111

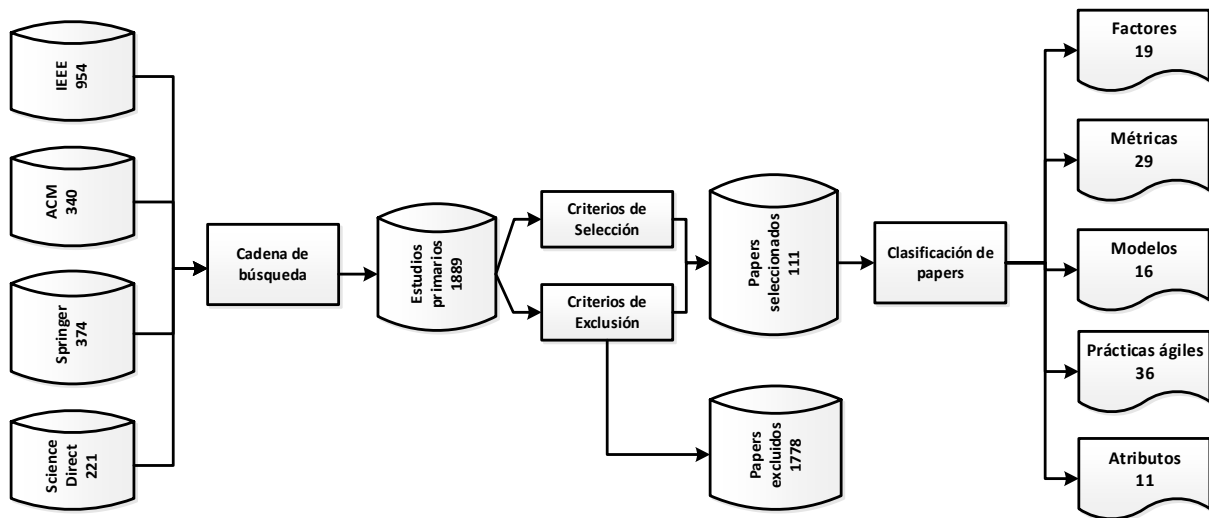


Figura 3-1. Proceso aplicado para la RSL

3.5 Evaluación de la calidad de los estudios primarios

Con el fin de determinar la calidad de los estudios primarios se utilizó la metodología aplicada por (Montagud et al., 2012), en la que cada estudio fue valorado en función de dos aspectos:

- 1) Relevancia del journal, conferencia, workshop donde el paper fue publicado y
- 2) Número de citaciones del paper.

Estos aspectos no fueron considerados en los criterios de exclusión de la revisión sistemática, porque su objetivo fue reunir todos los aspectos importantes para el desarrollo ágil y no era adecuado eliminar estudios que incluían esta información.

Para determinar la **relevancia** de la conferencia o journal donde el paper fue publicado,

los estudios fueron clasificados en tres categorías: “muy relevantes”, “relevantes” y “no muy relevantes”. Cuando se trataba de una conferencia, se consideraron las conferencias tipo A, B y C del CORE conference ranking (Computing Research & Education, 2022) and Scimago citations report (SJR) para el caso de las publicaciones en journals. La Tabla 3-3 muestra el criterio de clasificación de la relevancia de los papers analizados.

Tabla 3-3 Criterio de clasificación de la relevancia de los papers analizados

Descripción	Evaluación cualitativa	Evaluación cuantitativa
Publicaciones en conferencias calificadas como A en la clasificación CORE. Publicaciones en revistas incluidas en SJR.	Muy relevante	10
Publicaciones en conferencias calificadas como B = 6 y C = 4 en la clasificación CORE. Publicaciones en revistas no incluidas en SJR.	Relevante	5
Publicaciones en conferencias no indexadas en la clasificación CORE pero cuenta con indexaciones distintas a CORE.	No tan relevante	2

Con respecto al *número de citaciones* del paper, se realizó una nueva clasificación de los los papers dividiéndoles en tres categorías “alta”, “media” y “baja”. Esta evaluación se realizó con base en los resultados obtenidos en Google Scholar Citation Count. La calificación de los paper se realizó con criterio que se muestra en la Tabla 3-4.

Tabla 3-4 Criterio de clasificación del número de citaciones de los papers analizados.

Año de publicación	Número de citas	Evaluación cualitativa	Evaluación cuantitativa
Antes del 2017	Papers con más de 5 citas	Alta	10
	Papers con 1 – 5 citas	Media	5
	Papers no citados	Baja	2
En o después del 2017	Papers citados	Alto	10
	Papers no citados	Medio	5

Los papers publicados antes del 2017 fueron calificados de forma diferente al resto de papers; la razón es no penalizar a publicaciones recientes.

Para cada paper se calcularon estos dos aspectos, obteniéndose un promedio de 84,67% y 98,20% para la relevancia y citas respectivamente. Estos resultados evidencian que los estudios seleccionados tienen un alto nivel calidad. El Anexo C, presenta la evaluación de la calidad de los papers seleccionados.

3.6 Reporte de la revisión

3.6.1 Evolución de las publicaciones

La Figura 3-2 muestra que en los últimos 21 años ha existido un incremento de las publicaciones que cubren el área de la calidad de software aplicada al desarrollo ágil. Este incremento puede ser un reflejo de la importancia que los investigadores y profesionales han dado al tema de la calidad y a la adopción de las metodologías ágiles en las empresas. De acuerdo con los resultados presentados, se puede determinar que los primeros estudios relacionados con los aspectos de calidad surgen a partir del año 2004. Se nota una baja de publicaciones en el 2020, posiblemente debido al confinamiento por el Covid a nivel mundial.

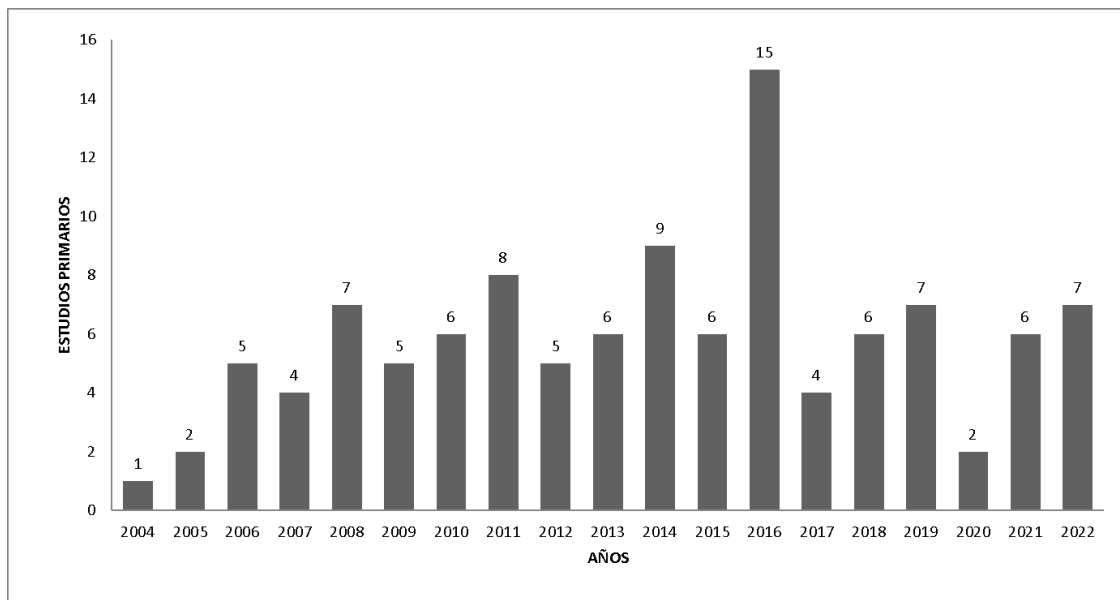


Figura 3-2 Evolución de publicaciones acerca de la calidad aplicado al desarrollo ágil

3.6.2 Fuentes de datos

Los resultados de la búsqueda evidencian que la mayor cantidad de estudios relacionados con la calidad del software han sido publicados en la base de datos IEEE, a través de journals, proceedings, symposiums y workshops; el 50,45% de todas las publicaciones corresponde a IEEE (ver Tabla 3-5). Es importante notar que 78,38% de los estudios publicados corresponden a proceedings, symposiums y workshops, y únicamente el 21,62% de los estudios han sido publicados en journals.

Tabla 3-5 Fuentes de las publicaciones sobre calidad en metodologías ágiles

	ACM	IEEE	Science Direct	Springer	Total	Porcentaje
Journals	0	7	10	7	24	21,62%
Proceedings	16	45	3	15	79	71,17%
Symposium	2	1	0	0	3	2,70%
Workshops	3	2	0	0	5	4,51%
Total	21	55	13	22	111	100,00%

3.6.3 Clasificación de los estudios

En la Tabla 3-6 se presentan las 111 investigaciones clasificadas en cinco grandes los aspectos estudiados, se muestra además las preguntas de investigación con las que se relaciona cada uno de ellos. Es importante aclarar que algunos estudios contribuyen con más de un aspecto de calidad, cuando estos casos se presentan, se los ha clasificado dentro del aspecto más relevante considerado en el paper.

Como se puede ver, el 32,43% de los estudios corresponden a prácticas ágiles; lo que nos lleva a concluir que éstos son los aspectos más estudiados de la calidad de software cuando se usan metodologías ágiles para su desarrollo. Así mismo, se puede determinar que el aspecto menos estudiado es el relacionado con las características o atributos de calidad con un 9,9%.

Los estudios analizados para responder a RQ4 (modelos de calidad), contribuyeron al análisis de las RQ5 y RQ7, es decir a la determinación de los principios ágiles y los aspectos de calidad que se aplican en los modelos de calidad estudiados.

Tabla 3-6 Clasificación de los estudios encontrados en RSL acerca de la aplicación de la calidad en el desarrollo de proyectos ágiles.

RQ	Aspectos de Calidad	IEEE	Science Direct	Springer	ACM	Total
RQ1	Factores	[S1], [S72],[S82]	[S2],[S3],[S5] [S6],[S7],[S5], [S74]	[S8],[S9],[S10], [S75],[S76]	[S11],[S12],[S14] [S73]	19
RQ2	Métricas	[S15],[S16],[S17],[S18], [S19], [S20],[S21],[S22], [S70],[S51], [S95],[S96], [S104]	[S25],[S111]	[S26],[S27],[S28], [S29], [S97],[S98], [S100],[S102] [S103]	[S23],[S30],[S31], [S24], [S101]	29
RQ3	Prácticas ágiles	[S32],[S34],[S35], [S36], [S37],[S38],[S39], [S40], [S41],[S42], [S43],[S44], [S45], [S46],[S47],[S71], [S52],[S77],[S78], [S79], [S80], [S81],[S84],[S86]	[S4], [S48]	[S87], [S88],[S93], [S94]	[S49], [S89], [S90], [S91],[S92],[S99]	36
RQ4, RQ5, RQ7	Modelos	[S53],[S54],[S55],[S56], [S57],[S58],[S59],[S60], [S83], [S85]	[S61]	[S62],[S63]	[S64],[S65], [S33]	16
RQ6	Atributos	[S66], [S67],[S13], [S105],[S106]	[S107]	[S68],[S108],[S109]	[S69],[S110]	11
Total		55	13	23	20	111

3.7 Análisis

En esta sección se responden a las preguntas de investigación planteadas en el ítem 3.3. Los resultados se presentan en tablas que contienen el aspecto a analizar, su cantidad, fuentes que lo referencian y la frecuencia que este aspecto es referenciado.

3.7.1 RQ1: ¿Cuáles son los factores que influyen en el éxito de un proyecto de desarrollo ágil?

La literatura existente ha evidenciado la existencia de 148 factores críticos de éxito, que han sido clasificados en las cinco dimensiones propuestas por (Chow & Cao, 2008): *organizativo*, *personal*, *técnico*, *proyecto*, y *proceso*; a éstas se les ha incrementado dos dimensiones considerados por (Rico, 2008), *desarrollo iterativo* y *feedback del cliente*. En el Anexo D, se listan los factores encontrados, de acuerdo a la dimensión a la que pertenecen. Un resumen del número de factores y sus frecuencias, clasificados por dimensiones, se presenta en la Tabla 3-7.

Tabla 3-7 Factores críticos de éxito de un proyecto de desarrollo ágil

No.	Dimensión	Número de Factores	Frecuencia	Referencias
1	Organizacional	41	58	[S2],[S3],[S4],[S6],[S7],[S8],[S9],[S10],[S11],[S14],[S50],[S72],[S74],[S75],[S76]
2	Personas	46	71	[S1],[S2],[S3],[S5],[S6],[S7],[S8],[S9],[S10],[S11],[S14],[S71],[S50],[S13],[S72],[S74],[S75],[S108]
3	Proceso	10	12	[S2],[S3],[S10],[S72],[S75],[S104]
4	Técnicos	26	33	[S1],[S2],[S3],[S10],[S14],[S50],[S72],[S75],[S110]
5	Proyecto	15	22	[S1],[S2],[S3],[S6],[S14],[S71],[S72]
6	Desarrollo iterativo	5	5	[S1],[S97]
7	Retroalimentación del cliente	5	5	[S1]
Total		148	206	

Los resultados muestran que la dimensión *Personas* contiene el mayor número de factores, representando el 31,08% de todos los factores encontrados; seguido por los factores *Organizacionales* que corresponden a un 27,70%. Los factores *Técnicos* también han sido considerados por el 17,57% de todos los factores que corresponden a este aspecto. Estos resultados son consistentes con los principios ágiles.

Los factores más frecuentemente referenciados por los autores son: *miembros del equipo con gran motivación*, y *habilidad y capacidad del equipo de trabajo*, que corresponden la dimensión *Personas*. El factor *Técnico* más referenciado es *Capacitación técnica apropiada para el equipo*, que también está relacionado con la dimensión *Personas*.

3.7.2 RQ2: ¿Cuáles son las métricas utilizadas para evaluar la calidad del software?

La Tabla 3-8 presenta ocho categorías de métricas, las mismas que agrupan a 137 métricas referenciadas en las investigaciones consideradas en este estudio, en estas categoría se

contemplan 3 suits: CK (Chidamber & Kemerer, 1994), MOOD (Brito e Abreu & Melo, 1996), y QMOOD (Bansiya & Davis, 2002), que están asociadas a la determinación de la calidad en la etapa del diseño de software orientado a objetos; además de éstas se han realizado otras propuestas de métricas de software que están siendo utilizadas para determinar la calidad del software. El Anexo E muestra la lista de las principales métricas referenciadas en los estudios primarios.

Tabla 3-8 Métricas aplicadas en los estudios seleccionados

No.	Categorías de métricas	Métricas	Frecuencia	Referencias
1	Métricas de uso	15	17	[S16],[S22],[S30],[S31],[S48], [S92], [S95],[S97], [S98],[S101], [S103]
2	Métricas de Calidad	6	6	[S30], [S102]
3	Métricas del proyecto	49	70	[S11],[S16],[S17], [S19],[S21], [S22],[S23],[S24],[S25],[S26], S27], [S28],[S29],[S31],[S32],[S48], [S82], [S98],[S101], [S102], [S103]
4	Métricas de testing	29	45	[S11],[S16],[S17],[S19],[S21], [S25],[S27], [S31], [S32], [S48], [S70],[S51], [S70], [S95], [S97], [S98], [S101], [S102]
5	Suit de métricas CK	6	35	[S18],[S26] [S27], [S28], [S29], [S32], [S98]
6	Suit de métricas MOOD	4	4	[S18]
7	Suit de métricas QMOOD	11	22	[S18], [S20], [S29], [S98]
8	Métricas de efectividad del equipo	17	28	[S15],[S19],[S31], [S97], [S95], [S98], [S101], [S102], [S103]
Total		137	227	

La mayor cantidad de métricas encontradas corresponden a la categoría de *métricas del proyecto* que representan el 35,77%, seguidas por las *métricas de testing* con un 21,7%. La métrica SLOC (Source Lines of Code), que corresponde a la categoría *métricas del proyecto*, es la más referenciadas en los estudios encontrados; seguida por las métricas CBO, DIT, NOC, RFC que en inglés significan “*coupling between object classes, depht of inheritance tree,*

number of children, response for a class” que corresponde a la suite CK Metrics.

3.7.3 RQ3: ¿Cuáles son las prácticas ágiles que influyen en la calidad de software?

A fin de dar contestación a esta pregunta, las prácticas ágiles han sido divididas en cinco categorías con el fin de facilitar su análisis (Ver Tabla 3-9), que son las siguientes:

- *Prácticas de Equipo de Trabajo*, son aquellas que se relacionan con la forma de trabajo y características del equipo que participa en un proyecto ágil.
- *Prácticas de Ingeniería* que incluyen las prácticas que permiten construir un buen diseño y codificación del software.
- *Prácticas de Documentación* que incorporan las prácticas utilizadas para documentar adecuadamente las etapas del desarrollo de software ágil.
- *Prácticas de Gestión* que agrupan las prácticas relacionadas al control y seguimiento del proyecto, derivadas de las funciones del líder del proyecto y la cultura organizacional.
- *Prácticas de Testing*, contienen las prácticas ágiles que son importantes para garantizar la calidad del software en su etapa de desarrollo.

El Anexo F presenta una lista de prácticas ágiles clasificadas por categorías.

Tabla 3-9 Prácticas ágiles aplicadas en los estudios encontrados

Categorías de prácticas ágiles	Número de prácticas	Frecuencia	Referencias
Prácticas del Equipo de Trabajo	48	138	[S4],[S34],[S35],[S36],[S38],[S39],[S40],[S41],[S42],[S43],[S44],[S45],[S46],[S47],[S48],[S49],[S71],[S52],[S78],[S80],[S81],[S82],[S84],[S85],[S89],[S90],[S92],[S93],[S94],[S102],[S108]
Prácticas de Ingeniería	41	84	[S4],[S34],[S35],[S36],[S37],[S38],[S39],[S44],[S45],[S46],[S47],[S48],[S49],[S71],[S52],[S13],[S71],[S78],[S80],[S81],[S82],[S84],[S85],[S89],[S90],[S93]
Prácticas de Documentación	16	18	[S4],[S34],[S40],[S44],[S45],[S46],[S47],[S48],[S85],[S110]
Prácticas de Gestión	37	123	[S34],[S35],[S37],[S38],[S39],[S41],[S42],[S43],[S44],[S45],[S46],[S47],[S48],[S49],[S71],[S52],[S71],[S78],[S80],[S81],[S82],[S84],[S85],[S89],

Categorías de prácticas ágiles	Número de prácticas	Frecuencia	Referencias
			[S90],[S92], [S93] [S94], [S97], [S98], [S108]
Prácticas de Testing	23	69	[S34],[S35],[S36],[S37],[S39],[S40],[S41],[S42],[S44],[S46],[S47],[S48],[S49],[S71],[S52],[S71],[S77],[S80],[S82],[S84],[S85],[S89],[S90], [S93], [S94]
Total	165	432	

Los resultados muestran la categoría de *Equipo de Trabajo* contiene la mayor cantidad de prácticas, que corresponden al 29,09% de todas las prácticas encontradas; seguidas por las *Prácticas de Ingeniería* del proyecto que representan el 24,85%.

Las prácticas referenciadas más frecuentemente por los autores son: *Reuniones diarias*, *Programación en pareja*, *cliente on-site*, *agilidad en la comunicación con el equipo* y frecuencia de las iteraciones con los clientes, que corresponden a la categoría *Equipo de Trabajo*; en lo que concierne a las *Prácticas de Ingeniería*, las más mencionadas por los autores son: *diseño simple*, *historias de usuario* y *estándares de codificación*. Es importante recalcar que la práctica más referenciada, considerando todas las categorías es: *Retrospectivas/Retroalimentación*, correspondiente a las *Prácticas de Gestión*.

3.7.4 RQ4: ¿Cuáles son los modelos de calidad orientados proceso de desarrollo ágil?

Catorce son los aportes encontrados relacionados con modelos de calidad orientados al desarrollo ágil, los mismos que se describen en la Tabla 3-10.

Tabla 3-10 Modelos de calidad aplicados al desarrollo ágil.

Id	Autor	Año	Propuesta	Ámbito del modelo
MD1	Sampaio [S62]	2004	Evaluación de Métodos ágiles: Un estudio empírico.	Propone un método ágil para el desarrollo de aplicaciones basadas en la web (XWebProcess)
MD2	Nord & Tomayko [S54]	2006	Método centrado en la arquitectura de software y métodos ágiles	Incluye un método de análisis y diseño para XP que ayuda a abordar los atributos de calidad de forma explícita y metodológica.
MD3	Hongying & Cheng [S55]	2011	Un modelo de garantía de calidad de software ágil personalizable	Propone el Modelo Agile Quality Assurance Modelo (AQAM) para aplicarlos en proyectos de desarrollo de software pequeños y medianos. AQAM tiene 20 Áreas clave de procesos (KPA). Cada KPA

Id	Autor	Año	Propuesta	Ámbito del modelo
				tiene: plantillas, beneficios, procesos, guías, mejores prácticas, personalización y niveles de madurez. Este modelo permite a las empresas personalizar modelo de procesos basado en el nivel de madurez que se desea alcanzar, su proceso de control de calidad y su propio estilo de gestión.
MD4	Desharnais et al. [S64]	2011	Utilización del Método COSMIC para evaluar la calidad de la documentación de historias de usuario ágiles.	Utiliza el método COSMIC para analizar y obtener informes sobre la calidad de la documentación de las historias del usuario.
MD5	Um et al. [S56]	2011	Método para la evaluación de atributos de calidad para un enfoque ágil.	Es una propuesta para mejorar los aspectos no funcionales (requisitos no funcionales) a través de características de calidad. El método propuesto ayuda a identificar aquellos aspectos que se oponen a la consecución de la calidad en las fases inicial de un proyecto; para ello, se categorizan los atributos de calidad.
MD6	Jeon, et al. [S65]	2011	Desarrollo ágil controlado por atributos de calidad	Se realiza la propuesta del Modelo ACRUM; que se basa en los valores y prácticas de Scrum, respeta el proceso de este modelo y considera los aspectos de agilidad. ACRUM propone tres actividades que se ejecutan en paralelo con los sprints que se considera en Scrum: 1) AQUA - Análisis de atributos de calidad, 2) RAM - Mapa de correlación RAM entre requisitos y 3) VAQ - Atributos de calidad.
MD7	Singh et al. [S63]	2011	Software Ágil: Asegurando la garantía de calidad y el proceso	Propone un modelo para el desarrollo de software ágil que puede asegurar la calidad del software, ser auditable y centrado en el proceso. El modelo considera la documentación, los comentarios de los clientes y el desarrollo de software iterativo como sus componentes. Según el autor: “Los conceptos ágiles combinados con el modelo de metodología de cascada conducen a resultados mucho más estables”.
MD8	Janus et al. [S60]	2012	Enfoque 3C para aseguramiento de la calidad ágil.	Consiste en equilibrar la Integración Continua, con la Medición Continua y la Mejora Continua. La propuesta interpreta las medidas y la planificación de acciones de mejora para conseguir la calidad interna del software. Además, establece algunas métricas para garantizar la calidad del software. Este modelo fue adaptado por (Jisha John, 2014) para que pueda ser utilizado con la metodología Scrum.
MD9	Ayed et al. [S57]	2012	Un enfoque basado en metamodelos para personalizar y evaluar métodos ágiles	Propone un enfoque general cuyo objetivo es ayudar a construir y refinar métodos ágiles personalizados basados mediciones de sus procesos. El enfoque se basa en un

Id	Autor	Año	Propuesta	Ámbito del modelo
				metamodelo que integra conceptos metodológicos (compartidos entre OPF, SPEM y SMSDM) y conceptos de evaluación de calidad de MoCQA.
MD10	Jinzenji et al. [S59]	2013	Un reporte de experiencias para la evaluación de la calidad del software en una metodología de desarrollo altamente iterativo utilizando métricas tradicionales	Propone un framework de métricas para evaluar la calidad del software y luego lo prueba experimentalmente en el desarrollo de software iterativo.
MD11	Ayed & Vanderose [S58]	2013	AM-QuICk: Un marco basado en mediciones para la personalización de métodos ágiles	Propone un framework llamado AM-QuICk, el mismo que está basado en las medidas de los modelos ágiles. AM-QuICk consta de tres ciclos, cada uno de ellos corresponde a un nivel: 1) Alto nivel: es el primer ciclo y está bajo la responsabilidad estratégica de la organización. 2) Nivel medio, considera el proceso de refinamiento que se ejecuta paralelamente durante la ejecución de los procesos. 3) Nivel Bajo, se refiere a la funcionalidad y evolución del producto.
MD12	Tarhan et al. [S61]	2013	Análisis sistemático y comparación del rendimiento de desarrollo y la calidad del producto del proceso incremental y el proceso ágil	Propone un análisis para comparar el rendimiento y la calidad de un producto entre un proceso incremental basado en una adaptación del modelo cascada y un proceso ágil que combina el proceso Unificado de desarrollo de software, eXtrem Programming y Scrum
MD13	Selleri et al. [S53]	2014	Un modelo de referencia para la garantía de calidad ágil: combinación de metodologías ágiles y modelos de madurez	Describe una propuesta para evaluar, sintetizar y presentar los resultados de la aplicación de CMMI conjuntamente con el desarrollo de software ágil.
MD14	Forouzani et al. [S33]	2016	Método para evaluar la calidad del software utilizando el análisis del código fuente	El Método ASQuS (Assessing Software Quality using Source code) propone evaluar la calidad del código fuente utilizando dos perspectivas: 1) Violación de las reglas y 2) Elementos estructurales.
MD15	Jain et al. [S106]	2018	Metodología ágil centrada en el cliente	Se propone un modelo de cinco fases: Análisis de requerimientos, Diseño, Implementación, pruebas y Mantenimiento del software> Estas fases afectan a la calidad de software y pueden ser analizadas con varios atributos de calidad aplicados al proceso ágil. Se propone atributos ágiles y atributos de calidad para cada fase de proceso de desarrollo.
MD16	Kula et.al [S72]	2022	Modelo Conceptual para entregas a tiempo en modelos ágiles	El modelo presenta 25 factores organizadas en 5 dimensiones (Proyecto, Técnicas, Personas, Proceso y Organizacional) que influyen la puntualidad de la entrega ágil.

A continuación, se describe brevemente los modelos de calidad, relacionados directamente con el desarrollo ágil.

Modelo: A Customizable Agile Software Quality Assurance Model

Autores: Gu Hongying , Yang Cheng

Motivación de los autores

El estudio de los autores (Hongying & Cheng, 2011), afirma que los modelos CMM (Caputo, 1998), CMMI (SEI, 2010) y (Ilene Burnstein et al., 1998), son generalmente más adecuados para organizaciones de gran tamaño. Por otra parte, solamente una pequeña parte de la CMM y CMMI cubren el aseguramiento de la calidad del software. TPI (Test Process Improvement) (Jari Andersin, 2004), se centra en las pruebas de software, y no en otras actividades de aseguramiento de la calidad del software, tales como la prevención de defectos, corrección de defectos o revisión por pares que se considera muy importante la garantía de calidad del software. Por otro lado, los autores mencionan que la eficiencia de los modelos ágiles se debe al énfasis en la comunicación del equipo de trabajo, por lo que es necesario centrarse en asegurar la calidad de los modelos ágiles, la misma que debe personalizarse de acuerdo al tamaño del proyecto y a los estilos de gestión.

Descripción del modelo

El modelo propuesto se conoce con el nombre de AQAM (Agile Quality Assurance Model), el cual se aplica al desarrollo de proyectos pequeños y de mediano tamaño. AQAM identifica las siguientes KPAs: Gestión de requerimientos, Auditoría, Planificación y gestión de casos de prueba, Revisión de pares, Análisis y reporte de defectos, pruebas unitarias, pruebas de rendimiento, Gestión de la configuración, Gestión del soporte, entrenamiento, Gestión del ambiente de pruebas, Automatización de pruebas, Integración continua, Gestión de la experiencia de usuario, Prevención de defectos y análisis estático.

Cada KPA contiene sus respectivas guías, beneficios, procesos, mejores prácticas, plantillas, personalización y niveles de madurez. La estructura del modelo y su proceso de implementación se representan en la Figura 3-3 y Figura 3-4 .

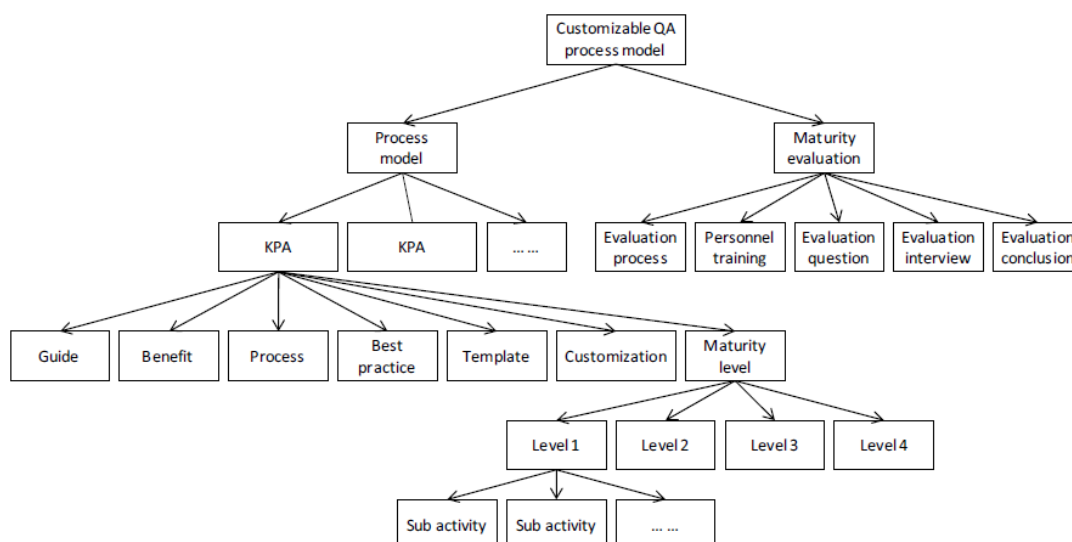


Figura 3-3 Arquitectura AQAM
Fuente: (Hongying & Cheng, 2011)

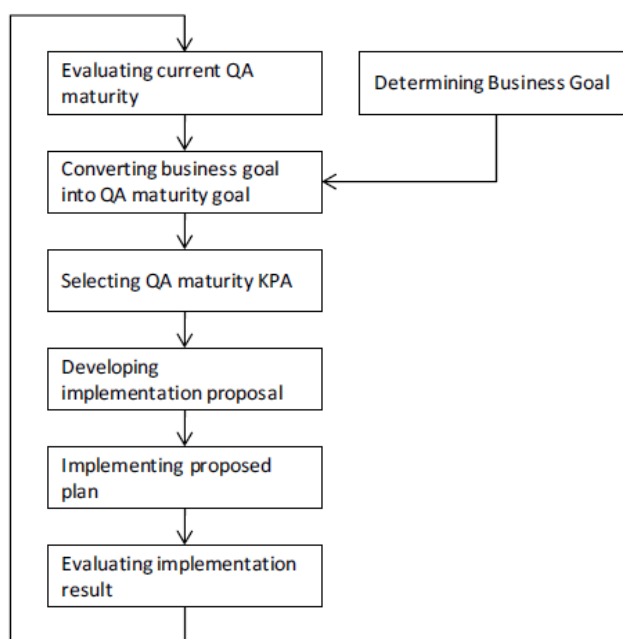


Figura 3-4 Proceso AQAM
Fuente: (Hongying & Cheng, 2011)

Principales resultados

De acuerdo con los autores, la personalización y evolución de los procesos de software y sus modelos están adquiriendo una importancia creciente en la comunidad del proceso de software. En este trabajo, se presenta AQAM como un nuevo modelo de proceso de aseguramiento de la calidad del software. Está diseñado principalmente para los equipos de desarrollo de software de tamaño pequeño y mediano. Con AQAM, las organizaciones pueden personalizar y administrar su modelo de proceso en base a su propia madurez, el proceso de calidad actual y el estilo de gestión.

Es uno de los pocos modelos de calidad aplicable a los modelos ágiles, sin embargo, es un modelo que incluye muchos artefactos que no están acorde con la premura que requiere un modelo ágil.

Modelo: Quality Attribute Driven Agile Development

Autor(es): Jeon, S., Han, M., Lee, E., & Lee, K.

Motivación de los autores

De acuerdo con una encuesta reciente, Scrum es uno de los métodos más populares que se están utilizando actualmente. A pesar de las conocidas ventajas de Scrum, se presentan algunas limitaciones como:

- a. Los product backlog y sprint backlog propios de Scrum consideran la funcionalidad del software, y no se evidencia con claridad el cumplimiento de sus características de calidad.
- b. Los backlogs de Scrum se centran en la implementación de los backlogs funcionales sin un análisis de trazabilidad de las relaciones entre los backlogs.
- c. Los métodos ágiles actuales están basados en prácticas que los equipos o los proyectos necesitan, pero, eso no es posible aplicar en organizaciones con múltiples equipos que trabajan cooperativamente.

De acuerdo al análisis realizado por los autores, se han realizado varios estudios para mejorar los problemas que se derivan del proceso ágil. La mayor parte de estos trabajos

priorizan el análisis del diseño de software; por lo tanto, si no hay una mejora en el enfoque de análisis del software, los proyectos de software aún siguen fallando debido a que sus atributos de calidad son deficientes. Por lo tanto, el autor propone un método de desarrollo ágil controlado por atributos de calidad llamado ACRUM.

Descripción del modelo

La contribución de los autores se resume en lo siguiente:

- Propuesta de AQUA (Analysis of Quality Attributes) método que analiza los atributos de calidad en el proceso Scrum, manteniendo la agilidad del software
- Exploración de una técnica para mantener la trazabilidad entre un backlog funcional y los atributos de calidad en Scrum, así como para mejorar la mantenibilidad vía RAM (Relation Association Matrix)
- Uso de VAQ (Validation of Quality Attribute)
- Prueba de los efectos de la solución propuesta en la aplicación en un proyecto Comercial en Samsung Electronics.

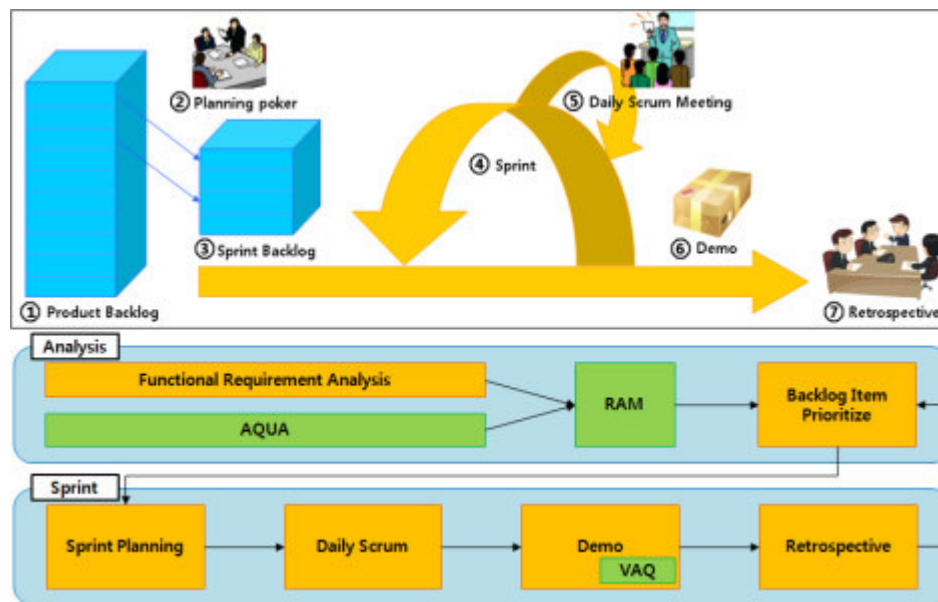


Figura 3-5 Proceso de desarrollo ACRUM

Fuente: (Jeon et al., 2011)

La Figura 3-5 presenta el proceso de desarrollo de ACRUM, que se basa en el modelo Scrum al que se le ha añadido 3 pasos para gestionar la calidad y gestionar los atributos de calidad. La calidad es controlada en el paso 1, basada en los atributos de calidad de los requerimientos funcionales. Otro aspecto importante es la colaboración del cliente, que es uno de los principios de los métodos ágiles, esto se puede fortalecer mediante un eficiente método de comunicación con los clientes a través de la práctica AQUA, como se indica en la Figura 3-6). Prácticas de RAM y VAQ también podrían mejorar aún más el valor de los métodos ágiles y acelerar su respuesta al cambio, estas prácticas se muestran en la Figura 3-7 y Figura 3-8.

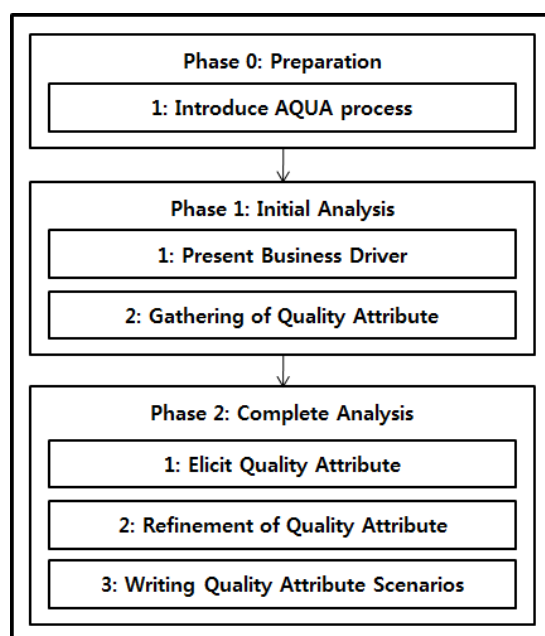


Figura 3-6 Proceso AQUA

Fuente: (Jeon et al., 2011)

Functional Backlog	Q.A Backlog	Data Storing or backup performance improvement	Extendability improvement of the multi format	Maintainability improvement	Security improvement of user information
Data Backup		○	X	○	X
Importing usage histories of my card		X	○	○	X
Supporting multi user		○	X	○	○
Export documentation		X	X	○	○
...		X	X	○	X
Achivement Ratio		50%	0%	100%	50%

Figura 3-7 Ejemplo de la Tabla RAM

Fuente: (Jeon et al., 2011)

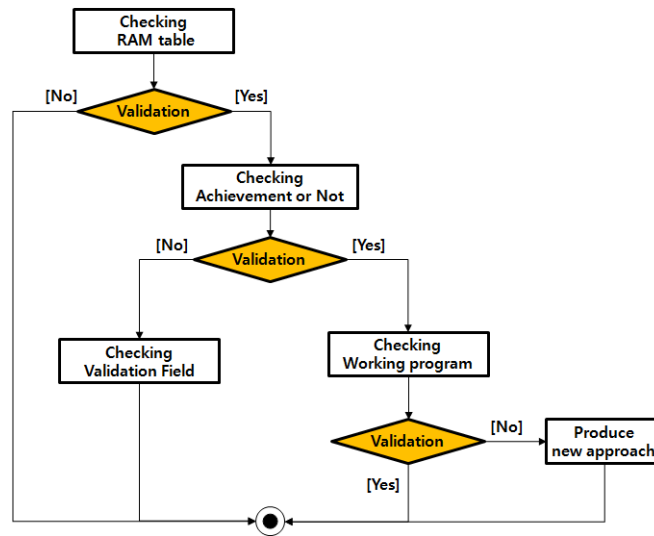


Figura 3-8 Proceso VAQ
Fuente: (Jeon et al., 2011)

Según los autores, los modelos ágiles han sido reconocidos como métodos eficientes para el desarrollo de software en cortos períodos de tiempo; ellos enfatizan en la satisfacción al cliente, rápida respuesta a cambios y versionamiento en menor tiempo. Las fallas de un proyecto son causadas por la falta de definición de las características de calidad ya que se enfatiza más en los requerimientos funcionales.

Principales resultados

Con la aplicación de ACRUM se obtiene que: la duración del proyecto se reduce en un 22% porque el tiempo de corregir los defectos se reduce debido al análisis y verificación de la calidad en forma periódica y la densidad y el tiempo de reducir los defectos fueron reducidos; la densidad de los defectos fue reducida en un 64% y el tiempo de corregir un defecto fue reducido en un 26%.

Los autores analizan varias técnicas utilizadas en estudios previos, existe una amplia descripción del estado del arte. ACRUM, es un modelo que se aplica al desarrollo ágil para asegurar la calidad en proyectos desarrollados con la metodología Scrum

Se puede tomar en cuenta más parámetros de calidad cuantitativos; el estudio solo considera 3. Los mismos autores lo dicen: “La limitación de este trabajo es que solo se orienta a un tipo de arquitectura, se pueden analizar varios enfoques de arquitectura de software durante el diseño; se necesita más investigación para mejorar la calidad del software, manteniendo un equilibrio entre la agilidad y el enfoque de arquitectura”.

Modelo: Agile Software: Ensuring Quality Assurance and Process.

Autor(es): Narinder Pal Singh y Rachna Soni

Motivación de los autores

El software es una entidad muy grande que contiene un gran número de procesos que se llevan a cabo antes de entregar un producto al cliente. Una de las preguntas muy difíciles de responder para la industria todavía es: ¿Cuáles serán los criterios de aceptación?, ¿Cuál será el proceso a seguir que pueda combinar el cambio de los requisitos de un proyecto con el tiempo que se dispone para culminarlo?. Cada proyecto tiene distintos requerimientos y necesidades. Para diseñar una solución de calidad, se han realizado una serie de propuestas, pero ninguna de ellas ha abordado el tema de la gestión del cambio. Todas las soluciones consideran a los requerimientos del software como una entidad estática, pero en la actualidad, es la entidad más volátil de todo el ciclo de vida del desarrollo.

Los cambios no administrados comienzan obstaculizando todo el ciclo de vida del proyecto y cambios de planificación, costos, la exigencia de respetar estándares de aseguramiento de calidad se convierten en el mayor mito de la documentación del proyecto. Este trabajo describe de cómo el desarrollo ágil puede ser útil en la garantía de calidad

Descripción del modelo

Los autores proponen un nuevo modelo para el desarrollo ágil de software, que incluye comentarios de los clientes y de la documentación del proyecto como elemento importante que

hace que el desarrollo ágil más auditable, responsable y centrada en el proceso. Este modelo toma en cuenta la retroalimentación del cliente, desarrollo iterativo de software y la documentación como sus componentes. Además, responde a las interrogantes planteadas por la administración, aseguramiento de la calidad, garantizando la correcta dirección del proyecto. El concepto ágil combinado con la Metodología en Cascada lleva a resultados mucho más estables, por lo que los autores proponen el modelo de desarrollo que se presenta en la Figura 3-9 de conformidad con el software ágil.

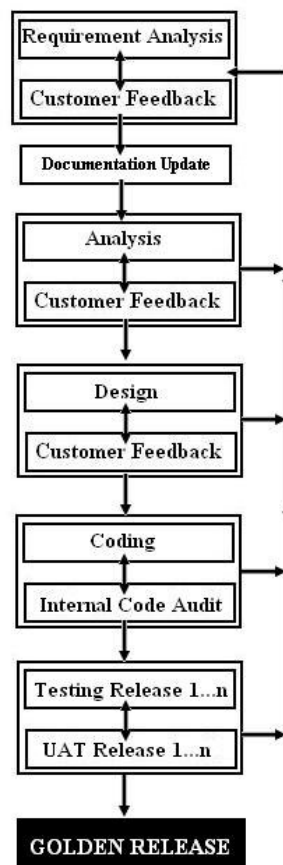


Figura 3-9 Modelo de Sinch
Fuente: (N. P. Singh & Soni, 2011)

Para cerrar la brecha de los requerimientos, gestión del cambio, especificaciones actuales y las especificaciones implementadas y para mejorar la calidad del software, el "Cliente" debe formar parte del ciclo de desarrollo. Para el desarrollo de cada versión del software, se debe rediseñar su proceso por lo que, el final del proyecto será siempre un software de alta calidad ya que ha sido sujeto a las pruebas de aceptación del usuario través de la UAT (Prueba de

aceptación del usuario).

Este modelo está orientado a garantizar la satisfacción del cliente y la calidad del software. Para los autores, la calidad es definida como el desarrollo de una solución de acuerdo a las especificaciones del cliente y que se adapten a los cambios que el cliente proponga.

No hay ninguna fase del modelo para el que no se realiza la retroalimentación de los clientes o una auditoría. El modelo está basado en el concepto "Desarrollo de versiones". En la que cada versión ha sido aprobada por el cliente, por lo tanto se garantiza un alto nivel de aceptación del software.

Principales resultados

De acuerdo con los autores, el proceso de desarrollo no es un proceso secuencial. La documentación debe reflejar la funcionalidad del software y debe estar visible para todo el equipo de desarrollo a través de un repositorio. La necesidad de cambio a ser incorporados en cada fase del Modelo en Cascada se debe convertirse en una versión de cada fase, además, es necesario una retroalimentación de los clientes y los cambios deben ser analizados desde el punto de vista del impacto en el resto de módulos del proyecto.

Es un modelo de calidad que integra un modelo tradicional con los principios del desarrollo ágil; sin embargo, no es un modelo que está aplicado y no describe las métricas que permitan probar el modelo

Modelo: Improving the Quality of Applying eXtreme Programming (XP) Approach

Autor(es): Nagy Ramadan Darwish

Motivación de los autores

A pesar de la gran cantidad de ventajas y características de XP, su uso no garantiza el éxito del proceso de desarrollo con un nivel de calidad aceptable. Además, los proyectos de software

se enfrentan a muchos desafíos que pueden conducir a fallas, por lo que es necesario asegurar la calidad del desarrollo de software.

Descripción del modelo

El investigador propone un enfoque de garantía de calidad para la aplicación del enfoque XP. El enfoque de aseguramiento de calidad propuesto puede ser utilizado para garantizar la calidad de la consecución de las fases XP. Se centra en elaborar un conjunto de pasos para el logro de cada fase XP, evaluando la calidad de cada fase y determinando la desviación de cada fase para mejorar la calidad del desarrollo de software.

Cada fase XP se puede lograr a través de una serie de pasos. El investigador elabora un conjunto de medidas para la consecución de cada fase XP: Exploración, Planificación, Iteraciones para la primera versión, Producción, Mantenimiento y Muerte. La Figura 3-10, ilustra el modelo de calidad propuesto. El enfoque propuesto de control de calidad incluye las siguientes actividades:

- a. Lograr las fases XP mediante los pasos elaborados.
- b. Evaluación de la calidad de la fase lograda.
- c. Identificación de la desviación entre la calidad real y el nivel de calidad aceptable.
- d. Análisis de la desviación para tomar acciones correctivas o de apoyo.

Principales resultados

Este estudio está orientado a la mejora de la calidad mediante la aplicación del enfoque XP. Por lo tanto, el investigador elabora un conjunto de actividades para la consecución de cada fase XP y propone un enfoque de aseguramiento de calidad para la aplicación de XP. Los desarrolladores y los clientes pueden utilizar los pasos elaborados como instrumento rector para lograr cada fase XP. Además, la propuesta puede ser utilizada para asegurar la calidad de la consecución de cada fase XP. Entonces, la desviación entre la calidad real y el nivel de calidad aceptable puede ser identificado y analizado. La aplicación del enfoque de XP para el desarrollo

de software no garantiza el éxito de este proceso. Por lo tanto, hay una necesidad de prácticas de garantía de calidad complementaria.

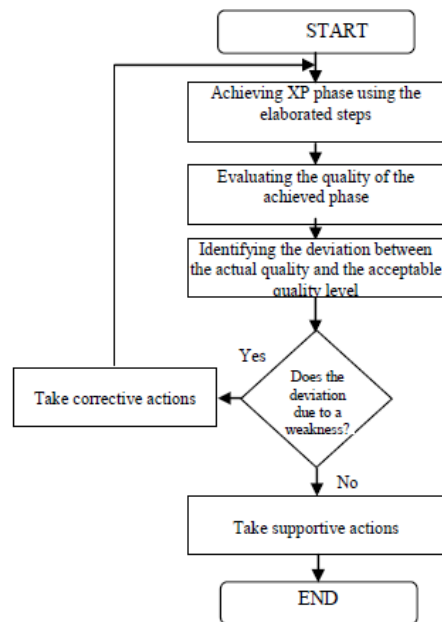


Figura 3-10 Enfoque de Aseguramiento de la Calidad de Darwish

Fuente: (Darwish, 2011)

El modelo propuesto es aplicable al desarrollo de software ágil, el autor presenta una amplia descripción del modelo, sin embargo, éste no ha probado en un caso práctico ni teórico. La propuesta no se centra en la evaluación de la calidad de cada fase XP.

Modelo: The 3C Approach for Agile Quality Assurance

Autor(es): André Janus, Reiner Dumke, Andreas Schmietendorf, Jens Jäger

Motivación de los autores

La integración continua es una práctica ágil para la integración continua de nuevo código fuente en el código base, incluyendo la compilación automatizada, construcción y test de pruebas. Desde el aseguramiento de la calidad tradicional, se han conocido métricas y métodos para medir la calidad del software; éstos son independientes de la metodología de desarrollo utilizada. Un gran reto para el mantenimiento y evolución de la aplicación es asegurar calidad

del software en base a un código fuente existente. La calidad interna del código fuente existente tiene el potencial para mejorar, pero existen algunas causas que impiden su mejora como:

- La falta de normas de calidad o de calidad "know-how",
- El código fuente se basa en una antigua versión del lenguaje de programación, por ejemplo: en el proyecto Java 1.3, que no se ajusta a los estándares de codificación actuales

Descripción del modelo

Los autores presentan un enfoque 3C, que es una extensión a la práctica ágil de Integración continua a la que se añade medición continua y mejora continua como actividades posteriores a CI (Integración Continua) y establece una base de métricas para un aseguramiento de la calidad ágil. De esta manera se integra el aseguramiento de la calidad tradicional con el proceso de desarrollo ágil y se establece métricas basadas en entradas de calidad para controlar la calidad interna del código fuente. Las 3C de este modelo incluyen: integración continua, medición continua y mejora continua.

1) Integración continua, integra un nuevo código a un código fuente base que ha sido desarrollado con anterioridad. La integración Continua se realiza con motores de integración continua que comprueban periódicamente el Sistema de Control de Versiones (VCS), a través de un chequeo de código dinámico. Si un nuevo código es encontrado, después que el desarrollador lo ha implementado en su entorno de desarrollo integrado (IDE), el Motor de Integración Continua compila el código fuente, y ejecuta las pruebas de manera automática. Si la compilación, la construcción o la ejecución de la prueba tiene cualquier error del motor de integración continua, se notifica al desarrollador (s), que han actualizado el código fuente. El modelo utiliza Q-Gates como se muestra en la Figura 3-11 para garantizar la no existencia de construcciones incorrectas como:

- Revisión de la programación por pares.
- Estándares de codificación y refactoring.

- Chequeo de código dinámico testing.

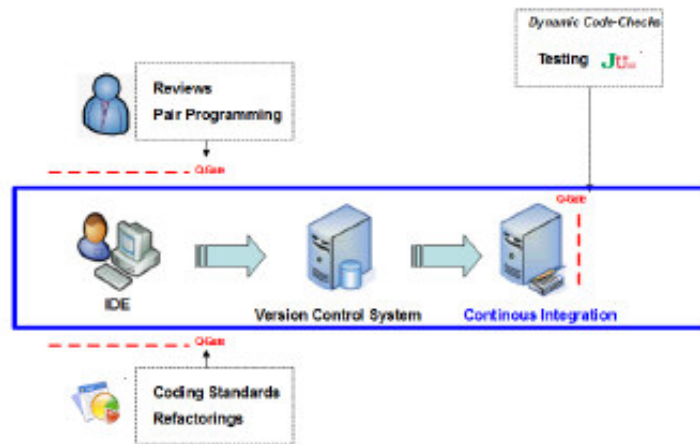


Figura 3-11 Integración Continua
Fuente: (Janus et al., 2012)

- 2) **Medición continua**, después de la ejecución y la aprobación de las pruebas, es una práctica común para conseguir algo más de información sobre el código fuente la adición de varias herramientas de medición de varias métricas tradicionales o violaciones al estilo de codificación.

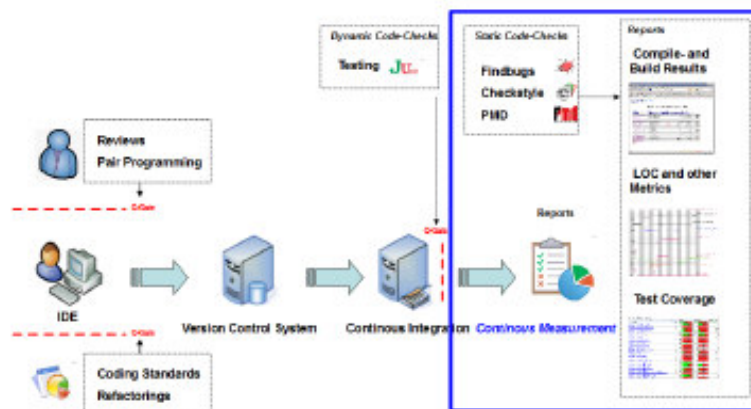


Figura 3-12 Medición Continua
Fuente: (Janus et al., 2012)

Es posible añadir ciertos umbrales para las métricas; de esta manera la integración continua ayuda a asegurar la calidad interna del software. La información directa del motor CI facilita la aceptación de la medida por los desarrolladores. Como se notifican

inmediatamente los desarrolladores pueden reaccionar de inmediato y hacer los cambios necesarios.

Se pueden utilizar algunas herramientas para obtener medidas que pueden integrarse a la Integración Continua, para determinar: LOC, número de paquetes, clases y métodos, número de test, cobertura de los test, proporción de crecimiento del test, etc.

- 3) **Mejora continua**, los resultados deben ser interpretados de forma manual con el fin de establecer acciones de mejora. En la Figura 3-13, se muestra que los resultados también evidencian que el gestor de calidad debe ser parte del equipo de desarrollo, de tal forma que los resultados sean interpretados correctamente. El gestor de calidad debe planificar refactorizaciones o los cambios que deban realizarse a los estándares de codificación o la guía de estilo de código. Para concretar las mejoras y determinar las métricas necesarias se utiliza el enfoque GQM (Goal – Question – Metric).

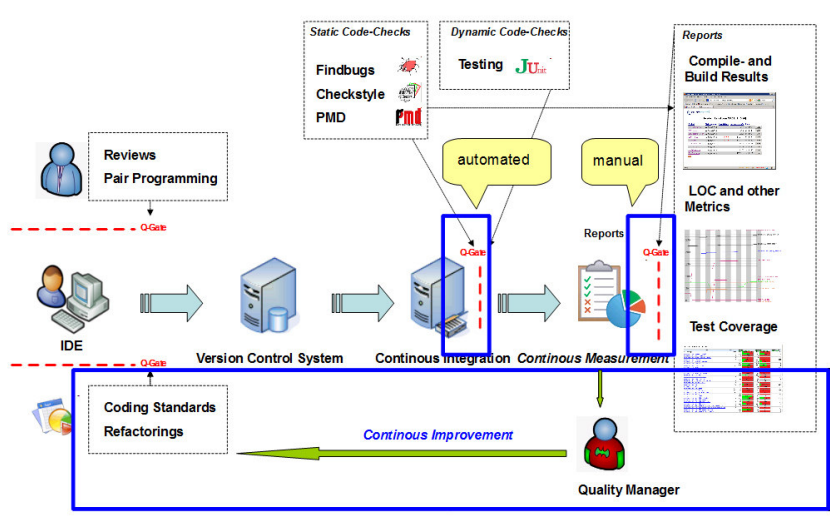


Figura 3-13 Mejora Continua
Fuente: (Janus et al., 2012)

Principales resultados

La propuesta fue desarrollada y probada en un proyecto de mantenimiento y evolución ágil de la industria automotriz de T-Systems International - una gran empresa de TIC alemán.

Dentro del proyecto se utilizó el enfoque de Aplicación Web basada en Java (legacy) incluyendo el uso de Herramientas de código abierto de Eco -System Java.

La práctica de Integración Continua en el desarrollo ágil de software ofrece nuevas oportunidades, haciendo que las medidas tradicionales sean obsoletas. Esta investigación provee una infraestructura para la Medición Continua para medir métricas tradicionales y ágiles y presenta métricas y atributos de calidad que pueden medirse a través del tiempo y conducen a actividades de mejora adecuadas y también "Continuas" seleccionados a través del enfoque GQM. El proceso de mejora no sólo da lugar a mejoras selectivas sino para puertas automáticas de calidad para preservar las mejoras. El uso del enfoque 3C podría mejorar significativamente la calidad interna del software y establecer puertas de calidad "in- process" para asegurar la calidad.

El enfoque propuesto se aplica a modelos ágiles de desarrollo de software, se establecen nuevas métricas relacionadas con la calidad del mantenimiento y evolución del software, el modelo se aplica en un caso real. Las métricas establecidas están orientadas a las métricas tradicionales y orientadas a modelos ágiles, además, se incluyen nuevos enfoques de calidad CK-Metrics y PMD

Modelo: Empirical study of Software Quality Evaluation in Agile Methodology Using Traditional Metrics

Autor(es): Kumi Jinzenji, Laurie Williams, Takashi Hoshino, Kenji Takahashi

Motivación de los autores

El uso de metodologías de desarrollo altamente iterativas tales como Agile y Lean, están creciendo, sin embargo, estas metodologías no proveen explícitamente prácticas para gestionar y medir la calidad y confiabilidad del software, como lo describe el estándar ISO/IEC 9126. El desarrollo de software altamente iterativo genera pequeñas piezas de software conocidas como "historias de usuario", que reducen los problemas de calidad del software. Sin embargo, a menudo en el desarrollo iterativo ágil, se realizan las pruebas, se descubren defectos y éstos son

arreglados al final de la iteración en vez de ser evaluado al final del ciclo de vida de cerca de la liberación del producto.

Como resultado de esto, el comparar la calidad de software prevista con los productos producidos en el pasado es difícil. En NTT Corporation, la inhabilidad de comparar la calidad de productos desarrollados en el pasado está siendo un obstáculo para la transformación de una metodología tradicional a una metodología altamente iterativa.

Descripción del modelo

Como se presenta en la Figura 3-14, se propone un esquema de evaluación de la calidad para desarrollo de software iterativo. El tipo de proceso de desarrollo determina el significado de conseguir artefactos de calidad válidos y útiles para calcular las métricas de este proceso iterativo. Hay que considerar los test de regresión porque su propósito es diferente entre el desarrollo secuencial y el desarrollo altamente iterativo.

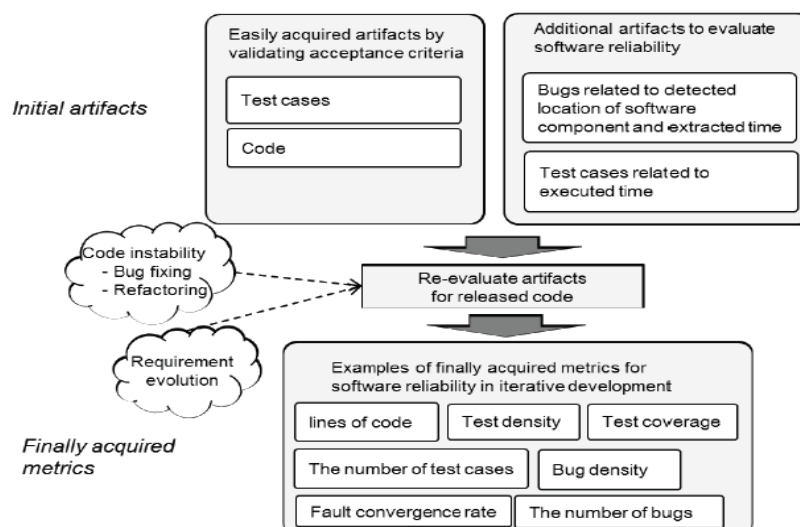


Figura 3-14 Esquema adquirido de métricas en el proceso iterativo

Fuente: (Jinzenji & Hoshino, 2013)

Los artefactos iniciales para evaluar la calidad son: casos de prueba y código. Para el caso del desarrollo iterativo se debe también recopilar los reportes de defectos de software para extraer su ubicación y tiempo y los casos de prueba relacionados en tiempo de ejecución. Los

artefactos son adquiridos de todos los valores del proceso de producción y luego son sumados. Los artefactos de los procesos de especificación y diseño pueden ser necesarios para evaluar la calidad, en estos casos, también deben ser acumulados y sumados (Ver Figura 3-15).

Las *pruebas de regresión*: son aplicadas a la corrección de errores, evolución de los requerimientos y refactorización. Hay una diferencia entre el desarrollo secuencial y el desarrollo altamente iterativo.

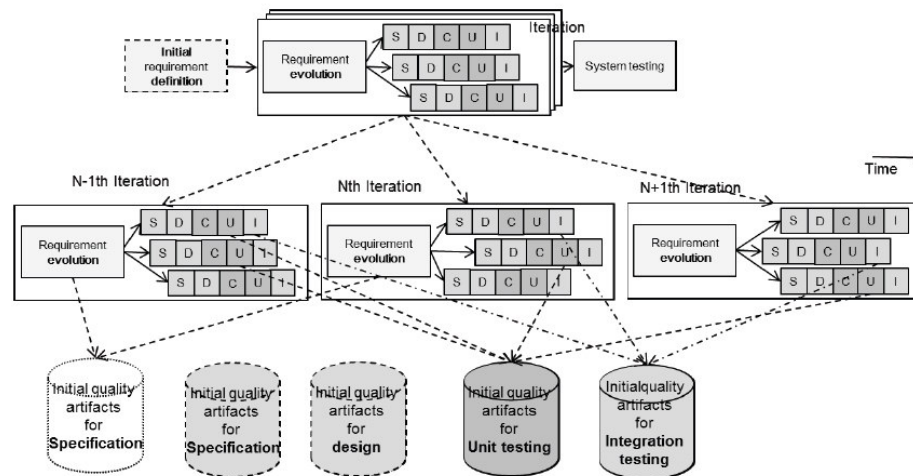


Figura 3-15 Sumarización de métricas
Fuente: (Jinzenji & Hoshino, 2013)

La re-evaluación de los artefactos de calidad: permite que los artefactos de calidad inicialmente adquiridos, los defectos de los reportes y los casos de pruebas tienen que ser re-evaluados porque la inestabilidad del software es propia del desarrollo altamente iterativo. En la Figura 3-16 se muestra un algoritmo para re-evaluación de los artefactos de calidad. Una vez que los artefactos son re-evaluados, se pueden calcular las métricas de calidad usando estos artefactos de calidad.

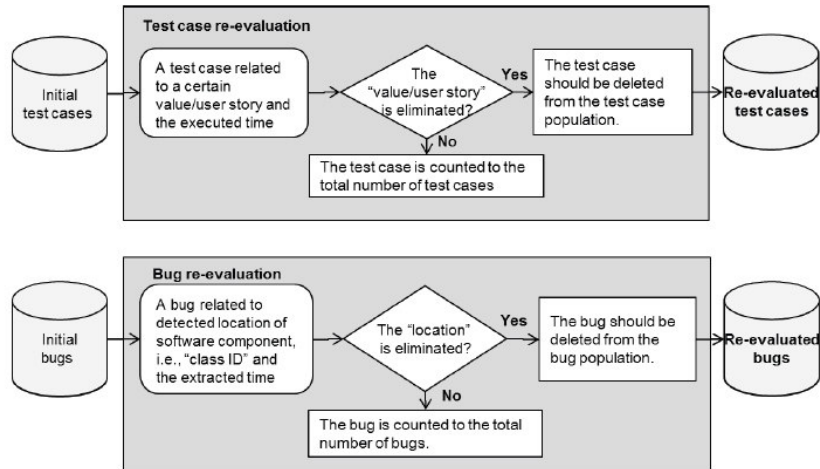


Figura 3-16 Estabilidad de código basada en artefactos de re-evaluación

Fuente: (Jinzenji & Hoshino, 2013)

La madurez para el desarrollo iterativo: después de re-evaluar los artefactos de calidad, se pueden adquirir las métricas de calidad y observar la madurez del software. En la Figura 3-17, se muestra las métricas de calidad típicas para pruebas de procesos se presentan las métricas típicas para pruebas en desarrollo de alta iteración, que son similares al desarrollo secuencial.

No .	Testing process	lines of code	Test coverage (C0,C1)	The number of testing cases	Test density	The number of bugs	Bug density	Bug converge nce rate
1	Unit testing	X	X	X	X	X	---	---
2	Integration testing	X	---	X	X	X	X	X
3	System testing	X	---	X	X	X	X	X

Figura 3-17 Métricas de calidad típicas para pruebas de procesos

Fuente: (Jinzenji & Hoshino, 2013)

La confiabilidad y especialmente la madurez del software también puede ser obtenida a través de tendencias temporales de la re-evaluación de artefactos.

Principales resultados

Los resultados de los experimentos realizados en esta investigación, muestran que el

desarrollo iterativo de software puede ser evaluado con las mismas métricas de calidad usadas en el desarrollo de software secuencial; eso significa que estas métricas son la base para transformar un desarrollo tradicional a un desarrollo ágil.

Existen consideraciones que se deben tomar en cuenta:

- El número de iteraciones y la cantidad de pruebas de regresión. una cierta cantidad de pruebas de regresión son esenciales, pero todas las pruebas de regresión no se deben considerar siempre para adaptarse a la curva de fiabilidad, incluso si el proyecto es corto
- Características del software ejemplo. La arquitectura del software ejemplo es muy estable durante la iteración. Esto no siempre se da en un desarrollo de altamente iterativo; por lo que se deberían examinar varios tipos de proyectos
- Trabajo extra para obtener y re-evaluar los artefactos. Los programadores deben registrar todos los artefactos especialmente errores y casos de pruebas en función del tiempo

Los resultados experimentales indican que las métricas de calidad tradicionales son aplicables al desarrollo iterativo, además, el desarrollo iterativo alcanzó la madurez suficiente para versiones comerciales. El modelo se ajusta a las metodologías ágiles.

Modelo: The 3C Approach for Agile Scrum Software Methodology

Autor(es): Jisha Johns, Akhil P Sivan, Prof. K Balachandran, Prof. B R Prathap

Motivación de los autores

Las metodologías de desarrollo ágil son las metodologías mas ampliamente usadas por pequeñas y medianas proyectos de desarrollo de software porque son livianas y se enfocan en la satisfacción del cliente.

Las metodologías ágiles permiten realizar cambios en los requerimientos para desarrollar un software, lo cual ayuda a la mejora de la eficiencia, productividad, etc. Estas metodologías enfatizan en las métricas ágiles, las cuales ayudan a dar seguimiento del éxito y la satisfacción

del cliente. Las métricas de los métodos tradicionales también ayudan si se las combina de una manera correcta.

Descripción del modelo

También tiene las tres actividades antes mencionadas en el Enfoque 3C original: Integración Continua, Medición continua y Mejora continua,

La métricas ágiles consideradas son: Número de tests, LOC, Tests de cobertura, Proporción de crecimiento del Test, Número de construcciones rotas, Sprint burn-down, velocidad del Sprint, Release burn-up.

Las métricas tradicionales que pueden tomarse en cuenta son: Violaciones de la codificación estándar, Violaciones al chequeo del código estático, Revisión de efectividad, Eficiencia en la eliminación de defectos (DRE), Métricas de pruebas

Principales resultados

Las metodologías de desarrollo de software ágil son ampliamente utilizadas en muchas organizaciones de software, ya que se centra en la evaluación de los avances del proyecto a lo largo del ciclo de vida de desarrollo y también ayuda en la entrega del producto correcto al cliente.

Las métricas ágiles ayudan a determinar el progreso del proyecto y la satisfacción del cliente, mientras que métrica tradicional se centra más en la productividad y la calidad del software.

Las métricas que se utilizaron en los modelos tradicionales se podría utilizar junto con las ágiles métricas, lo que sería muy útil para la determinación de las medidas de mejora que podrían ser implementado en los próximos Sprints

Se aplica a modelos ágiles de desarrollo de software, en este caso Scrum. Se establecen nuevas métricas ágiles y tradicionales para medir la calidad del software. Prácticamente el modelo propuesto es similar al modelo 3C, descrito anteriormente.

Modelo: Customer centered agile methodology

Autor(es): Parita Jain, Arun Sharma, Laxmi Ahuja

Motivación de los autores

Desde los últimos años, muchos productos nunca llegan a el mercado, y para aquellos que lo hacen, el éxito no está garantizado. La razón detrás de su fracaso es el proceso de ingeniería de software que se utiliza para desarrollar estos productos.

Inicialmente, el proceso utilizado se enfoca en la entrega oportuna del producto; los procesos de ingeniería de software se enfocan más en la especificación de requisitos, refactorización, tiempo de entrega, costo de desarrollo y minimización de esfuerzos y muchos más factores de los que depende el desarrollo de software. En general, la atención se centra más en mejorar las prácticas de desarrollo de software mejorando la reutilización, la comprensión de los requisitos, el tiempo de entrega del software y la rentabilidad y muchas otras características que conducen a la calidad.

Siendo la calidad el aspecto más importante en el desarrollo de software, debe enfocarse más para entregar un producto que sea aceptable para los clientes.

Descripción del modelo

El modelo propuesto da importancia a la satisfacción del cliente, interactuando con ellos con más frecuencia en cada fase de un sprint. El ciclo de vida del desarrollo de software propuesto consta de cinco fases, como se puede observar en la Figura 3-18, se han utilizado en varios modelos de desarrollo. Cada fase individualmente juega un papel importante e impacta la

calidad del producto de software.

Fase 1. Análisis de requisitos: es una fase de elicitación de requisitos en la que se obtienen las necesidades y requisitos de los clientes, especificando los requisitos funcionales y no funcionales.

Fase 2. Diseño de software: en esta fase se crea la arquitectura completa del software a desarrollar que especifica cómo funcionará y se verá el producto de software para el cliente.

Fase 3. Implementación del software: esta fase crea una versión ejecutable del producto de software. El desarrollo del software se basa en el lenguaje de programación utilizado con fines de codificación.

Fase 4. Pruebas de software: después de que se desarrolla el producto ejecutable, comienza la fase de pruebas. La prueba de software es un proceso para descubrir errores, fallos y fallas en el software desarrollado y corregirlos antes de que se entregue al cliente. Básicamente verifica y valida todas las funcionalidades del software.

Fase 5. Mantenimiento de software: esto lleva a cambios en el producto de software después de que se implemente en el lado del cliente si ocurre algún error o falla. También proporciona que el producto de software sea adaptable en el entorno cambiante mejorando el rendimiento del sistema de software.

A más de las fases, los autores consideran que todas las fases afectan a la calidad de software y pueden ser analizadas con varios atributos de calidad aplicados al proceso ágil; por esta razón proponen atributos ágiles y atributos de calidad para cada fase de proceso de desarrollo.

Principales resultados

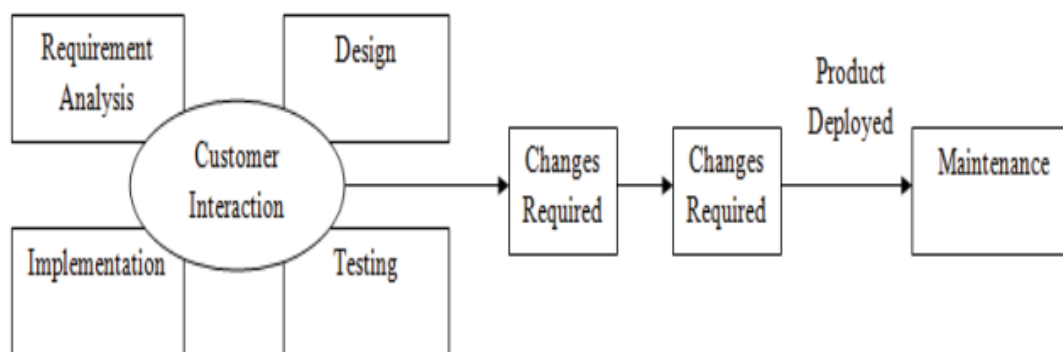


Figura 3-18 Metodología ágil centrada en el cliente

Fuente: (Jain et al., 2018)

Principales resultados

La calidad del software juega un papel importante al hacer un proyecto de software. Se deben satisfacer varios factores de calidad en algún nivel para tener un producto de software de calidad. El desarrollo de software consta de cinco fases que deben ponerse al día con los estándares de calidad durante el desarrollo. Al ser ágil, un enfoque flexible para el desarrollo de software, permite que se entregue un producto de calidad al cliente. El objetivo principal de este estudio es analizar el impacto del Software Agile Proceso de Desarrollo (ASDP) sobre la calidad del software. A través de diversos estudios se ha demostrado que todo atributo de calidad es importante en la perspectiva de tener un producto de calidad.

Para incorporar cada factor de calidad en el desarrollo de software, los atributos de calidad deben ser mapeados con los atributos ágiles del proceso de desarrollo de software y viceversa.

Modelo: A conceptual framework of on-time delivery

Autor(es): Elvan Kula, Eric Greuter, Arie van Deursen

Motivación de los autores

La entrega tardía y los sobrecostos han sido problemas comunes en la industria del software durante décadas. En promedio, los proyectos de software se ejecutan alrededor de un

30 por ciento de tiempo extra. Este porcentaje no parece haber disminuido desde la década de 1980. Aunque la estimación del esfuerzo la realizan casi todas las industrias, es especialmente desafiante en la industria del software. Esto se debe principalmente al hecho de que el desarrollo de software es una tarea compleja, afectada por una variedad de actores sociales y técnicos.

El éxito general percibido de un proyecto de software depende en gran medida del cumplimiento de las estimaciones de tiempo y costo. Por lo tanto, mejorar la estimación del esfuerzo es un objetivo fundamental para las organizaciones de software: puede ayudar a las empresas a reducir los retrasos y mejorar la satisfacción del cliente, al tiempo que les permite asignar recursos de manera eficiente, reducir costos y optimizar la entrega.

A pesar de la disponibilidad de muchos métodos y directrices de estimación la entrega a tiempo en el desarrollo de software ágil sigue siendo un desafío importante. Existen investigaciones que han identificado una gran cantidad de factores que pueden influir en el esfuerzo de desarrollo de software, pero no está claro qué factores tienen el mayor impacto. Se carece de una comprensión de las relaciones entre estos factores y cómo afectan la entrega a tiempo.

Descripción del modelo

El modelo conceptual propuesto por los autores que se presenta en la Figura 3-19, tiene cinco dimensiones/categorías, representadas en rectángulos. Como se puede apreciar en los 25 factores que influyen en la entrega ágil, son visualizados en elipses, conectadas por flechas bidireccionales que representan sus relaciones percibidas, las mismas que pueden ser positivas o negativas. Se encuentra que los factores tienen tres tipos de relaciones directas con la puntualidad de las entregas; pueden dar lugar a reelaboración necesaria (NR), tiempo de espera no planificado (WT) o cambios en la eficacia del equipo (TE). Los símbolos de estrella indican factores que están clasificados entre los 10 factores más relevantes en datos de encuestas o datos de repositorio.

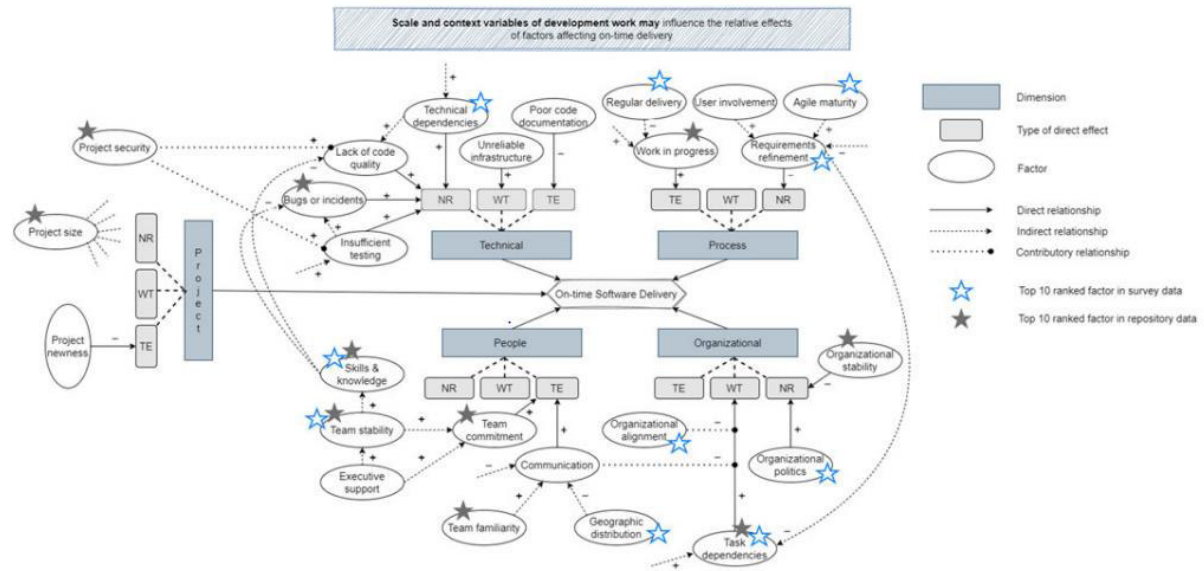


Figura 3-19 Modelo Conceptual de entrega a tiempo en proyectos ágiles

Fuente: (Kula et al., 2022)

Principales resultados

Los hallazgos clave de este estudio son:

- 1) Se percibe que el refinamiento de los requisitos, las dependencias de tareas, la alineación organizacional, la política organizacional y la distribución geográfica de los equipos tienen el mayor impacto en la entrega oportuna del software.
- 2) El tamaño del proyecto, el número de dependencias, el desempeño histórico de la entrega, la familiaridad con el equipo y la experiencia del desarrollador son las variables más importantes que explican las desviaciones del cronograma en las entregas de software.
- 3) Se encuentra que los factores interactúan jerárquicamente: se percibe que los factores organizacionales interactúan con los factores de las personas, que a su vez impactan los factores técnicos.

3.7.5 RQ5: ¿Cuáles son los principios ágiles usados por los modelos de calidad encontrados?

Para contestar esta pregunta, se tomará como base los doce principios declarados en el Manifiesto Ágil (Agile Alliance, 2001), éstos han sido confrontados con cada uno de los modelos encontrados.

En la Tabla 3-11 se detallan los principios ágiles que cumple cada uno de los modelos. Estos resultados indican que el Principio Ágil 2 “*Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente*”, es el más considerado por los modelos, seguido por el Principio 9; “*La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad*”, decir, los principios relacionados con la agilidad de responder al cambio y la simplicidad del diseño. Además, se puede ver que ningún modelo cumple con todos los principios ágiles y que el modelo AM-Quick (MD11) es el que cubre más principios.

Tabla 3-11 Principios ágiles que cumplen los modelos de calidad estudiados.

No	Principios ágiles	Models																Frec.
		MD1	MD2	MD3	MD4	MD5	MD6	MD7	MD8	MD9	MD10	MD11	MD12	MD13	MD14	MD15	MD16	
1	"Satisfacer al cliente mediante la entrega temprana y continua de software con valor"			√				√		√		√					√	5
2	"Aceptar que los requisitos cambien, incluso en etapas tardías del desarrollo"	√	√	√	√	√	√	√		√	√	√				√		11
3	"Entregar software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible"							√		√				√				3
4	"Los responsables de negocio y los desarrolladores trabajan juntos de forma cotidiana durante todo el proyecto"		√			√	√	√		√		√						6
5	"Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo"			√				√		√		√						4
6	"El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara"						√			√		√		√				4
7	"El software funcionando es la medida principal de progreso"				√	√	√	√					√		√			6
8	"Promover el desarrollo sostenible. Los promotores, desarrolladores y usuarios deben ser capaces de mantener un ritmo constante de forma indefinida"	√						√		√		√					√	5
9	"La atención continua a la excelencia técnica y al buen diseño mejora la agilidad"	√	√					√	√	√		√	√		√	√		9
10	"La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial"	√		√					√			√	√					5
11	"Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados"								√	√		√						3
12	"A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento"					√		√				√	√					4
TOTAL		4	3	4	2	4	4	9	3	9	1	10	4	2	2	2	2	65

3.7.6 RQ6: ¿Cuáles son las características de calidad que están contemplados en los modelos analizados?

Para este análisis se han considerando las características del estándar ISO 25010, que se presentan en la Figura 3-20, que corresponde a los atributos de calidad del software internas como externas del software. También se han tomado en cuenta los atributos derivados de investigaciones que enfocan exclusivamente este aspecto; la Tabla 3-12 presenta estos atributos, que se les ha denominado *características* de acuerdo al estándar.

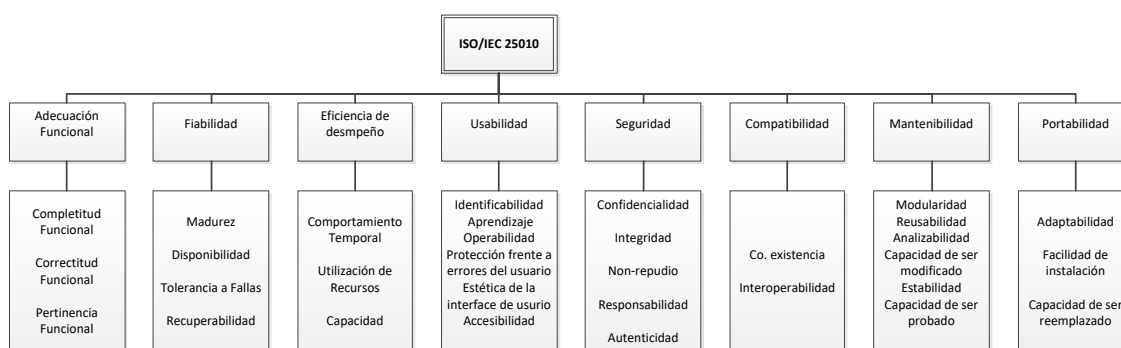


Figura 3-20 Características y subcaracterísticas del estándar ISO/IEC 25010.

Tabla 3-12 Número de atributos usados por los modelos de calidad y estudios primarios que aplican desarrollo de software ágil.

Características	Subcaracterísticas	Frecuencia	Referencias
Adecuación Funcional	3	2	[S20],[S51],[S13]
Fiabilidad	5	4	[S17],[S24],[S67],[S68],[S70]
Compatibilidad	2	2	[S67]
Usabilidad	10	8	[S20],[S67],[S69]
Rendimiento/Eficiencia	3	3	[S66], [S67]
Seguridad	5	2	[S66],[S68],[S13]
Mantenibilidad	8	13	[S20],[S66],[S67], S68],[S69],
Portabilidad	4	4	[S67],[S13]
Otros	31	55	[S11],[S20],[S24],[S66],[S69],[S13], [S70],[S82],[S106],[S107]
Total	71	93	

Los atributos de *Mantenibilidad* son los que más frecuentemente han sido estudiados por los autores con un 13,98%. En varios trabajos se han estudiado características de calidad que no pertenecen a la ISO/IEC 25010 tales como: “*Expandibilidad, Línea de tiempo, Integridad, Simplicidad, Generalidad, Escalabilidad, Robustez, , Reusabilidad, Modularidad en tiempo de*

ejecución, Flexibilidad, Correctitud, V&V” que corresponde al 41,94%. Además, existen estudios que analizan las características en una forma general, sin enfocarse en sus subcaracterísticas (17,20%). La lista de características estudiadas se encuentra en el Anexo G.

3.7.7 RQ7: ¿Cuáles son los aspectos de calidad que cumplen los modelos encontrados?

Para contestar esta pregunta se ha realizado una valoración de los aspectos de calidad mencionados anteriormente y adicionalmente se han incluido características relevantes como la *integración del modelo al proceso de desarrollo, validación del modelo propuesto* y cuál es el *modelo ágil al que se aplica*. La valoración se realizó a los modelos descritos en la Tabla 3-10 de acuerdo a parámetros que se detallan en la Tabla 3-13. La información obtenida está basada en el análisis de la información publicada por los autores y se la ha valorado como sigue: en rojo (no cumple), en amarillo (cumple parcialmente) y en verde (cumple completamente).

Tabla 3-13 Aspectos considerados para el análisis de los modelos.

ASPECTOS	DESCRIPCIÓN
Métricas	El modelo proporciona métricas y la forma como medir la calidad de proceso de desarrollo.
Atributos	El Modelo considera atributos de calidad.
Factores	El modelo incluye factores técnicos y/o humanos que afectan a la calidad.
Principios	El modelo cumple con los principios del manifiesto ágil.
Integración	El modelo se integra a todas las fases de desarrollo de software.
Validación	Aplicación experimental o empírica del modelo
Aplicado a:	Modelo de desarrollo ágil aplicado

En la Tabla 3-14 se muestra el estado de cumplimiento de los aspectos de calidad y características consideradas anteriormente para los modelos estudiados.

Tabla 3-14 Cumplimiento de los aspectos de calidad de los modelos estudiados.

Id.	Métricas/ atributos	Principios	Factores	Integrado al proceso	Validación	Aplicado a
MD1	X	P	X	√	√	XP
MD2	X	P	X	√	X	XP
MD3	√	P	√	X	X	NE
MD4	P	P	X	P	√	NE
MD5	√	P	X	√	X	NE
MD6	√	P	X	√	√	Scrum
MD7	X	P	X	√	X	NE

Id.	Métricas/ atributos	Principios	Factores	Integrado al proceso	Validación	Aplicado a
MD8	√	P	X	P	√	NE
MD9	P	P	X	X	X	XP, Scrum
MD10	√	P	X	√	√	NE
MD11	√	P	X	√	√	NE
MD12	√	P	X	P	X	Scrum
MD13	P	P	X	P	X	NE
MD14	√	P	X	P	X	NE
MD 15	√	P	X	√	X	NE
MD16	√	P	√	X	√	NE

NE. No especificado

Además, se puede evidenciar que la validación de los modelos estudiados a través de su aplicación experimental o empírica es del 46,67%. Un resultado similar del 46,67% de los modelos no se integran al proceso de desarrollo de software, debido a que son propuestas aisladas del proceso o se integran en forma parcial, es decir, cubre solo ciertas fases del proceso. El 66,67% de los modelos incluyen métricas y/o atributos para medir la calidad, los modelos que sí los contemplan aplican métricas tradicionales, métricas ágiles o atributos basados en estándares tradicionales. Todos los modelos de calidad analizados consideran parcialmente los principios del Manifiesto Ágil, es decir que ninguno de ellos cumple con los doce principios. Los modelos de calidad MD3 y MD5 son los únicos que consideran factores que influyen en el éxito de un proyecto, generalmente, este aspecto no ha sido considerado en las propuestas realizadas por los autores.

Tabla 3-15 Resumen del cumplimiento de los aspectos de calidad en los modelos estudiados.

Id.	Métricas/ atributos	Principios	Factores	Integrado al proceso	Validación	Promedio
X	18,75%	0,00%	87,50%	18,75%	56,25%	36,25%
P	18,75%	100,00%	0,00%	31,25%	0,00%	30,00%
√	62,50%	0,00%	12,50%	50,00%	43,75%	33,75%

El 31,25% de los modelos están diseñados para las metodologías SCRUM y XP; una de las razones que justifican estos resultados es que Scrum y sus distintas combinaciones son las metodologías de mayor uso en el mercado en la actualidad. Un alto porcentaje de modelos analizados correspondiente al 68,75% no indica la metodología de desarrollo con la que se valida el modelo; esto podría considerarse una ventaja ya que se puede generalizar su aplicación a cualquier modelo, siempre si cuando las propuestas se basen en principios y valores ágiles.

En la Tabla 3-15 muestran los principales resultados del cumplimiento de los aspectos de calidad considerados en la comparación de los modelos.

Por otro lado, se ha realizado una recopilación de las características de calidad que han sido consideradas por cada uno de los modelos encontrados, en la Tabla 3-16 se han incluido únicamente a los modelos que se basan en otros modelos o estándares de calidad como referencia.

Tabla 3-16 Características de calidad basados en otros estándares incluidas en los modelos estudiados.

Id.	Estándar Referenciado	Características de calidad obtenidas de los modelos de referencia
MD3	CMMI-TMMI	Modelo de proceso: Identifica 20 Áreas clave del negocio: Gestión de requerimientos, Proceso de Auditoría, Planificación de pruebas, Gestión de casos de pruebas, Revisión por pares, Análisis de Defectos, Reporte de defectos, Pruebas unitarias, Pruebas de rendimiento, Gestión de la Configuración, Gestión del Soporte, Capacitación, Gestión de ambientes de pruebas, Organización de las pruebas, Automatización de pruebas, Integración continua, Gestión de las experiencias del usuario, Nivel de pruebas, Prevención de defectos y análisis estático. Niveles de evaluación de la madurez CMMI: Inicial, Realizado, Gestionado, Optimizado.
MD4	ISO/IEC, ISO 19761:2003 COSMIC	Determina la calidad de la documentación de historias de usuario mediante el método de evaluación COSMIC; para ello, utiliza una escala de calidad de A – E de acuerdo a lo establecido en la guía de COSMIC.
MD5	ISO 9126-1	Principales atributos de calidad: Funcionalidad, Fiabilidad, Mantenibilidad, Usabilidad, Eficiencia y Portabilidad.
MD6	CMMI	Áreas CMMI: Gestión de requerimientos, Gestión de Riesgos, Mantenimiento y Aseguramiento de la calidad,.
MD9	ISO 25000	La calidad interna del software y la calidad del producto se miden a través de métricas del producto los cuales son indicadores para determinar los atributos de calidad.
MD11	ISO/IEC TR 9126-2	Determina la madurez del software, de acuerdo al estándar ISO/IEC 9126
MD12	CMMI	Usa CMMI como un modelo de referencia para realizar el mapeo de las actividades del proceso de desarrollo de software: Gestión de Requerimientos, Aseguramiento de la calidad del proceso y del producto, Desarrollo de requerimientos, Solución Técnica, Integración del producto, Validación, and verificación.
MD13	CMMI MPS.BR	Se basa en los modelos de madurez: CMMI y MPS.BR que generalmente tienen guías relacionadas a QA. En CMMI, Las

Id.	Estándar Referenciado	Características de calidad obtenidas de los modelos de referencia
MD14	ISO 25010	áreas de procesos consideradas son: Aseguramiento de la calidad organizacional, Gestión de lecciones aprendidas; Capacitación, Gestión del Conocimiento, Aseguramiento de la calidad, Gestión de la Integración (ITM), análisis de riesgos y Análisis de costos. Se basa en cuatro características de calidad del estándar ISO/IEC 25010: considera cuatro características de calidad: Seguridad, Rendimiento, Mantenibilidad y Fiabilidad.

3.8 Principales hallazgos

El resultado de esta revisión sistemática es un catálogo de factores, prácticas ágiles y métricas utilizadas en el proceso de desarrollo de software, lo que permitirá que investigadores y desarrolladores determinen cuáles pueden ser de utilidad para ellos. La información proporcionada como resultado de este estudio es un buen punto de partida que puede contribuir a la definición de modelos de calidad que puedan ser aplicados al desarrollo ágil. Los desarrolladores pueden ahorrar tiempo y esfuerzo en determinar los aspectos de calidad que pueden aplicar durante el desarrollo y mantenimiento de software.

Varios autores han abordado aspectos de calidad de software, sin embargo, no aún no existen conceptos estandarizados a los diferentes aspectos de calidad estudiados, por ejemplo, [S4] considera “principios” a lo que otros autores considera como “prácticas”, en este trabajo se consideran “principios” a los establecidos en el Manifiesto Ágil. Así mismo, existen autores que consideran “atributos ágiles” [S40] a lo que en este estudio se ha llamado “factores críticos de éxito”, otros autores llaman “características ágiles” [S52] a lo que se ha considerado como “prácticas ágiles”. [S67] denomina factores a lo definido como características y subcaracterísticas de calidad. Esto evidencia que la terminología aplicada en la calidad de software es diferente y no se aplica de manera estandarizada en la ingeniería de software.

Para determinar los factores que influyen en el éxito de un proyecto de software, se han tomado en cuenta las siete dimensiones considerados en los estudios [S1] y [S2]. Al relacionar estas dimensiones con los valores ágiles se puede evidenciar que el valor *“Individuos y su interacción, por encima de los procesos y las herramientas”* está relacionado con la dimensión

personas, ya que esta dimensión incluye aspectos como la competencia del equipo de trabajo, motivación y auto-organización. El valor “*software que funciona, por encima de la documentación exhaustiva*” está relacionada con la dimensión *técnica*, y los factores considerados en ella, tales como: *Diseño simple, cantidad correcta de documentación, pruebas de integración correcta*, etc. que están directamente relacionadas con este valor, así como con las prácticas ágiles. El “*valor colaboración con el cliente, por encima de la negociación contractual*”, está en concordancia con la dimensión *Retroalimentación con el cliente*. En conclusión, los factores críticos de éxito son coherentes con la aplicación de las prácticas y valores ágiles.

Muchos estudios analizados han considerado el tópico de métricas con el fin de determinar el aseguramiento de la calidad en sus etapas de diseño y programación. Los resultados evidencian que las métricas más utilizadas son el *suit* de métricas CK, es debido a que están relacionadas con la detección de problemas de calidad en etapas tempranas del desarrollo de software. Algunas prácticas ágiles como: *testing*, *diseño simple*, *integración continua*, *estándares de codificación* y *refactoring*, están relacionados con las métricas de *Testing* que permiten cuantificar prioritariamente casos de prueba, fallas y defectos del software. Adicionalmente, las métricas de diseño y *testing* de un proyecto están apoyadas con varias herramientas de software incorporadas a los entornos de desarrollo ágil, las mismas que facilitan su medición y seguimiento; algunos modelos de calidad como [S60] se basan en ellas. El seguimiento y evaluación de las métricas de software en un entorno ágil contribuirá a prevenir y evitar que se produzcan defectos y fallas en lugar de encontrarlos y corregirlos una vez que éstos ya se han convertido en un problema de calidad, es decir, que es importante asegurar la calidad interna del software.

Las prácticas ágiles es el aspecto más estudiado en la calidad de proyectos de software ágiles. Un estudio importante es [S47], que propone un modelo híbrido para integrar las prácticas ágiles en las empresas que desarrollan software mediante la clasificación de las prácticas ágiles en cuatro grupos: *Prácticas de ingeniería*, *Prácticas de testing* y *aseguramiento de la calidad*, *Prácticas de gestión* y *Prácticas de productividad y mantenimiento*. En estudio, se ha considerado los tres primeros y se ha incrementado dos nuevos grupos: *Prácticas de*

Documentación y Prácticas de Trabajo en Equipo, debido a que en la literatura se han encontrado una gran cantidad de prácticas relacionadas con el personal que trabaja en el proyecto y se consideró adecuado incorporarlos en una nueva categoría. Debido a que uno de los valores ágiles es “*software que funciona, por encima de la documentación exhaustiva*”, se ha visto adecuado incluir la categoría *Prácticas de documentación*, con el fin de resaltar las prácticas de documentación que los autores proponen para cumplir con este valor. Por otra parte, [S49], realiza un mapeo que considera 18 prácticas ágiles que ha criterio de los autores son las más utilizadas. Todas las prácticas ágiles encontradas en los estudios analizados contribuyeron a la identificación de 165 prácticas ágiles.

Los modelos de calidad que se relacionan con el proceso de desarrollo ágil están basados en estándares ya establecidos, como CMMI, se adaptan a una metodología en particular como Scrum o XP, o evolucionan un Modelo Tradicional como Cascada a un Modelo ágil. Las etapas de desarrollo correspondientes al diseño y las pruebas de software son las que más han sido analizadas por los autores, es importante recalcar que muchas propuestas que ya han sido validadas, priorizan una etapa específica del proceso de desarrollo. Dado que los factores humanos afectan al éxito de un proyecto, es importante que un modelo de calidad incluya en forma integral este aspecto, así como las técnicas y prácticas relacionadas con el proceso ágil, donde se da mayor importancia a la participación de los interesados y al desarrollo incremental con versiones de software. Los doce principios establecidos en el Manifiesto Ágil fueron confrontados con los modelos de calidad encontrados en este estudio; si bien es cierto, al establecer un modelo de calidad orientado al desarrollo ágil, éstos deben estar acorde con la filosofía ágil a través de sus principios y valores; sin embargo, éstos no ocurre con la totalidad de los modelos; como se refleja en los resultados, esto posiblemente se debe a que los modelos en su mayoría están orientados a etapas específicas del desarrollo de software; solo cuatro modelos han cubierto principios relacionados con la *motivación, auto-organización* del equipo, *ritmo de trabajo constante de los involucrados*. Con base en los resultados obtenidos, no existe un modelo que abarque todos los aspectos de calidad considerados en el presente estudio, el aspecto más ha sido incluido en los modelos son: métricas/atributos de calidad con un 56,25% de cumplimiento. La mayoría de los modelos podrían complementarse relacionándolos con todos los principios de calidad del Manifiesto Ágil y con factores críticos de éxito ya que

solamente un modelo contempla factores críticos de éxito, pero no está validado.

Las características de calidad han sido tratados desde distintos enfoques en el entorno ágil, existen modelos que se basan en las características y subcaracterísticas de calidad interna, calidad externa y calidad de uso establecidas en el estándar ISO 25010. No existe un modelo que se relacione con el aseguramiento de la calidad del software durante todas las etapas de su desarrollo, tomando en cuenta además que los requerimientos no funcionales del software son atributos de calidad deseados por los usuarios, y muchos de ellos están relacionados al proceso de desarrollo, tal es el caso de: la eficiencia, rendimiento, simplicidad, reusabilidad, flexibilidad, integridad, etc.

3.9 Conclusiones del capítulo

Este estudio presenta una revisión sistemática de literatura de 1889 estudios primarios que tienen relación con la calidad aplicada al desarrollo de software ágil, de los cuales se seleccionaron 111 estudios relevantes. El análisis de la información recopilada ha sido conducido en base a siete preguntas de investigación planteadas, las mismas que están relacionados a los siguientes aspectos de calidad: factores, métricas, principios, modelos y atributos de calidad.

Las prácticas ágiles es la temática más estudiada en la calidad de software que usa un enfoque ágil; sin embargo, pocas propuestas relacionan estas prácticas con las características de calidad, por esta razón, sigue siendo un tema para desarrollar futuros estudios. Considerando los factores de éxito, la dimensión más estudiada es “*personas*”, este resultado es esperado ya que una de las características de las metodologías ágiles es la motivación e implicación del equipo de desarrollo. En cuanto a los atributos de calidad, la “*mantenibilidad*”, es la característica más analizada dentro del contexto ágil, este resultado también es esperado ya que los modelos ágiles están orientados a los continuos cambios del software, por lo que muchos autores proponen la reusabilidad como práctica para reducir el tiempo de ejecución de pruebas.

Los modelos de calidad, en su mayoría, se han inspirado en modelos anteriores, lo cual hace

que tengan una gran posibilidad de encontrar divergencias y similitudes entre sus características. La comparación de los 16 modelos de calidad refleja que ninguno de ellos cumple con todos los aspectos analizados en este trabajo. El principio del manifiesto ágil No.2 “*Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente*”, es el más aplicado en los modelos estudiados. La evidencia indica que el aspecto de factores de calidad no ha sido incorporado a los modelos de calidad y el único modelo que lo contempla no está validado y no se integra al proceso de desarrollo.

CAPÍTULO 4: MODELO CONCEPTUAL DE INFLUENCIA DE LOS FACTORES CRÍTICOS DE ÉXITO Y PRÁCTICAS ÁGILES SOBRE LA CALIDAD DEL SOFTWARE

En el presente capítulo se presenta la estructura de un modelo conceptual para determinar la influencia de los factores críticos de éxito y prácticas ágiles sobre la calidad de software, basados en ocho características de calidad tomadas del estándar ISO/IEC 25010. Además, se fundamenta en los seis factores críticos de éxito y 17 prácticas ágiles que forman parte del modelo, así como las hipótesis derivadas de él. El modelo considera tres tipos de influencias: a) influencia de los factores sobre las prácticas ágiles, b) influencia de las prácticas ágiles sobre las características de calidad, y c) influencia de los factores sobre las características de calidad.

4.1 Metodología para estructurar el modelo conceptual

La estructura del Modelo Conceptual estuvo guiado por el desarrollo de seis pasos, los mismos que se muestran en la Figura 4-1:

El paso 1, *Extracción de datos de estudios primarios*, se basa en la información obtenida en el capítulo anterior como resultado de la Revisión Sistemática de Literatura realizada.

El paso 2, *Extracción de datos de las Teorías Organizacionales y del Comportamiento humano*, fue aplicado para la selección de nuevos factores que influyen en el desarrollo de software ágil y la calidad del producto desarrollado; para ello, se revisaron 77 teorías, de las cuales se realizó una pre-selección de aquellas que contribuían con la construcción del modelo conceptual, al final nueve fueron las teorías que contribuyeron a la determinación de seis nuevos factores críticos de éxito.

El paso 3, *Consolidación de Datos*, se aplicó tanto a los tres componentes principales del modelo: Factores críticos de éxito, Prácticas ágiles y Características de calidad de software, en esta etapa se eliminaron conceptos duplicados de manera explícita y duplicados en cuanto a su conceptualización considerando su relación con el desarrollo de software ágil. Luego, se determinaron las dimensiones de los Factores críticos de éxito, las categorías de las prácticas

ágiles, y características a analizar , las mismas que están basadas en el estándar ISO/IEC 25010. Además, en este paso se asignaron factores a las dimensiones y prácticas a las categorías, como se describirá en el desarrollo de este capítulo.

En el paso 4, *Identificación de relaciones*, se determinaron las relaciones existentes entre:
a) Factores críticos de éxito y su influencia en la aplicación de las Prácticas ágiles, b) Prácticas ágiles y su influencia en la calidad del software, y c) Factores críticos de éxito y su influencia en la calidad del software.

En el paso 5, *Revisión final*, se realizó un refinamiento del modelo y su fundamentación, de tal forma que cada componente del modelo tenga su adecuada fundamentación teórica.

El paso 6, Modelo conceptual propuesto, permitió contar con un modelo estructurado en cuatro componentes, los mismos que se detallan a continuación.

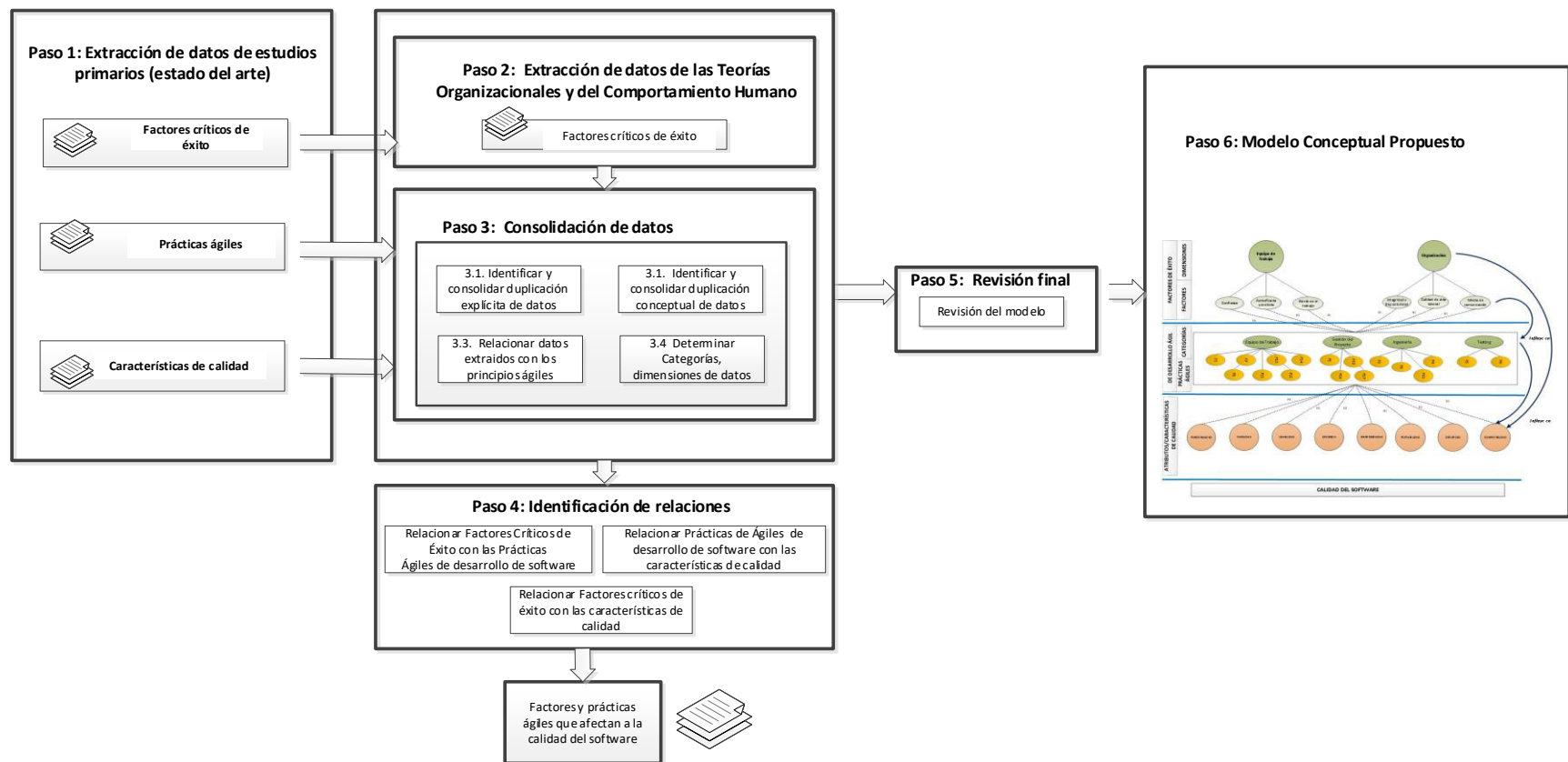


Figura 4-1 Metodología para la construcción del modelo conceptual

4.2 Estructura del modelo conceptual

El modelo conceptual propuesto tiene como propósito determinar nuevos factores críticos de éxito y su influencia positiva o negativa en la aplicación de prácticas ágiles en proceso de desarrollo de software; además, permite establecer la influencia de la aplicación de estas prácticas en la calidad del software. El modelo propuesto tiene los siguientes componentes.

- 1) Componente 1: Factores críticos de éxito.
- 2) Componente 2: Prácticas ágiles.
- 3) Componente 3: Características de calidad.
- 4) Componente 4: Hipótesis

La Figura 4-2 Modelo conceptual representa una visión general de los componentes del modelo conceptual. La descripción y fundamentación de estos componentes se describen a continuación.

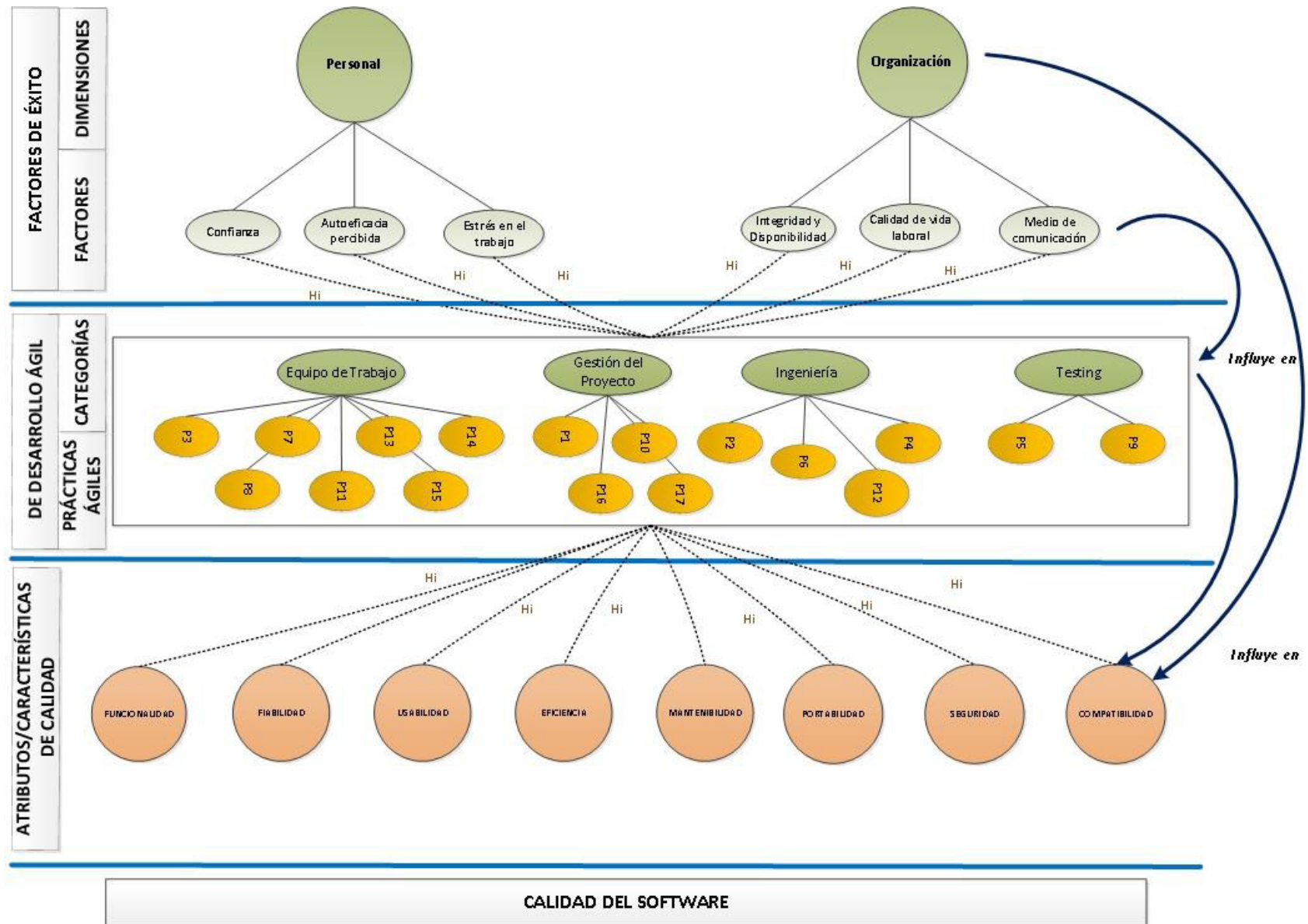


Figura 4-2 Modelo conceptual

4.2.1 Componente 1: Factores críticos de éxito

Los factores críticos de éxito presentados en este estudio han sido obtenidos de la revisión de teoría enfocadas al ámbito humano y comportamiento organizacional, estas teorías fueron tomadas de (Larsen & Eargle, 2017), una base de datos que provee a los investigadores una información resumida de las teorías más ampliamente usadas en investigaciones en sistemas de información, de las cuales fueron seleccionadas las teorías que se relacionan con los principios ágiles en el ámbito de personal y organizacional.

Cualquier transformación organizacional que involucre a numerosas personas enfrentará desafíos, basado en esto, (Dikert et al., 2016) encontró 35 desafíos, organizados en nueve categorías, muchos de ellos están relacionados con la organización y el personal, por ejemplo: resistencia al cambio, falta de inversión, gestión jerárquica y limitaciones organizacionales, desafíos de coordinación en entornos de equipos múltiples, entre otros.

Los factores críticos de éxito de este estudio se han obtenido considerando la duplicación explícita y conceptual de los factores de la literatura y las teorías organizacionales y técnicas. En base a esto se han determinado 6 factores importantes que influyen en el desarrollo de software utilizando modelos ágiles, éstos han sido clasificados en dos dimensiones:

- a) ***Personal***, aspectos relacionados con el comportamiento colectivo e individual de las personas que realizan el proyecto;
- b) ***Estructura organizacional***, relacionada con la forma de operar de una organización para contribuir al logro de sus objetivos y las condiciones del ambiente laboral que motiva el trabajo de sus trabajadores.

La Tabla 4-1 resume los factores críticos de éxito que han sido considerados en este estudio, su definición y teoría que la soporta.

Tabla 4-1 Factores críticos de éxito propuestos

Id	Factores Críticos de éxito	Definición	Teoría que lo soporta
Dimensión: personal			
FCO	Confianza	Es la opinión favorable en la que una persona o grupo es capaz de actuar de forma correcta en una determinada situación. La confianza también puede definirse como la seguridad que una persona tiene en ella misma o en otra persona.	Teoría de la Equidad (Adams & Freedman, 1976) Teoría del capital Social (Coleman, 2005)
FSE	Autoeficacia percibida	Es la confianza en la capacidad de uno mismo para llevar a cabo una tarea o alcanzar una meta determinada	Teoría de la autoeficacia (Bandura, 1977)
FSTR	Estrés	Estado de cansancio mental provocado por la exigencia de un rendimiento muy superior al normal; suele provocar diversos trastornos físicos y mentales.	Teoría del cierre de tareas (Straub & Karahanna, 1998)
Dimensión: Organización			
FIAI	Integridad y disponibilidad de la información y experiencias aprendidas.	<i>Integridad:</i> Garantía de la exactitud y completitud de la información y los métodos para su procesamiento. <i>Disponibilidad:</i> aseguramiento de que las personas autorizadas tienen acceso cuando lo requieran a la información y sus elementos asociados.	Teoría de la integración de la información. (Carroll, 1982) (Wegner, 1987) Teoría de la empresa basada en el conocimiento (Grant, 1996)
FQL	Calidad de vida laboral	Se refiere al carácter positivo o negativo de un ambiente laboral. La finalidad básica es crear un ambiente que sea excelente para los empleados, además de que contribuye a la salud económica de la organización.	Teoría Socio Técnica (Bostrom & Heinen, 1977)
FCM	Medios de comunicación usados	Es la tecnología que sirve como instrumento para mantener informados a los miembros un equipo de trabajo con intereses comunes. Hace una gran referencia al instrumento o forma de contenido por el cual se realiza el proceso comunicacional	Teoría de la riqueza de los medios (Daft & Lengel, 1984)

Dimensión: Personal

En la dimensión *Personal* los nuevos factores identificados son: *Confidencia*, *Autoeficacia percibida*, y *estrés en el trabajo*, que están basados en: *Teoría de la Equidad*, *Teoría del Capital Social*, *Teoría de la Autoeficacia*, *Teoría del Cierre de Tareas*, el sustento de cada uno de estos factores y su relación con las prácticas ágiles se indican a continuación:

La **Teoría de la Equidad**, está basada en una estructura de la equidad en el lugar de trabajo y la relación entre *entradas y resultados*; una de las *entradas* que esta teoría considera es la *confianza en los superiores*. Por otro lado, el principio ágil No. 4 indica que “*Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.*”, de acuerdo a esto, se establece una relación de confianza entre los responsables del negocio y los desarrolladores.

De la misma manera, la **Teoría del Capital Social**, es definida como un conjunto de características de una organización social, que está presente en las estructuras de relaciones interpersonales e intersectoriales de una sociedad representada forma de normas, redes de relaciones interpersonales y *confianza*. El principio No. 5 del manifiesto ágil “*Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo*”; como se puede apreciar, existe una relación entre la *Confianza* con los principios ágiles, razón por la cual, ha sido considerada como uno de los factores de éxito de un proyecto ágil que serán estudiados en este trabajo.

La **Teoría de la Autoeficacia**, se resume en que los individuos se programan a sí mismos para el éxito o para el fracaso. Si alguien tiene nivel de autoeficacia alto, también tendrá un alto porcentaje de cumplir con los objetivos planteados. Por otra parte, el principio ágil No. 12 indica que “*A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.*”, por esta razón, la autoeficacia percibida, es un factor importante directamente relacionado con las prácticas ágiles.

La **Teoría de Cierre de tareas** considera que la división de las tareas y la selección adecuada de los medios de comunicación, son muy importantes para obtener un mayor nivel de productividad, ya que los seres humanos sienten la necesidad de darle un cierre a una secuencia de comunicaciones y requieren de medios que les permitan sentir que han hecho esto; además, la selección de los medios de comunicación que lleva a la tarea de cierre, conducirán a niveles más bajos de la fragmentación de tareas y el *estrés laboral*; este último ha sido considerado en este estudio para determinar su influencia en la aplicación de prácticas ágiles.

Dimensión: Organización

Los factores relacionados con la organización son tres: 1) *Integridad y Disponibilidad de la Información y Experiencias Aprendidas*, 2) *Calidad de Vida en el Trabajo* y 3) *Medios de comunicación usados*, están basados en las teorías de *Integración de la información*, *Teoría basada en el conocimiento de la firma*, *Teoría Socio Técnica*, *Teoría de la riqueza de los medios*.

La *Teoría de la Integración de la información*, parte de dos aspectos importantes acerca de la conciencia: a) que tiene información sobre su propia experiencia, y b) que la experiencia se integra siempre y cuando estas partes son informativas unas de otras.

Así mismo, la *Teoría de la Memoria transitiva* se basa en la idea de que los participantes pueden compartir y beneficiarse de los conocimientos y experiencia de los otros, sobre todo en casos cuando se desarrolla una buena comprensión de la información y cuando ésta es compartida con quien más sabe en el equipo de trabajo.

La *Teoría basada en el conocimiento de la empresa* (KBF, por sus siglas en inglés), considera el conocimiento como el recurso de más importancia estratégica de una empresa; este conocimiento está inmerso en múltiples entidades incluyendo la cultura organizacional, políticas, rutinas, documentos, sistemas, y empleados; es decir que el conocimiento y experiencias adquiridas por el equipo de trabajo, deben ser compartidas para obtener una mayor productividad, y rendimiento y calidad de los productos. Así mismo, el principio No. 4 del desarrollo ágil indica que “*Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.*”, por lo tanto, para que el trabajo sea efectivo se debe compartir conocimientos y experiencias adquiridas; basados en estas teorías se ha considerado en este estudio el factor: *Integridad y disponibilidad de la información y experiencias adquiridas*.

La *Teoría Socio Técnica*, relaciona el subsistema técnico y el subsistema social que en conjunto forman una organización; *el subsistema técnico* comprende los dispositivos,

herramientas y técnicas necesarias para transformar los insumos en productos de una manera que mejora el rendimiento económico de la organización, el *subsistema social* comprende los empleados (en todos los niveles) y los conocimientos, habilidades, actitudes, valores y necesidades que traen al ambiente de trabajo, así como el sistema de recompensa y las estructuras de autoridad que existen en la organización. Una organización mejorará su rendimiento si existe un reconocimiento explícito de la interdependencia de estos dos subsistemas. Según esta teoría, la principal tarea es incrementar la *calidad de vida en el trabajo*, que ha sido considerado como un factor a analizarse en este estudio, y la satisfacción en el trabajo del empleado. Además, estos factores también se basan en el principio ágil No. 5, mencionado anteriormente.

La *Teoría del Cierre de tareas*, como se mencionó anteriormente, promueve la selección adecuada de los medios de comunicación para obtener un mayor nivel de productividad, ya que los seres humanos sienten la necesidad de darle un cierre a una secuencia de comunicaciones y requieren de medios que les permitan sentir que han hecho esto.

La *Teoría de Riqueza de los Medios de Comunicación*, clasifica y valora la riqueza de ciertos medios de comunicación, tales como llamadas telefónicas, videoconferencias, correo electrónico, etc. Esta teoría explica cuan ricos y eficaces son los medios de comunicación personales para la comunicación. Un mensaje simple con la intención de organizar un horario de reunión podría ser comunicada por un breve correo electrónico, pero un mensaje más detallado sobre el rendimiento y las expectativas de trabajo de una persona estaría mejor comunicada a través de la interacción cara a cara; como lo establece el principio No. 6 del desarrollo ágil indica que “*El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.*”; por lo tanto, el factor *Medios de comunicación usados*, fue considerado como un factor crítico de éxito a ser analizado.

4.2.2 Componente 2: Prácticas ágiles

El modelo propuesto ha considerado 17 prácticas o acciones utilizadas en el proceso de

desarrollo ágil. Se han considerado un conjunto de prácticas que son comunes a la mayoría de ellas, a pesar de las diferencias particulares que existen entre las distintas metodologías de ágiles.

Las prácticas que han sido seleccionadas se basan principalmente en el estudio de (Xu, 2009), quien considera once prácticas ágiles tomando en cuenta la perspectiva de la gestión de calidad aplicadas en XP que son: “*Planificación del juego, Pequeñas versiones, Diseño simple, Testing, Refactoring, Programación en pareja, Propiedad colectiva, Integración continua, 40 horas por semana, Cliente in situ, Estándares de codificación*” . De acuerdo a los últimos cinco Reportes Anuales del Estado de Ágil (Versionone, 2017), se consideran *reunionse diarias* y *retrospectivas*, dentro del top 5 agile techniques, razón por la que estas prácticas fueron incrementadas en este estudio. En el estudio de (Diebold & Dahlem, 2014), *Timeboxing* fue considerada la práctica ágil más utilizada, por lo que también fue incluida en este trabajo; de este mismo estudio se consideró las prácticas de: *Monitoreo del progreso* en vista que toda planificación realizada en los proyectos, debe ser controlada, y *Pequeños equipos multifuncionales*, que en este estudio se llamó *Equipos multifuncionales*, debido a la importancia del trabajo en equipo en las metodologías ágiles. Con este mismo criterio, se seleccionó la práctica de *Auto-organización del equipo*, tomada del estudio de (Ahmed et al., 2010).

Las prácticas seleccionadas fueron divididas en cuatro categorías, de acuerdo a lo descrito en el Capítulo 3. Un listado de estas prácticas y la categoría a la que pertenecen se muestran en la Tabla 4-2.

Tabla 4-2 Prácticas ágiles

ID	Dimensión	Prácticas ágiles
PTW	Equipo de Trabajo	P1 Programación en pareja
		P2 Cliente on-site
		P3 Reuniones diarias
		P4 Equipos auto-organizados
		P5 Equipos multifuncionales
		P6 Retrospectivas
		P7 Propiedad colectiva del código
PMP	Gestión del Proyecto	P8 Planificación
		P9 Monitoreo del progreso
		P10 40 horas por semana
		P11 Timeboxing

ID	Dimensión	Prácticas ágiles
PENG	Ingeniería	P12 Pequeñas versiones P13 Diseño simple P14 Refactoring P15 Estándares de codificación
PTEST	Testing	P16 Pruebas unitarias P17 Integración continua

4.2.3 Componente 3: Características de calidad

Tabla 4-3 Características de calidad del software

ID	Característica	Descripción
AFUN	Funcionalidad	“Conjunto de atributos que están relacionados con la existencia de las funciones del software y sus propiedades específicas”.
ARELI	Fiabilidad	“Atributos que representan la capacidad del software para mantener su nivel de rendimiento bajo ciertas condiciones y durante un período de tiempo establecido”.
AUSAB	Usabilidad	“Atributos que están relacionados con el esfuerzo requerido para aprender el manejo de una aplicación, trabajar con ella y conseguir resultados”.
APEF	Eficiencia	“Conjunto de atributos que representan la relación entre el nivel de rendimiento de software y la cantidad de recursos hardware y software utilizados bajo condiciones establecidas”.
ACOMP	Compatibilidad	“Capacidad de dos o más sistemas o componentes para intercambiar información y/o realizar sus funciones requeridas mientras comparten el mismo hardware o software”.
AMAIN	Mantenibilidad	“Conjunto de atributos relacionados con el esfuerzo necesario para realizar modificaciones”.
APORT	Portabilidad	“Atributos que representan la capacidad del software para ser transferido de un entorno a otro”.
ASEC	Seguridad	“Grado de protección de la información y de los datos, que hacen que las personas o sistemas no autorizados no puedan leerlos o modificarlos”.

Fuente: Estándar ISO/IEC 25010

El modelo propuesto considera las ocho características de calidad que constan en el estándar ISO/IEC 25010, cuya descripción consta en la Tabla 4-3; estas características permitirán

determinar la calidad interna del software, las mismas que dependen directamente de su proceso de desarrollo.

4.2.4 Componente 4: Hipótesis

El modelo propuesto representa dos grupos de relaciones, que se consideraron como hipótesis de la investigación.

- a) Las relaciones entre los factores críticos de éxito y la aplicación de las prácticas ágiles para determinar el tipo de influencia entre ellos
- b) Las relaciones entre la aplicación de prácticas ágiles y las características de calidad del software.

Las hipótesis a considerar fueron establecidas en reunión de expertos considerando las relaciones factibles entre los aspectos a relacionar. Por esta razón no se probaron todas las relaciones entre factores críticos de éxito y prácticas ágiles, así como tampoco se probaron todas las relaciones entre prácticas ágiles y características de calidad de software.

La eliminación de las hipótesis no probadas, se realizó mediante la asignación de una valoración correspondiente a la escala de Likert (1 para una relación débil y 5 para una relación fuerte) el cálculo de las medias de cada relación permitió realizar la selección de las hipótesis a ser validadas. Las relaciones cuyo valor de la media fue mayor que 3,4 fueron las seleccionadas. Los resultados de este proceso se encuentran en los Anexos H e I.

Grupo 1: Relaciones Factores críticos de éxito – prácticas ágiles

El modelo representa las relaciones para determinar la influencia positiva o negativa que existen entre los factores críticos de éxito y las prácticas ágiles. Considerando que existen 6 factores críticos de éxito, 4 categorías de prácticas ágiles a ser analizadas en el modelo. Se han generado 13 hipótesis derivadas, que han sido codificadas como $H_{i,j}$, siendo i el factor a analizarse y j la práctica ágil con la que se relaciona el factor. Por ejemplo, la hipótesis H6.2

corresponde a la relación entre el factor 6 *Medios de comunicación usados (FCM)* y la categoría de Prácticas 2, *Prácticas de Gestión de Proyectos (PMP)*.

Hipótesis del Factor 1: Confianza

La *Confianza* (FCO), es un factor humano que puede influir en la organización del equipo de trabajo, es decir, en las prácticas de *Programación en pareja*, *Autoorganización de equipos*, *Equipos multifuncionales*, *Retrospectivas* y *Propiedad colectiva del código*, ya que es importante que el grupo actúe de forma correcta, contribuya a solucionar los problemas y comparta la información generada durante todo el proceso de desarrollo de software; de allí se genera la *H1.1: “Existe una influencia significativa entre la Confianza y las Prácticas del equipo de trabajo”*.

El cumplimiento de las prácticas ágiles de *Diseño Simple*, *Estándares de Codificación* y *Refactoring* están influenciadas por el factor FCO, es decir, que dependerá de la Confianza que se tenga a los miembros del grupo de desarrollo; por lo tanto, se consideró la *H1.3: “Existe una influencia significativa entre la Confianza y las Prácticas de Ingeniería”*.

Como consecuencia de lo anterior, la realización completa y correcta de *Pruebas unitarias e Integración continua*, basada en la confianza que el tester realice completamente su trabajo; por lo que se plantea la *H1.4 “Existe una influencia significativa entre la Confianza y las Prácticas de testing”*.

Hipótesis del Factor 2: Autoeficacia percibida

La autoeficacia percibida (FSE), es decir, la autoconfianza en la capacidad de cada miembro del equipo de trabajo en realizar correctamente prácticas como: *Programación en pareja*, *Reuniones de pie*, *equipos auto-organizados*, *Equipos multifuncionales* y *Retrospectivas*, son importante para contribuir en la organización del equipo de trabajo y solucionar problemas que pueden ocurrir durante el proceso de desarrollo; de allí se genera la *H2.1: “Existe una influencia significativa entre la Autoeficacia percibida y las Prácticas de*

Equipos de Trabajo”.

Por otra parte, la realización correcta de las prácticas de *Pruebas unitarias e Integración continua*, dependerán de la autoconfianza de los testers en que estas tareas se han realizado en forma completa y correcta, de tal forma que contribuya a una entrega de un producto de calidad; esto conlleva a la H2.4: “*Existe una influencia significativa entre la autoeficacia percibida y Prácticas de Testing*”.

Hipótesis del Factor 3: Estrés

El *Estrés* (FSTR) en el equipo de trabajo es un factor ocasionado por la exigencia de un rendimiento superior al normal, debido posiblemente al no cumplimiento de las prácticas de *Planificación del proyecto y Progreso del monitoreo*, lo que puede provocar el trabajo de jornadas adicionales de trabajo y por lo tanto al incumplimiento de la práctica *Semana de 40 horas*; por lo que se propone la H3.2 “*Existe una influencia significativa entre el Estrés y las prácticas de gestión del proyecto*”.

Además, FSTR puede ocasionar que la optimización del código fuente a través de la Práctica de *Refactoring* no sea realizada adecuadamente; así mismo, puede hacer que el programador no considere los *estándares de codificación*, ocasionando dificultades en el futuro mantenimiento del software; esto permite plantear la H3.3 “*Existe una influencia significativa entre el Estrés and las Prácticas de Ingeniería*”.

Otro de los aspectos que pueden ser afectados con el *Estrés* es una incorrecta e incompleta realización de *pruebas unitarias e integración continua*, es decir, que es otro factor que influiría en la entrega de un producto de calidad; esto lleva a la H3.4: “*Existe una influencia significativa entre el Estrés y las Prácticas de Testing*”.

Hipótesis del Factor 4: Integridad y disponibilidad de la información y experiencias aprendidas

Toda la información generada en el desarrollo del proyecto, debe ser correcta, completa, y estar a disponibilidad de todo el equipo de trabajo; así mismo, es importante documentar todas las experiencias adquiridas, de tal forma que ayude a los nuevos miembros del equipo o a desarrolladores de otros proyectos; este factor contribuiría a que prácticas como *Programación en pareja*, *Cliente on-site*, *Equipos auto-organizados*, *Equipos multifuncionales*, *Retrospectivas* *Propiedad Colectiva del código*, puedan desarrollarse de una manera óptima; esto conduce a plantear la *H4.1*: “*Existe una influencia significativa entre la Integridad y Disponibilidad de la Información y experiencias aprendidas y las Prácticas del Equipo de Trabajo*”.

El código del software actualizado y disponible a todos los testers será indispensable para aplicar *Pruebas unitarias* y *Prácticas de Integración continua*; por lo que se obtiene la *H4.4*: “*Existe una influencia significativa entre Integridad y Disponibilidad de la Información y experiencias aprendidas y las Prácticas de Testing*”.

Hipótesis del Factor 5: Calidad de vida laboral

Este factor tiene como objetivo crear un ambiente óptimo para que el equipo de trabajo realice un trabajo eficiente y de calidad, el mismo que será reflejado en la calidad del producto entregado al cliente. Una de las prácticas que determinará la calidad del producto es la correcta y completa ejecución de las *Pruebas Unitarias* y *Prácticas de Integración Continua*; por lo que se plantea la *H5.4*: “*Existe una influencia significativa entre Calidad de Vida Laboral y Prácticas de Tesing*”.

Hipótesis del Factor 6: Medios de comunicación usados

Los *Medios de Comunicación Usados* (FCM), son un elemento indispensable para el cumplimiento de las Practicas del Equipo de Trabajo, ya que contribuirá a la realización de muchas tareas que no necesariamente requieren la interacción cara a cara entre sus actores, como es el caso de las siguientes prácticas: *Programación en Pareja*, *Cliente On-site*, *Reuniones de pie*, *Equipos auto-organizados*, *Retrospectivas* y *Propiedad Colectiva del Código*, por lo que se plantea la *H6.1*: “*Existe una influencia significativa entre los Medios de Comunicación Usados*

y los las Prácticas del Equipo de Trabajo”.

La adecuada *Gestión del Proyecto*, depende del factor FCM ya que su correcta utilización facilita las Prácticas de Timeboxing, Planificación y Monitoreo del Progreso, debido a esto se ha incluido H6.2: “*Existe una influencia significativa entre los Medios de Comunicación usados y las Prácticas de Gestión del Proyecto*”.

El resumen de las hipótesis del Grupo 1, que plantea las relaciones existentes entre los factores críticos de éxito y las prácticas ágiles, se muestran en la Tabla 4-4. Estas hipótesis han sido codificadas como $H_{i,j}$, siendo i los factores críticos de éxito a analizarse a analizarse y j la práctica ágil con la que se relaciona este factor. Por ejemplo, la hipótesis H2.1 corresponde a la relación entre el Factor 2 *Auto-Eficacia Percibida* (FSE) y la categoría de prácticas ágiles 1 *Prácticas del Equipo de trabajo* (PTW). Como se puede apreciar, 11 relaciones no han sido consideradas, en vista que no tienen el suficiente fundamento lógico que lo sustente.

Tabla 4-4 Hipótesis derivadas del Grupo 1

Factores	Categorías de Prácticas Ágiles			
	1. PWT	2. PMP	3. PENG	4. PTEST
1. FCO	H1.1		H1.3	H1.4
2. FSE	H2.1			H2.4
3. FSTR		H3.2	H3.3	H3.4
4. FIAI	H4.1			H4.4
5. FQWL				H5.4
6. FCM	H6.1	H6.2		

Grupo de hipótesis 2: Prácticas ágiles – Características de calidad

La segunda parte del modelo representa las relaciones entre las prácticas ágiles y las características de calidad. Considerando que existen 4 categorías de prácticas ágiles y 8 características de calidad a ser analizadas en el modelo, se han generado 13 hipótesis derivadas, las mismas que se muestran en la Tabla 4-5 **Hipótesis derivadas del Grupo 2**, estas hipótesis han sido codificadas como $H_{i,j}$, siendo i la categoría de práctica ágil a analizarse y j la

característica de calidad de software con la que se relaciona práctica. Por ejemplo, la hipótesis *H3.2* corresponde a la relación entre la Categoría de Prácticas Ágiles 3 *Prácticas de Ingeniería* (PENG) y la característica de calidad 2 *Fiabilidad* (ARELI). El sustento de las hipótesis planteadas se detalla a continuación.

Set de Hipótesis de la Categoría de Prácticas 1: Equipo de trabajo

La categoría de prácticas de *Equipo de Trabajo*, puede ejercer una influencia significativa en la característica de calidad de *Funcionalidad*, debido a la aplicación de prácticas como: *Cliente on-site*, *Reuniones diarias*, *Retrospectivas* y *Propiedad colectiva del código*, que contribuyen a la obtención de una funcionalidad completa y correcta del software. De allí que se plantea: *H1.1 “existe una influencia significativa de las prácticas de Equipo de Trabajo sobre la Afecución Funcional ”*

Set de Hipótesis de la Categoría de Prácticas 2: Gestión del proyecto

La categoría de prácticas de *Gestión del Proyecto*, incluyen prácticas relacionadas con la *Planificación*, *Monitoreo*, *Jornada de trabajo* y *Timeboxing*, las mismas que no influyen directamente en las características de calidad de software; sin embargo, se ha considerado que una correcta planificación y monitoreo del cumplimiento de los requisitos del usuario contribuirá a obtener una correcta funcionalidad. Por esta razón se ha planteado, la hipótesis: *H2.1 “existe una influencia significativa de las Prácticas de Gestión del proyecto sobre la Funcionalidad”*

Set de Hipótesis de la Categoría de Prácticas 3: Ingeniería

Se consideró que la categoría de prácticas de *Ingeniería*, es decir, la aplicación de *Pequeñas versiones* de software, *Diseño Simple*, *Refactoring* y *Estándares de Codificación* contribuyen en la construcción correcta del software, en sus etapas de análisis, programación y codificación del software y por lo tanto ayudarán a conseguir un software que cumpla con las características de calidad. Se ha obviado en este set de hipótesis la característica de calidad de *Portabilidad*,

por considerar que las prácticas de *Ingeniería* no contribuyen de manera significativa en la *Instalabilidad y Reemplazabilidad* del software. Como consecuencia de esto, se han considerado las siguientes siete hipótesis:

H3.1, H3.2, H3.3, H3.4, H3.5, H3.6, H3.8 “*Existe una influencia significativa de prácticas de Ingeniería sobre la Funcionalidad, Fiabilidad, Usabilidad, Eficiencia, Compatibilidad, Mantenibilidad y Seguridad*”.

Set de Hipótesis de la Categoría de Prácticas 4: Testing

Las prácticas correspondientes a la categoría de prácticas de *Testing*, son sumamente importantes en las etapas del proceso de desarrollo, garantiza que éste sea confiable y la satisfacción del cliente al recibir el producto final. El propósito de estas prácticas es encontrar errores desde el inicio del desarrollo de software y que se tomen acciones correctivas de una manera oportuna, contribuyendo de esta manera a tener un proceso y un producto de calidad.

En est trabajo se ha priorizado la influencia de la aplicación de las prácticas de testing en cuatro características de calidad: *Funcionalidad, Fiabilidad, Eficiencia y Portabilidad*, que han sido consideradas mediante las hipótesis H4.1, H4.2, H4.4 y H4.7.

Las características que no han sido incluidas en las hipótesis también son muy importantes, sin embargo, son consideradas como una consecuencia la aplicación de todas las prácticas ágiles, especialmente las prácticas de *Ingeniería*.

Tabla 4-5 Hipótesis derivadas del Grupo 2

Prácticas Ágiles	Características de Calidad							
	Funcionalidad	Fiabilidad	Usabilidad	Eficiencia	Compatibilidad	Mantenibilidad	Portabilidad	Seguridad
PTW	H1.1							
PMP	H2.1							
PENG	H3.1	H3.2	H3.3	H3.4	H3.5	H3.6		H3.8
PTEST	H4.1	H4.2		H4.4			H4.7	

Los siguientes capítulos están orientados a comprobar estas hipótesis planteadas.

4.3 Conclusiones del capítulo

Se ha propuesto un modelo conceptual para determinar la influencia de los factores críticos de éxito y prácticas ágiles sobre la calidad de software. El modelo está formado por cuatro componentes: 1) Seis factores críticos de éxito divididos en dos dimensiones, 2) 17 prácticas ágiles divididas en cuatro categorías, 3) Ocho características de calidad con base en el estándar ISO/IEC 25010 y, 4) 26 Relaciones/hipótesis derivadas de la relación factores críticos de éxito – prácticas ágiles y prácticas ágiles – características de calidad.

Los factores críticos de éxito que conforman el modelo están basados en nueve teorías del ámbito administrativo y de comportamiento humano.

CAPÍTULO 5: INFLUENCIA DE LOS FACTORES CRÍTICOS DE ÉXITO SOBRE LAS PRÁCTICAS ÁGILES DE DESARROLLO DE SOFTWARE

Este capítulo tiene como objetivo determinar la influencia de los factores de tipo organizativo y del personal que trabaja en el proyecto de desarrollo de software en las prácticas ágiles, que constituye la primera parte del modelo propuesto en el capítulo anterior. Para este fin se propone un modelo compuesto por seis factores críticos de éxito tomadas de las teorías del ámbito administrativo y comportamiento humano, cuatro grupos de prácticas ágiles y trece relaciones factores-prácticas. Además se detalla la metodología utilizada en la comprobación de las hipótesis establecidas y los resultados obtenidos de su aplicación.

5.1 Introducción

Las aplicaciones informáticas son muy importantes en todas las áreas de la actividad humana, sin embargo, el desarrollo de estas aplicaciones aún no considerado como un proceso perfecto. Para solucionar estos problemas, varios autores han propuesto la utilización de metodologías para la ingeniería de software, a pesar de ello, su desarrollo no ha sido consistentemente exitoso, lo que se evidencia que muchos proyectos de software se retrasan, se abandonan o son rechazados por los usuarios. Las investigaciones evidencian los factores como el costo, tiempo de entrega y ámbito del proyecto influyen en el éxito de un proyecto y en su calidad (Chow & Cao, 2008) (Agarwal & Rathod, 2006), (Wit, 1988), (Kerzner, 2017).

Tomando en cuenta que las metodologías ágiles fueron difundidas a fin de mejorar la calidad del software y responder más fácilmente a los cambios; las estadísticas demuestran que este fin aún no se ha logrado y la calidad sigue siendo preocupante en el área de la ingeniería de software.

Es importante considerar que la aplicación de prácticas ágiles en las metodologías de desarrollo de software tienen como principal objetivo mejorar la calidad del software; su

utilización se ha visto afectada por una serie de factores de tipo técnico, del personal, de la organización, del proyecto, etc. Varios autores han realizado estudios que identifican factores críticos de éxito que afectan a la calidad de software (Chow & Cao, 2008), (Acuña et al., 2009), (Acuña et al., 2008), (Begier, 2010), (Fernández-Sanz & Misra, 2011); así mismo, se han realizado varias investigaciones que analizan las relaciones entre las prácticas ágiles con la calidad (Santos, 2011), (Curcio et al., 2016a), (Gorla & Lin, 2010), (Bougroun et al., 2014); en las que se evidencian que éstas afectan a la calidad del software.

Estos estudios pueden ser complementados con un análisis de la influencia de los factores críticos de éxito con las prácticas ágiles, ya que es a través de ellas que se mejora la calidad del software, tal como lo establecen los principios ágiles; sin embargo, este tema ha sido poco investigado.

En este estudio se analiza el comportamiento del modelo para identificar los factores que influyen positiva o negativamente en la aplicación de las prácticas ágiles y por consiguiente en la calidad del software. El modelo propuesto, como se describió en el capítulo anterior, está constituido por seis factores críticos de éxito tomadas de las teorías del comportamiento humano y organizacional, clasificadas en dos dimensiones: equipo de trabajo y organización; 17 prácticas ágiles organizadas en cuatro categorías: Equipo de Trabajo, Gestión del Proyecto, Ingeniería y Testing y 13 hipótesis que fueron validadas mediante encuestas realizadas a 146 personas involucradas en el proceso de desarrollo de software utilizando metodologías ágiles en Ecuador.

Este capítulo presenta la primera parte de un estudio que determina la influencia de los factores críticos de éxito en las prácticas ágiles. Sus resultados permitirán determinar las prácticas ágiles que se ven influenciadas por los factores organizacionales y del equipo de trabajo. Esto contribuirá a tomar acciones oportunas para contribuir a mejorar la agilidad del proyecto tomando en cuentas los factores que influyen en su correcta aplicación.

5.2 Motivación

Los estudios que existen sobre factores críticos de éxito y prácticas ágiles se orientan a cuatro ámbitos, de acuerdo a sus contribuciones en los aspectos analizados referentes a la calidad del proceso de desarrollo ágil, como se indica en la

Tabla 5-1.

Tabla 5-1 Ámbitos cubiertos por las practicas ágiles y factores críticos de éxito

No.	Ámbito de estudio	Fuentes
1.	Prácticas ágiles y su relación con la calidad	(Santos, 2011), (Curcio et al., 2016a), (Gorla & Lin, 2010), (Bougroun et al., 2014)
2.	Factores críticos de éxito y su influencia en la calidad	(Chow & Cao, 2008), (Acuña et al., 2009), (Acuña et al., 2008), (Begier, 2010), (Fernández-Sanz & Misra, 2011), (Mirza et al., 2013)
3.	Identificación de factores críticos de éxito	(Chow & Cao, 2008), (Clarke & O'Connor, 2012), (Standish Group, 2015)
4.	Identificación de prácticas ágiles	(Gorla & Lin, 2010), (Subramanyam & Prasad, 2013), (Xu, 2009), (Bermejo et al., 2014), (Opelt & Beeson, 2008), (Versionone, 2015)

Los ámbitos 1 y 2, si bien analizan las relaciones entre las prácticas ágiles y factores críticos de éxito con la calidad, no se enfocan en la relación de estos dos aspectos. Los estudios de los ámbitos 3 y 4 han permitido identificar factores y prácticas ágiles que son utilizadas en el desarrollo ágil y contribuyen con la calidad del software; sin embargo, no se han encontrado estudios que analicen la influencia de los factores críticos de éxito en la aplicación de las prácticas ágiles durante el proceso de desarrollo. Por otro lado, muchos factores críticos de éxito se basan en teorías ya existentes, pero hay factores que se pueden derivar de estas teorías que pueden ser estudiadas y relacionadas con el proceso ágil; por lo que esta investigación aborda estos temas que contribuirán a las empresas e instituciones que desarrollan software a tomar mejores decisiones con respecto a la aplicación de las prácticas ágiles y a controlar los factores que pueden afectar a la aplicación de estas prácticas.

5.3 Modelo conceptual

El modelo conceptual propuesto contiene nuevos factores críticos de éxito y su influencia positiva o negativa en la aplicación de prácticas ágiles en proceso de desarrollo de software. En este capítulo se considerará la primera parte del Modelo Conceptual propuesto en el capítulo anterior, es decir, se considerarán los siguientes componentes:

- Factores críticos de éxito.
- Prácticas ágiles para el proceso de desarrollo ágil.
- Trece relaciones entre factores críticos de éxito - prácticas ágiles, consideradas como hipótesis del estudio.

Cada uno de estos componentes fueron descritos en el Capítulo 4 y se representan en la Figura 5-1.

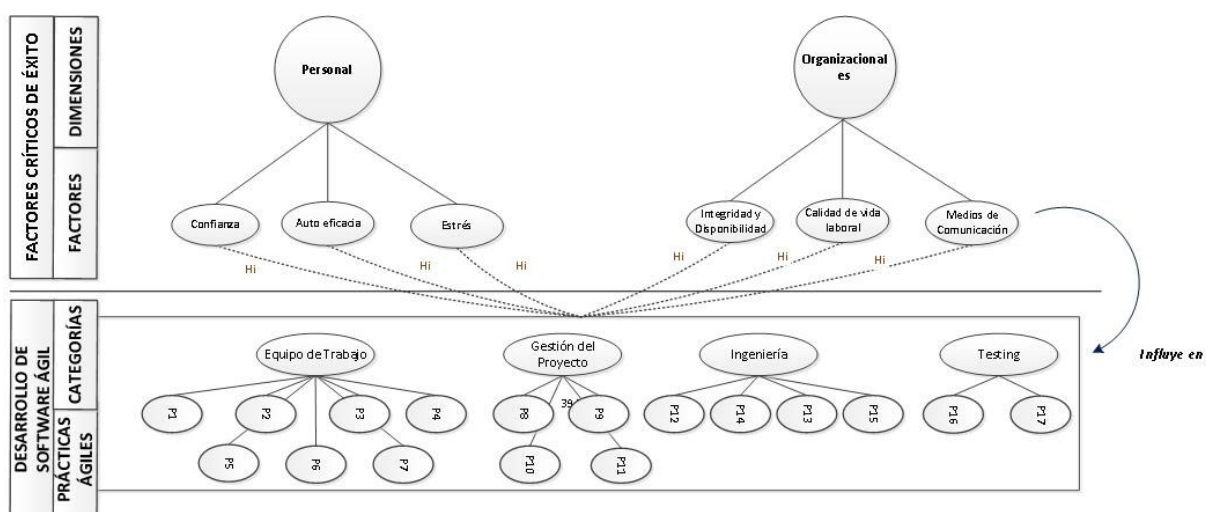


Figura 5-1 Modelo parcial 1: Relaciones entre factores críticos de éxito y prácticas ágiles

5.4 Metodología

El objetivo de este capítulo es determinar la influencia positiva o negativa de los factores críticos de éxito planteados en el modelo sobre las prácticas ágiles.

El estudio es de *tipo no experimental, descriptivo*, para ello se aplicó un *diseño de investigación mixto: cualitativo y cuantitativo*, basado en datos cualitativos, esto implicó la necesidad de un diseño de *investigación fijo* ya que las variables fueron establecidas antes que el estudio sea iniciado y fueron analizadas estadísticamente mediante el *análisis de residuos estandarizados*. (Anastas, 2000).

Esta investigación aplica el método *inductivo y analítico*, ya que interpreta un fenómeno a partir de las explicaciones que las personas presentan para comprender su punto de vista acerca del problema que se está analizando (Wohlin et al., 2012), además, el objeto de estudio ha sido dividido en partes tres partes para cumplir con los objetivos planteados.

La estrategia empírica para cumplir con este objetivo fue la *encuesta* (survey), considerado como un sistema para recopilar información de personas para describir, comparar o explicar sus conocimientos, actitudes y comportamiento (Fink, 2002). La estrategia metodológica aplicada se resume en la Figura 5-2.



Figura 5-2 Estrategia metodológica aplicada
Fuente: (Wohlin et al., 2012)

5.4.1 Recopilación de datos

La investigación se aplicó en un contexto organizacional en empresas e instituciones de Ecuador que se dedican al desarrollan software; para ello se aplicó la técnica de la encuesta a través de un cuestionario en línea generado mediante la herramienta Formularios de Google, la información fue recopilada durante cuatro meses comprendidos entre octubre de 2016 y enero de 2017.

La encuesta estuvo dirigida a 387 desarrolladores y personas involucradas en el desarrollo de software ágil que laboren en empresas e instituciones públicas y privadas de Ecuador. La selección de los participantes se realizó a través de la información proporcionada en el catálogo de empresas de desarrollo de software en este país, así como también de los registros de graduados de posgrado en Ingeniería de Software, Informática Aplicada y Computación de tres universidades de Ecuador. Como resultado de este proceso se obtuvo la respuesta de 155 participantes.

El objetivo de la encuesta fue determinar el punto de vista de los involucrados sobre la influencia de los factores críticos de éxito en la aplicación de prácticas ágiles, así como también, su influencia en las características de calidad del software.

La encuesta estuvo estructurada en 4 secciones: la Sección 1: *Datos Generales* (8 preguntas). La Sección 2: *Percepción de factores organizacionales y de personal que influyen en la aplicación de las prácticas ágiles* (6 preguntas), la Sección 3: *Percepción de prácticas ágiles que influyen en las características de calidad del software* (6 preguntas) y la Sección 4 que contenía aspectos varios relacionados con la experiencia del usuario en calidad y aplicación de las prácticas ágiles.

Las preguntas de la Sección 1 relacionadas con las metodologías ágiles utilizadas y el rol que desempeña el responder en los proyectos de desarrollo de software, fueron de respuesta múltiple ya que esta información puede tener varias respuestas de acuerdo al proyecto en que el encuestado trabaje. Las preguntas de las secciones 2 y 3 fueron evaluadas con a la escala de Likert aplicando una escala de 5 valores:

- 1-No influye,
- 2-Baja influencia,
- 3-Mediana Influencia,
- 4-Alta Influencia, y
- 5-Influencia Total.

Las preguntas de la encuesta se presenta en el Anexo J y se encuentra disponible en:
https://docs.google.com/forms/d/e/1FAIpQLSdtp5yP7OMvEVAYvRxwEtGh4-2I03cc62VdU-f_T435RW4mdQ/formResponse

Una vez que el instrumento fue elaborado, se implementó una prueba piloto a fin de:

- a) Asegurar la validez de la encuesta, es decir, confirmar que ésta mida lo que se necesita medir y esté en concordancia con las hipótesis planteadas.
- b) Analizar si las preguntas están redactadas de una forma adecuada, de tal forma que se garantice una buena comprensión por parte de las personas encuestadas.
- c) Detectar valores no esperados de las variables, determinar si existen flujos de preguntas erróneos, entre otros aspectos.
- d) Determinar si la duración del llenado del cuestionario era la adecuada.

Para esto se seleccionó un grupo de 5 personas relacionadas con el tema, dos masters y tres PhD. La prueba piloto dio permitió realizar algunas correcciones sugeridas por los expertos y mejorar el instrumento diseñado inicialmente. Los cambios incluidos fueron los siguientes:

- a) Incluir conceptos de las variables consideradas en las preguntas para evitar su mala interpretación.
- b) Incrementar preguntas para validar que las respuestas estén en concordancia con la experiencia de los participantes en prácticas ágiles.

Una vez que la información fue recopilada, se realizó una validación de los resultados para eliminar aquellas encuestas que presenten inconsistencias. El principal criterio de elegibilidad de las respuestas fue que los encuestados hayan aplicado metodologías ágiles de desarrollo de software; por lo que el número de cuestionarios válidos se redujo a 146. Los cuestionarios válidos fueron sometidos a un proceso de estandarización de preguntas abiertas y codificación de las variables objeto de estudio.

5.4.2 Método utilizado para el análisis de resultados

El análisis de los datos fue realizado a través de un enfoque cuantitativo. Los resultados están orientados a analizar estadísticamente los datos recopilados en las Secciones 1 y 2 de la encuesta realizada, las mismas que están relacionadas con el objetivo de este estudio y están aplicadas a todos los valores recopilados de las variables estudiadas. Los análisis realizados fueron los siguientes:

- a) Estadística descriptiva, elaborada para conocer las características demográficas de los encuestados.
- b) Test de fiabilidad y validez, mediante alfa de Conbrach para determinar la fiabilidad de la consistencia interna del instrumento y determinar que los ítems medidos en escala de Likert, están altamente correlacionados y su medición corresponde a un mismo constructo.
- c) Análisis de residuos estandarizados. Este análisis se aplicó a todas las respuestas obtenidas de las preguntas de las Secciones 2 y 3 de la encuesta.

Para analizar la hipótesis nula de variables representadas por filas y columnas de una tabla es necesario identificar el valor de cada una de las celdas individuales correspondientes (r) y definirlo como la probabilidad que una desviación estándar exceda el ajuste residual en un porcentaje específico.

Para tal fin se ha adaptado el procedimiento sugerido por (Haberman, 1973) que determina la estandarización de los residuales e_{ij} , mediante la siguiente ecuación:

$$e_{ij} = \frac{a_{ij} - \bar{a}}{\sqrt{\bar{a}}}$$

Donde:

- a_{ij} es el valor dado a cada uno de las celdas individuales formada en la tabla de contingencia,
- $\bar{a}$ es la media de todos los valores dados, lo que se define como el valor esperado.

Para cada e_{ij} la varianza es estimada mediante la ecuación:

$$v_{ij} = \left(1 - \frac{n_{i.}}{N}\right) \left(1 - \frac{n_{.j}}{N}\right)$$

Donde

n_i y n_j son las sumas de todos los valores a_{ij} para la fila i y para la columna j respectivamente y N es la suma de todos los valores a_{ij} .

De esta forma con los valores de e_{ij} y v_{ij} asignados para cada celda de la tabla de contingencia se puede calcular el ajuste residual d_{ij} , mediante:

$$d_{ij} = \frac{e_{ij}}{\sqrt{v_{ij}}}$$

Finalmente, para cada una de las celdas individuales que se encuentran en términos de d_{ij} se realiza un ajuste a la normal mediante la ecuación:

$$d_{ij}' = \frac{d_{ij}}{\sigma_r}$$

Donde σ_r es la desviación estándar de todos los d_{ij} .

Así todos los términos d_{ij}' de la tabla de contingencia se encuentran ajustadas a una distribución normal con media 0 y desviación estándar de 1 y por tanto se pueden comparar con los valores de la unidad tipificada z de una de la distribución normal estandarizada utilizando para este medio la escala estándar de nueve puntos (en inglés *Stanine*), cada *stanine* equivale a una división de 0.5 desviaciones estándar de ancho excluyendo el primer y último *stanine* los cuales incluyen las colas de la curva, por otra parte el *stanine* 5 tiene su centro en la media. La descripción cualitativa usada para cada uno de los nueve *stanines* se describen en la Figura 5-3.

z	Scale		
-1.75	1	Negative Influence	High
-1.25	2		Moderate
-0.75	3		Low
-0.25	4	Average Influence	Slightly below
0.25	5		Average
0.75	6		Slightly above
1.25	7	Positive Influence	Low
1.75	8		Moderate
	9		High

Figura 5-3 Escala Stanines definida descriptivamente.

5.4.3 Validez de los resultados

Se aplicará el Criterio de Confiabilidad Coeficiente alfa de Cronbach que genera como resultados valores entre cero (0) y uno (1). Su fórmula determina el grado de consistencia y precisión de acuerdo a la siguiente escala de valores:

- a) Valores -1 a 0: No es confiable
- b) Valores de 0,01 a 0,49: Baja confiabilidad
- c) Valores de 0,5 a 0,75: Moderada confiabilidad
- d) Valores de 0,76 a 0,89: Fuerte confiabilidad
- e) Valores de 0,9 a 1: Alta confiabilidad

Para esta investigación se consideró confiable un Coeficiente K superior al 75%.

5.5 Resultados de la influencia de los factores críticos de éxito sobre las prácticas ágiles de desarrollo de software

5.5.1 Estadística descriptiva

Se recopiló información de 146 participantes, de los cuales 27,4% mujeres y 72,6% hombres. El mayor porcentaje de los encuestados (76%), trabaja en empresas que tienen como principal actividad el desarrollo de software. El 6,8% de encuestados que trabajan en instituciones de educación superior, alrededor del 6,4% pertenecen a instituciones financieras. El porcentaje restante, correspondiente al 10,8% estuvo distribuido entre participantes que laboran en instituciones de salud, turismo, telecomunicaciones e instituciones gubernamentales. El tamaño de las empresas en donde trabajan los encuestados en su mayoría corresponde empresas pequeñas (52,8%), que tienen menos de 20 empleados; el 32,8% trabajan en empresas que tienen menos de 50 empleados y el 14,4% trabajan en empresas de más de 50 empleados.

En la Figura 5-4, se muestra que el 59,69% de los encuestados, aplican metodología Scrum, seguidos por XP y XP/Scrum híbridos con un 21,99% y 9,95% respectivamente; el porcentaje restante pertenecen a encuestados que utilizan otras metodologías como: Agile UP, FDD o metodologías propias adaptadas por sus empresas.

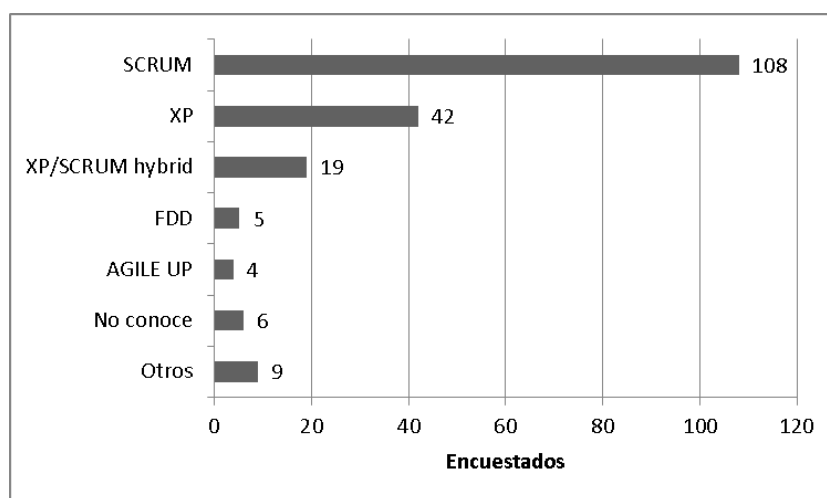


Figura 5-4 Metodologías ágiles utilizadas por los encuestados

En la Figura 5-5, se presenta que el 47,40% de los encuestados son desarrolladores de software que tienen experiencia en metodologías ágiles, seguido de los Jefes de Proyectos, Personal IT y Propietarios del Producto con un 17,92%, 11,56%, y 10,98% respectivamente, los

encuestados que tienen un perfil de testers corresponden únicamente al 4,05%. El 9,59% de los participantes en las encuestas trabajan como: analistas del negocio, analistas de sistemas, y agile coach.

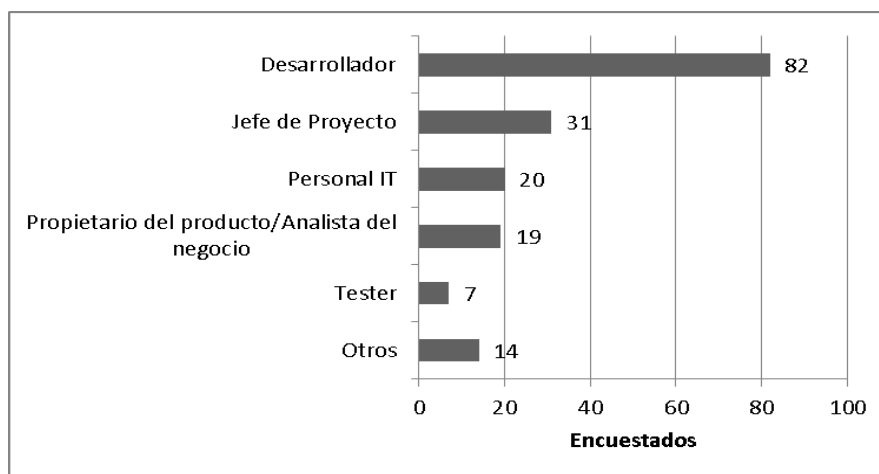


Figura 5-5 Perfil de los encuestados

El Anexo K, presenta un resumen de las estadísticas de las 24 variables analizadas en este capítulo, las mismas que están basadas en una tabla de contingencia de 6x4. Esta variables relacionan los factores críticos de éxito y las prácticas ágiles, de acuerdo a lo establecido en la metodología aplicada. La media de las variables fue de 3,818, los valores obtenidos están dentro de un rango de 0,788 y 3,411 y 4, La media de la varianza de los items fue 1,123 y tiene un rango de valores entre 0,812 y 1,499.

Tabla 5-2 Resumen de estadísticas

	Media	Mínimo	Máximo	Rango	Máximo Mínimo	Varianza	Número de items
Media de los items	3,818	3,411	4,199	0,788	1,231	0,042	24
Varianza de los items	1,123	0,812	1,499	0,687	1,846	0,035	24

5.5.2 Resultados de la influencia de factores sobre las prácticas ágiles

A continuación se presentan los resultados que corresponden al análisis de la influencia que ejercen los factores críticos de éxito sobre las prácticas ágiles (ver Figura 5-1), la misma que han sido calculadas en base la metodología descrita en la sección anterior.

Estadísticas correspondientes a la media y desviación estándar

Los valores obtenidos de la media y la desviación estándar fueron representados a través de un mapa de calor que relacionan los factores (seis filas) y las prácticas ágiles (cuatro columnas), que corresponden a las 24 variables analizadas. Como se puede observar en la Figura 5-6, la influencia FCM-PTW tiene la media más alta con un valor de 4,20, y una desviación estándar de 0,973 (representada con verde oscuro). Esto implica que para la mayoría de los encuestados el factor *Medios de Comunicación* usados ejerce una influencia positiva sobre las prácticas de *Equipo de Trabajo*. Por otra parte, la influencia FSTR-PTEST tiene la media más baja de 3,41 y una desviación estándar de 1,224, lo cual indica que la influencia del factor *Estrés* sobre el *Testing* es débil.

	Agile Practices			
Factors	PTW	PMP	PENG	PTEST
FCO	4,05 ±1,013	3,84 ±1,010	3,61 ±1,085	3,69 ±1,061
FSE	4,08 ±0,965	3,79 ±1,090	3,75 ±1,138	3,79 ±1,032
FSTR	3,69 ±1,224	3,55 ±1,221	3,47 ±1,146	3,41 ±1,224
FIAI	4,04 ±0,901	3,92 ±0,965	3,92 ±1,0	3,95 ±0,992
FQL	3,96 ±0,989	3,81 ±1,059	3,65 ±1,087	3,62 ±1,013
FCM	4,2 ±0,973	4,12 ±0,979	3,86 ±1,070	3,87 ±1,032

Figura 5-6 Mapa de calor de la media y la desviación estándar de las relaciones entre factores y prácticas ágiles

5.5.3 Fiabilidad y viabilidad del test

Para el test de fiabilidad de consistencia interna de los datos recopilados, se utilizó la prueba de Cronbach's Alpha, que utiliza valores entre 0 (una varianza consistente no puede ser definida) and 1 (la varianza es consistente). La fiabilidad y exactitud de las inferencias estadísticas de los

datos se garantizan con un alto valor de Cronbach's Alpha. Un valor aceptable puede considerarse a valores de Cronbach's Alpha igual o superior a 0.70 (Nunnally & Bernstein, 1994). La Tabla 5-3 muestra que el valor del Cronbach's Alpha para las 24 variables analizadas fue 0,958. Esto indica que la consistencia interna de los datos es altamente confiable

Tabla 5-3 Estadísticas de fiabilidad para las 24 variables analizadas

Cronbach's Alpha	Cronbach's Alpha basado en los ítems estandarizados	N. de ítems
0,958	0,958	24

5.5.4 Análisis de residuos ajustados

La Figura 5-7 presenta los residuos ajustados en forma de un mapa de calor. El color verde oscuro representa el más alto índice (influencia positiva) obtenidos por los encuestados. El color rojo oscuro representa el más bajo índice (influencia negativa) obtenidos en los resultados de las encuestas. La significancia está representada por la escala stanines (1-9), en concordancia con la Figura 5-3.

Factors	Agile Practices			
	PTW	PMP	PENG	PTES T
FCO	1,11 (7)	0,09 (5)	-1,02 (3)	-0,61 (4)
FSE	1,27 (8)	-0,12 (5)	-0,41 (4)	-0,16 (5)
FSTR	-0,59 (4)	-1,29 (2)	-1,67 (2)	-1,99 (1)
FIAI	1,12 (7)	0,5 (6)	0,48 (6)	0,67 (6)
FQL	0,67 (6)	-0,07 (5)	-0,8 (3)	-0,94 (3)
FCM	1,85 (9)	1,44 (8)	0,18 (5)	0,27 (6)

Figura 5-7 Mapa de calor de los residuos ajustados entre factores críticos de éxito y prácticas ágiles

Los resultados evidencian que la significancia de la influencia entre el factor *Medio de comunicación utilizado* (FCM) y las prácticas de *Equipo de Trabajo* (PTW) es Alta, $z = 1,85$, donde el valor de z está determinado por d' , determinando una probabilidad de casos mayor a

dicho valor, o una posible influencia positiva del 3%, Además, este factor influye de manera *Moderada* en las Prácticas de *Gestión del Proyecto* (PMP), $z = 1,44$, con una probabilidad de casos mayor que este valor del 7%. La influencia del factor *Autoeficacia Percibida* (FSE) sobre la práctica *Equipo de Trabajo* (PTW) se califica de *Moderada* $z = 1,27$, es decir, una probabilidad de casos mayor que el 10%.

El factor de *Estrés* (FSTR) muestra un resultado muy importante que evidencia una influencia negativa sobre las prácticas de *Testing* (PTEST), del valor de $z = -1,99$ es extremadamente alta, con un valor en la escala stanines menor a 2. Además, existe una influencia negativa *Moderada* de las prácticas de *Ingeniería* (PENG) y *Gestión del Proyecto* (PMP), $Z = -1,67$ y $z = -1,29$ en el 5% y 10% de probabilidad de casos menores que este valor o una posible influencia negativa en los dos casos.

5.6 Discusión de resultados

5.6.1 Aplicación de metodologías ágiles

La Tabla 5-4 muestra el ranking de las metodologías ágiles más usadas en Ecuador, las mismas que han sido contrastadas con los resultados obtenidos por otros estudios. La información obtenida en estos estudios más globalizados es similar a la obtenida en este trabajo para el caso de Ecuador. Los cuatro estudios ubican a Scrum como la metodología ágil más utilizada, XP y una metodología híbrida entre Scrum y XP ocupan los cuatro primeros lugares.

Tabla 5-4 Ranking de metodologías ágiles más usadas

Orden	Este estudio	Diebold (Diebold & Dahlem, 2014)	Ahmed (Ahmed et al., 2010)	Versionone (Versionone, 2020)
1	Scrum	Scrum	Scrum	Scrum
2	XP	Scrum/XP	Agile MSF	Scrumban
3	Scrum/ XP	XP	XP	Otros/Híbridos/ Múltiples metodologías
4	FDD	Scrum/XP/Lean	FDD	Scrum/XP

5.7 Prueba de Hipótesis

5.7.1 Hipótesis de la influencia de prácticas ágiles y características de calidad del software

En la Tabla 5-5 resume los resultados de las hipótesis planteadas sobre influencia de los factores críticos de éxito sobre las prácticas ágiles de desarrollo de software. De las 13 hipótesis propuestas, tres son hipótesis nulas H1.4, H2.4, y H4.4, porque el valor de d' está entre -0.75 y 0.75, valores que no son relevantes, debido a que su valor es insuficiente para determinar si existe una influencia positiva o negativa que permita rechazar la hipótesis nula. Las otras 10 hipótesis pueden ser confirmadas como verdaderas, lo que significa que la influencia entre los factores y las prácticas es suficiente para rechazar la hipótesis nula. Las hipótesis H1.1, H2.1, H4.1, H6.2.1, y H6.2.2 ejercen una influencia positiva, mientras que las hipótesis H1.3, H3.2, H3.3, H3.4 y H5.4 ejercen una influencia negativa.

Tabla 5-5 Resultados de las pruebas de hipótesis

Factores	Categorías de Prácticas Ágiles			
	1. PWT	2. PMP	3. PENG	4. PTEST
1. FCO	<u>H1.1+</u>		<u>H1.3-</u>	H1.4
2. FSE	<u>H2.1+</u>			H2.4
3. FSTR		<u>H3.2-</u>	<u>H3.3-</u>	<u>H3.4-</u>
4. FIAI	<u>H4.1+</u>			H4.4
5. FQWL				<u>H5.4-</u>
6. FCM	<u>H6.1+</u>	<u>H6.2+</u>		

De los resultados cuantitativos obtenidos en este trabajo, se ha evidenciado que todos los factores identificados influyen positiva o negativamente en las prácticas ágiles, como se puede ver en la Figura 5-8.

Los factores *Confianza (FCO)*, *Autoeficacia percibida (FSE)*, *Integridad y disponibilidad de la información y experiencias adquiridas (FIAI)* tiene una influencia positiva sobre las

prácticas ágiles de *Equipo de Trabajo*(PTW), es decir que, estos factores influyen directamente en las prácticas relacionadas con las personas.

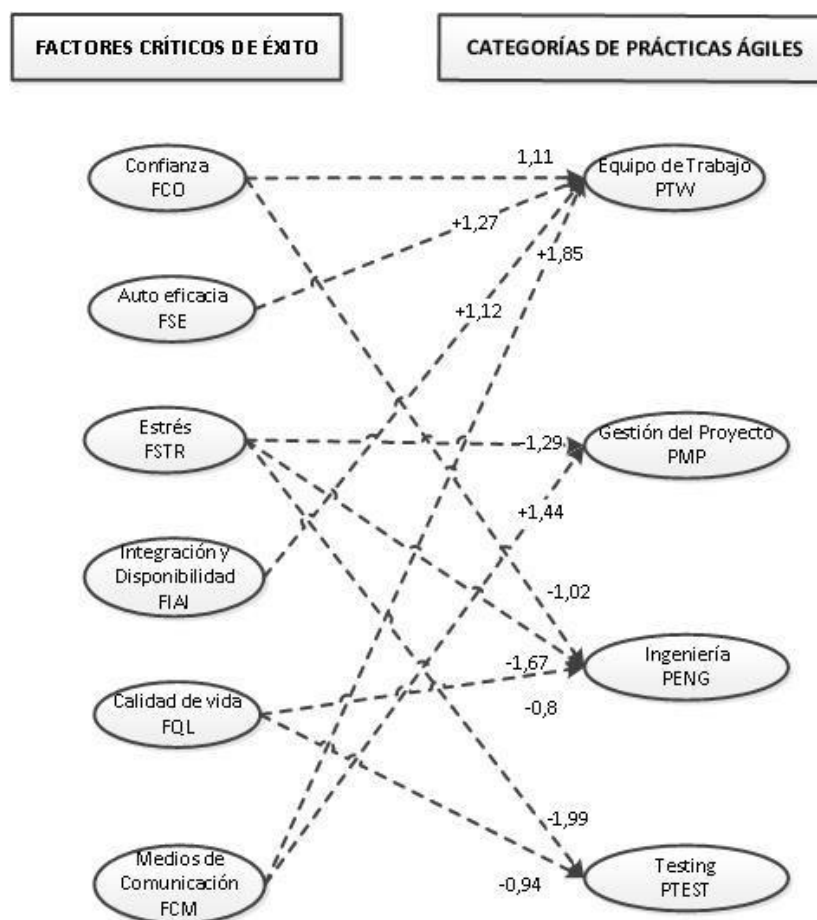


Figura 5-8 Resultado de la influencia de los factores críticos de éxito en las prácticas ágiles de desarrollo de software

Los Medios de Comunicación usados (FCM) es el factor que más positivamente afecta sobre las prácticas de *Equipo de Trabajo (PTW)* y *Gestión del Proyecto (PMP)*. Estos resultados reflejan que la comunicación elegida por el líder del proyecto y sus integrantes afecta la efectividad de sus acciones tales como la coordinación del trabajo, las reuniones, la resolución de problemas y el seguimiento del cumplimiento de las actividades planificadas; los *medios de comunicación usados (FCM)* influyen en la ejecución de estas actividades. Es importante indicar que de acuerdo con los principios y valores ágiles, la forma de comunicación más adecuada cara a cara; sin embargo, pueden utilizarse otras formas de comunicación como correos electrónicos, videoconferencias, etc.

Las *Prácticas de Ingeniería (PENG)*, *Gestión del Proyecto (PMP)*, and *Testing (PTEST)* son influidas negativamente por el factor *Estrés (FSTR)*. El *estrés* es comúnmente visto como una reacción física y emocional que los miembros del equipo de trabajo sufren cuando existen exigencias laborales que no están acorde con sus capacidades y recursos disponibles. Asimismo, el estrés del líder del equipo se puede transmitir al equipo de trabajo aumentando las expectativas puestas en el desarrollo y seguimiento de las actividades. El factor *Estrés (FSTR)* es el factor que más influye de manera negativa en la aplicación de prácticas ágiles.

Otro factor que afecta negativamente en las prácticas ágiles de *Ingeniería (PENG)* y *Testing (PTEST)* son los factores de *Confianza (FCO)* and *Calidad de Vida en el Trabajo (FQWL)* respectivamente. El Jefe del proyecto debe controlar estos factores a través de una adecuada gestión del proyecto, ya que se espera que un equipo con buenas condiciones de trabajo y que goce de la confianza de sus superiores realice su trabajo de manera consistente.

Hay que considerar que los demás factores analizados tienen un nivel de influencia sea esta positiva o negativa en las prácticas ágiles, sin embargo, los valores resultantes no fueron lo suficientemente significativos como para ser ubicados en un rango positivo o negativo. Esto permite determinar que, aunque estos factores se gestionen correctamente, su influencia en las prácticas ágiles no cambiará de forma significativa. Por ejemplo, de acuerdo con los resultados obtenidos, se demostró que FSTR tiene una influencia negativa en las prácticas de *Ingeniería*, *Gestión de Proyectos* y *Testing*. Por lo tanto, si hubiera un ajuste en el control de este factor, los resultados no reflejarían cambios significativos sobre estas prácticas.

El factor *Calidad de Vida Laboral* también tiene una influencia positiva en las prácticas de *Equipo de Trabajo*. Este resultado está en concordancia con lo que se indica en la Teoría Socio-Técnica, que establece el aumento en el desempeño y la productividad de un trabajador depende de la mejora en la calidad de vida laboral. La influencia de este factor, a pesar de ser positiva, no es muy significativa.

En esta parte del estudio se han planteado trece hipótesis, de las cuales diez han sido

confirmadas a través de las pruebas estadísticas presentadas. Las tres hipótesis no respaldadas (H1.4, H2.4 y H4.4) están relacionadas con la práctica de *Testing*. Cuando se estableció este modelo, las hipótesis fueron planteadas bajo el supuesto de que los factores *Confianza (FCO)* y *Autoeficacia Percibida (FSE)* influirían la práctica de *Testing (PTEST)*; sin embargo, los resultados muestran que estos no influyen significativamente en la correcta ejecución de *PTEST*, ya que la profesionalidad del equipo es independiente de la confianza depositada en el equipo de trabajo, y no afecta la completa y correcta ejecución de las prácticas de *Testing*. El factor *Integridad y Disponibilidad de la Información (FIAI)* tampoco influye significativamente en las pruebas. Estos resultados pueden justificarse debido a que la ejecución de las pruebas cuenta con información básica derivadas de la planificación e implementación de pruebas unitarias y de integración, que pueden estar documentadas de forma externa mente o a través del código fuente. El factor *FIAI* no ejerce una influencia sobre los estándares de codificación y las pruebas.

5.8 Conclusiones del capítulo

En este trabajo se ha analizado la influencia de seis factores críticos de éxito que influyen en las prácticas ágiles. Estos factores han sido divididos en dos dimensiones: Dimensión Personal que incluye los factores: *confianza*, *auto eficacia percibida* y *estrés en el trabajo*, que están basados en: Teoría de la Equidad, Teoría del Capital Social, Teoría de la Autoeficacia, Teoría del Cierre de Tareas. Dimensión Organización: que incluye los factores: *Integridad y Disponibilidad de la Información and Experiencias Aprendidas*, *Calidad de vida laboral* y *Medios de comunicación usados*, están basados en: Teorías de la Integración de Información, Teoría de la Memoria Transactiva, Teoría de la Integración de Información, Teoría basada en el Conocimiento de la firma , Teoría Socio Técnica, Teoría de la Memoria Transactiva y Teoría de la Riqueza de los Medios.

Las prácticas ágiles han sido divididas en cuatro categorías: *Equipo de Trabajo*, *Gestión del Proyecto*, *Ingeniería* y *Testing*, de las 165 prácticas identificadas en el estado del arte de este estudio, se han escogido 17 prácticas relacionadas en gran parte con las metodologías Scrum y XP, por ser las metodologías de mayor utilización en nuestro medio.

Los resultados obtenidos en un estudio aplicado a 155 encuestados involucrados en el proceso de desarrollo de software en el Ecuador, muestran que los factores: *Confianza*, (FCO), autoeficacia percibida (FSE), *Integridad y Disponibilidad de la información y experiencias aprendidas* (FIAI) y *Medio de comunicación utilizado* (FCM), tienen influencia significativa positiva con las prácticas ágiles; mientras que *Estrés* (FSTR) tiene influencia negativa sobre la mayoría de las prácticas ágiles; La práctica de *Calidad de vida laboral* (PQL) afecta negativamente a la práctica de *Testing*.

En conclusión, de las trece hipótesis planteadas, diez de ellas han evidenciado que los factores críticos de éxito tienen una influencia significativa en las prácticas ágiles.

CAPÍTULO 6: INFLUENCIA ENTRE PRÁCTICAS ÁGILES Y CARACTERÍSTICAS DE CALIDAD DEL SOFTWARE

Este capítulo tiene como objetivo determinar la influencia de las prácticas ágiles y las características de calidad de software, que constituyen la segunda parte del modelo propuesto en el capítulo cuatro. Para este fin se propone un modelo compuesto por cuatro categorías de prácticas ágiles, ocho características de calidad basadas en el estándar ISO/IEC 25010 y trece relaciones prácticas – atributos de calidad a ser probadas. Además, este estudio se complementa con el análisis de la relación entre los factores críticos de éxito y la calidad del software.

6.1 Introducción

La parte más importante en un proyecto de software es sin lugar a dudas su calidad (Xu, 2009). Algunas metodologías ágiles de desarrollo de software como: XP, Scrum; se basan en prácticas ágiles que han sido establecidas para mejorar la calidad del desarrollo de software y se sus productos. Por lo tanto, se puede afirmar que las prácticas ágiles contribuyen a asegurar la calidad del software (SQA) en un proyecto de desarrollo ágil. (Sagheer, Zafar, & Sirshar, 2015).

Existen estudios que han contribuido a identificar prácticas ágiles y características de calidad que influyen en el desarrollo ágil (Bermejo et al., 2014), (Gorla & Lin, 2010), (Versionone, 2015), (Subramanyam & Prasad, 2013), (Opelt & Beeson, 2008), (Xu, 2009). Además, se han realizado investigaciones que analizan las relaciones entre las prácticas ágiles con la calidad (Santos, 2011), (Curcio, Malucelli, Reinehr, & Paludo, 2016), (Gorla & Lin, 2010), (Bougroun, Zeaaraoui, & Bouchentouf, 2014), en las cuales, el concepto de calidad ha sido analizado como un todo, es decir, no se han considerado las características de calidad y cómo las prácticas ágiles contribuyen con cada una de ellas, esto es importante para que los administradores de proyectos puedan priorizar estas prácticas que contribuyen a la calidad del software.

En este capítulo se analiza la segunda parte del modelo conceptual explicado en el Capítulo 4 de este trabajo en el que se determinan las prácticas que influyen en las características de calidad del proceso de desarrollo aplicando metodologías ágiles. El modelo propuesto está constituido por cuatro categorías de prácticas ágiles: *equipo de trabajo*, *gestión de proyectos*, *ingeniería y testing* y ocho características de calidad basadas en el estándar ISO/IEC 25010 (ISO/IEC, 2005); y 13 relaciones entre las prácticas ágiles y las características de calidad.

6.2 Motivación

La revisión de literatura ha permitido determinar cuatro ámbitos de las investigaciones realizadas y sus contribuciones en los aspectos referentes a la calidad del proceso de desarrollo ágil; éstos han sido abordados desde distintos puntos de vista, los mismos que se resumen en la Tabla 6-1.

Tabla 6-1 Contribuciones realizadas sobre las prácticas ágiles y las características de calidad.

No.	Ámbitos de las investigaciones relacionadas	Autores
1	Prácticas ágiles y su relación con la calidad	(Santos, 2011), (Curcio et al., 2016b), (Gorla & Lin, 2010), (Bougroun et al., 2014), (Xu, 2009)
2.	Identificación de características de calidad	(Mnkandla & Dwolatzky, 2006), (Khomh & Guéhéneuc, 2008), (Sagheer et al., 2015), (Ran et al., 2009), (Christensen et al., 2010)
3.	Proceso de desarrollo de software y características de calidad	(B. Singh & Gautam, 2016a) (Arvanitou et al., 2017)
4.	Relación de prácticas ágiles y características de calidad	(Rico, 2008), (Bhasin, 2012)

Las investigaciones ubicadas en el ámbito 1 se ha considerado a la calidad de una manera general, sin considerar las características y/o subcaracterísticas de calidad ; en estos estudios no se determina cómo cada práctica ágil influye en los atributos de calidad.

Una gran cantidad de estudios correspondientes al ámbito 2, han contribuido a identificar características de calidad que influyen en el desarrollo ágil; sin embargo, no se han encontrado investigaciones que analicen la influencia de la aplicación de las prácticas ágiles, en las características identificadas.

En el ámbito 3, (B. Singh & Gautam, 2016a), realizaron un estudio del impacto del proceso de desarrollo de software en la calidad, este estudio divide este proceso en 4 etapas: Requerimentación, diseño, codificación y pruebas y la relaciona con varias características y subcaracterísticas de calidad, basado en una revisión de literatura; sin embargo, estas características no se rigen a un estándar específico. El estudio de (Arvanitou et al., 2017), similar al anterior, considera siete etapas de desarrollo de software: Gestión del proyecto, Requerimentación, Diseño, Arquitectura, Testing, Implementación y Mantenimiento y su relación con atributos de calidad y sus respectivas métricas.

De acuerdo al ámbito 4 (Rico, 2008), afirma que los métodos ágiles están caracterizados factores de desarrollo iterativo, que influye positivamente en calidad del diseño de los sitios web y retroalimentación del cliente, así como también en la privacidad y seguridad. Por otra parte, (Bhasin, 2012) considera en su estudio los aspectos de los métodos ágiles en la mejoran la calidad a través de reducción de defectos, detección temprana de defectos, mejora del tiempo y codificación de calidad, este estudio considera algunas prácticas ágiles como variables de control; sin embargo, no se basa en las características de calidad de software propuesto por el estándar ISO/IEC 25010.

Es importante conocer cuáles son las principales prácticas ágiles que contribuyen con el logro de las características de calidad del software, con el fin de que los administradores de proyectos prioricen en su equipo de trabajo la aplicación de estas prácticas y así desarrollar un producto de calidad, acorde con estándares establecidos.

6.3 Metodología

La metodología aplicada para determinar la relación entre las prácticas ágiles y las

características de calidad del software, es la misma aplicada en el capítulo anterior que ha sido descrita en el ítem 5.4.

Se realizó la prueba de la hipótesis para determinar las relaciones que fueron aceptadas y rechazadas, acerca de: a) prácticas ágiles y las características de calidad, y b) los factores críticos de éxito y las características de calidad de software

El Modelo Conceptual que guía el desarrollo de este capítulo, se encuentra representado en la Figura 6-1.

6.4 Resultados de la influencia de las prácticas ágiles sobre las características de calidad del software

Los resultados presentados a continuación, corresponden a la influencia entre las cuatro categorías de prácticas ágiles y las ocho características de calidad que forman parte del estándar ISO/IEC 25010 (Ver Figura 6-1). Basado en las encuestas, se obtuvieron los porcentajes resumidos, los mismos que se adjuntan en el Anexo L.

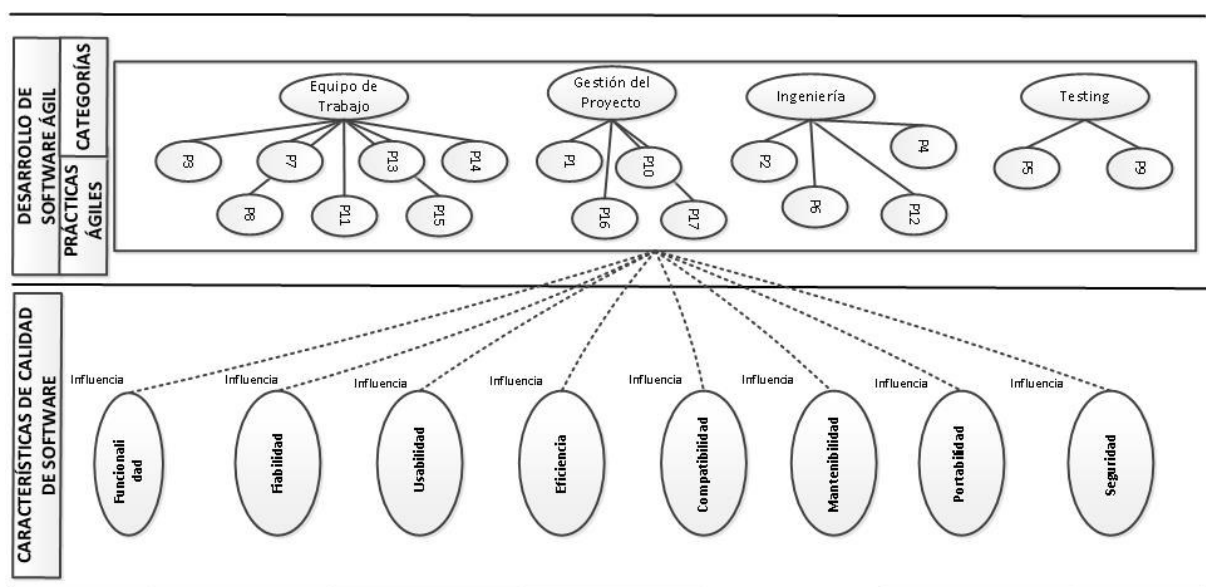


Figura 6-1 Modelo conceptual para determinar el impacto de la aplicación de prácticas ágiles sobre las características de calidad del software

6.4.1 Estadística descriptiva

La Tabla 6-2 presenta un resumen de estadísticas de las variables estudiadas. El valor de la media de 3,844, con un rango de 0,726 entre los valores de 3,418 y 4,144 de un total de 32 ítems considerados (usando una tabla de contingencia de 4 x 8). La varianza media de los ítems es 1,177 y un rango comprendido entre un valor máximo de 1,555 y un valor mínimo de 0,850.

Tabla 6-2 Resumen de Estadísticas

	Media	Mínimo	Máximo	Rango	Máximo/ Mínimo	Varianza	Número de ítems
Media de los ítems	3,844	3,418	4,144	0,726	1,12	0,034	32
Varianza de los ítems	1,177	0,850	1,555	0,705	1,829	0,031	32

Agile Practices	Characteristics of quality							
	AFUN	ARELI	AUSAB	APEF	ACOMP	AMANT	APORT	ASEG
PTW	4,14 ±0,983	3,86 ±0,996	3,84 ±1,133	3,9 ±1,106	3,77 ±1,161	3,91 ±1,043	3,6 ±1,142	3,87 ±1,097
PMP	3,88 ±0,975	3,7 ±1,147	3,55 ±1,187	3,69 ±1,067	3,46 ±1,238	3,68 ±1,132	3,42 ±1,247	3,6 ±1,178
PENG	4,09 ±0,996	4,02 ±0,972	4,07 ±0,922	4 ±1,05	3,92 ±1,064	3,97 ±1,053	3,93 ±1,048	3,98 ±1,02
PTEST	4,07 ±1,001	4 ±1,044	3,88 ±1,099	4,01 ±1,089	3,77 ±1,1	3,88 ±1,073	3,7 ±1,217	3,87 ±1,039

Figura 6-2 Mapa de calor de la media y desviación estándar de la relación entre prácticas ágiles y características de calidad

En la Figura 6-2, se muestran las estadísticas correspondientes a la media y desviación estándar mediante un mapa de calor, de las 32 variables que relacionan las prácticas ágiles (columnas) con las características de calidad (filas), allí se evidencia que la influencia PTW-AFUNC tiene la media más alta 4,14, con una desviación estándar de 0.983 (color verde oscuro), lo que implica que la mayor parte de los participantes de esta encuesta está de acuerdo en que las prácticas de *Equipo de trabajo* influye positivamente en la característica de *Funcionalidad*. Otro aspecto importante corresponde a la influencia PMP-APORT que tiene la menor media correspondiente a un valor de 3,42 y una desviación estándar de 1,247 (color rojo oscuro), este

resultado evidencia la práctica ágil de *Gestión del proyecto* sobre la característica de *Compatibilidad* tiene una débil influencia.

6.4.2 Fiabilidad y test de validez

En la Tabla 6-3, se puede ver que el valor de Cronbach's Alpha para las 32 variables analizadas fue de 0,975; por lo que se considera que la consistencia interna de los datos es altamente confiable.

Tabla 6-3 Estadísticas de fiabilidad para las 32 variables analizadas

Cronbach's Alpha	Cronbach's Alpha basado en ítems estandarizados	N. of ítems
0,975	0,976	32

6.4.3 Análisis de residuos ajustados

La Figura 6-3, muestra un mapa de calor en donde, el color verde oscuro corresponde al índice de influencia más alto determinados por los encuestados (alta influencia); el color rojo oscuro corresponde al menor índice de influencia determinado por los encuestados (baja influencia). La significancia de cada relación está determinada por la escala Stainines (1-9) que se presentó en el capítulo anterior.

Los resultados evidencian que la significancia de la influencia entre *la Prácticas de Equipo de Trabajo* (PTW) y la característica de calidad *funcionalidad* (AFUN) es alta con un ajuste residual de 1,63, que corresponde al valor de 8 en la escala Stanines; otras prácticas ágiles que influyen positivamente en la *funcionalidad* son las de *Ingeniería* (PENG) y *testing* (PTEST). Además, la característica de *Funcionalidad* tiene una influencia positiva del 75% de las categorías de prácticas ágiles analizadas, este resultado indica que ésta es la característica que más afectada por la aplicación de prácticas ágiles.

Otro resultado importante indica que las prácticas de gestión del proyecto afectan negativamente en el 87,5% de las características analizadas, siendo su influencia negativa mayor

en el atributo de *portabilidad*, seguido por las características de *compatibilidad*, *usabilidad* y *seguridad*, es decir que la gestión del proyecto tiene una influencia baja en estas características de calidad.

Agile Practices	Characteristics of quality							
	AFUN	ARELI	AUSAB	APEF	ACOMP	AMANT	APORT	ASEG
PTW	1,63 (8)	0,064 (5)	-0,048 (5)	0,287 (6)	-0,381 (4)	0,361 (6)	-1,343 (2)	0,138 (5)
PMP	0,174 (5)	-0,783 (3)	-1,591 (2)	-0,82 (3)	-2,065 (1)	-0,893 (3)	-2,282 (1)	-1,333 (2)
PENG	1,34 (8)	0,962 (7)	1,222 (7)	0,85 (7)	0,437 (6)	0,662 (6)	0,474 (6)	0,737 (6)
PTEST	1,222 (7)	0,846 (7)	0,213 (5)	0,921 (7)	-0,382 (4)	0,213 (5)	-0,789 (3)	0,138 (5)

Figura 6-3 Mapa de calor de los residuos ajustados entre las prácticas ágiles y las características de calidad.

6.5 Prueba de hipótesis

6.5.1 Hipótesis de la influencia de prácticas ágiles y características de calidad del software

En la Tabla 6-4 resume los resultados de las hipótesis planteadas sobre influencia de prácticas ágiles y las características de calidad de software. De las 13 hipótesis propuestas, cuatro son hipótesis nulas (H2.1, H3.5, H3.6 and H3.8), porque el valor d' está entre -0,75 y 0,75 y estos valores no son suficientes para rechazar la hipótesis nula, lo que implica que existe una influencia media. Se ha demostrado que las nueve hipótesis restantes son verdaderas, es decir que las hipótesis nulas han sido rechazadas debido a que la influencia de las prácticas ágiles en las características de calidad han sido lo suficientemente alta o baja que han permitido rechazarlas. Las hipótesis H1.1, H3.1, H4.1, H3.2, H4.2, H3.3, H3.4 y H4.4 ejercen una influencia superior a la media. De acuerdo a los resultados obtenidos se puede afirmar que la influencia de las prácticas ágiles de *testing* sobre la característica de calidad de *Portabilidad* (H4.7) es la única influencia significativamente baja que se ha encontrado. La hipótesis restante ejerce una influencia por debajo del promedio o nula entre la práctica y la característica de calidad (Ver Figura 6-5).

Tabla 6-4 Resultados de las pruebas de hipótesis

Prácticas ágiles	Características de Calidad							
	Funcionalidad	Fiabilidad	Usabilidad	Eficiencia	Compatibilidad	Mantenibilidad	Portabilidad	Seguridad
PTW	<u>H1.1+</u>							
PMP	H2.1							
PENG	<u>H3.1+</u>	<u>H3.2+</u>	<u>H3.3+</u>	<u>H3.4+</u>	H3.5	H3.6		H3.8
PTEST	<u>H4.1+</u>	<u>H4.2+</u>		<u>H4.4+</u>			<u>H4.7-</u>	

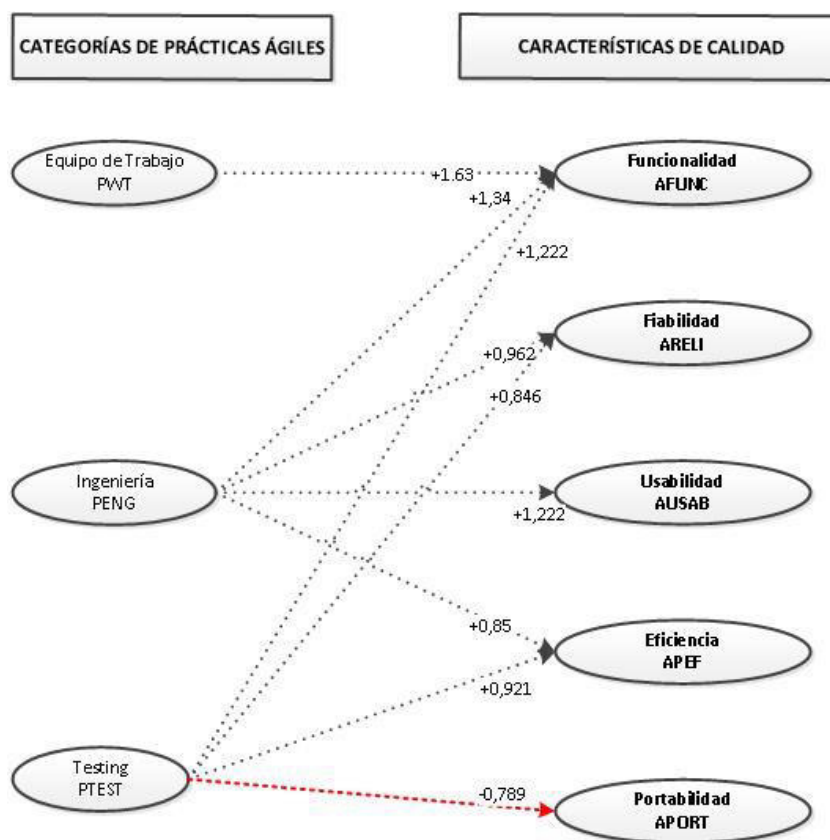


Figura 6-4 Resultados de la Influencia de categorías de prácticas ágiles en las características de calidad, con base en el estándar ISO/IEC 25010

De acuerdo a los resultados, las categorías de prácticas ágiles y características de calidad que tienen una influencia significativa, son presentados en la Figura 6-4. Como se puede observar, se ha eliminado la categoría de *Prácticas de gestión de proyectos*, la misma que no ejerce ninguna influencia significativa sobre ninguna de las prácticas ágiles. También, se puede

determinar que la aplicación de las categorías de prácticas ágiles analizadas, no son influyen significativamente en las características de calidad de *Mantenibilidad*, *Seguridad* y *Compatibilidad*.

6.5.2 Influencia de las prácticas ágiles en la calidad del software

Las prácticas de *Equipo de Trabajo*, ejercen una influencia significativamente positiva en la característica de calidad de *Funcionalidad*, este resultado se justifica por la aplicación de prácticas como *Cliente On-site*, *reuniones diarias*, *retrospectivas* y *propiedad colectiva del código*, que contribuyen a la obtención de una funcionalidad completa y correcta del software. Además, se ha observado que estas prácticas tienen una influencia significativamente baja en la característica de *Portabilidad*, es decir que la adaptabilidad, inestabilidad and reemplazabilidad del software no dependen de las prácticas de trabajo en equipo. Además, se puede evidenciar que las prácticas de *Equipo de Trabajo* no influyen significativamente en las demás características de calidad.

Los resultados obtenidos en este trabajo, evidencian que 3 grupos de prácticas ágiles: *Prácticas de Ingeniería*, *Equipo de Trabajo* y *Testing* tienen una influencia significativa positiva con las con el atributo de calidad *Funcionalidad*. Además, la aplicación de las *Prácticas de Gestión del Proyecto* tienen una influencia baja sobre todas las características de calidad, excepto la característica de *Funcionalidad*, en las que ejerce una influencia media. A pesar de que las otras prácticas analizadas también tienen un nivel de influencia sea esta alta, media o baja sobre las características de calidad, estos valores no han sido considerados debido a que los resultados no son lo suficientemente significativos; por lo que la calidad del software no va a ser influenciada por las prácticas ágiles, aunque éstas sean aplicadas en un proyecto de desarrollo de software.

También se evidencia una influencia positiva alta de las prácticas de *testing*, sobre la *funcionalidad*, *fiabilidad* y *eficiencia*, y una influencia media sobre la *usabilidad*, *mantenibilidad* y *seguridad*. En el primer caso, las pruebas, en su mayor parte, están asociadas con el diseño y la programación, mientras que en el segundo caso, se relacionan con la operación

del sistema, por esta razón los resultados obtenidos son los esperados. Se evidencia además, que las prácticas de *testing* tienen una baja influencia sobre la portabilidad.

De acuerdo con las hipótesis planteadas, existen tres características de calidad que no tienen influencia significativa con las prácticas ágiles, estas son: *compatibilidad*, *mantenibilidad* y *seguridad*; sin embargo, tienen una influencia positiva media con las *prácticas de ingeniería*.

En la revisión de literatura realizada por (B. Singh & Gautam, 2016a), se determina que todos los atributos de calidad son importantes, pero la capacidad de mantenimiento es el atributo más importante para los productos de software porque el mantenimiento del software es el proceso de refinar el software para garantizar que continúe satisfaciendo las necesidades del negocio; los autores también consideran que la característica de *Mantenibilidad* del software, contribuye a la mejora del rendimiento del sistema y otras características de calidad.

Basado en este análisis, cada fase del proceso de desarrollo de software es importante en el aseguramiento de la calidad, sin embargo, el diseño del software tiene una importancia muy relevante. Las *Prácticas de Ingeniería* incorporan varios aspectos relacionados con la etapa de diseño, se pueden destacar práctica como: *diseño simple*, *pequeñas versiones*, *estándares de codificación* y *refactoring*, que tienen alta influencia sobre las características de *Eficiencia*, *Funcionalidad*, *Fiabilidad* y *Usabilidad*. Además estas prácticas tienen una influencia media sobre las características de *Seguridad*, *Compatibilidad* y *Mantenibilidad*. Estos resultados coinciden con la afirmación realizada por (B. Singh & Gautam, 2016b).

6.6 Relación entre los factores críticos de éxito y características de calidad del software

Para determinar influencia entre los factores críticos de éxito y las características de calidad, se utilizó la matriz A (influencia de factores sobre las prácticas de calidad) y la matriz B (Influencia de las prácticas sobre las características de calidad), para crear una nueva matriz C. Cada celda de la matriz es calculada de la siguiente manera:

$$c_{ft} = \sqrt{\frac{\sum_{p=1}^n a_{fp} * b_{pt}}{n}}$$

Donde:

c_{ft} es la celda de la nueva matriz C la misma que relaciona el factor f con el atributo t .

a_{fp} es la celda de la matriz A que relaciona el factor f con la práctica p

b_{pt} es la celda de la matriz B que relaciona la práctica p con la característica t

n Número de prácticas analizadas.

La Figura 6-5 muestra un mapa de calor de los residuos ajustados, en este caso el cálculo no se realiza para cada encuestado. Por lo que, e_{ij} es la media de la matriz C , $\bar{c}$.

Los resultados evidencian que la significancia de la influencia entre el factor *Medio de comunicación utilizado* y el atributo de calidad *Funcionalidad* es alta (2,07), además, este factor influye significativamente en las características de *Eficiencia* y *Fiabilidad*. Así mismo, la funcionalidad del software se ve afectada positivamente por los factores *de Integración* y *Disponibilidad de la información y experiencias adquiridas y autoeficacia percibida*.

Otro resultado importante que se evidencia en el estudio, indica que el factor de estrés influye negativamente en 6 de las 8 características analizadas, las características afectadas por este factor son: *compatibilidad, fiabilidad, mantenimiento, portabilidad, seguridad y usabilidad*.

Características de Calidad	Factores Críticos de éxito					
	FSE	FCO	FQL	FIAI	FSTR	FCM
ACOMP	-0.39 (4)	-0.69 (4)	-0.87 (3)	0.15 (5)	-2.07 (1)	0.40 (6)
APEF	0.48 (6)	0.18 (5)	-0.01 (5)	1.04 (7)	-1.23 (2)	1.30 (8)
ARELI	0.45 (6)	0.14 (5)	-0.04 (5)	1.00 (7)	-1.27 (2)	1.26 (7)
AFUNC	1.24 (8)	0.93 (7)	0.74 (7)	1.79 (9)	-0.51 (4)	2.07 (9)
AMAIN	0.28 (6)	-0.02 (5)	-0.21 (5)	0.83 (7)	-1.43 (2)	1.09 (7)
APORT	-0.79 (3)	-1.09 (3)	-1.26 (2)	-0.24 (5)	-2.45 (1)	0.00 (5)
ASEC	0.11	-0.19	-0.37	0.66	-1.59	0.92

Características de Calidad	Factores Críticos de éxito					
	FSE	FCO	FQL	FIAI	FSTR	FCM
	(5)	(5)	(4)	(6)	(2)	(7)
AUSAB	0.14	-0.17	-0.35	0.69	-1.57	0.94
	(5)	(5)	(4)	(6)	(2)	(7)

Figura 6-5 Mapa de calor de residuos ajustados entre factores y características de calidad

6.7 Conclusiones del capítulo

En este capítulo, se ha analizado la influencia de cuatro categorías de prácticas ágiles (*Equipo de Trabajo, Gestión del Proyecto, Ingeniería y Testing*) con ocho características de calidad, correspondientes al estándar ISO/IEC 25010 (*Funcionalidad, Fiabilidad, Usabilidad, Eficiencia, Compatibilidad, Mantenibilidad, Portabilidad y Seguridad*).

Nueve de las trece hipótesis planteadas reflejan una influencia significativa de las prácticas ágiles sobre las características de calidad; éstas son las siguientes: *prácticas de Equipo de Trabajo* sobre la característica de calidad *funcionalidad*; *prácticas de Ingeniería* sobre la característica *de funcionalidad, fiabilidad, usabilidad eficiencia*; *Prácticas de testing* sobre las características *de funcionalidad, fiabilidad y eficiencia*.

Los resultados de la también evidencia una influencia positiva alta entre factor *Medio de comunicación utilizado* y el atributo de calidad *Funcionalidad*, además, este factor influye significativamente en las características de *Eficiencia y Fiabilidad*. Así mismo, la funcionalidad del software se ve afectada positivamente por los factores *de Integración y Disponibilidad de la información y experiencias adquiridas y autoeficacia percibida*.

CAPÍTULO 7: CONCLUSIONES Y TRABAJOS FUTUROS

7.1 Conclusiones

7.1.1 Conclusión General

Este trabajo ha contribuido con el aseguramiento de la calidad de software a través la propuesta de un modelo compuesto por: a) Seis factores críticos de éxito: *Autoeficacia percibida, Estrés en el trabajo, Experiencias aprendidas, Integridad y disponibilidad de la información, Calidad de vida laboral, y Medios de Comunicación usados.* b) Cuatro categorías de prácticas ágiles: *Equipo de trabajo, Gestión del Proyecto, Ingeniería y Testing.* c) Ocho características del estándar ISO/IEC 25010: *Funcionalidad, Fiabilidad, Usabilidad, Eficiencia, Portabilidad, Compatibilidad, Mantenibilidad y Seguridad.*

7.1.2 Conclusiones específicas

CE1 La revisión sistemática de literatura realizada en este estudio ha evidenciado la existencia de 148 factores críticos de éxito, clasificados en 7 categorías; 137 métricas divididas en 8 categorías y 165 prácticas ágiles categorizadas en 5 grupos

CE2 En este trabajo de han identificado 16 modelos de calidad aplicados a modelos de desarrollo ágil, estos modelos están basados en estándares ya establecidos, como CMMI, se adaptan a una metodología en particular como Scrum o XP, o evolucionan un modelo tradicional como Cascada a un modelo ágil. Con base en el análisis realizado, no existe un modelo que abarque todos los aspectos de calidad considerados en el presente estudio, el aspecto más ha sido incluido en los modelos son: métricas/atributos de calidad con un 62,50% de cumplimiento. La mayoría de los modelos podrían complementarse relacionándolos con todos los principios de calidad del Manifiesto Ágil y con factores críticos de éxito ya que solamente un modelo contempla factores críticos de éxito, pero no

está validado.

- CE3** Los resultados obtenidos en el primer aporte secundario identificaron que los factores de *Medios de comunicación usados (FCM)* , *Confianza (FCO)*, *Autoeficacia percibida (FSE)*, *Integridad y disponibilidad de la información y experiencias aprendidas (FAI)* tienen una influencia significativa positiva sobre las prácticas ágiles. Por otra parte, el *Estrés Stress (FSTR)* tiene una influencia negativa en casi todas las prácticas ágiles y el factor de *Calidad de vida laboral (FQWL)* afecta negativamente a las prácticas de *Testing*.
- CE4** En el segundo aporte secundario, se ha analizó, la influencia de cuatro categorías de prácticas ágiles (*Equipo de Trabajo, Gestión del proyecto, Ingeniería y Testing*) con ocho características de calidad, basadas en el estándar ISO/IEC 25010 (*Funcionalidad, Fiabilidad, Usabilidad, Eficiencia, Compatibilidad, Mantenibilidad, Portabilidad y Seguridad*). Se evidencia que 9 de las 13 hipótesis propuestas reflejan una influencia significativa de las prácticas ágiles sobre las características de calidad. Los resultados obtenidos identificaron que las siguiente prácticas ágiles tienen alta influencia en las características de calidad del software: *prácticas de trabajo en equipo* sobre la característica de calidad *Funcionalidad*; *prácticas de ingeniería* sobre la característica de *Funcionalidad, fiabilidad, usabilidad y eficiencia*; *Prácticas de Testing* sobre las características de *Funcionalidad, Fiabilidad y Eficiencia*. Además, se determinó una influencia baja de las prácticas ágiles de *Testing*, sobre la característica de *Portabilidad*.
- CE5** En el tercer aporte secundario, se evidencia una influencia positiva alta entre factor *Medio de comunicación utilizado* y el atributo de calidad *Funcionalidad*, además, este factor influye significativamente en las características de *Eficiencia y Fiabilidad*. Así mismo, la funcionalidad del software se ve afectada positivamente por los factores de *Integración y Disponibilidad de la información y experiencias adquiridas y autoeficacia percibida*.

7.2 Limitaciones

Una de las técnicas utilizadas en este estudio para la obtención de los resultados es la encuesta,

por lo tanto, está sujeto a ciertas limitaciones como las siguientes:

- El tipo y tamaño de los proyectos no fueron considerados, aspectos que pueden modificar los resultados de este estudio ya el nivel de aplicación de las prácticas ágiles puede ser diferente en dependencia al tipo y tamaño del proyecto.
- El 16,44% de los encuestados utilizan metodologías ágiles diferentes a Scrum y XP, razón por la cual, los resultados pueden tener un sesgo hacia las prácticas utilizadas por estas metodologías.
- En Ecuador, el 22% de las empresas de desarrollo de software tienen más de 50 empleados, estos datos han sido obtenidos del estudio realizado por la Asociación Ecuatoriana de Software, hasta el año 2015. El 14,4% de los encuestados pertenecen a este grupo de empresas. Además, no se puede evidenciar si esta muestra considerada cubre a todas las empresas existentes en el país; pudiendo existir una brecha en los resultados debido a la exclusión de algunas empresas.
- Las prácticas específicas que corresponden a cada categoría de prácticas no fueron analizadas, sin embargo, en el cuestionario se indicó cuáles son las prácticas consideradas en cada categoría; este aspecto pudo afectar a los resultados en caso de que este aspecto no haya sido considerado correctamente por los encuestados.

7.3 Trabajos futuros

Hay algunos aspectos que no se han considerado en la mayoría de los modelos y que podrían ser incorporados en futuros estudios, como por ejemplo, los factores humanos que pueden afectar a la calidad del software, muchos de ellos derivados de un mal liderazgo del jefe del proyecto o a su vez de los directivos del negocio para quienes se desarrolla el producto de software; éstos son factores que pueden complementarse a los atributos de los modelos existentes actualmente.

La agilidad en el aseguramiento de la calidad es un tema que debe profundizarse, ya que la aplicación de SQA normalmente requiere de muchos artefactos y procesos que no están en concordancia con los principios ágiles, o a su vez, se basan en estándares de calidad que se caracterizan por su excesiva documentación.

La mitigación de los factores que influyen negativamente en las prácticas ágiles y el fortalecimiento de los factores positivos en proyectos de desarrollo de software, es otro de los trabajos futuros ya que se podrían implementar estrategias que contribuyan a ratificar o refutar los resultados obtenidos en este estudio. Además, sería importante analizar el modelo propuesto en diferentes condiciones, como universidades, empresas privadas, empresas públicas o en entornos de desarrollo de software global, para saber si la influencia de diferentes factores en las prácticas ágiles varía según el entorno. Del mismo modo, el modelo podría analizarse considerando los perfiles de los involucrados en el proceso de desarrollo de software para determinar su punto de vista sobre la influencia de los factores en las prácticas ágiles.

La información obtenida en este trabajo puede servir como base para futuros estudios que consideren prácticas ágiles más específicas y su influencia en la calidad del software, ya que los resultados de este estudio corresponden a una categorización de prácticas ágiles y no a un estudio individual de cada una de ellas.

El tamaño de la muestra seleccionada es muy razonable para obtener una conclusión suficiente para la industria del software de Ecuador, por lo tanto, no se pretende generalizar los hallazgos de este estudio a otros países y regiones. Por esta razón se ha incluido la información demográfica de los participantes, con el fin de que este estudio, pueda ser probado en otros contextos.

REFERENCIAS

- Abdalhamid, S., Mohammed, A. O. M., & Mishra, A. (2019). Agile and Quality: A Systematic Mapping Study. *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)*, 1–7. <https://doi.org/10.1109/ICCSRE.2019.8807763>
- Abrantes, P. C., & Paula Furtado, A. (2021). Agile Development Practices Applied to Software Startups: A Systematic Mapping Review. *2021 16th Iberian Conference on Information Systems and Technologies (CISTI)*, 1–6. <https://doi.org/10.23919/CISTI52073.2021.9476368>
- Acuña, S. T., Gómez, M., & Juristo, N. (2008). Towards understanding the relationship between team climate and software quality—A quasi-experimental study. *Empir Software Eng*, 13(4), 401–434. <https://doi.org/10.1007/s10664-008-9074-8>
- Acuña, S. T., Gómez, M., & Juristo, N. (2009). How do personality, team processes and task characteristics relate to job satisfaction and software quality? *Information and Software Technology*, 51(3), 627–639. <https://doi.org/10.1016/j.infsof.2008.08.006>
- Acuña, S. T., Gómez, M. N., Hannay, J. E., Juristo, N., & Pfahl, D. (2015). Are team personality and climate related to satisfaction and software quality? Aggregating results from a twice replicated experiment. *Information and Software Technology*, 57, 141–156. <https://doi.org/10.1016/j.infsof.2014.09.002>
- Adams, J. S., & Freedman, S. (1976). Equity Theory Revisited: Comments and Annotated Bibliography. In L. B. and E. Walster (Ed.), *Advances in Experimental Social Psychology* (Vol. 9, pp. 43–90). Academic Press. <http://www.sciencedirect.com/science/article/pii/S0065260108600581>
- Agarwal, N., & Rathod, U. (2006). Defining ‘success’ for software projects: An exploratory revelation. *International Journal of Project Management*, 24(4), 358–370. <https://doi.org/10.1016/j.ijproman.2005.11.009>
- Agile Alliance. (2001). *Agile Alliance: The Twelve Principles of Agile Software*. <http://www.agilealliance.org/the-alliance/the-agile-manifesto/the-twelve-principles-of-agile-software/>
- Ahmed, A., Ahmad, S., Ehsan, N., Mirza, E., & Sarwar, S. Z. (2010). Agile software

- development: Impact on productivity and quality. *2010 IEEE International Conference on Management of Innovation and Technology (ICMIT)*, 287–291. <https://doi.org/10.1109/ICMIT.2010.5492703>
- Altuwaijri, F. S., & Ferrario, M. A. (2022). Factors affecting Agile adoption: An industry research study of the mobile app sector in Saudi Arabia. *Journal of Systems and Software*, 190, 111347. <https://doi.org/10.1016/j.jss.2022.111347>
- Ambu, W., Concas, G., Marchesi, M., & Pinna, S. (2006). Studying the Evolution of Quality Metrics in an Agile/Distributed Project. In P. Abrahamsson, M. Marchesi, & G. Succi (Eds.), *Extreme Programming and Agile Processes in Software Engineering* (pp. 85–93). Springer Berlin Heidelberg. http://link.springer.com.ezproxy.unal.edu.co/chapter/10.1007/11774129_9
- Anastas, J. W. (2000). *Research Design for Social Work and the Human Services: Second edition* (p. 608 Pages). Columbia University Press.
- Arnyndiasari, D., Ferdiana, R., & Santosa, P. I. (2022). Software Practices For Agile Developers: A Systematic Literature Review. *2022 1st International Conference on Information System & Information Technology (ICISIT)*, 238–243. <https://doi.org/10.1109/ICISIT54091.2022.9872874>
- Arvanitou, E. M., Ampatzoglou, A., Chatzigeorgiou, A., Galster, M., & Avgeriou, P. (2017). A mapping study on design-time quality attributes and metrics. *Journal of Systems and Software*, 127, 52–77. <https://doi.org/10.1016/j.jss.2017.01.026>
- Ayed, H., Habra, N., & Vanderose, B. (2013). AM-QuICK: A Measurement-Based Framework for Agile Methods Customisation. *2013 Joint Conference of the 23rd International Workshop on Software Measurement and the 2013 Eighth International Conference on Software Process and Product Measurement (IWSM-MENSURA)*, 71–80. <https://doi.org/10.1109/IWSM-Mensura.2013.21>
- Ayed, H., Vanderose, B., & Habra, N. (2012). A Metamodel-Based Approach for Customizing and Assessing Agile Methods. *Quality of Information and Communications Technology (QUATIC)*, 2012 Eighth International Conference on The, 66–74. <https://doi.org/10.1109/QUATIC.2012.11>
- Bandura, A. (1977). Self-efficacy: Toward a Unifying Theory of Behavioral Change. *Psychological Review*, 84(2), 191–215.

- Bansal, A., & Pundir, S. (2016). Software quality score board based on SQA framework to improvise software reliability. *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, 2421–2427.
- Bansiya, J., & Davis, C. G. (2002). A hierarchical model for object-oriented design quality assessment. *IEEE Transactions on Software Engineering*, 28(1), 4–17. <https://doi.org/10.1109/32.979986>
- Bastarrica, M. C., Espinoza, G., & Marín, J. (2018). Implementing agile practices: The experience of TSol. *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, 1–10. <https://doi.org/10.1145/3239235.3268918>
- Bawane M., P. I. of, & C.V. Srikrishna, F. (2010). *A Novel Method for Quantitative Assessment of Software Quality*. <http://cscjournals.org/csc/manuscript/Journals/IJCSS/volume3/Issue6/IJCSS-196.pdf>
- Beckhaus, A., Karg, L. M., & Hanselmann, G. (2009). Applicability of Software Reliability Growth Modeling in the Quality Assurance Phase of a Large Business Software Vendor. *Computer Software and Applications Conference, 2009. COMPSAC '09. 33rd Annual IEEE International*, 1, 209–215. <https://doi.org/10.1109/COMPSAC.2009.196>
- Begier, B. (2010). Users' involvement may help respect social and ethical values and improve software quality. *Inf Syst Front*, 12(4), 389–397. <https://doi.org/10.1007/s10796-009-9202-z>
- Behutiye, W., Seppänen, P., Rodríguez, P., & Oivo, M. (2020). Documentation of Quality Requirements in Agile Software Development. *Proceedings of the Evaluation and Assessment in Software Engineering*, 250–259. <https://doi.org/10.1145/3383219.3383245>
- Bellomo, S., Nord, R. L., & Ozkaya, I. (2013). Elaboration on an integrated architecture and requirement practice: Prototyping with quality attribute focus. *Twin Peaks of Requirements and Architecture (TwinPeaks)*, 2013 2nd International Workshop on The, 8–13. <https://doi.org/10.1109/TwinPeaks.2013.6614717>
- Bermejo, P. H. de S., Zambalde, A. L., Tonelli, A. O., Souza, S. A., Zuppo, L. A., & Rosa, P. L. (2014). Agile Principles and Achievement of Success in Software Development: A Quantitative Study in Brazilian Organizations. *Procedia Technology*, 16, 718–727.

<https://doi.org/10.1016/j.protcy.2014.10.021>

- Bhasin, S. (2012). Quality Assurance in Agile: A Study towards Achieving Excellence. *AGILE India (AGILE INDIA)*, 2012, 64–67. <https://doi.org/10.1109/AgileIndia.2012.18>
- Boerman, M. P., Lubsen, Z., Tamburri, D. A., & Visser, J. (2015). Measuring and Monitoring Agile Development Status. *2015 IEEE/ACM 6th International Workshop on Emerging Trends in Software Metrics*, 54–62. <https://doi.org/10.1109/WETSoM.2015.15>
- Boghossian, Z. J. (2002). *An Investigation into the Critical Success Factors of Software Development Process, Time, and Quality*. Pepperdine University.
- Bostrom, R. P., & Heinen, J. S. (1977). MIS Problems and Failures: A Socio-Technical Perspective, Part II: The Application of Socio-Technical Theory. *MIS Quarterly*, 1(4), 11–28. <https://doi.org/10.2307/249019>
- Bougroun, Z., Zeaaraoui, A., & Bouchentouf, T. (2014). The projection of the specific practices of the third level of CMMI model in agile methods: Scrum, XP and Kanban. *Information Science and Technology (CIST)*, 2014 Third IEEE International Colloquium In, 174–179. <https://doi.org/10.1109/CIST.2014.7016614>
- Brito e Abreu, F., & Melo, W. (1996). Evaluating the impact of object-oriented design on software quality. *Software Metrics Symposium, 1996., Proceedings of the 3rd International*, 90–99. <https://doi.org/10.1109/METRIC.1996.492446>
- Canedo, E. D., & Santos, G. A. (2019). Factors Affecting Software Development Productivity: An empirical study. *Proceedings of the XXXIII Brazilian Symposium on Software Engineering*, 307–316. <https://doi.org/10.1145/3350768.3352491>
- Canós, J. H., Letelier, P., & Penadés, Ma. C. (2003). Metodologías Ágiles de desarrollo de Software. *Metodologías Ágiles En El Desarrollo de Software*. <http://issi.dsic.upv.es/archives/f-1069167248521/actas.pdf>
- Caputo, K. (1998). *Cmm Implementation Guide: Choreographing Software Process Improvement*. Addison-Wesley,.
- Carroll, J. S. (1982). Review of Foundations of Information Integration Theory [Review of *Review of Foundations of Information Integration Theory*, by N. H. Anderson]. *The American Journal of Psychology*, 95(4), 708–711. <https://doi.org/10.2307/1422202>
- Chakravarty, K., & Singh, J. (2021). A Study of Quality Metrics in Agile Software Development. In D. Swain, P. K. Pattnaik, & T. Athawale (Eds.), *Machine Learning and*

- Information Processing* (pp. 255–266). Springer. https://doi.org/10.1007/978-981-33-4859-2_26
- Cheriyān, A., Gondkar, R. R., & Babu, S. S. (2020). Quality Assurance Practices and Techniques Used by QA Professional in Continuous Delivery. In M. Tuba, S. Akashe, & A. Joshi (Eds.), *Information and Communication Technology for Sustainable Development* (pp. 83–92). Springer. https://doi.org/10.1007/978-981-13-7166-0_8
- Chidamber, S. R., & Kemerer, C. F. (1994). A metrics suite for object oriented design. *IEEE Transactions on Software Engineering*, 20(6), 476–493. <https://doi.org/10.1109/32.295895>
- Chow, T., & Cao, D.-B. (2008). A survey study of critical success factors in agile software projects. *Journal of Systems and Software*, 81(6), 961–971. <https://doi.org/10.1016/j.jss.2007.08.020>
- Christensen, H. B., Hansen, K. M., & Lindstrøm, B. (2010). Lightweight and Continuous Architectural Software Quality Assurance Using the aSQA Technique. In M. A. Babar & I. Gorton (Eds.), *Software Architecture* (pp. 118–132). Springer Berlin Heidelberg. http://link.springer.com.ezproxy.unal.edu.co/chapter/10.1007/978-3-642-15114-9_11
- Clarke, P., & O'Connor, R. V. (2012). The situational factors that affect the software development process: Towards a comprehensive reference framework. *Information and Software Technology*, 54(5), 433–447. <https://doi.org/10.1016/j.infsof.2011.12.003>
- Coleman, J. S. (2005). Social Capital in the Creation of Human Capital. 94, | socialcapitalgateway.org. *The American Journal of Sociology*, 94, S95–S120.
- Computing Research & Education. (2022). *CORE Rankings Portal*. <https://sites.google.com/site/wwwcoreedu>
- Curcio, K., Malucelli, A., Reinehr, S., & Paludo, M. A. (2016a). An analysis of the factors determining software product quality: A comparative study. *Computer Standards & Interfaces*, 48, 10–18. <https://doi.org/10.1016/j.csi.2016.04.002>
- Curcio, K., Malucelli, A., Reinehr, S., & Paludo, M. A. (2016b). An analysis of the factors determining software product quality: A comparative study. *Computer Standards & Interfaces*, 48, 10–18. <https://doi.org/10.1016/j.csi.2016.04.002>
- Curcio, K., Santana, R., Reinehr, S., & Malucelli, A. (2019). Usability in agile software development: A tertiary study. *Computer Standards & Interfaces*, 64, 61–77.

<https://doi.org/10.1016/j.csi.2018.12.003>

- Daft, R. L., & Lengel, R. H. (1984). Information richness: A new approach to managerial behavior and organizational design. *Research in Organizational Behavior*, 6, 191–233.
- Darwish, N. R. (2011). *Improving the Quality of Applying eXtreme Programming (XP) Approach*. <http://123seminarsonly.com/Seminar-Reports/024/82097086-Improving-the-Quality-of-Applying-eXtreme-Programming-XP-Approach.pdf>
- de Souza Mariz, L. M. R., França, A. C. C., & da Silva, F. Q. B. (2010). An Empirical Study on the Relationship between the Use of Agile Practices and the Success of Software Projects that Use Scrum. *2010 Brazilian Symposium on Software Engineering (SBES)*, 110–117. <https://doi.org/10.1109/SBES.2010.17>
- Desharnais, J.-M., Kocaturk, B., & Abran, A. (2011). Using the COSMIC Method to Evaluate the Quality of the Documentation of Agile User Stories. *Software Measurement, 2011 Joint Conference of the 21st Int'l Workshop on and 6th Int'l Conference on Software Process and Product Measurement (IWSM-MENSURA)*, 269–272. <https://doi.org/10.1109/IWSM-MENSURA.2011.45>
- Destefanis, G., Counsell, S., Concas, G., & Tonelli, R. (2014). Software Metrics in Agile Software: An Empirical Study. In G. Cantone & M. Marchesi (Eds.), *Agile Processes in Software Engineering and Extreme Programming* (pp. 157–170). Springer International Publishing. http://link.springer.com.ezproxy.unal.edu.co/chapter/10.1007/978-3-319-06862-6_11
- Diebold, P., & Dahlem, M. (2014). Agile Practices in Practice: A Mapping Study. *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, 30:1-30:10. <https://doi.org/10.1145/2601248.2601254>
- Digital.ai. (2021). *15th State of Agile Report*. Digital.ai.
- Dikert, K., Paasivaara, M., & Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*, 119, 87–108. <https://doi.org/10.1016/j.jss.2016.06.013>
- Estayno, M. G., Dapozo, G. N., Cuenca Pletsch, L. R., & Greiner, C. L. (2009). Modelos y métricas para evaluar calidad de software. *XI Workshop de Investigadores En Ciencias de La Computación*. <http://sedici.unlp.edu.ar/handle/10915/19762>
- Fernández-Sanz, L., & Misra, S. (2011). Influence of human factors in software quality and

- productivity. In *Computational Science and Its Applications-ICCSA 2011* (pp. 257–269). Springer. http://link.springer.com/chapter/10.1007/978-3-642-21934-4_22
- Ferreira, M., García, F., Ruiz, F., Bertoa, M. F., Calero, C., Vallecillo, A., Piattini, M., & Mora, B. (2006). Medición del Software Ontología y Metamodelo. *Departamento de Tecnologías y Sistemas de La Información, Castilla La Mancha*. <http://www.uclm.net/dep/tsi/pdf/UCLM-TSI-001.pdf>
- Fink, A. G. (2002). *The Survey Handbook 2nd edition*.
- Foerderer, J., Kude, T., Mithas, S., & Heinzl, A. (2016). How Temporal Work Styles and Product Modularity Influence Software Quality and Job Satisfaction. *Proceedings of the 2016 ACM SIGMIS Conference on Computers and People Research*, 105–112. <https://doi.org/10.1145/2890602.2890608>
- Forouzani, S., Chiam, Y. K., & Forouzani, S. (2016). Method for Assessing Software Quality Using Source Code Analysis. *Proceedings of the Fifth International Conference on Network, Communication and Computing*, 166–170. <https://doi.org/10.1145/3033288.3033316>
- Fuggetta, A., & Di Nitto, E. (2014). Software process. *Future of Software Engineering Proceedings*, 1–12. <https://doi.org/10.1145/2593882.2593883>
- García-Mireles, G. A., Moraga, M. Á., García, F., & Piattini, M. (2015). Approaches to promote product quality within software process improvement initiatives: A mapping study. *Journal of Systems and Software*, 103, 150–166. <https://doi.org/10.1016/j.jss.2015.01.057>
- Garousi, V., Coşkunçay, A., Betin-Can, A., & Demirörs, O. (2015). A survey of software engineering practices in Turkey. *Journal of Systems and Software*, 108, 148–177. <https://doi.org/10.1016/j.jss.2015.06.036>
- Gorla, N., & Lin, S.-C. (2010). Determinants of software quality: A survey of information systems project managers. *Information and Software Technology*, 52(6), 602–610. <https://doi.org/10.1016/j.infsof.2009.11.012>
- Grant, R. M. (1996). Toward a knowledge-based theory of the firm. *Strategic Management Journal*, 17(S2), 109–122. <https://doi.org/10.1002/smj.4250171110>
- Greening, D. R. (2015). Agile Enterprise Metrics. *2015 48th Hawaii International Conference on System Sciences (HICSS)*, 5038–5044. <https://doi.org/10.1109/HICSS.2015.597>

- Haberman, S. J. (1973). The Analysis of Residuals in Cross-Classified Tables. *Biometrics*, 29(1), 205. <https://doi.org/10.2307/2529686>
- Hartmann, D., & Dymond, R. (2006). *Appropriate Agile Measurement: Using Metrics and Diagnostics to Deliver Business Value*. 126–134. <https://doi.org/10.1109/AGILE.2006.17>
- Hashmi, S. I., & Baik, J. (2007). Software Quality Assurance in XP and Spiral—A Comparative Study. *International Conference on Computational Science and Its Applications, 2007. ICCSA 2007*, 367–374. <https://doi.org/10.1109/ICCSA.2007.65>
- Hongying, G., & Cheng, Y. (2011). A customizable agile software Quality Assurance model. *Information Science and Service Science (NISS), 2011 5th International Conference on New Trends In*, 2, 382–387.
- Hossain, S. S., Ahmed, P., & Arafat, Y. (2021). Software Process Metrics in Agile Software Development: A Systematic Mapping Study. In O. Gervasi, B. Murgante, S. Misra, C. Garau, I. Blečić, D. Taniar, B. O. Apduhan, A. M. A. C. Rocha, E. Tarantino, & C. M. Torre (Eds.), *Computational Science and Its Applications – ICCSA 2021* (pp. 15–26). Springer International Publishing. https://doi.org/10.1007/978-3-030-87013-3_2
- Huijgens, H., Lamping, R., Stevens, D., Rothengatter, H., Gousios, G., & Romano, D. (2017). Strong agile metrics: Mining log data to determine predictive power of software metrics for continuous delivery teams. *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, 866–871. <https://doi.org/10.1145/3106237.3117779>
- Hulkko, H., & Abrahamsson, P. (2005). A Multiple Case Study on the Impact of Pair Programming on Product Quality. *Proceedings of the 27th International Conference on Software Engineering*, 495–504. <https://doi.org/10.1145/1062455.1062545>
- Ichu, E., & Nemani, R. (2011). The Role of Quality Assurance in Software Development Projects: Project Failures and Business Performance. *International Journal of Computer Technology and Applications*, 2(4). <http://www.doaj.org/doaj?func=fulltext&aId=782086>
- IEEE SA 610.2. (1990). *IEEE SA - 610.12-1990—IEEE Standard Glossary of Software Engineering Terminology*. <http://standards.ieee.org/findstds/standard/610.12-1990.html>
- ISO. (2018). *ISO 26000—Guidance on how to contribute to sustainable development*. ISO

- Standardization. <https://iso26000.info/standardization/>
- ISO/IEC. (2005). *ISO/IEC 25000:2005—Software Engineering—Software product Quality Requirements and Evaluation (SQuaRE)—Guide to SQuaRE*. http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=35683
- ISO/IEC 9126. (2004). *ISO/IEC 9126-1:2001—Software engineering—Product quality—Part 1: Quality model*. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=22749
- ISO/IEC 25000:2005—Software Engineering – Software product Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE*. (n.d.). Retrieved June 30, 2014, from http://www.iso.org/iso/catalogue_detail.htm?csnumber=35683
- Ivancevich, J. M., Lorenzi, P., Skinner, S. J., & Crosby, P. B. (1997). *Management: Quality and competitiveness* (2nd ed.). Irwin.
- Jain, P., Ahuja, L., & Sharma, A. (2016). Current state of the research in agile quality development. *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, 1177–1179.
- Jain, P., Sharma, A., & Ahuja, L. (2016). ISM based identification of quality attributes for agile development. *2016 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 615–619. <https://doi.org/10.1109/ICRITO.2016.7785028>
- Jain, P., Sharma, A., & Ahuja, L. (2018). The Impact of Agile Software Development Process on the Quality of Software Product. *2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 812–815. <https://doi.org/10.1109/ICRITO.2018.8748529>
- Janus, A., Dumke, R., Schmietendorf, A., & Jager, J. (2012). The 3C approach for Agile Quality Assurance. *2012 3rd International Workshop on Emerging Trends in Software Metrics (WETSoM)*, 9–13. <https://doi.org/10.1109/WETSoM.2012.6226998>
- Jari Andersin. (2004). TPI – a model for Test Process Improvement. *Seminar on Quality Models for Software Engineering*, 1–16.
- Jeeva Padmini, K. V., Perera, I., & Dilum Bandara, H. M. N. (2016). Applying agile practices

- to avoid chaos in User Acceptance Testing: A case study. *2016 Moratuwa Engineering Research Conference (MERCon)*, 96–101. <https://doi.org/10.1109/MERCon.2016.7480122>
- Jeon, S., Han, M., Lee, E., & Lee, K. (2011). Quality Attribute Driven Agile Development. *2011 9th International Conference on Software Engineering Research, Management and Applications (SERA)*, 203–210. <https://doi.org/10.1109/SERA.2011.24>
- Jharko, E. (2021). Some Aspects of Quality Assurance in the Development of Digital Systems. *2021 14th International Conference Management of Large-Scale System Development (MLSD)*, 1–5. <https://doi.org/10.1109/MLSD52249.2021.9600164>
- Jhons, J., Sivan, A. P., Balachandran, K., & Prathap, B. R. (2014). The 3C Approach for Agile Scrum Software Methodology. *International Journal of Innovative Research in Science, Engineering and Technology*, 3(3), 1791–1795.
- Jinzenji, K., & Hoshino, T. (2013). Empirical study of Software Quality Evaluation in Agile Methodology Using Traditional Metrics. *International Symposium on Software Reliability Engineering (ISSRE)* 2013. http://collaboration.csc.ncsu.edu/laurie/Papers/issre2013_submission_18.pdf
- Jinzenji, K., Hoshino, T., Williams, L., & Takahashi, K. (2012). Metric-Based Quality Evaluations for Iterative Software Development Approaches Like Agile. *2012 IEEE 23rd International Symposium on Software Reliability Engineering Workshops (ISSREW)*, 54–63. <https://doi.org/10.1109/ISSREW.2012.40>
- Jinzenji, K., Hoshino, T., Williams, L., & Takahashi, K. (2013). An experience report for software quality evaluation in highly iterative development methodology using traditional metrics. *2013 IEEE 24th International Symposium on Software Reliability Engineering (ISSRE)*, 310–319. <https://doi.org/10.1109/ISSRE.2013.6698884>
- Jisha John. (2014, March 4). *The 3C Approach for Agile Scrum Software Methodology*. 3(3), 1792–1795.
- Julian, B., Noble, J., & Anslow, C. (2019). Agile Practices in Practice: Towards a Theory of Agile Adoption and Process Evolution. In P. Kruchten, S. Fraser, & F. Coallier (Eds.), *Agile Processes in Software Engineering and Extreme Programming* (pp. 3–18). Springer International Publishing. https://doi.org/10.1007/978-3-030-19034-7_1
- Kassie, N. B., & Singh, J. (2020). A study on software quality factors and metrics to enhance

- software quality assurance. *International Journal of Productivity and Quality Management*. <https://www.inderscienceonline.com/doi/10.1504/IJPQM.2020.104547>
- Kausar, M., Ishtiaq, M., & Hussain, S. (2022). Distributed Agile Patterns-Using Agile Practices to Solve Offshore Development Issues. *IEEE Access*, 10, 8840–8854. <https://doi.org/10.1109/ACCESS.2021.3136923>
- Kerzner, H. (2017). *Project Management: A Systems Approach to Planning, Scheduling, and Controlling, 11th Edition—Harold R. Kerzner* (12th ed.). Wiley. <http://www.wiley.com/WileyCDA/WileyTitle/productCd-1118022270.html>
- Khomh, F., & Guéhéneuc, Y.-G. (2008). DEQUALITE: Building Design-based Software Quality Models. *Proceedings of the 15th Conference on Pattern Languages of Programs*, 2:1-2:7. <https://doi.org/10.1145/1753196.1753199>
- Kitchenham, B., & Charters, S. (2007). *Guidelines for performing Systematic Literature Reviews in Software Engineering*. <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.117.471>
- Kovacheva, T., & Todorov, N. (2011). Optimizing software development process: A case study for integrated Agile-CMMI process model. *2011 IEEE EUROCON - International Conference on Computer as a Tool*, 1–2. <https://doi.org/10.1109/EUROCON.2011.5929328>
- Kropp, M., Meier, A., Anslow, C., & Biddle, R. (2018). Satisfaction, Practices, and Influences in Agile Software Development. *Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018*, 112–121. <https://doi.org/10.1145/3210459.3210470>
- Kropp, M., Meier, A., & Biddle, R. (2016). Agile Practices, Collaboration and Experience. In P. Abrahamsson, A. Jedlitschka, A. Nguyen Duc, M. Felderer, S. Amasaki, & T. Mikkonen (Eds.), *Product-Focused Software Process Improvement* (pp. 416–431). Springer International Publishing. https://doi.org/10.1007/978-3-319-49094-6_28
- Kula, E., Greuter, E., van Deursen, A., & Gousios, G. (2022). Factors Affecting On-Time Delivery in Large-Scale Agile Software Development. *IEEE Transactions on Software Engineering*, 48(9), 3573–3592. <https://doi.org/10.1109/TSE.2021.3101192>
- Kunz, M., Dumke, R. R., & Schmietendorf, A. (2008). How to Measure Agile Software Development. In J. J. Cuadrado-Gallego, R. Braungarten, R. R. Dumke, & A. Abran

- (Eds.), *Software Process and Product Measurement* (pp. 95–101). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-85553-8_8
- Kunz, M., Dumke, R. R., & Zenker, N. (2008). Software Metrics for Agile Software Development. *19th Australian Conference on Software Engineering, 2008. ASWEC 2008*, 673–678. <https://doi.org/10.1109/ASWEC.2008.4483261>
- Kupiainen, E., Mäntylä, M. V., & Itkonen, J. (2014). Why Are Industrial Agile Teams Using Metrics and How Do They Use Them? *Proceedings of the 5th International Workshop on Emerging Trends in Software Metrics*, 23–29. <https://doi.org/10.1145/2593868.2593873>
- Larman, C. (2004). *Agile and Iterative Development: A Manager's Guide*. Addison-Wesley Professional.
- Larsen, K. R., & Eargle, D. (2017). *Theories Used in IS Research*. Theories Used in IS Research Wiki. https://is.theorizeit.org/wiki/Main_Page
- Layman, L., Williams, L., & Cunningham, L. (2006). Motivations and measurements in an agile case study. *Journal of Systems Architecture*, 52(11), 654–667. <https://doi.org/10.1016/j.sysarc.2006.06.009>
- Letelier, P., & Penadés, M. C. (2006). *Métodologías ágiles para el desarrollo de software: EXtreme Programming (XP)*. <http://www.cyta.com.ar/ta0502/v5n2a1.htm>
- Letelier, P., & Penades, M. C. (2017). AgileRoadmap: An Approach to Implement Agile Practices in Teams. *IEEE Latin America Transactions*, 15(7), 1295–1300. <https://doi.org/10.1109/TLA.2017.7959349>
- Liubchenko, V. (2016). A review of agile practices for project management. *2016 XIth International Scientific and Technical Conference Computer Sciences and Information Technologies (CSIT)*, 168–170. <https://doi.org/10.1109/STC-CSIT.2016.7589897>
- López, L., Burgués, X., Martínez-Fernández, S., Vollmer, A. M., Behutiye, W., Karhapää, P., Franch, X., Rodríguez, P., & Oivo, M. (2022). Quality measurement in agile and rapid software development: A systematic mapping. *Journal of Systems and Software*, 186, 111187. <https://doi.org/10.1016/j.jss.2021.111187>
- Łukasiewicz, K., & Górski, J. (2016). AgileSafe—A method of introducing agile practices into safety-critical software development processes. *2016 Federated Conference on Computer Science and Information Systems (FedCSIS)*, 1549–1552.

- Maddox, M., & Walker, S. (2021). Agile Software Quality Metrics. *2021 IEEE MetroCon*, 1–3. <https://doi.org/10.1109/MetroCon54219.2021.9666049>
- Manifesto for Agile Software Development*. (2001). <http://agilemanifesto.org/>
- Manjunath, K. N., Jagadeesh, J., & Yogeesh, M. (2013). Achieving quality product in a long term software product development in healthcare application using Lean and Agile principles: Software engineering and software development. *2013 International Multi-Conference on Automation, Computing, Communication, Control and Compressed Sensing (IMac4s)*, 26–34. <https://doi.org/10.1109/iMac4s.2013.6526379>
- Marcial, A. S. C., Soares, F. S. F., & Belchior, A. D. (2007). Mapping CMMI Project Management Process Areas to SCRUM Practices. *31st IEEE Software Engineering Workshop, 2007. SEW 2007*, 13–22. <https://doi.org/10.1109/SEW.2007.102>
- Marrington, A., Hogan, J. M., & Thomas, R. (2005). Quality assurance in a student-based agile software engineering process. *Software Engineering Conference, 2005. Proceedings. 2005 Australian*, 324–331. <https://doi.org/10.1109/ASWEC.2005.38>
- Matthies, C., Huegle, J., Dürschmid, T., & Teusner, R. (2019). Attitudes, Beliefs, and Development Data Concerning Agile Software Development Practices. *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, 158–169. <https://doi.org/10.1109/ICSE-SEET.2019.00025>
- Matthies, C., Kowark, T., Uflacker, M., & Plattner, H. (2016). Agile metrics for a university software engineering course. *2016 IEEE Frontiers in Education Conference (FIE)*, 1–5. <https://doi.org/10.1109/FIE.2016.7757684>
- Meier, A., Kropp, M., Anslow, C., & Biddle, R. (2018). Stress in Agile Software Development: Practices and Outcomes. In J. Garbajosa, X. Wang, & A. Aguiar (Eds.), *Agile Processes in Software Engineering and Extreme Programming* (pp. 259–266). Springer International Publishing. https://doi.org/10.1007/978-3-319-91602-6_18
- Mendoza, L. E., Pérez, M. A., & Griman, A. C. (2005). Prototipo de Modelo Sistémico de Calidad (MOSCA) del Software. *Computación y Sistemas*, Vol. 8(No. 3), 196–217.
- Michael A. West & James L. Farr. (1990). *The social psychology of innovation in groups*. John Wiley & Sons, Lt.
- Mirnalini, K., & Raya, V. R. (2010). Agile—A software development approach for quality

- software. *2010 International Conference on Educational and Information Technology (ICEIT)*, 1, V1-242-V1-244. <https://doi.org/10.1109/ICEIT.2010.5607732>
- Mirza, M. N., Pourzolfaghar, Z., & Shahnazari, M. (2013). Significance of Scope in Project Success. *Procedia Technology*, 9, 722–729. <https://doi.org/10.1016/j.protcy.2013.12.080>
- Misra, S. C., Kumar, V., & Kumar, U. (2009). Identifying some important success factors in adopting agile software development practices. *Journal of Systems and Software*, 82(11), 1869–1890. <https://doi.org/10.1016/j.jss.2009.05.052>
- Misra, S., Omorodion, F. M., & Damasevicius, R. (2021). Metrics for Measuring Progress and Productivity in Agile Software Development. In A. Abraham, H. Sasaki, R. Rios, N. Gandhi, U. Singh, & K. Ma (Eds.), *Innovations in Bio-Inspired Computing and Applications* (pp. 469–478). Springer International Publishing. https://doi.org/10.1007/978-3-030-73603-3_44
- Mnkandla, E., & Dwolatzky, B. (2006). Defining Agile Software Quality Assurance. *International Conference on Software Engineering Advances*, 36–36. <https://doi.org/10.1109/ICSEA.2006.261292>
- Mohan, M. J., Ferrer, R. M., & Narayan, J. (2016). Scaling Agile Scrum Software Development: Providing Agility and Quality to Platform Development by Reducing Time to Market. *2016 IEEE 11th International Conference on Global Software Engineering (ICGSE)*, 84–88. <https://doi.org/10.1109/ICGSE.2016.24>
- Montagud, S., Abrahão, S., & Insfran, E. (2012). A systematic review of quality attributes and measures for software product lines. *Software Qual J*, 20(3), 425–486. <https://doi.org/10.1007/s11219-011-9146-7>
- Mundra, A., Misra, S., & Dhawale, C. A. (2013). Practical Scrum-Scrum Team: Way to Produce Successful and Quality Software. *2013 13th International Conference on Computational Science and Its Applications (ICCSA)*, 119–123. <https://doi.org/10.1109/ICCSA.2013.25>
- Muñoz, C. C., Velthuis, M. G. P., & Rubia, M. Á. M. de la. (2010). *Calidad del producto y proceso software*. Editorial Ra-Ma.
- Nader-Rezvani, N. (2019a). Agile Quality Test Strategy. In N. Nader-Rezvani (Ed.), *An Executive's Guide to Software Quality in an Agile Organization: A Continuous*

- Improvement Journey* (pp. 121–138). Apress. https://doi.org/10.1007/978-1-4842-3751-9_7
- Nader-Rezvani, N. (2019b). Quality in Agile. In N. Nader-Rezvani (Ed.), *An Executive's Guide to Software Quality in an Agile Organization: A Continuous Improvement Journey* (pp. 1–19). Apress. https://doi.org/10.1007/978-1-4842-3751-9_1
- Nataraj, R., Gopalakrishnan, R., & Eswaramoorthi, G. (2016). A Study of Emerging Trends in Software Testing and Quality Assurance. *Int. J. Eng. Manag. Res*, 1(6), 376–380.
- Nazir, N., Hasteer, N., & Bansal, A. (2016). A survey on agile practices in the Indian IT industry. *2016 6th International Conference - Cloud System and Big Data Engineering (Confluence)*, 635–640. <https://doi.org/10.1109/CONFLUENCE.2016.7508196>
- Nord, R. L., & Tomayko, J. E. (2006). Software architecture-centric methods and agile development. *IEEE Software*, 23(2), 47–53. <https://doi.org/10.1109/MS.2006.54>
- Nundlall, C., & Nagowah, S. D. (2021). Factors Affecting Task Allocation and Coordination in Distributed Agile Software Development. In C. R. Panigrahi, B. Pati, B. K. Pattanayak, S. Amic, & K.-C. Li (Eds.), *Progress in Advanced Computing and Intelligent Engineering* (pp. 817–829). Springer. https://doi.org/10.1007/978-981-33-4299-6_66
- Nunnally, J., & Bernstein, L. (1994). *Psychometric theory*. McGraw-Hill Higher, INC.
- Olaque, H. M., Etzkorn, L. H., Gholston, S., & Quattlebaum, S. (2007). Empirical Validation of Three Software Metrics Suites to Predict Fault-Proneness of Object-Oriented Classes Developed Using Highly Iterative or Agile Software Development Processes. *IEEE Transactions on Software Engineering*, 33(6), 402–419. <https://doi.org/10.1109/TSE.2007.1015>
- Opelt, K., & Beeson, T. (2008). Agile Teams Require Agile QA: How to Make it Work, An Experience Report. *Agile, 2008. AGILE '08. Conference*, 229–232. <https://doi.org/10.1109/Agile.2008.59>
- Opiyo, E. Z., Horváth, I., & Vergeest, J. S. M. (2002). Quality assurance of design support software: Review and analysis of the state of the art. *Computers in Industry*, 49(2), 195–215. [https://doi.org/10.1016/S0166-3615\(02\)00082-9](https://doi.org/10.1016/S0166-3615(02)00082-9)
- Paez, N., Fontdevila, D., Gainey, F., & Oliveros, A. (2018). Technical and Organizational Agile Practices: A Latin-American Survey. In J. Garbajosa, X. Wang, & A. Aguiar (Eds.), *Agile Processes in Software Engineering and Extreme Programming* (pp. 146–159).

- Springer International Publishing. https://doi.org/10.1007/978-3-319-91602-6_10
- Pressman, R. (2010). *Ingeniería del Software: Un Enfoque Práctico*, Roger Pressman (Sexta Edición) \textbackslashtextbar Intercambios Virtuales. <http://www.intercambiosvirtuales.org/libros-manuales/ingenieria-del-software-un-enfoque-practico-roger-pressman-sexta-edicion>
- Przybilla, L., Wiesche, M., & Krcmar, H. (2018). The Influence of Agile Practices on Performance in Software Engineering Teams: A Subgroup Perspective. *Proceedings of the 2018 ACM SIGMIS Conference on Computers and People Research*, 33–40. <https://doi.org/10.1145/3209626.3209703>
- Rada, R., & Craparo, J. (2000). Sharing standards: Standardizing software projects. *Communications of the ACM*, 43(12), 21–25. <https://doi.org/10.1145/355112.355117>
- Ran, H., Zhuo, W., & Jianfeng, X. (2009). Web Quality of Agile Web Development. *IITA International Conference on Services Science, Management and Engineering, 2009. SSME '09*, 426–429. <https://doi.org/10.1109/SSME.2009.112>
- Rico, D. F. (2008). Effects of Agile Methods on Website Quality for Electronic Commerce. *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, 463–463. <https://doi.org/10.1109/HICSS.2008.137>
- Roden, P. L., Virani, S., Etzkorn, L. H., & Messimer, S. (2007). An Empirical Study of the Relationship of Stability Metrics and the QMOOD Quality Models Over Software Developed Using Highly Iterative or Agile Software Processes. *Seventh IEEE International Working Conference on Source Code Analysis and Manipulation, 2007. SCAM 2007*, 171–179. <https://doi.org/10.1109/SCAM.2007.29>
- Rossberg, J. (2016). Metrics in Agile Projects. In J. Rossberg (Ed.), *Agile Project Management using Team Foundation Server 2015* (pp. 131–145). Apress. https://doi.org/10.1007/978-1-4842-1870-9_7
- Sagheer, M., Zafar, T., & Sirshar, M. (2015). A Framework For Software Quality Assurance Using Agile Methodology. *International Journal of Scientific & Technology Research*, 4(2), 44–50.
- Sampaio, A., Vasconcelos, A., & Sampaio, P. R. F. (2004). Assessing agile methods: An empirical study. *J Braz Comp Soc*, 10(2), 21–48. <https://doi.org/10.1007/BF03192357>
- Sanabria, W., & Acebey, V. (2008). *Métodos Ágiles*. <http://metodosagiles.blogspot.mx/>

- Santos, M. de A. (2011). Agile Practices: An Assessment of Perception of Value of Professionals on the Quality Criteria in Performance of Projects. *Journal of Software Engineering and Applications*, 04(12), 700–709. <https://doi.org/10.4236/jsea.2011.412082>
- Scharff, C. (2011). Guiding global software development projects using Scrum and Agile with quality assurance. *2011 24th IEEE-CS Conference on Software Engineering Education and Training (CSEET)*, 274–283. <https://doi.org/10.1109/CSEET.2011.5876097>
- Schneider, J.-G., Eklund, P. W., Lee, K., Chen, F., Cain, A., & Abdelrazek, M. (2020). Adopting industry agile practices in large-scale capstone education. *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering Education and Training*, 119–129. <https://doi.org/10.1145/3377814.3381715>
- Schwaber, K., & y Jeff Sutherland. (2013). *La Guía de Scrum*.
- Schwanke, R., Xiao, L., & Cai, Y. (2013). Measuring architecture quality by structure plus history analysis. *2013 35th International Conference on Software Engineering (ICSE)*, 891–900. <https://doi.org/10.1109/ICSE.2013.6606638>
- SEI. (2010). *CMMI ® for Development, Version 1.3* (TECHNICAL REPORT). Software Engineering Process Management Program. <http://www.sei.cmu.edu/reports/10tr033.pdf>
- Selleri Silva, F., Santana Furtado Soares, F., Lima Peres, A., Monteiro de Azevedo, I., Pereira Pinto, P., & Romero de Lemos Meira, S. (2014). A Reference Model for Agile Quality Assurance: Combining Agile Methodologies and Maturity Models. *Quality of Information and Communications Technology (QUATIC), 2014 9th International Conference on The*, 139–144. <https://doi.org/10.1109/QUATIC.2014.25>
- Senapathi, M., & Srinivasan, A. (2014). An Empirical Investigation of the Factors Affecting Agile Usage. *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, 10:1-10:10. <https://doi.org/10.1145/2601248.2601253>
- Singh, B., & Gautam, S. (2016a). The Impact of Software Development Process on Software Quality: A Review. *2016 8th International Conference on Computational Intelligence and Communication Networks (CICN)*, 666–672. <https://doi.org/10.1109/CICN.2016.137>

- Singh, B., & Gautam, S. (2016b). The Impact of Software Development Process on Software Quality: A Review. *2016 8th International Conference on Computational Intelligence and Communication Networks (CICN)*, 666–672. <https://doi.org/10.1109/CICN.2016.137>
- Singh, N. P., & Soni, R. (2011). Agile Software: Ensuring Quality Assurance and Processes. In A. Mantri, S. Nandi, G. Kumar, & S. Kumar (Eds.), *High Performance Architecture and Grid Computing* (pp. 640–648). Springer Berlin Heidelberg. http://link.springer.com.ezproxy.unal.edu.co/chapter/10.1007/978-3-642-22577-2_86
- Singh, Y., Kaur, A., & Malhotra, R. (2010). Empirical validation of object-oriented metrics for predicting fault proneness models. *Software Qual J*, 18(1), 3–35. <https://doi.org/10.1007/s11219-009-9079-6>
- Souza, R., Rocha, L., Silva, F., & Machado, I. (2019). Investigating Agile Practices in Software Startups. *Proceedings of the XXXIII Brazilian Symposium on Software Engineering*, 317–321. <https://doi.org/10.1145/3350768.3350786>
- Spagnoletti, P., Kazemargi, N., & Prencipe, A. (2022). Agile Practices and Organizational Agility in Software Ecosystems. *IEEE Transactions on Engineering Management*, 69(6), 3604–3617. <https://doi.org/10.1109/TEM.2021.3110105>
- Standish Group. (2020). *Chaos Report 2021*.
- Statistics over IT Failure Rate. (2001). http://www.it-cortex.com/Stat_Failure_Rate.htm
- Straub, D., & Karahanna, E. (1998). Knowledge Worker Communications and Recipient Availability: Toward a Task Closure Explanation of Media Choice. *Organization Science*, 9(2), 160–175. <https://doi.org/10.1287/orsc.9.2.160>
- Subramanyam, V., & Prasad, C. G. (2013). Organizational practices that effects Software Quality In Software Engineering process. *International Journal of Engineering Trends and Technology*, 2(3), 25–30.
- Sultana, S., Motla, Y. H., Asghar, S., Jamal, M., & Azad, R. (2014). A hybrid model by integrating agile practices for Pakistani software industry. *2014 International Conference on Electronics, Communications and Computers (CONIELECOMP)*, 256–262. <https://doi.org/10.1109/CONIELECOMP.2014.6808600>
- Sutherland, J., Jakobsen, C. R., & Johnson, K. (2008). Scrum and CMMI Level 5: The Magic Potion for Code Warriors. *Hawaii International Conference on System Sciences*,

- Proceedings of the 41st Annual*, 466–466. <https://doi.org/10.1109/HICSS.2008.384>
- Tarhan, A., & Yilmaz, S. G. (2014). Systematic analyses and comparison of development performance and product quality of Incremental Process and Agile Process. *Information and Software Technology*, 56(5), 477–494. <https://doi.org/10.1016/j.infsof.2013.12.002>
- Theobald, S., & Diebold, P. (2017). Beneficial and Harmful Agile Practices for Product Quality. In M. Felderer, D. Méndez Fernández, B. Turhan, M. Kalinowski, F. Sarro, & D. Winkler (Eds.), *Product-Focused Software Process Improvement* (pp. 586–593). Springer International Publishing. https://doi.org/10.1007/978-3-319-69926-4_48
- Tommy, R., Mhaisekar, M., Kallepally, S., Varghese, L., Ahmed, S., & Somaraju, M. D. (2015). Dynamic quality control in agile methodology for improving the quality. *2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS)*, 233–236. <https://doi.org/10.1109/CGVIS.2015.7449927>
- Um, T., Kim, N., Lee, D., & In, H. P. (2011). A Quality Attributes Evaluation Method for an Agile Approach. *2011 First ACIS/JNU International Conference on Computers, Networks, Systems and Industrial Engineering (CNSI)*, 460–461. <https://doi.org/10.1109/CNSI.2011.93>
- Versionone. (2017). *State of Agile*. <https://explore.versionone.com/state-of-agile>
- Versionone. (2020). *14th Annual State of Agile Report*. <https://stateofagile.com/home>
- Wegner, D. M. (1987). Transactive Memory: A Contemporary Analysis of the Group Mind. In B. Mullen & G. R. Goethals (Eds.), *Theories of Group Behavior* (pp. 185–208). Springer New York. https://doi.org/10.1007/978-1-4612-4634-3_9
- Wit, A. D. (1988). Measurement of project success. *International Journal of Project Management*, 6(3), 164–170. [https://doi.org/10.1016/0263-7863\(88\)90043-9](https://doi.org/10.1016/0263-7863(88)90043-9)
- Wnuk, K., & Maddila, K. C. (2017). Agile and lean metrics associated with requirements engineering. *Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement*, 33–40. <https://doi.org/10.1145/3143434.3143437>
- Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2012). *Experimentation in Software Engineering*. Springer Science & Business Media.
- Xu, B. (2009). Towards High Quality Software Development with Extreme Programming Methodology: Practices from Real Software Projects. *International Conference on*

- Management and Service Science*, 2009. MASS '09, 1–4.
<https://doi.org/10.1109/ICMSS.2009.5302042>
- Yamada, S., & Kii, R. (2015). Software quality analysis for agile development. *2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions)*, 1–5.
<https://doi.org/10.1109/ICRITO.2015.7359201>
- Zhou, Z. Q., Xiang, S., & Chen, T. Y. (2016). Metamorphic Testing for Software Quality Assessment: A Study of Search Engines. *IEEE Transactions on Software Engineering*, 42(3), 264–284. <https://doi.org/10.1109/TSE.2015.2478001>

ANEXOS

Anexo A. Principales estándares de calidad de Software

No.	Número y Título del estándar
1	IEEE Std. 730-2002 Standard for Software Quality Assurance Plans
2	IEEE Std. 982.1-2005 Standard for Dictionary of Measures of the Software Aspects of Dependability
3	IEEE Std. 1012-2012 Standard for System and Software Verification and Validation
4	IEEE Std. 1028-2008 Standard for Software Reviews and Audits
5	IEEE Std. 1044-2009 Standard for Classification for Software Anomalies
6	IEEE Std. 1061-1998 Standard for Software Quality Metrics Methodology
7	IEEE Std. 1228-1994 Standard for Software Safety Plans
8	IEEE Std. 1633-2008 Recommended Practice for Software Reliability
9	IEEE Std. 15026.1-2011 Trial-Use Standard Adoption of ISO/IEC TR 15026-1:2010 Systems and Software Engineering—Systems and Software Assurance—Part 1: Concepts and Vocabulary
10	IEEE Std. 15026.2-2011 Standard Adoption of ISO/IEC 15026- 2:2011 Systems and Software Engineering—Systems and Software Assurance—Part 2: Assurance Case
11	ISO/IEC 15026-3 Systems and Software Engineering—Systems and Software Assurance—Part 3: System Integrity Levels
12	ISO/IEC 15026-4:2012 Systems and Software Engineering—Systems and Software Assurance—Part 4: Assurance in the Life Cycle
13	ISO/IEC 25000:2005 Software Engineering—Software Product Quality Requirements and Evaluation (SQuaRE)—Guide to SQuaRE
14	ISO/IEC 25000 through 25099 Software Engineering—Software Product Quality Requirements and Evaluation (SQuaRE)
15	ISO/IEC 25010:2011 Systems and Software Engineering—Systems and Software Quality Requirements and Evaluation (SQuaRE)—System and Software Quality Models
16	ISO/IEC 25060 through 25064 Software Engineering—Software Product Quality Requirements and Evaluation (SQuaRE)—Common Industry Format (CIF) for Usability
17	ISO 9000-1 Quality management and quality assurance standards, Part 1: Guidelines for selection and use
18	ISO 9001 Quality systems. Model for quality assurance in design, development, production, installation and servicing.
19	ISO 9002 Quality systems. Model for quality assurance in production, installation and servicing.
20	ISO 9003 Quality systems. Model for quality assurance in final inspection and test
21	ISO 9000-2 Quality management and quality assurance standards, Part 2: Generic guidelines for the applica-tion of ISO 9001, ISO 9002 and ISO 9003
22	ISO 9000-3 Quality management and quality assurance standards, Part 3: Guidelines for the application ofISO 9001 to the development, supply and maintenance of software.
23	ISO 9004-1Quality management and quality system elements, Part 1: Guidelines
24	ISO/IEC 9126 Software Product Evaluation - Quality Characteristics and Guidelines for the User.

No.	Número y Título del estándar
25	ISO/IEC 9126-1: Software Engineering -- Product quality -- Part 1: Quality model. 2001.
26	ISO 9126-2. 2003. ISO TR 9126-2: Software engineering –Product quality –Part 2: External metrics.
27	ISO 9126-3. 2003. ISO TR 9126-3: Software engineering –Product quality –Part 3: Internal metrics.
28	ISO 9126-4. 2004. ISO TR 9126-4: Software engineering –Product quality –Part 4: Quality in Use metrics.
29	IEEE Std. 90003:2008 Guide—Adoption of ISO/IEC 90003:2004 Software Engineering—Guidelines for the Application of ISO 9001:2000 to Computer Software
30	Capability Maturity Model Integration (CMMI)

Anexo B. Estudios incluidos en la Revisión Sistemática de Literatura

CÓDIGO	TÍTULO	AUTOR
[S1]	“Effects of Agile Methods on Website Quality for Electronic Commerce”.	(Rico, 2008)
[S2]	“A survey study of critical success factors in agile software projects”.	(Chow & Cao, 2008)
[S3]	“The situational factors that affect the software development process: Towards a comprehensive reference framework”.	(Clarke & O’Connor, 2012)
[S4]	“Agile Principles and Achievement of Success in Software Development: A Quantitative Study in Brazilian Organizations”.	(Bermejo et al., 2014)
[S5]	“How do personality, team processes and task characteristics relate to job satisfaction and software quality?”.	(Acuña et al., 2009)
[S6]	“Identifying some important success factors in adopting agile software development practices”.	(S. C. Misra et al., 2009)
[S7]	“Are team personality and climate related to satisfaction and software quality? Aggregating results from a twice replicated experiment”.	(Acuña et al., 2015)
[S8]	“Towards understanding the relationship between team climate and software quality—a quasi-experimental study”.	(Acuña et al., 2008)
[S9]	“Users’ involvement may help respect social and ethical values and improve software quality”.	(Begier, 2010)
[S10]	“Influence of Human Factors in Software Quality and Productivity”.	(Fernández-Sanz & Misra, 2011)
[S11]	“An Empirical Investigation of the Factors Affecting Agile”.	(Senapathi & Srinivasan, 2014)
[S12]	“An Empirical Study on the Relationship between the Use of Agile Practices and the Success of Software Projects that Use Scrum”.	(de Souza Mariz et al., 2010)
[S13]	“ISM based identification of quality attributes for agile development”.	(Jain, Sharma, et al., 2016)
[S14]	“How Temporal Work Styles and Product Modularity Influence Software Quality and Job Satisfaction”.	(Foerderer et al., 2016)
[S15]	“Agile Enterprise Metrics”.	(Greening, 2015)
[S16]	“Dynamic quality control in agile methodology for improving the quality”.	(Tommy et al., 2015)
[S17]	“Software quality analysis for agile development”.	(Yamada & Kii, 2015)
[S18]	“Empirical Validation of Three Software Metrics Suites to Predict Fault-Proneness of Object-Oriented Classes Developed Using Highly Iterative or Agile Software Development Processes”.	(Olague et al., 2007)
[S19]	“Appropriate Agile Measurement: Using Metrics and Diagnostics to Deliver Business Value”.	(Hartmann & Dymond, 2006)
[S20]	“An Empirical Study of the Relationship of Stability Metrics and the QMOOD Quality Models Over Software Developed Using Highly Iterative or Agile Software	(Roden et al., 2007)

CÓDIGO	TÍTULO	AUTOR
	Processes”.	
[S21]	“Applicability of Software Reliability Growth Modeling in the Quality Assurance Phase of a Large Business Software Vendor”.	(Beckhaus et al., 2009)
[S22]	“Measuring architecture quality by structure plus history analysis”.	(Schwanke et al., 2013)
[S23]	“Metric-Based Quality Evaluations for Iterative Software Development Approaches Like Agile”.	(Jinzenji et al., 2012)
[S24]	“Measuring and Monitoring Agile Development Status”.	(Boerman et al., 2015)
[S25]	“Motivations and measurements in an agile case study”.	(Layman et al., 2006)
[S26]	“How to Measure Agile Software Development”.	(Kunz, Dumke, & Schmietendorf, 2008)
[S27]	“Studying the Evolution of Quality Metrics in an Agile/Distributed Project”.	(Ambu et al., 2006)
[S28]	“Empirical validation of object-oriented metrics for predicting fault proneness models’.	(Y. Singh et al., 2010)
[S29]	“Software Metrics in Agile Software: An Empirical Study”.	(Destefanis et al., 2014)
[S30]	“A Multiple Case Study on the Impact of Pair Programming on Product Quality”.	(Hulkko & Abrahamsson, 2005)
[S31]	“Why Are Industrial Agile Teams Using Metrics and How Do They Use Them?”.	(Kupiainen et al., 2014)
[S32]	“Software Metrics for Agile Software Development”.	(Kunz, Dumke, & Zenker, 2008)
[S33]	“Method for Assessing Software Quality Using Source Code Analysis”	(Forouzani et al., 2016)
[S34]	“Optimizing software development process: A case study for integrated Agile-CMMI process model”.	(Kovacheva & Todorov, 2011)
[S35]	“The projection of the specific practices of the third level of CMMI model in agile methods: Scrum, XP and Kanban”.	(Bougroun et al., 2014)
[S36]	“Quality assurance in a student-based agile software engineering process”.	(Marrington et al., 2005)
[S37]	“Agile Teams Require Agile QA: How to Make it Work, An Experience Report”.	(Opelt & Beeson, 2008)
[S38]	“Scrum and CMMI Level 5: The Magic Potion for Code Warriors”.	(Sutherland et al., 2008)
[S39]	“Towards High Quality Software Development with Extreme Programming Methodology: Practices from Real Software Projects”.	(Xu, 2009)
[S40]	“Agile software development: Impact on productivity and quality”.	(Ahmed et al., 2010)
[S41]	“Agile - A software development approach for quality software”.	(Mirnalini & Raya, 2010)
[S42]	“Guiding global software development projects using Scrum and Agile with quality assurance”.	(Scharff, 2011)
[S43]	“Practical Scrum-Scrum Team: Way to Produce Successful”	(Mundra et al., 2013)
[S44]	“Achieving quality product in a long term software	(Manjunath et al., 2013)

CÓDIGO	TÍTULO	AUTOR
	product development in healthcare application using Lean and Agile principles: Software engineering and software development”.	
[S45]	“Mapping CMMI Project Management Process Areas to SCRUM Practices”.	(Marcal et al., 2007)
[S46]	“Elaboration on an integrated architecture and requirement practice: Prototyping with quality attribute focus”.	(Bellomo et al., 2013)
[S47]	“A hybrid model by integrating agile practices for Pakistani software industry”.	(Sultana et al., 2014)
[S48]	“A survey of software engineering practices in Turkey”.	(Garousi et al., 2015)
[S49]	“Agile Practices in Practice: A Mapping Study”.	(Diebold & Dahlem, 2014)
[S50]	“An analysis of the factors determining software product quality: A comparative study”.	(Curcio et al., 2016a)
[S51]	“Metamorphic Testing for Software Quality Assessment: A Study of Search Engines”.	(Zhou et al., 2016)
[S52]	“Quality Assurance in Agile: A Study towards Achieving Excellence”.	(Bhasin, 2012)
[S53]	“A Reference Model for Agile Quality Assurance: Combining Agile Methodologies and Maturity Models”.	(Selleri Silva et al., 2014)
[S54]	“Software architecture-centric methods and agile development”.	(Nord & Tomayko, 2006)
[S55]	“A customizable agile software Quality Assurance model”.	(Hongying & Cheng, 2011)
[S56]	“A Quality Attributes Evaluation Method for an Agile Approach”.	(Um et al., 2011)
[S57]	“A Metamodel-Based Approach for Customizing and Assessing Agile Methods”.	(Ayed et al., 2012)
[S58]	“A Measurement-Based Framework for Agile Methods Customisation”.	(Ayed et al., 2013)
[S59]	“An experience report for software quality evaluation in highly iterative development methodology using traditional metrics”.	(Jinzenji et al., 2013)
[S60]	“The 3C approach for Agile Quality Assurance”.	(Janus et al., 2012)
[S61]	“Systematic analyses and comparison of development performance and product quality of Incremental Process and Agile Process”.	(Tarhan & Yilmaz, 2014)
[S62]	“Assessing agile methods: An empirical study”.	(Sampaio et al., 2004)
[S63]	“Agile Software: Ensuring Quality Assurance and Processes”.	(N. P. Singh & Soni, 2011)
[S64]	“Using the COSMIC Method to Evaluate the Quality of the Documentation of Agile User Stories”.	(Desharnais et al., 2011)
[S65]	“Quality Attribute Driven Agile Development”.	(Jeon et al., 2011)
[S66]	“Defining Agile Software Quality Assurance”.	(Mnkandla & Dwolatzky, 2006)
[S67]	“Web Quality of Agile Web Development”.	(Ran et al., 2009)
[S68]	“Lightweight and Continuous Architectural Software Quality Assurance Using the aSQA Technique”.	(Christensen et al., 2010)
[S69]	“DEQUALITE: Building Design-based Software Quality	(Khomh & Guéhéneuc,

CÓDIGO	TÍTULO	AUTOR
	Models”.	2008)
[S70]	“Software quality score board based on SQA framework to improve software reliability”.	(Bansal & Pundir, 2016)
[S71]	“Scaling Agile Scrum Software Development: Providing Agility and Quality to Platform Development by Reducing Time to Market”.	(Mohan et al., 2016)
[S72]	“Factors Affecting On-Time Delivery in Large-Scale Agile Software Development”	(Kula et al., 2022)
[S73]	“Factors Affecting Software Development Productivity: An empirical study”	(Canedo & Santos, 2019)
[S74]	“Factors affecting Agile adoption: An industry research study of the mobile app sector in Saudi Arabia”	(Altuwaijri & Ferrario, 2022)
[S75]	Factors Affecting Task Allocation and Coordination in Distributed Agile Software Development	(Nundlall & Nagowah, 2021)
[S76]	Stress in Agile Software Development: Practices and Outcomes	(Meier et al., 2018)
[S77]	“Software Practices For Agile Developers: A Systematic Literature Review”	(Arnyndiasari et al., 2022)
[S78]	“Agile Practices and Organizational Agility in Software Ecosystems”	(Spagnoletti et al., 2022)
[S79]	“A survey on agile practices in the Indian IT industry”	(Nazir et al., 2016)
[S80]	“Agile Development Practices Applied to Software Startups : A Systematic Mapping Review”	(Abrantes & Paula Furtado, 2021)
[S81]	“A review of agile practices for project management”	(Liubchenko, 2016)
[S82]	“Attitudes, Beliefs, and Development Data Concerning Agile Software Development Practices”	(Matthies et al., 2019)
[S83]	“AgileSafe - a method of introducing agile practices into safety-critical software development processes”	(Łukasiewicz & Górski, 2016)
[S84]	“Distributed Agile Patterns-Using Agile Practices to Solve Offshore Development Issues”	(Kausar et al., 2022)
[S85]	“AgileRoadmap: An Approach to Implement Agile Practices in Teams”	(Letelier & Penades, 2017)
[S86]	“Applying agile practices to avoid chaos in User Acceptance Testing: A case study”	(Jeeva Padmini et al., 2016)
[S87]	Agile Practices in Practice: Towards a Theory of Agile Adoption and Process Evolution	(Julian et al., 2019)
[S88]	Agile Practices, Collaboration and Experience	(Kropp et al., 2016)
[S89]	Implementing agile practices: the experience of TSol	(Bastarrica et al., 2018)
[S90]	Investigating Agile Practices in Software Startups	(Souza et al., 2019)
[S91]	Adopting industry agile practices in large-scale capstone education	(Schneider et al., 2020)
[S92]	The Influence of Agile Practices on Performance in Software Engineering Teams: A Subgroup Perspective	(Przybilla et al., 2018)
[S93]	Technical and Organizational Agile Practices: A Latin-American Survey	(Paez et al., 2018)
[S94]	Beneficial and Harmful Agile Practices for Product Quality	(Theobald & Diebold, 2017)
[S95]	“Agile Software Quality Metrics”	(Maddox & Walker, 2021)
[S96]	Agile metrics for a university software engineering course	(Matthies et al., 2016)

CÓDIGO	TÍTULO	AUTOR
[S97]	A Study of Quality Metrics in Agile Software Development	(Chakravarty & Singh, 2021)
[S98]	Metrics in Agile Projects	(Rossberg, 2016)
[S99]	Satisfaction, Practices, and Influences in Agile Software Development	(Kropp et al., 2018)
[S100]	Strong agile metrics: mining log data to determine predictive power of software metrics for continuous delivery teams	(Huijgens et al., 2017)
[S101]	Agile and lean metrics associated with requirements engineering	(Wnuk & Maddila, 2017)
[S102]	Software Process Metrics in Agile Software Development: A Systematic Mapping Study	(Hossain et al., 2021)
[S103]	Metrics for Measuring Progress and Productivity in Agile Software Development	(S. Misra et al., 2021)
[S104]	Agile and Quality: A Systematic Mapping Study	(Abdalhamid et al., 2019)
[S105]	Current state of the research in agile quality development	(Jain, Ahuja, et al., 2016)
[S106]	The Impact of Agile Software Development Process on the Quality of Software Product	(Jain et al., 2018)
[S107]	Usability in agile software development: A tertiary study	(Curcio et al., 2019)
[S108]	Quality in Agile	(Nader-Rezvani, 2019b)
[S109]	Agile Quality Test Strategy	(Nader-Rezvani, 2019a)
[S110]	Documentation of Quality Requirements in Agile Software Development	(Behutiye et al., 2020)
[S111]	Quality measurement in agile and rapid software development: A systematic mapping	(López et al., 2022)

Anexo C. Evaluación de la Calidad de los papers analizados

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S1	Factores	IEEE	Proceedings	Effects of Agile Methods on Website Quality for Electronic Commerce	Rico, D.F.	2008	22	0.149	—	10	10
S10	Factores	Springer	Proceedings	Influence of Human Factors in Software Quality and Productivity	Fernández-Sanz L, Misra S.	2011	41	0.92	—	10	10
S11	Factores - Atributo	ACM	Proceedings	An Empirical Investigation of the Factors Affecting Agile Usage	Mali Senapathi, Ananth Srinivasan	2014	43	0.151	C	10	10
S12	Factores	ACM	SIMPOSIO	An empirical study on the relationship between the use of agile practices and the success of Scrum projects	A. César C. França, Fabio Q. B. da Silva, Leila M. R. de Souza Mariz	2010	36	0.172	B	10	10
S13	Atributos	IEEE	Conference	ISM Based Identification of Quality Attributes for Agile Development	Parita Jain, Arun Sharma, Laxmi Ahuja	2016	12	0.934	—	10	10
S14	Factores	ACM	Conference	How Temporal Work Styles and Product Modularity Influence Software Quality and Job Satisfaction	Jens Foerderer, Thomas Kude, Sunil Mithas, Armin Heinzl	2016	6	0,158	—	10	10
S15	Métricas	IEEE	Conference	Agile Enterprise Metrics	Daniel R Greening	2015	12	—	A	10	10
S16	Métricas	IEEE	Conference	Dynamic quality control in agile methodology for improving the quality	Robin Tommy ; Meghana Mhaisekar ; Sangeeta Kallepally ; Libin Varghese ; Shabbir Ahmed ; Madhuri Devi Somaraju	2015	9	—	B	10	6

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S17	Metrics-atributos	IEEE	Conference	Software quality analysis for agile development	Shigeru Yamada; Ryosuke Kii	2015	12	1,11	—	10	10
S18	Métricas	IEEE	Journal	Empirical validation of three software metrics suites to predict fault-proneness of object-oriented classes developed using highly iterative or agile software development processes	Olague, Hector M.; Etzkorn, Letha H.; Gholston, Sampson; et ál	2007	434	0,105	—	10	10
S19	Métricas	IEEE	Proceedings	Appropriate Agile Measurement: Using Metrics and Diagnostics to Deliver Business Value	Deborah Hartmann, Robin Dymond	2006	179	0.106	C	10	10
S2	Factores	Science Direct	Journal	A survey study of critical success factors in agile software projects	Chow, Tsun; Cao, Dac-Buu	2008	1483	0.104	C	10	10
S20	Métricas - Atributos	IEEE	Proceedings	An Empirical Study of the Relationship of Stability Metrics and the QMOOD Quality Models Over Software Developed Using Highly Iterative or Agile Software Processes	Roden, P.L. ; Virani, S. ; Etzkorn, L.H. ; Messimer, S.	2007	19	0.101	B	10	10
S21	Métricas	IEEE	Proceedings	Applicability of Software Reliability Growth Modeling in the Quality Assurance Phase of a Large Business Software Vendor	Beckhaus, A. ; Karg, L.M. ; Hanselmann, G.	2009	5	0.109	—	5	10
S22	Métricas	IEEE	Proceedings	Measuring architecture quality by structure plus history analysis	Schwanke, R. ; Lu Xiao ; Yuanfang Cai	2013	56	0.897	—	10	10

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S23	metricas	ACM	SIMPOSIO	Metric-Based Quality Evaluations for Iterative Software Development Approaches Like Agile	Jinzenji, K. ; Hoshino, T. ; Williams, L. ; Takahashi, K.	2012	6	0.92	—	10	10
S24	Métricas - Atributos	ACM	Workshop	Measuring and Monitoring Agile Development Status	Martin P. Boerman; Zeeger Lubsen; Damian A. Tamburri; Joost Visser	2015	18	—	—	10	2
S25	Métricas - Factores	Science Direct	Journal	Motivations and measurements in an agile case study	Layman, Lucas; Williams, Laurie; Cunningham, Lynn	2006	143	0,89	—	10	10
S26	Métricas	SPRINGER	Proceedings	How to Measure Agile Software Development	Martin Kunz, Reiner R. Dumke	2007	18	0.89	—	10	10
S27	Métricas	Springer	Proceedings	Studying the evolution of quality metrics in an agile/distributed project	Ambu, Walter; Concas, Giulio; Marchesi, Michele; et ál	2006	14	0.76	—	10	10
S28	Métricas	Springer	Journal	Empirical validation of object-oriented metrics for predicting fault proneness models	Yogesh Singh, Arvinder Kaur, Ruchika Malhotra	2010	245	0.105	—	10	10
S29	Métricas	Springer	Proceedings	Software Metrics in Agile Software: An Empirical Study	Giuseppe Destefanis, Steve Counsell, Giulio Concas, Roberto Tonelli	2014	43	0.65	—	10	10
S3	Factores	Science Direct	Journal	The situational factors that affect the software development process: Towards a comprehensive reference framework	Clarke, Paul; O'Connor, Rory V.	2012	420	0,192	—	10	10
S30	Métricas	ACM	Proceedings	A multiple case study on the impact of pair programming on product quality	Hanna Hulkko, Pekka Abrahamsson	2005	206	0,261	—	10	10

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S31	Métricas	ACM	Workshop	Why are industrial agile teams using metrics and how do they use them?	Eetu Kupiainen, Mika V. Mäntylä, Juha Itkonen	2014	35	0.139	—	10	10
S32	Métricas	IEEE	Proceedings	Software Metrics for Agile Software Development	Kunz, Martin ; Dumke, R.R. ; Zenker, N.	2008	46	0.106	C	10	10
S33	Método/Modelo/Enfoques	ACM	Conference	Method for Assessing Software Quality Using Source Code Analysis	Sepehr Forouzani, Yin Kia Chiam, Soroush Forouzani	2016	131	0.92	—	10	10
S34	Prácticas ágiles	IEEE	Conference	Optimizing software development process: A case study for integrated Agile-CMMI process model	Tsvetelina Kovacheva; Nikolay Todorov	2011	11	0,168	—	10	10
S35	Prácticas ágiles	IEEE	Conference	The projection of the specific practices of the third level of CMMI model in agile methods: Scrum, XP and Kanban	Zineb Bougroun, Adil Zeaaraoui, T.Bouchentouf	2014	25	0.172	—	10	10
S36	Prácticas ágiles	IEEE	Proceedings	Quality assurance in a student-based agile software engineering process	Marrington, A. ; Hogan, J.M. ; Thomas, R.	2005	13	0.348	A	10	10
S37	Prácticas ágiles	IEEE	Proceedings	Agile Teams Require Agile QA: How to Make it Work, An Experience Report	Opelt, K. ; Beeson, T.	2008	16	0.92	—	10	10
S38	Prácticas ágiles	IEEE	Proceedings	Scrum and CMMI Level 5: The Magic Potion for Code Warriors	Jeff Sutherland; Carsten Ruseng Jakobsen; Kent Johnson	2008	223	0,144	—	10	10

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S39	Prácticas ágiles	IEEE	Proceedings	Towards High Quality Software Development with Extreme Programming Methodology: Practices from Real Software Projects	Bin Xu	2009	22	0.134	—	10	10
S4	Factores	Science Direct	Conference	Agile principles and achievement of success in software development: A quantitative study in Brazilian organizations	Paulo Henrique de Souza Bermejoa,b,c,* , André Luiz Zambaldea,b, Adriano Olímpio Tonellib,d, Samara Alyne Souzaa, Larissa Avelino Zuppoa, Priscila Luiz Rosaa	2014	39	0.123	C	10	10
S40	Prácticas ágiles	IEEE	Proceedings	Agile software development: Impact on productivity and quality	Ahmed, A. ; Ahmad, S. ; Ehsan, N. ; Mirza, E. ; Sarwar, S.Z.	2010	136	0.817	A	10	10
S41	Prácticas ágiles	IEEE	Proceedings	Agile - A software development approach for quality software	Mirnalini, K. ; Raya, V.R.	2010	8	0.4	—	10	10
S42	Prácticas ágiles	IEEE	Proceedings	Guiding global software development projects using Scrum and Agile with quality assurance	Scharff, C.	2011	45	—	B	10	5
S43	Prácticas ágiles	IEEE	Proceedings	Practical Scrum-Scrum Team: Way to Produce Successful and Quality Software	Mundra, A. ; Misra, S. ; Dhawale, C.A.	2013	95	1.54	—	10	10

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S44	Prácticas ágiles	IEEE	Proceedings	Achieving quality product in a long term software product development in healthcare application using Lean and Agile principles: Software engineering and software development	Manjunath, K.N. ; Jagadeesh, J. ; Yogeesh, M.	2013	22	0.253	C	10	10
S45	Prácticas ágiles	IEEE	Workshop	Mapping CMMI Project Management Process Areas to SCRUM Practices	Ana Sofia C. Marcal; Felipe S. Furtado Soares; Arnaldo D. Belchior	2007	11	—	B	10	5
S46	Prácticas ágiles	IEEE	Workshop	Elaboration on an integrated architecture and requirement practice: Prototyping with quality attribute focus	Stephany Bellomo, Robert L. Nord, Ipek Ozkaya	2013	10	0.62	—	10	10
S47	Prácticas ágiles	IEEE	Conference	A Hybrid Model by Integrating Agile Practices for Pakistani Software Industry	Saadia Sultana, Yasir Hafeez Motla, Sohail Asghar, Muhammad Jamal, Romana Azad	2014	26	—	—	10	2
S48	Prácticas ágiles- Métricas	Science Direct	Journal	A survey of software engineering practices in Turkey	Vahid Garousia, , E-mail the corresponding author, Ahmet Coşkunçayb, , Aysu Betin-Canb, , Onur Demirörsb,	2015	114	0,348	A	10	10
S49	Prácticas ágiles	ACM	Proceedings	Agile Practices in Practice - A Mapping Study -	Philipp Diebold, Marc Dahlem	2014	115	0.137	—	10	10

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S5	Factores	Science Direct	Journal	How do personality, team processes and task characteristics relate to job satisfaction and software quality?	Acuña, Silvia T.; Gomez, Marta; Juristo, Natalia	2009	310	—	B	10	5
S50	Factores	Science Direct	Journal	An analysis of the factors determining software product quality: A comparative study	Karina Curcio, Andreia Malucelli, Sheila Reinehr, , Marco Antônio Paludo	2016	29	0,262	—	10	10
S51	Métricas	IEEE	Journal	Metamorphic Testing for Software Quality Assessment: A Study of Search Engines	Zhi Quan Zhou ; Shaowen Xiang ; Tsong Yueh Chen	2016	153	—	A	10	10
S52	Prácticas ágiles	IEEE	Proceedings	Quality Assurance in Agile: A Study towards Achieving Excellence	Bhasin, S.	2012	23	0,788	—	10	10
S53	Método/Modelo/Enfoques	IEEE	Conference	A Reference Model for Agile Quality Assurance Combining Agile Methodologies and Maturity Models	Fernando Selleri Silva, Felipe Santana Furtado Soares, Angela Lima Peres, Ivanildo Monteiro de Azevedo, Pietro Pereira Pinto and Silvio Romero de Lemos Meira	2014	22	0.172	B	10	10
S54	Método/Modelo/Enfoques	IEEE	Journal	Software architecture-centric methods and agile development	Nord, R.L. ; Tomayko, James E.	2006	177	0.1	—	10	10
S55	Método/Modelo/Enfoques	IEEE	Proceedings	A customizable agile software Quality Assurance model	Gu Hongying ; Yang Cheng	2011	29	0,143	—	10	10
S56	Método/Modelo/Enfoques	IEEE	Proceedings	A Quality Attributes Evaluation Method for an Agile Approach	Taehoon Um ; Neunghoe Kim ; Donghyun Lee ; Hoh Peter In	2011	17	—	—	10	2

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S57	Método/Modelo/Enfoques	IEEE	Proceedings	A Metamodel-Based Approach for Customizing and Assessing Agile Methods	Ayed, H. ; Vanderose, B. ; Habra, N.	2012	27	0.108	—	10	10
S58	Método/Modelo/Enfoques	IEEE	Proceedings	AM-QuICK: A Measurement-Based Framework for Agile Methods Customisation	Ayed, H. ; Habra, N. ; Vanderose, B.	2013	15	0.104	C	10	10
S59	Método/Modelo/Enfoques	IEEE	SIMPOSIO	An experience report for software quality evaluation in highly iterative development methodology using traditional metrics	Jinzenji, K. ; Hoshino, T. ; Williams, L. ; Takahashi, K.	2013	16	0.132	—	10	10
S6	Factores	Science Direct	Journal	Identifying some important success factors in adopting agile software development practices	Subhas Chandra Misra, Vinod Kumar, Uma Kumar	2009	522	0.132	—	10	10
S60	Método/Modelo/Enfoques	IEEE	Workshop	The 3C approach for Agile Quality Assurance	Janus, A. ; Dumke, R. ; Schmietendorf, A. ; Jager, J.	2012	62	0.146	—	10	10
S61	Método/Modelo/Enfoques	Science Direct	Journal	Systematic analyses and comparison of development performance and product quality of Incremental Process and Agile Process	Ayca Tarhan, Seda Gunes Yilmaz	2014	94	0.140	—	10	10
S62	Método/Modelo/Enfoques	Springer	Journal	Assessing agile methods: An empirical study	Américo Sampaio, Alexandre Vasconcelos...	2004	14	0.1	A	10	10
S63	Método/Modelo/Enfoques	Springer	Proceedings	Agile Software: Ensuring Quality Assurance and Processes	Narinder Pal Singh, Rachna Soni	2011	7	0,7	A	10	10

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S64	Método/Modelo/Enfoques	ACM	Proceedings	Using the COSMIC Method to Evaluate the Quality of the Documentation of Agile User Stories	Desharnais, J.-M. ; Kocaturk, B. ; Abran, A.	2011	20	0,192	—	10	10
S65	Método/Modelo/Enfoques	ACM	Proceedings	Quality Attribute driven Agile Development	Sanghoon Jeon ¹ , Myungjin Han ¹ , Eunseok Lee ² , Keun Lee ¹	2011	37	—	—	10	2
S66	Atributos	IEEE	Proceedings	Defining Agile Software Quality Assurance	Mnkandla, E. ; Dwolatzky, B.	2006	76	—	—	10	2
S67	Atributos	IEEE	Proceedings	Web Quality of Agile Web Development	Hu Ran ; Wang Zhuo ; Xu Jianfeng	2009	9	—	—	10	2
S68	Atributos	Springer	Proceedings	Lightweight and Continuous Architectural Software Quality Assurance Using the aSQA Technique	Henrik Bærbak Christensen, Klaus Marius Hansen, Bo Lindstrøm	2010	16	0,193	—	10	10
S69	Atributos	ACM	Proceedings	DEQUALITE: building design-based software quality models	Foutse Khomh, Yann-Gaël Guéhéneuc	2008	22	—	—	5	2
S7	Factores	Science Direct	Journal	Are team personality and climate related to satisfaction and software quality? Aggregating results from a twice replicated experiment	Silvia T. Acuña, , Marta N. Gómezb, , Jo E. Hannayc, , Natalia Juristod, , Dietmar Pfahle,	2015	68	0,1	—	10	10
S70	Métricas	IEEE	Conference	Software quality score board based on SQA framework to improvise software reliability	Anshu Bansal ; Sudhir Pundir	2016	2	0,152	—	5	10

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S71	Prácticas ágiles	IEEE	Conference	Scaling Agile Scrum Software Development: Providing Agility and Quality to Platform Development by Reducing Time to Market	Madan Mohan Jha; Rosa Maria Ferrer Vilardell; Jai Narayan	2016	28	0,721	—	10	10
S8	Factores	Springer	Journal	Towards understanding the relationship between team climate and software quality—a quasi-experimental study	Silvia T. Acuña, Marta Gómez, Natalia Juristo	2008	82	0,1	—	10	10
S9	Factores	Springer	Journal	Users' involvement may help respect social and ethical values and improve software quality	Barbara Begier	2010	25	0,428	—	10	10
S72	Factores	IEEE	Journal	Factors Affecting On-Time Delivery in Large-Scale Agile Software Development	Elvan Kula , Eric Greuter, Arie van Deursen, Georgios Gousios	2022	5	2,03	—	10	10
S73	Factores	ACM	Proceedings	Factors Affecting Software Development Productivity: An empirical study	Edna Dias Canedo, Giovanni Almeida Santos	2019	31	0,152	—	10	10
S74	Factores	Sicence Direct	Journal	Factors affecting Agile adoption: An industry research study of the mobile app sector in Saudi Arabia	Fahad S.Altuwaijri, Maria Angela Ferrario	2022	1	1,42	—	10	10
S75	Factores	Springer	Conferencia	Factors Affecting Task Allocation and Coordination in Distributed Agile Software Development	Chitra Nundlall & Soulakshmee D. Nagowah	2021	1	—	—	10	2

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S76	Factores	Springer	Proceedings	Stress in Agile Software Development: Practices and Outcomes	Andreas Meier, Martin Kropp, Craig Anslow & Robert Biddle	2018	25	—	—	10	2
S77	Prácticas ágiles	IEEE	Conferencia	Software Practices For Agile Developers: A Systematic Literature Review	Devina Arnyndiasari , Ridi Ferdiana , Paulus Insap Santosa	2022	—	—	—	5	2
S78	Prácticas ágiles	IEEE	Journal	Agile Practices and Organizational Agility in Software Ecosystems	Paolo spagnoletti, Niloofar Kazemargi, Andrea Prencipe	2022	7	1,42	—	10	10
S79	Prácticas ágiles	IEEE	Conferencia	A survey on agile practices in the Indian IT industry	Noshiba Nazir, Nitasha prisa , Abhay-bansal	2016	16	0,228	—	10	10
S80	Prácticas ágiles	IEEE	Conferencia	Agile Development Practices Applied to Software Startups : A Systematic Mapping Review	Paulo César Abrantes , Ana Paula Furtado	2021	1	0,146	—	10	10
S81	Prácticas ágiles	IEEE	Conferencia	A review of agile practices for project management	Vira Liubchenko	2016	19	0,166	C	10	10
S82	Factores	IEEE	Conferencia	Attitudes, Beliefs, and Development Data Concerning Agile Software Development Practices	Christoph Matthies,Johannes Huegle, Tobias Dürschmid, Ralf Teusner	2019	17	0,001	—	10	10
S83	Método/Modelo/Enfoques	IEEE	Conferencia	AgileSafe - a method of introducing agile practices into safety-critical software development processes	Katarzyna Łukasiewicz, Janusz Gorski	2016	11	0,226	B	10	10

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S84	Prácticas ágiles	IEEE	Journal	Distributed Agile Patterns-Using Agile Practices to Solve Offshore Development Issues	Maryam Kausar, M. Ishtiaq, Shariq Hussain	2022	17	0,927	—	10	10
S85	Método/Modelo/Enfoques	IEEE	Journal	AgileRoadmap: An Approach to Implement Agile Practices in Teams	Patricio Letelier, Ma Carmen Penades	2017	5	0,33	—	10	10
S86	Prácticas ágiles	IEEE	Conferencia	Applying agile practices to avoid chaos in User Acceptance Testing: A case study	K.V. Jeeva Padmini, Indika Perera, H. M. N. Dilum Bandara	2016	11	0,152	—	10	10
S87	Prácticas ágiles	Springer	Conferencia	Agile Practices in Practice: Towards a Theory of Agile Adoption and Process Evolution	Brendan Julian, James Noble & Craig Anslow	2019	23	—	—	10	2
S88	Prácticas ágiles	Springer	Conferencia	Agile Practices, Collaboration and Experience	Martin Kropp, Andreas Meier & Robert Biddle	2016	11	—	B	10	5
S89	Prácticas ágiles	ACM	Proceedings	Implementing agile practices: the experience of TSol	María Cecilia Bastarrica, Germán Espinoza, Jacqueline Marín	2018	11	0,44	A	10	10
S90	Prácticas ágiles	ACM	Proceedings	Investigating Agile Practices in Software Startups	Renata Souza, Larissa Rocha, Franklin Silva, Ivan Machado	2019	14	0,152	—	10	10
S91	Prácticas ágiles	ACM	Proceedings	Adopting industry agile practices in large-scale capstone education	Jean-Guy Schneider, Peter W. Eklund, Kevin Lee, Feifei Chen, Andrew Cain, Mohamed Abdelrazek	2020	15	0,001	—	10	10

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S92	Prácticas ágiles	ACM	Proceedings	The Influence of Agile Practices on Performance in Software Engineering Teams: A Subgroup Perspective	Leonard Przybilla, Manuel Wiesche, Helmut Krcmar	2018	35	0,165	—	10	10
S93	Prácticas ágiles	Springer	Proceedings	Technical and Organizational Agile Practices: A Latin-American Survey	Nicolás Paez, Diego Fontdevila, Fernando Gainey & Alejandro Oliveros	2018	10	—	—	10	2
S94	Prácticas ágiles	Springer	Conferencia	Beneficial and Harmful Agile Practices for Product Quality	Sven Theobald & Philipp Diebold	2017	4	—	B	10	5
S95	Métricas	IEEE	Conferencia	“Agile Software Quality Metrics”	Michelle Maddox, Sheila Caminante	2021	1	—	—	10	2
S96	Métricas	IEEE	Conferencia	Agile metrics for a university software engineering course	Christoph Matthies, Thomas Kowark, Matthias Uflacker, Hasso Plattner	2016	26	0,221	C	10	10
S97	Métricas	Springer	Proceedings	A Study of Quality Metrics in Agile Software Development	Krishna Chakravarty & Jagannath Singh	2021	3	—	B	10	5
S98	Métricas	Springer	Journal	Metrics in Agile Projects	Joachim Rossberg	2016	1	—	—	10	2
S99	Prácticas ágiles	ACM	Proceedings	Satisfaction, Practices, and Influences in Agile Software Development	Martin Kropp, Andreas Meier, Craig Anslow, Robert Biddle	2018	37	—	A	10	10
S100	Métricas	ACM	Proceedings	Strong agile metrics: mining log data to determine predictive power of software	Hennie Huijgens, Robert Lamping, Dick Stevens, Hartger Rothengatter,	2017	19	0,001	A	10	10

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
				metrics for continuous delivery teams	Georgios Gousios, Daniele Romano						
S101	Métricas	ACM	Workshop	Agile and lean metrics associated with requirements engineering	Krzysztof Wnuk, Kalyan Chakravarthy Maddila	2017	8	0,151	C	10	10
S102	Métricas	Springer	Conferencia	Software Process Metrics in Agile Software Development: A Systematic Mapping Study	Syeda Sumbul Hossain, Pollab Ahmed & Yeasir Arafat	2021	2	0,164	C	10	10
S103	Métricas	Springer	Conferencia	Metrics for Measuring Progress and Productivity in Agile Software Development	Sanjay Misra, F. Martha Omorodion & Robertas Damasevicius	2021	4	0,111	—	10	10
S104	Métricas	IEEE	Conferencia	Agile and Quality: A Systematic Mapping Study	Samia Abdalhamid, Alfaroq O. M. Mohammed, Alok Mishra	2019	8	0.001	—	10	10
S105	Atributos	IEEE	Conferencia	Current state of the research in agile quality development	Parita Jain, Laxmi Ahuja, Arun Sharma	2016	22	0,165	—	10	10
S106	Atributos	IEEE	Conferencia	The Impact of Agile Software Development Process on the Quality of Software Product	Parita Jain , Arun Sharma , Laxmi Ahuja	2018	45	0,001	—	10	10
S107	Atributos	Science Direct	Journal	Usability in agile software development: A tertiary study	Karina Curcio, Rodolfo Santana, Sheila Reinehr, Andreia Malucelli	2019	53	0,87	—	10	10
S108	Atributos	Springer	Journal	Quality in Agile	Navid Nader-Rezvani	2019	1	—	—	10	2

No.	ASPECTO	BASE DE DATOS	TIPO	TITULO	AUTOR	AÑO	No. citas	JSR	CORE	CALIDAD 1 CITAS	CALIDAD 2 RELEVANCIA
S109	Atributos	Springer	Journal	Agile Quality Test Strategy	Navid Nader-Rezvani	2019	2	—	—	10	2
S110	Atributos	ACM	Conferencia	Documentation of Quality Requirements in Agile Software Development	Woubshet Behutiye, Pertti Seppänen, Pilar Rodríguez, Markku Oivo	2020	13	—	—	10	2
S111	Métricas	Science Direct	Journal	Quality measurement in agile and rapid software development: A systematic mapping	Lidia López, Xavier Burgués, Silverio Martínez Fernandez, Maria Vollmerb , Woubshet Behutiye, Pertti Karhapaa, Xavier Franch, Pilar Rodriguez, Markku Oivo	2022	4	1,42	—	10	10
EVALUACIÓN DE LA CALIDAD DE LOS PAPERS										9,8198	8,4685
										98,20 %	84,67 %

Anexo D. Lista de Factores Críticos de Éxito encontrados en la RSL

Dimensión	No	Factor	Fuente	Frecuencia
Organizacional	1	Fuerte apoyo ejecutivo	[S2],[S3], [S50], [S72]	4
	2	Patrocinador o gerente comprometido	[S2]	1
	3	Cultura organizacional cooperativa en lugar de jerárquica	[S2],[S74]	2
	4	Cultura oral que valora mucho la comunicación cara a cara	[S2]	1
	5	Comunicación y negociación	[S6]	1
	6	Comunicación con los clientes	[S4],[S9], [S75]	3
	7	Configuración del entorno	[S4]	1
	8	Organizaciones donde la metodología ágil es universalmente aceptada	[S2], [S72]	2
	9	Ubicación de todo el equipo	[S2],[S10]	2
	10	Adecuada instalación de un ambiente de trabajo estilo ágil	[S2], [S3], [S74]	3
	11	Sistema de recompensas adecuado para ágil	[S2]	1
	12	Apoyo a la innovación	[S7], [S8]	2
	13	Relación con socios externos	[S3],[S4]	2
	14	Ventaja relativa	[S11]	1
	15	Rotación del personal	[S3],[S50]	2
	16	Estabilidad del equipo	[S72]	1
	17	Rotación de empleados en el sector de desarrollo	[S50]	1
	18	Compatibilidad	[S11]	1
	19	Satisfacción del cliente	[S6]	1
	20	Colaboración del cliente	[S6]	1
	21	Compromiso del cliente	[S6]	1
	22	Frecuencia con la que el usuario solicita cambios en el sistema	[S50]	1
	23	Cultura corporativa	[S4],[S6], [S74]	3
	24	Madurez	[S3]	1
	25	Tamaño	[S3]	1
	26	Estabilidad	[S3]	1
	27	Estructura organizativa	[S3],[S11]	2
	28	Tiempo de decisión	[S6]	1
	29	Acuerdos del pago	[S3]	1
	30	Herramientas de apoyo	[S11], [S75]	2
	31	Experiencia de la empresa	[S14]	1
	32	Presupuesto suficiente	[S50]	1
	33	Infraestructura suficiente	[S72]	1
	34	Número de empleados trabajando en el sector de desarrollo	[S50]	1
	35	Afrontar el estrés	[S76]	1
	36	Tiempo de trabajo	[S75]	1
	37	Sitio de trabajo	[S75]	1
	38	Ambiente organizacional	[S74]	1
	39	Política organizacional	[S72]	1
	40	Costo laboral	[S75]	1
	41	Posición de los directores en el organigrama corporativo	[S50]	1
Personas	42	Miembros del equipo con gran motivación	[S2],[S3],[S6],[S10],[S50], [S14], [S75]	6
	43	Experiencia en desarrollo	[S14], [S74],[S75]	1
	44	Tamaño del equipo	[S6],[S14]	2
	45	Distribución del equipo	[S6], [S75], [S72]	1
	46	Comunicación clara con todo el	[S108],[S74], [S72]	3

Dimensión	No	Factor	Fuente	Frecuencia
		equipo de trabajo		
	47	Mentalidad ágil	[S11]	1
	48	Entrenador ágil	[S11], [S74]	1
	49	Gerentes conocedores de procesos ágiles	[S1],[S2], [S75]	2
	50	Apoyo de la gerencia	[S11]	1
	51	Campeón de metodología	[S11]	1
	52	Gerentes que tienen un estilo de gestión ligero o adaptable	[S2], [S3]	2
	53	Trabajo en equipo coherente y auto-organizado	[S2], [S3], [S5]	3
	54	Buena relación con el cliente	[S1],[S2], [S74]	2
	55	Personalidad/Características personales	[S5],[S6],[S7]	3
	56	Interdependencia y autonomía	[S5]	1
	57	Experiencia del gerente	[S10], [S50]	2
	58	Moral/Etica	[S9],[S10]	2
	59	Seguridad participativa	[S7],[S8]	2
	60	Visión y estrategia	[S7],[S8]	2
	61	Metas y Objetivos	[S7],[S8]	2
	62	Cultura	[S3],[S6],[S9], [S75]	3
	63	Coordinación de las tareas	[S75]	1
	64	Compromiso	[S3], [S72]	1
	65	Entrenamiento y aprendizaje	[S6], [S75]	1
	66	No al cambio continuo de los gerentes del proyecto	[S3]	1
	67	Neuroticismo	[S7]	1
	68	Extraversión	[S7]	1
	69	Abierto a la experiencia	[S7]	1
	70	Agradable	[S7]	1
	71	Conciencia	[S7]	1
	72	Orientación de la tarea	[S7]	1
	73	Profesionalismo	[S13]	1
	74	Competencia del usuario	[S50]	1
	75	Entrenamiento del usuario	[S50]	1
	76	Conocimiento del usuario	[S50]	1
	77	Involucramiento del usuario	[S50]	1
	78	Nivel de habilidad del equipo	[S50]	1
	79	Conocimiento del equipo	[S14], [S75]	1
	80	Soporte en la gestión	[S74]	1
	81	Esfuerzo	[S75]	1
	82	Personal disponible	[S75]	1
	83	Transparencia	[S75]	1
	84	Madurez del equipo y la gerencia	[S75]	1
	85	Familiaridad del equipo	[S72]	1
	86	Fluencia en el lenguaje	[S75]	1
	87	Habilidad y capacidad del equipo de trabajo	[S50], [S74], [S75], [S72]	4
Proceso	88	Seguir un proceso de gestión de proyectos ágil	[S2]	1
	89	Seguir un proceso de gestión de la configuración orientado a ágil	[S2]	1
	90	Seguir un proceso de gestión de requisitos orientado a ágil	[S2]	1
	91	Cumplir con el horario de trabajo regular, sin horas extras	[S2],[S10]	2
	92	Fuerte enfoque de comunicación con reuniones diarias cara a cara	[S2]	1
	93	Fuerte compromiso y presencia del cliente	[S2],[S3], [S75], [S72]	2
	94	Mejora continua	[S104]	1

Dimensión	No	Factor	Fuente	Frecuencia
Técnico	95	Fechas de entrega de las tareas	[S75]	1
	96	Trabajo en progreso	[S72]	1
	97	Cliente tiene plena actividad	[S2]	1
	98	Estándares de codificación bien definidos por adelantado	[S2],[S3]	2
	99	Diseño simple	[S1],[S2]	2
	100	Tamaño de las tareas	[S72]	1
	101	Actividades rigurosas de refactorización	[S2]	1
	102	Cantidad correcta de documentación	[S2]	1
	103	Entrega regular de software	[S1],[S2], [S72]	2
	104	Entrega de características más importantes primero	[S2], [S75],[S72]	1
	105	Correcta prueba de integración	[S2]	1
	106	Documentación suficiente	[S75]	1
	107	Rendimiento	[S3]	1
	108	Reusabilidad	[S3]	1
	109	Perfil de implementación	[S3]	1
	110	Tipo, tamaño y conectividad de la aplicación	[S1],[S3]	2
	111	Diseño modular	[S1],[S14]	2
	112	Diseño portable	[S1]	1
	113	Diseño extensible	[S1]	1
	114	Arquitectura del producto	[S75]	1
	115	Entrenamiento técnico apropiado para el equipo	[S2],[S3],[S10]	3
	116	Idoneidad del sistema de gestión de bases de datos empleado	[S50]	1
	117	Idoneidad del método de desarrollo empleado en la organización	[S50]	1
	118	Soporte de los departamentos conectados a TI	[S50]	1
	119	Tipo de metodología de de desarrollo del sistema empleada	[S50]	1
	120	Tipo de lenguaje de programación empleado	[S50]	1
	121	Tipo de modelo utilizado por el sistema de gestión de bases de datos	[S50]	1
	122	Calidad de los requerimientos	[S110]	1
	123	Calidad de la documentación generada	[S50]	1
Proyecto	124	La naturaleza del proyecto no es crítica para la vida	[S2],[S3]	2
	125	Tipo de proyecto de alcance variable con requerimientos emergentes	[S2], [S3]	2
	126	Proyecto con cronograma dinámico y acelerado	[S2],[S3]	2
	127	Proyectos con pequeños equipos	[S1],[S2],[S3]	3
	128	Proyectos sin múltiples equipos independientes	[S2]	1
	129	Proyectos con evaluación inicial de costos realizada	[S2]	1
	130	Tareas dependientes	[S72]	1
	131	Tamaño del proyecto	[S72]	1
	132	Seguridad del proyecto	[S72]	1
	133	Pruebas suficientes	[S72]	1
	134	Documentación del código	[S72]	1

Dimensión	No	Factor	Fuente	Frecuencia
	135	Calidad del código	[S72]	1
	136	Planificación	[S6],[S71]	2
	137	Control	[S6]	1
	138	Proyectos con análisis de riesgos iniciales realizado	[S2],[S3]	2
Desarrollo Iterativo	139	Lanzamiento de timeboxes	[S1]	1
	140	Lanzamientos operacionales	[S1]	1
	141	Pequeñas versiones	[S1]	1
	142	Entrega a tiempo	[S97],	1
	143	Numerosas versiones	[S1]	1
Retroalimentación del cliente	144	Retroalimentación solicitada	[S1]	1
	145	Retroalimentación recibida	[S1]	1
	146	Frecuencia de retroalimentación	[S1]	1
	147	Calidad de la retroalimentación	[S1]	1
	148	Retroalimentación incorporada	[S1]	1
Total				206

Anexo E. Lista de Métricas encontradas en la RSL

Categoría	No.	Métrica	Fuente	Frecuencia
Métricas de Uso	1	Porcentaje de esfuerzo en programación en pareja	[S30]	1
	2	Esfuerzo gastado en las fases SDLC	[S48]	1
	3	Frecuencia de cambio de pareja	[S22]	1
	4	Productividad entre iteraciones: pareja y solo	[S30]	1
	5	Justificación de la programación en pareja	[S30]	1
	6	Defectos enviados por los clientes	[S31]	1
	7	Número de errores enviados por los evaluadores	[S98]	1
	8	Cambios solicitados por los clientes	[S31],[S101]	2
	9	Esfuerzo de mantenimiento	[S31]	1
	10	Cantidad de ingresos de un cliente	[S31]	1
	11	Porcentaje de obra adaptada o cambiada	[S97],[S95]	2
	12	Cantidad de solicitudes rápidas de extracción	[S82]	1
	13	Valor comercial ganado	[S103]	1
	14	Índice de mantenibilidad	[S98]	1
	15	Recuentos de defectos diferidos	[S31]	1
Métricas de Calidad	16	Densidad de las desviaciones de los estándares de codificación: par y solo	[S30]	1
	17	Relación de comentarios: par y solo	[S30]	1
	18	Densidad relativa del defecto: par y solo	[S30]	1
	19	Costo de calidad de datos	[S102]	1
	20	Costo de la calidad funcional	[S102]	1
	21	Madurez del equipo	[S16]	1
Métricas del Proyecto	22	Longitud de la liberación	[S25]	1
	23	Longitud de la liberación	[S25]	1
	24	Número de iteraciones	[S17],[S19]	2
	25	Esfuerzo gastado en el desarrollo	[S17]	1
	26	Frecuencia de emparejamiento	[S25]	1
	27	Frecuencia de inspección	[S25]	1
	28	Porcentaje de items añadidos al product backlog	[S24]	1
	29	Porcentaje de items cambiados en el product backlog	[S24]	1
	30	Porcentaje de items del backlog rechazados	[S24]	1
	31	Cambio de estimaciones	[S24]	1
	32	Cambio de prioridades	[S24]	1
	33	Pronóstico de tiempo	[S24],[S15],[S102]	3
	34	Tamaño de las características	[S31]	1
	35	Término del lanzamiento	[S23]	1
	36	Término del sprint	[S23]	1
	37	Número de instrucciones declaradas	[S29]	1
	38	Puntos de función	[S48],[S101]	2
	39	Puntos de historias	[S19]	1
	40	Puntos de casos de uso	[S48]	1
	41	Número de historias	[S19],[S101]	2
	42	Cantidad diaria de historias	[S82]	1
	43	Porcentaje de historias completas	[S31]	1
	44	Tamaño del proyecto	[S24]	1
	45	Tamaño de la historia de usuario	[S48],[S82]	2

	46	Tamaño restante del proyecto	[S24]	1
	47	SLOC Líneas de código fuente	[S11],[S17],[S28],[S29], [S48],[S98]	6
	48	Porcentaje de precisión en líneas de código	[S103]	1
	49	Tamaño del archive en código fuente en kilobytes	[S22]	1
	50	Número de referencias del archivo de código fuente a elementos de archivos externos	[S22]	1
	51	Número de referencias del archivo de código fuente a elementos de otros archivos.	[S22]	1
	52	Número de veces que se registra ese archivo	[S22]	1
	53	Número de tareas diferentes de JIRA y tickets de error para los que se modificó el archivo	[S22]	1
	54	Número de clases	[S27],[S29]	2
	55	Tamaño de la clase	[S27]	1
	56	Número de nombres de partes de un método (NNP)	[S26],[S32]	2
	57	Número de caracteres	[S26],[S32]	2
	58	Número de líneas de comentario (CL)	[S26],[S29],[S32]	3
	59	Número de variables locales (NLV)	[S26],[S29],[S32]	3
	60	Número de objetos creados (NCO)	[S26],[S32]	2
	61	Número de objetos referenciados (NRO)	[S26],[S32]	2
	62	Tiempo de ejecución	[S21]	1
	63	Cantidad de trabajos pendientes	[S82]	1
	64	Progreso real de una aplicación	[S103]	1
	65	Trabajo pendiente liberado	[S98]	1
	66	Variación de esfuerzo	[S97]	1
	67	Cartera de pedidos	[S98]	1
	68	Trabajos duplicados	[S82],[S82]	2
	69	Trabajo no planificado	[S98]	1
	70	Utilización de recursos	[S16]	1
Métricas de Testing	71	Prueba de cobertura	[S11],[S16],[S25]	3
	72	Cobertura de código	[S11],[S48],[S98]	3
	73	Prueba de frecuencia de ejecución	[S21],[S25]	2
	74	Prueba a clases nuevas / modificadas	[S25]	1
	75	Nuevas clases con su correspondiente prueba	[S25]	1
	76	LOC probadas/ LOC fuente	[S25]	1
	77	Densidad de defectos de pre-lanzamiento	[S19],[S31]	2
	78	Total de defectos	[S16],[S17],[S31],[S70],[S97]	5
	79	Eficiencia en la eliminación de defectos	[S70]	1
	80	Número de casos de prueba	[S11],[S17],[S27],[S101]	4
	81	Número de aseveraciones	[S27]	1
	82	Pruebas de ejecución de características (RTF)	[S32]	1
	83	Frecuencia de cambio de errores	[S22]	1
	84	Cobertura de errores	[S31]	1
	85	Número de errores	[S11]	1
	86	Eficacia de contención de defectos	[S70]	1
	87	Número de defectos de pre-lanzamiento	[S70]	1
	88	Número de defectos post-lanzamiento	[S70]	1
	89	Tasa de ocurrencia de anomalías	[S51]	1

	90	Tasa de ocurrencia de fallos	[S51],[S102]	2
	91	Tendencia de derrame de defectos	[S97]	1
	92	Ejecución de funciones probadas	[S97]	1
	92	Advertencias de compilador	[S98]	1
	94	Advertencias de análisis de código	[S98]	1
	95	Complejidad ciclomática	[S98]	1
	96	Porcentaje de requisitos cubiertos por casos de prueba	[S98], [S102]	2
	97	Eficiencia de la prueba	[S95]	1
	98	Eficiencia en la eliminación de errores	[S95],[S102]	2
	99	Coefficiente de similitud de Jaccard	[S51]	1
Métricas CK (Chidamber and Kemerer)	100	Métodos ponderados por clase (WMC)	[S18],[S26],[S27],[S28], [S32]	5
	101	Acoplamiento entre objetos (CBO)	[S18],[S26],[S27],[S28], [S32], [S98]	6
	102	Profundidad del árbol de herencia (DIT)	[S18],[S26],[S27],[S28], [S29], [S32],[S98]	7
	103	Número de hijos (NOC)	[S18],[S26],[S27], [S28],[S29],[S32]	6
	104	Respuesta para una clase (RFC)	[S18],[S26],[S27], [S28],[S29],[S32]	6
	105	Falta de cohesión de métodos (LCOM)	[S18],[S26],[S27],[S28], [S32]	5
MOOD Metric Suit	106	Factor de ocultación de atributos (AHF)	[S18]	1
	107	Factor de ocultación de métodos (MHF)	[S18]	1
	108	Factor de herencia de atributos (AIF)	[S18]	1
	109	Factor de herencia de métodos (MIF)	[S18]	1
QMOOD Metric Suite	110	Promedio del número de antecesores (ANA)	[S18],[S20]	2
	111	Cohesión entre métodos (CAM)	[S18],[S20]	2
	112	Tamaño de interfaz de clases (CIS)	[S18],[S20]	2
	113	Acoplamiento directo de clases (DCC)	[S18],[S20],[S98]	3
	114	Métricas de acceso a datos (DAM)	[S18],[S20]	2
	115	Medida de agregación (MOA)	[S18],[S20]	2
	116	Número de métodos (NOM)	[S18],[S20],[S29]	3
	117	Medida de abstracción funcional (MFA)	[S18],[S20]	2
	118	Tamaño del diseño en clases (DSC)	[S18],[S20]	2
	119	Número de jerarquías (NOH)	[S20],[S98]	1
Team's effectiveness metrics	120	Número de polimorfismo (NOP)	[S20]	1
	121	Longitud verdadera del sprint	[S15]	1
	122	Velocidad	[S15],[S19],[S31],[S97], [S98], [S101],[S102]	7
	123	Progreso porcentual	[S103]	1
	124	Eficiencia de eliminación de defectos	[S97]	1
	125	Desviación de la velocidad	[S15]	1
	126	Precisión de la estimación	[S102]	1
	127	Entrega a tiempo	[S97],[S102]	2
	128	Trabajo restante	[S98]	1
	129	Métricas de productividad	[S97],[S103]	2
	130	Tiempo de espera	[S15]	1
	131	Tiempo de entrega	[S97]	1
	132	Tiempo de ciclo	[S97],[S103],[S102]	3
	133	Evaluación del equipo	[S102]	1
	134	Liberación de trabajo pendiente	[S98]	1
	135	Número de requisitos a liberar	[S101],[S95]	2
	136	Complejidad de los requisitos	[S101]	1
	137	Impacto futuro	[S15]	1
Total				227

Anexo F. Lista de Prácticas ágiles encontradas en la RSL

Categoría	No.	Prácticas Ágiles	Fuente	Frecuencia
Prácticas de Equipo de Trabajo	1	Equipos pequeños	[S40],[S43],[S45],[S81], [S85]	5
	2	Equipos multifuncionales	[S45],[S49], [S94], [S85]	4
	3	Equipos múltiples	[S43], [S78]	2
	4	Reuniones diarias	[S34],[S38],[S41],[S43],[S45], [S44], [S49], [S89], [S92],[S80], [S85]	11
	5	Reuniones de objetivos	[S34],[S38],[S71]	3
	6	Reunión de pie	[S35],[S38],[S42],[S44],[S45], [S108]	6
	7	Cliente on-site	[S38],[S39],[S42],[S45],[S48], [S94], [S85]	7
	8	Programación en pareja	[S35],[S39],[S42],[S44],[S47], [S89], [S92], [S93], [S94], [S80]	10
	9	Agilidad en la comunicación del equipo	[S4], [S38],[S39],[S42],[S47], [S90], [S85]	7
	10	Capacidades de comunicación de las personas	[S4],[S36],[S71]	3
	11	Capacidades de comunicación del equipo	[S4],[S45],[S49],[S71], [S102]	5
	12	Competencia de las personas	[S4],[S45],[S46],[S71], [S78]	5
	13	Superación personal del equipo	[S78]	1
	14	Confianza entre los miembros del equipo	[S4]	1
	15	Participación de los líderes	[S4], [S71], [S85]	3
	16	Reconocimiento del líder	[S4], [S36],[S38], [S85]	4
	17	Comunicación diaria entre empresarios y desarrolladores	[S35],[S38],[S45], [S102]	4
	18	Frecuencia de iteraciones con los clientes	[S4], [S36],[S40],[S41],[S47], [S49], [S52]	7
	19	Motivación de los clientes	[S4]	1
	20	Responsabilidad y autoridad	[S38]	1
	21	Trabajos por turnos	[S44]	1
	22	Equipos auto-organizados	[S40],[S42],[S45],[S71],[S85]	5
	23	Formación de profesionales	[S38], [S40], [S45],[S48]	4
	24	Satisfacción del cliente con los proyectos	[S4], [S45]	2
	25	Interacción con socios en el desarrollo de proyectos	[S4], [S45]	2
	26	Interacción con socios externos	[S4], [S45]	2
	27	Apertura organizacional para hablar	[S4]	1
	28	Agilidad en las decisiones de diseño	[S4], [S42]	2
	29	Autonomía del equipo en proyectos	[S4], [S93]	2
	30	Líder de procesos QA	[S85]	1
	31	Equipos con múltiple ubicación	[S43]	1
	32	Entrenamiento del cliente	[S4]	1
	33	Transferencia del conocimiento	[S38],[S44],[S45],[S49]	4
	34	Semana de 40-horas	[S39], [S85]	2
	35	Ciclo de aprendizaje	[S49]	1
	36	Whiteboard	[S34]	1
	37	Revisión de pares	[S34]	1
	38	Involucrar al cliente	[S38],[S48],[S71]	3
	39	Trabajo en paralelo	[S82]	1
	40	Habilidad del equipo	[S71], [S85]	2
	41	Organización dinámica	[S46], [S84]	2
	42	Distribución del trabajo	[S82]	1
	43	Co-localización del equipo	[S85]	1
	44	Colaboración para solucionar problemas	[S78]	1

Categoría	No.	Prácticas Ágiles	Fuente	Frecuencia
Prácticas de ingeniería	45	Valores ágiles	[S82]	1
	46	Preparación de pedidos atrasados	[S94]	1
	47	Ritmo sostenible	[S84]	1
	48	Trabajo asignado	[S46]	1
	49	Cero deudas técnicas	[S44]	1
	50	Visión del producto	[S38], [S49]	2
	51	Casos de uso	[S48]	1
	52	Propiedad del código	[S82], [S84], [S89], [S93]	4
	53	Roadmap	[S46]	1
	54	Diagramas UML	[S48]	1
	55	Diagramas de flujo de datos	[S48]	1
	56	Sprints backlog	[S35],[S38],[S45],[S46],[S13]	5
	57	Historia de usuario	[S34],[S42],[S43],[S44], [S48], [S71] [S82],[S80]	8
	58	Tema / épica	[S71]	1
	59	Puntos de historia	[S45]	1
	60	Metáforas	[S39]	1
	61	Tarjetas CRC	[S37], [S44]	2
	62	Tarjetas de historia QA	[S37]	1
	63	Escenarios	[S41],[S52]	2
	64	Estándares de codificación	[S36],[S39],[S42],[S44],[S47], [S89]	6
	65	Diseño simple	[S34],[S35],[S36],[S39], [S40], [S44], [S47],[S89], [S90], [S85]	10
	66	Diseño emergente	[S93]	1
	67	Seguir un diseño estricto	[S48]	1
	68	Flexibilidad a los cambios	[S4],[S40], [S42], [S47]	4
	69	Prototipo	[S46],[S48]	2
	70	Ciclo de lanzamiento	[S46]	1
	71	Configuración del ambiente	[S4]	1
	72	Requerimientos capturados en alto nivel	[S42]	1
	73	Control de calidad	[S49]	1
	74	Análisis continuo de especificaciones	[S49]	1
	75	Alcance dinámico	[S78]	1
	76	Dirigido por características	[S34]	1
	77	Iteración del backlog	[S34], [S85]	2
	78	Iteración del sprint	[S45]	1
	79	Repositorios de arquitectura	[S78]	1
	80	Compilaciones frecuentes	[S81]	1
	81	Desarrollo sostenido e incremental	[S84], [S85]	2
	82	Implementación continua	[S80]	1
	83	Procesos interactivos	[S93]	1
	84	Paquetes de alto acoplamiento	[S81]	1
	85	Lanzamientos frecuentes	[S89], [S90], [S93]	3
	86	Gestionar convenientemente el re-trabajo	[S85]	1
	87	Aplicación de estándares para todo el equipo	[S85]	1
	88	Evitar invertir esfuerzo en adelantar trabajo que no esté comprometido y/o no esté cercano a su entrega	[S85]	1
	89	Product backlog	[S38],[S45], [S84], [S89], [S80]	5
Prácticas de documentación	90	Documentación reducida	[S40], [S85], [S110]	3
	91	Plan del proyecto	[S45]	1
	92	Informes periódicos del progreso	[S48]	1
	93	Documentación del sprint	[S47]	1
	94	Revisión de documentos	[S47]	1
	95	Diseño de documentos	[S47]	1
	96	Documento del caso del negocio	[S47]	1
	97	Documentación del proyecto	[S47]	1

Categoría	No.	Prácticas Ágiles	Fuente	Frecuencia
	98	Utilidad de la documentación para el equipo	[S4]	1
	99	Utilidad de la documentación para el cliente	[S4]	1
	100	Generación automática de documentación	[S44]	1
	101	Informe de factibilidad	[S47]	1
	102	Documentos wiki	[S34]	1
	103	Requerimientos documentados	[S48]	1
	104	Documentación constante	[S81]	1
	105	Documentos de visión con consideraciones de arquitectura	[S46]	1
Prácticas de gestión	106	Estudio de objetivos del negocio	[S42]	1
	107	Propiedad colectiva del código	[S39], [S82]	2
	108	Burndown charts	[S34],[S38],[S42],[S44], [S45], [S97], [S89], [S94],[S98]	9
	109	Kanban	[S35], [S71], [S89], [S94], [S85]	5
	110	Planificación	[S34],[S39],[S41],[S42], [S45], [S84], [S89], [S108]	8
	111	Evaluación de requerimientos del negocio	[S71]	1
	112	Velocidad del equipo	[S34],[S42],[S44],[S45], [S71], [S94]	6
	113	Estimación temprana	[S47], [S85]	2
	114	Reunión de planificación de la iteración	[S35],[S37],[S38], [S41],[S42],[S43], [S44],[S45],[S47], [S85]	10
	115	Reunión de planificación del sprint	[S71], [S84], [S89]	3
	116	Revisión del progreso del sprint/interacciones	[S71], [S84], [S94]	3
	117	Reuniones de hitos	[S71]	1
	118	Reuniones mensuales	[S71]	1
	119	Revisión de resultados	[S38],[S42], [S85]	3
	120	Retropectiva / Retroalimentación	[S34],[S35],[S38],[S39],[S42],[S44], [S45], [S46],[S47],[S52], [S89], [S92] [S93], [S94], [S78], [S80], [S85], [S108]	18
	121	Revisión de requerimiento con los desarrolladores y propietario del producto	[S35]	1
	122	Gestión de riesgos	[S45], [S47],[S71]	3
	123	Revisión del código	[S47], [S82], [S80]	3
	124	Alcance del proyecto	[S4],[S45]	2
	125	Calidad del proyecto	[S4]	1
	126	Plazo de los proyectos	[S4]	1
	127	Planificación del lanzamiento	[S37],[S42],[S43],[S44], [S45], [S94], [S46]	6
	128	Pequeñas entregas	[S35],[S39],[S41],[S47]	4
	129	Entrega incremental	[S47],[S49], [S85]	2
	130	Control de versiones	[S38],[S47], [S82], [S90]	4
	131	Priorización de requerimientos	[S42],[S45], [S47],[S52]	4
	132	Reducción de riesgos con socios externos	[S4]	1
	133	Time boxing	[S34],[S49]	2
	134	Monitoreo del progreso	[S34],[S38],[S45],[S46],[S48],[S49],[S71]	7
	135	Gestión de la configuración	[S38]	1
	136	Controles interactivos	[S81], [S85]	2
	137	Gestión integrada a nivel de equipo y de cada miembro	[S85]	1
	138	Definir que se entiende por trabajo terminado	[S85]	1
	139	Capacidad de declarar el proyecto	[S81]	1

Categoría	No.	Prácticas Ágiles	Fuente	Frecuencia
		en cualquier momento		
	140	Gestión continua del trabajo pendiente	[S85]	1
	141	Revisión semanal	[S71]	1
	142	Trazabilidad a las pruebas, código y tareas	[S34]	1
Prácticas de Testing	143	Pruebas tempranas y frecuentes	[S39],[S42], [S52]	3
	144	Integración continua	[S35],[S44],[S47], [S49],[S52], [S84], [S90], [S93], [S80]	9
	145	Refactoring	[S39],[S42], [S40],[S47],[S49],[S52], [S89], [S94], [S80]	9
	146	Planificación de pruebas	[S82]	1
	147	Detección de errores	[S44]	1
	148	Prueba de pareja	[S48]	1
	149	Prueba funcional	[S35],[S37]	2
	150	Prueba del producto	[S71]	1
	151	Pruebas del sistema	[S39],[S47],[S71]	3
	152	Pruebas de integración	[S47]	1
	153	Desarrollo dirigido por pruebas	[S34],[S37],[S46],[S48],[S47],[S52], [S89], [S93], [S94], [S77]	10
	154	Desarrollo guiado por componentes	[S80]	1
	155	Automatización de pruebas	[S82], [S93], [S85]	3
	156	Pruebas del último desarrollo	[S48]	1
	157	Pruebas unitarias	[S36],[S37],[S39],[S44],[S47], [S71], [S94], [S80]	8
	158	Pruebas de componentes	[S71]	1
	159	Pruebas de calificación construidas	[S71]	1
	160	Prueba del diseño	[S48]	1
	161	Pruebas de aceptación	[S42],[S41],[S47], [S71],[S52], [S80]	6
	162	Validación	[S48], [S49]	2
	163	Auditorías	[S42],[S48]	2
	164	Priorización de errores	[S37]	1
	165	Manejo de defectos	[S48]	1
Total				432

Anexo G. Lista de características de calidad del software encontradas en la RSL

Carácterística	No.	Subcarácterística	Estándar	Fuente	Frecuencia
Funcionalidad	1	Complejidad funcional	ISO/IEC 25010		0
	2	Correctitud funcional	ISO/IEC 25010	[S51],[S13]	2
	3	Adecuación funcional	ISO/IEC 25010		0
Fiabilidad	4	Madurez	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	5	Disponibilidad	ISO/IEC 25010	[S68]	1
	6	Tolerancia a fallas	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	7	Recuperabilidad	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	8	Conformidad	ISO/IEC 9126		0
Compatibilidad	9	Coexistencia	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	10	Interoperabilidad	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	11	Reconocibilidad de idoneidad	ISO/IEC 25010		0
	12	Comprensibilidad	ISO/IEC 9126	[S20],[S67],[S69]	3
	13	Capacidad de aprendizaje	ISO/IEC 25010	[S67],[S69]	2
			ISO/IEC 9126		
	14	Operabilidad	ISO/IEC 25010		0
	15	Protección de errores de usuario	ISO/IEC 25010		0
	16	Interface de usuario estética	ISO/IEC 25010		0
	17	Accesabilidad	ISO/IEC 25010		0
	18	Operabilidad	ISO/IEC 9126	[S67]	1
	19	Atractividad	ISO/IEC 9126	[S67]	1
	20	Conformidad	ISO/IEC 9126	[S67]	1
Rendimiento-Eficiencia	21	Comportamiento en el tiempo	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	22	Utilización de recursos	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	23	Conformidad	ISO/IEC 9126	[S67]	1
Seguridad	24	Confidencialidad	ISO/IEC 25010		0
	25	Integridad	ISO/IEC 25010	[S66],[S13]	2
	26	No-repudio	ISO/IEC 25010		0
	27	Responsabilidad	ISO/IEC 25010		0
	28	Autenticidad	ISO/IEC 25010		0
Mantenibilidad	29	Modularidad	ISO/IEC 25010	[S69]	1
	30	Reusabilidad	ISO/IEC 25010	[S20],[S66],[S69],[S13]	4
	31	Analizabilidad	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	32	Cambiabilidad	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	33	Modificación	ISO/IEC 25010	[S68]	1
	34	Estabilidad	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	35	Testabilidad	ISO/IEC 25010	[S67],[S68],[S13]	3
			ISO/IEC 9126		
	36	Conformidad	ISO/IEC 9126	[S67]	1
Portabilidad	37	Adaptabilidad	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	38	Instalabilidad	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	39	Reemplazabilidad	ISO/IEC 25010	[S67]	1
			ISO/IEC 9126		
	40	Conformidad	ISO/IEC 9126	[S67]	1
Funcionalidad	41	Idoneidad	ISO/IEC 9126	[S67]	1
	42	Exactitud	ISO/IEC 9126	[S67]	1
	43	Conformidad	ISO/IEC 9126	[S67]	1
	44	Seguridad	ISO/IEC 9126	[S67]	1
Portabilidad *	45			[S66],[S13],[S106]	3
Mantenibilidad*	46			[S66],[S13],[S106]	3
Usabilidad*	47			[S68],[S14],[S107]	3
Funcionalidad*	48			[S20],[S106]	2

Carácterística	No.	Subcarácterística	Estándar	Fuente	Frecuencia
Fiabilidad*	49			[S17],[S24],[S70], [S106]	4
Seguridad*	50			[S68]	1
Otros	51	Correctitud		[S66]	1
	52	Robustez		[S66],[S69]	2
	53	Extendibilidad		[S20],[S66],[S69], [S13]	4
	54	Compatibilidad		[S66]	1
	55	Eficiencia		[S66],[S13], [S106]	3
	56	Oportunidad		[S66]	1
	57	Verificación y Validación		[S66]	1
	58	Facilidad de uso		[S66]	1
	59	Costo-Beneficio		[S66],[S13]	2
	60	Simplicidad		[S69]	1
	61	Generalidad		[S69]	1
	62	Modularidad en tiempo de ejecución		[S69]	1
	63	Escalabilidad		[S69], [S106]	2
	64	Flexibilidad		[S20],[S13], [S106]	3
	65	Efectividad		[S11],[S20]	2
	66	Productividad		[S11], [S82]	2
	67	Factibilidad		[S24]	1
	68	Rendimiento		[S68], [S106]	2
	69	Utilidad		[S24]	1
	70	Reusabilidad		[S106]	1
	71	Complejidad		[S13], [S82]	2
Total					93

(*)Los autores analizaron las características sin considerar sus subcaracterísticas

Anexo H. Proceso de Selección de la hipótesis a probar. Relación factores críticos de éxito – Prácticas ágiles

DIMENSIÓN	ID	FACTOR	PTW EQUIPO DE TRABAJO								PMP GESTIÓN DEL PROYECTO					PENG INGENIERÍA					PTEST TESTING		
			Programación en pareja	Cliente On-site	Reuniones diarias	Equipos auto-organizados	Equipos multifuncionales	Retrospectivas	Propiedad colectiva del código	PROM	Planificación	Monitoreo del progreso	40 horas por semana	Time boxing	PROM	Pequeñas versiones	Diseño Simple	Refactoring	Estándares de Codificación	PROM	Pruebas Unitarias	Integración Continua	PROM
			P1	P2	P3	P4	P5	P6	P7		P8	P9	P10	P11		P12	P13	P14	P15		P16	P17	
Personal	FCO	Confianza	5	3	3	5	4	4	5	4,14	5	2	1	4	3,00	2	4	4	4	3,50	5	4	4,50
	FSE	Autoeficacia Percibida	5	2	4	4	4	4	1	3,43	3	1	1	3	2,00	2	3	2	3	2,50	5	4	4,50
	FSTR	STRESS	4	3	4	1	1	3	3	2,71	5	4	5	3	4,25	3	3	4	4	3,50	5	3	4,00
Organizacional	FIAI	Integridad y Disponibilidad	4	4	4	4	4	4	3	3,86	4	4	1	1	2,50	2	5	2	4	3,25	4	4	4,00
	F2.2	Calidad de Vida Laboral	5	1	3	5	4	3	2	3,29	2	2	5	4	3,25	1	4	4	4	3,25	4	3	3,50
	F2.3	Medios de comunicación usados	4	5	4	5	2	4	4	4,00	5	3	1	5	3,50	1	2	3	4	2,50	4	1	2,50

Anexo I. Proceso de Selección de la hipótesis a probar. Relación Prácticas ágiles – características de calidad

CATEG.	ID	PRÁCTICA	AFUN Funcionalidad				ARELI Fiabilidad					AUSAB Usabilidad						APEF Eficiencia				ACOMP Compatibilidad			AMAIN Mantenibilidad						APORT Portabilidad				ASEC Seguridad							
			A1	A2	A3	Prom	A4	A5	A6	A7	PROM	A8	A9	A10	A11	A12	A13	Prom	A14	A15	A16	Prom	A17	A18	Prom	A19	A20	A21	A22	A23	A24	Prom	A25	A26	A27	Prom	A28	A29	A30	A31	A32	Prom
PWT Equipo de Trabajo	P1	Programación en pareja	4	3	4		4	4	3	3		3	3	2	4	3	3		2	4	3		2	2		3	2	1	1	1	1		3	1	2		3	4	3	4	1	
	P2	Cliente On-site	4	3	4		4	3	3	3		5	5	5	3	3	3		3	3	2		2	2		4	4	4	4	4	4		4	5	2		4	2	2	4	3	
	P3	Reuniones diarias	2	2	2		2	4	2	2		2	3	2	3	2	3		2	1	1		1	1		2	1	1	1	1	1		1	1	1		2	2	1	3	1	
	P4	Equipos auto-organizados	5	4	4		4	2	3	3		5	5	4	3	2	3		4	5	1		1	1		4	4	3	4	3	4		3	3	2		2	2	2	3	3	
	P5	Equipos multifuncionales	4	3	3		4	2	4	4		3	2	4	3	3	3		4	4	1		3	3		2	3	3	4	3	4		2	3	2		2	3	3	3	3	
	P6	Retrospectivas	4	3	4		4	2	3	4		4	4	3	3	3	3		4	4	3		4	4		2	4	3	4	3	4		3	3	2		4	2	2	3	3	
	P7	Propiedad colectiva del código	4	4	4		4	2	3	3		2	4	3	3	3	4		2	3	3		3	3		4	4	2	2	2	2		1	1	2		3	3	2	4	3	
		PROM				3,57					3,14							3,24				2,81			2,29								2,83				2,24					
PMP Gestión del Proyecto	P8	Planificación	5	5	4		2	2	3	3		2	4	2	3	3	4		1	2	2		3	4		4	4	2	3	2	3		3	4	2		2	2	2	4	3	
	P9	Monitoreo del progreso	5	5	4		4	2	4	4		2	2	4	2	3	4		3	4	3		3	3		3	3	3	3	3	3		2	3	2		3	2	2	4	3	
	P10	40 horas por semana	2	2	2		2	2	2	2		2	1	2	2	2	1		2	2	2		2	1		1	2	1	1	1	1		1	1	1		1	2	1	4	1	
	P11	Time boxing	4	4	3		2	2	1	1		4	4	4	5	1	2		2	2	3		2	2		2	4	1	1	1	1		3	3	1		2	2	1	3	1	
		PROM				3,44					2,38							2,71				2,33			2,5							2,21				2,17						2,25
PENG Ingeniería	P12	Pequeñas versiones	4	3	4		4	4	4	3		4	4	4	4	2	3		3	4	3		5	5		4	4	5	5	3	4		4	4	4		3	4	4	2	4	
	P13	Diseño Simple	4	2	2		2	3	4	4		3	3	3	2	2	2		4	4	4		5	5		5	5	5	5	2	3		2	2	2		4	4	4	2	3	
	P14	Refactoring	3	3	4		4	2	3	3		3	3	3	4	2	2		4	4	4		4	3		4	4	3	4	2	2		2	3	2		4	4	4	1	3	
	P15	Estándares de Codificación	4	4	4		4	3	4	4		3	4	4	5	4	4		3	3	3		2	2		4	4	4	3	2	2		2	2	2		4	4	4	3	3	
		PROM				3,44					3,44							3,21				3,58			3,88							3,67				2,58						3,4
PTEST Testing	P16	Pruebas Unitarias	4	4	3		4	4	4	4		3	2	4	3	4	4		5	5	4		3	4		2	2	2	2	2	2		4	4	4		4	4	2	3	2	
	P17	Integración Continua	4	5	3		2	3	3	4		2	2	2	2	3	3		3	3	3		2	4		2	3	2	2	2	2		3	3	3		2	3	2	2	2	
		PROM				3,63					3,5							2,83				3,83			3,25							2,08				3,5						2,6

Anexo J. Encuesta aplicada

Encuesta para determinar los factores y atributos que influyen en la calidad del proceso de desarrollo de software que aplican metodologías ágiles

Esta encuesta está dirigida a los equipos de desarrollo de software que aplican metodologías ágiles. Sus respuestas son anónimas y serán utilizadas estrictamente con fines investigativos.

La Encuesta está dividida en 4 secciones. La Sección 1 está relacionada con la caracterización de la empresa y de las personas que llenan el cuestionario. La Sección 2 corresponde a preguntas que determinen la influencia de los factores críticos de éxito en la aplicación de prácticas ágiles, la Sección 3 incluye preguntas que determinen la influencia de las prácticas ágiles en los atributos de calidad del software. La Sección 4, contiene preguntas que complementan el estudio

¡Gracias por su colaboración !

Section 1: Datos Generales

Ingrese su mail: *

- 1.1 ¿La empresa en la que usted trabaja utiliza metodologías ágiles de desarrollo de software?. Si su respuesta es NO, no llene este cuestionario
Si; No
- 1.2 ¿Qué metodología ágil utiliza la empresa en la que usted trabaja?
Scrum; XP; FDD; Agile UP; Hybrid XP/Scrum; No conozco; Otra.
- 1.3 ¿Cuál es su posición en la compañía?
Jefe de Proyecto; Desarrollador; Propietario del producto/Analista del negocio; Personal de tecnologías de la Información; Tester; Otros
- 1.4 Género
Masculino, Femenino
- 1.5 ¿Cuál es el tamaño del equipo de desarrollo en la empresa donde usted trabaja?
Menos de 10 personas; 10-19 personas; 20-29 personas; 30-39 personas; 40-49 personas; Más de 50 personas.
- 1.6 ¿Cuál es la principal actividad de la empresa que usted labora?
Desarrollo de Software; Servicios Financieros, Servicios Profesionales; Salud; Institución Pública, Educación, Telecomunicaciones; Seguros, Otra.
- 1.7 ¿En qué país está ubicada la empresa en la que labora?
(Respuesta abierta)
- 1.8 ¿En qué ciudad está ud trabajando actualmente?
(Respuesta abierta)

Sección 2. Relación Factores críticos de éxito – Prácticas Ágiles

Son técnicas que se pueden aplicar durante el proceso de desarrollo ágil. Se han clasificado como

sigue:

PRÁCTICAS ÁGILES DE TRABAJO EN EQUIPO

Programación en pareja - Cliente on-site - Equipos multidisciplinarios - Equipos auto-organizados - Reuniones diarias - Propiedad colectiva del código - Retrospectivas

PRÁCTICAS ÁGILES DE GESTIÓN DEL PROYECTO

Planificación del Juego - 40 horas de trabajo por semana - Timeboxing - Monitoreo del progreso

PRÁCTICAS ÁGILES DE DISEÑO DE SOFTWARE

Versiones pequeñas - diseño simple - Refactoring - Estándares de codificación

PRÁCTICAS ÁGILES DE TESTING

Testing - Integración continua

Evalúe de acuerdo a una escala de 1 a 5 (1: No influencia influencia negativa, 5: Influencia positiva completa)

Recursos Humanos

2.1 ¿Cómo influye la CONFIANZA en las PRÁCTICAS ÁGILES que se indican a continuación ?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

2.2 ¿Cómo influye la AUTOEFICACIA DEL PERSONAL en las PRÁCTICAS ÁGILES que se indican a continuación?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

2.3 ¿Cómo influye el ESTRÉS en las PRÁCTICAS ÁGILES que se indican a continuación?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

Organización

2.4 ¿Cómo influye la INTEGRIDAD Y DISPONIBILIDAD DE LA INFORMACIÓN Y EXPERIENCIAS APRENDIDAS en las PRÁCTICAS ÁGILES que se indican a continuación?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

2.5 ¿Cómo influye la CALIDAD DE VIDA LABORAL en las PRÁCTICAS ÁGILES que se indican a continuación?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

2.6 ¿Cómo influye los MEDIOS DE COMUNICACIÓN en las PRÁCTICAS ÁGILES que se indican a continuación?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

Section 3. Relación entre las prácticas ágiles y características de calidad.

Evalúe de acuerdo a una escala de 1 - 5 (1: No influencia, 5: Alta influencia)

Adecuación Funcional:

Degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions.

3.1 ¿Cómo influyen las siguientes prácticas ágiles en la FUNCIONALIDAD del software?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

Fiabilidad:

Grado en que un sistema o componente realiza las funciones requeridas bajo condiciones específicas durante un período determinado de tiempo.

3.2 ¿Cómo influyen las siguientes prácticas ágiles en la FIABILIDAD del software?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

Usabilidad:

Grado en el que un producto puede ser utilizado por usuarios específicos para alcanzar unos objetivos especificados con efectividad, eficiencia y satisfacción en un contexto de uso dado.

3.3 ¿Cómo influyen las siguientes prácticas ágiles en la USABILIDAD del software?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

Eficiencia:

Performance relative to the amount of resources used under stated conditions.

3.4 ¿Cómo influyen las siguientes prácticas ágiles en la EFICIENCIA del software?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

Mantenibilidad:

Grado en el que el producto puede ser modificado.

3.5 ¿Cómo influyen las siguientes prácticas ágiles en la MANTENIBILIDAD del software?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

Portabilidad:

Facilidad con que el sistema o componente puede ser transferido de un entorno hardware o software a otro.

3.6 ¿Cómo influyen las siguientes prácticas ágiles en la PORTABILIDAD del software?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

Seguridad

Grado de protección de la información y de los datos que hace que personas o sistemas no autorizados no puedan leerlos o modificarlos y que a las personas o sistemas autorizados no se les niegue el acceso a los mismos.

3.7 ¿Cómo influyen las siguientes prácticas ágiles en la SEGURIDAD del software?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

Compatibilidad

Capacidad de dos o más sistemas o componentes para intercambiar información y/o realizar sus funciones requeridas mientras comparten el mismo entorno hardware o software.

3.8 ¿Cómo influyen las siguientes prácticas ágiles en la COMPATIBILIDAD del software?

Equipo de trabajo	1; 2; 3; 4; 5
Gestión del Proyecto	1; 2; 3; 4; 5
Ingeniería	1; 2; 3; 4; 5
Testing	1; 2; 3; 4; 5

Section 4. Otros Aspectos

4.1 Escoja la(s) prácticas ágiles que usted aplica en el desarrollo de software

Programación en parejas, Cliente on-site, Reuniones Stand-up, Equipos auto-organizado, Equipos multidisciplinarios, Retrospectivas, Propiedad colectiva del código, Planificación del juego, Monitoreo del progreso, 40 horas por semana, Timeboxing, Pequeñas versiones, Diseño simple, Refactoring, Estándares de codificación, Testing, Integración continua.

4.2 ¿Aplica su empresa control de calidad en el proceso de desarrollo de software?

Si; No; Parcialmente

Anexo K. Resumen obtenido de los datos de la encuesta. Relación Factor – Prácticas Ágiles

Factor	Práctica	1 (%)	2 (%)	3 (%)	4 (%)	5 (%)
FSE	PDİS	5	12	16	38	29
	PMP	4	10	16	41	28
	PTW	1	8	12	40	39
	PTEST	3	10	21	40	27
<i>Media FSE</i>		<i>3.3</i>	<i>9.8</i>	<i>16.3</i>	<i>40.1</i>	<i>30.7</i>
FCO	PDİS	5	9	26	38	21
	PMP	3	7	23	38	29
	PTW	2	6	18	33	41
	PTEST	3	8	31	31	27
<i>Media FCO</i>		<i>3.4</i>	<i>7.5</i>	<i>24.5</i>	<i>35.1</i>	<i>29.5</i>
FQWL	PDİS	4	14	16	45	21
	PMP	1	14	18	37	30
	PTW	2	8	16	42	33
	PTEST	5	11	20	43	21
<i>Media FQWL</i>		<i>3.3</i>	<i>11.5</i>	<i>17.5</i>	<i>41.6</i>	<i>26.2</i>
FIAI	PDİS	3	5	23	36	33
	PMP	2	6	20	42	30
	PTW	1	6	16	43	34
	PTEST	3	3	25	34	35
<i>Media FDİSP</i>		<i>2.1</i>	<i>5.1</i>	<i>20.9</i>	<i>38.9</i>	<i>33.0</i>
FSTR	PENG	5	14	29	29	22
	PMP	8	11	28	25	28
	PTW	5	13	23	25	34
	PTEST	8	15	27	27	23
<i>Mean FSTR</i>		<i>6.7</i>	<i>13.4</i>	<i>26.7</i>	<i>26.5</i>	<i>26.7</i>
FCM	PDİS	3	9	22	33	34
	PMP	1	6	16	33	44
	PTW	1	5	18	24	51
	PTEST	3	7	23	35	32
<i>Media FCM</i>		<i>1.9</i>	<i>6.8</i>	<i>19.9</i>	<i>31.2</i>	<i>40.2</i>

Anexo L. Resumen obtenido de los datos de la encuesta. Influencia Prácticas Ágiles - Características de Calidad

Práctica	Característica	1 (%)	2 (%)	3 (%)	4 (%)	5 (%)
FDÍS	ACOMP	3	7	19	35	36
	AEFI	3	5	20	32	40
	AFIAB	1	5	21	34	38
	AFUNC	2	3	23	27	45
	AMAIN	3	5	20	34	38
	APOINT	1	9	23	28	38
	ASEC	2	7	20	34	38
	AUSAB	0	6	21	34	40
<i>Subtotal FDÍS</i>		<i>2.1</i>	<i>6.0</i>	<i>20.8</i>	<i>32.1</i>	<i>39.0</i>
FGP	ACOMP	10	13	17	40	20
	AEFI	4	8	27	35	25
	AFIAB	5	10	21	35	28
	AFUNC	1	9	19	42	29
	AMAIN	5	11	23	34	27
	APOINT	11	13	18	38	19
	ASEC	7	11	23	34	25
	AUSAB	7	14	21	36	23
<i>Subtotal FGP</i>		<i>6.3</i>	<i>11.1</i>	<i>21.3</i>	<i>36.6</i>	<i>24.7</i>
FTE	ACOMP	6	9	17	37	31
	AEFI	4	8	19	33	36
	AFIAB	3	7	21	42	28
	AFUNC	1	5	17	29	47
	AMAIN	3	9	16	40	33
	APOINT	5	11	27	31	25
	ASEC	2	12	18	32	36
	AUSAB	4	10	20	32	35
<i>Subtotal FTE</i>		<i>3.6</i>	<i>8.8</i>	<i>19.3</i>	<i>34.4</i>	<i>33.8</i>
FTEST	ACOMP	6	8	14	47	25
	AEFI	3	7	17	30	42
	AFIAB	4	2	23	31	40
	AFUNC	3	4	18	34	41
	AMAIN	4	7	18	38	33
	APOINT	9	5	23	32	31
	ASEC	3	5	26	33	33
	AUSAB	5	4	21	35	34
<i>Subtotal FTEST</i>		<i>4.8</i>	<i>5.2</i>	<i>20.2</i>	<i>34.8</i>	<i>34.9</i>

Anexo M. Publicaciones realizadas

Arcos-Medina, G., & Mauricio, D. (2019). Aspects of software quality applied to the process of agile software development: A systematic literature review. *International Journal of System Assurance Engineering and Management*. <https://doi.org/10.1007/s13198-019-00840-7>

Arcos-Medina, G., & Mauricio, D. (2020). Identifying Critical Success Factors and Influence on the Application Agile Practices for the Software Development. *International Journal of Information Technologies and Systems Approach*.

Arcos-Medina, G., & Mauricio, D. (2020). Influence of the Application of Agile Practices in Software Quality Based on ISO/IEC 25010 Standard. *International Journal of Information Technologies and Systems Approach*.

Título	Aspects of software quality applied to the process of agile software development: a systematic literature review
Autores	G. Arcos Medina, D. Mauricio
DOI link	https://doi.org/10.1007/S13198-019-00840-7
Revista	International Journal of System Assurance Engineering and Management (IJSA)
País	India
Impacto	SJR 0,29 Q3

SpringerLink

Please take [this quick survey](#) to tell us about what happens after you publish a paper.

[International Journal of System Assurance Engineering and Management](#)
pp 1-31 | [Cite as](#)

Aspects of software quality applied to the process of agile software development: a systematic literature review

Authors [Authors and affiliations](#)

Gloria Arcos-Medina  David Mauricio

Review Papers
First Online: 03 August 2019

15 Downloads

Abstract

The importance of quality in software engineering is widely recognized today. As a result, agile

[Download PDF](#)

[Cite article](#)

[Share article](#)

Article

Abstract

1 Introduction

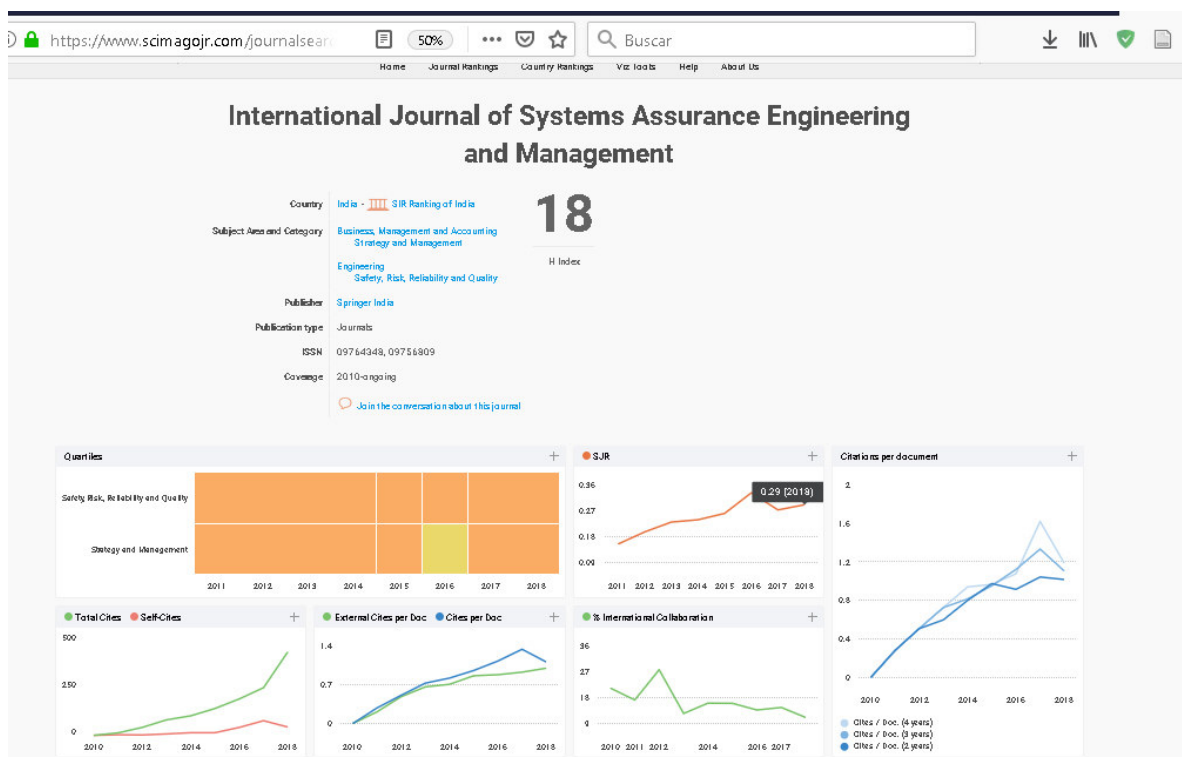
2 Quality and agile

3 Research methodology

4 Analysis

5 Discussion

6 Conclusions





Aspects of software quality applied to the process of agile software development: a systematic literature review

Gloria Arcos-Medina^{1,2} · David Mauricio²

Received: 12 January 2018 / Revised: 17 June 2019

© The Society for Reliability Engineering, Quality and Operations Management (SREQOM), India and The Division of Operation and Maintenance, Lulea University of Technology, Sweden 2019

Abstract The importance of quality in software engineering is widely recognized today. As a result, agile methods have incorporated quality practices in the software development process to achieve quality software quickly and in harmony with rapidly changing customer requirements. Experts have published a substantial amount of information related to the diverse aspects of software quality and agile development. This work aims to perform a systematic literature review of the following characteristics: critical success factors, metrics to measure quality, quality attributes, agile practices, quality models, and agile principles considered by these models. 773 papers related to quality in the agile development process were identified, of which, for the preparation of this article, 71 papers were selected. This study catalogs 118 critical success factors, 137 agile practices, 102 metrics, and 70 quality attributes that affect the agile software development. In addition, 14 quality models, which do not cover all the aspects of quality related to agile development, were studied to prepare this paper. 28.17% of the studies analyzed consider the agile practices as they affect the quality of software, which is the most studied aspect.

Keywords Quality assurance · Agile practices · Quality metrics · Quality attributes · Agile model · Systematic literature review

1 Introduction

Quality is a term that has become very important over time, as it is considered one of the chief methods a country can utilize to improve its global competitive position (Ivancevich et al. 1997). Software quality can be defined as “the totality of characteristics of a software product that satisfy stated or implied needs” (ISO/IEC 9126 2001). According to (IEEE SA 610.2 1990), quality software assurance that ensures that the software development lifecycle process and its products according to the requirements, standards, and procedures of the industry through the planned and systematic set of operations. This concept allows us to classify software quality into two groups: product quality and software development process quality (Muñoz et al. 2010), the latter being the research topic of this work.

The software development process is an important part of software engineering that influences the outcome of the product. It is a complex process that includes activities

Título	Identifying Critical Success Factors and Influence on the Application Agile Practices for the Software Development
Autores	G. Arcos Medina, D. Mauricio
DOI	10.31341/jios.44.1.1
Revista	Journal of Information and Organizational Sciences (JIOS)
País	Croacia
Impacto	SJR 0,29 Q3

Journal of Information and Organizational Sciences

[Current](#)
[Archives](#)
[Announcements](#)
[About](#)

Search

[Home](#) / [Archives](#) / [Vol 44 No 1 \(2020\)](#) / Articles

Identifying Factors Influencing on Agile Practices for Software Development

Gloria ArcosMedina
Escuela Superior Politécnica de Chimborazo

David Mauricio
Facultad de Informática y Sistemas, Universidad Nacional Mayor de San Marcos

DOI: <https://doi.org/10.31341/jios.44.1.1>

Keywords: Agile Practices, Critical success factors, Software quality, ASD, Social theory

Abstract

Agile practices are activities or actions that are utilized during software development to improve the quality and productivity of the resulting software. Those practices are influenced by a set of factors that affect the realization of their objectives. The purpose of this study is to determine

[PDF](#)

Published
2020-06-24

How to Cite
G. ArcosMedina and D. Mauricio, "Identifying Factors Influencing on Agile Practices for Software Development", *JIOS*, vol. 44, no. 1, Jun. 2020.

More Citation Formats

Issue
[Vol 44 No 1 \(2020\)](#)

Section

Current Issue

[JIOS 44.1](#)
[JIOS 44.2](#)
[JIOS 44.3](#)

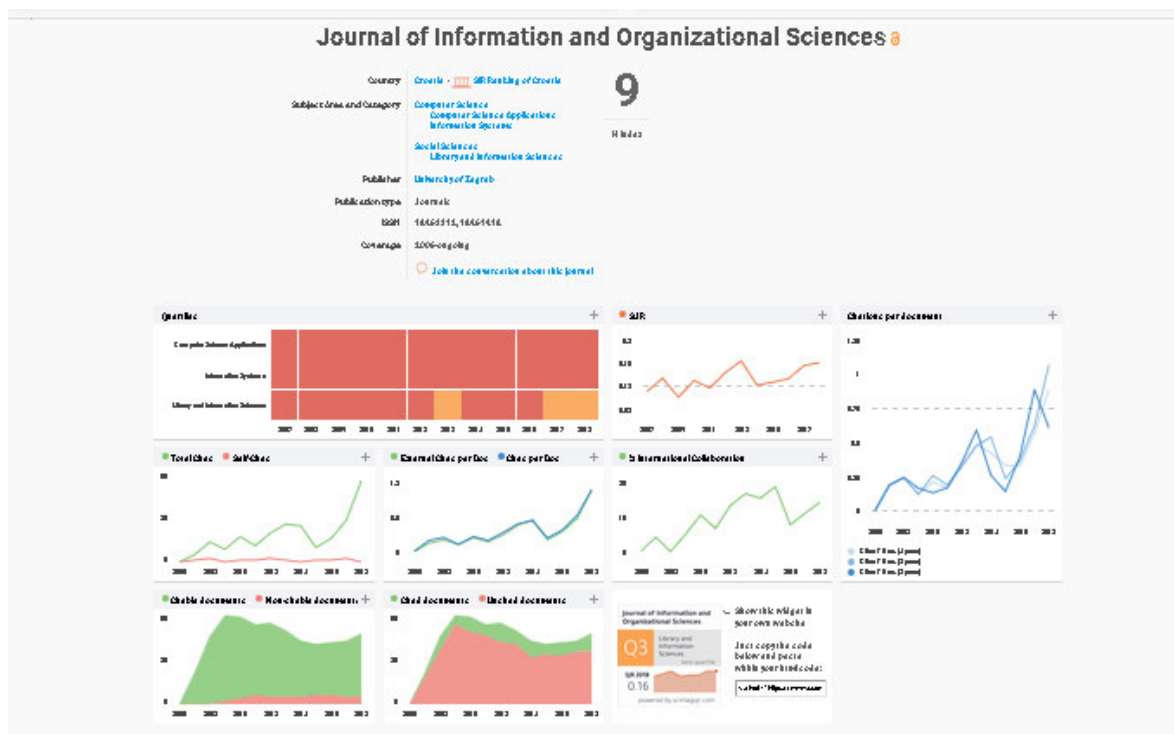
[Make a Submission](#)

Information

[For Readers](#)
[For Authors](#)
[For Librarians](#)

Usage Statistics Information

We log anonymous usage statistics. Please read the [privacy information](#) for details.



Identifying Factors Influencing on Agile Practices for Software Development

Gloria Arcos-Medina

garcos@esPOCH.edu.ec

Facultad de Informática y Electrónica

Escuela Superior Politécnica de Chimborazo, Riobamba, Ecuador

Facultad de Informática y Sistemas

Universidad Nacional Mayor de San Marcos, Lima, Perú

David Mauricio

dmauricios@unmsm.edu.pe

Facultad de Informática y Sistemas

Universidad Nacional Mayor de San Marcos, Lima, Perú

Abstract

Agile practices are activities or actions that are utilized during software development to improve the quality and productivity of the resulting software. Those practices are influenced by a set of factors that affect the realization of their objectives. The purpose of this study is to determine the influence of organizational factors and the personnel working on the software development project to the agile practices. To this end, a model has been proposed that is composed of six critical success factors derived from the theories of administrative and human behavior and four groups of agile practices, and resulted in identifying 13 factor-practice relationships. The results from 146 questionnaires addressed to individuals involved in the software development process show that the factors of Confidence, Perception of Self-efficacy, Integrity and Availability of the Information and Experiences Learned, and Media Used all have a noteworthy positive influence on teamwork agile practices.

Keywords: agile practices, critical success factors, software quality, ASD, social theory.

The Influence of the Application of Agile Practices in Software Quality Based on ISO/IEC 25010 Standard

Gloria Arcos-Medina, Escuela Superior Politécnica de Chimborazo, Ecuador; Universidad Nacional Mayor de San Marcos, Perú

David Mauricio, Universidad Nacional Mayor de San Marcos, Perú

 <https://orcid.org/0000-0001-9262-626X>

ABSTRACT

Agile practices are activities or procedures that are applied during the software development process in order to improve its quality and productivity. The objective of this study is to determine the influence of agile practices on software quality. For this purpose, a model composed of 4 groups of agile practices and 8 quality characteristics according to the ISO/IEC 25010 standard has here been proposed. The results of 146 questionnaires addressed to people involved in the software development process show that the application of agile engineering and project management practices have a significant positive influence on the quality attribute functionality. On the other hand, project management practices have a low impact on the quality characteristics of compatibility, portability, security, and usability.

KEYWORDS

Agile Practices, Agile Software Development, Conceptual Model, Quality Attributes, Software Quality