



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

**Un Marco de Trabajo para el Desarrollo de Software
Web Seguro con Metodologías Ágiles**

TESIS

Para optar el Grado Académico de Magíster en Ingeniería de
Sistemas e Informática con mención en Ingeniería de Software

AUTOR

Gilmer Glicerio VALDERRAMA HERRERA

ASESOR

Dr. José Alfredo HERRERA QUISPE

Lima, Perú

2025



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.

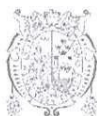
Referencia bibliográfica

Valderrama, G. (2025). *Un Marco de Trabajo para el Desarrollo de Software Web Seguro con Metodologías Ágiles*. [Tesis de maestría, 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	Gilmer Glicerio Valderrama Herrera
Tipo de documento de identidad	DNI
Número de documento de identidad	18203834
URL de ORCID	https://orcid.org/0009-0002-7065-9081
Datos de asesor	
Nombres y apellidos	José Alfredo Herrera Quispe
Tipo de documento de identidad	DNI
Número de documento de identidad	40362859
URL de ORCID	https://orcid.org/0000-0002-8207-9714
Datos del jurado	
Presidente del jurado	
Nombres y apellidos	Cesar Luza Montero
Tipo de documento	DNI
Número de documento de identidad	06111988
Miembro del jurado 1	
Nombres y apellidos	Adegundo Mario Camara Figueroa
Tipo de documento	DNI
Número de documento de identidad	09095759
Miembro del jurado 2	
Nombres y apellidos	Jorge Leoncio Guerra Guerra
Tipo de documento	DNI
Número de documento de identidad	08473333
Datos de investigación	
Línea de investigación	C.0.3.22. Ingeniería de software
Grupo de investigación	No aplica.

Agencia de financiamiento	Sin financiamiento.
Ubicación geográfica de la investigación	Universidad Nacional Mayor de San Marcos y sus coordenadas geográficas. Latitud: -12.05593 Longitud: -77.08449
Año o rango de años en que se realizó la investigación	2022 - 2024
URL de disciplinas OCDE	Ingeniería de sistemas y comunicaciones https://purl.org/pe-repo/ocde/ford#2.02.04 Otras ingenierías y tecnologías https://purl.org/pe-repo/ocde/ford#2.11.02



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 DE TESIS PARA OPTAR EL GRADO ACADÉMICO DE MAGÍSTER
EN INGENIERÍA DE SISTEMAS E INFORMÁTICA CON MENCIÓN EN INGENIERÍA DE
SOFTWARE**

A los siete (21) días del mes de octubre de 2025, siendo las 10:00 a.m., se reunieron en el Auditorio Alfredo Celso Alva Bravo, el Jurado de Tesis conformado por los siguientes docentes:

Mg. Cesar Luza Montero (Presidente)
Mg. Adegundo Mario Camara Figueroa (Miembro)
Mg. Jorge Leoncio Guerra Guerra (Miembro)
Dr. José Alfredo Herrera Quispe (Miembro Asesor)

Se inició la Sustentación invitando al candidato a Magíster **GILMER GLICERIO VALDERRAMA HERRERA**, para que realice la exposición oral de la tesis para optar el Grado Académico de Magíster en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software, siendo la Tesis intitulada:

**“Un Marco de Trabajo para el Desarrollo
de Software Web Seguro con Metodologías Ágiles”.**

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

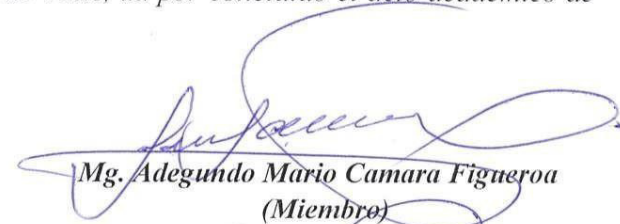
..... *Diez y siete (17)* *Muy Bueno*

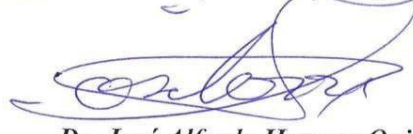
Por tanto, el presidente del Jurado, de acuerdo con el Reglamento General de Estudios de Posgrado, otorga al Bachiller **GILMER GLICERIO VALDERRAMA HERRERA** el Grado Académico de Magíster en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software.

Siendo las *11:30* horas, el presidente del Jurado de Tesis, da por concluido el acto académico de Sustentación de Tesis.


Mg. Cesar Luza Montero
(Presidente)


Mg. Jorge Leoncio Guerra Guerra
(Miembro)


Mg. Adegundo Mario Camara Figueroa
(Miembro)


Dr. José Alfredo Herrera Quispe
(Miembro Asesor)

CERTIFICADO DE SIMILITUD

Yo JOSÉ ALFREDO HERRERA QUISPE en mi condición de asesor acreditado con Dictamen N° 000302-2022-UPG-VDIP-FISI/UNMSM de la tesis, cuyo título es “Un Marco de Trabajo para el Desarrollo de Software Web Seguro con Metodologías Ágiles” presentado por el bachiller GILMER GLICERIO VALDERRAMA HERRERA para optar el grado académico de Magíster en Ingeniería de Sistemas e Informática con Mención en Ingeniería de Software, CERTIFICO que se ha cumplido con lo establecido en la Directiva de Originalidad y de Similitud de Trabajos Académicos, de Investigación y Producción Intelectual. Según la revisión, análisis y evaluación mediante la plataforma PIDS, el documento evaluado cuenta con el porcentaje de 13% de similitud, nivel **PERMITIDO** para continuar con los trámites correspondientes y para su publicación en el repositorio institucional.

Se emite el presente certificado en cumplimiento de lo establecido en las normas vigentes, como uno de los requisitos para la obtención del grado académico de Magíster en Ingeniería de Sistemas e Informática con Mención en Ingeniería de Software.



Lima, 29 de octubre de 2024

Firma del asesor:

Nombres y apellidos del asesor: JOSÉ ALFREDO HERRERA QUISPE

DNI: 40362859

DEDICATORIA

A mi Adorada Madre OLINDA, cuya inquebrantable fe en mí y amor incondicional han sido mi mayor fuente de fortaleza y motivación. Su ejemplo de resiliencia, sacrificio y bondad inagotable han iluminado cada paso de este viaje.

A mi Amada Esposa OLGA, por tu amor, paciencia y comprensión incansables. Tu apoyo en los momentos más desafiantes de este camino ha sido el faro que me ha guiado hacia este logro. Gracias por ser mi compañera, mi confidente y mi mejor amiga.

A mis Adorados Hijos ISABEL y MATIAS, son la luz de mi vida y la alegría de cada día. Este logro es también vuestro; es el fruto de los sacrificios que hemos compartido y de las incontables horas de trabajo que nos han mantenido separados. Espero que este esfuerzo les sirva de inspiración para perseguir sus propios sueños.

A mi Querida Tía YOLANDA, tu sabiduría, tu apoyo incondicional y tu amor han sido pilares fundamentales en mi vida. Tu ejemplo de fortaleza y determinación ha sido una

fuentes de inspiración constante en este viaje académico. Gracias por creer en mí.

A mis Queridos Tíos, NORMA, WILMER, ARISTIDES, ALEJANDRO, En cada etapa de mi vida, ustedes han sido más que familia, han sido mentores, amigos y un gran apoyo. Su sabiduría, sus consejos y su amor incondicional han sido luces guía en mi camino.

AGRADECIMIENTOS

Primero y, ante todo, expreso mi más profundo agradecimiento a Dios por darme fortaleza, sabiduría y serenidad a lo largo de este desafiante pero gratificante camino.

Mi más sincero agradecimiento a mi asesor de tesis, Dr. José Alfredo Herrera Quispe, por su invaluable orientación, paciencia y apoyo a lo largo de este proceso. Su conocimiento experto y su enfoque crítico han sido fundamentales en la formación de esta investigación

Mi más profundo agradecimiento a todas aquellas personas, familiares y amigos, por su amistad, apoyo y ánimo. Ustedes más que amigos; han sido una fuente de alegría, refugio en los momentos difíciles y compañeros constantes en esta aventura. Cada palabra de aliento, cada momento de diversión y cada gesto de apoyo ha contribuido inmensamente a mi bienestar y éxito.

Mi enorme gratitud a todos los profesionales y colegas que generosamente aportaron su tiempo, conocimientos y experiencia para el enriquecimiento de esta tesis. Su disposición a compartir su perspectiva profesional y sus valiosas contribuciones han sido fundamentales para dar profundidad y rigor a mi investigación.

ÍNDICE GENERAL

CAPÍTULO I. INTRODUCCIÓN	13
1.1 Situación Problemática.....	13
1.2 Formulación del Problema.....	16
1.2.1 Problema General.....	16
1.2.2 Problemas específicos	17
1.3 Justificación Teórica.....	17
1.4 Justificación Práctica.....	18
1.5 Objetivos	19
1.5.1 Objetivo Principal	19
1.5.2 Objetivos Específicos.....	19
CAPÍTULO II. MARCO TEORICO	21
2.1 Marco Filosófico o Epistemológico de la Investigación	21
2.1.1 Epistemología del Desarrollo Ágil.....	21
2.1.2 Epistemología del Desarrollo Seguro de Software	23
2.2 Bases Teóricas	24
2.2.1 Metodologías ágiles de desarrollo de software	24
2.2.2 Software Seguro	26
2.2.3 Ingeniería de Seguridad de software	27
2.2.4 Amenazas de seguridad en el software web según OWASP TOP 10 (2021)	27
2.2.5 Actividades de seguridad en el desarrollo de software	28
2.2.6 Modelos de categorización y clasificación de amenazas	31
2.2.7 Diagrama de flujo de datos (DFD).....	33
2.2.8 Historias de abuso	33
2.2.9 Historias de usuario relacionadas a la seguridad.....	34
2.2.10 Metodologías de Desarrollo de software Seguro	34
CAPÍTULO III. ESTADO DEL ARTE.....	39
3.1 Revisión sistemática de la literatura	39
3.2 Planificación de la revisión de la literatura	40
3.2.1 Objetivos de la revisión.....	40
3.2.2 Preguntas de investigación.....	40
3.2.3 Cadenas de búsqueda	41

3.2.4	Fuentes	41
3.2.5	Criterios de inclusión y exclusión	42
3.3	Ejecución de la revisión de la literatura	42
3.3.1	Búsqueda de fuentes.....	42
3.3.2	Selección de artículos.....	43
3.3.3	Artículos científicos relacionados con el trabajo de investigación	44
3.4	Resultados y análisis de la revisión de la literatura	49
3.4.1	Q1: Identificación de prácticas de seguridad en metodologías ágiles.....	50
3.4.2	Q2: Evaluación del grado de agilidad de las prácticas de seguridad	53
3.4.3	Q3: Análisis de costo-beneficio	54
3.5	Conclusiones de la revisión sistemática.....	56
CAPÍTULO IV. METODOLOGÍA.....		57
4.1	Tipo y Diseño de la Investigación	57
4.1.1	Tipo de investigación	57
4.1.2	Diseño de investigación	57
4.2	Unidad de Análisis	58
4.3	Población de Estudio.....	58
4.4	Tamaño de Muestra	59
4.5	Selección de Muestra	61
4.6	Técnica de Recopilación de Datos	62
4.7	Análisis e Interpretación de la Información	63
CAPÍTULO V. MARCO DE TRABAJO PROPUESTO.....		65
5.1	Metodología Seleccionada y Actividades de Seguridad.....	68
5.1.1	Selección de Metodología Ágil	68
5.1.2	Selección de Actividades de Seguridad	69
5.1.3	Costo-Beneficio de la Seguridad.....	70
5.1.4	Consideraciones Sobre Agilidad y Costo-Beneficio.....	70
5.1.5	Criterios de Selección de Actividades de Seguridad.....	71
5.1.6	Validación de la Selección: Impacto Medido y Cuantificado.....	72
5.1.7	Medición Empírica del Impacto de la Seguridad	72
5.2	Propuesta de Marco de Trabajo para el Desarrollo de Software Web Seguro con Metodologías Ágiles.....	73
5.3	Descripción del Marco de Trabajo.....	74
5.3.1	Inicio	74
5.3.2	Planificación.....	76

5.3.3 Implementación.....	82
5.3.4 Liberación	84
CAPÍTULO VI. RESULTADOS	86
6.1 Fiabilidad de los Datos (Alpha Cronbach)	86
6.2 Análisis Descriptivo de los Resultados de la Muestra.....	86
6.2.1 Causa, frecuencia e impacto de ataque(s) a un software web sin actividades de desarrollo seguro	87
6.2.2 Experiencia en el desarrollo de software usando Scrum.....	89
6.2.3 Incorporación de un especialista en desarrollo de software seguro usando Scrum	90
6.2.4 Costo de incorporar la actividad de seguridad "Modelado de Amenazas" durante el desarrollo	91
6.2.5 Beneficio de incorporar la actividad de seguridad "Modelado de Amenazas" durante el desarrollo.....	92
6.2.6 Tiempo adicional que demanda incorporar actividades de seguridad en el desarrollo con SCRUM.....	92
6.3 Análisis del Modelado de Amenazas (Costo-Beneficio).....	93
6.3.1 Estadísticos descriptivos	94
6.3.2 Resumen de la Prueba de Rangos de Wilcoxon.....	95
6.3.3 Interpretación de la prueba de rangos de Wilcoxon:.....	96
6.4 Análisis de la Dimensión Tiempo.....	97
6.5 Grado de Agilidad al Aplicar el Marco de Trabajo Propuesto	99
6.5.1 Cálculo del factor de ponderación de incorporación de un especialista en desarrollo de software seguro respecto al grado de agilidad al aplicar el marco de trabajo propuesto.....	100
6.5.2 Cálculo del factor de ponderación de la dimensión tiempo	102
6.5.3 Cálculo del grado de agilidad aplicando los factores de ponderación	103
6.6 Implementación del Marco de Trabajo Propuesto	105
CAPÍTULO VII. DISCUSIÓN	108
7.1 Sobre costo y beneficio.....	109
7.2 Sobre Dimensión Tiempo	110
7.3 Sobre la Incorporación de un Especialista en Desarrollo Seguro	111
7.4 Sobre Grado de Agilidad.....	112
CAPÍTULO VIII. CONCLUSIONES Y Trabajos FUTUROS	113
8.1 Conclusiones.....	113
8.2 Aportes de Este Trabajo.....	116
8.2.1 ¿Qué problema resuelve esta tesis?	116

8.2.2	¿Cuál es su contribución principal?	117
8.2.3	Impacto y beneficios del aporte	117
8.3	Trabajos Futuros	118

LISTA DE FIGURAS

<i>Figura 1.</i>	Reporte de vulnerabilidades por año (CVEDetails, n.d.).....	14
<i>Figura 2.</i>	Reporte de ataques más comunes a software web, 2024. (CVEDetails, n.d.).....	15
<i>Figura 3.</i>	Reporte de tipo de ataques a software web por año, 2024. (CVEDetails, n.d.).....	15
<i>Figura 4.</i>	Top 10 de amenazas de seguridad adaptado Fuente. (OWASP Foundation, 2021)..	28
<i>Figura 5.</i>	Prácticas y actividades CLASP (Oyetoyan et al., 2017).....	35
<i>Figura 6.</i>	Actividades de seguridad propuesta por Microsoft Security Development Lifecycle Process.	36
<i>Figura 7.</i>	Actividades que contempla el estándar de criterios comunes (Common Criteria) (Rindell, 2019).	37
<i>Figura 8.</i>	TouchPoints (Herath & Wimalaratne, 2018).....	38
<i>Figura 9.</i>	Flujo de la revisión de la literatura.	43
<i>Figura 10.</i>	Artículos potenciales.....	49
<i>Figura 11.</i>	Artículos seleccionados.....	50
<i>Figura 12.</i>	Modelo conceptual de marco de trabajo propuesto.	66
<i>Figura 13.</i>	Marco de trabajo propuesto.	67
<i>Figura 14.</i>	Actividades de desarrollo seguro en la fase de inicio.....	75
<i>Figura 15.</i>	Actividades de desarrollo seguro en la fase de planificación.	76
<i>Figura 16.</i>	Modelado de amenazas.....	77
<i>Figura 17.</i>	Actividades de desarrollo seguro en la fase de implementación.....	82
<i>Figura 18.</i>	Actividad de desarrollo seguro en la fase de liberación	84
<i>Figura 19.</i>	Resultado de fiabilidad de datos del cuestionario aplicado en el presente trabajo de investigación.....	86
<i>Figura 20.</i>	Gráfico de cajas de la comparación de Costo y Beneficio de Incorporar el Modelado de Amenazas.....	96
<i>Figura 21.</i>	Tiempo adicional por sprint al aplicar el marco de trabajo propuesto.....	99
<i>Figura 22.</i>	Grado de agilidad al aplicar el marco de trabajo propuesto (aplicando factores de ponderación).	104

LISTA DE CUADROS

Cuadro 1.	Cadenas de búsqueda aplicados a los campos de los bancos de datos.	41
Cuadro 2.	Criterios de inclusión y exclusión.	42
Cuadro 3.	Cadenas de búsqueda aplicados a los campos de los bancos de datos.	43
Cuadro 4.	Prácticas de seguridad en metodologías ágiles de acuerdo con la revisión sistemática de la literatura.	51
Cuadro 5.	Evaluación de agilidad de las prácticas de seguridad en metodologías ágiles de acuerdo con la revisión sistemática de la literatura.	54
Cuadro 6.	Análisis Costo-Beneficio de las Prácticas de seguridad en metodologías ágiles de acuerdo con la revisión sistemática de la literatura.	55
Cuadro 7.	Valores de grado de confianza y su correspondiente distribución Z.	59
Cuadro 8.	Valores para calcular el tamaño de la muestra.	60
Cuadro 9.	Profesionales expertos en desarrollo de software basado en web encuestados.	62
Cuadro 10.	Agilidad de metodologías ágiles.	69
Cuadro 11.	Rangos costo – beneficio.	70
Cuadro 12.	Agilidad, costo y beneficio de las actividades de seguridad.	71
Cuadro 13.	Resultado (frecuencias) de tiempo adicional que demanda incorporar las actividades de seguridad.	93
Cuadro 14.	Resultados costo y beneficio al incorporar en el proceso la actividad de “Modelado de Amenazas” (Estadísticos Descriptivos).	95
Cuadro 15.	Resumen resultado de prueba de Wilcoxon.	95
Cuadro 16.	Valores para el cálculo de media de datos agrupados.	97
Cuadro 17.	Resultado de las medias (horas) de tiempo adicional por cada una de las actividades de seguridad que se incorporan al marco de trabajo propuesto.	98
Cuadro 18.	Datos para determinar el grado de agilidad.	104
Cuadro 19.	Resultado final de incorporar las actividades de seguridad en el marco de trabajo propuesto.	108

RESUMEN

El desarrollo de software bajo un enfoque ágil, utilizando SCRUM, ha experimentado un crecimiento significativo en la industria tecnológica, posicionándose como una alternativa eficiente frente a los modelos tradicionales. Este enfoque se caracteriza por ser incremental e iterativo, permitiendo entregas continuas de código que aseguran productos funcionales y adaptables a los cambios. No obstante, dicha flexibilidad también plantea desafíos en términos de seguridad, ya que en este entorno las amenazas y vulnerabilidades pueden no abordarse adecuadamente debido a factores como plazos ajustados y cambios frecuentes en los requisitos.

En este contexto se propone un marco de trabajo que es una mejora al modelo "Agility of Security Practices and Agile Process Models: An Evaluation of Cost for Incorporating Security in Agile Process Models" de H. Maria Maqsood y Andrea Bondavalli. Se han seleccionado ocho actividades de seguridad que se incorporan durante el proceso de desarrollo, considerando su grado de agilidad, costo-beneficio. Para validar el marco de trabajo se diseñó una encuesta orientada a gerentes/directores, analistas de sistemas, programadores, analistas de calidad de software, scrums másteres y gestores de proyectos de software tomando en cuenta factores de ponderación como: la incorporación de un especialista de desarrollo de software seguro y la dimensión de tiempo adicional para incorporar las actividades de seguridad. Allí se determinó el grado de agilidad del marco propuesto y su factibilidad de uso en procesos de desarrollo SCRUM. De la implementación del marco de trabajo en un caso de estudio se logra mitigar satisfactoriamente amenazas y vulnerabilidades evidenciando así el desarrollo de un software web seguro.

Palabras clave: software Seguro, Desarrollo Seguro, Desarrollo Agile Seguro.

ABSTRACT

Software development using an agile approach, specifically through the SCRUM methodology, has experienced significant growth in the technology industry, positioning itself as an efficient alternative to traditional models. This approach is characterized by its incremental and iterative nature, enabling continuous code deliveries that ensure functional and adaptable products. However, such flexibility also presents security challenges, as threats and vulnerabilities may not be adequately addressed due to tight deadlines and frequent changes in requirements.

In this context, a framework is proposed based on the model *"Agility of Security Practices and Agile Process Models: An Evaluation of Cost for Incorporating Security in Agile Process Models"* by H. Maria Maqsood and Andrea Bondavalli. Eight security activities were selected to be incorporated during the development process, considering their level of agility and cost-effectiveness. To validate the framework, a survey was designed targeting managers, system analysts, developers, quality assurance analysts, scrum masters, and project managers. The survey evaluated key factors such as the inclusion of secure software development specialists and the additional time required to integrate security practices.

The results helped determine the framework's level of agility and its feasibility for use within SCRUM-based development processes. The implementation of the framework in a case study successfully mitigated threats and vulnerabilities, thereby demonstrating its effectiveness in enabling secure web software development.

Keywords: Secure software, Secure Development, Secure Agile Development.

CAPÍTULO I. INTRODUCCIÓN

1.1 Situación Problemática

El desarrollo de software mediante el uso de metodologías ágiles, como SCRUM, ha ido ganando popularidad en la industria de desarrollo de software debido a su capacidad para adaptarse rápidamente a los cambios y entregar productos funcionales de manera incremental (Schwaber, & Sutherland, 2020). Sin embargo, este enfoque prioriza la entrega rápida de productos funcionales, relegando en muchas ocasiones aspectos críticos como la seguridad del software (Maqsood, & Bondavalli, 2020). Las aplicaciones web, en particular, enfrentan serias vulnerabilidades debido a la percepción de que la seguridad debe gestionarse en las fases de pruebas finales del desarrollo, o que es solo un atributo de calidad secundario (Alenezi et al., 2022; Sharma & Bawa, 2020), la ausencia de una cultura de seguridad en los equipos de desarrollo ágil, sumada a la presión por cumplir con plazos ajustados, hace que los riesgos de seguridad se pasen por alto (Mohino et al., 2019)(Schwaber, & Sutherland, 2020)(Maqsood, & Bondavalli, 2020).

Actualmente, muchas organizaciones que adoptan metodologías ágiles no integran adecuadamente prácticas de seguridad en sus procesos de desarrollo, lo que incrementa la probabilidad de generar aplicaciones web y móviles con vulnerabilidades (Maqsood & Bondavalli, 2020; Sharma & Bawa, 2020). Una falla de seguridad puede tener repercusiones graves, como pérdida de datos confidenciales, el daño a la reputación de la empresa, pérdidas económicas (OWASP Foundation, 2021). Además, los desarrolladores suelen carecer de una formación específica en seguridad, lo que agrava

el riesgo de que estas vulnerabilidades no se detecten ni se gestionen adecuadamente (Moyón et al., 2020; Maier, Ma, & Bloem, 2017) (OWASP Foundation, 2021).

Según la plataforma de reporte de vulnerabilidades CVE Details (CVEDetails, n.d.), las vulnerabilidades en aplicaciones web vienen aumentando año tras año, afectando la confidencialidad e integridad de los datos, la reputación de las organizaciones y su estabilidad financiera (ver Figura 1) (CVEDetails, n.d.).

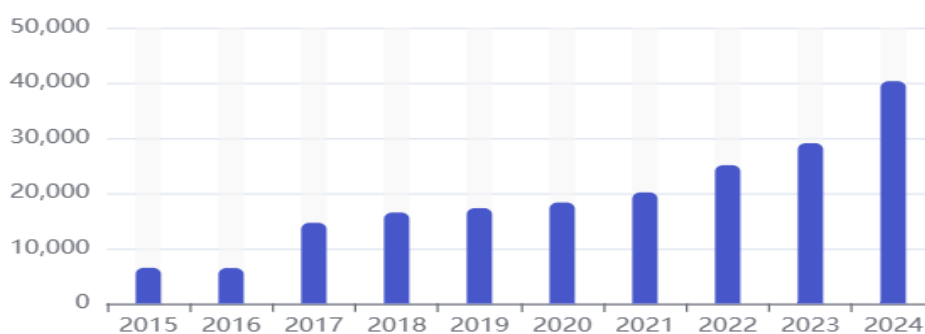


Figura 1. Reporte de vulnerabilidades por año (CVEDetails, n.d.).

Los ataques cibernéticos han evolucionado, siendo las inyecciones SQL, el cross-site scripting (XSS), los más comunes, de acuerdo con el detalle de vulnerabilidades y exposiciones comunes (CVE) reportado por el sitio web CVE Details (CVEDetails, n.d.), (ver Figura 2 y Figura 3).

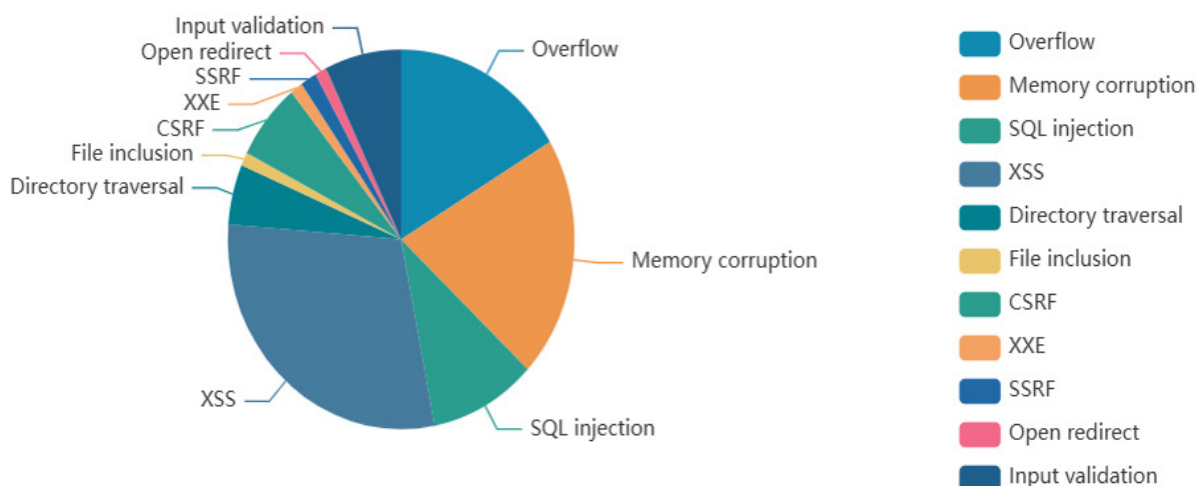


Figura 2. Reporte de ataques más comunes a software web, 2024. (CVEDetails, n.d.).

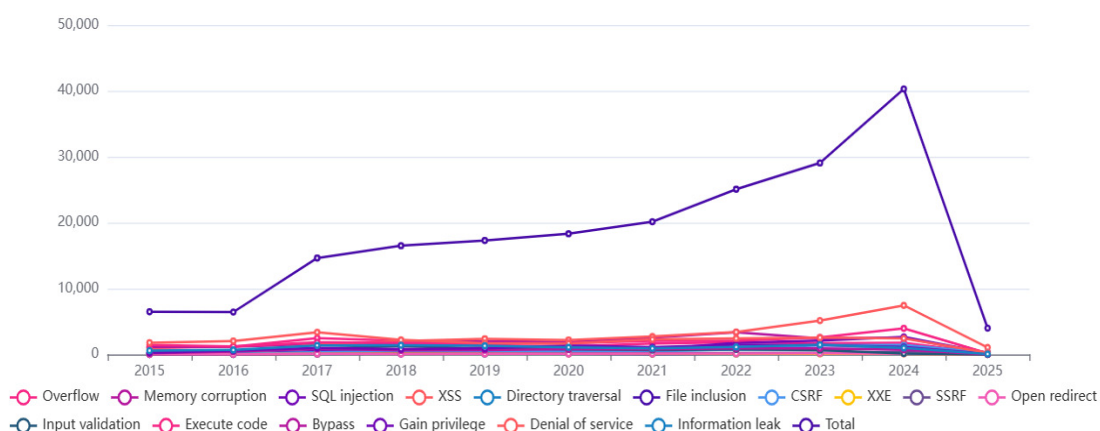


Figura 3. Reporte de tipo de ataques a software web por año, 2024. (CVEDetails, n.d.).

Para mitigar estos riesgos, se han desarrollado diversos marcos de seguridad en entornos de desarrollo tradicional, como el *Microsoft Security Development Lifecycle (MSDL)* y el *Comprehensive Lightweight Application Security Process (CLASP)* (Sharma & Bawa, 2020). Sin embargo, estos modelos no están diseñados para metodologías ágiles,

donde la velocidad y la flexibilidad son esenciales (Moyón et al., 2020). Como resultado, las organizaciones que buscan incorporar seguridad en el desarrollo ágil enfrentan barreras como el aumento de costos, la reducción de la agilidad y la falta de recursos especializados (Maqsood, & Bondavalli, 2020). Por tanto, se puede afirmar que existe una necesidad de establecer qué actividades de seguridad para el desarrollo de software seguro se integran mejor con las metodologías ágiles, sin que estas afecten su grado de agilidad, sean de bajo costo y aporten un mayor beneficio (Moyón et al., 2020)(Maqsood, & Bondavalli, 2020).

Dado este panorama, surge la necesidad de un marco de trabajo que permita la integración de actividades de seguridad en SCRUM sin comprometer su naturaleza ágil (grado de agilidad). Esto permitiría a las organizaciones desarrollar software web más seguro sin afectar la productividad y eficiencia de los equipos de desarrollo.

1.2 Formulación del Problema

1.2.1 Problema General

En base a la situación problemática surge la siguiente pregunta **¿Cómo integrar de manera efectiva un conjunto de actividades de seguridad en el proceso de desarrollo ágil SCRUM, con el fin de reducir las vulnerabilidades del software sin afectar su agilidad, y asegurando que las prácticas de seguridad implementadas sean de bajo costo y alto beneficio?**

1.2.2 Problemas específicos

Para tal efecto se determina los siguientes problemas específicos:

- ¿Qué actividades de seguridad son más adecuadas para incorporarse en cada fase del ciclo de desarrollo ágil SCRUM, considerando su grado de agilidad, impacto en la seguridad y relación costo-beneficio?
- ¿Cómo validar que la implementación del marco propuesto contribuye efectivamente a la mitigación de amenazas y vulnerabilidades, manteniendo la percepción de agilidad en los equipos de desarrollo?

1.3 Justificación Teórica

El desarrollo de este trabajo de investigación se realiza con el propósito de aportar al conocimiento existente sobre las actividades de desarrollo de software seguro adaptados por entornos de desarrollo tradicional y que pueden ser integrados en entornos ágiles, particularmente en SCRUM.

El marco propuesto permitirá mejorar la seguridad de software web desarrollado bajo metodologías ágiles sin comprometer el grado de agilidad, a un bajo costo y de alto beneficio proporcionando una solución viable para mitigar riesgos y vulnerabilidades.

1.4 Justificación Práctica

El desarrollo de esta investigación provee a la ingeniería de Software un marco de trabajo para abordar el desarrollo de software seguro en metodologías ágiles a fin de reducir ataques por agentes externos o internos (ataques de inyección, pérdida de control de acceso y fallos de identificación y autenticación, fallos criptográficos, fallas de integridad de software y datos, Fallas en el registro de seguridad y monitoreo, configuración incorrecta de seguridad, componentes vulnerables y obsoletos).

Las ventajas, aportes e impactos positivos que aporta este marco de trabajo en el desarrollo de software basado en web son:

- El marco de trabajo generará un impacto positivo para las organizaciones, dado que el marco reduce el riesgo de ataques mal intencionados al que puede estar expuesto el software web.
- Los equipos de desarrollo podrán integrar actividades de seguridad en actividades ágiles en función a su grado de agilidad, bajo costo y alto beneficio mitigando las amenazas y vulnerabilidades a las que podría estar expuesta un producto de software basado en web.
- Este marco de trabajo ayudará a las organizaciones a optimizar sus recursos, mejorar la calidad del software y asegurar la continuidad de sus operaciones.

1.5 Objetivos

1.5.1 *Objetivo Principal*

Diseñar y validar un marco de trabajo ligero para el desarrollo de software web seguro en metodologías ágiles, que integre de manera efectiva un conjunto de actividades de seguridad de bajo costo y alto beneficio, sin comprometer el grado de agilidad del proceso, con el fin de mitigar vulnerabilidades desde las primeras fases del ciclo de vida del desarrollo.

1.5.2 *Objetivos Específicos*

- Analizar el estado del arte sobre actividades y prácticas de seguridad en metodologías ágiles, con énfasis en aquellas aplicables al desarrollo seguro de software basados en web.
- Seleccionar las prácticas y/o actividades de desarrollo de software seguro específicas para su integración en el marco de trabajo propuesto.
- Evaluar mediante una encuesta a profesionales expertos (gerentes/directores, analistas de sistemas, programadores, analistas de calidad de software, scrums másteres y gestores de proyectos de software), para validar la propuesta y así determinar la factibilidad de incorporar en el proceso de desarrollo ágil actividades de desarrollo de software seguro, teniendo en cuenta las dimensiones: grado de agilidad (basado en factores de ponderación), costo-beneficio y tiempo adicional que demandan desarrollar dichas actividades.

- Implementar y evaluar la metodología en un caso de estudio identificando aciertos y oportunidades de mejora.

CAPÍTULO II. MARCO TEORICO

En este capítulo se desarrollan los siguientes conceptos:

En Marco Filosófico o Epistemológico de la Investigación: Se desarrolla las definiciones de la epistemología tanto del desarrollo ágil como la epistemología del desarrollo seguro de software (seguridad).

En Bases Teóricas: Se incluyen definiciones clave sobre metodologías ágiles, seguridad en software y modelos de integración de seguridad en ciclos de desarrollo.

2.1 Marco Filosófico o Epistemológico de la Investigación

2.1.1 Epistemología del Desarrollo Ágil

La epistemología del desarrollo ágil se fundamenta en una visión pragmática y empírica del conocimiento, donde el aprendizaje se genera de forma continua a partir de la experiencia, la colaboración y la respuesta al cambio. A diferencia de enfoques tradicionales que privilegian la planificación extensa y el conocimiento previo, las metodologías ágiles como SCRUM promueven ciclos iterativos de construcción de conocimiento a través de entregas funcionales, retroalimentación temprana y mejora continua (pequeños ciclos de construir → probar → aprender). Cada iteración convierte la experiencia directa del equipo y la retroalimentación del cliente en mejoras verificables del producto.

Estudios recientes profundizan en los elementos que sustentan esta lógica de aprendizaje:

- **Conocimiento como activo dinámico:** Los equipos ágiles transforman saberes tácitos en “recursos basados en conocimiento” empleando prácticas ligeras (por ejemplo, *pair programming*, *code reviews*), lo que acelera la resolución de problemas (Ouriques, Gorschek, Méndez, & Fagerholm, 2023).
- **Movilización continua del saber:** Más que producir documentación extensa, los proyectos exitosos se apoyan en rituales breves (*dailies*, retrospectivas) y tableros visuales que facilitan la circulación constante de información crítica (Hustad et al., 2022).
- **Seguridad psicológica para aprender:** El aprendizaje iterativo prospera cuando los miembros perciben que pueden equivocarse y discutir errores sin temor a represalias, fortaleciendo la mejora continua (Alami et al., 2023).
- **Factores humanos críticos:** Confianza, comunicación transparente y autonomía explican gran parte de la variabilidad en el rendimiento de equipos ágiles a gran escala (Barros et al., 2024).

En conjunto, estos hallazgos definen una epistemología **emergente, distribuida y contextual**: el conocimiento se cocrea dentro del equipo, se valida con hechos y se adapta al entorno cambiante. Este marco justifica la propuesta de integrar prácticas de seguridad directamente en el flujo ágil, de modo que las lecciones sobre amenazas se aprendan y apliquen con la misma rapidez que cualquier requisito funcional.

2.1.2 *Epistemología del Desarrollo Seguro de Software*

El conocimiento sobre desarrollo seguro de software ha evolucionado en los últimos años desde una visión puramente técnica (listas de controles, auditorías finales), hacia un enfoque sociotécnico, distribuido y empírico.

A continuación, se sintetizan los rasgos epistemológicos que definen esa transformación:

- **Seguridad como propiedad emergente:** Prates y Pereira (2024) demuestran, que las organizaciones con mejores resultados de seguridad son las que desplazan las verificaciones al final del proceso e integran controles automatizados desde el inicio. La evidencia obliga a replantear la epistemología clásica de ciclo en cascada por una orientación iterativa-evolutiva del conocimiento.
- **Conocimiento situado y just-in-time:** Equipos ágiles asimilan conceptos de seguridad justo cuando los necesitan; el saber se distribuye entre historias de usuario, repositorios y pipelines, más que en manuales centralizados (Theurich et al., 2023).
- **Madurez progresiva basada en evidencia:** Lange y Kunz (2024) proponen una “escalera de madurez” que permite a las organizaciones autoevaluarse con criterios cuantificables, desplazando la discusión de “cumpló/no cumpló” a “qué evidencias empíricas justifican mi estado”
- **Alineación con el valor de negocio:** Zhao, Clear y Lal (2024) identifican cinco dimensiones clave de DevSecOps (valor de negocio, automatización, cultura, conformidad y métricas) y concluyen que las decisiones técnicas de seguridad no pueden desvincularse del costo de oportunidad de la entrega continua. De este

modo, la epistemología del desarrollo seguro pasa de ser normativa a ser explicativa-predictiva: se seleccionan prácticas que maximizan la reducción de riesgo por iteración de forma medible.

- **Validación empírica permanente:** La fiabilidad se demuestra con “experimentos de seguridad” (por ejemplo, ataques de *chaos engineering* controlados), lo que desplaza la verificación deductiva a una inducción basada en datos operativos (Prates & Pereira, 2024; Zhao et al., 2024).

En conjunto, estas evidencias configuran una epistemología del desarrollo seguro dinámica y empírica: la seguridad se codifica en pruebas automatizadas, *pipelines*, y su valor se mide por la reducción continua de amenazas sin ralentizar la entrega.

2.2 Bases Teóricas

2.2.1 Metodologías ágiles de desarrollo de software

Las metodologías ágiles combinan una filosofía y un conjunto de prácticas centradas en la entrega temprana, continua y flexible de software funcional. Según Pressman y Maxim (2021), los valores del Manifiesto Ágil priorizan la colaboración, la adaptabilidad y la simplicidad. Este enfoque iterativo fomenta la participación del cliente y equipos autónomos, capaces de adaptarse al cambio y entregar valor de forma incremental (Pressman & Maxim, 2021).

2.2.1.1 Marcos de trabajo (frameworks) ágiles

Existen diversos marcos de trabajo ágiles, como Scrum, XP, Kanban y DevOps. Todos ellos promueven la adaptación continua y una planificación ligera. La agilidad, más allá de la metodología, es una mentalidad que fomenta equipos comunicativos, enfoque en software funcional, integración del cliente y entrega rápida (Pressman & Maxim, 2021).

2.2.1.2 Desarrollo ágil de software y agilidad

Según Özkan y Gok (2022), en el contexto del desarrollo de software, la agilidad se entiende como la capacidad de percibir y responder con rapidez a cambios impredecibles en el entorno (interno y externo) mediante la reconfiguración ágil de recursos, relaciones y capacidades. Esta habilidad incluye no solo adaptarse, sino también reorganizar procesos y estructuras para mantenerse eficaz ante la incertidumbre (Özkan & Gok, 2022).

2.2.1.3 Atributos de agilidad

Según los autores los autores (Maqsood & Bondavalli, 2020) son cuatro los atributos de agilidad:

Flexibilidad: Capacidad de adaptación a los cambios esperados o inesperados en cualquier momento.

Eficiencia: Se refiere a la mejora de los productos y servicios basados en la retroalimentación de los clientes con el objetivo de maximizar el valor y reducir los desperdicios.

Capacidad de Respuesta: Se refiere a la reacción apropiada contra los cambios esperados o inesperados (habilidad de adaptarse rápidamente a los cambios y a las necesidades cambiantes).

Velocidad: La velocidad se refiere al desarrollo rápido e iterativo para versiones pequeñas.

2.2.2 Software Seguro

Allen et al. (2008) definen al software seguro como aquel que ha sido pensado desde el inicio para proteger la información que maneja. Esto significa que su diseño, construcción y mantenimiento toman en cuenta la seguridad en todo momento, no solo al final. Busca evitar fallos, detectar posibles ataques y reaccionar ante ellos rápidamente. Incluso si llega a ser atacado, debe seguir funcionando correctamente y recuperarse lo más rápido posible. En resumen, es un software preparado para resistir amenazas sin poner en riesgo la infraestructura asociada, los datos que procesa, información generada ni a sus usuarios (Allen et al., 2008).

2.2.2.1 Propiedades básicas de software seguro

Para que un software sea verdaderamente seguro, debe cumplir con varias propiedades esenciales a lo largo de todo su ciclo de vida. En primer lugar, debe tener una **ejecución predecible**, es decir, funcionar tal como fue diseñado y resistir entradas maliciosas. También debe ser **fiable**, evitando vulnerabilidades explotables, y cumplir con **normas y estándares establecidos** (conformidad). Además, debe asegurar la **confidencialidad**, protegiendo la información frente a accesos no autorizados; la **integridad**, resistiendo modificaciones maliciosas o accidentales; y la **disponibilidad**, garantizando que los usuarios autorizados puedan acceder al sistema cuando lo necesiten, mientras se bloquea a los no autorizados. Otras propiedades clave incluyen la **responsabilidad**, que permite

rastrear todas las acciones relevantes del sistema, y el **no repudio**, que impide que los usuarios nieguen haber realizado una acción. Todas estas características en conjunto fortalecen la seguridad del software desde su diseño hasta su operación final (OWASP Foundation, n.d.).

2.2.3 Ingeniería de Seguridad de software

La **ingeniería de seguridad de software** consiste en aplicar técnicas y actividades que fortalecen la seguridad del software desde las etapas iniciales de desarrollo (Pressman & Maxim, 2021). Aunque los ingenieros de software no necesitan ser expertos en ciberseguridad, sí deben colaborar estrechamente con profesionales especializados en esta área, ya sea internos o externos al equipo (Pressman & Maxim, 2021). Estas técnicas no solo se centran en proteger sistemas operativos con firewalls o cifrados, sino en **crear software que sea seguro por diseño**, desde la definición de requisitos hasta su implementación (Pressman & Maxim, 2021). Actividades como el modelado de amenazas, los análisis de riesgos, la codificación segura y los casos de uso indebido ayudan a reducir vulnerabilidades desde el principio del ciclo de vida del software (Pressman & Maxim, 2021). Este enfoque proactivo mejora significativamente la calidad y confiabilidad del producto final.

2.2.4 Amenazas de seguridad en el software web según OWASP TOP 10 (2021)

OWASP Top 10 es un documento de concienciación estándar para desarrolladores y seguridad de aplicaciones web. Representa un amplio consenso sobre los riesgos de seguridad más críticos para las aplicaciones web (OWASP Foundation, 2021). Las organizaciones deben adoptar este documento y comenzar el proceso de garantizar que sus aplicaciones web minimicen estos riesgos. Usar OWASP Top 10 es quizás el primer

paso más efectivo para cambiar la cultura de desarrollo de software dentro de su organización a una que produzca código más seguro (OWASP Foundation, 2021), (ver Figura 4).

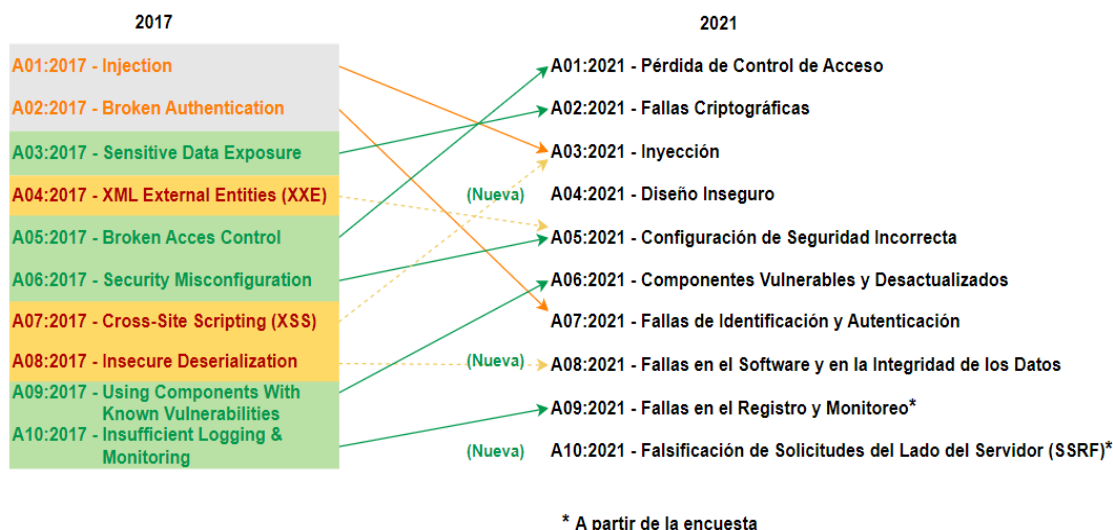


Figura 4. Top 10 de amenazas de seguridad adaptado Fuente. (OWASP Foundation, 2021).

2.2.5 Actividades de seguridad en el desarrollo de software

2.2.5.1 Educación y sensibilización en materia de seguridad

La formación en seguridad es clave para desarrollar software protegido desde el inicio. Involucra capacitar a todo el equipo para identificar y prevenir vulnerabilidades, y promover una cultura que priorice la protección de datos y sistemas. Así, se busca que el software no solo funcione correctamente, sino que también sea resistente a amenazas (Allen et al., 2008). Esta sensibilización debe integrarse en todas las etapas del ciclo de vida del desarrollo.

2.2.5.2 Identificación de Roles

Según la European Union Agency for Cybersecurity (2019), definir claramente los roles y perfiles de usuario es esencial para la seguridad del software. Esta práctica establece quién puede hacer qué dentro del sistema, aplicando el principio de mínimo privilegio. Así se reduce el riesgo de accesos indebidos o abusos internos, y se facilita el control, auditoría y seguimiento. Una correcta asignación de roles fortalece la protección de los recursos del sistema.

2.2.5.3 Análisis de riesgos de seguridad (Identificación de ataques)

El análisis de riesgos de seguridad (identificación de ataques), es una etapa clave en el desarrollo de software seguro, ya que permite detectar y analizar posibles amenazas antes de que causen daño. Según OWASP Foundation (2021), este análisis ayuda a reconocer tácticas comunes de los atacantes y anticipar fallos en el sistema. Al integrarse desde el diseño y la codificación, permite reforzar la seguridad del software de forma proactiva. Así se minimizan los riesgos y se protege la aplicación frente a ataques maliciosos.

2.2.5.4 Análisis de requisitos de seguridad

Según Mažeika y Butleris (2020), los requisitos de seguridad deben abordarse desde las primeras fases del desarrollo, mediante actividades como la elicitación, especificación y análisis. Estos requisitos actúan como restricciones que complementan a los requisitos funcionales, con el objetivo de evitar daños reales. A diferencia de la seguridad funcional (safety), que se enfoca en fallos accidentales, la seguridad (security) trata amenazas intencionales, como ataques cibernéticos. Además, mientras los fallos de safety afectan al entorno físico del sistema, los de seguridad pueden comprometer tanto

el sistema como su entorno. Por eso, distinguir ambos enfoques es clave para proteger eficazmente los activos digitales (Mažeika & Butleris, 2020).

2.2.5.5 Modelado de amenazas

El **modelado de amenazas** es una práctica clave para anticipar posibles ataques o vulnerabilidades en el software antes de que sea implementado. Según OWASP Foundation (n.d.), este proceso permite identificar y entender las amenazas desde las etapas tempranas del desarrollo, facilitando la integración de medidas de seguridad directamente en el diseño del sistema.

El proceso comienza descomponiendo la aplicación: se analizan sus usos, puntos de entrada, activos valiosos y niveles de acceso, lo cual suele representarse mediante Diagramas de Flujo de Datos (DFD). Luego, se identifican y clasifican las amenazas usando métodos como STRIDE (para tipificar amenazas como suplantación o alteración) y DREAD (para evaluar su impacto y probabilidad). Por último, se determinan contramedidas como eliminar, mitigar o aceptar los riesgos, priorizando según la gravedad. Este enfoque estructurado permite construir software más robusto y resistente a ataques (OWASP Foundation, n.d.).

2.2.5.6 Análisis estático de código

El **análisis de código estático** es una técnica usada durante la fase de implementación del desarrollo seguro, y forma parte de las pruebas de caja blanca. Según OWASP Foundation (n.d.), consiste en revisar el código fuente sin ejecutarlo, usando herramientas que detectan posibles vulnerabilidades mediante análisis de flujo de datos o patrones inseguros. Aunque estas herramientas no reemplazan al analista, sí lo ayudan a identificar de forma más eficiente errores que podrían ser explotados por atacantes. Su uso permite reforzar la seguridad del software desde el momento en que se escribe el código.

2.2.5.7 Pruebas de penetración

Las **pruebas de penetración** son una técnica de evaluación de seguridad que simula ataques reales para descubrir vulnerabilidades en sistemas o aplicaciones. Según OWASP Foundation (2023), estas pruebas se realizan sin conocer el funcionamiento interno del software (caja negra), imitando a un atacante externo. Los evaluadores actúan como usuarios legítimos para detectar fallos y, en ocasiones, se les da acceso para explorar más a fondo. Es una práctica clave para fortalecer la seguridad antes de que ocurran ataques reales.

2.2.5.8 Planificación de respuesta a incidentes

La **planificación de respuesta a incidentes** es fundamental para enfrentar eventos de seguridad en el desarrollo de software. Consiste en preparar un plan claro para identificar, contener, erradicar y recuperarse ante incidentes, además de aprender de ellos para mejorar futuras respuestas (Sharma & Bawa, 2020). Shinde y Kulkarni (2021) proponen que esta planificación debe ser flexible y adaptativa, integrándose en todo el ciclo de vida del software seguro (SSDLC). Esto es especialmente importante en entornos ágiles, donde los cambios constantes requieren respuestas rápidas y eficaces. Aplicar este enfoque reduce el impacto de los ciberataques y fortalece la capacidad de reacción organizacional (Shinde & Kulkarni, 2021).

2.2.6 Modelos de categorización y clasificación de amenazas

Tarandach y Coles (2020) proponen un enfoque práctico de modelado de amenazas adaptado al desarrollo ágil y despliegue continuo. Su método busca que todo el equipo participe, sin importar su experiencia en seguridad. Para ello, recomiendan aplicar técnicas como STRIDE y DREAD de forma iterativa y flexible durante todo el ciclo de vida del software.

2.2.6.1 **STRIDE**

El acrónimo STRIDE representa los siguientes tipos de amenazas: Suplantación de identidad (Spoofing), Manipulación (Tampering), Repudio (Repudiation), Información (Information Disclosure), Denegación de servicio (Denial of Service) y Elevación de privilegios (Elevation of Privilege). Este marco conceptual es una herramienta fundamental para los profesionales de seguridad, ya que proporciona una metodología estructurada para analizar y mitigar los riesgos de seguridad desde la fase de diseño de un sistema (Tarandach & Coles, 2020).

2.2.6.2 **DREAD**

El modelo DREAD es un marco que ayuda a clasificar y priorizar riesgos de seguridad en el desarrollo de software, evaluando cinco aspectos clave: Daño (*Damage potential*), Reproducibilidad (*Reproducibility*), Explotabilidad (*Exploitability*), Afectados (*Affected users*), y Descubrimiento (*Discoverability*) (Tarandach & Coles, 2020). Cada uno de estos factores se califica con una puntuación, lo que permite estimar cuán grave puede ser una amenaza. Esta evaluación ayuda a los equipos a decidir qué vulnerabilidades atender primero y cómo distribuir mejor los recursos. Por ejemplo, un problema de seguridad (inyección de código SQL malicioso en el campo de usuario del formulario de inicio de sesión para acceder como administrador), puede tener puntuaciones de (D = 3, R = 3, E = 2, A = 3, D = 2) para un valor de riesgo total de 13. Este valor indica que es una amenaza **alta** y debe ser **priorizada urgentemente** para su mitigación (Tarandach & Coles, 2020).

2.2.6.3 **Modelo ágil de gestión de riesgo**

El modelo ágil de gestión de riesgos busca integrar la seguridad dentro del desarrollo ágil de software, promoviendo la identificación temprana y continua de riesgos y su

tratamiento de forma iterativa (Ouriques, Machado, & Vicari, 2023). Se basa en la colaboración entre equipos, en la priorización de riesgos según su exposición, y en fomentar una cultura de seguridad a todos los niveles. Este enfoque permite equilibrar la flexibilidad del desarrollo ágil con la atención oportuna a las amenazas. Se recomienda usar la metodología de evaluación de riesgos de OWASP (https://owasp.org/www-community/OWASP_Risk_Rating_Methodology), ya que es ligera y considera aspectos técnicos y de negocio. Herramientas web como <https://www.owasp-risk-rating.com/> y <https://javierolmedo.github.io/OWASP-Calculator/> ayudan a agilizar esta tarea.

2.2.7 Diagrama de flujo de datos (DFD)

El Diagrama de Flujo de Datos (DFD) es una herramienta clave en el modelado de amenazas porque permite visualizar cómo circulan los datos dentro de un sistema. Según OWASP Foundation (n.d.), los DFD muestran los procesos, puntos de entrada y salida, y almacenes de datos, ayudando a detectar posibles puntos vulnerables. Esta visualización facilita la identificación de controles de seguridad necesarios para proteger la información. Además, los DFD son útiles para analizar las fronteras de confianza, es decir, los límites entre partes seguras y no seguras del sistema. Su uso mejora el análisis de riesgos desde etapas tempranas del desarrollo.

2.2.8 Historias de abuso

González y Pérez (2023) proponen el uso de historias de abuso como una herramienta útil para identificar requisitos de seguridad desde las primeras fases del desarrollo de software. Estas historias describen posibles acciones maliciosas de un atacante, lo que permite anticiparse a riesgos y diseñar controles adecuados (Andrade et al., 2023). Por ejemplo, Una historia de abuso podría describir cómo un atacante podría realizar una inyección SQL para acceder a la base de datos de un sistema informático web, o cómo

podrían explotar una configuración débil de autenticación para obtener acceso no autorizado. Además, pueden adaptarse fácilmente a metodologías ágiles como Scrum, mejorando la respuesta frente a vulnerabilidades sin frenar la agilidad del proyecto (Andrade, Torres, Ortiz-Garcés, Miño, & Almeida, 2023). Su aplicación sistemática mejora tanto la protección como la conciencia de seguridad del equipo (Andrade et al., 2023).

2.2.9 Historias de usuario relacionadas a la seguridad

Las historias de usuario relacionadas con la seguridad en el desarrollo de software son descripciones breves y funcionales que se centran en las características de seguridad necesarias para proteger a los usuarios y la aplicación de posibles amenazas. Estas historias ayudan a incorporar requisitos de seguridad de forma integrada en el proceso de desarrollo ágil, asegurando que las funcionalidades de seguridad se consideren desde el principio del ciclo de vida del desarrollo (Rindell et al., 2021) (Khan et al, 2022). Un ejemplo podría ser: "Como usuario de la plataforma bancaria en línea, quiero que se me solicite una verificación de dos factores al iniciar sesión, para asegurar que solo yo pueda acceder a mi cuenta" (McGraw, 2006). Otro ejemplo sería: "Como cliente de una aplicación de salud, quiero que mis datos personales y registros médicos estén encriptados, para que mi información confidencial esté protegida contra accesos no autorizados" (Rindell et al., 2021) (Khan et al, 2022).

2.2.10 Metodologías de Desarrollo de software Seguro

2.2.10.1 Proceso integral ligero de seguridad de aplicaciones (CLASP)

El modelo **CLASP (Proceso Integral y Ligero de Seguridad de Aplicaciones)**, desarrollado por MITRE Corporation (n.d.), es un enfoque estructurado que busca integrar la seguridad desde las primeras etapas del desarrollo de software. Se compone de un conjunto de buenas prácticas y actividades fácilmente adaptables a cualquier

proceso de desarrollo (MITRE Corporation, n.d.). CLASP se basa en siete prácticas clave, como la concientización, evaluaciones de aplicaciones, captura de requisitos de seguridad, implementar prácticas de desarrollo seguro, construir procedimientos de corrección de vulnerabilidades, definir y monitorear métricas, publicar directrices de seguridad operativa (Oyetoyan, Jaatun, & Cruzes, 2017). Además, ofrece 24 actividades divididas por roles específicos, facilitando su implementación por parte de gestores, arquitectos, desarrolladores o auditores. También considera la protección de servicios esenciales como la autenticación, autorización, confidencialidad, disponibilidad, responsabilidad y no repudio. Este enfoque prescriptivo promueve la adopción de herramientas automatizadas para facilitar su ejecución (MITRE Corporation, n.d.). La Figura 5 muestra el proceso del modelo CLASP.

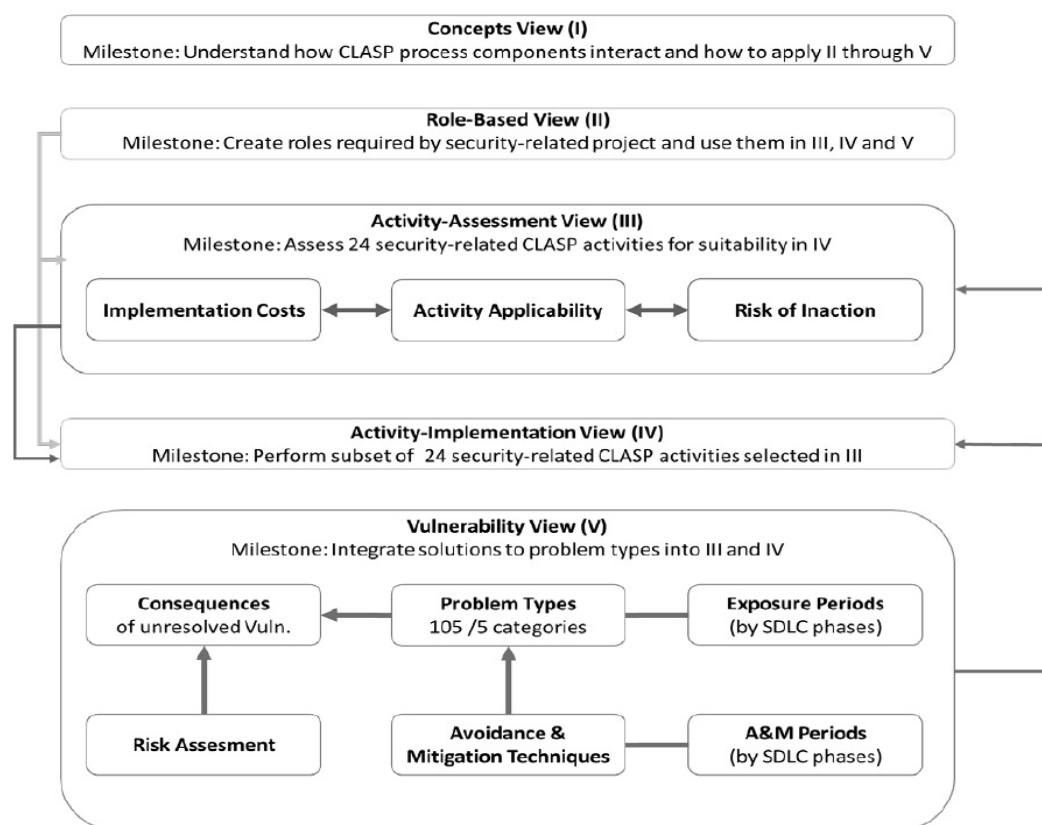


Figura 5. Prácticas y actividades CLASP (Oyetoyan et al., 2017).

2.2.10.2 Ciclo de vida del desarrollo seguro de Microsoft (SDL)

El proceso de ciclo de vida de desarrollo seguro de Microsoft (SDL) es un proceso de desarrollo de software utilizado y propuesto por Microsoft para reducir los costos de mantenimiento de software y aumentar fiabilidad del software respecto a la seguridad relacionados con los errores de software (Herath & Wimalaratne, 2018). Se basa en el modelo espiral clásico, pero hay intentos de hacer más ágil posible (Sharma & Bawa, 2020). La Figura 6 muestra las actividades propuestas de Microsoft Security Development Lifecycle Process (Herath & Wimalaratne, 2018).

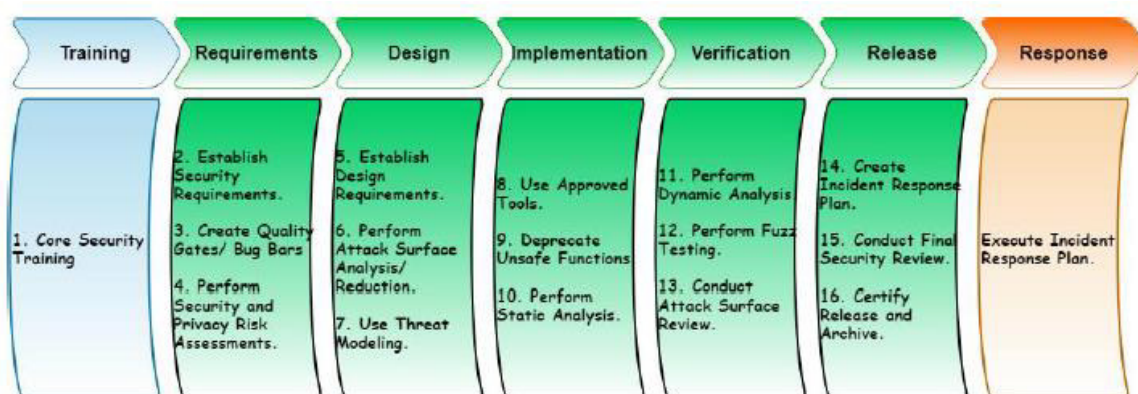


Figura 6. Actividades de seguridad propuesta por Microsoft Security Development Lifecycle Process.

2.2.10.3 Criterios comunes (CC)

El estándar **Common Criteria (CC)** tiene como finalidad establecer un marco común para evaluar y garantizar la seguridad de productos y procesos de software, promoviendo la comparabilidad entre organizaciones y aplicaciones (Rindell, 2019; Moyo & Mnkandla, 2020). Este enfoque ayuda a definir objetivos claros de seguridad, aplicar mecanismos para lograrlos y evaluar el grado de cumplimiento alcanzado. Common Criteria formaliza la **ingeniería de seguridad** y el **aseguramiento de la seguridad** desde un punto de vista estructurado (Rindell, 2019). A través de su marco

metodológico, permite que la seguridad del software no solo sea implementada, sino también **medible y verificable**. Además, sirve como base para que los desarrolladores alineen sus procesos con requisitos internacionales de ciberseguridad (Moyo & Mnkanla, 2020). En la Figura 7 se muestra las actividades que contempla el estándar de criterios comunes.

CODE	CLASS NAME	FUNCTIONAL SUMMARY
FAU	Security audit	Auditing data, events and analysis
FCO	Communication	Non-repudiation of origin and receipt
FCS	Cryptographic support	Key management and operations
FDP	User data protection	Data and communications protection
FIA	Identification and authentication	User authentication Failure management
FMT	Security Management (normative)	Security roles, attributes and functions
FPR	Privacy	Anonymity and observability
FPT	Protection of the TSF	Technical protection Secure failure and recovery
FRU	Resource utilization	Fault tolerance, priorities, resources
FTA	TOE Access	Access session and history management
FTP	Trusted path/channels	Inter-TSF trusted channels and paths

Figura 7. Actividades que contempla el estándar de criterios comunes (Common Criteria) (Rindell, 2019).

2.2.10.4 Puntos de contacto de seguridad del software (CT)

Los touchpoints o puntos de contacto de seguridad del software son buenas prácticas que surgieron en 2004 (publicadas por primera vez en IEEE Security & Privacy) y han sido ampliamente adoptadas por organizaciones como el Departamento de Seguridad Nacional de EE. UU. (Rindell et al., 2021; Khan et al., 2022). Se consideran uno de los tres pilares clave de la seguridad del software, junto con la gestión de riesgos aplicada y el conocimiento, los cuales deben implementarse de forma gradual y equilibrada (Rindell et al., 2021; Khan et al., 2022). Estas prácticas incluyen tanto actividades **ofensivas** (como pruebas de penetración o análisis de abusos) como **defensivas** (como revisión de código, análisis de arquitectura o requisitos de seguridad), lo que permite

una protección integral del software (Adel, Alqatawna, & Mohammad, 2017). Herath y Wimalaratne (2018) destacan que estos touchpoints se integran directamente en los artefactos del desarrollo, ayudando a fortalecer cada fase del ciclo de vida del software con un enfoque práctico y realista (Ver Figura 8).

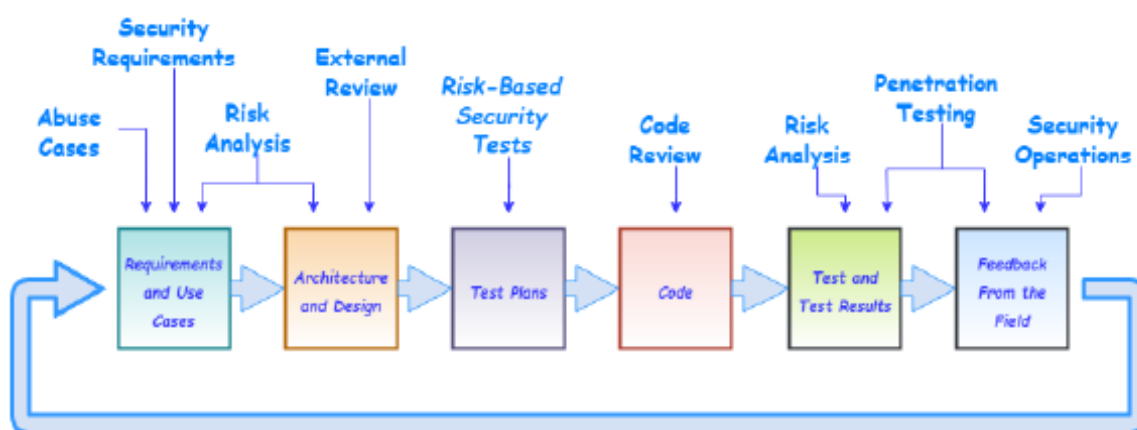


Figura 8. TouchPoints (Herath & Wimalaratne, 2018).

CAPÍTULO III. ESTADO DEL ARTE

Por su extensión, el estado del arte se presenta como un capítulo autónomo y está organizado en cinco secciones: (1) Revisión sistemática de la literatura, (2) Planificación de la revisión de literatura, (3) Ejecución de la revisión de literatura, (4) Resultados y análisis de la revisión de literatura y (5) Conclusiones de la revisión de la literatura.

3.1 Revisión sistemática de la literatura

Con el fin de sustentar el marco teórico y las decisiones metodológicas, se llevó a cabo una revisión sistemática de la literatura (RSL) siguiendo la estrategia en tres fases propuesta por Julca Mejía (2024): planificación, ejecución y análisis de resultados. La RSL no solo consolida el estado del arte, sino que también guía la selección de las ocho actividades de seguridad introducidas en el marco de trabajo propuesto y proporciona la base empírica para su validación posterior.

3.2 Planificación de la revisión de la literatura

3.2.1 *Objetivos de la revisión*

La presente revisión sistemática de la literatura tiene los siguientes objetivos:

- O1:** Identificar las prácticas de desarrollo de software seguro integradas en metodologías ágiles.
- O2:** Identificar el grado de agilidad (nivel de adaptación), de las prácticas de desarrollo de software seguro al ser incorporadas en metodologías ágiles.
- O3:** Identificar el costo-beneficio de integrar prácticas de desarrollo de software seguro en metodologías ágiles.

3.2.2 *Preguntas de investigación*

Para orientar la revisión sistemática, se establecieron las siguientes preguntas de investigación:

- Q1:** ¿Cuáles son las prácticas de desarrollo de software seguro que han sido integradas en metodologías ágiles, según la literatura existente?
- Q2:** ¿Cuál es el grado de agilidad (nivel de adaptación), de las prácticas de desarrollo de software seguro al ser incorporadas en metodologías ágiles, según la literatura existente?

Q3: ¿Cuáles es el costo-beneficio de integrar prácticas de desarrollo de software seguro en metodologías ágiles, según la literatura existente?

3.2.3 Cadenas de búsqueda

La cadena de búsqueda se diseñó combinando términos clave relacionados con los objetivos y preguntas de investigación: (“*secure software development*” OR “*security practices*” OR “*security activities*”) AND (“*agile methodologies*” OR “*agile process models*” OR “*Scrum*” OR “*agile development*”) AND (“*cost-benefit*” OR “*degree of agility*”) aplicados a los campos de Título, Resumen y Palabras clave, (ver Cuadro 1).

Cuadro 1. Cadenas de búsqueda aplicados a los campos de los bancos de datos.

Cadena de Búsqueda	Campo de Búsqueda
(“ <i>secure software development</i> ” OR “ <i>security practices</i> ” OR “ <i>security activities</i> ”)	Title, Abstract and Keywords
(“ <i>agile methodologies</i> ” OR “ <i>agile process models</i> ” OR “ <i>Scrum</i> ” OR “ <i>agile development</i> ”)	Title, Abstract and Keywords
(“ <i>cost-benefit</i> ” OR “ <i>degree of agility</i> ”)	Title, Abstract and Keywords

Fuente. Elaboración propia.

3.2.4 Fuentes

Las fuentes de datos consideradas en esta revisión sistemática de la literatura son:

- ACM Digital Library
- IEEE Xplore
- Springer
- SCOPUS
- ScienceDirect

3.2.5 Criterios de inclusión y exclusión

Los criterios de inclusión y exclusión utilizados en el proceso de revisión de la literatura para este estudio se muestran en el Cuadro 2.

Cuadro 2. Criterios de inclusión y exclusión.

Criterios de Inclusión	Criterios de Exclusión
Artículos que aborden el desarrollo de software seguro en metodologías ágiles.	Artículos que no aborden específicamente la integración de seguridad en metodologías ágiles.
Artículos científicos publicados en revistas y conferencias indexadas	Artículos que no estén relacionados con el desarrollo de software seguro
Investigaciones que evalúen el grado de agilidad de las prácticas de seguridad en entornos ágiles.	Estudios que no proporcionen datos cuantitativos o cualitativos sobre el grado de agilidad o el costo-beneficio de las prácticas de seguridad en metodologías ágiles
Investigaciones que analicen el costo-beneficio de incorporar prácticas de seguridad en metodologías ágiles.	Artículos que no aborden específicamente la integración de seguridad en metodologías ágiles.
Investigaciones en inglés	
Artículos publicados entre enero 2017 y diciembre 2022.	

Fuente. Elaboración propia.

3.3 Ejecución de la revisión de la literatura

3.3.1 Búsqueda de fuentes

Durante esta etapa, se emplearon las fuentes de información establecidas en la fase de planificación, conforme a los criterios de selección (inclusión/exclusión), consignados en el Cuadro 9. La Figura 9 detalla el diagrama del proceso de búsqueda bibliográfica.

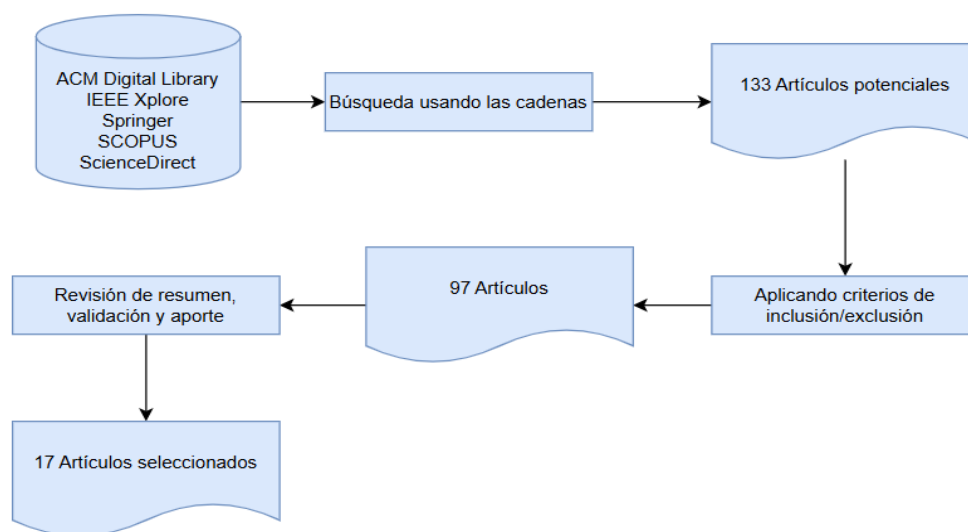


Figura 9. Flujo de la revisión de la literatura.

3.3.2 Selección de artículos

Se encontraron 133 artículos potenciales después de realizar una búsqueda inicial de artículos en las bases de datos de ACM Digital Library, IEEE Xplore, Springer, SCOPUS y ScienceDirect. Posteriormente, luego de aplicar criterios de inclusión y exclusión y eliminar los duplicados la lista se redujo a 97 artículos. Finalmente, luego de la revisión de resúmenes, validación y aportes la lista se redujo a 17 artículos. El Cuadro 3 muestra los artículos seleccionados.

Cuadro 3. Cadenas de búsqueda aplicados a los campos de los bancos de datos.

Base de Datos (Fuentes)	Artículos Potenciales	Artículos Seleccionados
ACM Digital Library	22	4
SCOPUS	83	8
IEEE Xplore	16	3
ScienceDirect	8	0
SpringerLink	4	2

Fuente. Elaboración propia.

3.3.3 Artículos científicos relacionados con el trabajo de investigación

De la revisión sistemática de la literatura con relación a las prácticas de desarrollo de software seguro y su integración con las metodologías ágiles, se destacan los siguientes artículos científicos que abordan el tema de la presente investigación.

3.3.3.1 *Cómo influir en la priorización de la seguridad de un proyecto de desarrollo ágil de software*

Tøndel et al. (2022) estudiaron cómo lograr que la seguridad sea una prioridad en equipos ágiles, especialmente en pequeñas empresas donde no hay personal especializado. Descubrieron cinco factores fundamentales que permiten que la seguridad se integre: impulso, visibilidad, motivación, margen de acción y compatibilidad con el proceso. En vez de imponer reglas, sugieren trabajar colaborativamente y adaptar las prácticas de seguridad al equipo y al contexto.

3.3.3.2 *Hacia un proceso de desarrollo de software ágil y seguro: un modelo basado en la práctica*

Ardo, Bass y Gaber (2022) proponen un modelo de 26 actividades de seguridad que se integran en todo el ciclo ágil (planificación, requisitos, diseño, desarrollo, pruebas y despliegue). Además, introducen el concepto de “*compliance sprint*”, que garantiza que la seguridad no se pase por alto sin afectar la agilidad.

3.3.3.3 *Modelo de aseguramiento de seguridad para el desarrollo de software dirigido a proveedores globales de desarrollo de software*

Khan et al. (2022) desarrollaron el modelo SAM, una guía estructurada para organizaciones que desarrollan software a nivel internacional. El modelo identifica 46

riesgos clave y 388 prácticas de seguridad, distribuidas en siete niveles, lo que permite aplicar mejoras graduales y adaptadas al contexto.

3.3.3.4 Sintetizar las actividades de desarrollo de software seguro para los modelos de ciclo de vida lineal y ágil

Alenezi et al. (2022) recopilan y sintetizan prácticas seguras desde la fase de requerimientos hasta mantenimiento. Lo innovador es que ofrecen una tabla comparativa de actividades validadas por expertos, que combina metodologías como MSSDL-A, Baca, Rindell, Bezerra y Maier.

3.3.3.5 Mejora de la seguridad del software en la metodología de programación extrema

Ababneh et al. (2021) adaptaron prácticas de seguridad al marco de programación extrema. Agregar roles de seguridad, pruebas de vulnerabilidades y diseño de amenazas puede aumentar el esfuerzo inicial, pero reduce significativamente el mantenimiento posterior y mejora la calidad del producto.

3.3.3.6 Integración de seguridad y privacidad en HCD-Scrum

Baldassarre et al. (2021) desarrollaron el marco HCD-Security Scrum, que incorpora principios de diseño centrado en el ser humano. Introducen la herramienta VIS-PRISE para facilitar decisiones seguras en las etapas de diseño y validación, apoyando especialmente los aspectos de privacidad.

3.3.3.7 Análisis costo-beneficio de la incorporación de seguridad y evaluación de sus efectos en las distintas fases del desarrollo de software ágil

Kumar et al. (2021) demostraron, con datos, que invertir en seguridad desde el principio del desarrollo ágil reduce el riesgo financiero, las multas y el retrabajo. También presentan métricas para calcular el ROI de las medidas de seguridad.

3.3.3.8 Adopción del modelado de amenazas en proyectos de desarrollo de software ágil

Bernsmed et al. (2021) evaluaron cómo se implementa el modelado de amenazas en equipos ágiles. Aunque existen herramientas como STRIDE, los equipos carecen de tiempo y experiencia. Recomiendan la inclusión de expertos para facilitar este tipo de análisis desde etapas tempranas.

3.3.3.9 Seguridad en el desarrollo de software ágil: una encuesta a profesionales

Rindell et al. (2021) encuestaron a 61 desarrolladores y encontraron que la mayoría valora más las prácticas tempranas como el análisis de requisitos y modelado de amenazas. Concluyen que la clave es integrar la seguridad sin frenar la agilidad, priorizando acciones de alto valor.

3.3.3.10 ¿Cómo integrar los requisitos de cumplimiento de seguridad con la ingeniería de software ágil a escala?

Moyón et al. (2020) desarrollaron el modelo S2C-SAFe, que adapta el marco SAFe para cumplir con la norma de seguridad IEC 62443-4-1. Este modelo ayuda a grandes organizaciones como Siemens a alinear agilidad y cumplimiento regulatorio sin fricción.

3.3.3.11 Agilidad de las prácticas de seguridad y modelos de procesos ágiles: una evaluación del costo de la incorporación de la seguridad en los modelos de procesos ágiles

Maqsood y Bondavalli (2020) analizaron cómo 11 prácticas de seguridad afectan la agilidad de seis metodologías (Scrum, XP, FDD, DSDM, etc.). Usando encuestas con escala Likert, identificaron las actividades más compatibles con atributos ágiles como flexibilidad y rapidez.

3.3.3.12 Identificación e integración de actividades de seguridad para un desarrollo ágil seguro

Sharma y Bawa (2020) utilizaron métodos multicriterio (AHP y PROMETHEE) para seleccionar 16 prácticas de seguridad con bajo costo y alto beneficio para integrar en desarrollo ágil. El marco resultante guía la priorización eficiente de actividades de seguridad.

3.3.3.13 Una metodología novedosa y ligera de desarrollo de software con prácticas de seguridad óptimas

Moyo y Mnkandla (2020) diseñaron Secure-SSDM, un enfoque para desarrolladores solitarios que combina seguridad, calidad y agilidad. Incluye prácticas ligeras como revisiones de código simuladas y pruebas automatizadas. Es útil, pero requiere mayor capacitación.

3.3.3.14 Hacia un marco conceptual para el trabajo de requisitos de seguridad en el desarrollo ágil de software

Tøndel y Jaatun (2020) proponen un marco conceptual que aborda cómo los equipos ágiles perciben y abordan los requisitos de seguridad. Destacan la importancia de los factores humanos, como la presión por entregar rápido y la comunicación entre roles.

3.3.3.15 Aplicación de un nuevo ciclo de vida de desarrollo de software seguro (S-SDLC) con metodologías ágiles

Mohino et al. (2019) presentan el modelo S-SDLC, que incorpora la seguridad desde el inicio del ciclo de vida ágil. El modelo promueve detección temprana de vulnerabilidades sin afectar cronogramas ni costos, aumentando la calidad del software entregado.

3.3.3.16 Adaptación de la seguridad al desarrollo ágil de software

Rindell et al. (2018) vinculan prácticas de ingeniería de seguridad con procesos de XP y Scrum. Mapean actividades como codificación segura, TDD y revisiones al ciclo de vida seguro, mostrando que se pueden incorporar sin frenar la productividad.

3.3.3.17 Hacia un proceso SCRUM seguro para el desarrollo ágil de aplicaciones web

Maier et al. (2017) proponen Secure Scrum, una versión adaptada del marco Scrum que incorpora actividades como modelado de amenazas, planificación de respuesta ante incidentes y pruebas de penetración. Evaluado en proyectos reales, demostró ser útil y equilibrado.

3.4 Resultados y análisis de la revisión de la literatura

En el proceso de búsqueda de artículos en varias bases de datos, SCOPUS obtuvo la mayor cantidad inicial, mientras que ACM Digital Library recibió la segunda mayor cantidad, seguido por IEEE Xplore, ScienceDirect y SpringerLink, en ese orden, (ver Figura 10 y Figura 11). Finalmente se seleccionaron ocho artículos de SCOPUS, cuatro de la biblioteca Digital Library de ACM, tres de IEEE Xplore y dos de SpringerLink.

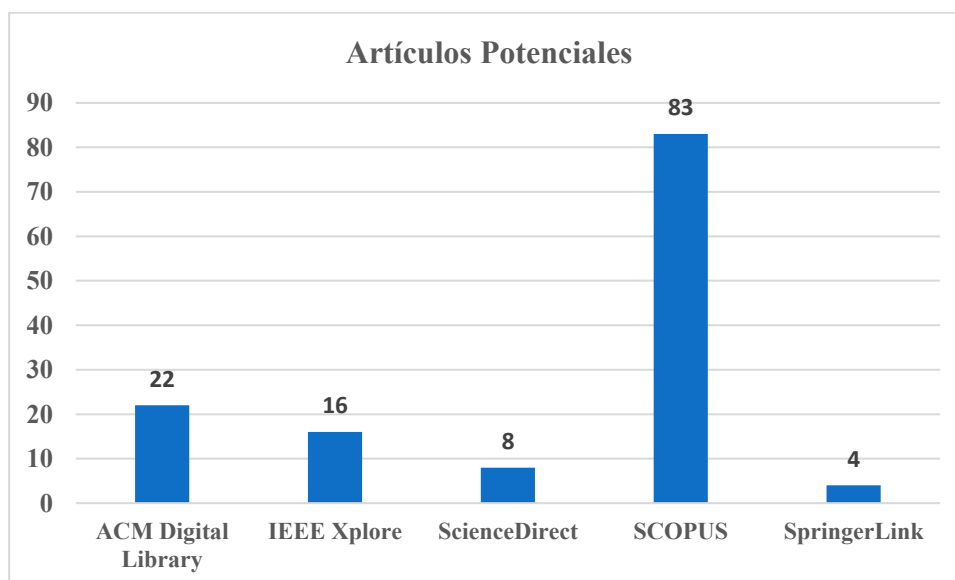


Figura 10. Artículos potenciales.

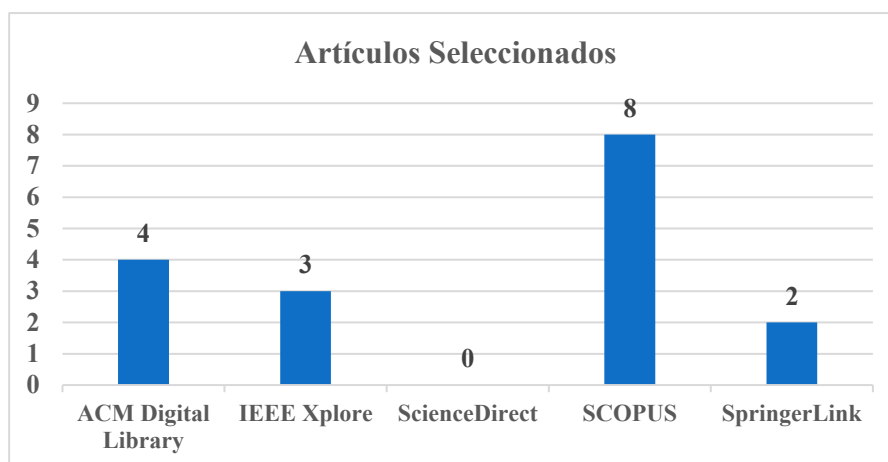


Figura 11. Artículos seleccionados.

3.4.1 Q1: Identificación de prácticas de seguridad en metodologías ágiles

De la revisión de la literatura se evidencia que estos trabajos de investigación proponen, analizan y demuestran la incorporación de prácticas de seguridad durante el proceso de desarrollo usando metodologías ágiles, por ejemplo: Tøndel et al. (2022) estudiaron cómo un especialista en seguridad influye en equipos ágiles, proponiendo un modelo de cinco factores que ayudan a priorizar la seguridad sin imponer reglas. Ardo et al. (2022) desarrollaron un modelo incluyendo 26 prácticas de seguridad integradas en cada fase del desarrollo ágil, e incluyeron una ceremonia llamada “*compliance sprint*”. Khan et al. (2022) propusieron el modelo *Security Assurance Model* - SAM, una guía de 7 niveles con 388 prácticas de seguridad para empresas globales, validada en entornos reales. Alenezi et al. (2022) sintetizaron buenas prácticas de seguridad para modelos ágiles y lineales, validadas por expertos y aplicables a múltiples fases del ciclo de vida del software. Ababneh et al. (2021) integraron prácticas de seguridad en Programación Extrema (XP), logrando reducir riesgos y mantenimiento a largo plazo, pese al esfuerzo inicial. Baldassarre, Caivano y Piccinno (2021) proponen el modelo HCD-Security Scrum, que une seguridad, privacidad y diseño centrado en el usuario, apoyado por una herramienta llamada VIS-PRISE. Kumar et al. (2021) demostraron que invertir en seguridad desde el inicio reduce hasta un 40 % los costos futuros y mejora el retorno de

inversión (ROI). Bernsmed et al. (2021) exploraron cómo se adopta el modelado de amenazas en entornos ágiles, destacando desafíos y el valor de contar con expertos en seguridad. Rindell et al. (2021) encuestaron a desarrolladores finlandeses y hallaron que las prácticas de seguridad tempranas son más valoradas, aunque no siempre bien integradas. Moyón et al. (2020) proponen el marco de trabajo llamado S2C-SAFe, que adapta el estándar IEC 62443-4-1 a SAFe, facilitando la seguridad en desarrollos ágiles a gran escala. Maqsood y Bondavalli (2020) evaluaron la agilidad de 11 prácticas de seguridad en distintos modelos ágiles; algunas como *threat modeling* muestran alta adaptabilidad. Sharma y Bawa (2020) aplicaron técnicas como AHP y PROMETHEE para seleccionar 16 prácticas de seguridad con bajo costo y alto beneficio para equipos ágiles. Moyo & Mnkandla (2020) diseñaron el marco de trabajo Secure-SSDM para desarrolladores individuales, combinando calidad y seguridad sin perder agilidad. Tøndel & Jaatun (2020) presentan un marco conceptual que explica cómo factores humanos y organizacionales afectan el manejo de requisitos de seguridad en ágil. Mohino, Bermejo, Bermejo y Sicilia (2019) propusieron un modelo llamado S-SDLC que mejora la detección temprana de vulnerabilidades sin aumentar tiempos ni costos del proyecto. Rindell, Hyrynsalmi y Leppänen (2018) mapearon prácticas de seguridad en artefactos y procesos de XP y Scrum para facilitar su integración sin frenar el ritmo ágil. Maier et al. (2017) desarrollaron el modelo Secure Scrum, un proceso con enfoque ágil y seguro alineado al estándar SSE-CMM (*Systems Security Engineering Capability Maturity Model*), calificado como medianamente ágil y costo-efectivo, (ver Cuadro 4).

Cuadro 4. Prácticas de seguridad en metodologías ágiles de acuerdo con la revisión sistemática de la literatura.

Nº	Prácticas de Seguridad en Metodologías Ágiles	Referencia
1	<i>Influencing the security prioritisation of an agile software development project</i>	Tøndel et al. (2022)
2	<i>Towards Secure Agile Software Development Process: A Practice-Based Model</i>	Ardo et al. (2022)

3	<i>Security Assurance Model of Software Development for Global Software Development Vendors</i>	Khan, Khan, Alzahrani & Ilyas (2022)
4	<i>Synthesizing secure software development activities for linear and agile lifecycle models</i>	(Alenezi et al., 2022)
5	<i>Improving Software Security in Extreme Programming Methodology</i>	(Ababneh et al., 2021)
6	<i>Integrating Security and Privacy in HCD-Scrum</i>	(Baldassarre, Caivano, & Piccinno, 2021)
7	<i>Cost Benefit Analysis of Incorporating Security and Evaluation of Its Effects on Various Phases of Agile Software Development</i>	Kumar, Caer, Jolly, Baz y Cheikhrouhou (2021)
8	<i>Adopting threat modelling in agile software development projects</i>	Bernsmed et al. (2021)
9	<i>Security in agile software development: A practitioner survey</i>	Rindell et al. (2021)
10	<i>How to Integrate Security Compliance Requirements with Agile Software Engineering at Scale?</i>	Moyón et al.,2020
11	<i>Agility of Security Practices and Agile Process Models: An Evaluation of Cost for Incorporating Security in Agile Process Models</i>	(Maqsood & Bondavalli, 2020)
12	<i>Identification and integration of security activities for secure agile development</i>	(Sharma & Bawa,2020)
13	<i>A Novel Lightweight Solo Software Development Methodology with Optimum Security Practices</i>	(Moyo & Mnkandla, 2020)
14	<i>Towards a Conceptual Framework for Security Requirements Work in Agile Software Development</i>	(Tøndel & Jaatun, 2020)
15	<i>The Application of a New Secure Software Development Life Cycle (S-SDLC) with Agile Methodologies</i>	(Mohino et al., 2019)
16	<i>Fitting Security into Agile Software Development</i>	(Rindell et al., 2018)
17	<i>Towards a Secure SCRUM Process for AgileWeb Application Development</i>	(Maier, Ma, & Bloem, 2017)

Fuente. Elaboración propia.

3.4.2 Q2: Evaluación del grado de agilidad de las prácticas de seguridad

De los estudios de investigación revisados se han identificado estudios que proponen la inclusión de prácticas de seguridad en el desarrollo de software en metodologías ágiles y además contemplan la evaluación de adaptabilidad de estas prácticas durante todo el ciclo de desarrollo, donde se destaca tanto los desafíos como las estrategias para mantener el equilibrio entre agilidad y seguridad. A continuación, se describen los principales hallazgos: Ardo et al. (2022) proponen un modelo que integra 26 prácticas de seguridad en el proceso de desarrollo ágil, adaptándolas a roles como DevOps o equipos autoorganizados, e introducen la ceremonia “*compliance sprint*” logrando su adopción sin afectar el ritmo ágil. Alenezi et al. (2022) sintetizan prácticas de seguridad para modelos ágiles y lineales, verificadas por expertos. Identifican que, en la fase de requerimientos, prácticas como el análisis de riesgos ágiles y la definición de requisitos de seguridad se adaptan bien a la interactividad y flexibilidad de los enfoques ágiles. Kumar et al. (2021) muestran que las prácticas de seguridad preventivas se alinean con la agilidad es decir si se incorporan de manera estratégica desde etapas tempranas esto asegura una adaptabilidad aceptable, siempre que exista una planificación. Moyón et al. (2020) proponen, un marco que adapta requisitos de seguridad (IEC 62443-4-1) a SAFe (S2C-SAFE). Su enfoque muestra una alta compatibilidad con la agilidad a gran escala. Baldassarre, Caivano y Piccinno (2021) proponen un marco denominado HCD-Security Scrum, que integra seguridad en Scrum mediante herramientas como VIS-PRISE. Este enfoque facilita la adopción de prácticas de seguridad sin comprometer los principios ágiles. Maqsood y Bondavalli (2020) realizaron una evaluación empírica del grado de agilidad de 11 prácticas de seguridad aplicadas en metodologías como Scrum y XP, considerando atributos como flexibilidad, velocidad, simplicidad y capacidad de respuesta. Algunas prácticas como roles *identification*, *threat modeling* y *penetration testing* mostraron alta adaptabilidad (≥ 0.86), mientras que otras como *security education* o *incident response planning* obtuvieron niveles más bajos de agilidad, (ver Cuadro 5).

Cuadro 5. Evaluación de agilidad de las prácticas de seguridad en metodologías ágiles de acuerdo con la revisión sistemática de la literatura.

Nº	Evaluación de Agilidad de las Prácticas de Seguridad en Metodologías Ágiles	Referencia
1	<i>Towards Secure Agile Software Development Process: A Practice-Based Model</i>	Ardo et al. (2022)
2	<i>Synthesizing secure software development activities for linear and agile lifecycle models</i>	Alenezi et al. (2022)
3	<i>Integrating Security and Privacy in HCD-Scrum</i>	(Baldassarre, Caivano, & Piccinno, 2021)
4	<i>Cost Benefit Analysis of Incorporating Security and Evaluation of Its Effects on Various Phases of Agile Software Development</i>	Kumar, Caer, Jolly, Baz y Cheikhrouhou (2021)
5	<i>How to Integrate Security Compliance Requirements with Agile Software Engineering at Scale?</i>	Moyón et al., 2020
6	<i>Agility of Security Practices and Agile Process Models: An Evaluation of Cost for Incorporating Security in Agile Process Models</i>	(Maqsood & Bondavalli, 2020)

Fuente. **Elaboración propia.**

3.4.3 Q3: Análisis de costo-beneficio

De la revisión de la literatura se han identificado trabajos de investigación que analizan los costos (tiempo, recursos, esfuerzo) y beneficios (seguridad mejorada, reducción de riesgos, retorno de inversión), que aportan las prácticas de seguridad al ser incorporadas en el proceso de desarrollo usando metodologías ágiles. A continuación, se resumen los hallazgos clave: Kumar et al. (2021) basaron su estudio en un análisis cuantitativo, centrado en el cálculo de costos directos e indirectos al incluir seguridad desde las primeras etapas del desarrollo ágil, costos como capacitación del equipo, herramientas de seguridad y posibles demoras iniciales. Su estudio demostró que invertir en seguridad desde el inicio reduce los costos totales del proyecto en un 30-40% y mejora el ROI. Khan et al. (2022) proponen un modelo (*Security Assurance Model* - SAM), que permite a las organizaciones adoptar prácticas de seguridad de forma progresiva y alineada a su capacidad operativa, optimizando así sus costos en función a su nivel de madurez. El

modelo fue validado en empresas reales, evidenciando su utilidad práctica para equilibrar costos y beneficios de manera escalonada. Ababneh et al. (2021) proponen integrar prácticas de seguridad en el marco de trabajo de Programación Extrema (XP) identifica que estas prácticas incrementan el costo y el tiempo en las primeras iteraciones, pero reducen errores en etapas avanzadas y facilitan el mantenimiento, lo que representa un beneficio sostenido a lo largo del ciclo de vida del producto. Rindell et al. (2021) basaron su estudio en encuestas y encontraron que las prácticas de seguridad incorporadas en fases tempranas del ciclo ágil (como análisis de riesgos y modelado de amenazas), son percibidas como de alto beneficio por su impacto en la calidad. Sin embargo, también identificaron una brecha entre el uso y la efectividad percibida de algunas prácticas. (Sharma, & Bawa, 2020) mediante el uso de métodos empíricos basado de una encuesta bajo el formato de escala Likert (1932) y el método de prueba estadística ¿Wilcoxon–Mann–Whitney or t-test?, para la selección y evaluación de las prácticas de seguridad, estas prácticas se probaron sobre la base de dos factores que incluyen el beneficio y el costo y seleccionaron 16 actividades de seguridad para las diferentes fases de un proceso ágil de desarrollo, (Ver Cuadro 6).

Cuadro 6. Análisis Costo-Beneficio de las Prácticas de seguridad en metodologías ágiles de acuerdo con la revisión sistemática de la literatura.

Nº	Costo-Beneficio de las Prácticas de Seguridad en Metodologías Ágiles	Referencia
1	<i>Security Assurance Model of Software Development for Global Software Development Vendors</i>	Khan, Khan, Alzahrani & Ilyas (2022)
2	<i>Improving Software Security in Extreme Programming Methodology</i>	(Ababneh et al., 2021)
3	<i>Cost Benefit Analysis of Incorporating Security and Evaluation of Its Effects on Various Phases of Agile Software Development</i>	Kumar, Caer, Jolly, Baz y Cheikhrouhou (2021)
4	<i>Identification and integration of security activities for secure agile development</i>	(Sharma & Bawa, 2020)
5	<i>Fitting Security into Agile Software Development</i>	(Rindell et al., 2018)

Fuente. Elaboración propia.

3.5 Conclusiones de la revisión sistemática

La revisión de literatura evidencia, según investigaciones recientes, la preocupación por integrar prácticas de seguridad en los procesos de desarrollo ágil de software. Estos estudios fundamentan sus propuestas en metodologías de desarrollo de software seguro como: el ciclo de vida del desarrollo seguro de Microsoft (SDL) (Mohino et al., 2019), proceso integral ligero de seguridad de aplicaciones (*Comprehensive, Lightweight Application Security Process*, CLASP), Criterios Comunes (*Common Criteria*, CC) y puntos de contacto de seguridad del software (*Digital Touchpoints*, CT), (Sharma, & Bawa, 2020). Varios de los modelos analizados se centran en metodologías ágiles específicas para el desarrollo de software, como programación extrema en el modelo propuesto por (Ababneh et al., 2021); el marco de trabajo SCRUM en los modelos propuestos por (Baldassarre, Caivano, & Piccinno, 2021), SAFe (marco ágil escalado) en el modelo propuesto por (Moyón et al., 2020). En tanto otros trabajos de investigación incluyen en sus modelos a programación extrema y SCRUM a la vez; como es el caso de (Sharma, & Bawa, 2020), (Maqsood, & Bondavalli, 2020), (Rindell et al., 2018). Algunas investigaciones también evalúan el grado de agilidad de las prácticas de seguridad al ser incorporadas en las metodologías ágiles, como la investigación realizada por los autores (Maqsood, & Bondavalli, 2020), investigaciones como de los autores (Sharma, & Bawa, 2020), también evalúan su costo-beneficio de incorporar prácticas de seguridad en metodologías ágiles.

CAPÍTULO IV. METODOLOGÍA

4.1 Tipo y Diseño de la Investigación

4.1.1 Tipo de investigación

Se emplea un enfoque mixto, para lograr determinar el impacto de las actividades y/o prácticas de seguridad al ser incorporadas en el proceso de desarrollo de software web, tomando en cuenta las dimensiones de grado de agilidad, costo-beneficio y tiempo en el desarrollo de software con metodologías ágiles (SCRUM).

4.1.2 Diseño de investigación

El diseño de la presente investigación es “no experimental”, se obtuvo la percepción de los expertos sobre: causas, frecuencia y consecuencias de ataques a un software web (Anexo 1 Sección 2), experiencia en desarrollo de software usando SCRUM (Anexo 1 Sección 3), la incorporación de un especialista en desarrollo de software seguro usando SCRUM (Anexo 1 Sección 4), el costo-beneficio de la actividad de seguridad “modelado de amenazas” (Anexo 1 Sección 5 y Sección 6), y el tiempo adicional requerido al incorporar las actividades de seguridad (Anexo 1 Sección 7), en el proceso de desarrollo de software web usando metodologías ágiles (SCRUM).

4.2 Unidad de Análisis

La unidad de análisis son gerentes/directores, analistas de sistemas, programadores, analistas de calidad de software, scrum masters, gestores de proyectos de software, a quienes se les aplica la encuesta para identificar la percepción sobre causa, frecuencia e impacto de un ataque a software web si no se consideran actividades de desarrollo seguro, experiencia en el desarrollo de software usando SCRUM, incorporación de un especialista en desarrollo de software seguro en un proceso de desarrollo ágil usando SCRUM, el costo de la práctica de seguridad “modelado de amenazas”, el beneficio de la práctica de seguridad “modelado de amenazas” y el tiempo adicional requerido al incorporar las actividades de seguridad en el proceso de desarrollo de software web usando metodologías ágiles.

4.3 Población de Estudio

La población del presente estudio de investigación está conformada por profesionales expertos que participan en proyectos de desarrollo de software basado en web, incluye a quienes toman decisiones estratégicas (gerentes/directores de TI), a los que coordinan la ejecución ágil (scrum masters y gestores de proyectos de software), analistas de calidad que validan la entrega continua y reportan defectos, analistas de sistemas que traducen requisitos en modelos funcionales y programadores que transforman esos modelos en código.

4.4 Tamaño de Muestra

Basado en el trabajo de (Melgarejo, 2022), el tamaño de muestra en el presente trabajo de investigación se estableció considerando la siguiente fórmula:

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

Donde:

α (Grado de Confianza): Deviación estándar de la población.

Z (Distribución Z): Valor estadístico que depende del nivel de confianza. De modo que posee un valor constante que depende del grado de confianza, siendo 99% el valor más alto (2.575) y 90% (1.645) el valor mínimo, tal como se muestra en el Cuadro 7.

Cuadro 7. Valores de grado de confianza y su correspondiente distribución Z

α	Z
0.90	1.645
0.95	1.96
0.98	2.33
0.99	2.575

Fuente. Melgarejo (2022).

p : Es un porcentaje de la población que posee la particularidad que interesa ser medida. Cuando no pueda calcularse se asume el valor de 0.5, es decir el 50% de la población posee la particularidad que interesa ser medida.

$q=1-p$: Es el porcentaje de la población que no posee la particularidad que interesa ser medida

e : Representa el límite aceptable de error muestral, para este caso utilizamos el límite de error de 5% (0.05).

N : Tamaño de la población

Respecto al valor de significancia 0.05, (Melgarejo, 2022) afirma que quien investiga posee el 95% de seguridad para generalizar sin error y solamente el 5% en contra.

Para el presente trabajo de investigación el tamaño de muestra se obtiene considerando los siguientes datos (ve Cuadro 8):

Cuadro 8. Valores para calcular el tamaño de la muestra.

α	95%
Z	1.96
p	0.5
q	0.5
e	0.05
N	51

Fuente. Melgarejo (2022).

Finalmente reemplazando los valores en la fórmula

$$n = \frac{(1.96)^2 * (0.5) * (0.5) * 51}{(0.05)^2 * (51 - 1) + (1.96)^2 * (0.5) * (0.5)}$$

$$n = 45$$

Por tanto, el valor de tamaño de la muestra es **45** con un margen de error de 5%, es decir son 45 profesionales con experiencia comprobada en proyectos de software web (véase a los que se les aplicará la encuesta).

4.5 Selección de Muestra

La muestra se obtuvo mediante un muestreo no probabilístico por conveniencia, invitando a profesionales con experiencia demostrable en proyectos web desarrollados con Scrum (gerentes/directores de TI, analistas de sistemas y de calidad, programadores, Scrum Masters y gestores de proyectos) (Melgarejo, 2022). A cada especialista se le envió un correo con el enlace al cuestionario online.

El proceso culminó con 45 respuestas válidas: 8 gerentes/directores de TI, 7 analistas de sistemas, 2 analistas de calidad, 6 Scrum Masters, 11 gestores de proyectos y 11 programadores (ver Cuadro 9). Esta distribución refleja la diversidad de roles clave en el ciclo de vida ágil y aporta una base sólida para los análisis estadísticos posteriores.

Cuadro 9. Profesionales expertos en desarrollo de software basado en web encuestados.

N°	Expertos	
1	Gerentes/Directores de TI	8
2	Analistas de sistemas	7
3	Analistas de calidad de software	2
4	Scrum Masters	6
5	Gestores de proyectos de software	11
6	Programadores	11
Total		45

Fuente. Elaboración propia.

4.6 Técnica de Recopilación de Datos

Para generar evidencia empírica sobre el impacto de integrar prácticas de seguridad en proyectos web con SCRUM, se diseñó un **cuestionario on-line autocontestado** en Microsoft Forms (Anexo 1), bajo los siguientes aspectos:

- **Objetivo:** Recoger la percepción de especialistas sobre (i) causas, frecuencia e impacto de ataques cuando no se aplica desarrollo seguro, (ii) experiencia con SCRUM, (iii) conveniencia de añadir un especialista en seguridad, (iv) costo-beneficio del Modelado de Amenazas y (v) horas adicionales por sprint al incorporar las ocho actividades de seguridad seleccionadas.
- **Estructura:** El instrumento contiene 7 secciones y 15 ítems de respuesta cerrada basados en la escala de Likert de cinco puntos, más campos numéricos para estimar tiempo en horas.

- **Aplicación:** El enlace se distribuyó por correo a profesionales de TI en julio de 2023; el muestreo combinó conveniencia y “bola de nieve”, pidiendo a cada encuestado que lo reenviara a colegas con el mismo perfil.
- **Cobertura:** Se obtuvieron 45 respuestas válidas de gerentes, scrum masters, analistas, desarrolladores y testers con experiencia demostrada en proyectos web ágiles (Anexo 3).
- **Fiabilidad y análisis:** Los datos se procesaron en IBM SPSS v29; se verificó consistencia interna con Alfa de Cronbach y se realizaron análisis descriptivos, prueba de rangos de Wilcoxon (costo–beneficio) y cálculo de medias agrupadas para la dimensión tiempo.

Esta estrategia permitió captar información cuantitativa y comparable, bajo la experiencia del encuestado, suministrando la base estadística para evaluar el marco propuesto.

4.7 Análisis e Interpretación de la Información

Luego de obtener los datos a través del cuestionario se realizaron los siguientes análisis estadísticos (se utilizó la herramienta IBM SPSS Statistics Versión 29.0.0.0): i) Fiabilidad de datos (prueba de fiabilidad usando Alfa de Cronbach). ii) Análisis descriptivo de la muestra, análisis de aspectos demográficos de la encuesta: causa, frecuencia y consecuencias(impacto) de ataque a un software web; años de experiencia en el desarrollo web usando SCRUM, incorporación de un especialista en desarrollo de software seguro; costo y beneficio de incorporar actividades de seguridad y tiempo

adicional que demandaría incorporar actividades de seguridad durante el proceso de desarrollo. iii) Análisis de modelado de amenazas basado en datos no paramétricos para comparar las muestras relacionadas (costo/beneficio), utilizando la prueba de rango de Wilcoxon. iv) Análisis de la dimensión tiempo basado en la media aritmética de datos agrupados.

CAPÍTULO V. MARCO DE TRABAJO PROPUESTO

Se presenta el modelo conceptual del marco de trabajo propuesto (ver Figura 12) y el marco de trabajo detallado (ver Figura 13). A partir de la evidencia presentada por Maqsood y Bondavalli (2020), se eligió **Scrum** como el marco ágil de referencia (véase Cuadro 10); este servirá de base para el marco de trabajo propuesto. Las actividades de seguridad que han sido incorporadas teniendo en cuenta su grado de agilidad, costo y la evaluación del impacto de la dimensión tiempo son: educación y sensibilización en materia de seguridad, identificación de roles, modelado de amenazas, análisis de riesgos de seguridad (identificación de ataques), análisis de requisitos de seguridad, análisis estático de código, pruebas de penetración, planificación de respuesta a incidentes.

Existen dos contribuciones en el presente trabajo la primera que se desarrolla es la propuesta de “Un Marco de Trabajo para el Desarrollo de Software Web Seguro con Metodologías Ágiles” (**sección 5.3**), basado en el grado de agilidad, costo, beneficio de incorporar actividades de seguridad durante el proceso de desarrollo web usando SCRUM. La segunda contribución se desarrolla en la **sección 6.5** y está referida a: Evaluar la agilidad del marco de trabajo propuesto luego de incorporar las actividades de seguridad teniendo en cuenta las dimensiones de grado de agilidad, costo, beneficio y tiempo.

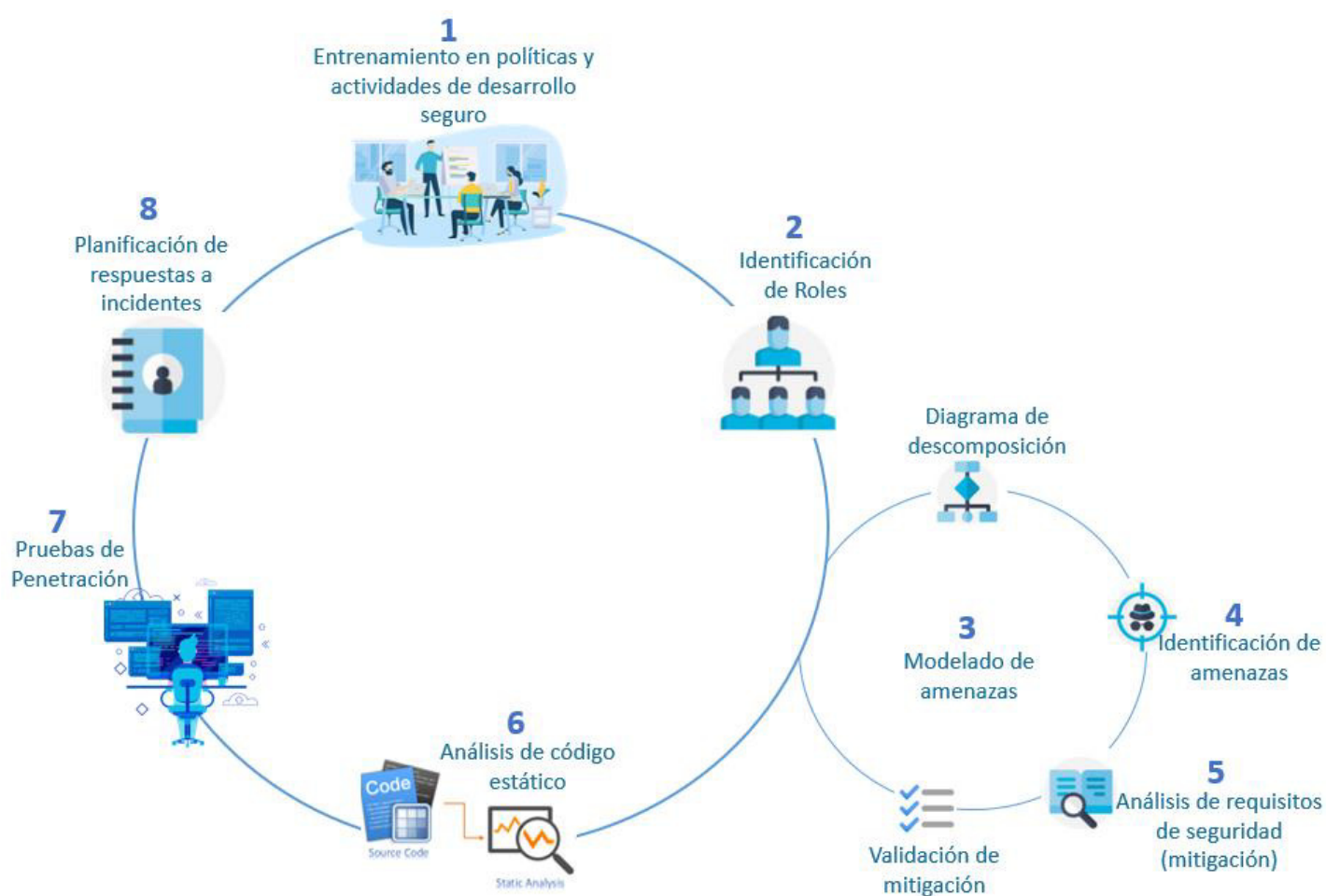


Figura 12. Modelo conceptual de marco de trabajo propuesto.

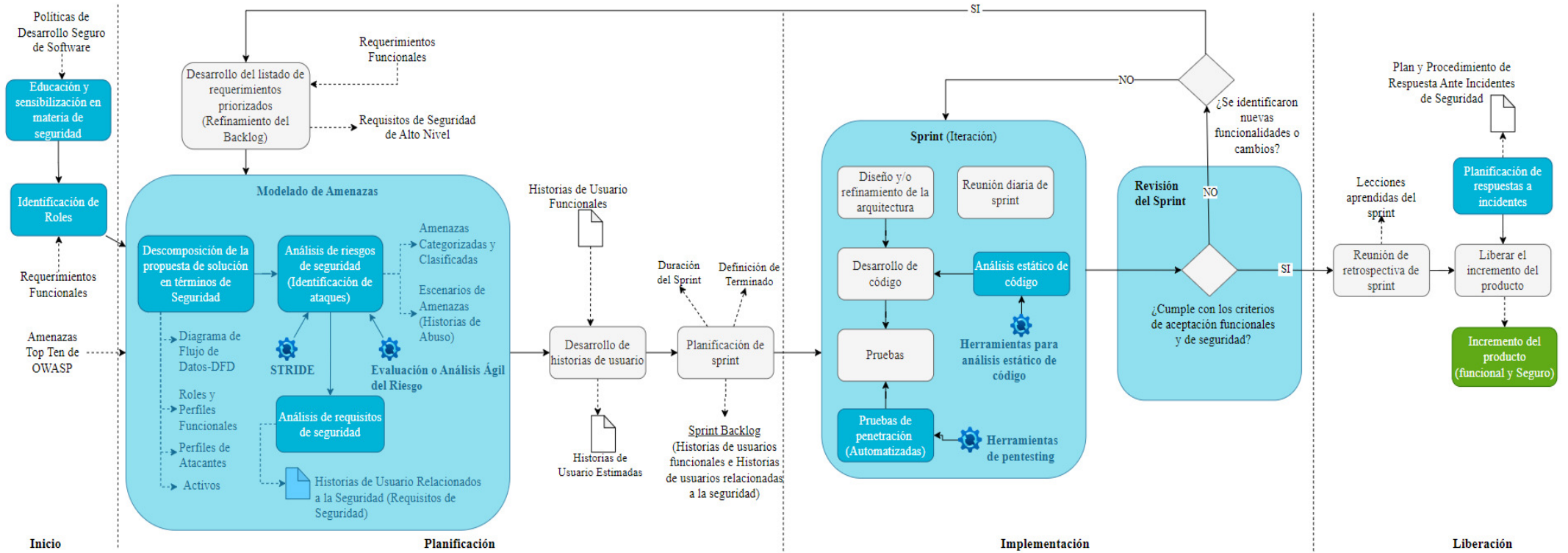


Figura 13. Marco de trabajo propuesto.

5.1 Metodología Seleccionada y Actividades de Seguridad

5.1.1 Selección de Metodología Ágil

En el estudio comparativo realizado por Maqsood y Bondavalli (2020) se evaluaron seis metodologías ágiles considerando las principales características que determinan su nivel de agilidad: **flexibilidad, eficiencia, velocidad y capacidad de respuesta**. Los autores establecieron un índice ponderado de agilidad en una escala de 0 a 1, donde un valor más alto representa una mayor capacidad de adaptación y efectividad de la metodología frente a los cambios del entorno de desarrollo.

Los resultados del análisis muestran que **Scrum** alcanza un **grado de agilidad de 0.7**, constituyéndose en la metodología con el valor más alto entre las analizadas, seguida de **Extreme Programming (XP)** con 0.68, y posteriormente **ASD, FDD, Cristal y DSDM**, con valores inferiores (ver Cuadro 10). Estos resultados evidencian que **Scrum logra un equilibrio óptimo entre estructura, flexibilidad y capacidad de respuesta**, cualidades que la convierten en la opción más adecuada para proyectos que requieren alta adaptabilidad, colaboración continua y entregas iterativas.

En consecuencia, y conforme a la evidencia empírica presentada por **Maqsood y Bondavalli (2020)**, **Scrum se adopta como la metodología base del marco de trabajo propuesto**, al representar el enfoque más coherente con los principios de agilidad requeridos y permitir la integración progresiva de prácticas de seguridad sin comprometer la dinámica ágil del desarrollo.

Cuadro 10. Agilidad de metodologías ágiles.

N	Características de Agilidad	Flexibilidad	Eficiencia	Velocidad	Capacidad de Respuesta	Suma de Características Ágiles	Nro. de Práctica + Fases en una Metodología Ágil	Agilidad Total de la Metodología Ágil
	Metodologías Ágiles							
1	XP	15	6	13	15	49	18	0.68
2	SCRUM	9	0	10	9	28	10	0.7
3	Desarrollo de Software Adaptable-ASD	10	0	12	10	32	12	0.66
4	Desarrollo impulsado por Características-FDD	10	0	10	10	30	13	0.57
5	Cristal	9	0	11	11	32	11	0.7
6	Método de desarrollo de sistemas dinámicos-DSDM	10	0	11	11	32	15	0.53

Fuente. Cuadro adaptado de Maqsood & Bondavalli (2020).

5.1.2 Selección de Actividades de Seguridad

Maqsood y Bondavalli (2020): Este trabajo es uno de los más relevantes porque proporciona una evaluación cuantitativa sobre el impacto de las actividades de seguridad en la agilidad de SCRUM. Mediante un estudio empírico basado en encuestas a profesionales del sector, estos autores analizaron cómo 11 actividades de seguridad afectan la flexibilidad, esbeltez, velocidad y capacidad de respuesta del desarrollo ágil.

Métrica clave: Uso de la escala de Likert de 5 puntos para medir el grado de agilidad de cada actividad de seguridad.

Impacto evaluado: Identificación del equilibrio entre seguridad y eficiencia en metodologías ágiles.

Conclusión: Se demuestra que ciertas prácticas pueden mantener la velocidad de SCRUM sin comprometer la protección del software.

5.1.3 Costo-Beneficio de la Seguridad

Sharma & Bawa (2020): Este estudio complementa el trabajo de Maqsood & Bondavalli al abordar la selección de actividades de seguridad desde un enfoque de costo-beneficio. Aplicando métodos empíricos como AHP y PROMETHEE, identificaron qué actividades generan el mayor impacto en seguridad con el menor costo en términos de tiempo y recursos. Para la valoración del costo y beneficio de las actividades de seguridad incorporadas a una metodología de desarrollo de software bajo enfoques ágil (a excepción de la actividad de modelado de amenazas), se han determinado considerando ciertos rangos como se muestra en el Cuadro 11.

Cuadro 11. Rangos costo – beneficio

Rango	Costo	Beneficio
99.3% < Resultado	Muy costoso	Muy Beneficioso
99.0% < Resultado < 99.3%	Costoso	Beneficioso
95.0% < Resultado < 99.0%	Poco costoso	Poco Beneficioso

Fuente. Cuadro adaptado de Sharma & Bawa (2020).

5.1.4 Consideraciones Sobre Agilidad y Costo-Beneficio

- Sharma & Bawa (2020), Maqsood y Bondavalli (2020) evalúan empíricamente el impacto de la seguridad en SCRUM.

- Combinan la medición del grado de agilidad con el análisis de costo-beneficio, proporcionando un marco integral para seleccionar actividades de seguridad.
- Validan sus resultados con datos de la industria del software, asegurando que las prácticas recomendadas sean viables en entornos reales

5.1.5 Criterios de Selección de Actividades de Seguridad

Basándonos en estos estudios, se han seleccionado ocho actividades de seguridad que cumplen con los siguientes criterios estratégicos:

- Alto impacto en la reducción de vulnerabilidades (Sharma, & Bawa, 2020).
- Bajo costo de implementación (Maqsood, & Bondavalli, 2020).
- Mínimo impacto en la velocidad de SCRUM (Maqsood, & Bondavalli, 2020).
- Compatibilidad con los principios ágiles (Sharma, & Bawa, 2020).

Se seleccionaron ocho actividades validadas en ambos estudios para SCRUM considerando su grado de agilidad (rango de 0 a 1, donde 1 representa a la característica o actividad de seguridad altamente ágil y 0 nada ágil), costo y beneficio (a excepción de la actividad de modelado de amenazas), (ver Cuadro 12).

Cuadro 12. Agilidad, costo y beneficio de las actividades de seguridad.

N	Actividades de Seguridad	Agilidad	Costo	Beneficio
1	<i>Educación y sensibilización en materia de seguridad</i>	0.75	Poco Costoso	Muy Beneficioso
2	<i>Identificación de Roles</i>	0.89	Costoso	Muy Beneficioso
3	<i>Modelado de amenazas</i>	0.88	?	?
4	<i>Análisis de riesgos de seguridad (Identificación de ataques)</i>	0.87	Costoso	Muy Beneficioso

5	<i>Análisis de requisitos de seguridad</i>	0.83	Costoso	Muy Beneficioso
6	<i>Análisis estático de código</i>	0.85	Poco Costoso	Muy Beneficioso
7	<i>Pruebas de penetración</i>	0.86	Poco Costoso	Muy Beneficioso
8	<i>Planificación de respuesta a incidentes</i>	0.81	Poco Costoso	Muy Beneficioso

Fuente. Cuadro adaptado de Sharma y Bawa (2020), (Maqsood, & Bondavalli, 2020).

5.1.6 Validación de la Selección: Impacto Medido y Cuantificado

Los resultados obtenidos en los estudios de Maqsood y Bondavalli (2020) y Sharma & Bawa (2020) han permitido validar que es posible mantener la rapidez de SCRUM sin comprometer la seguridad:

- **Costo-beneficio:** Se comprobó que implementar seguridad desde el inicio reduce costos operativos en comparación con corregir vulnerabilidades en etapas tardías.
- **Tiempo adicional requerido en SCRUM:** Se verificó que las actividades seleccionadas no generan retrasos significativos en los sprints.
- **Grado de agilidad mantenido:** Se confirmó que la integración de estas actividades mantiene la flexibilidad y eficiencia del desarrollo ágil.

5.1.7 Medición Empírica del Impacto de la Seguridad

Los estudios de Maqsood y Bondavalli (2020) y Sharma & Bawa (2020) miden empíricamente el impacto de la seguridad en SCRUM, permitiendo una selección objetiva y cuantificable de actividades.

Gracias a su enfoque, se tiene evidencias que es posible desarrollar software seguro sin sacrificar la velocidad de SCRUM. Este marco de trabajo establece un nuevo estándar en la industria, permitiendo que la seguridad y la agilidad coexistan de manera efectiva

5.2 Propuesta de Marco de Trabajo para el Desarrollo de Software Web Seguro con Metodologías Ágiles

El presente trabajo de investigación propone un marco para el desarrollo de software web seguro con metodologías ágiles considerando las dimensiones de agilidad, tiempo costo (esfuerzo), y beneficio. Las actividades de seguridad seleccionadas se integran durante todo el proceso de desarrollo de software ágil (SCRUM) y se agrupan en las fases de inicio, planificación, implementación y liberación (por cada *sprint*), (ver Figura 12 y Figura 13). El marco de trabajo propuesto contempla la incorporación del rol de especialista en desarrollo de software seguro, quien es el responsable de asegurar que todas las actividades de seguridad sean ejecutadas por el equipo de desarrollo. A continuación, se describe cada actividad de seguridad seleccionadas, que se incorporan en cada una de las fases del marco de trabajo propuesto:

- **Inicio**
 - Educación y sensibilización en materia de seguridad.
 - Identificación de roles.
- **Planificación**

- Modelado de amenazas
- Análisis de riesgos de seguridad (identificación de ataques).
- Análisis de requisitos de seguridad.
- **Implementación**
 - Análisis estático de código.
 - Pruebas de penetración.
- **Liberación**
 - Planificación de respuesta a incidentes

5.3 Descripción del Marco de Trabajo

5.3.1 Inicio

En esta fase, se integran las actividades de educación y sensibilización en materia de seguridad, así como la identificación de roles, basándose en los requerimientos funcionales y las políticas de desarrollo seguro. Como resultado, se obtiene un equipo de desarrollo consciente de las políticas y prácticas de seguridad en el proceso de desarrollo de software, y se identifican los requisitos de seguridad de alto nivel, (ver Figura 14).

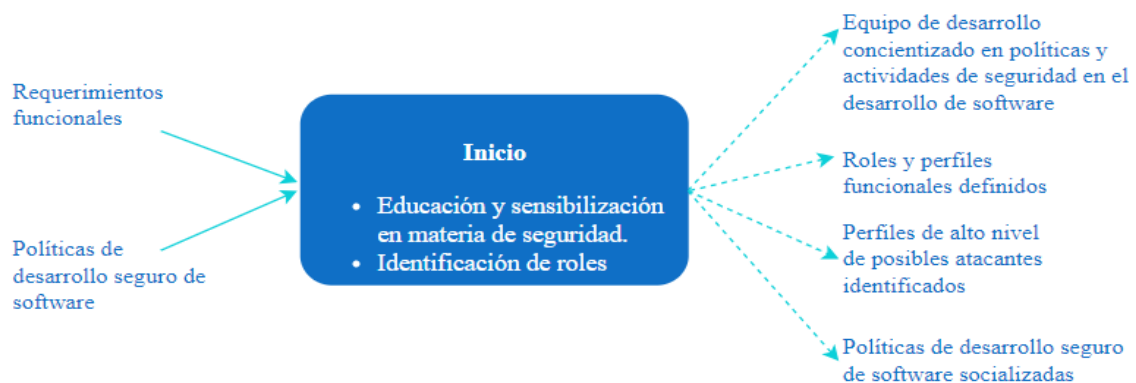


Figura 14. Actividades de desarrollo seguro en la fase de inicio.

5.3.1.1 Educación y sensibilización en materia de seguridad

Definidos los requerimientos funcionales (*product backlog*), y las políticas de desarrollo seguro se llevan a cabo las siguientes actividades:

- El especialista de desarrollo de software seguro expone las políticas y lineamientos de seguridad en el desarrollo de software.
- El especialista de desarrollo de software seguro realiza la sensibilización sobre las actividades de seguridad que se realizarán durante el proceso de desarrollo a cada uno de los miembros del equipo dependiendo de su rol.
- El especialista de desarrollo de software seguro junto al equipo y todos los interesados del proyecto identifican los requisitos de seguridad de alto nivel que formaran parte de los criterios de aceptación de cada una de las historias de usuarios funcionales.
- Todos los integrantes del equipo e interesados del proyecto de desarrollo deben ser conscientes de la importancia de la seguridad y de los fundamentos de la ingeniería de seguridad (*Security Engineering -SE*), descrito en la **subsección 2.2.3.**

5.3.1.2 Identificación de roles

- En base a los requerimientos funcionales el especialista de desarrollo de software seguro junto al equipo y todos los interesados del proyecto identifican los perfiles (niveles de confianza), para todos los usuarios que tendrán acceso al aplicativo informático (software web).
- Durante esta actividad también se identifican los perfiles de alto nivel de los posibles atacantes que se han de tener en cuenta al momento de realizar las pruebas en la fase de planeación e implementación.

5.3.2 Planificación

En esta fase, el equipo de desarrollo, consciente de las políticas y prácticas de desarrollo de software seguro, se basa en los requisitos de alto nivel y en las 10 amenazas más comunes identificadas por OWASP (OWASP Foundation, 2021), (ver Figura 15), para llevar a cabo las siguientes actividades de seguridad:

- Modelado de amenazas
- Análisis de riesgos de seguridad (identificación de ataques).
- Análisis de requisitos de seguridad.



Figura 15. Actividades de desarrollo seguro en la fase de planificación.

5.3.2.1 Modelado de amenazas

El objetivo de esta actividad de seguridad es identificar posibles ataques (amenazas de seguridad) a los que un sistema informático (software web) puede estar expuesto y definir un conjunto de requisitos de seguridad para mitigarlos (ver Figura 16).

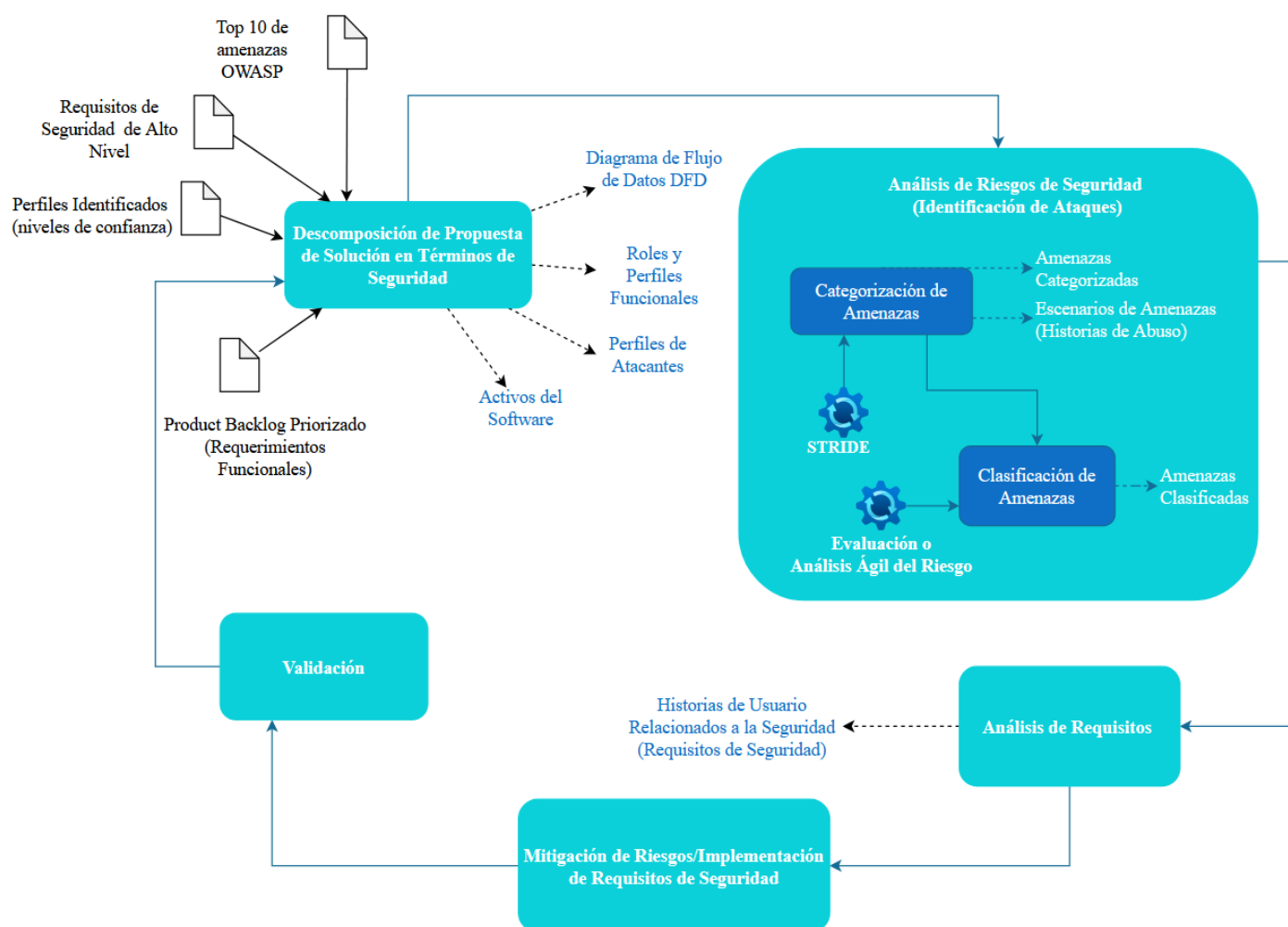


Figura 16. Modelado de amenazas.

Esta actividad se enmarca en el proceso de modelado de amenazas (*Threat Modeling Process*) definido por OWASP (OWASP Foundation, n.d.) que incluye dos actividades

clave del modelo de desarrollo seguro propuesto (el análisis de riesgos de seguridad y el análisis de requisitos de seguridad):

- a. Descomposición de la propuesta de solución en términos de seguridad.
- b. Análisis de riesgos de seguridad (identificación de ataques).
- c. Análisis de requisitos de seguridad.
- d. Mitigación de riesgos/implementación de requisitos de seguridad
- e. Validación.

a. Descomposición de la propuesta de solución, en términos de seguridad

- Los puntos de partida de esta actividad son los requisitos de seguridad de alto nivel identificados en la fase de inicio, el listado de requerimientos priorizados (product backlog priorizado), los roles y/o perfiles identificados (niveles de confianza), y las 10 amenazas más comunes de seguridad en el desarrollo de software (OWASP Foundation, 2021), explicadas en la **subsección 2.2.4**.
- Guiados por el especialista en desarrollo seguro el equipo de desarrollo identifica los posibles objetivos de amenazas, desde la perspectiva de un atacante, representándolos visualmente a través de un diagrama de flujo de datos (DFD), la parte teórica de un DFD se describe en la **subsección 2.2.7**.
- El diagrama de flujo de datos permite al equipo de desarrollo identificar los denominados “puntos de entrada” (vulnerabilidades de acceso), por donde un atacante potencial podría acceder al sistema informático (software web) y causar daño.
- El resultado de la descomposición de la propuesta de solución, en términos de seguridad es:
 - El diagrama de Flujo de Datos (DFD).

- La identificación de los puntos de entrada para ver dónde un atacante potencialmente podría interactuar maliciosamente con el software producido.
- La identificación de los activos, es decir, elementos o áreas en las que el atacante potencialmente estaría interesado.
- Roles y/o perfiles funcionales y perfiles de atacantes.

b. Análisis de riesgos de seguridad (identificación de ataques)

- Los puntos de partida de esta actividad son el diagrama de flujo de datos (DFD), las 10 amenazas más comunes identificadas por OWASP (Top Ten), los niveles de confianza y los roles y/o perfiles de atacantes ya identificados.
- El equipo de desarrollo incluido el *product owner*, guiados por el especialista de desarrollo seguro, identifican las exposiciones o ataques a las que puede estar expuesto el *software web* (amenazas), vulnerabilidades e impacto.
- Las amenazas identificadas son traducidas en un lenguaje de historias de abuso (escenarios de amenazas bajo procesos de desarrollo ágil), estas historias son una variante de las historias de usuario (Andrade, Torres, Ortiz-Garcés, Miño, & Almeida, 2023), un mayor detalle referido a historia de abuso se redacta en la **subsección 2.2.8**.
- Identificadas las amenazas, vulnerabilidades e impacto se procede a su categorización para ello se hace uso del modelo de categorización de amenazas STRIDE (*Spoofing, Tempe ring, Repudio, Information disclosure, Denal of service, Elevation of privileges*), cuya definición se detalla en la **subsección 2.2.6.1**.
- Categorizadas las amenazas y definidas las historias de abuso, el equipo de desarrollo procede a clasificarlos desde la perspectiva de factores de riesgo (probabilidad e impacto de ocurrencia), en el presente trabajo se le define como evaluación ágil de riesgos y está basado en el modelo de clasificación de riesgos definido por OWASP (la base teórica se detalla en la **subsección 2.2.6.3**), para

esta actividad se puede usar las herramientas informáticas propuestas por OWASP como: <https://www.owasp-risk-rating.com/>, <https://javierolmedo.github.io/OWASP-Calculator/> o usar la hoja de cálculo (Plantilla Gestión de Riesgo-OWASP) del Anexo 4 y que puede descargarlo desde el siguiente enlace: https://drive.google.com/drive/u/0/folders/1xoszX6NMg3x_WDt7wZUSYpo22mNJRvkT; sin embargo, el equipo de desarrollo puede optar por hacer uso del modelo de evaluación de riesgos, basado en el valor, conocido como **DREAD** (cuya base teórica se detalla en la **subsección 2.2.6.2**)

c. Análisis de requisitos de seguridad

- Los puntos de partida en este paso del modelado de amenazas son: las amenazas identificadas y definidas como escenarios de amenazas (historias de abuso), categorizadas y clasificadas (en función de la evaluación de riesgos).
- El equipo de desarrollo junto al especialista en desarrollo de software seguro identifica y definen las nuevas funciones del software (requisitos de seguridad), que permitan resolver uno o más problemas de seguridad en específico, eliminando una o más vulnerabilidades potenciales y evitar que una amenaza se materialice.
- Los requisitos de seguridad identificados se formulan y describen como historias de usuario relacionadas a la seguridad (su descripción teórica se desarrolla en la **subsección 2.2.9**), un ejemplo de historia de usuario relacionado a la seguridad puede ser: “Yo como cliente deseo que mis datos sean protegidos para no ser divulgados a otros clientes o personas desconocidas”. Se recomienda tomar como base las plantillas de prácticas de seguridad y tareas de seguridad para entornos de desarrollo ágiles (*practical security stories and security tasks for agile development*), y que las puede descargar del siguiente enlace: https://drive.google.com/drive/u/0/folders/1xoszX6NMg3x_WDt7wZUSYpo22mNJRvkT.

- Los requisitos de seguridad forman parte de la definición de hecho (*Definition of Done*) de cada sprint en el presente marco propuesto, una vez que estos hayan completado se da por concluido todas las actividades del sprint.
 - En esta parte del proceso de modelado de amenazas, se tiene como salida: los requisitos de seguridad expresado como historias de usuario relacionadas a la seguridad (constituye el producto backlog de seguridad), y posibles nuevas políticas de seguridad.
- d. Implementación de requisitos de seguridad (Mitigación del Riesgo)**
- En lo que respecta a la mitigación de riesgos o implementación de requisitos de seguridad estos se llevan a cabo durante la implementación del sprint (fase de implementación), es decir mediante el desarrollo o implementación de cada una de las historias de usuario relacionadas a la seguridad correspondiente a cada uno los requisitos de seguridad identificados.
- e. Validación de la implementación de los requisitos de seguridad**
- La validación de la implementación de los requisitos de seguridad (historias de usuario relacionados a la seguridad), se realiza al final de la fase de implementación durante la revisión del sprint, y validando si se ha cumplido con lo establecido en las definiciones de terminado (*Definitions of Done*).

De esta fase (planificación) se tiene como resultado:

- Las definiciones de terminado (Definitions of Done).
- El sprint backlog (historias de usuarios funcionales e historias de usuarios relacionadas a la seguridad estimadas).
- La duración del sprint.
- Los escenarios de amenazas (historias de abuso).
- Las posibles nuevas políticas de desarrollo de software seguro.
- Los roles y los perfiles funcionales del aplicativo informático, así como los perfiles de los posibles atacantes que se usaran en las pruebas.

5.3.3 Implementación

Durante esta fase el equipo de desarrollo implementa los requerimientos funcionales e historias de usuario relacionadas a la seguridad que han sido planificadas durante el *sprint planning* y que deben implementarse durante el sprint, (ver Figura 17). El especialista en desarrollo de software seguro orienta al equipo de desarrollo en la ejecución de las siguientes actividades de seguridad:

- Análisis estático de código (pruebas de caja blanca).
- Pruebas de penetración.

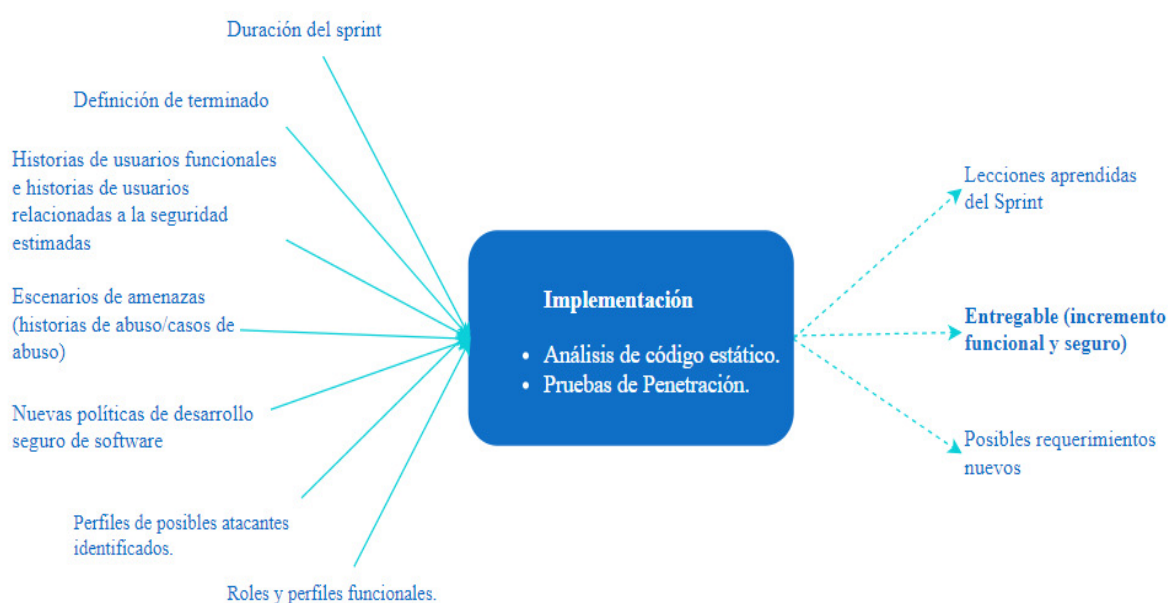


Figura 17. Actividades de desarrollo seguro en la fase de implementación

5.3.3.1 Análisis estático de código

- Bajo la orientación del especialista en desarrollo de software seguro el equipo de desarrollo planifica, diseña y ejecuta las pruebas de código estático, la explicación teórica sobre análisis estático de código se encuentra detallada en la subsección 2.2.5.6.

- Para esta actividad el equipo de desarrollo puede hacer uso de herramientas automatizadas de análisis estático de código conocidas como *Static Application Security Testing* (SAST), algunas herramientas ya se introducen en los entornos de programación (IDE) facilitando aún más el análisis estático de código. OWASP también proporciona una lista de herramientas automatizadas para realizar el análisis estático de código (OWASP Foundation, s.f). Esta actividad debe realizarse poniendo en práctica las pruebas de penetración por pares.
- En paralelo se recomienda, como parte de esta actividad, tomar en cuenta el Top 25 de los errores de programación más peligrosos “*Top 25 Most Dangerous Programming Errors*” (MITRE, 2024).

5.3.3.2 Pruebas de penetración (pentesting)

- El equipo de desarrollo debe realizar las pruebas de penetración al finalizar cada sprint antes de su liberación o pase a producción, actividad que debe realizarse poniendo en práctica las pruebas de penetración por pares. Con la orientación del especialista en desarrollo de software seguro realiza las pruebas de penetración, para simular acciones de un atacante informático con el objetivo de descubrir posibles vulnerabilidades resultantes de errores de codificación, configuración u otras debilidades, una explicación teórica sobre las pruebas de penetración (*pentesting*), se encuentra detallada en la **subsección 2.2.5.7**.
- El equipo de desarrollo se debe apoyar en el uso de herramientas automatizadas para escanear de manera externa (desde la perspectiva de un atacante), el sistema informático en búsqueda de vulnerabilidades de seguridad, estas herramientas son conocidas como pruebas dinámicas de seguridad de aplicaciones (Dynamic Application Security Testing-DAST).

- OWASP proporciona una lista de herramientas para realizar las pruebas de penetración accesible desde el siguiente enlace web: https://owasp.org/www-community/Free_for_Open_Source_Application_Security_Tools. De esta lista se sugiere usar la herramienta para prueba dinámica conocida como Zed Attack Proxy (ZAP) y la puede descargar desde el siguiente enlace web: <https://owasp.org/www-project-zap/>.

5.3.4 Liberación

En esta fase del proceso se realizan todas las actividades para la puesta en producción (implantación), del **entregable** (incremento funcional y seguro), adicional a ello el equipo de desarrollo bajo la orientación del especialista en desarrollo de software seguro realizan la planificación de respuesta a incidentes, (ver Figura 18).

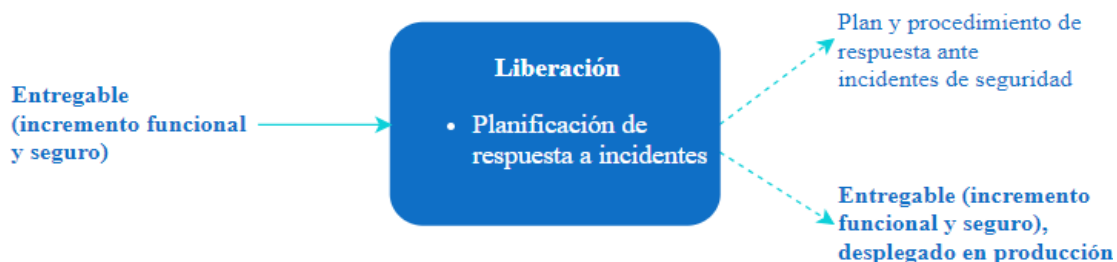


Figura 18. Actividad de desarrollo seguro en la fase de liberación

5.3.4.1 Planificación de respuesta a incidentes

- Como primera acción se debe conformar el equipo de respuesta ante incidentes de seguridad que sea responsable de analizar, categorizar y responder a los incidentes de seguridad, mayor referencia se detalla en la **subsección 2.2.5.8** del presente trabajo de investigación.

- El objetivo de contar con un plan de respuesta ante incidentes de seguridad es responder ante los incidentes de seguridad de manera sistemática (siguiendo una metodología coherente de gestión de incidentes), para que se adopten las medidas adecuadas a fin de reducir al mínimo su impacto y como recuperarse producido el incidente. El plan debe contener como mínimo:
 - Estrategias y objetivos.
 - Aprobación de la alta dirección.
 - Metodología (que define el procedimiento de respuesta ante incidentes de seguridad).
 - Roles y responsabilidades.
 - Estrategia de comunicaciones para comunicarse con el resto de la organización y posiblemente con otras organizaciones.
 - Métricas para medir la capacidad de respuesta ante incidentes y su eficacia.
 - Registro de las actividades de respuesta (bitácora de respuestas ante incidentes de seguridad).
 - Actualización del plan basado en lecciones aprendidas.
 - Cronograma y resultado de las pruebas del plan.
- Puede tomar como referencia cualquiera de las plantillas que están disponibles para su descargar desde el siguiente enlace: https://drive.google.com/drive/u/0/folders/1xoszX6NMg3x_WDt7wZUSYpo22mNJRvkT.

El resultado de esta actividad es contar con un plan aprobado y un equipo de respuestas ante incidentes de seguridad con roles y responsabilidades definidas.

CAPÍTULO VI. RESULTADOS

6.1 Fiabilidad de los Datos (Alpha Cronbach)

Según el trabajo de investigación de Alarcón Loayza (2024), un valor aceptable para el coeficiente alfa de Cronbach es 0.7. Obtener un valor superior a este indica una fuerte relación entre las preguntas, lo que valida el instrumento. En el presente trabajo de investigación, utilizando la herramienta *IBM SPSS Statistics*, se obtuvo un alfa de Cronbach de **0.737**, (ver Figura 19), lo que confirma que la validez del instrumento es aceptable.

Estadísticas de fiabilidad	
Alfa de Cronbach	N de elementos
,737	15

Figura 19. Resultado de fiabilidad de datos del cuestionario aplicado en el presente trabajo de investigación.

6.2 Análisis Descriptivo de los Resultados de la Muestra

A continuación, se presentan los datos obtenidos mediante un instrumento de recolección diseñado para este estudio. Este instrumento, compuesto por 15 ítems, fue

aplicado a 45 profesionales y expertos en desarrollo de software web con SCRUM. El análisis descriptivo de los datos se realizó utilizando la herramienta *IBM SPSS Statistics*.

6.2.1 Causa, frecuencia e impacto de ataque(s) a un software web sin actividades de desarrollo seguro

6.2.1.1 Ausencia de actividades de desarrollo seguro como causa principal de ataques

Para evaluar la percepción de los profesionales encuestados sobre la ausencia de actividades de desarrollo seguro como causa principal de ataques, se les formuló la siguiente pregunta: '**¿Considera usted que una de las principales causas de ataques a un software web es la falta de incorporación de actividades de desarrollo seguro durante su proceso de desarrollo?**'. Los resultados obtenidos del análisis estadístico fueron los siguientes:

- Media: **3.96**
- Mediana: **4.0**
- Moda: **5**
- Desviación estándar: **1.33**
- Varianza: **1.77**
- Cantidad de respuestas: **45**

Los resultados muestran que la mayoría de los profesionales encuestados están de acuerdo con la afirmación, con una media cercana a 'De Acuerdo' y una moda en 'Totalmente de Acuerdo'. Aunque la desviación estándar indica cierta variabilidad en las respuestas, la tendencia general es positiva hacia la idea de que la ausencia de actividades de desarrollo seguro se considera una causa principal de los ataques a un software web.

6.2.1.2 Frecuencia de ataques a un software web si durante su desarrollo no se incorporaron actividades de desarrollo seguro

Para evaluar la percepción de los profesionales encuestados sobre la frecuencia de ataques a un software web sin actividades de desarrollo seguro, se les formuló la siguiente pregunta: **‘¿Con qué frecuencia sería atacado un software web si durante el desarrollo no se incorporaron actividades de desarrollo seguro?’**. Los resultados obtenidos del análisis estadístico fueron los siguientes:

- Media: **4.09**
- Mediana: **4.0**
- Moda: **4**
- Desviación estándar: **0.63**
- Varianza: **0.40**
- Cantidad de respuestas: **45**

Los resultados indican que la mayoría de los encuestados creen que, si no se incorporan actividades de desarrollo seguro durante el proceso de desarrollo de un software web, es muy probable que el software sea atacado con frecuencia. **Esto subraya la importancia de implementar prácticas de seguridad durante el desarrollo para mitigar el riesgo de ataques.**

6.2.1.3 Impacto luego de un ataque a un sistema informático(software)

Para evaluar la percepción de los profesionales encuestados sobre el nivel de impacto de un ataque a un sistema informático (software), se les formuló la siguiente pregunta: **‘¿Cuál es el nivel de impacto que se da en una organización luego de producirse un**

ataque a su sistema informático?’. Los resultados obtenidos del análisis estadístico fueron los siguientes:

- Media: **1.58**
- Mediana: **1.0**
- Moda: **1**
- Desviación estándar: **0.81**
- Varianza: **0.66**
- Cantidad de respuestas: **45**

Los resultados indican que la mayoría de los encuestados creen que el impacto de un ataque a un sistema informático (web), en una organización es extremadamente negativo. Esto destaca la importancia de la seguridad informática y la necesidad de medidas preventivas para proteger los sistemas y mitigar los riesgos de ataques que puedan tener consecuencias graves para la organización.

6.2.2 Experiencia en el desarrollo de software usando Scrum

Para determinar la experiencia de los profesionales encuestados y asegurar que sus respuestas estén basadas en esta, se les formuló la siguiente pregunta: ‘**¿Cuántos años de experiencia tiene en el desarrollo de software usando SCRUM?’**. Los resultados obtenidos del análisis estadístico fueron los siguientes:

- Media: **2.6**
- Mediana: **2.0**
- Moda: **2**
- Desviación estándar: **1.25**
- Varianza: **1.56**
- Cantidad de respuestas: **45**

Los resultados indican que la mayoría de los encuestados tienen entre 3 y 5 años de experiencia en el desarrollo de software usando SCRUM. Sin embargo, hay una variabilidad considerable, con algunos encuestados que tienen menos experiencia y otros que tienen más de 10 años de experiencia. Esto sugiere que los encuestados tienen un rango amplio de experiencia, lo cual es importante para obtener una perspectiva diversa sobre las prácticas y desafíos en el desarrollo de software seguro con SCRUM.

6.2.3 Incorporación de un especialista en desarrollo de software seguro usando Scrum

Para evaluar la percepción de los profesionales encuestados sobre la incorporación de un especialista en desarrollo de software seguro usando SCRUM, durante el proceso de desarrollo se les formuló la siguiente pregunta: **‘¿Cuál es su posición respecto a la incorporación de un especialista en desarrollo de software seguro usando SCRUM?’**. Los resultados obtenidos del análisis estadístico fueron los siguientes:

- Media: **4.53**
- Mediana: **5.0**
- Moda: **5**
- Desviación estándar: **0.73**
- Varianza: **0.53**
- Cantidad de respuestas: **45**

Estos resultados indican que la mayoría de los encuestados están fuertemente de acuerdo con la afirmación, con una media y una mediana cercanas a "Totalmente de Acuerdo" y una moda en "Totalmente de Acuerdo". La baja desviación estándar sugiere que hay poca variabilidad en las respuestas, lo que refuerza el consenso positivo hacia la incorporación de un especialista en desarrollo seguro en equipos que usan SCRUM.

6.2.4 Costo de incorporar la actividad de seguridad "Modelado de Amenazas" durante el desarrollo

Para evaluar la percepción de los profesionales encuestados respecto al costo que demanda el incorporar la actividad de seguridad “Modelado de Amenazas” durante el proceso de desarrollo, se les formuló la siguiente pregunta: **‘¿Qué tan costoso considera usted que significaría incorporar la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo de software usando SCRUM?’**. Los resultados obtenidos del análisis estadístico fueron los siguientes:

- Media: **2.98**
- Mediana: **3.0**
- Moda: **3**
- Desviación estándar: **0.92**
- Varianza: **0.84**
- Cantidad de respuestas: **45**

Los resultados indican que, en general, los encuestados tienen una percepción neutral sobre el costo de incorporar el modelado de amenazas durante el desarrollo de software usando SCRUM. Esto sugiere que, aunque algunos pueden considerar que es una inversión significativa, otros pueden verlo como un costo manejable o necesario para mejorar la seguridad del software. La opinión está equilibrada, lo que implica que la percepción del costo puede variar dependiendo del contexto y de los recursos de cada organización.

6.2.5 Beneficio de incorporar la actividad de seguridad "Modelado de Amenazas" durante el desarrollo

Para evaluar la percepción de los profesionales encuestados respecto al beneficio de incorporar la actividad de seguridad "Modelado de Amenazas" durante el desarrollo, se les formuló la siguiente pregunta: '**¿Qué tan beneficioso considera usted que significaría incorporar la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo de software usando SCRUM?**'. Los resultados obtenidos del análisis estadístico fueron los siguientes:

- Media: **4.6**
- Mediana: **5.0**
- Moda: **5**
- Desviación estándar: **0.58**
- Varianza: **0.34**
- Cantidad de respuestas: **45**

Los resultados indican que la mayoría de los encuestados creen que la incorporación de la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo de software usando SCRUM es altamente beneficiosa. Esto sugiere que los encuestados valoran mucho la inclusión de esta práctica de seguridad y reconocen su importancia para mejorar la calidad y la seguridad del software desarrollado.

6.2.6 Tiempo adicional que demanda incorporar actividades de seguridad en el desarrollo con SCRUM

El resultado (frecuencias), de la encuesta respecto a la pregunta sobre el tiempo adicional (en horas), que demanda aplicar cada actividad de seguridad se muestra en el Cuadro 13.

Cuadro 13. Resultado (frecuencias) de tiempo adicional que demanda incorporar las actividades de seguridad

N	Actividades de Seguridad	Respuestas				
		Menos de 2 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 8 y 10 horas	Mas de 10 horas
1	Educación y Sensibilización en Materia de Seguridad (entrenamiento en políticas y actividades de desarrollo seguro)	1	13	15	9	6
2	Identificación de Roles	8	23	8	5	1
3	Modelado de Amenazas	3	11	13	9	9
4	Análisis de Riesgos de Seguridad (Identificación de Ataques)	4	12	17	7	5
5	Análisis de Requisitos de Seguridad	5	11	16	9	4
6	Análisis estático de código	3	17	7	11	7
7	Pruebas de Penetración	4	6	13	12	10
8	Planificación de Respuesta a Incidentes	4	13	14	11	3

Fuente. Elaboración propia.

6.3 Análisis del Modelado de Amenazas (Costo-Beneficio)

Para comparar el costo-beneficio de incorporar la actividad de seguridad “Modelado de Amenazas” en el desarrollo de software basado en los resultados de la encuesta aplicada en el presente trabajo de investigación (subsecciones 6.2.4 y 6.2.5), se utilizó la prueba

de rango de Wilcoxon (McKnight, & Najab, 2010) (prueba de hipótesis estadística no paramétrica para comparar dos muestras relacionadas).

Hipótesis:

H_0 : No hay diferencias entre las observaciones pareadas

H_1 : Sí hay diferencias entre las observaciones pareadas

Para el caso de estudio:

H_0 : No hay diferencias entre costo y beneficio al aplicar la actividad de seguridad de modelado de amenazas en el marco de trabajo propuesto

H_1 : Sí hay diferencias entre costo y beneficio al aplicar la actividad de seguridad de modelado de amenazas en el marco de trabajo propuesto

Utilizando la herramienta *IBM SPSS Statistics* Versión 29.0.0.0 y con un nivel de significancia de 0.05 se encontraron los siguientes resultados:

6.3.1 Estadísticos descriptivos

En base a valores de la escala de Likert obtenidos en la encuesta referidos al costo y beneficio de incorporar la actividad de seguridad “Modelado de Amenazas” durante el proceso de desarrollo de software, se calcularon los estadísticos descriptivos (media, desviación estándar, valores mínimo y máximo), para las 45 respuestas. Los resultados se presentan en el Cuadro 14.

Cuadro 14. Resultados costo y beneficio al incorporar en el proceso la actividad de “Modelado de Amenazas” (Estadísticos Descriptivos).

	N	Media	Desviación Estandard	Mínimo	Máximo
Costo	45	2,9778	,91674	1,00	5,00
Beneficio	45	4,6000	,57997	3,00	5,00

Fuente. Elaboración propia.

6.3.2 Resumen de la Prueba de Rangos de Wilcoxon

En el siguiente cuadro se muestra el resumen de los resultados obtenidos de la prueba de rangos de Wilcoxon para las muestras relacionadas (costo-beneficio) de incorporar la actividad de “Modelado de Amenazas” al proceso de desarrollo de software.

Cuadro 15. Resumen resultado de prueba de Wilcoxon

Resumen de prueba de rangos con signo de Wilcoxon para muestras relacionadas	
N total	45
Estadístico de prueba	0.0
Error estándar	0.137
Estadístico de prueba estandarizado	11.871
Sig. asintótica (prueba bilateral)	7.58e-08

Fuente. Elaboración propia.

Dado que el p-valor es extremadamente bajo (7.58e-08), rechazamos la hipótesis nula (H_0) con un nivel de significancia de 0.05. Esto indica que hay diferencias significativas entre las observaciones de costo y beneficio al aplicar la actividad de seguridad de “Modelado de Amenazas” en el proceso de desarrollo de software, (ver Cuadro 15).

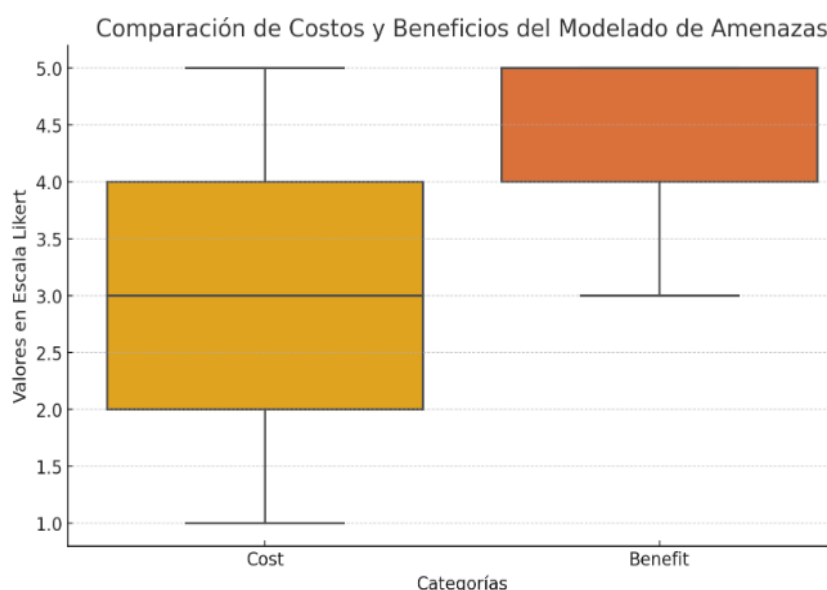


Figura 20. Gráfico de cajas de la comparación de Costo y Beneficio de Incorporar el Modelado de Amenazas.

El gráfico de cajas (ver Figura 20) compara las distribuciones de los valores de costo y beneficio para la actividad de seguridad de “Modelado de Amenazas”. Este gráfico destaca las diferencias entre las dos categorías, con el beneficio mostrando valores más altos en la escala Likert en comparación con el costo.

6.3.3 Interpretación de la prueba de rangos de Wilcoxon:

Podemos decir que, como el valor de p (Sig. asintótica bilateral) es menor que 0,05, entonces se rechaza la hipótesis nula y se concluye que Sí hay diferencias entre costo y beneficio al aplicar la actividad de seguridad de modelado de amenazas en el marco de trabajo propuesto en conclusión los resultados del análisis estadístico y la visualización (gráfico de cajas), indican una diferencia significativa entre las percepciones de costo y beneficio. Esta diferencia sugiere que los encuestados perciben a la actividad de seguridad de “Modelado de Amenazas” como una actividad **Muy Beneficiosa** a pesar de ser **Costoso**.

6.4 Análisis de la Dimensión Tiempo

Para poder calcular la media en base a los resultados obtenidos de la encuesta (**sección 6.2**), se ha tomado en cuenta la media estadística de datos agrupados para los rangos de 0-2, de 2-5, de 5-8, de 8-10 y mayor a 10 horas. Por ejemplo, para la actividad de seguridad “Educación y sensibilización en materia de seguridad (entrenamiento en políticas y actividades de desarrollo seguro)”, obtenemos su media a partir de los resultados obtenidos de la encuesta que se muestran en el Cuadro 16:

Cuadro 16. Valores para el cálculo de media de datos agrupados

Rango de Tiempo Adicional (Horas)	x_i	f_i	$x_i \cdot f_i$
Mayor a 10	11	6	66
(8-10)	9	9	81
(5-8)	6.5	15	97.5
(2-5)	3.5	13	45.5
(0-2)	1	1	1
Sumas		44	291

Fuente. Elaboración propia.

Donde:

x_i = Valor representativo del intervalo (marca de clase)

f_i = Frecuencia absoluta

La media para datos agrupado se calcula usando la siguiente fórmula:

$$\bar{x} = \sum \frac{x_i * f_i}{n}$$

Reemplazando:

$$\bar{x} = (291)/44$$

$$\bar{x}=6.61$$

Tenemos que para la actividad de seguridad “Educación y sensibilización en materia de seguridad” el tiempo adicional, para su implementación, considerado por los expertos y/o profesionales consultados es de 6.61 horas. Usando el mismo procedimiento para obtener la media de las otras actividades de seguridad se tiene los resultados que se muestran en el Cuadro 17.

De acuerdo con el resultado obtenido para la actividad de seguridad 'Educación y sensibilización en materia de seguridad (ver Cuadro 16), se puede afirmar que los expertos y profesionales consultados estimaron que incorporar esta actividad de seguridad demanda un tiempo adicional de **6.61 horas**. Aplicamos el mismo procedimiento para obtener la media de las todas las actividades de seguridad que forman parte del marco de trabajo propuesto, los resultados se muestran en el Cuadro 17 y Figura 21:

Cuadro 17. Resultado de las medias (horas) de tiempo adicional por cada una de las actividades de seguridad que se incorporan al marco de trabajo propuesto

Nº	Actividades de Seguridad	Tiempo (horas)
1	Educación y Sensibilización en Materia de Seguridad	6.61
2	Identificación de Roles	5.04
3	Modelado de amenazas	6.80
4	Identificación de ataques	6.10
5	Análisis de requisitos de seguridad	6.06
6	Análisis estático de código	7.47
7	Pruebas de penetración	7.28
8	Planificación de respuesta a incidentes	6.06
Total		51.42

Fuente. Elaboración propia.

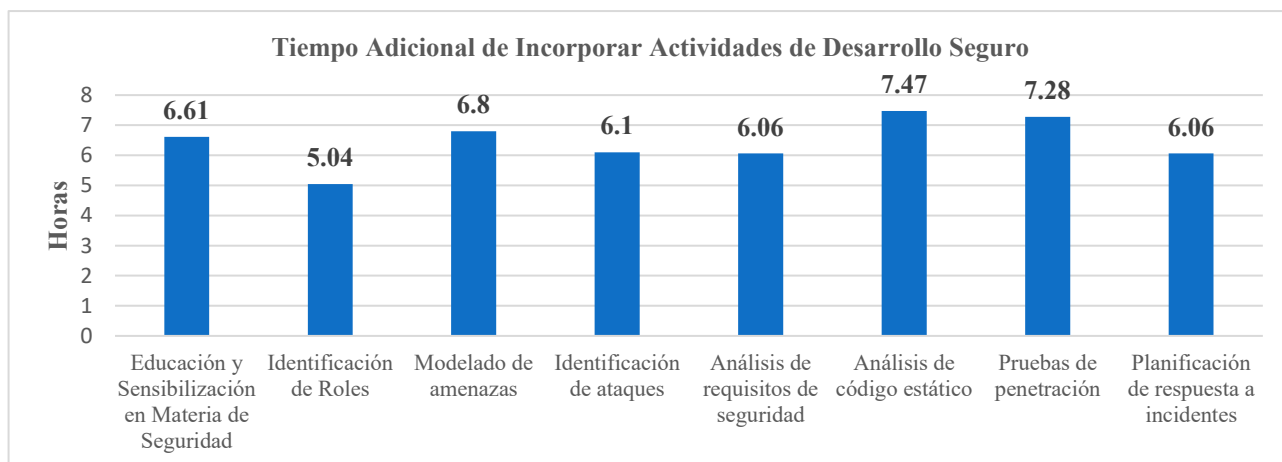


Figura 21. Tiempo adicional por sprint al aplicar el marco de trabajo propuesto.

De acuerdo con los resultados se tiene que incorporar las 8 actividades de seguridad en el proceso de desarrollo de software usando SCRUM (marco de trabajo propuesto), es de **51.42** horas, considerando que un día consta de 8 horas de trabajo ideales, en días da como resultado **6.43** días adicionales por sprint.

6.5 Grado de Agilidad al Aplicar el Marco de Trabajo Propuesto

Para validar en términos de agilidad el marco de trabajo propuesto se ha tomado en cuenta los siguientes factores de ponderación: años de experiencia en el desarrollo de software usando SCRUM, incorporación de un especialista en desarrollo de software seguro y tiempo adicional (dimensión tiempo).

De acuerdo con el trabajo de investigación de los autores (Maqsood, & Bondavalli, 2020) el grado de agilidad luego de aplicar las actividades de seguridad a un proceso de desarrollo de software usando SCRUM se calcula de acuerdo con la siguiente fórmula:

$$AAAS = [(ART1 + ART2 + \dots + ARTn) \div n] * AOM$$

Donde por sus siglas en inglés:

AAAS: Agility After Application of Security (agilidad después de aplicar actividades de seguridad)

ART: Agility Reduction Tolerance (tolerancia a la reducción de agilidad, calculado como grado de agilidad de cada actividad de seguridad)

AOM: Actual Agility of Model (agilidad de actual de SCRUM 0.7)

Reemplazamos en la fórmula los valores de ART de las ocho actividades de seguridad tomadas en cuenta en el marco de trabajo propuesto.

$$AAAS = [(0.75 + 0.89 + 0.87 + 0.83 + 0.88 + 0.85 + 0.86 + 0.81) \div 8] * 0.7$$

$$AAAS = (6.74 \div 8) * 0.7$$

$$AAAS = 0.59$$

El valor del grado de agilidad después de aplicar actividades de seguridad (AAAS), en el trabajo de investigación de los autores (Maqsood, & Bondavalli, 2020) es de **0.59**.

6.5.1 Cálculo del factor de ponderación de incorporación de un especialista en desarrollo de software seguro respecto al grado de agilidad al aplicar el marco de trabajo propuesto

Para el cálculo del factor de ponderación de incorporación de un especialista en desarrollo de software seguro basado en su experiencia desarrollando software con SCRUM se toma como base el resultado de la encuesta aplicada en el presente trabajo

de investigación, y la fórmula aplicada en el trabajo de investigación de los autores (Maqsood, & Bondavalli, 2020), usando una ponderación del 1 al 5:

La ponderación 5 se asigna a la categoría A que corresponde a los profesionales o especialistas con más de 10 años de experiencia, la ponderación 4 a la categoría B que corresponde a los profesionales o especialistas con más de 7 y menos de 10 años de experiencia, la ponderación 3 a la categoría C que corresponde a los profesionales o especialistas con más de 5 y menos de 7 años de experiencia, la ponderación 2 a la categoría D que corresponde a los profesionales o especialistas con más de 3 y menos de 5 años de experiencia, la ponderación 1 a la categoría E que corresponde a los profesionales o especialistas con menos de 3 años de experiencia.

Formula basada en el trabajo de investigación de (Maqsood, & Bondavalli, 2020):

$$\text{FIEDS} = \{[(\text{Suma de valores de categoría A} \times 5 + \text{Suma de valores de categoría B} \times 4 + \text{Suma de valores de categoría C} \times 3 + \text{Suma de valores de categoría D} \times 2 + \text{Suma de valores de categoría E} \times 1) \div (\text{Frecuencia de categoría A} \times 5 + \text{Frecuencia de categoría B} \times 4 + \text{Frecuencia de categoría C} \times 3 + \text{Frecuencia de categoría D} \times 2 + \text{Frecuencia de categoría E} \times 1)] \div 8\} \times \text{AAAS}$$

FIEDS=Factor de Incorporación de Especialista en Desarrollo Seguro

Del resultado de la encuesta se tiene:

Frecuencia de categoría A=5

Frecuencia de categoría B=6

Frecuencia de categoría C=8

Frecuencia de categoría D=18

Frecuencia de categoría E=8

La suma de valores (para la incorporación de un especialista en desarrollo de software seguro), marcados por los expertos de la categoría A=23, la suma de valores marcados por los expertos de la categoría B=29, la suma de valores marcados por los expertos de la categoría C=39, la suma de valores marcados por los expertos de la categoría D=77, la suma de valores marcados por los expertos de la categoría E=36.

Reemplazando en la formula:

$$\text{FIEDS} = \{[(23*5+29*4+39*3+77*2+36*1) \div (5*5+6*4+8*3+18*2+8*1)] \div 8\} * \text{AAAS}$$

$$\text{FIEDS} = [(538 \div 117) \div 8] * \text{AAAS}$$

$$\text{FIEDS} = (0.575) (0.59)$$

$$\text{FIEDS} = 0.34$$

Por tanto, el valor obtenido para el factor de incorporación de especialista en desarrollo seguro (FIEDS), en el marco de trabajo propuesto es igual a **0.34**.

6.5.2 Cálculo del factor de ponderación de la dimensión tiempo

El factor de ponderación en función al tiempo adicional por sprint al aplicar el marco de trabajo propuesto se obtiene tomando en cuenta la dimensión tiempo (expresado en días), luego de incorporar las actividades de seguridad durante el proceso de desarrollo (**sección 6.4**), y aplicando la siguiente fórmula:

$$\text{FDT} = (\text{DT} \div \text{Días ideales por sprint}) * \text{AAAS}$$

Donde:

FDT= Factor de ponderación de la dimensión tiempo

DT= Dimensión tiempo (**sección 6.4**)

Reemplazando los valores en la fórmula:

$$\mathbf{FDT = (6.43 \div 30) * 0.59}$$

$$\mathbf{FDT = 0.214 * 0.59}$$

$$\mathbf{FDT = 0.13}$$

Por tanto, luego de aplicar la fórmula se obtiene que el valor del factor de ponderación basado en la dimensión tiempo es de **-0.13** (aproximación al centésimo), con signo negativo dado que la integración de actividades de seguridad requiere un tiempo adicional por sprint.

$$\mathbf{FDT = -0.13}$$

6.5.3 Cálculo del grado de agilidad aplicando los factores de ponderación

El grado de agilidad luego de aplicar el marco de trabajo propuesto (considerando los factores de ponderación), se obtiene sumando: el grado de agilidad obtenido en el trabajo de investigación de (Maqsood, & Bondavalli, 2020), donde AAAS=0.59, el factor de ponderación de incorporar un especialista de desarrollo seguro al proceso y el factor de ponderación de la dimensión tiempo; tomando en cuenta que al incorporar un especialista de desarrollo seguro la agilidad del proceso se ve impactada positivamente en tanto que incorporar las actividades de seguridad demandan de un tiempo adicional por lo que se le asigna un signo negativo (ver Cuadro 18).

Por tanto, por lo descrito en el párrafo anterior el grado de agilidad luego de aplicar el marco de trabajo propuesto (**AAAS'**) se obtiene aplicando la siguiente fórmula

$$AAAS' = AAAS + FIEDS - FDT$$

Cuadro 18. Datos para determinar el grado de agilidad

AAAS	FIEDS (+)	FDT (-)
0.59	0.34	0.13

Fuente. Elaboración propia.

$$AAAS' = 0.59 + 0.34 - 0.13$$

$$AAAS' = 0.80$$

Luego de reemplazar los valores en la fórmula el grado de agilidad aplicando el marco de trabajo propuesto (AAAS') es **0.80**.

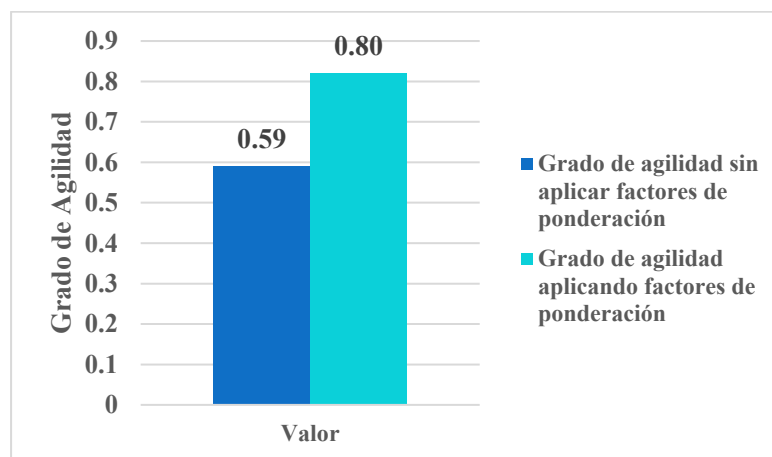


Figura 22. Grado de agilidad al aplicar el marco de trabajo propuesto (aplicando factores de ponderación).

Para determinar qué tan ágil es el marco de trabajo propuesto en una escala de 0 a 1 se tiene:

$$0.9 < AAAS' < 1.0 = \text{Muy Ágil}$$

$0.7 < AAAS' < 0.9 = \text{Ágil}$

$0.5 < AAAS' < 0.7 = \text{Ligeramente Ágil}$

$0.0 < AAAS' < 0.5 = \text{Nada Ágil}$

Dado que $AAAS' = 0.80$, se observa que el marco de trabajo propuesto se mantiene como **Ágil**.

6.6 Implementación del Marco de Trabajo Propuesto

Se inició la aplicación del marco de trabajo propuesto en la adecuación al nuevo marco legal del Sistema Catastral Rural (SCR) en el Ministerio de Desarrollo Agrario y Riego (Sprint 1), este sistema informático consolida la información legal y geoespacial de los predios rurales del Perú. Su objetivo es proveer un registro único, actualizado y accesible para ciudadanos, gobiernos regionales (GORE) y entidades como SUNARP, facilitando la formalización de la propiedad, la titulación y la toma de decisiones sobre desarrollo agrario. La aplicación del marco propuesto al desarrollo del sistema catastral rural abarca las siguientes actividades de desarrollo seguro de software: (1) Políticas de Desarrollo Seguro de Software, (2) Identificación de Roles (3) Modelado de Amenazas, (4) Análisis de Riesgos de Seguridad (Identificación de Ataques) y Análisis de Requisitos de Seguridad, (5) Análisis Estático de Código y Pruebas de Penetración (se detallan en el Anexo 5). Las mejoras identificadas en este Sprint 1 son:

- **Cumplimiento normativo y alineación institucional**
 - El marco propuesto se alinea completamente con la Política SGSI-PO-021 del MIDAGRI (**Anexo 5**), especialmente en lo relativo a:
 - Codificación segura.

- Modelado de amenazas.
 - Análisis de riesgos y requisitos de seguridad.
 - Pruebas de seguridad (SAST/DAST).
 - Gestión de sesiones, autenticación y autorización.
- Esto facilita la adecuación del sistema catastral al marco legal vigente, especialmente en lo relacionado con la protección de datos geoespaciales y personales de productores rurales.
- **Reducción de vulnerabilidades desde fases tempranas**
 - El enfoque propuesto incorpora actividades como modelado de amenazas, análisis estático de código, historias de abuso, y pruebas de penetración, alineado con el OWASP Top 10.
 - Se previene proactivamente:
 - Inyecciones SQL/XSS.
 - Accesos no autorizados a datos catastrales.
 - Fallas de control de acceso o autenticación.
- **Agilidad preservada con seguridad integrada**
 - El marco equilibra seguridad y agilidad, permitiendo a los equipos SCRUM incorporar actividades seguras sin sacrificar velocidad ni flexibilidad, por tanto el impacto sobre la velocidad de entrega por sprint es bajo.
- **Fortalecimiento de capacidades del equipo**
 - Se promueve la incorporación de un especialista en desarrollo seguro en el equipo SCRUM.
 - Mejora la madurez del equipo en seguridad y garantiza el cumplimiento del SGSI.
- **Trazabilidad y resiliencia**
 - Mejora la trazabilidad del ciclo de vida del software y fortalece la capacidad de recuperación ante incidentes mediante:
 - Logging seguro.
 - Respuesta ante incidentes bien definida.
 - Gestión de versiones y cambios segura.

- **Beneficio estimado**

- Reducción significativa del riesgo de ataques y costos derivados (cada vulnerabilidad crítica no detectada puede ocasionar costos significativos en daños y/o sanciones).
- Mayor confianza institucional y cumplimiento de normativa en proyectos financiados por el Estado y cooperantes.

Del análisis de costo-beneficio presentado en el capítulo VI de la tesis (especialmente en la **sección 6.2.4 y 6.3**), los costos identificados de aplicar el marco de trabajo propuesto en el sprint 1 en el desarrollo de la adecuación del Sistema Catastral Rural (SCR) del MIDAGRI:

- **Costo directo (por sprint)**

- Tiempo adicional a las planificadas por todas las actividades de seguridad implementadas en el sprint 1, depende la inversión adicional en contar con personal especializado en prácticas de desarrollo seguro.

- **Inversión adicional**

- La incorporación de un especialista en desarrollo seguro demandará de un presupuesto adicional de: ~S/. 8,000 mensuales.
- La capacitación en desarrollo seguro de software demandará de un presupuesto adicional de: ~S/. 12,000 por taller técnico (para todo el equipo).

CAPÍTULO VII. DISCUSIÓN

De los autores (Maqsood, & Bondavalli, 2020) y (Sharma, & Bawa, 2020) se seleccionaron 8 actividades de seguridad basados en el grado de agilidad, costo beneficio respectivamente y del análisis a los resultados del instrumento del presente trabajo de investigación se obtuvo el tiempo adicional que demanda el incorporar las actividades de seguridad en proceso de desarrollo de software web, así como la relación costo-beneficio de incorporar la actividad “Modelado de Amenazas” (Ver Cuadro 19).

Cuadro 19. Resultado final de incorporar las actividades de seguridad en el marco de trabajo propuesto

N	Actividades de Seguridad	Agilidad	Costo	Beneficio	Tiempo (Horas)
1	Educación y sensibilización en materia de seguridad	0.75	Poco Costoso	Muy Beneficioso	6.61
2	Identificación de Roles	0.89	Costoso	Muy Beneficioso	5.04
3	Modelado de amenazas	0.88	Costoso	Muy Beneficioso	6.80
4	Análisis de riesgos de seguridad (Identificación de ataques)	0.87	Costoso	Muy Beneficioso	6.10
5	Análisis de requisitos de seguridad	0.83	Costoso	Muy Beneficioso	6.06
6	Análisis estático de código	0.85	Poco Costoso	Muy Beneficioso	7.47
7	Pruebas de penetración	0.86	Poco Costoso	Muy Beneficioso	7.28
8	Planificación de respuesta a incidentes	0.81	Poco Costoso	Muy Beneficioso	6.06

Fuente. Elaboración propia.

De los resultados obtenidos en el presente trabajo de investigación, a continuación, se presenta la discusión sobre costo-beneficio, dimensión tiempo, incorporación de un especialista en desarrollo seguro y el grado de agilidad al aplicar el marco de trabajo propuesto.

7.1 Sobre costo y beneficio

Las preguntas sobre costo y beneficio incluidas en la encuesta a profesionales y expertos en desarrollo de software web respecto a la incorporación de la actividad de 'Modelado de Amenazas' en el marco de trabajo propuesto, se realizaron con el objetivo de determinar su viabilidad en términos de costo-beneficio. Esto permite evaluar qué tan costosa y beneficiosa es la incorporación del 'Modelado de Amenazas' durante el proceso de desarrollo y complementarlo con el análisis de costo-beneficio obtenido para las otras siete actividades de seguridad, según el trabajo de Sharma & Bawa (2020). De esta manera, se completa la evaluación costo-beneficio de las ocho actividades de desarrollo seguro consideradas en el marco de trabajo propuesto.

Las encuestas (**subsección 6.3.3**), revelan que la actividad de seguridad “Modelado de Amenazas” presenta un **alto beneficio** en relación con su costo (esfuerzo). En conjunto estas actividades que se incorporan al marco de trabajo propuesto no solo reducen significativamente las vulnerabilidades, sino que también mejoran la calidad del software desarrollado. El beneficio percibido, medido en términos de mitigación de riesgos y aumento de la robustez del software, justifica plenamente los costos iniciales asociados con la implementación de estas prácticas.

7.2 Sobre Dimensión Tiempo

Incorporar las actividades de seguridad en el proceso de desarrollo de software web usando SCRUM demandan de un tiempo adicional del ya requerido para el desarrollo de las actividades relacionadas a las funcionalidades, esto impacta de una manera en la agilidad del proceso toda vez que la velocidad ya no será la misma, para poder determinar en cuanto tiempo en promedio se incrementa (en horas), se realizó la encuesta a los profesionales y/o expertos en desarrollo de software web usando SCRUM. Del análisis a los resultados de la encuesta (**sección 6.4**), se obtiene que el tiempo adicional al aplicar el marco de trabajo propuesto por sprint se incrementa en un aproximado de **51.42** horas, considerando que los días ideales por sprint son 30 y cada día es de 8 horas ideales, el tiempo adicional expresado en días es de **6.43** días, lo que representa un incremento en términos porcentuales de **21.4%** del tiempo por sprint, por tanto, en base a los resultados de la encuesta el factor de ponderación de la dimensión tiempo obtenido es de **-0.13** (**sección 6.4**), con signo negativo dado que la integración de actividades de seguridad requiere un tiempo adicional por sprint, pero este tiempo es manejable y no afecta negativamente la productividad del equipo. Actividades como el análisis estático de código y la planificación de respuesta a incidentes fueron identificadas como las que requieren más tiempo, mientras que la identificación de roles y la educación en seguridad pueden integrarse más fácilmente sin grandes interrupciones.

7.3 Sobre la Incorporación de un Especialista en Desarrollo Seguro

El marco de trabajo propuesto incluye la incorporación de un especialista en desarrollo de software seguro, cuyo rol principal es desarrollar y mejorar las políticas y buenas prácticas de seguridad, capacitar al equipo en desarrollo seguro, y garantizar el cumplimiento de estas políticas y prácticas. En resumen, este especialista asegura la aplicación del marco de trabajo propuesto a lo largo de todo el proceso de desarrollo de software. La participación del especialista en desarrollo seguro impacta positivamente en la agilidad del proceso, ya que fortalece al equipo, haciéndolo más competente en las prácticas y actividades de desarrollo seguro.

Para evaluar la aceptación de esta incorporación, se planteó en la encuesta una pregunta a los expertos y especialistas sobre su acuerdo con la inclusión de un especialista en desarrollo seguro en el equipo de SCRUM. Los resultados de la encuesta (**subsección 6.2.3**) mostraron que el **60%** está totalmente de acuerdo, el **37.8%** está de acuerdo, y solo el **2.2%** está totalmente en desacuerdo. En general, el **97.8%** de los encuestados considera que se debe incorporar un especialista en desarrollo seguro al equipo.

Dado este resultado, y basado en el trabajo de investigación de Maqsood y Bondavalli (2020), se calculó el factor de ponderación para la incorporación del especialista en desarrollo seguro, obteniendo un valor de **0.34** (**subsección 6.5.1**). Este factor se utiliza para medir el grado de agilidad al aplicar el marco de trabajo propuesto (**subsección 6.5.3**).

7.4 Sobre Grado de Agilidad

Para calcular el grado de agilidad al aplicar el marco de trabajo propuesto, se consideraron dos factores: la incorporación de un especialista en desarrollo seguro y la dimensión del tiempo (el tiempo adicional requerido para implementar las actividades de desarrollo seguro). Basándose en el trabajo de investigación de Maqsood y Bondavalli (2020), se calculó el grado de agilidad (**AAAS'**), tras la implementación del modelo, obteniendo un valor de **0.80** (**Subsección 6.5.3**). Este resultado destaca el papel decisivo del especialista en desarrollo seguro en la aplicación del marco de trabajo propuesto.

El grado de agilidad obtenido de **0.80** demuestra que **el marco de trabajo propuesto mantiene su carácter ágil**, (ver Figura 22). Además, en comparación con un grado de agilidad de **0.59** sin considerar los factores de ponderación (**AAAS**), se observa una mejora de **0.21** puntos. Esto confirma la importancia de tener en cuenta los factores de ponderación para lograr un desarrollo ágil y seguro.

CAPÍTULO VIII. CONCLUSIONES Y TRABAJOS FUTUROS

8.1 Conclusiones

- Se ha logrado diseñar y validar un marco de trabajo ligero para el desarrollo de software web seguro en metodologías ágiles, que integre de manera efectiva un conjunto de actividades de seguridad de bajo costo y alto beneficio, sin comprometer el grado de agilidad del proceso, con el fin de mitigar vulnerabilidades desde las primeras fases del ciclo de vida del desarrollo.
- Existe una preocupación constante por parte de la comunidad relacionada al desarrollo de software en encontrar alternativas de integración de actividades de seguridad en el proceso de desarrollo de software usando SCRUM (procesos ágiles).
- Considerando el análisis de costo-beneficio al aplicar el marco de trabajo propuesto, se concluye que, aunque la incorporación de actividades de seguridad durante el proceso de desarrollo requiere un esfuerzo adicional, la literatura y los resultados obtenidos demuestran que su **implementación es altamente beneficiosa**. Esto significa que aplicar el marco de trabajo propuesto **resulta en un producto de software web significativamente más seguro**.
- Incorporar un especialista en desarrollo de software seguro y que adicional a ello cuente con experiencia en procesos de desarrollo como SCRUM es un factor crucial que mejora significativamente la seguridad del producto final, el resultado de la encuesta aplicada en el presente trabajo de investigación revela que el **60%** de encuestados está totalmente de acuerdo con su incorporación.

- Las encuestas revelaron que los especialistas en desarrollo de software seguro no solo mejoran la calidad del software, sino que también aumentan la conciencia y la cultura de seguridad dentro del equipo. Esto tiene un efecto multiplicador, ya que todos los miembros del equipo se vuelven más conscientes de los aspectos de seguridad, lo que reduce las probabilidades de cometer errores que puedan llevar a vulnerabilidades.
- Un aporte y también factor crucial para obtener el grado de agilidad en la aplicación del marco de trabajo propuesto es haber considerado el factor tiempo adicional que demanda la incorporación de actividades de seguridad durante el proceso de desarrollo de software usando SCRUM.
- El tiempo adicional requerido para incorporar actividades de seguridad varía según la actividad, pero en general, una planificación adecuada y la integración temprana de estas actividades minimizan el impacto negativo en la velocidad del equipo. Actividades como la educación en seguridad y la identificación de roles pueden ser más fáciles de integrar sin interrumpir significativamente el flujo de trabajo ágil, en conclusión, se puede afirmar que el tiempo adicional por sprint puede ser mitigado mediante una planificación adecuada. Esto asegura que las actividades de seguridad se conviertan en una parte integral del proceso de desarrollo, minimizando las interrupciones y manteniendo la eficiencia del equipo ágil.
- La capacidad del marco propuesto para mantener un alto grado de agilidad mientras se integran actividades de seguridad demuestra que es posible combinar la seguridad con la agilidad, **el resultado del presente trabajo** de investigación (**subsección 6.5.3**), tomando en cuenta el factor de incorporación de un especialista en desarrollo de software seguro y el factor de tiempo adicional que demandan la incorporación las actividades de seguridad así lo demuestra. Esto refuerza la viabilidad del marco y su aplicabilidad en entornos de desarrollo de software que requieren tanto seguridad como rapidez en la entrega.

- Los resultados de las encuestas también destacaron que la implementación de estas actividades de seguridad, aunque tiene un costo asociado, es percibida como una inversión necesaria para garantizar la seguridad y fiabilidad del software. La percepción de un alto beneficio en términos de reducción de riesgos y aumento de la confianza en el producto final refuerza la viabilidad del marco propuesto.
- Los resultados obtenidos de las encuestas respaldan la viabilidad y los beneficios de integrar prácticas de seguridad en el desarrollo ágil, asegurando que el software producido sea robusto y confiable sin comprometer la eficiencia del proceso ágil.
- La implementación piloto del marco de trabajo en el desarrollo del Sistema Catastral Rural (SCR) del MIDAGRI demuestra su viabilidad técnica y organizacional. Durante el desarrollo (tres sprints completos), se integraron de manera incremental las actividades de seguridad (descritas en el Anexo 5), sin comprometer el ritmo de entrega. La integración de estas actividades de seguridad evidenció la identificación y reducción de vulnerabilidades críticas y de severidad media.

Entre los **aciertos** se destacan:

- **Shift-left efectivo:** Realizar el modelado de amenazas al inicio de cada sprint permitió detectar fallos de diseño antes de la codificación.
- **Cultura colaborativa:** Las retrospectivas focalizadas en seguridad incrementaron la participación del equipo y mejoraron la trazabilidad de las historias de usuario seguras.
- **Backlog “híbrido”:** Incorporar las historias de usuario relacionados a la seguridad, como resultado del análisis de requisitos, en una única lista de tareas pendientes disminuye el retrabajo y el costo de corregir fallos tardíos.

Las oportunidades de mejora identificadas incluyen:

- Profundizar la formación del equipo en prácticas DevSecOps para reducir la dependencia del especialista externo.

- Ampliar la cobertura de pruebas de carga y resiliencia frente a ataques de denegación de servicio, aspecto aún poco automatizado.
- Refinar los indicadores de valor de negocio, de modo que las métricas de seguridad se correlacionen con la productividad y los niveles de servicio exigidos por MIDAGRI.
- Automatizar los controles de análisis SAST/DAST en la misma “pipeline” para evitar retrabajos.

8.2 Aportes de Este Trabajo

El aporte de esta tesis es la propuesta de un **marco de trabajo** que permite desarrollar **software web seguro** sin comprometer el grado de agilidad de las **metodologías ágiles**, particularmente **SCRUM**, al incorporar actividades de seguridad, tomando como referencia los trabajos de investigación de (Maqsood, & Bondavalli, 2020) y (Sharma, & Bawa, 2020). En la actualidad, el desarrollo ágil de software prioriza la entrega rápida de productos, lo que a menudo deja de lado la seguridad, generando vulnerabilidades que pueden ser explotadas por atacantes. La investigación aborda esta deficiencia y plantea una solución efectiva y aplicable en la industria del desarrollo del software.

8.2.1 *¿Qué problema resuelve esta tesis?*

En el desarrollo de software ágil, la seguridad suele tratarse al final del proceso o cuando ya han ocurrido incidentes. Esto genera **altos costos** en correcciones y **riesgos significativos** para las empresas y sus clientes. La tesis resuelve este problema al integrar actividades de seguridad dentro del flujo de trabajo de **SCRUM**, asegurando

que la protección del software no sea una tarea secundaria, sino una parte fundamental del desarrollo desde el inicio.

8.2.2 ¿Cuál es su contribución principal?

La investigación propone **un conjunto de actividades de seguridad** que pueden incorporarse en cada fase del proceso de desarrollo ágil sin afectar la productividad ni generar retrasos significativos. Se identificaron **ocho prácticas de seguridad** que tienen un **bajo costo y alto beneficio**, basadas en estudios previos y evaluaciones empíricas.

Para validar la efectividad del marco propuesto, se realizó una encuesta entre profesionales del sector, midiendo su impacto en tres aspectos clave:

1. **Costo-beneficio** de las actividades de seguridad.
2. **Tiempo adicional requerido** para implementarlas en SCRUM.
3. **Grado de agilidad** mantenido al aplicar la seguridad en el proceso de desarrollo.

Los resultados demostraron que es posible **mantener la rapidez de SCRUM** y, al mismo tiempo, **reducir vulnerabilidades de seguridad**, optimizando los procesos de desarrollo sin que la seguridad se convierta en una barrera.

8.2.3 Impacto y beneficios del aporte

- **Reducción de vulnerabilidades:** Las empresas podrán desarrollar software más seguro sin afectar sus plazos de entrega.
- **Menores costos de corrección:** Al integrar seguridad desde el inicio, se evitan gastos innecesarios en reparaciones posteriores.
- **Mayor confianza en los productos:** Se mejora la reputación y fiabilidad de las aplicaciones web, evitando incidentes de seguridad que puedan afectar a los usuarios.

- **Aplicabilidad real en la industria:** El marco de trabajo puede ser adoptado fácilmente en empresas que usan metodologías ágiles.

8.3 Trabajos Futuros

- El marco de trabajo propuesto en el presente trabajo de investigación es un referente en el desarrollo de software seguro y puede ser la base para el desarrollo futuro de otros marcos de trabajo usando otras dimensiones, factores y variables que en este trabajo de investigación no se han contemplado, como por ejemplo considerar la dimensión costo en términos monetarios de incorporar actividades de seguridad en procesos de desarrollo ágil.
- El marco de trabajo propuesto ofrece una base sólida para integrar la inteligencia artificial (IA) en los procesos de desarrollo de software seguro. Utilizando IA, se pueden analizar eficazmente los reportes de las principales amenazas, vulnerabilidades y malas prácticas en el desarrollo de software, así como realizar análisis de riesgos más detallados. El objetivo final de esta integración es alcanzar una mayor precisión y efectividad en los resultados del desarrollo de software seguro, mejorando así la seguridad y la confiabilidad de los sistemas desarrollados, en trabajos futuros se puede integrar la inteligencia artificial (IA) por ejemplo para:
 - Mediante algoritmos de aprendizaje profundo mejorar la precisión en la detección de vulnerabilidades durante el análisis estático de código.
 - Incorporar sistemas de IA que automaticen las pruebas de penetración, simulando ataques sofisticados que evolucionen constantemente para descubrir nuevas vulnerabilidades.

- Implementar IA para prever incidentes de seguridad y planificar respuestas más eficaces, utilizando análisis predictivos basados en datos históricos de seguridad.
- Incorporar chatbots o asistentes virtuales impulsados por IA para proporcionar educación continua y asistencia en tiempo real sobre prácticas de seguridad a los desarrolladores.
- Incorporar herramientas de IA para optimizar la planificación de sprints, incorporando actividades de seguridad de manera eficiente sin afectar negativamente la productividad del equipo.

REFERENCIAS BIBLIOGRÁFICAS

- Ababneh, A. A., Al-Rudaini, M. A., Abdallah, A., Khasawneh, M. A., & Khasawneh, F. A. (2021, March 7-11).** Improving software security in extreme programming methodology. *In Proceedings of the 11th Annual International Conference on Industrial Engineering and Operations Management* (pp. 588-598). IEOM Society International.
- Adel, M. H., Alqatawna, J., & Mohammad, A. (2017, May).** Secure software engineering: Evaluation of emerging trends. *In 2017 8th International Conference on Information Technology (ICIT)* (pp. 646-651). IEEE. <https://doi.org/10.1109/ICITECH.2017.8079952>
- Alami, A., Zahedi, M., & Krancher, O. (2023).** Antecedents of psychological safety in agile software-development teams. *Information and Software Technology*, 162, 107267. <https://doi.org/10.1016/j.infsof.2023.107267>
- Alarcón Loayza, L. A. (2024).** *Un framework para apoyar las decisiones de outsourcing de SI/TI en instituciones públicas del Perú usando el Proceso Analítico Jerárquico AHP* [Master's thesis, Universidad Nacional Mayor de San Marcos]. Cybertesis UNMSM. <https://cybertesis.unmsm.edu.pe/item/4f5e3e1e-5397-4968-877e-0e947cc37b3b>
- Alenezi, M., Basit, H. A., Beg, M. A., & Shaukat, M. S. (2022).** Synthesizing secure software-development activities for linear and agile lifecycle models. *Software: Practice and Experience*, 52(12), e3072. <https://doi.org/10.1002/spe.3072>
- Allen, J. H., Barnum, S., Ellison, R. J., McGraw, G., & Mead, N. R. (2008).** *Software security engineering: A guide for project managers*. Addison-Wesley Professional.
- Andrade, R., Torres, J., Ortiz-Garcés, I., Miño, J., & Almeida, L. (2023).** An exploratory study gathering security requirements for the software development process. *Electronics*, 12(17), 3594.

<https://doi.org/10.3390/electronics12173594>

Baldassarre, M., Caivano, D., & Piccinno, A. (2021). Integrating security and privacy in HCD-Scrum. In *CHIItaly 2021: 14th Biannual Conference of the Italian SIGCHI Chapter (CHIItaly '21)* (pp. 1-5). ACM.
<https://doi.org/10.1145/3464385.3464746>

Barros, L., Tam, C., & Varajão, J. (2024). Agile software-development projects: Unveiling the human-related critical success factors. *Information and Software Technology*, 170, 107432.
<https://doi.org/10.1016/j.infsof.2024.107432>

Bezerra, C., Sampaio, S., & Marinho, M. (2020). Secure agile software development: Policies and practices for agile teams. In *Proceedings of the International Conference on the Quality of Information and Communications Technology* (pp. 343-357). Springer.
https://doi.org/10.1007/978-3-030-58793-2_28

Brasoveanu, R., Karabulut, Y., & Pashchenko, I. (2022, agosto). Security maturity self-assessment framework for software development lifecycle. In *ARES '22: Proceedings of the 17th International Conference on Availability, Reliability and Security* (pp. 1-8). ACM.
<https://doi.org/10.1145/3538969.3543806>

CVEDetails. (n.d.). *Vulnerabilities by types*. En CVE Details. Recuperado el 1 de julio de 2025 de <https://www.cvedetails.com/vulnerabilities-by-types.php>

Díaz Álvarez, J. K. (2020). *Factores para implementar prácticas para el alineamiento estratégico de TI en el sector público: Un enfoque desde el gobierno de TI* [Tesis de maestría, Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería de Sistemas e Informática, Unidad de Posgrado]. Cybertesis UNMSM.
<https://cybertesis.unmsm.edu.pe/community/d2dd9c8c-0bfd-4502-aeb1-52184b55f950/browse/title>

- European Union Agency for Cybersecurity. (2019).** *Good practices for security of Internet of Things in the context of smart manufacturing.* ENISA. <https://www.enisa.europa.eu/publications/good-practices-for-security-of-iot>
- Herath, H. M. K. K. B., & Wimalaratne, G. D. S. P. (2018).** A collaborative platform featuring visibility, tracking, monitoring and awareness for building security. In *International Journal of Computer and Communication Engineering*, 7(4), 145–166. <https://doi.org/10.17706/ijcce.2018.7.4.145-166>
- Hustad, E., Nakayama, M., Sutcliffe, N., & Beckfield, M. (2022).** Knowledge mobilization in agile information-systems projects: A literature analysis. In *Proceedings of the 23rd European Conference on Knowledge Management* (pp. 533-541). <https://doi.org/10.34190/eckm.23.1.561>
- Julca Mejía, W. (2024).** *Sistema de recomendación basado en modelos híbridos de filtrado colaborativo para jueces de programación en línea* [Tesis de maestría, Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería de Sistemas e Informática, Unidad de Posgrado]. Repositorio institucional Cybertesis UNMSM. <https://cybertesis.unmsm.edu.pe/item/ffa74701-4716-4c2d-9a6c-1ca834733d57>
- Lange, F., & Kunz, I. (2024).** Evolution of secure development lifecycles and maturity models in the context of hosted solutions. *Journal of Software: Evolution and Process*, 36(12), e2711. <https://doi.org/10.1002/smr.2711>
- Likert, R. (1932).** A technique for the measurement of attitudes. *Archives of Psychology*, 22(140), 1-55.
- Maier, P., Ma, Z., & Bloem, R. (2017).** Towards a secure Scrum process for agile web-application development. In *Proceedings of the 12th International Conference on Availability, Reliability and Security* (Article 49, 1-8). ACM. <https://doi.org/10.1145/3098954.3103171>

- Maqsood, M., & Bondavalli, A. (2020, April 23-24).** Agility of security practices and agile process models: An evaluation of cost for incorporating security in agile process models. *In Proceedings of the 15th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2020)* (pp. 15-26). SCITEPRESS. <http://dx.doi.org/10.5220/0009356403310338>
- Mažeika, D., & Butleris, R. (2020).** Integrating security requirements engineering into MBSE: Profile and guidelines. *Security and Communication Networks*, 2020, Article ID 5137625, 1–12. <https://doi.org/10.1155/2020/5137625>
- Melgarejo Solis, R. A. (2022).** *Model business canvas para implementar cursos online masivos abiertos (MOOC) en empresas de servicios e-learning* [Tesis de maestría, Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería de Sistemas e Informática, Unidad de Posgrado]. Repositorio institucional Cybertesis UNMSM. <https://cybertesis.unmsm.edu.pe/item/fa589dd0-23a2-4c05-bd6c-966af718a0fa>
- MITRE Corporation. (n.d.).** *The CLASP application security process*. Recuperado el 16 de abril de 2025 de <https://cwe.mitre.org/documents/sources/TheCLASPApplicationSecurityProcess.pdf>
- MITRE. (2024).** *2024 CWE top 25 most dangerous software weaknesses*. Recuperado el 9 de julio de 2025 de https://cwe.mitre.org/top25/archive/2024/2024_cwe_top25.html
- Mohino, J., Bermejo, J., Bermejo, J. R., & Sicilia, J. (2019).** The application of a new secure software development life cycle (S-SDLC) with agile methodologies. *Electronics*, 8(11), 1218. <https://doi.org/10.3390/electronics8111218>
- Moyón, F., Méndez-Fernández, D., Beckers, K., & Klepper, S. (2020, November).** How to integrate security compliance requirements with

agile software engineering at scale? In *Product-Focused Software Process Improvement: Proceedings of the 21st International Conference (PROFES 2020)* (pp. 3-19). Springer. https://doi.org/10.1007/978-3-030-64148-1_5

Moyo, S., & Mnkandla, E. (2020). A novel lightweight solo software-development methodology with optimum security practices. *IEEE Access*, 8, 32777-32789. <https://doi.org/10.1109/ACCESS.2020.2971000>

McKnight, P. E., & Najab, J. (2010). Mann-Whitney U test. In N. J. Salkind (Ed.), *Encyclopedia of Research Design* (pp. 715–718). SAGE Publications.

McGraw, G. (2006). *Software security: Building security in*. Addison-Wesley Professional. ISBN 9780321356703

Oyetoyan, T. D., Jaatun, M. G., & Cruzes, D. S. (2017). A lightweight measurement of software security skills, usage and training needs in agile teams. In *Proceedings of the 12th International Conference on Availability, Reliability and Security (ARES 2017)*. ACM. <https://doi.org/10.4018/IJSSE.2017010101>

Özkan, N., & Gok, M. S. (2022). Definition synthesis of agility in software development: Comprehensive review from theory to practice. *International Journal of Modern Education and Computer Science*, 14(3), 26–44. <https://doi.org/10.5815/ijmecs.2022.03.02>

Ouriques, R. A. B., Gorschek, T., Méndez, D., & Fagerholm, F. (2023). Connecting the dots of knowledge in agile software development. arXiv. <https://doi.org/10.48550/arXiv.2306.05742>

Ouriques, R., Machado, L., & Vicari, R. (2023). Towards the integration of security practices in agile software-development lifecycles. *Applied Sciences*, 13(7), 4578. <https://doi.org/10.3390/app13074578>

- OWASP Foundation. (2023).** *Penetration testing. OWASP Web Security Testing Guide (WSTG) v4.2.* OWASP. Recuperado el 16 de abril de 2025 de <https://owasp.org/www-project-web-security-testing-guide/stable/2-Introduction/README#penetration-testing>
- OWASP Foundation. (2022).** *OWASP software assurance maturity model.* Recuperado el 16 de abril de 2025 de <https://owasp.org/www-project-samm/>
- OWASP Foundation. (2021).** *Threat modeling.* Recuperado el 16 de abril de 2025, de https://owasp.org/www-community/Threat_Modeling
- OWASP Foundation. (2021).** *OWASP top 10: Los 10 principales riesgos de seguridad en aplicaciones web [Versión en español].* Recuperado el 16 de abril de 2025 de <https://owasp.org/Top10/es/>
- OWASP Foundation. (n.d.).** *Threat modeling process.* OWASP. Recuperado el 16 de abril de 2025 de https://owasp.org/www-community/Threat_Modeling_Process
- OWASP Foundation. (n.d.).** *Static code analysis.* OWASP. Recuperado el 16 de abril de 2025 de https://owasp.org/www-community/controls/Static_Code_Analysis
- OWASP Foundation. (n.d.).** *Source code analysis tools.* OWASP. Recuperado el 16 de abril de 2025 de https://owasp.org/www-community/Source_Code_Analysis_Tools
- Prates, L., & Pereira, R. (2024).** DevSecOps practices and tools: A multivocal literature review. *International Journal of Information Security*, 24, Article 11. <https://doi.org/10.1007/s10207-024-00914-z>
- Pressman, R. S., & Maxim, B. R. (2021).** *Ingeniería de software: Un enfoque práctico* (9.^a ed.). McGraw-Hill Interamericana.
- Rindell, K., Hyrynsalmi, S., & Leppänen, V. (2018).** Aligning security objectives with agile software development. En *Proceedings of the 19th International Conference on Agile Software Development: Companion*

(p. 9). ACM. <https://doi.org/10.1145/3234152.3234187>

Rindell, K. (2019). *Development of secure software: Rationale, standards and practices* [Tesis doctoral, University of Turku]. Turku Centre for Computer Science (TUCS). <https://www.utupub.fi/handle/10024/147785>

Rindell, K., Ruohonen, J., Holvitie, J., Hyrynsalmi, S., & Leppänen, V. (2021). Security in agile software development: A practitioner survey. *Information and Software Technology*, 131, 106488. <https://doi.org/10.1016/j.infsof.2020.106488>

Sharma, A., & Bawa, R. (2020). Identification and integration of security activities for secure agile development. *International Journal of Information Technology*, 14(2), 1117–1130. <https://doi.org/10.1007/s41870-020-00446-4>

Shinde, N., & Kulkarni, P. (2021). Cyber incident response and planning: A flexible approach. *Computer Fraud & Security*, 2021(1), 14–19. [https://doi.org/10.1016/S1361-3723\(21\)00009-9](https://doi.org/10.1016/S1361-3723(21)00009-9)

Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide: The definitive guide to Scrum: The rules of the game* (2020 edition) [PDF]. Scrum.org. <https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf>

Tøndel, I. A., Cruzes, D. S., Jaatun, M. G., & Sindre, G. (2022). Influencing the security prioritisation of an agile software development project. *Computers & Security*, 118, 102744. <https://doi.org/10.1016/j.cose.2022.102744>

Tøndel, I. A., & Jaatun, M. G. (2020). Towards a conceptual framework for security requirements work in agile software development. *International Journal of Systems and Software Security and Protection*, 11(1), 33–62. <https://orcid.org/0000-0001-7127-6694>

- Theurich, P., Witt, J., & Richter, S. (2023).** *Threat modeling practices in agile environments: Challenges and enablers.* *Informatik-Spektrum*, 46(5), 453–465. <https://doi.org/10.1007/s00287-023-01549-5>
- Zhao, X., Clear, T., & Lal, R. (2024).** *Identifying the primary dimensions of DevSecOps: A multivocal literature review.* *Information and Software Technology*, 164, 107476. <https://doi.org/10.1016/j.infsof.2024.107476>

ANEXOS

ANEXO 1: Cuestionario para determinar la incorporación de actividades de seguridad en el desarrollo de software usando SCRUM

El presente cuestionario tiene el objetivo de recolectar información acerca de las actividades y/o prácticas de desarrollo seguro dentro de un proceso de desarrollo ágil (SCRUM), y está dirigido a los miembros de equipos de desarrollo de software con experiencia en el referido proceso. Con este cuestionario se espera obtener, bajo la experiencia del encuestado, datos más reales que permitan realizar el análisis respecto a las actividades de desarrollo seguro dentro de un proceso SCRUM y su impacto.

El cuestionario está dividido en 7 secciones: La Sección 1 muestra los datos del encuestador y el modelo conceptual del marco de trabajo propuesto. La sección 2 relacionada a la causa, frecuencia y consecuencias de ataques a software web. La Sección 3 relacionada a la experiencia, que tienen los miembros de los equipos de desarrollo de software usando SCRUM. La Sección 4 relacionada a la incorporación de un especialista en desarrollo de software seguro en un proceso de desarrollo ágil usando SCRUM. La Sección 5 relacionada al costo que de incorporar la actividad de seguridad “Modelado de Amenazas” en SCRUM. La Sección 6 relacionada al beneficio que se obtendría el incorporar la actividad de seguridad “Modelado de Amenazas” en SCRUM. La Sección 7 corresponde tiempo adicional que demanda incorporar actividades de seguridad en el desarrollo con SCRUM.

Las preguntas de la sección 2, 4, 5 y 6 deben contestarse de acuerdo con una calificación de 5 valores, para recabar información referida:

- Principal causa de ataque a un software web si durante su desarrollo no se incorporaron actividades de desarrollo de software seguro

5 = Totalmente de Acuerdo
4 = De Acuerdo
3 = Neutral
2 = En Desacuerdo

1 = Totalmente en Desacuerdo

- Frecuencia de ataque a un software web si durante su desarrollo no se incorporaron actividades de desarrollo de software seguro.

5 = Muy Frecuente

4 = Frecuente

3 = Neutral

2 = Poco Frecuente

1 = Nada Frecuente

- Nivel de impacto (técnico y del negocio), ¿que se da en la organización como consecuencia de un ataque sobre un software web?

5 = Nada Negativo

4 = Poco Negativo

3 = Neutral

2 = Negativo

1 = Totalmente Negativo

- A la incorporación de un especialista en desarrollo de software seguro en un proceso de desarrollo ágil usando SCRUM.

5 = Totalmente de Acuerdo

4 = De Acuerdo

3 = Neutral

2 = En Desacuerdo

1 = Totalmente en Desacuerdo

- al costo y beneficio de incorporar la actividad de seguridad “Modelado de Amenazas” para contrarrestar las vulnerabilidades más comunes.

5 = Nada Costoso

4 = Poco Costoso

3 = Neutral

2 = Costoso

1 = Muy Costoso

5 = Muy Beneficioso

4 = Beneficioso

3 = Neutral

2 = Poco Beneficioso

1 = Nada Beneficioso

Sección 2: Causas, frecuencia y consecuencias de ataques a un software web

En base a su experiencia en el desarrollo ágil de software con SCRUM responda:

1. ¿Considera usted que una de las principales causas de ataques a un software web es que durante el proceso de su desarrollo no se incorporaron ACTIVIDADES DE DESARROLLO SEGURO?

- ☐ Totalmente en Desacuerdo
- ☐ En Desacuerdo
- ☐ Neutral
- ☐ De Acuerdo
- ☐ Totalmente de Acuerdo

2. ¿Con qué frecuencia sería atacado un software web si durante el desarrollo no se incorporaron ACTIVIDADES DE DESARROLLO SEGURO?

- ☐ Nada Frecuente
- ☐ Poco Frecuente
- ☐ Neutral
- ☐ Frecuente
- ☐ Muy Frecuente

3. ¿Cuál es el nivel de impacto que se da en una organización luego de producirse un ataque a su sistema informático?

- ☐ Nada Negativo
- ☐ Poco Negativo
- ☐ Neutral
- ☐ Negativo
- ☐ Totalmente Negativo

Sección 3: Experiencia en Desarrollo de Software Usando SCRUM

Conteste la siguiente pregunta relacionado a la experiencia en el desarrollo de software usando SCRUM, ya sea liderando el equipo de desarrollo o desempeñando cualquier otro rol.

4. ¿Cuántos años de experiencia tiene en el desarrollo de software usando SCRUM?

- ☐ Más de 10 años
- ☐ Entre 7 y 10 años
- ☐ Entre 5 y 7 años
- ☐ Entre 3 y 5 años
- ☐ Menor a 3 años

Sección 4: Incorporar un Especialista en Desarrollo de software seguro usando SCRUM

Basado en su experiencia en el desarrollo ágil con SCRUM conteste la siguiente pregunta.

5. ¿Cuál es su posición respecto a la incorporación de un especialista en desarrollo seguro de software usando SCRUM?

- ☐ Totalmente de Acuerdo
- ☐ De Acuerdo
- ☐ Neutral
- ☐ En Desacuerdo
- ☐ Totalmente en Desacuerdo

Sección 5: Costo de Incorporar la Actividad de Seguridad "Modelado de Amenazas" en SCRUM

La actividad de seguridad "Modelado de Amenazas" permite identificar y clasificar sistemáticamente las amenazas que tienen más probabilidades de afectar a un sistema.

Incorporar la actividad de seguridad "Modelado de Amenazas" en el proceso de desarrollo con SCRUM demanda de un costo adicional (esfuerzo o grado de dificultad). En base a su experiencia en el desarrollo con SCRUM responda la siguiente pregunta.

6. ¿Qué tan costoso considera usted que significaría incorporar la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo de software usando SCRUM?

- ☐ Nada Costoso
- ☐ Poco Costoso
- ☐ Neutral
- ☐ Costoso
- ☐ Muy Costoso

Sección 5: Beneficio de Incorporar la Actividad de Seguridad "Modelado de Amenazas" en SCRUM

Incorporar la actividad de seguridad "Modelado de Amenazas" en el desarrollo de software con SCRUM para mitigar las vulnerabilidades genera un beneficio.

Al respecto en base a su experiencia en el desarrollo ágil de software con SCRUM responda la siguiente pregunta.

7. ¿Qué tan beneficioso considera usted que significaría incorporar la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo de software usando SCRUM?

- ☐ Muy Beneficioso
- ☐ Beneficioso
- ☐ Neutral
- ☐ Poco Beneficioso
- ☐ Nada Beneficioso

Sección 7: Tiempo Adicional que Demanda Incorporar Actividades de Seguridad en el Desarrollo con SCRUM

El incorporar actividades de seguridad en el desarrollo de software usando SCRUM para mitigar las amenazas, demanda de un tiempo adicional sobre el que ya se tiene comprometido para el desarrollo de la funcionalidad.

En ese sentido, tomando en cuenta que estas actividades se deben realizar durante todo el proceso de desarrollo con SCRUM y en base a su experiencia responda la siguiente pregunta.

8. **Educación y Entrenamiento en Políticas y Actividades de Desarrollo Seguro** para concientizar el equipo de desarrollo en prácticas de desarrollo seguro, así como socializar las políticas y requisitos de seguridad de alto nivel, en ese sentido responda la siguiente pregunta.

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "**Entrenamiento en Políticas y Actividades de Desarrollo Seguro**" durante el proceso de desarrollo con SCRUM?

- ☐ Más de 10 horas
- ☐ Entre 8 y 10 horas
- ☐ Entre 5 y 8 horas
- ☐ Entre 2 y 5 horas

- ☐ Menos de 2 horas

9. **Identificación de Roles**, Identificación de Roles, La identificación de roles es importante porque ayuda a garantizar que los usuarios solo tengan acceso a los datos y sistemas a los que necesitan acceder. Esto ayuda a proteger los datos de los usuarios de acceso no autorizado y también ayuda a proteger los sistemas de ataques posibles ataques.

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Identificación de Roles" durante el proceso de desarrollo con SCRUM?

- ☐ Más de 10 horas
☐ Entre 8 y 10 horas
☐ Entre 5 y 8 horas
☐ Entre 2 y 5 horas
☐ Menos de 2 horas

10. **Análisis de Riesgos de Seguridad (Identificación de Ataques)**, el objetivo de esta actividad de seguridad es identificar las amenazas, a las que podría estar expuesto el sistema informático, para ello se definen escenarios de amenazas expresadas como **Historias de Abuso**, luego estas se categorizan y son clasificadas en función a los riesgos.

Un ejemplo de Historia de Abuso puede ser: **“Yo como atacante deseo utilizar la denegación de servicio para hacer inaccesible al sistema”**, tomando en cuenta lo descrito responda a la siguiente pregunta.

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad **"Análisis de Riesgos de Seguridad (Identificación de Ataques)"** durante el proceso de desarrollo con SCRUM?

- ☐ Más de 10 horas
☐ Entre 8 y 10 horas
☐ Entre 5 y 8 horas
☐ Entre 2 y 5 horas
☐ Menos de 2 horas

11. **Análisis de Requisitos de Seguridad**, el objetivo de esta actividad identificar las contramedidas o requisitos de seguridad (controles, políticas) y determinar su mitigación para evitar que una amenaza se materialice, los requisitos de seguridad identificados se formulan y describen como **historias de usuario relacionadas a la seguridad**.

Un ejemplo de **historia de usuario relacionado a la seguridad** puede ser: “Yo como cliente deseo que mis datos sean protegidos para no ser divulgados a otros clientes o personas desconocidas”, tomando en cuenta lo descrito responda a la siguiente pregunta.

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "**Análisis de Requisitos de Seguridad**" durante el proceso de desarrollo con SCRUM?

- ☐ Más de 10 horas
- ☐ Entre 8 y 10 horas
- ☐ Entre 5 y 8 horas
- ☐ Entre 2 y 5 horas
- ☐ Menos de 2 horas

12. La actividad de seguridad "**Modelado de Amenazas**" tiene como objetivo identificar las vulnerabilidades del sistema que pueden ser explotadas por atacantes y desarrollar controles para mitigar estas vulnerabilidades.

El proceso de modelado de amenazas está compuesto por:

- Descomposición de la propuesta de solución en términos de seguridad (desarrollo de diagrama de flujo de datos-DFD).
- Análisis de riesgos de seguridad (identificación de ataques).
- Análisis de requisitos de seguridad (Mitigación)
- Validación de la Mitigación.

En ese sentido responda a la siguiente pregunta.

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "**Modelado de Amenazas**" durante el proceso de desarrollo con SCRUM?

- ☐ Más de 10 horas

- ☐ Entre 8 y 10 horas
- ☐ Entre 5 y 8 horas
- ☐ Entre 2 y 5 horas
- ☐ Menos de 2 horas

13. La actividad de seguridad "**Análisis estático de código**" se realiza para encontrar errores, incluidos errores de programación, errores de lógica y vulnerabilidades de seguridad.

Ejemplos de análisis estático de código pueden ser: Desbordamientos de búfer, inyección de código y ataques de tiempo de ejecución.

En ese sentido responda a la siguiente pregunta.

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "**Análisis estático de código**" durante el proceso de desarrollo con SCRUM?

- ☐ Más de 10 horas
- ☐ Entre 8 y 10 horas
- ☐ Entre 5 y 8 horas
- ☐ Entre 2 y 5 horas
- ☐ Menos de 2 horas

14. La actividad de seguridad "**Pruebas de Penetración**" son una forma de evaluar la seguridad de un sistema informático simulando un ataque de un hacker. Los pentesters intentan encontrar y explotar vulnerabilidades en el sistema para comprometerlo y obtener acceso no autorizado, se realiza antes de la puesta en producción del sistema.

Las pruebas de penetración pueden ayudar a identificar una amplia gama de vulnerabilidades, incluidas: vulnerabilidades de seguridad de red, vulnerabilidades de seguridad web, vulnerabilidades de seguridad de aplicaciones, vulnerabilidades de seguridad de bases de datos.

En ese sentido responda a la siguiente pregunta.

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "**Pruebas de Penetración**" durante el proceso de desarrollo con SCRUM?

- ☐ Más de 10 horas
- ☐ Entre 8 y 10 horas
- ☐ Entre 5 y 8 horas
- ☐ Entre 2 y 5 horas
- ☐ Menos de 2 horas

15. La actividad de seguridad "**Planificación de Respuesta a Incidentes**" consiste en definir un plan y procedimiento de respuesta ante incidentes de seguridad teniendo como base una política de gestión de incidentes de seguridad de los activos o sistemas informáticos.

En ese sentido responda a la siguiente pregunta.

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "**Planificación de Respuesta a Incidentes**" durante el proceso de desarrollo con SCRUM?

- ☐ Más de 10 horas
- ☐ Entre 8 y 10 horas
- ☐ Entre 5 y 8 horas
- ☐ Entre 2 y 5 horas
- ☐ Menos de 2 horas

ANEXO 2: Validación de cuestionario propuesto



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
Universidad del Perú. DECANA DE AMÉRICA

Lima, 26 de junio del 2023

Señor. -

Dra. Lenis Rossi WONG PORTILLO

Presente. -

Es grato dirigirme a usted para manifestarle mi saludo cordial. Dada su experiencia profesional y méritos académicos, le solicito su inapreciable colaboración como experta para la validación de contenido de los ítems que conforman los instrumentos de recolección de datos, que serán aplicados a una muestra seleccionada que tiene como finalidad recoger la información directa para el trabajo de investigación titulada: "Un Marco de Trabajo Para el Desarrollo de Software Web Seguro con Metodologías Ágiles", para obtener el grado académico de Magister en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software.

Para efectuar la validación del instrumento, usted deberá leer cuidadosamente cada enunciado y sus correspondientes alternativas de respuesta, en donde se pueden seleccionar una alternativa de acuerdo con el criterio personal y profesional que corresponda al instrumento.

Se le agradece cualquier sugerencia a la redacción, el contenido, la pertinencia y congruencia u otro aspecto que considere relevante para mejorar el mismo.

Atentamente,

Gilmer Glicerio VALDERRAMA HERRERA
DNI: 18203834



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
 Universidad del Perú. DECANA DE AMÉRICA

**MAESTRÍA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA CON
 MENCIÓN EN INGENIERÍA DE SOFTWARE**

TÍTULO DE TESIS: Un Marco de Trabajo Para el Desarrollo de Software Web Seguro con Metodologías Ágiles, 2023

AUTOR: Gilmer Glicerio VALDERRAMA HERRERA

ASESOR: Dr. José Alfredo HERRERA QUISPE

APELLIDOS Y NOMBRES DEL JUEZ:	WONG PORTILLO, LENIS ROSSI
DNI:	10438282
FORMACIÓN ACADÉMICA:	DOCTORA EN INGENIERÍA DE SISTEMAS E INFORMÁTICA
ÁREA DE EXPERIENCIA PROFESIONAL:	Profesora Asociada UNMSM
INSTITUCIÓN:	UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS

Firma del experto informante



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
Universidad del Perú. DECANA DE AMÉRICA

Lima, 7 de junio del 2023

Señor. -

Dr. Hugo Froilán VEGA HUERTA

Presente. -

Es grato dirigirme a usted para manifestarle mi saludo cordial. Dada su experiencia profesional y méritos académicos, le solicito su inapreciable colaboración como experta para la validación de contenido de los ítems que conforman los instrumentos de recolección de datos, que serán aplicados a una muestra seleccionada que tiene como finalidad recoger la información directa para el trabajo de investigación titulada: "Un Marco de Trabajo Para el Desarrollo de Software Web Seguro con Metodologías Ágiles", para obtener el grado académico de Magíster en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software.

Para efectuar la validación del instrumento, usted deberá leer cuidadosamente cada enunciado y sus correspondientes alternativas de respuesta, en donde se pueden seleccionar una alternativa de acuerdo con el criterio personal y profesional que corresponda al instrumento.

Se le agradece cualquier sugerencia a la redacción, el contenido, la pertinencia y congruencia u otro aspecto que considere relevante para mejorar el mismo.

Atentamente,

Gilmer Glicerio VALDERRAMA HERRERA
DNI: 18203834



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
Universidad del Perú. DECANA DE AMÉRICA

**MAESTRÍA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA CON
MENCIÓN EN INGENIERÍA DE SOFTWARE**

TÍTULO DE TESIS: Un Marco de Trabajo Para el Desarrollo de Software Web Seguro con Metodologías Ágiles, 2023

AUTOR: Gilmer Glicerio VALDERRAMA HERRERA

ASESOR: Dr. José Alfredo HERRERA QUISPE

APELLIDOS Y NOMBRES DEL JUEZ:	VEGA HUERTA, Hugo Froilán
DNI:	06147737
FORMACIÓN ACADÉMICA:	DOCTOR EN INGENIERÍA DE SISTEMAS
ÁREA DE EXPERIENCIA PROFESIONAL:	Profesor Principal UNMSM
INSTITUCIÓN:	UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS

Firma del experto informante



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
Universidad del Perú. DECANA DE AMÉRICA

060586790

Lima, 7 de junio del 2023

Señor. -

Lic. Gustavo ARREDONDO CASTILLO

Presente. -

Es grato dirigirme a usted para manifestarle mi saludo cordial. Dada su experiencia profesional y méritos académicos, le solicito su inapreciable colaboración como experta para la validación de contenido de los ítems que conforman los instrumentos de recolección de datos, que serán aplicados a una muestra seleccionada que tiene como finalidad recoger la información directa para el trabajo de investigación titulada: **"Un Marco de Trabajo Para el Desarrollo de Software Web Seguro con Metodologías Ágiles"**, para obtener el grado académico de Magíster en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software.

Para efectuar la validación del instrumento, usted deberá leer cuidadosamente cada enunciado y sus correspondientes alternativas de respuesta, en donde se pueden seleccionar una alternativa de acuerdo con el criterio personal y profesional que corresponda al instrumento.

Se le agradece cualquier sugerencia a la redacción, el contenido, la pertinencia y congruencia u otro aspecto que considere relevante para mejorar el mismo.

Atentamente,

Gilmer Glicerio VALDERRAMA HERRERA
DNI: 18203834



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
Universidad del Perú. DECANA DE AMÉRICA

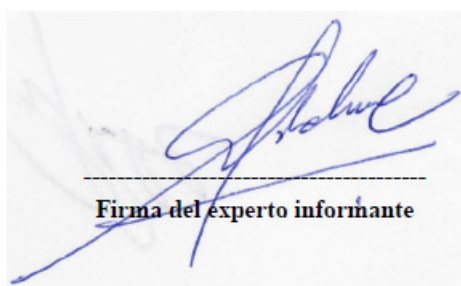
**MAESTRÍA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA CON
MENCIÓN EN INGENIERÍA DE SOFTWARE**

TÍTULO DE TESIS: Un Marco de Trabajo Para el Desarrollo de Software Web Seguro con Metodologías Ágiles, 2023

AUTOR: Gilmer Glicerio VALDERRAMA HERRERA

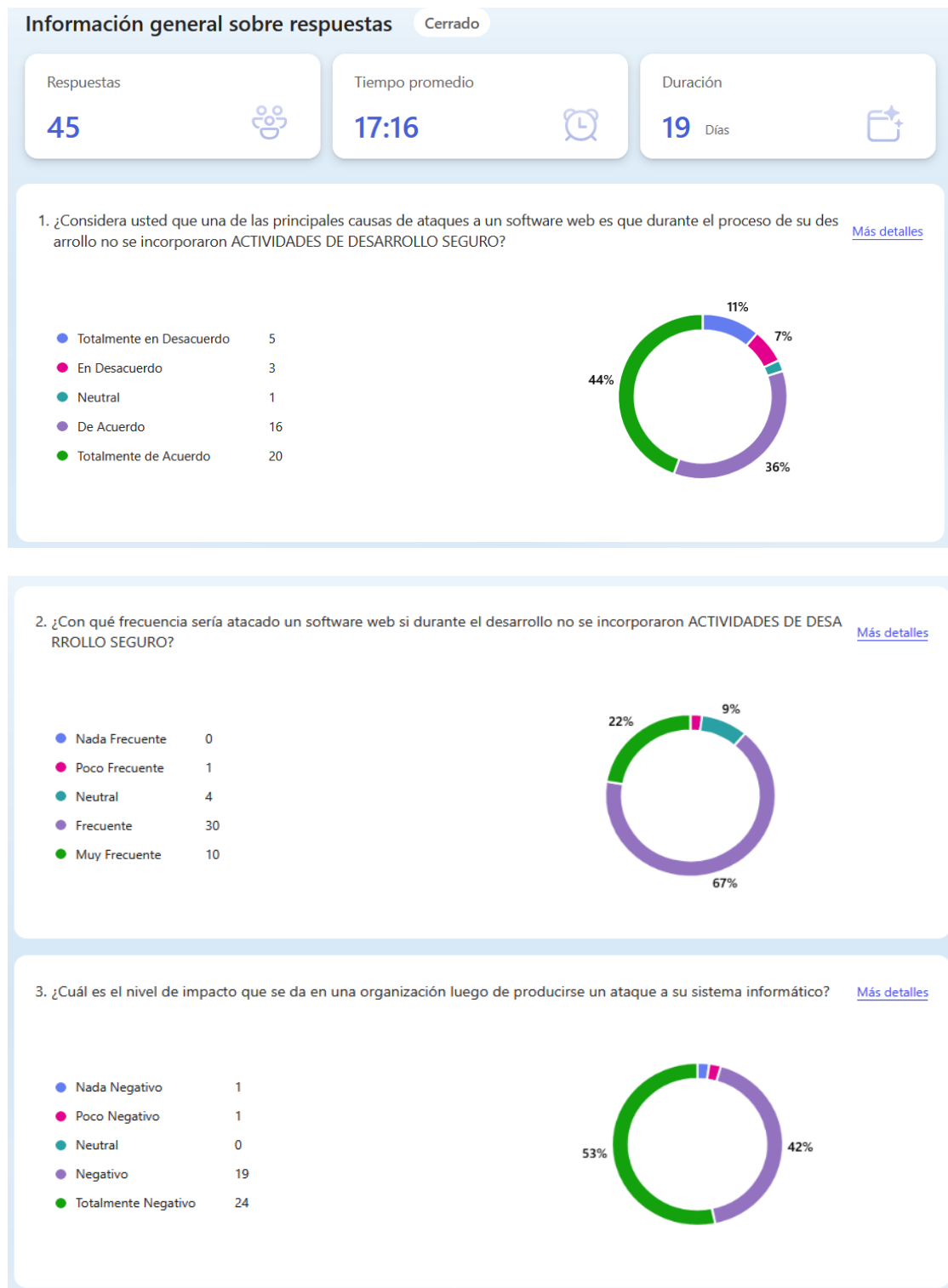
ASESOR: Dr. José Alfredo HERRERA QUISPE

APELLIDOS Y NOMBRES DEL JUEZ:	ARREDONDO CASTILLO, GUSTAVO
DNI:	06058679
FORMACIÓN ACADÉMICA:	LICENCIADO EN COMPUTACIÓN
ÁREA DE EXPERIENCIA PROFESIONAL:	Profesor Auxiliar UNMSM
INSTITUCIÓN:	UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS



Firma del experto informante

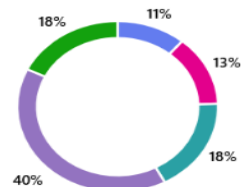
ANEXO 3: Resultado de cuestionario para validar el marco de trabajo propuesto



4. ¿Cuántos años de experiencia tiene en el desarrollo de software usando SCRUM?

[Más detalles](#)

Más de 10 años	5
Entre 7 y 10 años	6
Entre 5 y 7 años	8
Entre 3 y 5 años	18
Menor a 3 años	8



5. ¿Cuál es su posición respecto a la incorporación de un especialista en desarrollo seguro de software usando SCRUM?

[Más detalles](#)

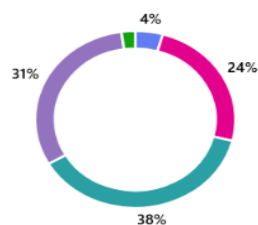
Totalmente de Acuerdo	27
De Acuerdo	17
Neutral	0
En Desacuerdo	0
Totalmente en Desacuerdo	1



6. ¿Qué tan costoso considera usted que significaría incorporar la actividad de seguridad "**Modelado de Amenazas**" durante el proceso de desarrollo de software usando SCRUM?

[Más detalles](#)

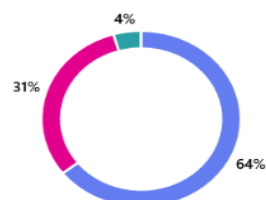
Nada Costoso	2
Poco Costoso	11
Neutral	17
Costoso	14
Muy Costoso	1



7. ¿Qué tan beneficioso considera usted que significaría incorporar la actividad de seguridad "**Modelado de Amenazas**" durante el proceso de desarrollo de software usando SCRUM?

[Más detalles](#)

Muy Beneficioso	29
Beneficioso	14
Neutral	2
Poco Beneficioso	0
Nada Beneficioso	0

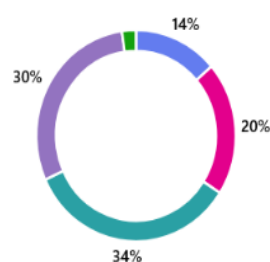


8. **Entrenamiento en Políticas y Actividades de Desarrollo Seguro** para concientizar el equipo de desarrollo en prácticas de desarrollo seguro, así como socializar las políticas y requisitos de seguridad de alto nivel, en ese sentido responda la siguiente pregunta.

[Más detalles](#)

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "**Entrenamiento en Políticas y Actividades de Desarrollo Seguro**" durante el proceso de desarrollo con SCRUM?

● Más de 10 horas	6
● Entre 8 y 10 horas	9
● Entre 5 y 8 horas	15
● Entre 2 y 5 horas	13
● Menos de 2 horas	1

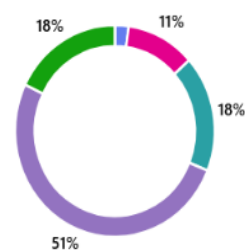


9. **Identificación de Roles.** La identificación de roles es importante porque ayuda a garantizar que los usuarios solo tengan acceso a los datos y sistemas a los que necesitan acceder. Esto ayuda a proteger los datos de los usuarios de accesos no autorizados y también ayuda a proteger los sistemas de ataques posibles.

[Más detalles](#)

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "**Identificación de Roles**" durante el proceso de desarrollo con SCRUM?

● Más de 10 horas	1
● Entre 8 y 10 horas	5
● Entre 5 y 8 horas	8
● Entre 2 y 5 horas	23
● Menos de 2 horas	8

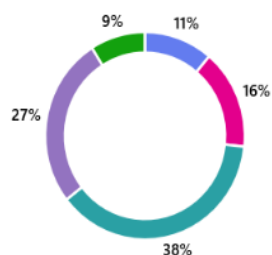


10. **Análisis de Riesgos de Seguridad (Identificación de Ataques)**, el objetivo de esta actividad de seguridad es identificar las amenazas, a las que podría estar expuesto el sistema informático, para ello se definen escenarios de amenaza expresadas como **Historias de Abuso**, luego estas se categorizan y son clasificadas en función a los riesgos.

Un ejemplo de Historia de Abuso puede ser: *"Yo como atacante deseo utilizar la denegación de servicio para hacer inaccesible al sistema"*, tomando en cuenta lo descrito responda a la siguiente pregunta. [Más detalles](#)

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad **"Análisis de Riesgos de Seguridad (Identificación de Ataques)"** durante el proceso de desarrollo con SCRUM?

Más de 10 horas	5
Entre 8 y 10 horas	7
Entre 5 y 8 horas	17
Entre 2 y 5 horas	12
Menos de 2 horas	4

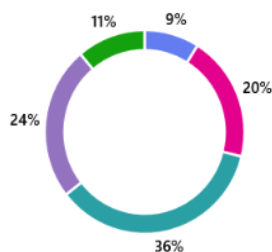


11. **Análisis de Requisitos de Seguridad**, el objetivo de esta actividad identificar las contramedidas o requisitos de seguridad (controles, políticas) y determinar su mitigación para evitar que una amenaza se materialice, los requisitos de seguridad identificados se formulan y describen como **historias de usuario relacionadas a la seguridad**.

Un ejemplo de **historia de usuario relacionado a la seguridad**, puede ser: *"Yo como cliente deseo que mis datos sean protegidos para no ser divulgados a otros clientes o personas desconocidas"*, tomando en cuenta lo descrito responda a la siguiente pregunta. [Más detalles](#)

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad **"Análisis de Requisitos de Seguridad"** durante el proceso de desarrollo con SCRUM?

Más de 10 horas	4
Entre 8 y 10 horas	9
Entre 5 y 8 horas	16
Entre 2 y 5 horas	11
Menos de 2 horas	5



12. La actividad de seguridad "**Modelado de Amenazas**" tiene como objetivo identificar las vulnerabilidades del sistema que pueden ser explotadas por atacantes y desarrollar controles para mitigar estas vulnerabilidades.

El proceso de modelado de amenazas está compuesto por:

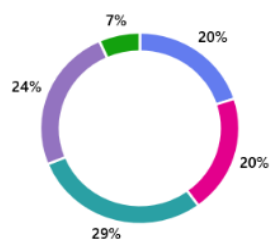
- Descomposición de la propuesta de solución en términos de seguridad (desarrollo de diagrama de flujo de datos-DFD).
- Análisis de riesgos de seguridad (identificación de ataques).
- Análisis de requisitos de seguridad (Mitigación)
- Validación de la Mitigación.

[Más detalles](#)

En ese sentido responda a la siguiente pregunta.

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "**Modelado de Amenazas**" durante el proceso de desarrollo con SCRUM?

● Más de 10 horas	9
● Entre 8 y 10 horas	9
● Entre 5 y 8 horas	13
● Entre 2 y 5 horas	11
● Menos de 2 horas	3



13. La actividad de seguridad "**Análisis de Código Estático**" se realiza para encontrar errores, incluidos errores de programación, errores de lógica y vulnerabilidades de seguridad..

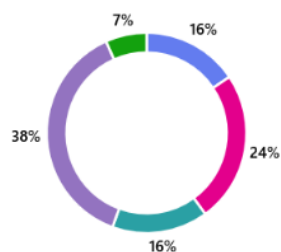
Ejemplos de análisis de código estático pueden ser: Desbordamientos de búfer, inyección de código y ataques de tiempo de ejecución.

[Más detalles](#)

En ese sentido responda a la siguiente pregunta.

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "**Análisis de Código Estático**" durante el proceso de desarrollo con SCRUM?

● Más de 10 horas	7
● Entre 8 y 10 horas	11
● Entre 5 y 8 horas	7
● Entre 2 y 5 horas	17
● Menos de 2 horas	3



14. La actividad de seguridad "**Pruebas de Penetración**" son una forma de evaluar la seguridad de un sistema informático simulando un ataque de un hacker. Los pentesters intentan encontrar y explotar vulnerabilidades en el sistema para comprometerlo y obtener acceso no autorizado, se realiza antes de la puesta en producción del sistema.

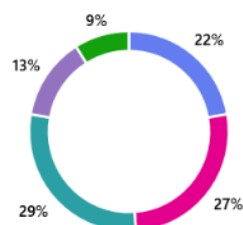
Las pruebas de penetración pueden ayudar a identificar una amplia gama de vulnerabilidades, incluidas: vulnerabilidades de seguridad de red, vulnerabilidades de seguridad web, vulnerabilidades de seguridad de aplicaciones, vulnerabilidades de seguridad de bases de datos.

[Más detalles](#)

En ese sentido responda a la siguiente pregunta.

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "**Pruebas de Penetración**" durante el proceso de desarrollo con SCRUM?

● Más de 10 horas	10
● Entre 8 y 10 horas	12
● Entre 5 y 8 horas	13
● Entre 2 y 5 horas	6
● Menos de 2 horas	4



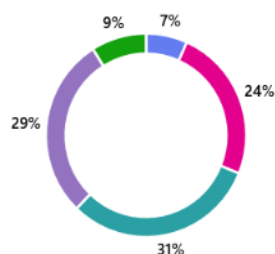
15. La actividad de seguridad "**Planificación de Respuesta a Incidentes**" consiste en definir un plan y procedimiento de respuesta ante incidentes de seguridad teniendo como base una política de gestión de incidentes de seguridad de los activos o sistemas informáticos.

En ese sentido responda a la siguiente pregunta.

[Más detalles](#)

¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "**Planificación de Respuesta a Incidentes**" durante el proceso de desarrollo con SCRUM?

● Más de 10 horas	3
● Entre 8 y 10 horas	11
● Entre 5 y 8 horas	14
● Entre 2 y 5 horas	13
● Menos de 2 horas	4



ID	¿Considera usted que una de las principales causas de ataques a un software web es que durante el proceso de su desarrollo no se incorporaron ACTIVIDADES DE DESARROLLO SEGURO?	¿Con qué frecuencia sería atacado un software web si durante el desarrollo no se incorporaron ACTIVIDADES DE DESARROLLO SEGURO?	¿Cuál es el nivel de impacto que se da en una organización luego de producirse un ataque a su sistema informático?	¿Cuántos años de experiencia tiene en el desarrollo de software usando SCRUM?	¿Cuál es su posición respecto a la incorporación de un especialista en desarrollo seguro de software usando SCRUM?	¿Qué tan costoso considera usted que significaría incorporar la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo de software usando SCRUM?	¿Qué tan beneficioso considera usted que significaría incorporar la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo de software usando SCRUM?
1	Totalmente en Desacuerdo	Frecuente	Totalmente Negativo	Entre 7 y 10 años	De Acuerdo	Muy Costoso	Beneficioso
2	De Acuerdo	Frecuente	Totalmente Negativo	Entre 3 y 5 años	Totalmente de Acuerdo	Costoso	Muy Beneficioso
3	Totalmente de Acuerdo	Frecuente	Negativo	Entre 3 y 5 años	Totalmente de Acuerdo	Costoso	Muy Beneficioso
4	Totalmente de Acuerdo	Muy Frecuente	Negativo	Entre 5 y 7 años	Totalmente de Acuerdo	Neutral	Muy Beneficioso
5	Totalmente de Acuerdo	Muy Frecuente	Totalmente Negativo	Menor a 3 años	Totalmente de Acuerdo	Nada Costoso	Muy Beneficioso
6	De Acuerdo	Frecuente	Negativo	Entre 3 y 5 años	De Acuerdo	Poco Costoso	Muy Beneficioso
7	De Acuerdo	Frecuente	Totalmente Negativo	Entre 3 y 5 años	Totalmente de Acuerdo	Neutral	Muy Beneficioso
8	De Acuerdo	Frecuente	Negativo	Entre 3 y 5 años	De Acuerdo	Neutral	Beneficioso
9	De Acuerdo	Frecuente	Negativo	Menor a 3 años	Totalmente de Acuerdo	Neutral	Muy Beneficioso
10	Totalmente de Acuerdo	Muy Frecuente	Totalmente Negativo	Entre 5 y 7 años	Totalmente de Acuerdo	Neutral	Muy Beneficioso
11	Totalmente de Acuerdo	Frecuente	Negativo	Entre 3 y 5 años	De Acuerdo	Poco Costoso	Beneficioso
12	De Acuerdo	Neutral	Totalmente Negativo	Menor a 3 años	Totalmente de Acuerdo	Neutral	Muy Beneficioso
13	Totalmente en Desacuerdo	Frecuente	Totalmente Negativo	Entre 5 y 7 años	Totalmente de Acuerdo	Neutral	Muy Beneficioso

ID	¿Considera usted que una de las principales causas de ataques a un software web es que durante el proceso de su desarrollo no se incorporaron ACTIVIDADES DE DESARROLLO SEGURO?	¿Con qué frecuencia sería atacado un software web si durante el desarrollo no se incorporaron ACTIVIDADES DE DESARROLLO SEGURO?	¿Cuál es el nivel de impacto que se da en una organización luego de producirse un ataque a su sistema informático?	¿Cuántos años de experiencia tiene en el desarrollo de software usando SCRUM?	¿Cuál es su posición respecto a la incorporación de un especialista en desarrollo seguro de software usando SCRUM?	¿Qué tan costoso considera usted que significaría incorporar la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo de software usando SCRUM?	¿Qué tan beneficioso considera usted que significaría incorporar la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo de software usando SCRUM?
14	De Acuerdo	Frecuente	Totalmente Negativo	Entre 5 y 7 años	Totalmente de Acuerdo	Costoso	Muy Beneficioso
15	De Acuerdo	Frecuente	Negativo	Entre 3 y 5 años	Totalmente en Desacuerdo	Neutral	Beneficioso
16	En Desacuerdo	Frecuente	Negativo	Entre 3 y 5 años	De Acuerdo	Costoso	Neutral
17	De Acuerdo	Frecuente	Poco Negativo	Entre 5 y 7 años	Totalmente de Acuerdo	Poco Costoso	Muy Beneficioso
18	En Desacuerdo	Frecuente	Negativo	Entre 5 y 7 años	De Acuerdo	Costoso	Beneficioso
19	Totalmente de Acuerdo	Muy Frecuente	Totalmente Negativo	Entre 3 y 5 años	Totalmente de Acuerdo	Poco Costoso	Muy Beneficioso
20	Totalmente de Acuerdo	Frecuente	Negativo	Menor a 3 años	De Acuerdo	Costoso	Beneficioso
21	Totalmente de Acuerdo	Frecuente	Negativo	Menor a 3 años	De Acuerdo	Neutral	Beneficioso
22	Totalmente de Acuerdo	Frecuente	Totalmente Negativo	Menor a 3 años	De Acuerdo	Poco Costoso	Muy Beneficioso
23	Totalmente en Desacuerdo	Neutral	Totalmente Negativo	Menor a 3 años	De Acuerdo	Neutral	Beneficioso
24	Totalmente de Acuerdo	Neutral	Totalmente Negativo	Entre 5 y 7 años	Totalmente de Acuerdo	Nada Costoso	Muy Beneficioso
25	Totalmente de Acuerdo	Neutral	Totalmente Negativo	Más de 10 años	Totalmente de Acuerdo	Poco Costoso	Muy Beneficioso
26	Totalmente de Acuerdo	Frecuente	Totalmente Negativo	Menor a 3 años	Totalmente de Acuerdo	Costoso	Muy Beneficioso
27	De Acuerdo	Muy Frecuente	Totalmente Negativo	Entre 3 y 5 años	Totalmente de Acuerdo	Poco Costoso	Muy Beneficioso
28	Totalmente de Acuerdo	Muy Frecuente	Totalmente Negativo	Entre 7 y 10 años	Totalmente de Acuerdo	Costoso	Muy Beneficioso
29	De Acuerdo	Frecuente	Negativo	Entre 7 y 10 años	Totalmente de Acuerdo	Poco Costoso	Beneficioso

ID	¿Considera usted que una de las principales causas de ataques a un software web es que durante el proceso de su desarrollo no se incorporaron ACTIVIDADES DE DESARROLLO SEGURO?	¿Con qué frecuencia sería atacado un software web si durante el desarrollo no se incorporaron ACTIVIDADES DE DESARROLLO SEGURO?	¿Cuál es el nivel de impacto que se da en una organización luego de producirse un ataque a su sistema informático?	¿Cuántos años de experiencia tiene en el desarrollo de software usando SCRUM?	¿Cuál es su posición respecto a la incorporación de un especialista en desarrollo seguro de software usando SCRUM?	¿Qué tan costoso considera usted que significaría incorporar la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo de software usando SCRUM?	¿Qué tan beneficioso considera usted que significaría incorporar la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo de software usando SCRUM?
30	Totalmente de Acuerdo	Frecuente	Totalmente Negativo	Entre 3 y 5 años	De Acuerdo	Neutral	Muy Beneficioso
31	Neutral	Frecuente	Negativo	Entre 7 y 10 años	Totalmente de Acuerdo	Neutral	Muy Beneficioso
32	Totalmente de Acuerdo	Muy Frecuente	Totalmente Negativo	Entre 3 y 5 años	Totalmente de Acuerdo	Neutral	Neutral
33	De Acuerdo	Frecuente	Totalmente Negativo	Más de 10 años	De Acuerdo	Costoso	Muy Beneficioso
34	Totalmente de Acuerdo	Frecuente	Negativo	Entre 3 y 5 años	De Acuerdo	Costoso	Muy Beneficioso
35	De Acuerdo	Frecuente	Totalmente Negativo	Entre 3 y 5 años	Totalmente de Acuerdo	Neutral	Muy Beneficioso
36	Totalmente de Acuerdo	Frecuente	Totalmente Negativo	Más de 10 años	Totalmente de Acuerdo	Neutral	Muy Beneficioso
37	Totalmente de Acuerdo	Frecuente	Negativo	Entre 7 y 10 años	Totalmente de Acuerdo	Costoso	Muy Beneficioso
38	Totalmente en Desacuerdo	Frecuente	Negativo	Más de 10 años	Totalmente de Acuerdo	Poco Costoso	Beneficioso
39	De Acuerdo	Poco Frecuente	Nada Negativo	Más de 10 años	De Acuerdo	Poco Costoso	Beneficioso
40	Totalmente de Acuerdo	Frecuente	Totalmente Negativo	Entre 3 y 5 años	De Acuerdo	Costoso	Beneficioso
41	En Desacuerdo	Muy Frecuente	Negativo	Entre 5 y 7 años	Totalmente de Acuerdo	Poco Costoso	Beneficioso
42	Totalmente en Desacuerdo	Muy Frecuente	Negativo	Entre 7 y 10 años	Totalmente de Acuerdo	Costoso	Muy Beneficioso
43	De Acuerdo	Frecuente	Negativo	Entre 3 y 5 años	De Acuerdo	Neutral	Beneficioso
44	De Acuerdo	Muy Frecuente	Totalmente Negativo	Entre 3 y 5 años	De Acuerdo	Costoso	Muy Beneficioso
45	Totalmente de Acuerdo	Frecuente	Totalmente Negativo	Entre 3 y 5 años	Totalmente de Acuerdo	Neutral	Muy Beneficioso

ID	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Entrenamiento en Políticas y Actividades de Desarrollo Seguro" durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Identificación de Roles" durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Análisis de Riesgos de Seguridad (Identificación de Ataques)" durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Análisis de Requisitos de Seguridad durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Análisis estático de código" durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Pruebas de Penetración" durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Planificación de Respuesta a Incidentes" durante el proceso de desarrollo con SCRUM?
1	Más de 10 horas	Entre 8 y 10 horas	Entre 8 y 10 horas	Entre 5 y 8 horas	Entre 8 y 10 horas	Entre 8 y 10 horas	Entre 8 y 10 horas	Entre 8 y 10 horas
2	Entre 8 y 10 horas	Entre 5 y 8 horas	Más de 10 horas	Entre 8 y 10 horas	Más de 10 horas	Entre 8 y 10 horas	Más de 10 horas	Entre 8 y 10 horas
3	Entre 2 y 5 horas	Menos de 2 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Más de 10 horas	Más de 10 horas	Entre 8 y 10 horas	Más de 10 horas
4	Entre 8 y 10 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas
5	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 2 y 5 horas
6	Más de 10 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 8 y 10 horas	Entre 8 y 10 horas	Más de 10 horas	Entre 5 y 8 horas
7	Entre 8 y 10 horas	Entre 2 y 5 horas	Entre 8 y 10 horas	Entre 5 y 8 horas	Entre 8 y 10 horas	Entre 8 y 10 horas	Entre 8 y 10 horas	Entre 5 y 8 horas
8	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 2 y 5 horas
9	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 8 y 10 horas	Entre 8 y 10 horas	Entre 5 y 8 horas	Entre 8 y 10 horas	Entre 5 y 8 horas	Entre 5 y 8 horas
10		Menos de 2 horas	Menos de 2 horas	Menos de 2 horas	Menos de 2 horas	Menos de 2 horas	Menos de 2 horas	Menos de 2 horas
11	Entre 8 y 10 horas	Menos de 2 horas	Menos de 2 horas	Menos de 2 horas	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 8 y 10 horas
12	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 5 y 8 horas
13	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 5 y 8 horas
14	Entre 5 y 8 horas	Menos de 2 horas	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Menos de 2 horas	Entre 5 y 8 horas
15	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 8 y 10 horas	Más de 10 horas	Más de 10 horas	Más de 10 horas	Entre 8 y 10 horas
16	Entre 5 y 8 horas	Entre 8 y 10 horas	Más de 10 horas	Más de 10 horas	Más de 10 horas	Entre 8 y 10 horas	Más de 10 horas	Entre 8 y 10 horas
17	Más de 10 horas	Más de 10 horas	Más de 10 horas	Entre 8 y 10 horas	Entre 8 y 10 horas	Más de 10 horas	Entre 8 y 10 horas	Entre 8 y 10 horas
18	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 2 y 5 horas

ID	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Entrenamiento en Políticas y Actividades de Desarrollo Seguro" durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Identificación de Roles" durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Análisis de Riesgos de Seguridad (Identificación de Ataques)" durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Análisis de Requisitos de Seguridad durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Modelado de Amenazas" durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Análisis estático de código" durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Pruebas de Penetración" durante el proceso de desarrollo con SCRUM?	¿Qué tiempo adicional considera usted que demandaría incorporar la actividad de seguridad "Planificación de Respuesta a Incidentes" durante el proceso de desarrollo con SCRUM?
19	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 8 y 10 horas	Entre 5 y 8 horas	Entre 8 y 10 horas	Entre 5 y 8 horas	Entre 8 y 10 horas	Entre 5 y 8 horas
20	Menos de 2 horas	Menos de 2 horas	Menos de 2 horas	Menos de 2 horas	Menos de 2 horas	Menos de 2 horas	Menos de 2 horas	Menos de 2 horas
21	Entre 8 y 10 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 8 y 10 horas	Más de 10 horas	Entre 8 y 10 horas	Entre 8 y 10 horas	Entre 5 y 8 horas
22	Entre 5 y 8 horas	Menos de 2 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 8 y 10 horas	Entre 2 y 5 horas
23	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Más de 10 horas	Más de 10 horas
24	Entre 8 y 10 horas	Menos de 2 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Más de 10 horas	Entre 8 y 10 horas	Más de 10 horas	Entre 5 y 8 horas
25	Entre 8 y 10 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Más de 10 horas	Entre 8 y 10 horas	Entre 8 y 10 horas	Entre 2 y 5 horas	Menos de 2 horas
26	Entre 5 y 8 horas	Entre 2 y 5 horas	Menos de 2 horas	Menos de 2 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Menos de 2 horas
27	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Más de 10 horas	Entre 5 y 8 horas	Entre 5 y 8 horas
28	Entre 5 y 8 horas	Entre 2 y 5 horas	Más de 10 horas	Más de 10 horas	Más de 10 horas	Más de 10 horas	Más de 10 horas	Más de 10 horas
29	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 2 y 5 horas
30	Entre 8 y 10 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 8 y 10 horas	Entre 5 y 8 horas	Entre 2 y 5 horas
31	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 8 y 10 horas	Entre 2 y 5 horas
32	Más de 10 horas	Entre 5 y 8 horas	Más de 10 horas	Más de 10 horas	Más de 10 horas	Más de 10 horas	Más de 10 horas	Entre 8 y 10 horas
33	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 8 y 10 horas	Entre 5 y 8 horas	Entre 8 y 10 horas	Entre 8 y 10 horas	Más de 10 horas	Entre 8 y 10 horas
34	Entre 8 y 10 horas	Entre 8 y 10 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 8 y 10 horas	Entre 5 y 8 horas
35	Entre 2 y 5 horas	Entre 8 y 10 horas	Entre 5 y 8 horas	Entre 8 y 10 horas	Más de 10 horas	Más de 10 horas	Entre 8 y 10 horas	Entre 8 y 10 horas
36	Entre 2 y 5 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 5 y 8 horas	Entre 2 y 5 horas	Entre 5 y 8 horas	Entre 5 y 8 horas

ANEXO 4: Plantilla Excel para evaluar riesgo de seguridad

Riesgo Ejemplo: Robo total de bases de datos del centro de datos

Factores de Probabilidad							
Factores de Agente de Amenaza				Factores de Vulnerabilidad			
Nivel de Habilidad	Motivación	Oportunidad	Tamaño		Facilidad de Descubrimiento	Facilidad de Explotar	Detección de Intruso
1 - Sin habilidades técnicas	9 - Recompensa/interés alto	especial o recursos necesarios	Desarrolladores, administradores de sistemas		3 - Difícil	3 - Difícil	9 - De Conocimiento Público
							1 - Detección activa en la aplicación
Probabilidad General:				4.000	MEDIO		

Factores de Impacto							
Impacto Técnico				Impacto en el Negocio			
Pérdida de Confidencialidad	Pérdida de Integridad	Pérdida de Disponibilidad	Pérdida de Responsabilidad		Daños Económicos Financieros	Daño en la Reputación	Incumplimiento
2 - Revelación mínima no confidenciales	1 - Datos mínimos ligeramente dañados o corruptos	1 - Servicios secundarios mínimos interrumpidos	7 - Posiblemente rastreable		7 - Efecto significativo en la utilidad anual	1 - Daño mínimo	5 - Infracción notoria
							3 - Un individuo
Impacto General Técnico:				2.750	BAJO		
Impacto General:				3.375	MEDIO		

Gravedad General del Riesgo = Probabilidad x Impacto				
Impacto	ALTO	Medio	Alto	Crítico
	MEDIO	Bajo	Medio	Alto
	BAJO	Muy Bajo	Bajo	Medio
		BAJO	MEDIO	ALTO
Probabilidad				

Niveles de Probabilidad e Impacto	
0 to <3	BAJO
3 to <6	MEDIO
6 to 9	ALTO

ANEXO 5: Implementación del Marco de Trabajo Propuesto en el Desarrollo de la Adecuación al Marco Legal del Sistema Catastral Rural en el Ministerio de Desarrollo Agrario y Riego.

1. Implementación de Educación y sensibilización en materia de seguridad (Políticas de Desarrollo Seguro de Software en el Ministerio de Desarrollo Agrario y Riego-SCR)

El Ministerio de Desarrollo Agrario y Riego (MIDAGRI) establece, mediante el documento “**SGSI-PO-021 Política de desarrollo seguro MIDAGRI V.01**”, un marco obligatorio para proteger la información en todo el ciclo de vida de sus sistemas y servicios digitales. La política aplica a cualquier software nuevo o en mantenimiento (desarrollado internamente o por terceros), y a todo el personal que interviene en esos proyectos. Sus lineamientos se agrupan en cinco ejes:

- a) Ambientes segregados y control de cambios:** Se exige disponer de entornos separados (desarrollo, pruebas y producción) con configuraciones de seguridad equivalentes, control de versiones, copias de respaldo y procedimientos estandarizados para las liberaciones. Queda prohibido mover datos sensibles a entornos de prueba.
- b) Codificación segura y validación temprana:** Los equipos deben aplicar prácticas de codificación defensiva, análisis SAST/DAST en el pipeline CI/CD, revisiones por pares y formación continua en amenazas actuales (p. ej., OWASP Top 10). Se enfatiza la validación de entradas, manejo adecuado de errores y uso de autenticación multifactor.

- c) **Gestión de dependencias y criptografía robusta:** Toda biblioteca de terceros se escanea con herramientas como Snyk o Dependabot; los datos sensibles se cifran en tránsito (TLS) y reposo (AES-256), y las claves se gestionan con procedimientos seguros.
- d) **Procesos de autenticación, autorización y sesiones:** Se adoptan principios de mínimo privilegio, bloqueo tras intentos fallidos, protección contra CSRF y administración estricta del ciclo de vida de sesiones con cookies seguras y rotación de tokens.
- e) **Monitoreo, respuesta a incidentes y trazabilidad:** La política ordena integrar SIEM para detección en tiempo real, registrar operaciones críticas, practicar ejercicios de respuesta a incidentes y emplear modelado de amenazas desde la fase de diseño.

En conjunto, estas directrices convierten la seguridad en un requisito transversal: cada cambio de código, entorno o proveedor debe demostrar evidencia objetiva de controles implementados antes de pasar a producción, puede acceder al documento completo desde el siguiente enlace web:

https://drive.google.com/drive/u/0/folders/1xoszX6NMg3x_WDt7wZUSYpo22mNJRvkT

	Sistema de Gestión de Seguridad de la Información	Código:	SGSI-PO-021
		Versión:	1.0
	Política de Desarrollo Seguro	Clasificación:	Uso Interno



MINISTERIO DE DESARROLLO AGRARIO Y RIEGO
MIDAGRI

POLÍTICAS DE DESARROLLO SEGURO

V 1.0

	Sistema de Gestión de Seguridad de la Información	Código:	SGSI-PO-021
		Versión:	1.0
	Política de Desarrollo Seguro	Clasificación:	Uso Interno

1. OBJETIVO

Establecer la seguridad de la información en procesos de desarrollo de productos de software durante la prestación de los servicios ofrecidos por el MIDAGRI.

2. ALCANCE

La presente política se aplica a todos los sistemas informáticos desarrollados y/o actualizados en el MIDAGRI, ya sea en forma interna como por empresas o consultores externos contratados para tales efectos. Es aplicable a todos los funcionarios (planta, contratado, reemplazos y suplencia), personal a honorarios y terceros (proveedores, compra de servicios, etc.), que presten servicios al MIDAGRI.

3. POLITICA DE DESARROLLO SEGURO

Ambientes seguros

- Contar con ambientes de desarrollo, pruebas/calidad y producciones independientes, los ambientes de pruebas y producción deben ser homologados y configurados con los mismos controles de seguridad, a efectos de mitigar vulnerabilidades en el desarrollo.
- Probar los cambios de los sistemas en un entorno de prueba o de calidad antes de aplicarlos a los sistemas de producción en el marco de la MCVS vigente.
- Asignar nombres de dominio a los ambientes de desarrollo, pruebas y producción diferentes y usando una nomenclatura adecuada para reducir el riesgo de error.
- No copiar información sensible o confidencial a los ambientes de desarrollo y de prueba.
- Contar con seguimiento de cambios (de fuentes de software producido), en los ambientes de desarrollo, pruebas y producción.
- Realizar copias de seguridad de los ambientes de acuerdo a las políticas de copias de respaldo definidos por la OGTI.
- La información generada durante el desarrollo debe ser almacenada en repositorios seguros, es decir, con control de accesos y copias de respaldo.
- Asegurar que el cambio de versión de programas implementados en el ambiente de producción se hace de acuerdo con los procedimientos estandarizados de control de cambios.
- Examinar exhaustivamente los componentes de software antes de reutilizarlos, en particular los componentes de autenticación y cifrado. Solicitar su aprobación en caso de reutilizarlo.
- Manejar versionamiento de los proyectos de desarrollo y/o mantenimiento de software de acuerdo al procedimiento definido en la MCVS vigente.

Codificación segura

- Los equipos de desarrollo y pruebas deben tener conocimiento y experiencia en técnicas y/o actividades de desarrollo seguro de acuerdo al lenguaje y framework de programación en el que se desarrolle el producto. En función al avance de la tecnología de programación el personal debe ser capacitado en estos temas.

	Sistema de Gestión de Seguridad de la Información	Código:	SGSI-PO-021
		Versión:	1.0
	Política de Desarrollo Seguro	Clasificación:	Uso Interno

- Documentar las prácticas de codificación comunes e históricas y defectos que conducen a vulnerabilidades de software producido.
- Utilizar mensajes de error genéricos que no den pistas a los usuarios finales sobre ningún aspecto sensible del software producido.
- El software producido debe manejar los errores sin depender de los mensajes de error del servidor que se muestran a los usuarios.
- Cualquier lógica de control de acceso que conduzca a un error debe denegar el acceso por defecto.
- Utilizar mensajes de error genéricos que no den pistas a los usuarios finales sobre ningún aspecto sensible de la aplicación.
- La aplicación debe manejar los errores sin depender de los mensajes de error del servidor que se muestran a los usuarios.
- Considerar la seguridad de la información en la metodología en el ciclo de vida del desarrollo de software.
- Manejar las excepciones o errores que pueden ocurrir durante la ejecución de un bloque de código mediante técnicas apropiadas como el try/catch/finally, para prevenir la terminación abrupta del programa, mejorar de la experiencia del usuario mediante mensajes de errores claros y concisos, continuidad de la ejecución del programa, y la realización de cierre de todos los recursos.
- Registrar en un log todas las excepciones inesperadas en un registro específico donde indicar, entre otras, la fecha/hora del fallo, el usuario que lo causó y el método donde se produjo el fallo, así como información de la excepción. Este registro debería encontrarse en un entorno seguro sólo accesible a usuarios autorizados.
- Validar y sanear todas las entradas de usuario para evitar ataques de inyección (como SQL Injection) y otras formas de manipulación de datos. Esto incluye la validación de tipos de datos, longitud, formatos, rangos, uso de listas blancas para aceptar solo valores determinados y sanitización de entradas para eliminar caracteres potencialmente peligrosos, por ejemplo, utilizar OWASP ESAPI para validar entradas de usuario en formularios.
- Implementar mecanismos robustos para autenticar y autorizar usuarios, garantizando que solo usuarios autorizados accedan a recursos específicos. Esto incluye asignar los permisos y privilegios a roles de aplicación, nunca directamente a los usuarios, autenticación multifactor (MFA), uso de OAuth 2.0 y OpenID Connect, políticas de contraseñas fuertes y gestión segura de tokens, por ejemplo, utilizar MFA para acceder a servicios sensibles.
- Integrar herramientas automatizadas de análisis de seguridad en el ciclo de desarrollo para identificar vulnerabilidades de manera temprana. Esto incluye análisis estático de código (SAST) para detectar vulnerabilidades en el código fuente, análisis dinámico de aplicaciones (DAST) para identificar vulnerabilidades en aplicaciones en ejecución, por ejemplo, usar y/o integrar herramientas como SonarQube (SAST) y OWASP ZAP (DAST) en sus pipelines de CI/CD, asegurando que cada versión del software sea escaneada en busca de vulnerabilidades.
- Desarrollar programas de formación continua para desarrolladores sobre prácticas de desarrollo seguro y las últimas amenazas de seguridad. Esto incluye talleres y cursos en línea, simulaciones de ataque, ejercicios de defensa y certificaciones en seguridad de software, por ejemplo, realizar capacitaciones que incluyan módulos específicos sobre OWASP Top Ten, así como nuevas amenazas emergentes y cómo mitigarlas.

	Sistema de Gestión de Seguridad de la Información	Código:	SGSI-PO-021
		Versión:	1.0
	Política de Desarrollo Seguro	Clasificación:	Uso Interno

Autenticación

- Garantizar que las contraseñas no se almacenan en un formato legible, usando funciones que garanticen la irreversibilidad de la operación, como el uso de funciones hash fuertes.
- Asegurar que cada página de la aplicación tiene un enlace de desconexión, que la sesión expira cuando el usuario se desconecta y que la sesión expira cuando pasa un tiempo prudente de uso sin actividad.
- Implementar el bloqueo de la cuenta después de tres (3) intentos fallidos de conexión y una forma de ponerse en contacto con el administrador para desbloquearla.
- Implementar CAPTCHA para mitigar los ataques de fuerza bruta.
- Proporcionar mensajes de error genéricos en caso de fallo de autenticación, como "El usuario y/o la contraseña proporcionada no son correctos".
- No almacenar de forma persistente la cookie de autenticación en el ordenador del cliente, y no utilizarla para otros fines como la personalización.
- Asegurarse de que la sesión es diferente cada vez un usuario se ha conectado con éxito a la aplicación.
- Registrar todos los intentos exitosos o fallidos de autenticación sin exponer la clave utilizada.
- Registrar en un registro específico de seguridad los intentos maliciosos de acceso. Detección de múltiples intentos fallidos de autenticación, detección de intentos de inyección, como SQL o LDAP, detección de múltiples usuarios para mismas IPs, etc.

Autorización

- Garantizar la implementación del principio de mínimo privilegio: los usuarios tienen acceso restringido sólo a las funciones y datos que realmente necesitan para realizar su trabajo con normalidad.
- Asignar los permisos y privilegios a roles de aplicación, nunca directamente a los usuarios. Los usuarios lo que deben tener son roles y sus privilegios son tomados de éstos.
- Verificar que la navegación por los directorios esté deshabilitada.
- Asegurarse de que la aplicación utiliza tokens aleatorios fuertes anti-CSRF o que implementa otro mecanismo de protección de transacciones.
- Registrar todas las operaciones sobre datos sensibles en un registro específico de seguridad donde conste, al menos: la fecha/hora de la operación, la operación (lectura, creación, borrado, actualización), el nombre del dato, el proceso, función o servicio que generó la operación, el usuario, el rol del usuario que tiene el privilegio para la operación, el resultado de la operación (si fue exitosa o no) y un mensaje opcional sobre el resultado de la operación (en caso de error).

Sesiones

- Asegurarse de que el ID de sesión nunca se exponga en el tráfico no cifrado.
- Implementar cabeceras seguras con directivas como cachecontrol o strict-transport security.
- Compruebe que las sesiones se invalidan cuando el usuario cierra la sesión.
- Las sesiones deben expirar después de un tiempo de inactividad específico.

2. Identificación de Roles

Estos roles se apoyan en un esquema de usuarios internos vs. externos con diferentes niveles de exposición a las API y servicios REST/WMS/WFS. Los internos pueden consumir la funcionalidad completa; los externos solo servicios restringidos y previamente filtrados por la capa de integración/servicios:

Rol / Perfil	Tipo de usuario	Principales atribuciones dentro del SCR
Administrador DIGESPACR	Interno – nivel central	Parametriza el sistema (capas GIS, procesos, servicios de interoperabilidad), gestiona usuarios globales y valida la calidad de la data que remiten los GOREs
Administrador GORE	Interno – gobierno regional	Administra el SCR en su región: crea usuarios locales, asigna permisos y supervisa la carga de información
Administrador GIS	Interno – gobierno regional	Mantiene la geodatabase, publica servicios cartográficos y valida consistencia geométrica
Analista GIS	Interno – gobierno regional	Edita y analiza información geoespacial necesaria para la titulación; apoya a equipos de campo
Diagnosticador	Interno – gobierno regional	Conduce y documenta el diagnóstico físico-legal de unidades territoriales antes del saneamiento
Calificador	Interno – gobierno regional	Verifica el cumplimiento normativo de expedientes y determina su procedencia
Autorizador	Interno – gobierno regional	Emite la autorización formal para la expedición de instrumentos de titulación
Digitador	Interno – gobierno regional	Ingresa masivamente datos de fichas catastrales y resultados de campo
Generador de Padrones	Interno – gobierno regional	Consolida y publica padrones de poseedores aptos en procesos masivos
Generador de Instrumentos	Interno – gobierno regional	Produce títulos, resoluciones y demás instrumentos legales para inscripción registral
Generador de Certificados	Interno – gobierno regional	Emite certificados catastrales y reportes cartográficos
Visualizador Interno	Interno – cualquier entidad estatal habilitada	Consulta (solo-lectura) la información catastral para fines de gestión pública
Visualizador Externo	Externo – SUNARP, otras entidades estatales o ciudadanos	Accede vía servicios web/IDE a subconjuntos de datos autorizados

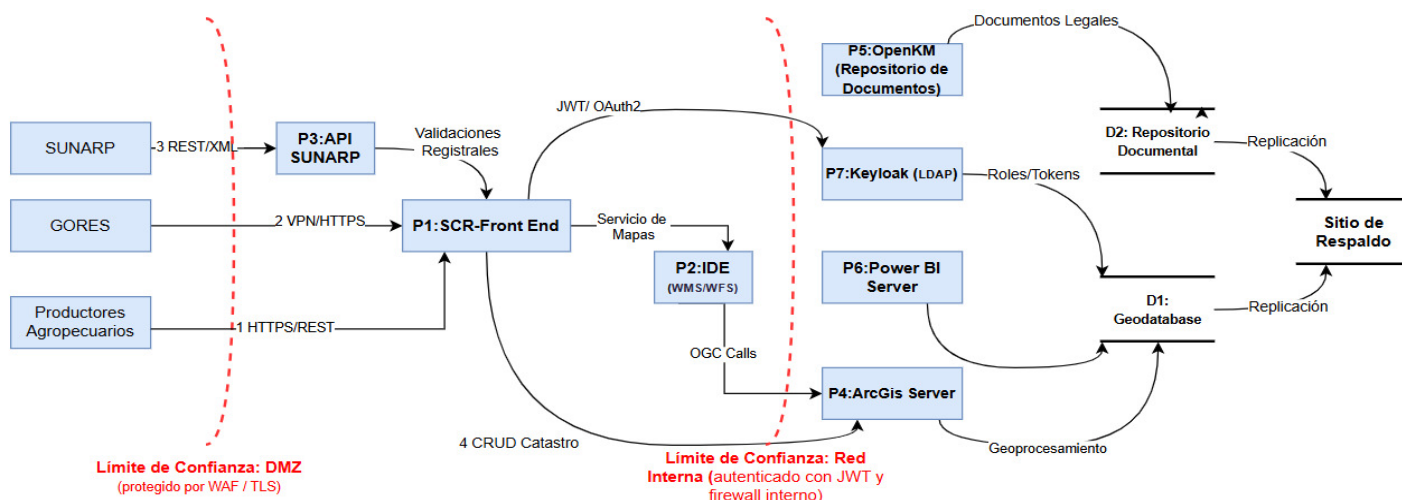
Perfiles de posibles atacantes identificados a partir de los requerimientos funcionales del SCR:

Perfil de atacante	Motivación	Capacidades / vector esperado	Riesgos principales para el SCR
Empleado interno descontento	Venganza, beneficio personal	Conoce flujos de negocio y credenciales; puede abusar de privilegios de <i>Administrador GORE</i> , <i>Analista GIS</i> o <i>Digitador</i>	Manipulación de datos catastrales, borrado selectivo, filtración de información sensible
Contratista / proveedor	Económica	Acceso temporal a la infraestructura (p. ej. mantenimiento de ArcGIS Server o BD); podría introducir puertas traseras	Inyección de código malicioso, robo de credenciales de administradores
Ciberdelincuente externo	Ganancia económica (venta de tierras, extorsión)	Phishing a usuarios internos, explotación de servicios web expuestos (REST/WFS/WMS)	Alteración de límites y titularidad de predios, ransomware sobre la geodatabase
Atacante de cadena de suministro	Comprometer a terceros para llegar al SCR	Inserta malware en actualizaciones de software (framework .NET, librerías GIS)	Ejecución remota de código con privilegios de sistema; exfiltración gradual de la base catastral
Hacktivista / actor político	Desprestigio o presión social	Ataques de denegación de servicio a portales públicos (SICAR web) o defacement	Interrupción de servicios de consulta pública; pérdida de confianza ciudadana
Amenaza estatal / APT	Inteligencia territorial, intereses estratégicos	Recursos avanzados, ataques dirigidos (0-days, spear-phishing)	Acceso prolongado y sigiloso, alteración o copia masiva de información sobre propiedad rural

3. Modelado de Amenazas

a. Modelado del Diagrama de Flujo de Datos

Se modela el Diagrama de Flujo de Datos (descomposición de la aplicación) bajo la notación OWASP DFD, respetando los límites de confianza (*trust boundaries*) recomendados en el proceso de *Threat Modeling* de OWASP de acuerdo con el marco de trabajo propuesto, para identificar actores, procesos, datos y límites.



Modelado de Amenazas-Diagrama de Flujo de Datos

Explicación de los procesos y almacenes (*Data Store*):

Id	Tipo	Nombre	Función principal
P1	Proceso	SCR Front-End (Vue/.NET)	Gestor de trámites catastrales, padrones, certificaciones y reportes
P2	Proceso	IDE/OGC Node	Publica servicios WMS/WFS/WPS para visores y externos
P3	Proceso	API SUNARP	Intercambio de estados registrales, títulos y validaciones
P4	Proceso	ArcGIS Server	Geoprocetamiento y mapas internos/externos
P5	Proceso	OpenKM	Repositorio de documentos de respaldo legal
P6	Proceso	Power BI Server	Cuadros de mando operativos (KPIs, avance de proyectos)
P7	Proceso	Gestión de Identidades	SSO, MFA, emisión de JWTs para micro-servicios
D1	Data Store	Geodatabase Única	Modelo normalizado + Catálogo de Objetos DIGESPACR
D2	Data Store	Repositorio Documental	Expedientes PDF, resoluciones, planos firmados

Límites de confianza clave:

- Internet → DMZ – protegido por WAF / TLS.
- DMZ → Servicios internos – autenticado con JWT y firewall interno.
- Servicios → Datos – red interna aislada; sólo P4 y P7 tienen acceso directo a D1/D2.

b. Identificación de posibles amenazas aplicando el modelo STRIDE

- **Spoofing (Suplantación de Identidad):** Protejamos el punto P7 activando verificación en dos pasos y pidiendo un certificado digital al usuario, de modo que nadie pueda hacerse pasar por él.
- **Tampering (Manipulación de Datos):** Todo archivo que envíen los gobiernos regionales debe ir firmado y ser verificado al llegar, y la comunicación debe viajar cifrada con la versión segura de TLS para que no pueda alterarse en tránsito.
- **Repudiation (Negación de Acciones):** Guardemos cada operación en un registro que no se pueda borrar ni modificar, añadiendo además la fecha y hora exactas, de forma que nadie pueda negar lo que hizo.
- **Information Disclosure (Filtración de Información):** Cifremos las bases de datos D1 y D2 “en reposo” y clasifiquemos los datos según su sensibilidad para que solo las personas autorizadas puedan verlos.
- **Denial of Service (Ataques de Saturación):** Limitemos el número de peticiones por usuario desde el firewall de aplicaciones y usemos colas de mensajes entre los procesos P1 y P4 para que el sistema siga respondiendo aún bajo carga.
- **Elevation of Privilege (Escalada de privilegios):** Apliquemos permisos mínimos basados en roles tanto en ArcGIS como en la base de datos, otorgando a cada usuario solo lo estrictamente necesario para su trabajo.

c. Clasificación de los riesgos con OWASP Risk Rating o DREAD.

d. Selección de contramedidas (requisitos de seguridad) y vincularlas como historias de usuario relacionadas a la seguridad.

4. Análisis de Riesgos de Seguridad (Identificación de Ataques) y Análisis de Requisitos de Seguridad

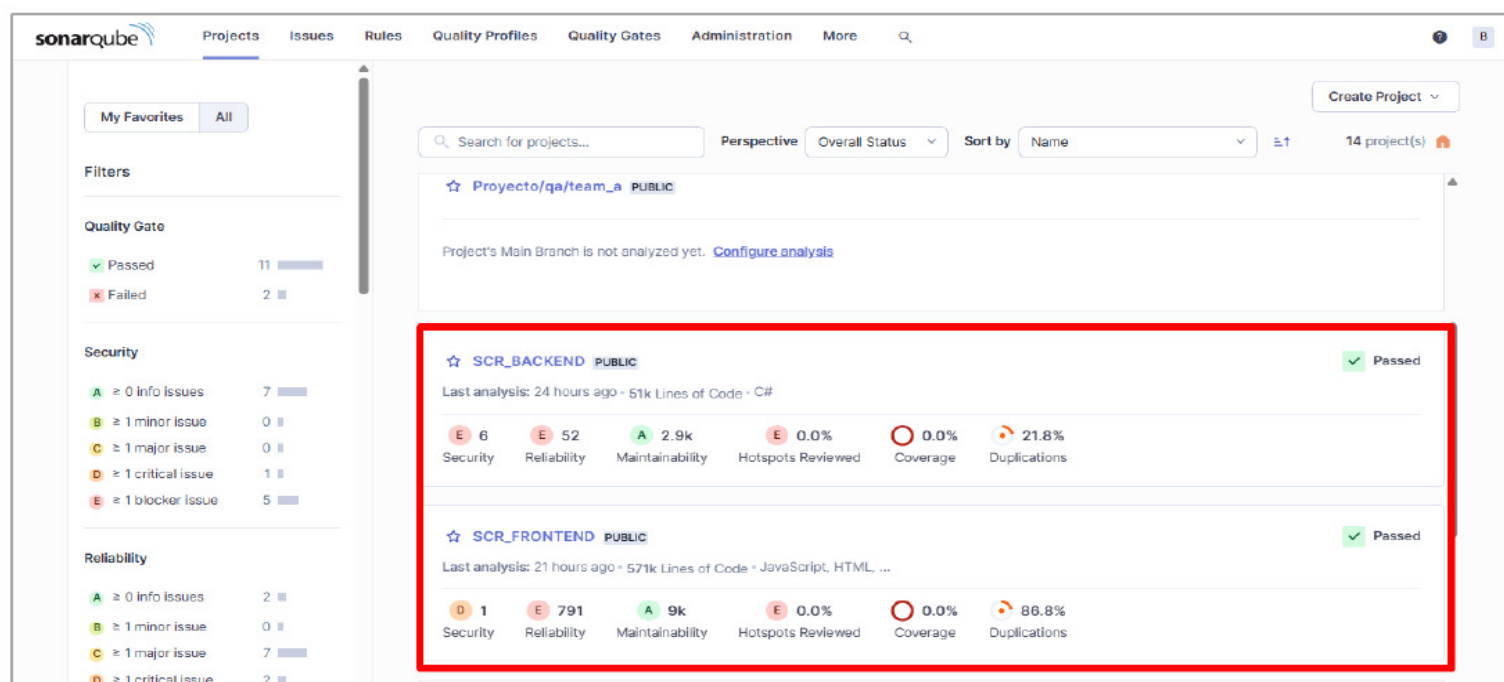
A partir del Modelado de Amenazas se identificaron y clasificaron los riesgos de seguridad expresados como historias de abuso (HA), así como las contramedidas o requisitos de seguridad expresado como historias de usuario relacionadas a la seguridad (HUS).

Historias de Abuso (Amenazas)	Historias de Usuario Relacionadas a Actividades de Seguridad
HA-01 – “Login fantasma”: Un atacante externo roba credenciales de un usuario y entra al portal SCR para consultar o modificar datos catastrales.	HUS-01 – Autenticación robusta: Como ciudadano o funcionario quiero autenticarme con doble factor (MFA) y tokens OAuth 2.0 emitidos por Keycloak para impedir que intrusos usen credenciales robadas.
HA-02 – “El paquete adulterado”: Un intruso intercepta el tráfico entre el front-end y ArcGIS Server y altera una petición de actualización de parcela.	HUS-02 – Canales cifrados y firmados: Como backend de geoprocesamiento quiero aceptar solo peticiones TLS 1.2+ firmadas con JWS para garantizar que ningún mensaje sea modificado en tránsito.
HA-03 – “Yo no fui”: Un gobierno regional niega haber cargado un shapefile incorrecto que generó superposiciones.	HUS-03 – Trazabilidad inalterable: Como auditor del MIDAGRI quiero disponer de logs sellados y hash SHA-256 de cada archivo cargado para demostrar quién subió qué y cuándo, sin posibilidad de repudio.
HA-04 – “Robo silencioso”: Un insider descarga la geodatabase y filtra planos de predios privados.	HUS-04 – Cifrado y vistas restringidas: Como administrador de base de datos quiero tener TDE/AES-256 y vistas con mascarado de datos para que información sensible sea ilegible sin permisos adecuados.
HA-05 – “Ataque de tráfico pesado”: Bots saturan los servicios WMS/WFS provocando caída del visor geoespacial.	HUS-05 – Resiliencia a picos de carga: Como usuario legítimo del visor quiero que el sistema limite peticiones por segundo y use colas asíncronas para mantener la disponibilidad aún bajo ataques de denegación de servicio.
HA-06 – “Token zombificado”: Un atacante reutiliza un token expirado para escalar privilegios y editar datos.	HUS-06 – Tokens de vida corta y validación de ámbito: Como microservicio interno quiero verificar que cada token tenga ≤ 10 min de vigencia y el rol mínimo necesario para evitar elevaciones de privilegio por reutilización.
HA-07 – “Shapefile venenoso”: Un archivo subido por un GORE contiene <i>malware</i> o datos corruptos que comprometen el servidor.	HUS-07 – Escaneo y validación de cargas: Como servicio de ingesta de archivos quiero analizar cada carga con antivirus/YARA y validar el esquema geoespacial para aceptar solo contenidos seguros y bien formados.
HA-08 – “Inyección geoespacial”: Un atacante envía parámetros OGC manipulados para ejecutar SQL malicioso.	HUS-08 – Consultas parametrizadas y WAF: Como ArcGIS Server quiero usar consultas preparadas y reglas WAF contra patrones de inyección para neutralizar intentos de ejecución SQL/OGC.
HA-09 – “Respallos perdidos”: Un fallo de infraestructura borra la única copia de la geobase y de los documentos legales.	HUS-09 – Estrategia 3-2-1 de backup: Como responsable de continuidad de negocio quiero mantener tres copias en dos medios distintos y un sitio remoto cifrado para restaurar la información catastral ante cualquier desastre.
HA-10 – “Curiosidad indebida”: Un empleado sin relación con un expediente accede a documentos notariales en OpenKM.	HUS-10 – Control de acceso granular: Como gestor documental quiero aplicar ACL por expediente y auditar cada descarga para impedir accesos no autorizados y detectar intentos de husmeo.
HA-11 – “El cartógrafo resentido”: Un Analista GIS con acceso legítimo modifica silenciosamente los límites de varias parcelas para favorecer a un tercero.	HUS-11 – Firma y control de versiones en la geodatabase: Como Administrador GIS quiero que cada edición sobre capas catastrales quede firmada digitalmente y versionada automáticamente para poder rastrear a qué usuario pertenece cada cambio y revertirlo si es malicioso.
HA-12 – “Actualización envenenada”: Un proveedor externo inserta código malicioso en un “hot-fix” de ArcGIS Server aprovechando su acceso de mantenimiento.	HUS-12 – Revisión de código y SBOM: Como responsable DevSecOps quiero que cualquier paquete o parche pase por análisis SCA/SAST y registro SBOM antes de desplegarse en el <i>cluster</i> para impedir que código no verificado llegue a producción.
HA-13 – “RansomGIS”: Un ciberdelincuente externo consigue credenciales de un Digitador vía phishing, exfiltra la geobase y cifra los servidores exigiendo rescate.	HUS-13 – Copias 3-2-1 y detección de ransomware: Como encargado de continuidad de negocio quiero tener backups inmutables diarios verificados automáticamente y una sonda que detecte patrones de cifrado masivo para restaurar rápidamente el servicio sin pagar rescates.
HA-14 – “Cortina de humo”: Un hacktivista lanza un DDoS contra las APIs públicas (WMS/WFS) para denunciar supuestos conflictos de tierras, dejando el portal inoperativo.	HUS-14 – Protección contra DDoS y rate-limiting: Como usuario público legítimo quiero que el SCR disponga de WAF con limitación de peticiones y mitigación DDoS en la nube para garantizar la continuidad de la consulta ciudadana aún bajo ataques de saturación.
HA-15 – “Mapeador fantasma”: Un APT estatal se infiltra de forma sigilosa y descarga la cartografía completa para analizar recursos estratégicos.	HUS-15 – Segmentación zero-trust y monitoreo de exfiltración: Como equipo SOC quiero implementar microsegmentación, reglas DLP y alertas de tráfico inusual hacia el exterior para detectar y cortar cualquier transferencia masiva no autorizada.

5. Análisis Estático de Código y Pruebas de Penetración

Se realizó el análisis estático de código y pruebas de penetración, del software producido, usando la herramienta SonarQube revelando que tanto el Back-end (API .NET Core) como el Front-end presentan riesgos importantes:

- **Seguridad:** 6 vulnerabilidades críticas y 7 security hotspots en el Back-end; 1 vulnerabilidad y 32 hotspots en el Front-end. Estas fallas requieren corrección prioritaria para evitar exposiciones en producción.
- **Fiabilidad:** 52 problemas altos en el Back-end y 791 en el Front-end indican potenciales fallos de ejecución si no se refactoriza el código.
- **Mantenibilidad:** aunque Sonar asigna calificación “A”, existen $\approx 2\,900$ incidencias en el Back-end y 9 000 en el Front-end que degradan la legibilidad y elevan la deuda técnica.
- **Duplicación de código:** 21.8 % en el Back-end y un alarmante 86.8 % en el Front-end, lo que incrementa costes de mantenimiento y riesgo de errores.



Resultado de análisis de código con SonarQube al Front-End y Back-End del SCR