UNIVERSIDAD DE COSTA RICA

SISTEMA DE ESTUDIOS DE POSGRADO

PROPUESTA DE UN MODELO METODOLÓGICO PARA LA GESTIÓN Y ADOPCIÓN DE PRÁCTICAS SEGURAS, EN LOS EQUIPOS DE DESARROLLO DE *SOFTWARE*, EN UNA ENTIDAD BANCARIA COSTARRICENSE

Trabajo final de investigación aplicada sometido a la consideración de la Comisión del Programa de Estudios de Posgrado en Tecnologías de Información y Comunicación para la Gestión Organizacional para optar por el grado y título de Maestría Profesional en Tecnologías de Información y Comunicación para la Gestión Organizacional

LUIS ÁLVARO MORALES ULATE

Ciudad Universitaria Rodrigo Facio, Costa Rica

2026

DEDICATORIA

Dedicado a mi familia y a Dios, por su apoyo en los momentos complicados que pasé al inicio de este posgrado y que pusieron en duda el poder comenzar, finalmente se pudo, todo gracias a su apoyo en todo momento, a la fortaleza mental, determinación y coraje necesarios para superar etapas en las que la vida pareciera ser complicada.

También deseo dedicarle este trabajo a mi coordinadora de tesis Sindy Porras, a los distintos profesores, lectores y compañeros cuyo aporte me permitió poder finalizar esta etapa, un paso importante para mi mayor deseo a corto plazo: ser profesor.

AGRADECIMIENTO

Al profesor Kenneth Sánchez, por su ayuda y consejo al momento de elegir cursar este posgrado, fue de ayuda importante cuando terminé el bachillerato, su consejo me ayudó a decidir el momento en el que era más provechoso para mí el cursar esta maestría.

Agradezco al profesor James Mc Intosh, por sus comentarios en la lectura de este documento y por sus enseñanzas en los distintos cursos del posgrado, un profesor que tiene vocación y de quien sin duda me llevo muchas de sus estrategias para aplicarlas en mis futuras clases.

A mis compañeros y amigos, Omar Miranda y Josué Barrantes, juntos comenzamos este reto y con trabajo en equipo hemos podido superarlo con orgullo y excelencia.

Finalmente, a todo el personal docente y administrativo de la universidad, el esfuerzo individual de cada uno nos permite a nosotros los estudiantes cumplir nuestros sueños con educación de calidad.

Este trabajo final de investigación aplicada fue aceptado por la Comisión del Programa de Estudio de Posgrado en Tecnologías de Información y Comunicación para la Gestión Organizacional de la Universidad de Costa Rica, como requisito parcial para optar el grado y título en Maestría Profesional en Tecnologías de Información y Comunicación para la Gestión Organizacional.

AARON GALAGARZA CARRILLO (FIRMA)
Firmado digitalmente por AARON GALAGARZA CARRILLO (FIRMA)
Fecha: 2026.04.08 09:40:09 -06'00'

M. Sc. Aarón Galagarza Carrillo

Representante de la Decanatura

Sistema de Estudios de Posgrado

SINDY PORRAS SANTAMARIA (FIRMA)
Firmado digitalmente por SINDY PORRAS SANTAMARIA (FIRMA)
Fecha: 2026.04.08 09:19:07 -06'00'

M. Sc. Sindy Porras Santamaría

Profesora Guía

KENNETH SANCHEZ SANCHEZ (FIRMA)
Firmado digitalmente por KENNETH SANCHEZ SANCHEZ (FIRMA)
Fecha: 2026.04.07 17:06:48 -06'00'

M. Sc. Kenneth Sánchez Sánchez

Lector

JAMES FRANCISCUS MC INTOSH MOLINA (FIRMA)
Firmado digitalmente por JAMES FRANCISCUS MC INTOSH MOLINA (FIRMA)
Fecha: 2026.04.08 08:59:19 -06'00'

M. Sc. James Franciscus Mc Intosh Molina

Lector

YORLENY SALAS ARAYA (FIRMA)
Firmado digitalmente por YORLENY SALAS ARAYA (FIRMA)
Fecha: 2026.04.08 14:29:20 -06'00'

M. Sc. Yorleny Salas Araya

Directora

Programa de Posgrado en Tecnologías de Información y Comunicación para la Gestión Organizacional

LUIS ALVARO MORALES ULATE (FIRMA)
Firmado digitalmente por LUIS ALVARO MORALES ULATE (FIRMA)
Fecha: 2026.04.07 16:23:02 -06'00'

Luis Álvaro Morales Ulate

Sustentante

TABLA DE CONTENIDO

DEDICATORIA ---------- ii

AGRADECIMIENTO ---------- iii

HOJA DE APROBACIÓN ---------- iv

RESUMEN ---------- ix

ABSTRACT ---------- x

LISTA DE TABLAS ---------- xi

LISTA DE FIGURAS ---------- xii

LISTA DE ABREVIATURAS ---------- xiii

- Problemática ---------- 2
- Justificación ---------- 4
- Objetivos ---------- 5
  - Objetivo general ---------- 5
  - Objetivos específicos ---------- 5
- Fundamentación teórica y conceptual ---------- 6
  - Contexto internacional ---------- 6
  - Contexto costarricense ---------- 8
- Marco referencial ---------- 9
- Marco conceptual ---------- 10
- Descripción de la empresa u organización y su entorno ---------- 11
- Marco metodológico ---------- 12
- Alcance de la investigación ---------- 13
- Métodos y técnicas de recolección de datos ---------- 13
  - Población ---------- 13
  - Fuentes de información ---------- 13
  - Técnicas de recolección de datos ---------- 14
- Mecanismos para la tabulación y el análisis ---------- 15
- Tabulación y análisis de la información ---------- 16

CAPÍTULO II. ANÁLISIS DEL ESTADO ACTUAL DE LAS PRÁCTICAS DE DESARROLLO DE *SOFTWARE* SEGURO EN UNA ENTIDAD DEL SECTOR BANCARIO PÚBLICO COSTARRICENSE ---------- 18

CAPÍTULO III. ANÁLISIS DE DIFERENTES MARCOS METODOLÓGICOS Y PRÁCTICAS EXISTENTES CORRESPONDIENTES AL DESARROLLO DE *SOFTWARE* SEGURO ---------- 32

CAPÍTULO IV. PROPUESTA DE MODELO PARA ADAPTAR LAS PRÁCTICAS DE DESARROLLO DE *SOFTWARE* SEGURO EN UNA ENTIDAD PÚBLICA DEL SECTOR BANCARIO COSTARRICENSE ---------- 37
1. Principios rectores del modelo ---------- 37
1.1 Seguridad como responsabilidad compartida ---------- 37
1.2 Enfoque *Secure by Design, by Default and by Deployment* ---------- 37
1.3 Minimización del riesgo y enfoque basado en amenazas ---------- 38
1.4 Trazabilidad y auditabilidad ---------- 38
1.5 Mejora continua y madurez incremental ---------- 38
1.6 Automatización como eje para la eficiencia y la reducción del error humano ---------- 39
1.7 Comunicación transparente y gestión del cambio cultural ---------- 39
1.8 Protección del consumidor financiero y cumplimiento normativo ---------- 40
2. Estructura del modelo ---------- 40
2.1 Capa estratégica: gobernanza y dirección del programa ---------- 40
2.2 Capa de procesos: integración con el ciclo de vida del desarrollo ---------- 41
2.3 Capa de soporte: cultura, capacitación y herramientas ---------- 43
2.4 Capa de métricas y evaluación continua ---------- 43
3. Arquitectura del modelo ---------- 44
3.1 Estructura general del modelo ---------- 44
3.2 Mapa de integración entre los dominios del modelo y los estándares ---------- 45
3.3 Estructura interna del modelo ---------- 46
4. Dominios y prácticas ---------- 46
4.1 Gobernanza y dirección del programa ---------- 47
4.1.1 Propósito ---------- 47
4.1.2 Prácticas del dominio ---------- 47
4.1.3 Actividades mínimas ---------- 48
4.1.4 Evidencias y artefactos ---------- 49
4.2 Gestión del ciclo de vida del desarrollo seguro (SSDLC) ---------- 50
4.2.1 Propósito ---------- 50
4.2.2 Prácticas del dominio ---------- 50
4.2.3 Actividades mínimas ---------- 51
4.2.4 Evidencias y artefactos ---------- 52
4.3 Gestión de requisitos de seguridad en el SDLC ---------- 53
4.3.1 Propósito ---------- 53

4.3.2 Prácticas del dominio ---------- 53
4.3.3 Actividades mínimas ---------- 54
4.3.4 Evidencias y artefactos ---------- 54
4.4 Gestión de incidentes y respuesta segura ---------- 55
4.4.1 Propósito ---------- 55
4.4.2 Prácticas del dominio ---------- 55
4.4.3 Actividades mínimas ---------- 56
4.4.4 Evidencias y artefactos ---------- 57
4.5 Gestión de proveedores y artefactos externos ---------- 58
4.5.1 Propósito ---------- 58
4.5.2 Prácticas del dominio ---------- 58
4.5.3 Actividades mínimas ---------- 59
4.5.4 Evidencias y artefactos ---------- 60
4.6 Cultura, formación y concienciación ---------- 60
4.6.1 Propósito ---------- 60
4.6.2 Prácticas del dominio ---------- 61
4.6.3 Actividades mínimas ---------- 61
4.6.4 Evidencias y artefactos ---------- 62
4.7 Herramientas, automatización y soporte tecnológico ---------- 63
4.7.1 Propósito ---------- 63
4.7.2 Prácticas del dominio ---------- 63
4.7.3 Actividades mínimas ---------- 64
4.7.4 Evidencias y artefactos ---------- 65
4.8 Métricas, auditoría y mejora continua ---------- 66
4.8.1 Propósito ---------- 66
4.8.2 Prácticas del dominio ---------- 66
4.8.3 Actividades mínimas ---------- 67
4.8.4 Evidencias y artefactos ---------- 67
CAPÍTULO V. RECOMENDACIONES PARA APLICAR EL MODELO ---------- 69
1. Establecer un gobierno claro del programa de desarrollo seguro ---------- 69
2. Integrar el SSDLC dentro de los procesos actuales de desarrollo ---------- 69
3. Formalizar los requisitos de seguridad en todos los proyectos ---------- 70
4. Fortalecer la gestión de incidentes desde el desarrollo ---------- 70

5. Gestionar adecuadamente proveedores y artefactos externos---------- 70
6. Construir una cultura organizacional orientada al desarrollo seguro ---------- 70
7. Fortalecer el soporte tecnológico ---------- 70
8. Implementar métricas, auditoría y mejora continua---------- 71
9. Adoptar el modelo según nivel de madurez ---------- 71
CAPÍTULO VI. CONCLUSIONES Y RECOMENDACIONES ---------- 74
REFERENCIAS BIBLIOGRÁFICAS ---------- 76
Anexos ---------- 79

RESUMEN

El sector bancario costarricense enfrenta un entorno de creciente exposición a amenazas cibernéticas, en un contexto donde la digitalización de los servicios financieros incrementa la superficie de ataque y la dependencia tecnológica institucional. A pesar de la existencia de normativas nacionales e internacionales orientadas a la protección de la información, persisten brechas significativas en la integración sistemática de prácticas de desarrollo de *software* seguro dentro de los equipos técnicos, particularmente en las fases tempranas del ciclo de vida del desarrollo.

La presente investigación aplicada tuvo como objetivo proponer un modelo metodológico estructurado que facilite la gestión y adopción de prácticas de desarrollo de *software* seguro en una entidad del sector bancario público costarricense. Para ello, se desarrolló un estudio con enfoque mixto, que combinó un diagnóstico del estado actual mediante la aplicación de una encuesta a 25 profesionales del área de desarrollo, junto con un análisis comparativo de los marcos internacionales OWASP SAMM, Microsoft Security Development Lifecycle (SDL) y NIST Secure Software Development Framework (SSDF).

Los resultados evidencian una alta valoración conceptual de la seguridad por parte de los profesionales encuestados; no obstante, se identificó una implementación predominantemente reactiva, concentrada en fases de codificación y pruebas, con limitada integración en etapas de diseño y planificación. Asimismo, se constató una baja adopción formal de estándares internacionales, lo que refleja la ausencia de una guía metodológica adaptada al contexto organizacional y regulatorio nacional.

Como respuesta a estas brechas, se diseñó un modelo metodológico compuesto por ocho dominios estructurados, que integran gobernanza, ciclo de vida seguro (SSDLC) gestión de requisitos, respuesta a incidentes, gestión de proveedores, cultura organizacional, automatización tecnológica y métricas para la mejora continua. El modelo articula principios de *Secure by Design*, responsabilidad compartida y madurez incremental, alineando buenas prácticas internacionales con las características operativas del sector bancario costarricense.

Se concluye que la adopción de un enfoque metodológico estructurado y contextualizado no solo fortalece la postura de ciberseguridad institucional, sino que también contribuye a la transparencia, resiliencia operativa y confianza pública en el sistema financiero nacional.

ABSTRACT

The Costa Rican banking sector operates in an increasingly complex cybersecurity landscape, where digital transformation has expanded the attack surface and heightened institutional technological dependency. Despite the existence of national and international regulatory frameworks aimed at safeguarding financial information, significant gaps remain in the systematic integration of secure software development practices within technical teams, particularly during the early stages of the software development lifecycle.

This applied research aimed to propose a structured methodological model to facilitate the management and adoption of secure software development practices within a public banking institution in Costa Rica. A mixed-methods approach was employed, combining a diagnostic assessment through a survey administered to 25 software development professionals with a comparative analysis of internationally recognized frameworks: OWASP SAMM, Microsoft Security Development Lifecycle (SDL), and the NIST Secure Software Development Framework (SSDF).

Findings reveal a strong conceptual awareness of the importance of security among professionals; however, implementation remains largely reactive, concentrated in coding and testing phases, with limited integration during design and planning stages. Additionally, a low formal adoption of international standards was identified, highlighting the absence of a context-adapted methodological guide aligned with national regulatory and organizational realities.

In response to these gaps, an eight-domain methodological model was designed, integrating governance, secure software development lifecycle (SSDLC), security requirements management, incident response, third-party management, organizational culture, technological automation, and metrics for continuous improvement. The model incorporates principles such as Secure by Design, shared responsibility, and incremental maturity, aligning global best practices with the operational characteristics of the Costa Rican banking sector.

It is concluded that adopting a structured and context-sensitive methodological approach strengthens institutional cybersecurity posture while enhancing transparency, operational resilience, and public trust in the national financial system.

LISTA DE TABLAS

Tabla 1 Relación entre objetivos, actividades e instrumentos de investigación 17
Tabla 2 Categorización de prácticas de seguridad reportadas por los encuestados 22
Tabla 3 Principales barreras en la implementación de prácticas de desarrollo seguro 23
Tabla 4 Factores facilitadores para la adopción de prácticas de desarrollo seguro 24
Tabla 5 Alineación de la cultura organizacional con la importancia de la seguridad en el proceso de desarrollo de software 28
Tabla 6 Comparación entre los marcos OWASP SAMM, Microsoft SDL y NIST CSF/SSDF 34
Tabla 7 Matriz de brechas de los marcos de desarrollo de software seguro 36
Tabla 8 Mapa de integración entre los dominios del modelo y los estándares 45

LISTA DE FIGURAS

Figura 1 Distribución de experiencia laboral 19
Figura 2 Importancia de la seguridad en el desarrollo de software 20
Figura 3 Fases de aplicación de prácticas de seguridad 21
Figura 4 Uso de estándares internacionales 25
Figura 5 Frecuencia de capacitaciones 26
Figura 6 Alineación de la cultura organizacional 27

LISTA DE ABREVIATURAS

- BCCR: Banco Central de Costa Rica.
- DDoS: Denegación de servicio distribuido.
- SUGEF: Superintendencia General de Entidades Financieras.
- SUPEN: Superintendencia de Pensiones
- SUGESE: Superintendencia General de Seguros
- SUGEVAL: Superintendencia General de Valores.
- CONASSIF: Consejo Nacional de Supervisión del Sistema Financiero Nacional.
- SSDLC: Ciclo de Vida de Desarrollo de *Software* Seguro.
- SDLC: Ciclo de Vida de Desarrollo de *Software*.
- DST: División de Servicios Tecnológicos.
- SGSI: Sistema de Gestión de Seguridad de la Información.
- API: Interfaz de programación de aplicaciones.
- CI/CD: Integración continua/Despliegue continuo.
- DMZ: Zona desmilitarizada.
- CSF: *Cybersecurity Framework*.
- SSDF: *Secure Software Development Framework.*
- SDL: *Security Development Lifecycle.*
- NIST: Instituto Nacional de Estándares y Tecnología.
- PMO: *Project Manager Office.*
- PW: *Practice Workflow.*
- SAST: *Static Application Security Testing.*
- DAST: *Dynamic Application Security Testing.*
- SCA: *Software Composition Analysis.*
- CI: Integración Continua.
- CD: Despliegue Continuo.
- CSIRT: *Computer Security Incident Response Team*.
- RACI: *Responsible, Accountable, Consulted, Informed.*
- SCA: *Software Composition Analysis.*
- API: *Application Programming Interface.*
- SAST: *Static Application Security Testing.*
- DAST: *Dynamic Application Security Testing*.
- SCA: *Software Composition Analysis.*
- DevSecOps: *Development Security Operations*

CAPÍTULO I. INTRODUCCIÓN

El sector bancario costarricense ha desempeñado un papel crucial en la estabilidad económica del país, destacándose en momentos clave, por ejemplo, en la crisis financiera global de 2008. En este contexto, la banca nacional implementó estrategias que permitieron generar más recursos a menores costos financieros y realizar una colocación responsable del crédito, lo que contribuyó a mantener el sistema financiero estable antes, durante y después de la crisis (Alcázar Villalobos, Carvajal Cascante, & Vallejo Esquivel, 2020).

A pesar de no contar con la solidez en plataformas financieras de otros países como Estados Unidos o España, la banca costarricense logró evitar el colapso que experimentaron algunas economías norteamericanas y europeas, posicionándose como un pilar de resiliencia económica.

Aun así, en los últimos años, la banca costarricense se ha visto envuelta en controversias que van más allá del desempeño financiero. Diversas investigaciones periodísticas, legislativas y de organismos de control como la Superintendencia General de Entidades Financieras (SUGEF) han revelado supuestos manejos indebidos y tráfico de influencias dentro de algunas instituciones bancarias, lo que ha generado un impacto negativo en la confianza pública. Estos problemas han puesto en evidencia las debilidades estructurales del sector bancario, que van desde la falta de transparencia en las decisiones hasta una preocupante influencia política en la gestión de las entidades financieras (Alcázar Villalobos et al., 2020).

A la luz de estos desafíos, surge la necesidad de fortalecer los mecanismos de transparencia y control, no solo en la administración del negocio bancario, sino también en los procesos técnicos y operativos, como el desarrollo de *software*, que son críticos para garantizar la seguridad de la información financiera. La banca costarricense enfrenta múltiples desafíos relacionados con la ética, la imagen, la credibilidad y la confianza, aspectos clave para su sostenibilidad y desarrollo (Alcázar Villalobos, Carvajal Cascante, & Vallejo Esquivel, 2020).

En este sentido, el desarrollo de *software* seguro es una prioridad, pues el manejo responsable de los sistemas digitales es clave para evitar vulnerabilidades que puedan ser explotadas por actores maliciosos, comprometiendo la estabilidad financiera y la confianza de los usuarios.

Existen normativas vigentes — tanto nacionales como internacionales— como el ISO/IEC 27001, reconocido estándar a nivel mundial que busca orientación para establecer, implementar, mantener y mejorar continuamente un sistema de gestión de la seguridad de la información. Este, junto con las regulaciones de la SUGEF, buscan garantizar la integridad y seguridad de los sistemas financieros. Sin embargo, cumplir con estas normativas implica superar importantes desafíos, especialmente para los equipos de desarrollo de *software*, quienes deben incorporar prácticas de seguridad desde las primeras fases del ciclo de vida de este.

Pese a su relevancia, la adopción de estas prácticas en el sector bancario costarricense ha sido lenta y, en muchos casos, ineficiente, debido a la falta de un enfoque metodológico claro que guíe a los equipos en la implementación efectiva de estas prácticas.

Con frecuencia, las empresas se enfocan en medidas de seguridad como antivirus, cortafuegos, políticas de acceso y cifrado para protegerse. No obstante, la seguridad en los sistemas informáticos suele considerarse demasiado tarde, ya cuando el producto está terminado y se aborda a través de pruebas de penetración. Por ello, si se detectan vulnerabilidades, estas se corrigen y se lanzan actualizaciones o "parches" para las aplicaciones de usuario. Aun así, este enfoque no es muy efectivo, ya que puede implicar cambios costosos y complejos, además los usuarios podrían no aplicar las actualizaciones a tiempo o nunca hacerlo.

Por lo anterior, esta investigación tiene como objetivo proponer un modelo metodológico que facilite la gestión y adopción de prácticas de desarrollo de *software* seguro en los equipos de desarrollo de una entidad del sector bancario costarricense. Este modelo busca no solo mejorar la seguridad de los productos de *software*, sino también contribuir a la transparencia y a la construcción de una cultura organizacional enfocada en la seguridad y la ética en la gestión bancaria. Así, se espera que el sector bancario costarricense fortalezca su resiliencia ante ciberataques y garantice la confianza en su infraestructura tecnológica.

- **Problemática**

La seguridad en el desarrollo de *software* es un aspecto crítico, especialmente en el sector bancario, donde la protección de datos sensibles es fundamental. Sin embargo, a pesar de la importancia de la seguridad, muchas entidades bancarias aún no integran prácticas de desarrollo seguro desde las primeras etapas del ciclo de vida. En lugar de ello, suelen aplicar “parches de seguridad” después del desarrollo, lo que puede resultar en sistemas más vulnerables y costosos de mantener (CCN-CERT, 2023).

Además, la creciente sofisticación de los ciberataques subraya la necesidad de integrar la seguridad desde el inicio del desarrollo. Un estudio de PowerDMARC (2023) destaca que las entidades financieras enfrentan amenazas constantes como el *malware, ransomware* y ataques de *phishing*, las cuales pueden mitigarse mediante prácticas de desarrollo seguro.

El *malware* es un *software* malicioso diseñado para dañar o infiltrarse en sistemas informáticos (IBM, 2024). Dentro de esta categoría, el *ransomware* se distingue por cifrar los datos de la víctima y exigir un rescate para restaurar el acceso (IBM, 2024). Por otro lado, el *phishing* es una técnica de ingeniería social que engaña a las personas para que revelen información confidencial, como contraseñas o datos bancarios, a través de correos electrónicos o sitios web fraudulentos (IBM, 2024).

Para fortalecer la seguridad del *software* desde sus fases iniciales, se recomienda la implementación de metodologías como el *Secure Software Development Lifecycle* (SSDLC) y el uso de herramientas automatizadas para la detección de vulnerabilidades (Innowise, 2023).

La pandemia de COVID-19 exacerbó estos desafíos, ya que el aumento del trabajo remoto incrementó la superficie de ataque y la exposición a amenazas cibernéticas. Según un informe de Iuvity, la ciberseguridad se ha convertido en una de las principales preocupaciones para las

instituciones financieras, destacando la necesidad de adoptar medidas de seguridad más robustas y flexibles (Ciberseguridad bancaria, n.d.).

El sector bancario costarricense enfrenta actualmente desafíos significativos relacionados con la seguridad de la información en sus sistemas de *software*, los cuales son críticos para la protección de los datos financieros de sus clientes y la continuidad de las operaciones bancarias. Prueba de ello fue el ataque de denegación de servicio ocurrido en enero del 2024, que hizo que las páginas web del BCCR, la SUGEF, la Superintendencia de Pensiones (Supen), la Superintendencia General de Seguros, el Consejo Nacional de Supervisión del Sistema Financiero Nacional (Conassif) y la Superintendencia General de Valores (Sugeval) estuvieran caídas durante varias horas (Delfino.cr, 2024).

Otro de los desafíos es la necesidad de concienciación y capacitación de los desarrolladores, arquitectos y usuarios sobre cómo crear *software* seguro. La educación en mejores prácticas de seguridad es fundamental para mitigar riesgos y vulnerabilidades (Adelyar & Norta, 2016; Howard & Lipner, 2006).

Aunque el sector ha implementado medidas tradicionales de ciberseguridad, como cortafuegos, antivirus y servicios de detección y contención, estas acciones suelen ser insuficientes, ya que la seguridad se aborda principalmente al final del ciclo de desarrollo de *software*. Este enfoque reactivo, en lugar de preventivo, lleva a soluciones costosas y complejas, ya que las vulnerabilidades detectadas deben corregirse mediante actualizaciones, lo cual depende de que los usuarios adopten los "parches", lo que no siempre ocurre de manera oportuna.

Entre las acciones tomadas por parte del sector bancario costarricense destacan la implementación de políticas específicas de ciberseguridad, el monitoreo continuo de sus sistemas mediante tecnologías avanzadas y la capacitación constante de su personal en la gestión de riesgos cibernéticos. Asimismo, se han establecido protocolos para la detección y respuesta ante incidentes, asegurando una gestión proactiva frente a posibles amenazas (Banco Central de Costa Rica, s. f.).

La creciente sofisticación de los ciberataques y la alta dependencia de la tecnología digital, aumentan la presión sobre las instituciones bancarias para reforzar la seguridad desde las etapas iniciales de desarrollo de sus sistemas. Sin embargo, en Costa Rica, el sector bancario y gubernamental ha mostrado dificultades para integrar prácticas de desarrollo seguro en sus equipos. Esto se debe en gran parte a la falta de un modelo metodológico adaptado que guíe a los desarrolladores en la implementación efectiva de medidas de seguridad durante todo el ciclo de vida del *software* institucional y el abierto al público, alineadas con el contexto local y las normativas vigentes.

Este escenario se complica aún más por una falta generalizada de concienciación y capacitación en seguridad dentro de los equipos de desarrollo. Si bien existen estándares internacionales, como ISO/IEC 27001 y modelos como BSIMM y OWASP SAMM, que ofrecen pautas para la adopción de prácticas de seguridad en el desarrollo de *software*, muchos de estos modelos no se ajustan completamente a las necesidades y capacidades del sector bancario costarricense.

Esto limita la efectividad de los esfuerzos de seguridad y aumenta el riesgo de exposición a ciberataques que pueden afectar tanto la estabilidad financiera como la confianza pública en las instituciones bancarias.

El sector bancario costarricense enfrenta un desafío significativo en la integración de prácticas de desarrollo seguro en sus procesos de *software*. La ausencia de un modelo metodológico claro dificulta la adopción efectiva de estrategias que garanticen la seguridad desde las fases iniciales del desarrollo. Esta deficiencia aumenta la vulnerabilidad de las instituciones ante ciberataques, comprometiendo la integridad de los sistemas y la confianza de los usuarios. Además, la falta de una cultura organizacional enfocada en la seguridad y la transparencia agrava el riesgo, dejando expuestas a las entidades financieras a amenazas cada vez más sofisticadas.

- **Justificación**

El desarrollo seguro del *software* enfrenta varios desafíos significativos que deben ser abordados para garantizar la integridad y la seguridad de las aplicaciones informáticas. Uno de los principales retos es el incremento de las amenazas cibernéticas. La creciente cantidad de delitos informáticos y la posibilidad de explotar vulnerabilidades en el *software* representan un riesgo constante para las organizaciones, comprometiendo su misión y operaciones (Jamil et al., 2018; Kamuni et al., 2019).

En este contexto, la seguridad informática en el sector bancario costarricense representa una preocupación creciente, especialmente en un entorno donde la dependencia de los sistemas digitales continúa aumentando. A pesar de que existen normativas y estándares internacionales que buscan garantizar la protección de la información financiera, los equipos de desarrollo de *software* en la banca aún enfrentan dificultades para integrar prácticas de seguridad desde las fases iniciales del ciclo de desarrollo.

Además, el enfoque tradicional hacia la seguridad en el desarrollo de *software* suele ser tardío o ineficiente, ya que, en general, la seguridad se aborda solo al final del ciclo de desarrollo, a través de pruebas de penetración. Este enfoque conlleva soluciones costosas y modificaciones significativas que, en ocasiones pueden no ser adoptadas por los usuarios, lo que pone en riesgo la efectividad de las medidas de seguridad implementadas (McGraw, 2004; Mohammed et al., 2017). Esta falta de adopción temprana de prácticas de seguridad genera vulnerabilidades que pueden ser explotadas por actores maliciosos, afectando la estabilidad del sistema bancario y la confianza de los usuarios.

En este sentido, esta investigación se centrará en proponer un modelo metodológico que facilite la gestión y adopción de prácticas de desarrollo de *software* seguro dentro de los equipos de desarrollo de *software* de una entidad del sector bancario costarricense. Este servirá como una herramienta clave para fortalecer la seguridad desde las etapas iniciales del desarrollo, lo que reducirá significativamente los riesgos asociados a ciberataques y permitirá a las instituciones bancarias ofrecer productos más seguros y confiables.

A largo plazo, esta propuesta contribuirá al fortalecimiento del sector financiero, no solo en Costa Rica, sino también en otros contextos donde se enfrenten problemas similares. Además, abrirá nuevas oportunidades para la investigación y aplicación de soluciones de seguridad informática en la industria bancaria. La seguridad del *software* exige la aplicación de diversas soluciones y recursos a lo largo de su ciclo de vida, lo que puede ser un desafío en términos de tiempo y costos (Pérez Lastra et al., 2023).

La comparación de distintos marcos de desarrollo seguro de *software* tiene como finalidad desarrollar un marco único adaptado a la necesidad de la organización que lo requiera. En un estudio realizado por Wee, Kudriavtseva y Gadyatskaya (2024) donde se aplicaron 37 encuestas y 8 entrevistas a profesionales del desarrollo de *software* se obtuvo que estos profesionales son conscientes de la existencia de marcos para desarrollo seguro de *software*, las organizaciones utilizan principalmente marcos hechos a medida que ofrecen más flexibilidad y se alinean mejor con los procesos de desarrollo.

Por lo anterior, la creación de un marco adaptado a las condiciones nacionales se convierte en una necesidad en estos días, donde la seguridad informática es un aspecto no negociable. Basado en la literatura, la adopción de varios marcos es una herramienta que puede impulsar la obtención de los mejores resultados, al aprovechar los aspectos de los marcos más reconocidos y eficientes.

- **Objetivos**
    - **Objetivo general**
        - Proponer un modelo metodológico estructurado y documentado que integre prácticas de desarrollo de *software* seguro, basado en el análisis comparativo de marcos internacionales y en el diagnóstico del estado actual de una entidad del sector bancario público costarricense.
    - **Objetivos específicos**
        - Diagnosticar el estado actual de las prácticas de desarrollo de *software* seguro mediante la aplicación de un instrumento de recolección de datos a profesionales del sector bancario público costarricense.
        - Comparar los marcos OWASP SAMM, Microsoft SDL y NIST SSDF mediante una matriz estructurada de criterios técnicos, organizacionales y de aplicabilidad al contexto costarricense.
        - Diseñar un modelo metodológico compuesto por dominios, prácticas, actividades mínimas y métricas, integrando los hallazgos del diagnóstico y del análisis comparativo.

- **Fundamentación teórica y conceptual**
    - **Contexto internacional**

El sector bancario costarricense, aunque demostró resiliencia durante la crisis financiera global de 2008 (Alcázar Villalobos, Carvajal Cascante, & Vallejo Esquivel, 2020), enfrenta una creciente presión para fortalecer su postura de ciberseguridad. Como se menciona en la introducción de este documento, investigaciones periodísticas, legislativas y de organismos de control como la SUGEF han revelado supuestos manejos indebidos y tráfico de influencias dentro de algunas instituciones bancarias, impactando negativamente la confianza pública. Estos problemas, junto con la necesidad de mantener la estabilidad económica, subrayan la crítica necesidad de medidas de seguridad robustas, particularmente en el desarrollo de *software*, para proteger datos financieros sensibles y mantener la integridad operativa.

Según el State of Cybersecurity Report 2021 de Accenture, el sector financiero es uno de los más afectados por ciberataques debido a la sensibilidad de los datos que maneja y su importancia crítica en la economía global. El informe destaca que amenazas como el *ransomware*, el *phishing* y los ataques de denegación de servicio (DDoS) son comunes en este sector y que la pandemia de COVID-19 exacerbó estos riesgos al acelerar la transformación digital y el trabajo remoto. Además, se enfatiza la necesidad de adoptar un enfoque proactivo en ciberseguridad y fomentar una cultura de seguridad organizacional para mitigar estos riesgos (Accenture, 2021).

Estos ataques varían desde *malware* y *ransomware* hasta *phishing* y ataques de denegación de servicio distribuido (DDoS), como se destaca en estudios de PowerDMARC (2023) e IBM (2024). Las consecuencias pueden ser severas, incluyendo pérdidas financieras, daño reputacional y sanciones regulatorias.

Varias normativas nacionales e internacionales buscan garantizar la integridad y seguridad de los sistemas financieros. Sin embargo, cumplir con estas normativas implica superar importantes desafíos, especialmente para los equipos de desarrollo de *software*, quienes deben incorporar prácticas de seguridad desde las primeras fases del ciclo de vida del *software*.

Varios marcos y estándares internacionales han surgido para guiar a las organizaciones en la construcción de prácticas de desarrollo de *software* seguro. Algunos ejemplos notables incluyen:

- ISO/IEC 27001: Este estándar, internacionalmente reconocido, proporciona un marco integral para establecer, implementar, mantener y mejorar continuamente un Sistema de Gestión de Seguridad de la Información (SGSI) (ISO/IEC 27001:2022). Ayuda a las organizaciones a gestionar los riesgos de seguridad de la información y asegurar la confidencialidad, integridad y disponibilidad de sus datos. El BCCR y la SUGEF han adoptado este estándar como parte de sus directrices para la gestión de la seguridad de la información en el sector financiero (Banco Central de Costa Rica, s. f.).

- Ciclo de Vida de Desarrollo de *Software* Seguro (SSDLC): El SSDLC es un marco que incorpora prácticas de seguridad en cada etapa del ciclo de vida del desarrollo de *software*, desde la planificación y el diseño hasta la implementación, las pruebas y el despliegue (OWASP, 2021). Al integrar la seguridad desde el principio y con frecuencia, las organizaciones pueden identificar y mitigar las vulnerabilidades antes de que sean explotadas.

- *Building Security In Maturity Model* (BSIMM): BSIMM es un marco para medir y mejorar las iniciativas de seguridad de *software* (Black Duck, n.d.). Proporciona un enfoque basado en datos para evaluar las prácticas de seguridad de una organización en comparación con los puntos de referencia de la industria e identificar áreas de mejora.

- OWASP *Software Assurance Maturity Model* (SAMM): SAMM es un marco abierto que ayuda a las organizaciones a formular e implementar una estrategia para la seguridad del *software* que se adapte a los riesgos específicos que enfrenta la organización (OWASP, 2021). Proporciona un enfoque flexible e iterativo para construir un programa de desarrollo de *software* seguro.

Ejemplos de casos reales e implementaciones:

- Violación de Datos de Equifax (2017): Esta violación, que comprometió los datos de 147 millones de personas, se atribuyó a la falta de aplicación de un parche para una vulnerabilidad conocida de manera oportuna (Perlroth & Goldman, 2017). Equifax enfrentó importantes daños financieros y reputacionales como resultado.

- Ataque DDoS a instituciones gubernamentales de Costa Rica (2024): El ataque DDoS de enero de 2024 que afectó a los sitios web de varias agencias gubernamentales clave de Costa Rica, incluyendo el BCCR y la SUGEF (Delfino.cr, 2024), demuestra la vulnerabilidad de la infraestructura digital de la nación y la necesidad de medidas de seguridad mejoradas. Este ataque subraya la necesidad de implementar medidas de protección contra ataques DDoS, como la implementación de *firewalls* de aplicaciones web (WAF) y servicios de mitigación de DDoS.

El sector bancario costarricense ha implementado políticas de ciberseguridad, monitoreo continuo de sistemas y capacitación constante de personal. Sin embargo, la creciente sofisticación de los ciberataques y la alta dependencia de la tecnología aumentan la presión sobre las instituciones bancarias para reforzar la seguridad desde las etapas iniciales de desarrollo de sus sistemas. Esto se complica aún más por una falta generalizada de concienciación y capacitación en seguridad dentro de los equipos de desarrollo, lo que limita la efectividad de los esfuerzos de seguridad.

- **Contexto costarricense**

En Costa Rica, el sector bancario también enfrenta desafíos en cuanto a la seguridad de la información, especialmente en el desarrollo de *software* seguro. Aunque hay avances, todavía hay áreas que necesitan mejorar.

Regulaciones y normativas

a) La SUGEF ha establecido reglas para proteger los sistemas financieros en Costa Rica. Estas reglas incluyen controles de seguridad y gestión de riesgos, pero la adopción de prácticas de desarrollo seguro en los equipos de *software* ha sido lenta (SUGEF, 2023).

b) El BCCR también ha implementado políticas de ciberseguridad, como el monitoreo constante de sistemas y la capacitación del personal. Sin embargo, aún hay retos en integrar la seguridad desde el inicio del desarrollo de *software* (BCCR, 2023). Dicha entidad también cuenta con políticas específicas de ciberseguridad en las que establece como alcance los servicios brindados por la División Servicios Tecnológicos (DST) del BCCR, para velar por la adecuada gestión de la seguridad tecnológica y brindar una protección adecuada a los recursos de información de la institución custodiados mediante la plataforma tecnológica.

   También cuenta con políticas para la gestión de la seguridad de la información mediante una estructura de gobierno basada en las mejores prácticas internacionales, con el fin de asegurar la protección de los activos de información del BCCR de usos no autorizados, modificación, daños o destrucción accidental o intencional.

c) Las demás entidades públicas financieras del país cuentan con departamentos de desarrollo y ciberseguridad propios. En algunos casos, estas entidades tercerizan la creación de sus sistemas a empresas consultoras con profesionales calificados, para reducir costos operativos, y si bien muchas veces esta medida es provechosa, se aleja de la supervisión en materia de ciberseguridad, puesto que no es común que estas entidades exijan que sus sistemas estén desarrollados bajo estándares internacionales en esta área.

Ciberataques recientes

d) En enero de 2024, Costa Rica sufrió un ataque de denegación de servicio (DDoS) que afectó a varias instituciones financieras, incluyendo el BCCR y la SUGEF. El ataque consistió en enviar múltiples peticiones a los sistemas que estas entidades exponen en internet, de tal forma que degradó estos servicios hasta hacerlos colapsar. Este tipo de ataques es muy utilizado por delincuentes informáticos para exigir dinero a cambio de detener el ataque. Las entidades lograron contener el ataque, sin embargo, quedo en evidencia que esto puede ocurrir en cualquier momento, de ahí la importancia de contar con prácticas seguras de desarrollo de *software* en los sistemas del sector bancario y la necesidad de adoptar prácticas de desarrollo seguro para prevenir futuros ataques (Delfino.cr, 2024).

Falta de concienciación y capacitación

e) Un estudio de Adelyar y Norta (2016) señala que uno de los mayores problemas en Costa Rica es la falta de concienciación y capacitación en seguridad entre los desarrolladores de *software*. Es crucial educar a los equipos en mejores prácticas de seguridad para reducir riesgos.

Experiencias en el sector público

f) En el sector público costarricense, los bancos han tenido dificultades para integrar prácticas de desarrollo seguro. Esto se debe, en parte, a la falta de un modelo metodológico adaptado que guíe a los desarrolladores en la implementación de medidas de seguridad durante todo el ciclo de vida del *software* (Ramírez, Aiello, & Lincke, 2020).

- **Marco referencial**

Modelos de madurez en seguridad del software

Los modelos de madurez en seguridad del *software* son marcos que ayudan a las organizaciones a evaluar y mejorar sus prácticas de seguridad en el desarrollo de *software*. Estos modelos proporcionan un conjunto estructurado de prácticas y actividades que las organizaciones pueden seguir para incrementar su nivel de madurez en seguridad, desde la implementación de controles básicos hasta la integración de prácticas avanzadas de seguridad en todo el ciclo de vida del desarrollo de *software* (OWASP, 2021; BSIMM, 2023).

Microsoft Security Development Lifecycle (SDL)

El Microsoft *Security Development Lifecycle* (SDL) es un marco de desarrollo seguro que integra prácticas de seguridad en todas las fases del ciclo de vida del *software*. Este enfoque estructurado incluye actividades como la identificación de requisitos de seguridad, la modelización de amenazas, la revisión de código, pruebas de seguridad y la gestión de incidentes. El SDL se destaca por su enfoque proactivo, que permite identificar y mitigar vulnerabilidades desde las etapas iniciales del desarrollo. Es especialmente relevante para el sector bancario, ya que proporciona controles específicos para cumplir con normativas internacionales como PCI DSS y GDPR, además de integrarse con tecnologías ampliamente utilizadas en el ámbito financiero.

OWASP Software Assurance Maturity Model (SAMM)

SAMM, desarrollado por OWASP, es un marco de madurez que ayuda a las organizaciones a evaluar y mejorar sus prácticas de seguridad en el desarrollo de *software*. SAMM se estructura en cuatro dominios clave: gobernanza, construcción, verificación y despliegue. Este modelo es altamente flexible y permite a las organizaciones avanzar de manera gradual en la implementación de prácticas seguras, adaptándose a su nivel de madurez y necesidades específicas. Para el sector bancario, SAMM ofrece métricas claras para medir el progreso en

seguridad, lo que facilita el cumplimiento de auditorías y regulaciones. Además, fomenta una cultura de seguridad al involucrar a todos los equipos en la adopción de prácticas seguras.

NIST Secure Software Development Framework (SSDF)

El NIST *Secure Software Development Framework* (SSDF), desarrollado por el Instituto Nacional de Estándares y Tecnología (NIST) de EE. UU., es un marco que proporciona prácticas recomendadas para el desarrollo seguro de *software*. El SSDF se centra en cuatro áreas clave: preparar la organización, proteger el *software*, producir *software* seguro y responder a vulnerabilidades. Este marco está alineado con estándares internacionales como ISO/IEC 27001 y NIST SP 800-53, lo que lo hace ideal para entornos altamente regulados, como el sector bancario. El SSDF enfatiza la gestión de riesgos y la respuesta proactiva a incidentes, aspectos críticos para proteger los datos sensibles de los clientes y cumplir con las normativas locales e internacionales.

- **Marco conceptual**

Ciberseguridad

El sitio web OWASP (n.d.) es una organización internacional sin ánimo de lucro que se enfoca en la seguridad de las aplicaciones web, menciona que la ciberseguridad consiste en comprender, gestionar y mitigar el riesgo de que sus datos críticos se divulguen (confidencialidad), se alteren (integridad) o se denieguen (disponibilidad).

Una de las compañías de seguridad informática más grande del mundo define la ciberseguridad como la práctica de defender las computadoras, los servidores, los dispositivos móviles, los sistemas electrónicos, las redes y los datos de ataques maliciosos. También se conoce como seguridad de tecnología de la información o seguridad de la información electrónica (Kaspersky, n.d.).

Desarrollo seguro

El concepto de desarrollo seguro se refiere a la disciplina dentro de la ingeniería de *software* que busca garantizar la seguridad de este desde sus primeras fases de desarrollo. Esto implica la implementación de mecanismos y la producción de evidencias que aseguren que el *software* cumple con las propiedades de seguridad requeridas. Un *software* seguro es aquel que minimiza el impacto de ataques, tolera aquellos que no puede resistir y se recupera rápidamente de los que no puede tolerar (Ortega, M., n.d).

El desarrollo seguro abarca un conjunto de técnicas, normas y conocimientos que permiten crear programas que no puedan ser modificados de forma ilegítima y que estén libres de fallos que comprometan la seguridad del sistema en el que operan. Además, se enfatiza la importancia de cambiar la mentalidad de los desarrolladores y mejorar los procesos de ingeniería del *software* para construir sistemas de información más robustos y confiables, contribuyendo así a la seguridad informática en general (Ortega, M., n.d).

Ciclo de vida del desarrollo de software seguro (SSDLC)

El Ciclo de Vida del Desarrollo de *Software* Seguro (SSDLC) es un enfoque integral que incorpora prácticas de seguridad en cada fase del desarrollo de *software*, desde la planificación hasta el mantenimiento. Este enfoque asegura que la seguridad no sea una consideración de último momento, sino una parte integral del proceso de desarrollo. Las etapas típicas del SSDLC incluyen la planificación, el análisis de requisitos, el diseño, el desarrollo, las pruebas, la implementación y el mantenimiento, con medidas de seguridad específicas implementadas en cada fase para identificar y mitigar vulnerabilidades de manera proactiva (Red Hat, 2022).

Normativas y estándares de seguridad

Las normativas y estándares de seguridad son conjuntos de directrices y requisitos diseñados para proteger la información y los sistemas de las organizaciones. Entre los más importantes se encuentran la ISO/IEC 27001, que establece los requisitos para un sistema de gestión de la seguridad de la información, y el NIST *Cybersecurity Framework*, que proporciona un marco de referencia para gestionar y reducir los riesgos de ciberseguridad. Estos estándares ayudan a las organizaciones a implementar controles de seguridad efectivos y a cumplir con las regulaciones legales y contractuales (ISO, 2024; NIST, 2023). La ISO/IEC 27001, en particular, se centra en la protección de la confidencialidad, integridad y disponibilidad de la información mediante un sistema de gestión de riesgos y controles de seguridad (SensorsTech, 2023).

Amenazas y vulnerabilidades comunes

Las amenazas y vulnerabilidades comunes en ciberseguridad incluyen una variedad de riesgos que pueden comprometer la integridad, confidencialidad y disponibilidad de los sistemas y datos. Entre las amenazas más comunes se encuentran el *malware*, el *ransomware*, los ataques de *phishing* y los ataques de denegación de servicio (DDoS). Las vulnerabilidades, por otro lado, son debilidades en los sistemas que pueden ser explotadas por estas amenazas, como errores de configuración, *software* desactualizado y contraseñas débiles (Ambit BST, 2020).

- **Descripción de la empresa u organización y su entorno**

La institución financiera objeto de esta investigación, la cual no mencionamos por motivos confidenciales, es una entidad de carácter público que desempeña un papel estratégico en la economía del país. Esta institución opera en un entorno altamente regulado, donde la transparencia y la seguridad de la información son pilares fundamentales. Dada su naturaleza estratégica, la entidad está sujeta a normativas nacionales e internacionales que buscan garantizar la integridad de sus sistemas y la protección de los datos financieros de los ciudadanos.

Además, en su rol de política monetaria, esta entidad colabora estrechamente con otras entidades financieras de carácter estatal que ofrecen servicios bancarios a la población. Estas

entidades, también propiedad del Estado, tienen como objetivo principal brindar acceso a servicios financieros inclusivos, como ahorro, préstamos e inversiones, con una fuerte presencia en todo el territorio nacional.

El entorno en el que opera esta institución está marcado por desafíos significativos en materia de ciberseguridad. En los últimos años, el aumento de ciberataques dirigidos a entidades financieras ha puesto en evidencia la necesidad de fortalecer los sistemas de seguridad, especialmente en el desarrollo de *software*. Incidentes recientes, como ataques de denegación de servicio (DDoS) y brechas de seguridad, han subrayado la importancia de adoptar prácticas de desarrollo seguro desde las etapas iniciales del ciclo de vida del *software* (Delfino.cr, 2024; Innowise, 2023).

La entidad cuenta con un departamento de ciberseguridad que colabora en el proceso de inducción y capacitación de los funcionarios en temas de esta índole. Este departamento también realiza análisis de vulnerabilidades y ejercicios de evaluación periódicos para reforzar el conocimiento en ciberseguridad, lo que sugiere que está involucrado en múltiples facetas de la ciberseguridad institucional, desde la capacitación hasta el análisis de riesgos.

Además, la entidad cuenta con equipos de desarrollo de *software* adaptados al uso de metodologías agiles para la creación de sus sistemas. Estos equipos se encargan de la creación de sistemas de información de uso interno o público, dirigido a la población en general, para automatizar diversos trámites necesarios para construir un sistema bancario robusto y a la vanguardia.

En este contexto, la institución ha implementado políticas específicas de ciberseguridad, como el monitoreo continuo de sus sistemas y la capacitación del personal en gestión de riesgos. Sin embargo, persisten desafíos relacionados con la integración de prácticas de desarrollo seguro en los equipos de *software*, lo que ha llevado a la necesidad de proponer un modelo metodológico adaptado a sus necesidades y al contexto local.

- **Marco metodológico**

El estudio comparativo propuesto en esta investigación tiene como objetivo analizar y contrastar los siguientes marcos de desarrollo de *software* seguro: OWASP SAMM, SDL y NIST, con el fin de crear un modelo metodológico tropicalizado para los equipos de desarrollo de una entidad bancaria costarricense.

Este tipo de diseño de *software* seguro implica la revisión exhaustiva de metodologías ampliamente reconocidas, como OWASP SAMM, SDL y NIST, donde cada una ofrece un enfoque distinto hacia la integración de prácticas de seguridad dentro del ciclo de vida del desarrollo de *software*, lo que permite establecer comparaciones en términos de estructura, implementación, roles, métricas y resultados esperados.

- **Alcance de la investigación**

El estudio se llevará a cabo identificando las fortalezas y debilidades de cada marco en contextos similares al costarricense, evaluando su adaptabilidad a las características y necesidades del sector bancario local. Se considerarán aspectos como el tamaño y complejidad de los equipos de desarrollo, los recursos tecnológicos disponibles, el nivel de madurez en ciberseguridad de las instituciones bancarias y las normativas locales vigentes que regulan la seguridad en el desarrollo de *software*.

A través de esta comparación, se pretende identificar los elementos más relevantes de cada marco, con el objetivo de adaptarlos y combinarlos en una propuesta metodológica práctica y aplicable al contexto específico del sector bancario costarricense. Este análisis comparativo no solo destacará las mejores prácticas a nivel global, sino que también fundamentará la estructura y las fases del modelo metodológico propuesto, garantizando su pertinencia.

Este enfoque garantizará que la propuesta final se base en experiencias y conocimientos validados a nivel internacional, adaptados cuidadosamente para su implementación exitosa en el entorno local. Además, se considerarán las limitaciones y oportunidades específicas del sector bancario costarricense, asegurando su relevancia y efectividad.

- **Métodos y técnicas de recolección de datos**
    - **Población**

La población meta para esta propuesta está compuesta por empleados desarrolladores de *software* de una organización perteneciente al sector bancario costarricense. Esta elección se fundamenta en los objetivos específicos de la propuesta, que buscan identificar y contrastar las mejores prácticas para el desarrollo de *software* seguro.

Sector: Organización del sector bancario con departamentos de desarrollo de *software* propio, enfocado en el sector público. Este es particularmente relevante debido a las limitaciones y reglamentaciones que deben aplicar por ser entidades públicas.

Tipo de empleados: La población incluirá tanto a colaboradores remotos como a aquellos que continúan trabajando en modalidad presencial en la entidad bancaria. Esto permitirá conocer si es posible recomendar prácticas distintas según su lugar de trabajo.

Niveles jerárquicos: Se considerará la inclusión de empleados de diferentes niveles jerárquicos, desde desarrolladores hasta personal de pruebas y líderes de equipo. Esto es esencial para obtener una visión integral de los equipos que desarrollan *software* en estas entidades.

    - **Fuentes de información**

La recolección de datos para esta investigación se realizará a través de diversas técnicas e instrumentos que permitirán fundamentar teóricamente el modelo metodológico propuesto. En

primer lugar, se llevará a cabo una revisión de literatura académica y normativas especializadas en desarrollo de software seguro. Esto permitirá identificar fuentes relevantes que aporten una sólida base teórica al estudio.

Adicionalmente, se realizará un análisis comparativo de distintos marcos metodológicos internacionales. Para ello, se utilizará una matriz comparativa estructurada, diseñada para identificar y contrastar las fortalezas y debilidades de cada modelo en relación con el contexto costarricense. Esta herramienta no solo facilitará la evaluación de los marcos existentes, sino que también permitirá la incorporación de los elementos más relevantes adaptados a las características y necesidades del sector bancario costarricense.

Asimismo, se recopilará información cualitativa a través de una encuesta a desarrolladores de *software*, personal de pruebas y líderes de equipo, tanto de modalidad presencial como remota. Esto proporcionará una perspectiva integral sobre las prácticas actuales y las percepciones de los profesionales en el área, enriqueciendo la propuesta metodológica con información de primera mano.

En última instancia, el enfoque en la revisión documental, el análisis comparativo y la recolección de datos cualitativos permitirá una comprensión profunda del estado actual y de las oportunidades para fortalecer las prácticas de desarrollo seguro en el sector bancario costarricense.

- **Técnicas de recolección de datos**

Durante el desarrollo de esta investigación, se utilizarán diversas técnicas e instrumentos para analizar y sustentar teóricamente el modelo metodológico propuesto, sin recurrir a la implementación práctica. A continuación, se presenta un resumen de las técnicas seleccionadas junto con los instrumentos específicos que se emplearán:

Análisis comparativo de modelos existentes: Se llevará a cabo un análisis comparativo entre diferentes marcos metodológicos internacionales como OWASP SAMM, SDL y NIST, evaluando sus características en relación con el contexto costarricense.

- Instrumentos:
  - Matriz comparativa estructurada, diseñada para identificar y contrastar las fortalezas, debilidades y aplicabilidad de cada modelo.
  - *Checklist* de criterios
    - Algunos posibles indicadores:
      - Soporte para capacitación en equipos pequeños.
      - Integración con marcos legales (por ejemplo, la Ley de Protección de Datos de Costa Rica).

Encuesta a desarrolladores de software, profesionales de QA y líderes de equipo: Se llevarán a cabo entrevistas a expertos en desarrollo de *software* seguro para identificar barreras y facilitadores que puedan influir en la adopción del modelo metodológico.

- Instrumentos:
    - Encuesta estructurada, diseñada para abordar temas clave relacionados con las prácticas y desafíos en el desarrollo de *software* seguro.
    - Consentimiento informado
        - Objetivo de la entrevista.
        - Confidencialidad y uso de datos.

Población de estudio: La población incluirá a personal de distintos niveles de experiencia en desarrollo de *software* que trabajen para una entidad financiera del sector bancario público en Costa Rica.

Criterios de selección:

- Experiencia: Elegir personal con al menos 1 año de experiencia en el área.
- Diversidad de departamentos: Incluir representantes de diferentes departamentos competentes al desarrollo de *software* para obtener variedad de opiniones.
- Áreas de especialización: Asegurarse de incluir personal enfocado en el desarrollo, liderazgo y pruebas de *software*.

Tamaño de la muestra: La muestra estuvo conformada por 25 participantes, seleccionados mediante un muestreo no probabilístico de tipo intencional. Esta decisión metodológica se fundamenta en el carácter exploratorio y aplicado de la investigación, cuyo objetivo no es realizar inferencias estadísticas a una población amplia, sino comprender el estado actual de las prácticas de desarrollo de *software* seguro desde la perspectiva de profesionales directamente involucrados en estos procesos.

Método de recolección de datos: Mediante la realización de un cuestionario a expertos para conocer su opinión sobre las prácticas de desarrollo seguro y los desafíos que enfrentan.

- **Mecanismos para la tabulación y el análisis**

Para garantizar un análisis riguroso y sistemático de la información obtenida a lo largo de la investigación, se implementarán los siguientes mecanismos de tabulación y análisis:

- Análisis descriptivo: La información recopilada a partir de la revisión documental y encuestas se organizará inicialmente para proporcionar un panorama claro de los marcos metodológicos y sus características principales.

- Matrices comparativas: Se elaborarán matrices comparativas estructuradas para evaluar y contrastar las características de los marcos metodológicos. Esta herramienta facilitará la identificación de similitudes y diferencias en términos de estructura, implementación, roles y métricas. Las matrices permitirán visualizar de manera clara qué elementos son más relevantes y aplicables al contexto local.

- Análisis cualitativo de contenido: Se hará uso de la herramienta Flourish para cargar un conjunto de datos que permita graficar los resultados obtenidos en la encuesta.

- Tabulación de resultados: Los datos cuantitativos y cualitativos se consolidarán en resúmenes, imágenes y tablas que faciliten la visualización clara de los hallazgos. Esto incluirá gráficos y diagramas que representen patrones y tendencias identificadas, facilitando la interpretación y presentación de resultados en la investigación.

- **Tabulación y análisis de la información**

Los datos obtenidos se analizarán para identificar patrones y tendencias que apoyen la propuesta de modelo propuesto.

El análisis de datos de esta investigación se enfocará en procesar información teórica proveniente de la revisión de literatura y el análisis comparativo de modelos. Seguidamente, se presentan las etapas clave del análisis.

Análisis descriptivo: Se organizará la información de los marcos metodológicos revisados, sus características principales y prácticas comunes en contextos similares al costarricense.

Análisis comparativo: Se elaborarán matrices comparativas para contrastar los modelos en términos de madurez, flexibilidad, cobertura organizacional y facilidad de adopción, entre otros aspectos. Este análisis ayudará a identificar patrones y a justificar la elección de elementos relevantes para el modelo propuesto.

Análisis de factibilidad teórica: Se evaluará la viabilidad del modelo propuesto, identificando barreras y facilitadores en función de los recursos y del nivel de madurez de los equipos de desarrollo en el sector bancario costarricense. Se analizarán patrones teóricos que permitan prever los desafíos en la adopción.

Identificación de patrones comunes: Se buscará, en los marcos revisados, patrones comunes y prácticas universales que puedan ser adaptadas al sector bancario local, ayudando a definir los componentes esenciales del modelo metodológico.

Presentación de conclusiones: Las conclusiones se basarán en las relaciones encontradas entre los modelos comparados, sintetizando hallazgos que justifiquen la construcción del modelo metodológico propuesto.

Este enfoque permitirá discernir patrones y relaciones clave para formular un modelo robusto y adaptado a la realidad del sector bancario costarricense.

Tabla 1
Relación entre objetivos, actividades e instrumentos de investigación

Objetivo	Actividades	Fuente de información	Instrumento	Resultados esperados
Analizar el estado actual de las prácticas de desarrollo de *software* seguro en una entidad del sector bancario público costarricense.	Realización de entrevistas a desarrolladores y expertos en seguridad de *software*.	Empleados de entidades bancarias.	Guía de entrevistas estructurada, ver anexo 1.	Recopilación de información cualitativa sobre la percepción de prácticas actuales en desarrollo seguro.
Analizar diferentes marcos metodológicos y prácticas existentes correspondientes al desarrollo de *software* seguro, con el fin seleccionar los elementos más relevantes de cada marco.	Análisis comparativo de marcos existentes (OWASP SAMM, SDL y NIST).	Literatura académica sobre marcos de desarrollo seguro.	Matriz comparativa estructurada, diseñada para identificar y contrastar las fortalezas, debilidades y aplicabilidad de cada modelo.	Un conjunto de matrices que resalten las fortalezas y debilidades de cada marco.
Elaborar un modelo para adaptar las prácticas de desarrollo de *software* seguro para una entidad pública del sector bancario costarricense.	Desarrollo de un modelo metodológico basado en los hallazgos de la investigación.	Resultados de entrevistas y análisis comparativo.	Propuesta formal del modelo metodológico.	Propuesta de un modelo metodológico adaptado a las necesidades del sector.
	Presentación de un documento de recomendaciones prácticas para la implementación del modelo.	Revisión de literatura sobre mejores prácticas.	Documento de recomendaciones.	Documento que detalle recomendaciones específicas para la implementación del modelo en la práctica.

CAPÍTULO II. ANÁLISIS DEL ESTADO ACTUAL DE LAS PRÁCTICAS DE DESARROLLO DE *SOFTWARE* SEGURO EN UNA ENTIDAD DEL SECTOR BANCARIO PÚBLICO COSTARRICENSE

En el contexto actual, donde las amenazas cibernéticas evolucionan constantemente, resulta fundamental evaluar cómo las instituciones financieras integran la seguridad en sus procesos de desarrollo de *software*. Con este fin, se realizó una encuesta, dirigida a profesionales de una entidad del sector bancario costarricense, durante el mes de agosto del año 2025, a un total de 25 personas, obteniendo la misma cantidad de respuestas. El objetivo de dicha encuesta consistió en analizar el estado actual de las prácticas de desarrollo de *software* seguro en una entidad del sector bancario público costarricense.

Si bien el tamaño de la muestra es acotado, los resultados obtenidos permiten identificar tendencias claras y consistentes que sirven como insumo para la construcción del modelo metodológico propuesto.

El estudio se diseñó con un enfoque mixto, combinando preguntas cuantitativas y cualitativas para capturar tanto datos medibles como perspectivas profundas de los participantes (Poth, 2023). Las preguntas se estructuraron en tres ejes principales: la percepción de la seguridad dentro de las organizaciones, las prácticas adoptadas en el ciclo de vida de desarrollo de *software* y los obstáculos que enfrentan los equipos para priorizar la ciberseguridad.

Uno de los aspectos clave explorados fue la valoración que los profesionales asignan a la seguridad en su trabajo diario. Mediante una escala Likert (Likert, 1932) del 1 al 5, los encuestados calificaron su importancia.

Además, la encuesta indagó sobre las fases específicas del desarrollo donde se aplican medidas de seguridad, identificando patrones y omisiones críticas. Estos datos cualitativos enriquecen el estudio, ofreciendo un panorama completo de los factores que dificultan la adopción de mejores prácticas en la industria.

Finalmente, el objetivo último de esta encuesta es servir como base para recomendaciones accionables, tanto para las instituciones bancarias como para futuras investigaciones académicas. Al entender cómo se implementa la seguridad hoy, se puede proponer estrategias más efectivas para cerrar las brechas identificadas y fortalecer los estándares del sector.

Este análisis no solo contribuye a la discusión sobre ciberseguridad en entornos financieros, sino que también establece un precedente para evaluaciones similares en otros contextos geográficos y organizacionales.

El cuestionario utilizado para recopilar los datos puede ser consultado en el anexo 1.

- Validación y confiabilidad del instrumento de encuesta

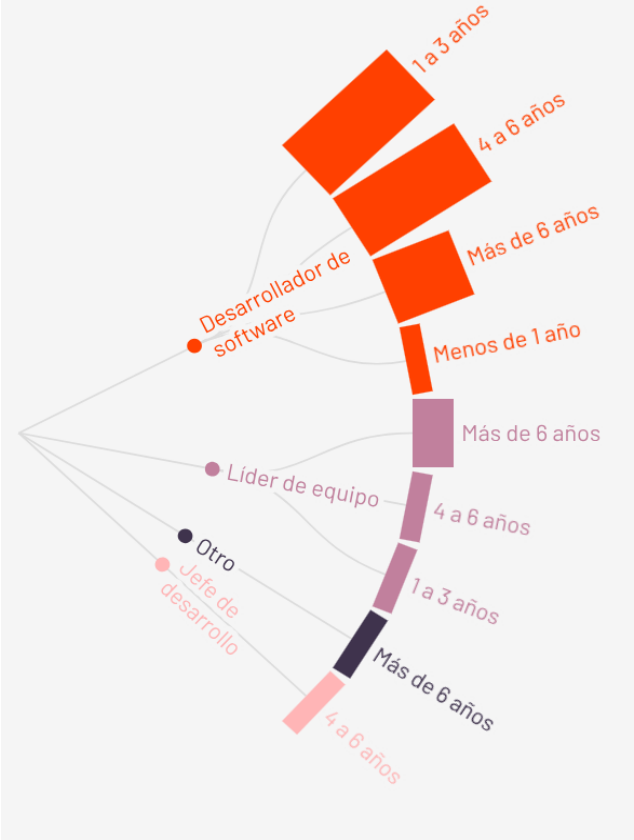
Con el fin de garantizar la validez y confiabilidad del instrumento de recolección de datos utilizado en esta investigación, se aplicaron diversas estrategias metodológicas. En primer lugar, se aseguró la validez de contenido mediante la alineación directa de las preguntas del cuestionario con los objetivos específicos de la investigación y los ejes conceptuales definidos en el marco teórico, particularmente aquellos relacionados con prácticas de desarrollo de *software* seguro, cultura organizacional y adopción de estándares internacionales.

Adicionalmente, el instrumento fue sometido a un juicio de expertos, en el cual profesionales con experiencia, principalmente en desarrollo de *software* del sector bancario costarricense, mediante un conversatorio, hicieron sugerencias y revisaron el cuestionario con el objetivo de evaluar la claridad, pertinencia y coherencia de las preguntas. A partir de esta revisión, se realizaron ajustes menores de redacción para mejorar la comprensión de los ítems.

- Información general del entrevistado

Esta sección está compuesta por dos preguntas. La primera fue: ¿Cuál es su posición en la entidad bancaria? Y la segunda: ¿Cuánto tiempo lleva trabajando en desarrollo de *software* en el sector bancario costarricense?, y se obtuvo los siguientes resultados:

Figura 1
Distribución de experiencia laboral



Fuente: Elaboración propia.

La Figura 1 muestra la distribución de experiencia laboral en desarrollo de *software* dentro de una entidad del sector bancario costarricense, agrupando las respuestas de las interrogantes antes mencionadas. Se analizan cuatro roles: desarrollador de *software*, líder de equipo, jefe de desarrollo y otros. De un total de 25 encuestados, el sector que más respuestas obtuvo fue el sector enfocado al desarrollo de *software*.

La mayoría de los desarrolladores de *software* tienen entre 1 y 3 años de experiencia, lo que sugiere una base técnica joven en el sector. En contraste, los jefes de desarrollo presentan una mayor concentración en la categoría de más de 6 años, lo que refleja una trayectoria profesional más consolidada en puestos de liderazgo.

Los líderes de equipo muestran una distribución más equilibrada entre las categorías de 1 a 3 años y 4 a 6 años, indicando que muchos han ascendido tras una experiencia intermedia. El grupo “Otro” presenta una dispersión amplia, lo que podría incluir perfiles mixtos o funciones no técnicas con distintos niveles de experiencia.

Los resultados evidencian una estructura jerárquica basada en la experiencia: a mayor antigüedad, mayor probabilidad de ocupar cargos de liderazgo. Esto puede ser útil para entender dinámicas de promoción interna y planificación de carrera en el sector bancario costarricense.

- Percepción sobre la importancia de la seguridad en el desarrollo

Esta nueva sección está compuesta por dos preguntas. Ante la primera consulta ¿qué tan importante considera la seguridad en el proceso de desarrollo de *software* en su entidad?, los resultados muestran, en la Figura 2, que el 85 % de los profesionales califican la seguridad como "Muy" o "Extremadamente importante" (escala 4-5). Solo un 8 % la considera "Poco importante".

Figura 2
Importancia de la seguridad en el desarrollo de software

Fuente: Elaboración propia.

  - Hallazgos clave:
    - Existe clara conciencia de la importancia estratégica de la seguridad.
    - Los profesionales con mayor experiencia (+6 años) muestran mayor énfasis en su valoración.

La Figura 3 revela que al ser consultados por ¿En qué fases del ciclo de desarrollo se integran actualmente las prácticas de seguridad?, los encuestados revelaron una implementación desigual de seguridad en las fases del ciclo de desarrollo. Las etapas de desarrollo/codificación (92 %) y pruebas (85 %) muestran mayor adopción, mientras que diseño (62 %) y despliegue (54 %) presentan cobertura insuficiente.

Figura 3

Fases de aplicación de prácticas de seguridad

Fuente: Elaboración propia.

Esta distribución refleja un enfoque predominantemente reactivo, donde la seguridad se aplica tarde en el proceso. La baja integración en diseño es particularmente preocupante, ya que las vulnerabilidades identificadas tardíamente son más costosas de corregir.

Los resultados explican parcialmente la brecha entre la alta valoración teórica de la seguridad (85 %) y su implementación práctica limitada en fases críticas.

- Prácticas de seguridad actuales

Esta sección está compuesta por tres preguntas. Inicialmente se les solicitó a los encuestados describir brevemente qué prácticas específicas de seguridad se utilizan en su equipo durante el desarrollo de *software*. La Tabla 2 contiene una categorización de la totalidad de las respuestas.

Tabla 2
Categorización de prácticas de seguridad reportadas por los encuestados

Categoría	***Frecuencia***	***Porcentaje***	***Ejemplos clave***
Gestión de accesos	6	24 %	Control por perfiles, autenticación con certificados, OAuth, JWT, autorización de API
Protección de datos	5	20 %	Encriptación de datos, validación de entradas, manejo de información sensible
Análisis de código	8	32 %	SonarQube, Fortify, revisiones de código, pruebas estáticas/dinámicas
Cumplimiento	2	8 %	Normativas PCI, estándares internos, procesos organizacionales
Arquitectura segura	3	12 %	DMZ, principio de mínimo privilegio, restricción de tecnologías
Prácticas culturales	1	4 %	Conocimiento experiencial, revisiones entre pares

Fuente: Elaboración propia.

Las respuestas revelan seis categorías principales de prácticas de seguridad:

Gestión de accesos: Incluye control mediante perfiles, autenticación con certificados digitales, *tokens* JWT/OAuth y validación estricta de permisos. Destaca el uso recurrente de mecanismos de autorización para API, para un total de 28%.

Protección de datos: Se reporta encriptación en reposo (bases de datos) y en tránsito, junto con validación de entradas para prevenir inyecciones. Un 32 % menciona específicamente el cifrado de información sensible.

Análisis de código: Herramientas como SonarQube y Fortify aparecen en el 58 % de respuestas, complementadas con revisiones manuales y pruebas de seguridad integradas en CI/CD.

Cumplimiento: Se reporta adhesión a normativas PCI y estándares internos, con documentación de procesos y controles de auditoría. Un 15% de las respuestas menciona explícitamente la importancia del cumplimiento regulatorio y la alineación con políticas organizacionales.

Arquitectura segura: Incluye diseño en capas, uso de DMZ y principios de mínimo privilegio. Solo el 18 % menciona explícitamente estas prácticas arquitectónicas.

Prácticas culturales: Incluye conocimiento experiencial compartido entre equipos, revisiones informales entre pares y compromiso colectivo con la seguridad. Solo un 8 % de los encuestados destacó estos aspectos, reflejando su baja formalización en las organizaciones.

- Hallazgo crítico:

Existe una clara predominancia de medidas técnicas reactivas (análisis de código, protección de datos) sobre prácticas preventivas en diseño. Esto concuerda con la baja integración de seguridad en fases tempranas detectada previamente.

- Brecha identificada:

Mientras el 85 % valora la seguridad como importante, solo el 22 % de las prácticas reportadas corresponden a estrategias proactivas (como *threat modeling* o seguridad en diseño).

Asimismo, se consultó: ¿Qué barreras o desafíos ha identificado en la implementación de prácticas de desarrollo seguro en su entidad? Y fueron claros al declarar que sus desafíos se clasifican en:

Tabla 3
Principales barreras en la implementación de prácticas de desarrollo seguro

Categoría	***Tipo de barrera***	***Porcentaje***
Factores organizacionales	Limitaciones de tiempo/presupuesto	23 %
	Falta de documentación/estándares	15 %
	Resistencia al cambio cultural	7 %
	Subtotal	45 %
Desafíos técnicos	Código heredado no seguro	18 %
	Herramientas obsoletas	12 %
	Falta de integración en CI/CD	5 %
	Subtotal	35 %
Brechas de conocimiento	Capacitación insuficiente	15 %
	Falta de especialización	5 %
	Subtotal	20 %

Fuente: Elaboración propia.

- Hallazgos clave

  El análisis revela una brecha crítica: mientras el 85 % valora teóricamente la seguridad, solo el 35 % invierte en soluciones concretas. El 65 % de las barreras prevenibles mediante gestión y formación persisten, con solo un 18 % de mejora reciente. Esto refleja una desconexión entre prioridad declarada y acción efectiva.

Los encuestados manifestaron, amparados en su experiencia, los factores que facilitan la adopción de prácticas de desarrollo seguro de *software*, entre las que destacan:

Tabla 4
Factores facilitadores para la adopción de prácticas de desarrollo seguro

Categoría principal	***Subcategorías***	***Porcentaje***
Soporte organizacional	Apoyo de alta dirección	18 %
	Cultura colaborativa DevSecOps	12 %
	Estándares y políticas claras	10 %
	Subtotal	40 %
Capacitación y concienciación	Formación continua	25 %
	Evaluación de conocimientos	10 %
	Subtotal	35 %
Automatización y herramientas	Revisiones de código automatizadas	15 %
	Herramientas integradas	10 %
	Subtotal	25 %

Fuente: Elaboración propia.

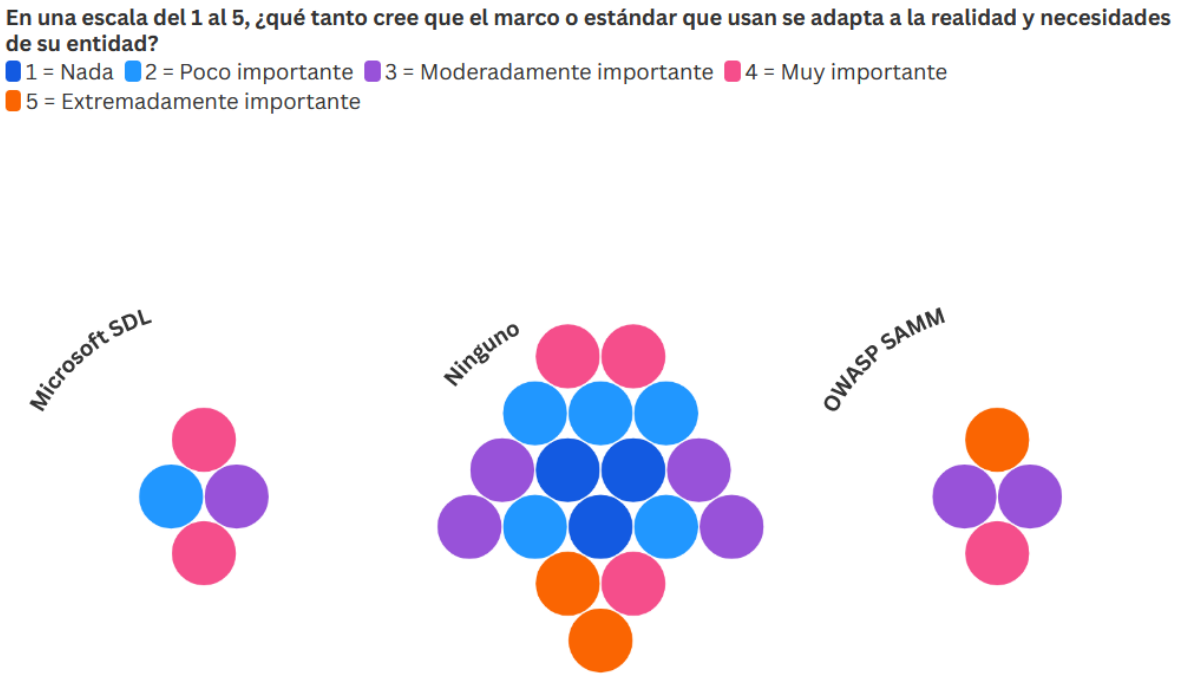
- Hallazgos clave
  - El 75 % de los facilitadores son no-técnicos (gestión + formación)
  - La alta dirección aparece como factor crítico (18 %)
  - Existe correlación positiva entre:
    - Automatización ↔ Reducción de errores
    - Capacitación ↔ Adopción de estándares
- Contraste con barreras

Aunque el 65 % de las barreras son prevenibles, los facilitadores revelan que las soluciones existen internamente, pero requieren mayor formalización mediante políticas y procesos estructurados.

- Conocimiento y uso de marcos de trabajo o estándares

Esta sección cuenta con dos preguntas fusionadas en la que se consultó sobre el uso de estándares internacionales para el apoyo en el proceso de desarrollo de *software* y el uso que se les da a estos en sus equipos. Los resultados se pueden apreciar en la Figura 4.

Figura 4
Uso de estándares internacionales



Fuente: Elaboración propia.

De la Figura 4, se puede deducir:

- Hallazgo principal

Los encuestados indicaron el uso de estándares como Microsoft SDL y OWASP SAMM. Sin embargo, la minoría de los encuestados califica que estos carecen de una adopción total ante sus realidades y necesidades, lo cual indica una desconexión entre los marcos teóricos y las necesidades prácticas de la entidad.

No se puede dejar pasar que la mayoría indicó que no usa ningún estándar que le apoye en sus labores de desarrollo de *software*.

- Relación con datos previos

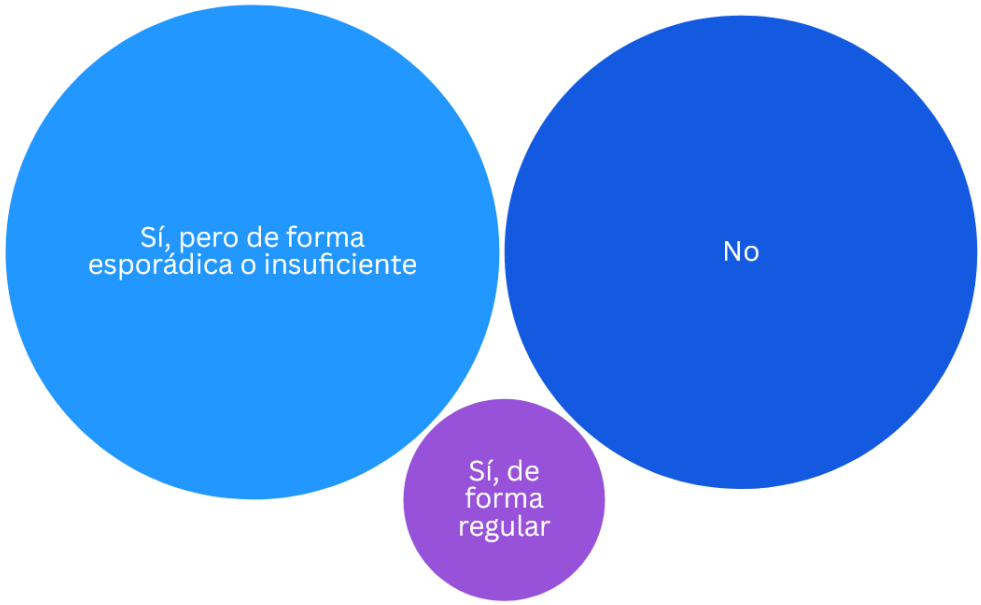
Este resultado explica parcialmente:

- La baja adopción de estándares (solo 30 % usa OWASP/ISO 27001).
- La dependencia de prácticas reactivas (identificadas en análisis anteriores).

- Implicaciones
  - Los marcos existentes no resuelven desafíos específicos del sector bancario costarricense.
  - Existe necesidad de:
    - Personalizar estándares internacionales.
    - Desarrollar guías contextualizadas.

- Capacitación y cultura organizacional

Figura 5
Frecuencia de capacitaciones



Fuente: Elaboración propia.

Esta sección se compone de dos preguntas. En la primera se les consultó sobre si reciben capacitación periódica en temas de seguridad informática relacionada con el desarrollo de *software*, y como es posible apreciar en la Figura 5, los encuestados manifestaron que no están siendo capacitados con la frecuencia que amerita la formación de un equipo como este, para

conseguir un excelente desempeño. Con 11 votos para el “No”, 12 para “Sí, pero de forma esporádica o insuficiente” y finalmente 2 votos para “Sí, de forma regular”.

La literatura consultada en este estudio y los resultados obtenidos en la Figura 5 coinciden en señalar la insuficiencia de la capacitación en seguridad dentro de los equipos de desarrollo. En los antecedentes se advierte que “uno de los mayores problemas en Costa Rica es la falta de concienciación y capacitación en seguridad entre los desarrolladores de software. Es crucial educar a los equipos en mejores prácticas de seguridad para reducir riesgos” (Adelyar & Norta, 2016).

Este planteamiento se confirma en los hallazgos empíricos, donde la Figura 5 muestra que la mayoría de los encuestados indicó no recibir capacitaciones regulares, limitándose a formación esporádica o insuficiente. Así, ambos enfoques reflejan una misma realidad: la falta de una estrategia estructurada de aprendizaje y actualización en seguridad, lo que debilita la consolidación de una verdadera cultura organizacional orientada al desarrollo de *software* seguro.

Figura 6
Alineación de la cultura organizacional

Fuente: Elaboración propia.

Por otra parte, los encuestados también indicaron que la cultura organizacional no está alineada con la importancia de la seguridad. Ante la consulta: ¿qué tan alineada está la cultura organizacional de su entidad con la importancia de la seguridad en el desarrollo de *software*?, un gran porcentaje de las respuestas indicaron que está poco o medianamente alineada (ver Figura 6).

La literatura consultada en este estudio señala que la falta de una cultura organizacional sólida en torno a la seguridad es uno de los principales obstáculos para la adopción de prácticas de desarrollo seguro. En particular, se menciona que “el sector bancario costarricense ha mostrado dificultades para integrar prácticas de desarrollo seguro en sus equipos, debido en gran parte a la falta de un modelo metodológico adaptado y a la ausencia de una cultura organizacional enfocada en la seguridad” (Ramírez, Aiello, & Lincke, 2020).

La figura 6 confirma este señalamiento al mostrar que casi la mitad de los encuestados percibe que la cultura organizacional de su entidad está poco o nada alineada con la seguridad en el desarrollo de *software*. Solo un 20 % considera que existe una alineación sólida, lo que refleja una desconexión entre la importancia declarada de la seguridad y las acciones concretas de las instituciones.

En conjunto, tanto la revisión teórica como los resultados empíricos coinciden en que la falta de alineación cultural constituye una brecha crítica. Mientras la literatura destaca la ausencia de una visión organizacional que priorice la seguridad desde el inicio, la evidencia práctica demuestra que esta carencia se traduce en equipos que operan con bajo compromiso estructural hacia la seguridad, debilitando su efectividad.

- Distribución de respuestas

Tabla 5
Alineación de la cultura organizacional con la importancia de la seguridad en el proceso de desarrollo de software

Nivel de alineación	***Porcentaje de respuestas***
5 - Totalmente alineada	8 %
4 - Bien alineada	12 %
3 - Alineación media	32 %
2 - Poco alineada	40 %
1 - Nada alineada	8 %

Fuente: Elaboración propia

- Hallazgos clave
  - Dominio de baja alineación**:** El 48 % (2+1) reporta poca o nula alineación, mientras solo el 20 % (5+4) percibe una alineación sólida.
  - Brecha crítica**:** Contrasta con el 85 % que considera la seguridad "importante" (datos previos), evidenciando una desconexión cultura-valores.
- Patrones relevantes

        - El 40 % en "Poco alineada" refleja:
            - Falta de priorización operativa
            - Discrepancias entre discurso y acciones
        - Solo la quinta parte de los equipos opera con alineación efectiva
    - Relación con otros resultados
        - Explica por qué:
            - Solo el 62 % integra seguridad en diseño
            - El 70 % no usa estándares internacionales
        - Corrobora barreras como:
            - Resistencia al cambio (35 %)
            - Falta de soporte gerencial (18 %)
- Sugerencias

Para esta última sección, se preguntó por recomendaciones para mejorar la adopción de prácticas de desarrollo de *software* seguro en su entidad. Las respuestas fueron categorizadas y arrojaron los siguientes resultados:

    - Categorización de propuestas
        - Capacitación continua (40 %):
            - Formación técnica específica
            - Demostraciones de vulnerabilidades
            - Actualización en nuevas tecnologías
        - Implementación de marcos (30 %):
            - Adopción de estándares (OWASP SAMM)
            - Modelos adaptados a la organización
            - Guías claras por tipo de proyecto
        - Cambios organizacionales (20 %):

        - Involucramiento activo del equipo de seguridad
        - Cultura de responsabilidad compartida
        - Soporte de la gerencia
    - Modernización técnica (10 %):
        - Migración de sistemas *legacy*
        - Automatización de pruebas
    - Hallazgos clave
        - El 70 % de las soluciones son no-técnicas (formación + gestión)
        - Existe consenso en:
            - Necesidad de capacitación práctica (no teórica)
            - Adopción de marcos estandarizados
        - Solo el 10 % menciona modernización, pese a ser barrera clave
    - Relación con hallazgos previos
      Estas recomendaciones abordan directamente:
        - La baja alineación cultural (48 % poco/nada alineada)
        - Uso limitado de estándares (70 % no los aplica)
        - Brecha en formación identificada como barrera principal

- Comentarios finales

Finalmente, se solicitó a los encuestados que realizaran un comentario final sobre prácticas de desarrollo seguro o seguridad informática en el sector bancario costarricense. Los comentarios revelan una clara paradoja. Si bien existe consenso sobre la importancia crítica de la seguridad en el sector bancario, su implementación sigue siendo inconsistente e insuficiente.

    - Áreas de oportunidad
        - La implementación sistemática de marcos
        - Mayor inversión en automatización de seguridad
        - Desarrollo continuo de competencias técnicas

- Brechas identificadas
  Los profesionales destacan tres problemas clave:
  - Adopción limitada de estándares internacionales
  - Uso irregular de herramientas básicas de protección
  - Cultura reactiva en lugar de preventiva

El sector muestra conciencia del problema, pero avanza lentamente en soluciones. Urge transformar el discurso en acción mediante: liderazgo comprometido, presupuestos dedicados y metodologías probadas.

La seguridad efectiva requiere más que herramientas, exige un cambio cultural donde cada colaborador asuma su rol protagónico en la protección de datos. El camino es claro, pero requiere decisión para recorrerlo.

Los resultados mostraron, por ejemplo, que mientras algunas etapas como la de pruebas, reciben atención, otras como el diseño inicial a menudo quedan desatendidas, exponiendo potenciales vulnerabilidades desde las primeras etapas de los proyectos.

Las respuestas abiertas proporcionaron valiosas perspectivas sobre los desafíos prácticos, desde limitaciones de tiempo hasta la falta de capacitación especializada.

CAPÍTULO III. ANÁLISIS DE DIFERENTES MARCOS METODOLÓGICOS Y PRÁCTICAS EXISTENTES CORRESPONDIENTES AL DESARROLLO DE *SOFTWARE* SEGURO

En el contexto actual de creciente complejidad y sofisticación de las amenazas informáticas, las organizaciones requieren marcos de trabajo estructurados que integren la seguridad en el desarrollo de *software* y en la gestión del riesgo organizacional. En este capítulo se presentan y analizan tres marcos ampliamente reconocidos, en el área del desarrollo de *software* seguro: OWASP SAMM, Microsoft SDL y los marcos del NIST, CSF y el SSDF.

La elección de estos marcos responde a tres razones fundamentales. Primero, su reconocimiento institucional: OWASP SAMM está respaldado por una comunidad global sin fines de lucro enfocada en la seguridad de aplicaciones. SDL es el modelo adoptado por Microsoft, líder mundial en ingeniería de *software*. Los marcos NIST poseen un respaldo gubernamental y normativo que los convierte en estándares de referencia internacional (OWASP, 2020; Microsoft, 2024; NIST, 2024a).

Segundo, su cobertura complementaria: SAMM se orienta a medir la madurez del proceso de desarrollo seguro, SDL detalla prácticas operativas dentro del ciclo de vida del *software* y los marcos del NIST ofrecen un enfoque integral de ciberseguridad y gobernanza.

Tercero, su adaptabilidad: los tres pueden implementarse en organizaciones de diferentes tamaños y niveles de madurez, lo que los hace idóneos para entornos reales y heterogéneos.

- OWASP SAMM

OWASP SAMM, es un marco abierto desarrollado por *Open Worldwide Application Security Project* (OWASP) con el objetivo de ayudar a las organizaciones a formular e implementar una estrategia de seguridad de *software* adaptada a sus riesgos y objetivos específicos (OWASP, 2020). Según su documentación oficial, proporciona una vía eficaz y medible para que las organizaciones analicen y mejoren su postura de seguridad en *software* (OWASP, 2020, p. 3).

La versión 2.0 del modelo estructura las prácticas de seguridad en cinco funciones de negocio: *Governance, Construction, Verification, Deployment* y *Operations*. Cada función contiene tres prácticas específicas, sumando un total de quince. Cada práctica se evalúa según dos flujos (*streams*) y cuatro niveles de madurez (de 0 a 3) con criterios de calidad definidos que permiten medir objetivamente el progreso. Las evaluaciones de madurez permiten identificar brechas (*gap analysis*) priorizar acciones y establecer planes de mejora continua mediante guías oficiales de evaluación y autoevaluación (OWASP, 2020).

Entre sus ventajas destacan:

- Flexibilidad metodológica, pues puede aplicarse en entornos ágiles, DevOps o tradicionales.

- Carácter medible, al incluir niveles de madurez verificables.

- Apoyo comunitario, lo que asegura actualización constante y acceso abierto.

OWASP SAMM se elige por su utilidad como marco de madurez adaptable que permite diagnosticar, mejorar y monitorear el progreso de la seguridad del *software* en cualquier tipo de organización.

- Microsoft SDL (*Security Development Lifecycle*)

Microsoft SDL es el proceso definido por Microsoft para integrar la seguridad en todas las fases del desarrollo de *software*, con el fin de reducir vulnerabilidades y mejorar la resiliencia del producto (Microsoft, 2024). Microsoft SDL se basa en la experiencia práctica adquirida en el desarrollo de productos como Windows, Azure y Office.

De acuerdo con la documentación oficial, Microsoft SDL abarca siete fases principales: entrenamiento, requisitos, diseño, implementación, verificación, liberación y respuesta (Microsoft, 2024). En cada fase se establecen prácticas obligatorias, tales como el modelado de amenazas, la revisión de diseño, el análisis estático y dinámico del código, pruebas para detectar vulnerabilidades, la validación criptográfica y la revisión de la seguridad antes del despliegue.

Microsoft SDL destaca por su claridad en la forma de actuar, ofreciendo pasos claros y herramientas concretas para que los equipos incorporen la seguridad en su proceso de desarrollo. Su enfoque es práctico, orientado a la reducción de errores comunes y a la prevención de vulnerabilidades desde etapas tempranas (Microsoft, 2024).

Entre sus fortalezas sobresalen la claridad en las actividades, la evidencia empírica de su efectividad y la facilidad de adopción en equipos con madurez intermedia o baja. Sin embargo, a diferencia de OWASP SAMM, no establece niveles de madurez, sino que se enfoca en la ejecución de controles específicos en cada etapa del ciclo (OWASP, 2020; Microsoft, 2024).

Por su naturaleza, Microsoft SDL, se selecciona como marco de referencia operativo que complementa los modelos de madurez como OWASP SAMM, proporcionando guías prescriptivas y comprobadas para implementar prácticas seguras en proyectos reales.

- Marco NIST CSF y SSDF

NIST ofrece una guía estructurada para la gestión integral del riesgo de ciberseguridad. Su versión 2.0 define seis funciones principales: *Govern, Identify, Protect, Detect, Respond* y *Recover*, que representan las actividades esenciales del ciclo de vida de la ciberseguridad (NIST, 2024a).

El Cybersecurity *Framework* (CSF) se compone de tres elementos: el *Core* (que detalla las funciones y categorías de resultados) los *Profiles* (que describen el estado actual y el deseado de la organización) y los *Tiers* (niveles de implementación que miden el grado de integración y formalidad de las prácticas) (NIST, 2024a). Su propósito es proporcionar un lenguaje común para mejorar la gestión del riesgo de ciberseguridad a nivel institucional.

El *Secure Software Development Framework* (SSDF) describe prácticas fundamentales para integrar la seguridad en el desarrollo de *software*, independientemente de la metodología utilizada (NIST, 2024b). El SSDF define qué debe realizarse (por ejemplo, definir requisitos de seguridad, verificar el código y gestionar la cadena de suministro), pero deja abierta la implementación, enlazando a estándares externos como OWASP SAMM o Microsoft SDL (NIST, 2024b).

El NIST fue seleccionado porque aporta un marco de gobernanza y cumplimiento normativo ampliamente reconocido, integrando tanto la gestión estratégica de riesgos (CSF) como las prácticas técnicas de desarrollo seguro (SSDF). De este modo, ofrece una visión integral que abarca desde la planeación organizacional hasta la práctica de codificación.

- Comparación entre los marcos

La siguiente es una comparación entre los marcos OWASP SAMM, Microsoft SDL y NIST CSF/SSDF considerando sus características más relevantes:

Tabla 6
Comparación entre los marcos OWASP SAMM, Microsoft SDL y NIST CSF/SSDF

Criterio	***OWASP SAMM***	***Microsoft SDL***	***NIST CSF / SSDF***
Enfoque principal	Madurez y mejora continua de la seguridad en el desarrollo de *software*	Integración de prácticas de seguridad en cada fase del ciclo de desarrollo	Gestión integral del riesgo (CSF) y desarrollo seguro (SSDF)
Nivel de prescriptividad	Intermedio: define prácticas y niveles de madurez	Alto: prescribe actividades específicas y obligatorias	Bajo a medio: establece resultados esperados y enlaza con guías externas
Medición/madurez	Sí, niveles 0 - 3 con criterios de calidad	No, se centra en prácticas y controles	CSF define *Tiers* de implementación; SSDF no define niveles de madurez
Flexibilidad metodológica	Alta: se adapta a ágil, DevOps o cascada	Media: adaptable, pero más estructurado	Alta: independiente de metodología o tecnología
Cobertura organizacional	Centrada en el desarrollo de *software*, con componentes de gobernanza	Foco en procesos de ingeniería y desarrollo	Amplia: incluye gobernanza, riesgo, detección y recuperación
Reconocimiento y adopción	Amplio en la comunidad de seguridad de aplicaciones	Amplio en el ecosistema Microsoft	Muy alto en sectores regulados y gobierno
Complementariedad	Mide la madurez de las prácticas	Define cómo ejecutar las prácticas	Define qué resultados lograr y cómo integrarlos a nivel organizacional

La sinergia entre los tres marcos se evidencia en su compatibilidad:

- OWASP SAMM puede mapearse con NIST SSDF, ya que ambos definen prácticas de desarrollo seguro orientadas a la mejora continua (OWASP, 2020; NIST, 2024b).
- Microsoft SDL puede verse como un nivel operativo dentro de SAMM, ofreciendo prácticas concretas que permiten alcanzar niveles de madurez específicos (Microsoft, 2024).
- NIST CSF proporciona la capa de gobernanza estratégica que contextualiza la aplicación de SAMM y SDL en la gestión global de la seguridad (NIST, 2024a).

El análisis comparativo de los diferentes marcos metodológicos y prácticas existentes en el desarrollo de *software* seguro pone en evidencia que, si bien existen estándares internacionales ampliamente reconocidos, ninguno de ellos se adapta de manera completa y directa a las particularidades del sector bancario costarricense. Esto evidencia la necesidad de construir un modelo metodológico personalizado que combine los elementos más relevantes y efectivos de cada marco, ajustándolos a la realidad normativa, cultural y operativa local.

Este modelo debe fomentar la integración de prácticas de seguridad desde las primeras etapas del ciclo de vida del *software*, superando el enfoque reactivo centrado en la aplicación de correcciones posteriores. Asimismo, es fundamental que acompañe la implementación técnica con un proceso continuo de capacitación y concienciación dentro de los equipos de desarrollo, fortaleciendo así una cultura organizacional orientada a la seguridad y la ética en la gestión de la información financiera.

La adopción de un modelo flexible y escalable permitirá a las entidades bancarias no solo reducir significativamente los riesgos asociados a ciberataques, sino también fortalecer la confianza pública en sus sistemas y servicios. Además, esta iniciativa puede servir como un referente para otros contextos similares en Costa Rica, contribuyendo al fortalecimiento de la resiliencia digital del sector financiero en el país.

En definitiva, la construcción de un modelo adaptado y coherente con las necesidades locales representa una vía estratégica para elevar los estándares de seguridad en el desarrollo de *software* en el sector bancario y garantizar la protección efectiva de los datos sensibles, impulsando la estabilidad y confiabilidad del sistema financiero costarricense.

Con el propósito de profundizar en el análisis comparativo de los marcos OWASP SAMM, Microsoft SDL y NIST SSDF y superar una evaluación meramente descriptiva, se incorpora a continuación un análisis de brechas. Este análisis permite identificar las diferencias existentes entre las capacidades que ofrecen dichos marcos y las necesidades específicas del sector bancario público costarricense.

La matriz de brechas se construye a partir de dimensiones críticas identificadas en el diagnóstico del estado actual de las prácticas de desarrollo de *software* seguro, considerando aspectos normativos, culturales, técnicos y organizacionales. De esta forma, se evidencian las

limitaciones, vacíos y oportunidades de adaptación de cada marco en el contexto local, lo cual fundamenta la necesidad de proponer un modelo metodológico ajustado a la realidad del sector bancario costarricense.

Tabla 7
Matriz de brechas de los marcos de desarrollo de software seguro

Dimensión crítica	***OWASP SAMM***	***Microsoft SDL***	***NIST SSDF***
Adaptación a regulaciones costarricenses	Brecha media: requiere contextualización normativa local	Brecha alta: orientado a entornos corporativos privados	Brecha baja: ajustable con marcos regulatorios
Facilidad de adopción en entidades públicas	Media: flexible, pero exige madurez	Baja: alto nivel de formalismo	Media-alta: adaptable por fases
Enfoque cultural y formativo	Alta cobertura	Cobertura limitada	Cobertura parcial
Requerimientos técnicos y de herramientas	Medios	Altos	Medios
Escalabilidad y madurez progresiva	Alta	Media	Alta
Integración con metodologías ágiles	Alta	Media	Media
Adecuación al contexto bancario público costarricense	Parcial	Limitada	Parcial/alta

El análisis de brechas evidencia que, si bien los marcos OWASP SAMM, Microsoft SDL y NIST SSDF ofrecen fundamentos sólidos para el desarrollo de *software* seguro, ninguno de ellos responde de forma integral y directa a las particularidades del sector bancario público costarricense.

OWASP SAMM destaca por su flexibilidad y enfoque en madurez progresiva; sin embargo, presenta brechas en cuanto a alineación explícita con regulaciones locales y estructuras formales del sector público. Microsoft SDL, aunque robusto desde el punto de vista técnico, evidencia limitaciones significativas para su adopción en organizaciones con restricciones presupuestarias y alta carga regulatoria, como las entidades bancarias estatales. Por su parte, NIST SSDF ofrece una mejor alineación con entornos regulados, pero requiere un esfuerzo de contextualización para integrarse plenamente con prácticas ágiles y realidades operativas locales.

Estas brechas justifican la necesidad de un modelo metodológico adaptado, que integre los elementos más relevantes de cada marco, mitigando sus limitaciones y alineándolos con el contexto normativo, cultural y operativo del sector bancario costarricense.

CAPÍTULO IV. PROPUESTA DE MODELO PARA ADAPTAR LAS PRÁCTICAS DE DESARROLLO DE *SOFTWARE* SEGURO EN UNA ENTIDAD PÚBLICA DEL SECTOR BANCARIO COSTARRICENSE

1. Principios rectores del modelo

La propuesta de este modelo se fundamenta en un conjunto de principios rectores que orientan la adopción sistemática, consistente y sostenible de prácticas de desarrollo de *software* seguro. Estos principios sintetizan las bases conceptuales y los enfoques más robustos presentes en NIST SSDF (SP 800-218), OWASP SAMM v2 y Microsoft SDL, integrándolos con las particularidades regulatorias, operativas y culturales del sector financiero costarricense.

Estos principios constituyen el marco normativo y estratégico desde el cual se articulan los procesos, actividades, roles y métricas del modelo, asegurando coherencia con las necesidades de seguridad, cumplimiento, eficiencia y resiliencia institucional.

1.1 Seguridad como responsabilidad compartida

La seguridad del *software* no es exclusiva del equipo de ciberseguridad o del área de desarrollo. Es una responsabilidad transversal que involucra a todos los participantes del ciclo de vida del *software*.

Este principio se fundamenta en:

- NIST SSDF, que enfatiza la necesidad de involucrar a equipos mixtos, incluyendo desarrollo, infraestructura, riesgo, proveedores y gestión.
- Microsoft SDL, que establece que arquitectos, desarrolladores, personal para pruebas, PMO y líderes de producto comparten tareas de seguridad.
- OWASP SAMM, que distribuye prácticas en dominios organizacionales, de gobernanza, diseño, implementación y verificación.

En el contexto costarricense, este principio es crítico porque los bancos públicos suelen operar bajo estructuras rígidas y roles muy delimitados.

1.2 Enfoque *Secure by Design, by Default and by Deployment*

El modelo adopta los pilares del SD3+C del SDL de Microsoft (*Secure by Design, Secure by Default, Secure in Deployment* + *Communication*), los cuales se articulan con:

- Énfasis de NIST SSDF en integrar seguridad desde etapas tempranas (*shift-left*).
- Práctica de OWASP SAMM de incorporar seguridad en diseño, construcción y despliegue.

Este principio establece que la seguridad debe incorporarse desde la concepción del sistema, mantenerse por defecto en sus configuraciones y ser asegurada en los procesos de implementación y operación.

En una organización bancaria, esto reduce la superficie de ataque en aplicaciones que procesan información crítica financiera, transaccional y personal.

1.3 Minimización del riesgo y enfoque basado en amenazas

Todo desarrollo debe orientarse a reducir riesgos reales y potenciales mediante:

- Modelado de amenazas sistemático (Microsoft SDL).
- Identificación de debilidades y vulnerabilidades (NIST SSDF).
- Priorización de acciones basada en criticidad (OWASP SAMM).

El sector bancario enfrenta amenazas avanzadas, persistentes y motivadas económicamente (fraude, intrusión, suplantación, manipulación de datos), por lo que el modelo propone:

- Priorizar activos críticos.
- Realizar modelado de amenazas obligatorio para sistemas sensibles.
- Gestionar vulnerabilidades de manera continua.

1.4 Trazabilidad y auditabilidad

Debido a los estrictos requisitos regulatorios del sistema financiero costarricense (SUGEF, Ley 8968, normativa interna), las prácticas de seguridad deben ser documentables, trazables y demostrables, ante auditorías internas y externas.

Este principio se alinea con:

- Prácticas de evidencia y documentación requeridas por NIST SSDF.
- El dominio de *Governance* de OWASP SAMM.
- Los artefactos generados en cada fase del Microsoft SDL.

1.5 Mejora continua y madurez incremental

El modelo reconoce que las organizaciones bancarias tienen diferentes niveles de madurez en desarrollo seguro. Por ello incorpora:

- Capacidad de madurez progresiva de OWASP SAMM.

- Orientación a procesos repetibles del Microsoft SDL.
- Enfoque adaptativo del NIST SSDF.

Esto implica que:

- No todas las prácticas se implementan simultáneamente.
- Se definen niveles de avance medibles.
- Se priorizan actividades de máximo impacto y menor costo inicial.
- Se establece un *roadmap* hacia niveles superiores de madurez.

1.6 Automatización como eje para la eficiencia y la reducción del error humano

Inspirado por:

- Los principios de automatización de OWASP SAMM en *Secure Build*.
- Las recomendaciones de NIST sobre CI/CD seguro.
- Automatización de pruebas, escaneo y validación en SDL.

Este principio orienta a:

- Automatizar compilación, escaneo, pruebas y despliegues.
- Reducir la dependencia excesiva de validaciones manuales.
- Disminuir costos y tiempos asociados al aseguramiento.

1.7 Comunicación transparente y gestión del cambio cultural

La implementación de un modelo de desarrollo seguro requiere transformar, mentalidades, hábitos, incentivos y mecanismos de liderazgo. Este principio se fundamenta principalmente en el SDL (Comunicaciones) y en el pilar *Governance* de SAMM.

Para el sector bancario, acostumbrado a culturas de bajo cambio, este principio es indispensable y propone:

- Campañas de sensibilización.
- Comunicación ejecutiva clara.
- Incentivos al cumplimiento.
- Incorporación de métricas de seguridad en el desempeño.

1.8 Protección del consumidor financiero y cumplimiento normativo

Dado que en el sector bancario costarricense se manejan datos personales sensibles, transacciones financieras, autenticación robusta e información sometida a secreto bancario, el modelo debe garantizar:

- Alineamiento con leyes locales (Ley de Protección de Datos, normativa bancaria).
- Protección de la privacidad (PD3+C del SDL).
- Mecanismos de detección temprana de fraude y manipulación.

Este principio justifica el enfoque en seguridad como condición de confianza pública.

2. Estructura del modelo

La estructura del modelo metodológico integrado define la arquitectura conceptual y operativa que permitirá articular los tres marcos seleccionados, en una guía en capas, coherente, aplicable y adaptable al contexto del sector bancario costarricense.

2.1 Capa estratégica: gobernanza y dirección del programa

Esta capa establece los lineamientos de alto nivel que permitirán que el modelo sea viable, sostenible y medible. Está orientada a la toma de decisiones y asegura que la seguridad forme parte de los objetivos institucionales del banco.

- Componentes:
    - Política institucional de desarrollo seguro

Define los principios, responsabilidades, obligatoriedad y alcance del programa de desarrollo seguro. Integra elementos del dominio *Governance* de SAMM y los prerrequisitos de capacitación del SDL.

- Comité de seguridad del ciclo de vida del *software*

Órgano responsable de la supervisión, priorización y revisión de métricas, compuesto por TI, desarrollo, seguridad, riesgo operativo y auditoría interna.

- Modelo de madurez

Basado principalmente en OWASP SAMM, que permite evaluar el estado inicial, priorizar inversiones y definir rutas de mejora continua.

2.2 Capa de procesos: integración con el ciclo de vida del desarrollo

Esta capa representa el núcleo del modelo, donde se integran las prácticas de seguridad dentro de las fases tradicionales del SDLC, según los lineamientos convergentes del NIST SSDF, el enfoque incremental del SAMM y la secuenciación clara del Microsoft SDL.

- Fases del ciclo con sus objetivos y prácticas:
  - Fase de requisitos y planificación

    Objetivo: Incorporar la seguridad desde la concepción del proyecto.

    Refuerza la necesidad detectada de incorporar la seguridad antes del desarrollo, lo cual los equipos valoraron como una de las principales ausencias actuales.

    - Prácticas integradas:
      - Definición de requisitos de seguridad y cumplimiento (SSDF: PW.1 - *Prepare the Organization*; SDL: *Requirements Phase*).
      - Análisis de riesgos inicial del producto.
      - Definición de criterios de aceptación de seguridad.
  - Fase de diseño

    Objetivo: Garantizar que la arquitectura contemple mecanismos de protección, control y resiliencia.

    La criticidad del sector financiero obliga a contemplar ataques comunes, fraude, manipulación de transacciones y riesgos reputacionales.

    - Prácticas integradas:
      - Modelado de amenazas (SDL: *Threat Modeling*; SSDF: PW.3).
      - Diseño seguro basado en principios como mínimo privilegio, reducción de superficie y defensa en profundidad.
      - Validación del diseño mediante revisiones formales entre equipos.
  - Fase de implementación

    Objetivo: Asegurar que el código desarrollado cumpla con estándares de seguridad y que las dependencias externas no introduzcan riesgos.

Considera el nivel actual de automatización y madurez observado en la investigación, permitiendo escalabilidad según capacidades reales del banco.

- Prácticas integradas:
  - Revisión de código (SAMM: *Implementation - Defect Management*).
  - Uso de SAST/DAST automatizado (SDL: *Implementation*).
  - Gestión segura de dependencias (SAMM: *Secure Build*; SSDF: PW.4).

- Fase de verificación

  Objetivo: Validar que el producto cumple con los requisitos de seguridad antes de pasar a producción.

  - Prácticas integradas:
    - Pruebas de seguridad internas (*penetration testing, fuzzing,* análisis dinámico).
    - Validación independiente por el equipo de Seguridad o un tercero (SSDF: RV.x - *Review and Verify*).
    - Evaluación del cumplimiento de los criterios definidos en fases previas.

- Fase de liberación y despliegue

  Objetivo: Garantizar que el *software* liberado sea seguro, trazable y gestionable una vez puesto en operación.

  - Prácticas integradas:
    - Aprobación formal por Seguridad (SDL: *Final Security Review*).
    - Validación de integridad de artefactos (SSDF: PS.x – *Protect Software*).
    - Gestión segura de secretos y configuraciones (SAMM: *Secure Deployment*).

- Fase posdespliegue

  Objetivo: Mantener la seguridad del sistema durante toda su vida útil.

- Prácticas integradas:
  - Monitoreo activo de vulnerabilidades.
  - Gestión de parches con priorización basada en riesgo.
  - Retroalimentación al inicio del ciclo para mejoras continuas (SAMM: *Governance - Strategy & Metrics*).

2.3 Capa de soporte: cultura, capacitación y herramientas

Es la base habilitadora del modelo. Sin estas capacidades, la implementación no sería sostenible.

- Componentes:
  - Capacitación continua

    Basada en prerrequisitos del SDL, adaptada a roles:
    - Desarrolladores → codificación segura.
    - QA → pruebas de seguridad.
    - DevOps → CI/CD seguro.
    - Arquitectos → modelado de amenazas.
  - Herramientas de soporte

    Incluye: SAST, DAST, SCA, repositorios, control de versiones, herramientas de modelado de amenazas y CI/CD seguro.
  - Cultura organizacional

    Promoción de la responsabilidad compartida, métricas visibles, retroalimentación continua e incentivos a la calidad del código.

2.4 Capa de métricas y evaluación continua

Cada dominio del modelo debe medir su desempeño para verificar mejoras y demostrar resultados al banco.

- Tipos de métricas:
  - Cumplimiento: % de proyectos que incluyen modelado de amenazas.
  - Calidad: reducción de incidentes o vulnerabilidades detectadas tarde.

- Madurez: niveles SAMM para cada práctica.
- Eficiencia: tiempo promedio de reparación de vulnerabilidades.

Estas métricas pueden integrarse en un tablero mensual revisado por el comité de seguridad del SDLC.

3. Arquitectura del modelo

Este apartado presenta la arquitectura del modelo propuesto. El objetivo del marco es proporcionar una ruta clara y adaptable para que los equipos de desarrollo adopten prácticas seguras dentro del SDLC institucional, sin interrumpir su operación actual y elevando progresivamente el nivel de madurez.

3.1 Estructura general del modelo

La estructura general del modelo metodológico propuesto se organiza en ocho dominios, distribuidos de manera coherente con las cuatro capas definidas previamente: estratégica, procesos, soporte y métricas. Cada dominio articula un conjunto de prácticas provenientes de OWASP SAMM, Microsoft SDL y NIST SSDF/CSF, integradas y contextualizadas al entorno operativo y normativo del sector bancario público costarricense.

A continuación, se presenta la estructura completa del modelo:

- ***Dominio 1. Gobernanza y dirección del programa***

Define políticas, roles, responsabilidades y modelo de gobierno.
Incluye: selección de estándares, establecimiento de objetivos, gestión de riesgos, definición del apetito de riesgo.

- ***Dominio 2. Gestión del ciclo de vida del desarrollo seguro (SSDLC)***

Establece cómo se integra la seguridad en todas las fases del desarrollo: definición de requisitos, diseño seguro, construcción, pruebas, despliegue y mantenimiento.
Incluye: modelos de amenazas, validación de requisitos, prácticas *Secure by Design*, revisión segura del diseño.

- ***Dominio 3. Gestión de requisitos de seguridad en el SDLC***

Contempla actividades técnicas y automatizadas para identificar y remediar fallas.
Incluye: SAST, DAST, SCA, *fuzzing*, análisis de configuración, escaneo de contenedores, pruebas continuas en CI/CD.

- ***Dominio 4. Gestión de incidentes y respuesta segura***

Define procedimientos y capacidades para la detección, notificación, contención y recuperación frente a incidentes.

Incluye: *playbooks*, coordinación con CSIRT interno, retroalimentación al ciclo SSDLC.

- ***Dominio 5. Gestión de proveedores y artefactos externos***

Garantiza que *software* de terceros, librerías, API y componentes externos cumplan con requisitos de seguridad.

Incluye: *software supply chain*, verificación SBOM, políticas de uso de librerías.

- ***Dominio 6. Cultura, formación y concienciación***

Abarca programas de capacitación, actividades de sensibilización, cultura de transparencia y prácticas colaborativas DevSecOps.

Incluye: entrenamientos obligatorios, laboratorios seguros, métricas de aprendizaje, programas de *champions*.

- ***Dominio 7. Herramientas, automatización y soporte tecnológico***

Estandariza herramientas, define lineamientos de automatización, integra seguridad en *pipelines* CI/CD y gestiona repositorios.

Incluye: *pipelines* seguros, escaneo automático, repositorios confiables, reglas de calidad.

- ***Dominio 8. Métricas, auditoría y mejora continua***

Define indicadores, auditorías, mecanismos de revisión y rutas de madurez.

Incluye: KPI, métricas de defectos, métricas de cobertura de prácticas, cumplimiento de SDL y SSDF.

3.2 Mapa de integración entre los dominios del modelo y los estándares

Tabla 8
Mapa de integración entre los dominios del modelo y los estándares

Dominio	***NIST SSDF***	***OWASP SAMM v2***	***Microsoft SDL***
1. Gobernanza y dirección del programa	*Governance* (*Strategy & Metrics, Policy & Compliance*)	SDL – *Requirements, Training, Governance*	SSDF PO.1, PO.2, PO.3 / CSF GOV
2. Gestión del ciclo de vida del desarrollo seguro (SSDLC)	*Design & Implementation* (*Threat Assessment, Secure Architecture, Secure Build*)	SDL – *Requirements, Design, Implementation*	SSDF PS.1, PS.2, PW.1, PW.2

3. Gestión de requisitos de seguridad en el SDLC	Verification (*Security Testing, Code Review*)	SDL – Verification (*Testing, Security Push*)	SSDF PW.3, PW.4 / CSF DET, PROTECT
4. Gestión de incidentes y respuesta segura	Operations (*Incident Management, Environment Hardening*)	SDL – *Release, Response*	SSDF RV.1, RV.2 / CSF *RESPOND, RECOVER*
5. Gestión de proveedores y artefactos externos	*Governance → Policy & Compliance* (3rd *party*)	SDL – *Requirements (Dependencies), Design Reviews*	SSDF PO.5, PS.3 (Supply Chain) / CSF *SUPPLY CHAIN*
6. Cultura, formación y concienciación	*Governance → Education & Guidance*	SDL – *Training*	SSDF PO.4
7. Herramientas, automatización y soporte tecnológico	SAMM – *Implementation* (*Secure Build, Deployment*)	SDL – *Tooling & Automation*	SSDF PW.5, PW.6
8. Métricas, auditoría y mejora continua	*Governance → Strategy & Metrics*	SDL – *Post-Release Metrics & Lessons Learned*	SSDF PO.1, RV.3 / CSF *GOVERN & IDENTIFY*

3.3 Estructura interna del modelo

Cada uno de los cuatro dominios incluye:

1. Objetivo del dominio
2. Prácticas incluidas
3. Relación con estándares
4. Actividades mínimas requeridas (nivel básico)
5. Actividades avanzadas (nivel intermedio y avanzado)
6. Roles involucrados
7. Artefactos generados
8. Métricas sugeridas

4. Dominios y prácticas

Los dominios constituyen la columna vertebral del modelo y permiten organizar las actividades, responsabilidades, artefactos y evidencias requeridas para una adopción progresiva. Cada dominio contiene propósito, prácticas obligatorias del modelo, actividades mínimas, roles involucrados y evidencias requeridas.

4.1 Gobernanza y dirección del programa

4.1.1 Propósito

Establecer una estructura de gobernanza clara que asegure que la seguridad en el desarrollo de *software* sea una prioridad institucional, alineada con los objetivos del banco, las regulaciones nacionales (SUGEF, CONASSIF, BCCR) y los marcos internacionales seleccionados (OWASP SAMM, NIST SSDF/CSF, Microsoft SDL).

Este dominio busca garantizar que exista:

- Una dirección estratégica de la seguridad.
- Roles y responsabilidades claramente asignados.
- Procesos de evaluación continua y alineación con el riesgo institucional.
- Sostenibilidad del modelo a través de políticas, métricas y cultura organizacional.
- Es el dominio que habilita al resto: sin gobernanza, no hay procesos consistentes ni mejora continua.

4.1.2 Prácticas del dominio

Este dominio incorpora prácticas derivadas de los tres marcos:

Desde OWASP SAMM – Dominio de Gobernanza

- Estrategia y métricas de seguridad.
- Políticas organizacionales.
- Gestión de roles y recursos.

Desde NIST SSDF – PO (*Prepare the Organization*)

- Definir roles de seguridad en el SDLC.
- Establecer políticas formales.
- Integrar riesgo de *software* en gobernanza corporativa.

Desde Microsoft SDL – Fase Pre-SDL y Requisitos

- Capacitación obligatoria anual.
- Reglas organizacionales de cumplimiento.

- Definición de criterios de seguridad que son obligatorios para todos los equipos.

Prácticas integradas del modelo

- Definición institucional del programa de desarrollo seguro (SSDLC).
- Alineamiento con marcos regulatorios costarricenses (SUGEF, BCCR, Ley 8968).
- Gobernanza basada en riesgo y criticidad de sistemas bancarios.
- Creación de un Comité de Seguridad de *Software*.
- Integración con el SGSI institucional y auditorías internas.

4.1.3 Actividades mínimas

A continuación, las actividades esenciales que toda entidad bancaria debe implementar para cumplir este dominio:

1. Definir y aprobar una política de desarrollo seguro
    - Alcance: sistemas internos, expuestos al público y API.
    - Obligatoriedad: todos los proyectos, sin excepción.
    - Integración con SDLC/Agile de la institución.
2. Asignación formal de roles y responsabilidades
    - Propietarios de riesgos de *software*.
    - Comité SSDLC.
    - CISO, jefatura de desarrollo, arquitectos y líderes de QA.
3. Plan anual de seguridad en el desarrollo
    - Presupuesto.
    - Capacitación obligatoria según SDL (80 % cumplimiento).
    - *Roadmap* de mejoras.
4. Clasificación de sistemas según criticidad
    - Clave para banca costarricense (impacto financiero/regulatorio).
5. Definición de métricas estratégicas

Ejemplos:

- % de proyectos con modelado de amenazas.
- % de vulnerabilidades críticas resueltas dentro del SLA.
- % de desarrolladores capacitados.
- Cumplimiento de controles de NIST/SDL/SAMM.

6. Establecer auditorías internas periódicas
    - Evaluar cumplimiento SSDLC.
    - Revisar desviaciones y planes de acción.

4.1.4 Evidencias y artefactos

Estos son los artefactos que deben generarse para demostrar implementación, alineados con el enfoque investigativo:

1. Documentación estratégica
    - Política Institucional de Desarrollo Seguro (PIDS).
    - Plan anual del programa SSDLC.
    - Mapa de roles y responsabilidades (RACI).
    - Matriz de madurez inicial alineada con SAMM/SSDF.
2. Instrumentos de gobernanza
    - Actas del Comité SSDLC.
    - Minutas de reuniones técnicas.
    - Informes trimestrales sobre el avance del programa.
3. Evidencias de cumplimiento
    - Registro de capacitaciones de SDL (80 % del personal).
    - Informes de cumplimiento de normativas SUGEF/CONASSIF.
    - Listado de proyectos con adopción obligatoria del modelo.

4. Métricas y KPI
    - *Dashboard* de indicadores.
    - Reporte de evolución de madurez.
    - Resultados de auditorías internas y externas.
5. Integración con el SGSI
    - Controles NIST/ISO vinculados a seguridad en desarrollo.
    - Evidencias de la incorporación del modelo a los procesos de gestión de riesgos.

4.2 Gestión del ciclo de vida del desarrollo seguro (SSDLC)

4.2.1 Propósito

Establecer, documentar y mantener requisitos de seguridad claros, verificables y trazables que guíen el desarrollo de *software* en la entidad bancaria. Este dominio asegura que las necesidades de seguridad se definan desde la etapa de análisis, integrándose al *backlog* funcional y alineándose con:

- Normativa nacional (SUGEF, Ley 8968, políticas internas del BCCR).
- Estándares internacionales (NIST SSDF/CSF, OWASP SAMM, SDL).
- El modelo de riesgos propio de la institución.
- La protección del consumidor financiero.

Este dominio evita que la seguridad se incorpore de forma tardía.

4.2.2 Prácticas del dominio

Las prácticas provienen de la integración de los tres marcos:

1. Identificación de requisitos de seguridad
    - Derivación a partir de análisis de riesgos (NIST SSDF PW.1, OWASP SAMM *Governance/Strategy*).
    - Identificación de requisitos regulatorios obligatorios.
    - Requisitos específicos del sector bancario: *autenticación* fuerte, integridad transaccional, antifraude, uso de criptografía robusta, segregación de funciones.

2. Clasificación y priorización

    - Establecer criticidad (alto, medio, bajo) según impacto operacional o financiero.
    - Alinear con categorías del NIST CSF (PR.AC, PR.DS, PR.IP, DE.CM).

3. Incorporación al *backlog*

    - Los requisitos de seguridad se registran como historias de usuario de seguridad, criterios de aceptación o *Definition of Done*.
    - Mantener trazabilidad bidireccional hacia amenazas, controles y casos de prueba.

4. Validación y refinamiento

    - Revisión de requisitos por un Equipo SSDLC/*Security Champion.*
    - Revisión de cumplimiento con prácticas del SDL (por ejemplo, uso obligatorio de criptografía FIPS 140-2).

4.2.3 Actividades mínimas

Estas actividades son obligatorias para que un *software* pueda avanzar al diseño:

1. Recolección de requisitos

    - Analizar requisitos funcionales y convertirlos a requisitos de seguridad.
    - Reuniones con Arquitectura, Ciberseguridad y Producto.

2. Elaboración del *Security Requirements Document* (SRD)

Debe contener mínimamente:

- Requisitos regulatorios aplicables.
- Requisitos técnicos.
- Requisitos de privacidad de datos.
- Requisitos de continuidad y disponibilidad.
- Requisitos para integridad transaccional.
- Requisitos criptográficos.
- Controles por tipo de API, servicio o canal digital.

3. Validación del SRD

    - Revisión formal con el *Security Champion* o Comité SSDLC (si existe).
    - Revisión con ciberseguridad para aceptación técnica.

4. Integración al *backlog*

    - Creación de historias como: “Como entidad bancaria, debo validar entradas para evitar inyecciones SQL según OWASP ASVS nivel 2”.
    - Incorporación de criterios de aceptación de seguridad.

5. Actualización después de cambios

    - Si un cambio funcional introduce riesgo (por ejemplo, una nueva API), se gatilla una actualización del SRD y del *threat model*.

4.2.4 Evidencias y artefactos

Estos elementos son fundamentales para auditoría, trazabilidad y cumplimiento normativo.

1. *Security Requirements Document* (SRD)

Documento formal de requisitos de seguridad. Incluye:

- Riesgos asociados.
- Controles esperados.
- Responsables.
- Referencias regulatorias.

2. Historias de usuario de seguridad

Ejemplos:

- “Como usuario del sistema, mis datos deben estar cifrados en tránsito usando TLS 1.3 según política del banco”.
- “Como administrador, requiero autenticación multifactor conforme políticas corporativas del BCCR”.

3. Matriz de trazabilidad de seguridad

Relación entre:

- Requisitos → amenazas → controles → pruebas.

4. Minutas de revisión del Comité SSDLC / *Security Champion*

- Registro de aprobación o rechazo de requisitos.

5. Evidencias de análisis normativo

- Mapeo SUGEF, NIST, SAMM, SDL → requisitos aplicados.

6. Versiones validadas del *backlog*

- Con criterios de aceptación de seguridad incorporados.

4.3 Gestión de requisitos de seguridad en el SDLC

4.3.1 Propósito

Establecer un proceso formal, repetible y verificable para la definición, documentación, priorización y mantenimiento de los requisitos de seguridad en los productos de *software* desarrollados por la entidad bancaria. Este dominio garantiza que la seguridad se integre desde la fase inicial del ciclo de vida, alineándose con el enfoque "Seguro por Diseño, por Defecto y por Despliegue" así como con los principios rectores definidos.

4.3.2 Prácticas del dominio

1. Identificación de requisitos de seguridad

Consiste en levantar y documentar los requisitos de seguridad que debe cumplir el *software*, considerando amenazas, escenarios de riesgo, normativas regulatorias y políticas internas de la entidad bancaria.

Incluye la participación de analistas funcionales, arquitectos de *software*, equipos de ciberseguridad y líderes técnicos.

2. Priorización y clasificación

Establece la relevancia de cada requisito según su criticidad, impacto en la operación bancaria, exposición a internet y sensibilidad de los datos procesados. La clasificación permite planificar capacidades técnicas y asignar recursos de manera eficiente.

3. Integración con requisitos funcionales

Asegura que los requisitos de seguridad no se gestionen como anexos, sino como parte integral de los requisitos funcionales del sistema. La seguridad forma parte de las historias de usuario, criterios de aceptación y definición de "hecho" en los equipos ágiles.

4. Gestión de cambios

Define mecanismos para actualizar requisitos de seguridad cuando ocurran modificaciones en el diseño, aparezcan nuevas amenazas o se actualicen normativas regulatorias. Permite que el sistema evolucione, manteniendo niveles adecuados de protección.

4.3.3 Actividades mínimas

1. Levantamiento inicial de requisitos de seguridad. Se documentan requisitos basados en análisis de riesgos, modelos de amenazas y políticas institucionales.

2. Consulta obligatoria a ciberseguridad. Los equipos deben someter sus requisitos a revisión por parte del departamento de ciberseguridad, especialmente en sistemas expuestos a internet o que procesan datos sensibles.

3. Clasificación de requisitos según impacto. Se utiliza categorización mínima: Alto, Medio y Bajo, considerando confidencialidad, integridad y disponibilidad.

4. Inclusión de requisitos en las historias de usuario. Toda historia que involucre datos, conexiones externas, autenticación o lógica crítica debe incorporar criterios de seguridad en su definición de aceptación.

5. Validación durante revisiones de *sprint*. En metodologías ágiles, cada iteración revisa evidencia de cumplimiento de los requisitos.

6. Actualización mediante control de cambios. Toda modificación a un diseño, arquitectura o integración debe desencadenar reevaluación de los requisitos de seguridad correspondientes.

4.3.4 Evidencias y artefactos

Estos artefactos sirven como insumos para auditorías internas, revisiones de cumplimiento regulatorio y trazabilidad solicitada por la gobernanza del sector bancario costarricense.

Artefactos obligatorios

- Documento de requisitos de seguridad del proyecto.

  Incluye requisitos regulatorios, técnicos y operativos.

- Lista priorizada de requisitos de seguridad.

  Con justificación de prioridad y riesgos asociados.

- Matriz de trazabilidad de requisitos.

  Relaciona requisitos → historias de usuario → pruebas → evidencias.

- Revisión de seguridad por área de ciberseguridad.

  Con observaciones, riesgos abiertos y acuerdos.

- Bitácora de cambios de requisitos.

  Registra decisiones de actualización, impacto y responsables.

Evidencia en procesos ágiles

- Historias de usuario con criterios de seguridad.

  Por ejemplo: “El sistema debe validar entradas para prevenir inyección”.

- Actas de revisión de *sprint*.

  Con resultados documentados de la verificación de cumplimiento.

- *Checklist* de seguridad aplicado en cada iteración.

4.4 Gestión de incidentes y respuesta segura

4.4.1 Propósito

El propósito de este dominio es establecer un conjunto estructurado de prácticas, procesos y capacidades que permitan a la organización prevenir, detectar, contener, erradicar y recuperarse de incidentes de seguridad vinculados al *software* desarrollado internamente o adquirido externamente. Este dominio busca asegurar que los equipos de desarrollo, operaciones, ciberseguridad y continuidad del negocio actúen de forma coordinada ante fallos, vulnerabilidades, ataques o comportamientos anómalos que comprometan la confidencialidad, integridad o disponibilidad de los servicios bancarios.

En el contexto del sector bancario costarricense, este dominio resulta crítico, especialmente debido a la alta regulación, la sensibilidad de los datos financieros y el incremento de incidentes como ataques de denegación de servicio, inyección de código y explotación de componentes vulnerables. Su implementación fortalece la resiliencia institucional y reduce tiempos de indisponibilidad que podrían impactar a la ciudadanía.

4.4.2 Prácticas del dominio

1. Definición del marco de gestión de incidentes.

   Establece lineamientos, roles, responsabilidades y procedimientos para la actuación coordinada ante incidentes de seguridad.

2. Integración del equipo de desarrollo en el ciclo de respuesta.

Asegura que los desarrolladores participen en la investigación y corrección de fallas originadas en el *software*.

3. Monitoreo continuo y detección temprana.

   Implementa mecanismos automatizados y manuales para identificar vulnerabilidades, anomalías o comportamientos maliciosos.

4. Gestión de vulnerabilidades.

   Incluye la clasificación, priorización, asignación y remediación de fallas basadas en riesgo.

5. Comunicación interna y externa.

   Define protocolos para la notificación oportuna a la alta dirección, áreas técnicas, reguladores y usuarios cuando corresponda.

6. Lecciones aprendidas y retroalimentación al SSDLC.

   Incorpora los hallazgos de los incidentes en las fases de requisitos, diseño, implementación y pruebas de futuros desarrollos.

4.4.3 Actividades mínimas

1. Definir un procedimiento formal de respuesta a incidentes.

   Debe incluir etapas de identificación, clasificación, contención, erradicación, recuperación y cierre.

2. Establecer un equipo de respuesta a incidentes (Equipo CSIRT).

   Debe involucrar a personal de ciberseguridad, desarrollo, infraestructura y gestión del riesgo.

3. Clasificar los incidentes según su criticidad.

   Debe considerar categorías como indisponibilidad, fuga de información, ejecución remota de código o acceso no autorizado.

4. Implementar monitoreo continuo del *software* en operación.

   Incluye registros (*logs*) alertas, indicadores de compromiso y supervisión de vulnerabilidades en componentes externos.

5. Documentar cada incidente.

Debe registrarse el impacto, causa raíz, tiempo de respuesta, acciones correctivas y responsables.

6. Integrar la gestión de incidentes con la planificación del desarrollo.

   Las vulnerabilidades identificadas deben traducirse en requisitos no funcionales, historias técnicas o controles obligatorios para futuros desarrollos.

4.4.4 Evidencias y artefactos

1. Procedimiento institucional de respuesta a incidentes.

   Incluye flujos de trabajo, roles, matriz RACI (Responsable, Aprobador, Consultado, Informado) tiempos máximos de atención y políticas relacionadas.

2. Registro de incidentes de seguridad.

   Documento o sistema que almacena las descripciones, fechas, severidad, impacto, responsables y acciones tomadas.

3. Informes post-incidente.

   También conocidos como informes de “lecciones aprendidas”, describen la causa raíz, las mejoras implementadas y recomendaciones.

4. Bitácoras y registros de monitoreo.

   Incluye archivos de eventos, alertas de seguridad, reportes de herramientas automatizadas y evidencias de detección temprana.

5. Plan de continuidad tecnológica y recuperación.

   Documento que detalla protocolos para restaurar servicios afectados por incidentes de seguridad.

6. Historial de remediación de vulnerabilidades.

   Incluye actualizaciones aplicadas, parches instalados, análisis de verificación y fechas de cierre de brechas.

7. Comunicaciones internas y externas.

   Mensajes a reguladores, dirección, usuarios o proveedores relacionados con incidentes críticos.

4.5 Gestión de proveedores y artefactos externos

4.5.1 Propósito

El propósito del dominio 5 es garantizar que los proveedores externos, contratistas, consultoras de *software* y los artefactos que estos suministran, como librerías, paquetes, componentes, servicios en la nube, API externas y herramientas de terceros, cumplan con los requisitos de seguridad establecidos por la entidad bancaria pública. Este dominio busca reducir los riesgos asociados al uso de *software* de terceros, asegurar la trazabilidad de los componentes externos y mantener un control riguroso sobre dependencias, versiones, vulnerabilidades y prácticas de desarrollo adoptadas por los proveedores.

Este dominio es especialmente relevante en el sector bancario costarricense debido a que varias instituciones tercerizan parcial o totalmente el desarrollo de sistemas críticos, lo que genera una superficie de riesgo considerable si no se implementan controles robustos de validación y supervisión.

4.5.2 Prácticas del dominio

1. Evaluación y clasificación de proveedores.

   Establecer un proceso formal para evaluar el nivel de madurez en seguridad de cada proveedor, considerando su cumplimiento con estándares internacionales, certificaciones, historial de incidentes y mecanismos de protección de datos.

2. Validación de seguridad de componentes externos.

   Revisar y aprobar los artefactos de *software* externos antes de su incorporación, utilizando análisis de composición de *software* (SCA, *Software Composition Analysis*) revisión documental y pruebas independientes.

3. Requisitos contractuales de seguridad.

   Incluir en los contratos, cláusulas que obliguen a los proveedores a implementar prácticas de desarrollo seguro, gestionar vulnerabilidades, brindar parches oportunos y aceptar auditorías.

4. Gestión del riesgo de la cadena de suministro digital.

   Asegurar que los componentes externos no introduzcan vulnerabilidades, *malware* o dependencias no autorizadas, aplicando un monitoreo continuo.

5. Supervisión continua del proveedor.

   Evaluar de forma periódica el desempeño de los proveedores mediante indicadores de cumplimiento, revisiones de seguridad, reportes de vulnerabilidades y validación de versiones.

6. Control de dependencias abiertas y cerradas.

   Mantener una trazabilidad completa del origen, uso, alcance y versiones de todas las librerías y paquetes utilizados en los desarrollos internos.

4.5.3 Actividades mínimas

1. Clasificar a los proveedores según el nivel de criticidad del servicio.

   Elegir criterios como impacto, tipo de información manejada, exposición a Internet, nivel de acceso otorgado y relevancia para procesos financieros.

2. Solicitar evidencias del proceso de desarrollo seguro del proveedor.

   Requerir elementos como políticas, certificaciones, resultados de pruebas, procesos de control de cambios y mecanismos de protección de datos.

3. Incorporar cláusulas mínimas de seguridad en todos los contratos.

   Exigir:

   - Notificación de incidentes en plazos definidos.
   - Corrección obligatoria de vulnerabilidades.
   - Cumplimiento de NIST, OWASP y políticas internas.
   - Acceso a auditorías técnicas cuando la entidad lo solicite.

4. Realizar análisis SCA (*Software Composition Analysis*) al recibir componentes externos.
   Verificar vulnerabilidades conocidas, licenciamiento, antigüedad y reputación del componente.

5. Revisar y aprobar los artefactos antes de su integración al repositorio institucional.

   Hay que asegurar que ningún componente entre sin validación.

6. Monitorear el ciclo de vida del proveedor mediante indicadores.

   Evaluar atrasos, vulnerabilidades recurrentes, calidad de entregables y cumplimiento contractual.

7. Mantener inventarios actualizados de dependencias externas.

   Documentar sus versiones, origen, riesgos, fechas de actualización y responsables internos.

4.5.4 Evidencias y artefactos

1. Registro de evaluación de proveedores.

   Matriz con criterios, calificaciones, nivel de riesgo y decisiones de aprobación.

2. Catálogo actualizado de componentes externos.

   Incluye: nombre del componente, versión, proveedor, riesgo asociado, fecha de última revisión.

3. Resultados de escaneos SCA.

   Reportes de vulnerabilidades, licencias detectadas y recomendaciones técnicas.

4. Contratos y anexos con cláusulas de seguridad.

   Documentos firmados que evidencian obligaciones de seguridad del proveedor.

5. Actas de revisión y aprobación de artefactos externos.

   Validación formal de cada módulo, librería o servicio.

6. Informes periódicos de desempeño del proveedor.

   Evaluaciones trimestrales o semestrales según criticidad.

7. Historial de incidentes asociados a proveedores.

   Bitácora con fechas, impacto, medidas correctivas y acuerdos posteriores.

8. Registros de seguimiento de actualizaciones y parches.

   Evidencia de que los proveedores cumplen con la entrega de correcciones.

4.6 Cultura, formación y concienciación

4.6.1 Propósito

El propósito de este dominio es establecer una cultura organizacional en la que la seguridad del *software* sea un valor transversal y sostenido en el tiempo. Busca asegurar que todos los actores involucrados en el ciclo de vida de desarrollo comprendan su rol en la protección de los sistemas bancarios, reciban capacitación continua y participen en actividades que fortalezcan la responsabilidad colectiva de mantener un ecosistema tecnológico seguro.

Este dominio responde directamente a los hallazgos del análisis institucional, que identificaron falta de capacitación, baja madurez cultural y débil adopción de estándares internacionales

como barreras centrales en la entidad bancaria estudiada. Por ello, se formaliza un conjunto de acciones estructuradas y medibles para elevar la madurez cultural en seguridad.

4.6.2 Prácticas del dominio

1. Programa institucional de formación en seguridad

   Capacitar de forma periódica a todos los miembros del equipo de desarrollo, pruebas, arquitectura y gestión, según su responsabilidad dentro del ciclo de vida del *software*.

2. Concienciación continua en buenas prácticas de desarrollo seguro

   Implementar campañas, recordatorios, actividades prácticas y difusión de casos reales para mantener la seguridad como un tema cotidiano y no eventual.

3. Integración de la seguridad en el proceso de inducción

   Hay que asegurar que todo funcionario que ingrese a la institución reciba formación obligatoria en temas de desarrollo de *software* seguro, políticas internas, estándares aplicables y responsabilidades individuales.

4. Fomento de una cultura colaborativa orientada a DevSecOps

   Promover espacios de intercambio técnico, revisiones entre pares, mentorías internas y creación de comunidades de práctica para fortalecer el aprendizaje colectivo.

5. Evaluación periódica del nivel de conocimiento

   Medir el grado de asimilación mediante evaluaciones, ejercicios prácticos, pruebas internas y simulaciones de seguridad.

6. Comunicación interna estratégica sobre temas de seguridad

   Difundir boletines, alertas internas, indicadores relevantes y recomendaciones basadas en incidentes recientes del sector bancario costarricense.

4.6.3 Actividades mínimas

1. Capacitación anual obligatoria

   - Al menos un curso formal sobre desarrollo de *software* seguro por año.
   - Debe incluir temas como análisis de amenazas, validación de entradas, cifrado, autenticación robusta y despliegue seguro.

2. Talleres prácticos trimestrales

- Actividades técnicas orientadas a aplicar conocimientos: revisión de código seguro, ejercicios de modelado de amenazas, uso de herramientas SAST (pruebas estáticas), DAST (pruebas dinámicas) y SCA (análisis de composición de *software*).

3. Simulaciones de incidentes

   - Prácticas controladas en las que se simulan vulnerabilidades reales para evaluar la respuesta de los equipos.

4. Incorporación del tema en reuniones de trabajo

   - Espacios periódicos para revisar buenas prácticas, alertas recientes y lecciones aprendidas.

5. Registro y seguimiento del historial de capacitación

   - Mantener un sistema institucional que permita verificar el cumplimiento y detectar brechas.

6. Evaluaciones anuales de conocimiento

   - Pruebas escritas y prácticas para medir el nivel de madurez del personal.

4.6.4 Evidencias y artefactos

1. Plan anual de capacitación

   Documento formal aprobado por la organización que detalle actividades, contenidos, periodicidad, responsables y mecanismos de evaluación.

2. Registro de participación del personal

   Listados, certificados, bitácoras y reportes de asistencia.

3. Material de formación

   Presentaciones, manuales, videos educativos internos, guías técnicas y laboratorios prácticos utilizados durante los talleres.

4. Resultados de evaluaciones y métricas de aprendizaje

   Informes sobre el nivel de cumplimiento, puntuaciones obtenidas, brechas detectadas y evolución histórica.

5. Informes de simulaciones de incidentes

   Reportes que documenten los ejercicios, hallazgos, tiempos de respuesta y mejoras necesarias.

6. Comunicados y campañas de concienciación

   Boletines, infografías, alertas internas y campañas trimestrales de sensibilización.

7. Evidencia de actividades DevSecOps culturales

   Registros de sesiones de revisión entre pares, comunidades de práctica, *tech talks* internas y reuniones de intercambio.

4.7 Herramientas, automatización y soporte tecnológico

4.7.1 Propósito

El dominio de herramientas, automatización y soporte tecnológico tiene como propósito establecer un conjunto estructurado de capacidades técnicas que permitan integrar la seguridad dentro de los procesos de desarrollo de *software* mediante herramientas especializadas, automatización continua y mecanismos que reduzcan el riesgo operativo y el error humano.

Su objetivo central es garantizar que los equipos de desarrollo en una entidad bancaria pública costarricense cuenten con la infraestructura tecnológica necesaria para incorporar prácticas de seguridad de forma consistente, repetible y verificable en cada fase del ciclo de vida del *software*. Esto incluye la integración de análisis automatizados, la adopción de herramientas aprobadas institucionalmente y el establecimiento de estándares técnicos que aseguren coherencia entre proyectos.

Este dominio responde directamente a los hallazgos del análisis del estado actual (Capítulo IV), donde se evidenció:

- La baja integración de seguridad en fases tempranas de desarrollo.
- La ausencia de herramientas unificadas entre equipos.
- El uso limitado de la automatización en los procesos de verificación.
- La falta de controles consistentes en *pipelines* de integración y despliegue continuos.
- La necesidad de reducir la dependencia de prácticas reactivas y aumentar la prevención.

4.7.2 Prácticas del dominio

1. Estándares corporativos de herramientas de seguridad.

   Define las herramientas autorizadas para análisis de código, composición de *software*, pruebas dinámicas, gestión de vulnerabilidades y protección de artefactos.

2. Integración de seguridad en la integración y despliegue continuos (CI/CD).

Garantiza que las verificaciones de seguridad sean parte obligatoria del flujo automatizado de construcción, pruebas y despliegue.

3. Escaneo automatizado en todas las fases del desarrollo.

   Incluye análisis estático, dinámico, interactivo, de dependencias y configuración, incorporados directamente en el proceso de construcción.

4. Gestión centralizada de vulnerabilidades.

   Permite identificar, priorizar y gestionar riesgos técnicos mediante herramientas institucionales y paneles de control.

5. Inventario y control de artefactos tecnológicos.

   Mantiene un catálogo actualizado de librerías, dependencias, contenedores y componentes reutilizables, protegiendo su integridad.

6. Seguridad del entorno de desarrollo.

   Establece controles para entornos locales, servidores de compilación, repositorios de código y plataformas colaborativas.

7. Automatización para la reducción del error humano.

   Implementa validaciones, plantillas, repositorios seguros y mecanismos automáticos para prevenir configuraciones inseguras.

4.7.3 Actividades mínimas

1. Definir y publicar el catálogo oficial de herramientas permitidas.

   Incluye criterios de selección, niveles de riesgo aceptables y responsables de su administración.

2. Integrar verificaciones obligatorias en los pipelines de CI/CD.

   Los proyectos no deben proceder a etapas de despliegue si existen vulnerabilidades clasificadas como críticas o altas.

3. Configurar análisis automáticos recurrentes.

   Los sistemas deben ser evaluados con una frecuencia mínima predefinida según criticidad del proyecto.

4. Implementar repositorios seguros de artefactos.

Los paquetes, librerías y contenedores deben almacenarse en repositorios validados, firmados y protegidos.

5. Establecer reglas de calidad y seguridad para el análisis estático.

   Incluye definición de umbrales, requisitos mínimos y estándares internos adaptados al contexto bancario.

6. Integrar gestión centralizada de vulnerabilidades.

   Todas las herramientas deben reportar hacia una plataforma unificada de monitoreo y priorización.

7. Proteger los entornos de desarrollo.

   Incluye control de accesos, distribución controlada de secretos, protección de llaves y supervisión de cambios.

4.7.4 Evidencias y artefactos

1. Catálogo institucional de herramientas aprobadas.

   Documento oficial que define herramientas permitidas, restricciones, criterios de selección y responsables.

2. Configuración del *pipeline* de CI/CD.

   Evidencia técnica que demuestra integración de verificaciones de seguridad.

3. Reportes automáticos de análisis SAST, DAST y SCA.

   Incluyen métricas, vulnerabilidades detectadas, severidad y recomendaciones.

4. Matrices de gestión de vulnerabilidades.

   Consolidado de riesgos, fechas de descubrimiento, responsables y acciones de mitigación.

5. Registro del inventario de componentes y artefactos.

   Incluye versiones, repositorios asociados, nivel de criticidad y fecha de aprobación.

6. Evidencia de cumplimiento de umbrales de calidad y seguridad.

   Reportes emitidos automáticamente por herramientas corporativas.

7. Registros de seguridad en los entornos de desarrollo.

Pruebas de cumplimiento de controles, auditorías internas, revisiones de configuración y *logs*.

4.8 Métricas, auditoría y mejora continua

4.8.1 Propósito

Este dominio tiene como propósito asegurar que el programa de desarrollo de *software* seguro evolucione de manera sostenida y medible. Para lograrlo, se establecen sistemas estructurados de métricas, auditorías internas y mecanismos de retroalimentación que permitan evaluar la eficacia de las prácticas aplicadas en el ciclo de vida del desarrollo seguro. El objetivo central es mantener una visión clara y cuantificable del desempeño, facilitando la toma de decisiones basadas en evidencia, reduciendo riesgos y alineando las actividades con los principios de mejora definidos en el modelo metodológico.

4.8.2 Prácticas del dominio

1. Establecimiento de métricas clave de seguridad.

   Definir indicadores estratégicos y operativos que permitan medir la madurez, eficacia y cumplimiento de las prácticas de desarrollo seguro.

2. Monitoreo periódico del desempeño.

   Supervisar la evolución de las métricas durante las fases del ciclo de vida del desarrollo seguro para identificar desviaciones, riesgos emergentes y oportunidades de mejora.

3. Auditorías internas especializadas.

   Realizar revisiones estructuradas en intervalos definidos para evaluar la adherencia al modelo, la calidad de los artefactos generados y la correcta aplicación de controles de seguridad.

4. Gestión de hallazgos y no conformidades.

   Establecer un procedimiento formal para clasificar, priorizar, corregir y verificar la resolución de hallazgos relacionados con prácticas de seguridad.

5. Ciclos formales de mejora continua.

   Utilizar la información proveniente de métricas, auditorías y retroalimentación para ajustar procesos, actualizar lineamientos y fortalecer la gobernanza del programa.

6. Informes de avance y toma de decisiones basada en datos.

   Generar informes periódicos para la alta dirección que respalden decisiones estratégicas en materia de seguridad, madurez y priorización de iniciativas.

4.8.3 Actividades mínimas

1. Definir el catálogo institucional de métricas de seguridad.

   Debe incluir indicadores asociados al ciclo de vida del desarrollo seguro (SSDLC), calidad del código, vulnerabilidades detectadas, cumplimiento normativo y desempeño de herramientas automatizadas.

2. Configurar un panel de control institucional (cuadro de mando).

   Integrar métricas centralizadas que permitan visualizar el estado de madurez, las tendencias de riesgo y las comparaciones entre proyectos y direcciones de tecnología.

3. Ejecutar auditorías internas al menos dos veces al año.

   Estas auditorías deben revisar prácticas, procesos, artefactos y alineación con los dominios del modelo y los estándares adoptados.

4. Registrar y dar seguimiento a hallazgos.

   Cada hallazgo debe contar con clasificación por criticidad, responsable asignado y fecha de cierre comprometida.

5. Realizar retrospectivas de seguridad.

6. Evaluar la efectividad de las prácticas aplicadas y documentar mejoras, lecciones aprendidas y riesgos emergentes.

7. Elaborar informes ejecutivos trimestrales.

   Estos informes deben permitir a la alta dirección evaluar el avance del programa, identificar debilidades y priorizar inversión o capacitación.

4.8.4 Evidencias y artefactos

1. Catálogo institucional de métricas de seguridad.

   Documento formal con definiciones, fórmulas, periodicidad y responsables de cada indicador.

2. Cuadro de mando institucional de seguridad del *software*.

   Panel actualizado que consolida métricas de vulnerabilidades, cumplimiento, calidad y desempeño del SSDLC.

3. Informes de auditoría interna.

Registros oficiales de revisiones, hallazgos, no conformidades y recomendaciones emitidas por los auditores internos.

4. Plan de tratamiento de hallazgos.

   Documento que detalla acciones correctivas, fechas, compromisos y responsables.

5. Historial de métricas y análisis de tendencias.

   Reportes que evidencian evolución, comparación interanual y proyecciones.

6. Actas de retrospectivas de seguridad.

   Registros estructurados con lecciones aprendidas, mejoras propuestas y acciones priorizadas.

7. Informes ejecutivos trimestrales para la alta dirección.

   Síntesis formal del estado del programa, nivel de madurez, riesgos y necesidades estratégicas.

CAPÍTULO V. RECOMENDACIONES PARA APLICAR EL MODELO

La adopción del modelo metodológico propuesto requiere una implementación gradual, estructurada y alineada con las capacidades, contexto regulatorio y nivel de madurez del sector bancario público costarricense. Las siguientes recomendaciones orientan la correcta aplicación del modelo y facilitan su sostenibilidad en el tiempo.

1. **Establecer un gobierno claro del programa de desarrollo seguro**

   - Crear un Comité de Desarrollo Seguro (Comité SSDLC).

     Este comité debe estar conformado por representantes de áreas como desarrollo, arquitectura, ciberseguridad, calidad, gestión de riesgos y la alta dirección. Su función principal será dirigir, supervisar y priorizar las iniciativas relacionadas con la seguridad en el ciclo de vida del *software*.

   - Definir roles y responsabilidades formales mediante matrices RACI.

     Esto permitirá evitar ambigüedades y asegurará que cada práctica tenga un responsable primario.

   - Incorporar el modelo dentro de la gobernanza tecnológica institucional.

     Debe estar alineado con políticas de ciberseguridad, marcos de control interno y lineamientos emitidos por reguladores financieros.

2. **Integrar el SSDLC dentro de los procesos actuales de desarrollo**

   - Adoptar el modelo de manera incremental.

     Dado que los equipos trabajan con metodologías ágiles, la integración debe realizarse *sprint* por *sprint*, incorporando actividades mínimas de seguridad desde el diseño hasta el despliegue.

   - Priorizar la seguridad en las fases tempranas del SDLC.

     El análisis del estado actual demostró que la seguridad se incorpora principalmente en fases tardías. Se recomienda iniciar con:

     - Definición de requisitos de seguridad.
     - Modelado de amenazas.
     - Revisiones de arquitectura segura.

   - Incorporar herramientas automatizadas dentro de CI/CD.

La adopción progresiva de SAST (Análisis estático), DAST (Análisis dinámico) y SCA (Análisis de composición) ayudará a reducir vulnerabilidades antes del despliegue.

3. **Formalizar los requisitos de seguridad en todos los proyectos**

   - Crear plantillas institucionales para requisitos de seguridad.

     Dichas plantillas deben alinearse con NIST SSDF, Microsoft SDL y OWASP SAMM.

   - Asegurar trazabilidad con historias de usuario, criterios de aceptación y evidencias.

   - Incorporar revisiones obligatorias de requisitos por parte del área de ciberseguridad.

4. **Fortalecer la gestión de incidentes desde el desarrollo**

   - Integrar procesos de respuesta a incidentes desde las fases iniciales del SDLC.

   - Incluir escenarios de prueba relacionados con ataques reales sufridos por entidades financieras.

   - Establecer coordinación permanente con el CSIRT institucional.

5. **Gestionar adecuadamente proveedores y artefactos externos**

   - Crear criterios mínimos de seguridad para proveedores, incluyendo el uso obligatorio de análisis SAST/DAST/SCA y el cumplimiento de regulaciones locales.

   - Evaluar el riesgo de bibliotecas, *frameworks* y API antes de incorporarlas.

   - Solicitar evidencias de seguridad como condición contractual.

6. **Construir una cultura organizacional orientada al desarrollo seguro**

   - Implementar un programa anual de capacitación segmentado por rol.

   - Incorporar ejercicios prácticos: laboratorios, retos de seguridad, análisis de casos y simulaciones.

   - Promover la cultura DevSecOps, entendida como colaboración activa entre desarrollo, seguridad y operaciones.

7. **Fortalecer el soporte tecnológico**

   - Seleccionar y estandarizar herramientas de análisis de seguridad.

- Asegurar su integración total con *pipelines* de CI/CD.
- Designar un equipo de soporte para administración, actualización y monitoreo.

8. **Implementar métricas, auditoría y mejora continua**

- Definir un tablero de indicadores, por ejemplo:
  - Número de vulnerabilidades por aplicación.
  - Tiempo promedio de remediación.
  - Porcentaje de automatización en pruebas de seguridad.
  - Variabilidad de vulnerabilidades por *sprint*, entre otros.
- Ejecutar auditorías trimestrales del SSDLC.
- Aplicar retrospectivas orientadas exclusivamente al componente de seguridad.

9. **Adoptar el modelo según nivel de madurez**

Se recomienda utilizar un enfoque progresivo basado en tres niveles:

- Nivel 1 (Inicial): Cumplimiento mínimo de actividades esenciales.
- Nivel 2 (Intermedio): Integración completa en metodologías ágiles e incremento de automatización.
- Nivel 3 (Avanzado): Cultura DevSecOps institucional, métricas robustas y mejora continua.

Cada institución puede avanzar según recursos, cultura y madurez.

- Validación del modelo propuesto y trabajo futuro

Si bien el modelo metodológico propuesto se fundamenta en un análisis del estado actual de las prácticas de desarrollo de *software* seguro, así como en la integración de marcos internacionales ampliamente reconocidos, su validación empírica no fue abordada dentro del alcance de esta investigación. Esto se debe a que el presente trabajo tiene un enfoque de investigación aplicada orientado a la formulación de una propuesta metodológica y no a su implementación práctica.

No obstante, con el fin de fortalecer la aplicabilidad del modelo y evaluar su efectividad en contextos reales, se proponen como líneas de trabajo futuro diversas estrategias de validación empírica, entre las cuales se destacan las siguientes:

- Aplicación piloto controlada

  Como primera alternativa, se propone la ejecución de un piloto del modelo metodológico en un proyecto interno de desarrollo de *software* dentro de una entidad del sector bancario público costarricense. Esta aplicación permitiría evaluar la viabilidad del modelo en un entorno real, considerando variables como la carga operativa, la aceptación por parte de los equipos de desarrollo y la integración con los procesos existentes.

- Evaluación por un panel de expertos

  Adicionalmente, se recomienda la validación del modelo mediante un panel de expertos en ciberseguridad bancaria y desarrollo de *software* seguro, quienes podrían evaluar el modelo a partir de criterios como pertinencia, claridad, completitud, alineación normativa y factibilidad de adopción. Esta evaluación permitiría identificar oportunidades de mejora antes de una implementación a mayor escala.

- Evaluación mediante métricas pre y postimplementación

  Como complemento a las estrategias anteriores, se plantea la definición de métricas comparativas que permitan medir el impacto del modelo antes y después de su implementación. Entre estas métricas podrían considerarse la reducción de vulnerabilidades detectadas en etapas tempranas, el incremento en la adopción de prácticas de seguridad en fases de diseño y el nivel de cumplimiento de estándares de desarrollo seguro.

- Proyección del modelo como instrumento de mejora continua

  Finalmente, la validación empírica del modelo permitiría su evolución hacia un instrumento de mejora continua, ajustable a distintos niveles de madurez organizacional y replicable en otras entidades del sector financiero público costarricense, contribuyendo así al fortalecimiento de la seguridad del *software* a nivel nacional.

- Limitaciones del estudio

Este trabajo presenta una serie de limitaciones que deben ser consideradas al interpretar los resultados y el alcance del modelo metodológico propuesto.

En primer lugar, el estudio se basa en una muestra intencional de 25 profesionales del desarrollo de *software* pertenecientes a una única entidad del sector bancario público costarricense. Si bien esta muestra resulta adecuada para un estudio de carácter exploratorio y aplicado, sus resultados no pretenden ser generalizables a la totalidad del sector financiero nacional.

En segundo lugar, el análisis del estado actual de las prácticas de desarrollo de *software* seguro se centra en una sola organización, lo cual limita la diversidad de contextos organizacionales considerados. No obstante, la entidad seleccionada comparte características estructurales, regulatorias y operativas comunes a otras instituciones bancarias públicas del país, lo que permite extraer hallazgos relevantes a nivel contextual.

Asimismo, el modelo metodológico propuesto no fue validado mediante una implementación práctica o aplicación piloto durante el desarrollo de la investigación. En consecuencia, su evaluación empírica queda planteada como una línea de trabajo futuro, orientada a medir su efectividad en entornos reales de desarrollo de *software*.

Finalmente, el alcance del estudio es de naturaleza propositiva, enfocándose en la formulación de un modelo metodológico adaptado al contexto costarricense, sin abordar aspectos de ejecución, gestión del cambio organizacional o medición de impacto a largo plazo.

CAPÍTULO VI. CONCLUSIONES Y RECOMENDACIONES

Este capítulo presenta las conclusiones generales de la investigación y las recomendaciones para futuras acciones que fortalezcan la práctica del desarrollo de *software* seguro en el sector bancario costarricense.

Conclusiones

- La seguridad en el desarrollo de *software* constituye un elemento crítico para la sostenibilidad del sector bancario público costarricense.

  El análisis del contexto nacional revela incidentes recientes, como ataques DDoS a instituciones del Estado, que demuestran la vulnerabilidad del ecosistema financiero.

- La investigación evidenció una brecha importante entre la percepción de importancia de la seguridad y su implementación real.

  Aunque el 85 % de profesionales considera la seguridad como fundamental, la mayoría de las prácticas se aplican en fases tardías, lo que incrementa costos, riesgos y exposición institucional.

- Los estándares internacionales existentes no responden completamente a las necesidades del contexto costarricense.

  La falta de adopción plena de NIST, OWASP SAMM o SDL se debe a que no están adaptados a la realidad del sector bancario costarricense, lo cual justificó la construcción de un modelo tropicalizado.

- El modelo metodológico propuesto integra los elementos más robustos de OWASP SAMM, NIST SSDF/CSF y Microsoft SDL, organizados en ocho dominios que cubren gobernanza, procesos, soporte cultural y mejora continua.

- El modelo permite su adopción progresiva, respetando las limitaciones de recursos, madurez y capacidades técnicas de los equipos de desarrollo del sector bancario público.

- La integración del desarrollo seguro depende principalmente de factores no técnicos, como apoyo de la alta dirección, capacitación y cultura organizacional, coincidiendo con los resultados de la encuesta aplicada.

Recomendaciones

- Adoptar el modelo de forma gradual, iniciando con actividades mínimas por dominio.

- Asignar un patrocinador institucional de alto nivel, que impulse la adopción del modelo.

- Incluir métricas de seguridad obligatorias en los tableros de indicadores institucionales.

- Estandarizar herramientas y procesos, evitando la fragmentación tecnológica observada en el análisis del estado actual.

- Institucionalizar un programa formal de concienciación y capacitación continua en desarrollo seguro, que incluya formación técnica especializada (modelado de amenazas, revisión de código seguro, uso de herramientas automatizadas) campañas internas de cultura de seguridad y evaluación periódica de competencias.

- Revisar y actualizar el modelo al menos cada dos años, conforme avancen las prácticas de ciberseguridad y surjan nuevas normativas nacionales.

REFERENCIAS BIBLIOGRÁFICAS

Adelyar, A., & Norta, A. (2016). *Principios de seguridad en el desarrollo de software.*

Alcázar Villalobos, J. P., Carvajal Cascante, O., & Vallejo Esquivel, G. M. (2020). La banca costarricense y sus retos, desde una perspectiva de ética, imagen, credibilidad y confianza. *Revista Nacional de Administración,* 11(1), 33–46. https://doi.org/10.22458/RNA.V11I1.3010

Ambit BST. (2020). *Tipos de vulnerabilidades y amenazas informáticas.* https://www.ambit-bst.com/blog/tipos-de-vulnerabilidades-y-amenazas-inform%C3%A1ticas

Alcázar Villalobos, J. P., Carvajal Cascante, O., & Vallejo Esquivel, G. M. (2020). La banca costarricense y sus retos, desde una perspectiva de ética, imagen, credibilidad y confianza. *Revista Nacional De Administración*, *11*(1). https://doi.org/10.22458/rna.v11i1.3010

Banco Central de Costa Rica. (s. f.). *Políticas específicas de ciberseguridad.* https://www.bccr.fi.cr/transparencia-institucional/Doc_Seguridad_Informacion/Politicas_Especificas_Ciberseguridad.pdf

Building Security Maturity Model (BSIMM) Consulting Services | Black Duck. (n.d.). https://www.blackduck.com/services/security-program/bsimm-maturity-model.htmlhttps://www.blackduck.com/services/security-program/bsimm-maturity-model.html

CCN-CERT. (2023). *Recomendaciones sobre desarrollo seguro*. https://www.ccn-cert.cni.es/es/seguridad-al-dia/novedades-ccn-cert/12411-nuevo-informe-de-buenas-practicas-bp-28-recomendaciones-sobre-desarrollo-seguro-2.html

Ciberseguridad bancaria: avances, retos y prácticas recomendadas. (n.d.). https://www.iuvity.com/es/blog/ciberseguridad-bancaria-avances-retos-y-como-evitar-las-estafas

Delfino.cr. (2024, enero 13). BCCR confirma haber recibido ataque cibernético el pasado 12 de enero. *Delfino.cr*. https://delfino.cr/2024/01/bccr-confirma-haber-recibido-ataque-cibernetico-el-pasado-12-de-enero

Howard, M., & Lipner, S. (2006). *The Security Development Lifecycle: SDL: A Process for Developing Demonstrably More Secure Software.* Microsoft Press.

IBM. (2024). *¿Qué es el phishing?* https://www.ibm.com/es-es/topics/phishing

IBM. (2024). *¿Qué es el ransomware?* https://www.ibm.com/es-es/topics/ransomware

IBM. (2024). *¿Qué es el malware?* https://www.ibm.com/topics/malware

Innowise. (2023). *Ciberseguridad en el sector bancario.* https://innowise.com/es/blog/cybersecurity-in-banking/

ISO. (2024). ISO/IEC 27001:2022 - *Information security management systems.* Recuperado de ISO. NIST. (2023). Framework for Improving Critical Infrastructure Cybersecurity. Recuperado de NIST. SensorsTech. (2023). Estándares de seguridad: Todo lo que necesitas saber. https://sensorstechforum.es/cuales-son-los-estandares-de-seguridad/

ISO/IEC 27001:2022 - *Information security management systems.* (n.d.). https://www.iso.org/standard/27001

Jamil, F., Asif, M., Ashraf, M., Mehmood, A., & Mustafa, A. (2018). Cybersecurity threats and challenges.

Kamuni, A., Asfia, S., Sheikh, A., & Patel, S. (2019). *Cybersecurity in the modern world.*

Kaspersky. (n.d.). *¿Qué es la ciberseguridad?* Kaspersky. https://latam.kaspersky.com/resource-center/definitions/what-is-cyber-security?srsltid=AfmBOoqzXy9l27b91aqruNn4CyBsBNb0CNk4_hkv-clhT5ruENUd2KWm

Likert, R. (1932). A technique for the measurement of attitudes. *Archives of Psychology,* *22*(140), 1–55.

Microsoft. (2023). *Microsoft Security Development Lifecycle (SDL).* https://www.microsoft.com/en-us/securityengineering/sdlhttps://www.microsoft.com/en-us/securityengineering/sdl

Microsoft. (2024). *Security Development Lifecycle (SDL). Microsoft Security Engineering.* https://www.microsoft.com/en-us/securityengineering/sdl

McGraw, G. (2004). *Software Security: Building Security In.* Addison-Wesley.

Mohammed, N., Niazi, M., & Mahmood, A. (2017). *Creating secure software: A comprehensive approach.*

NIST. (2023). *Framework for Improving Critical Infrastructure Cybersecurity.* https://www.nist.gov/cyberframework

NIST. (2024a). *The NIST Cybersecurity Framework (CSF) 2.0.* National Institute of Standards and Technology. https://www.nist.gov/cyberframework

NIST. (2024b). *NIST Special Publication 800-218: Secure Software Development Framework (SSDF) Version 1.1.* National Institute of Standards and Technology. https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-218.pdf

OpenSAMM. (2010). *Software Assurance Maturity Model (SAMM) version 1.0 [PDF].* *https://opensamm.org/downloads/SAMM-1.0-es_MX.pdf*

Ortega, M. (n.d.). *Desarrollo seguro en ingeniería del software Aplicaciones seguras con Android, NodeJS, Python y C++.*

OWASP. (2020). *OWASP Software Assurance Maturity Model (SAMM) 2.0.* Open Worldwide Application Security Project. https://owaspsamm.org

OWASP. (2021). *Software Assurance Maturity Model (SAMM).* https://owasp.org/www-project-samm/https://owasp.org/www-project-samm/

Pérez Lastra, O., Gainza Reyes, D., & González Brito, H. R. (2023). *Resultados de la investigación sobre desarrollo seguro de software.* Serie Científica de la Universidad de las Ciencias Informáticas.

Perlroth, N., & Goldman, A. (2017). Equifax breach was 'entirely preventable,' report says. *The New York Times.* https://www.nytimes.com/2017/09/07/business/equifax-cyberattack.html

Poth, C. N. (Ed.). (2023). The Sage Handbook of Mixed Methods Research Design. Sage.

PowerDMARC. (2023). *Ciberseguridad en la banca: Principales amenazas y mejores formas de prevenirlas.* https://powerdmarc.com/es/cyber-security-in-banking/

Ramírez, A., Aiello, A., & Lincke, S. J. (2020). A survey and comparison of secure software development standards. *13th CMI Conference on Cybersecurity and Privacy - Digital Transformation - Potentials and Challenges, CMI 2020.* https://doi.org/10.1109/CMI51275.2020.9322704

Red Hat. (2022). *Seguridad en el ciclo de vida de desarrollo del software*. https://www.redhat.com/es/topics/security/software-development-lifecycle-security

Secureframe. (n.d.). *Modelo de Madurez en la Construcción de Seguridad (BSIMM).* https://secureframe.com/es-es/frameworks-glossary/bsimm

State of Cybersecurity Report 2021 | *4th Annual Report* | Accenture | Enhanced Reader. (n.d.).

Wee, A. Kudriavtseva and O. Gadyatskaya, "Have Heard of it": A Study with Practitioners on Adoption of Secure Software Development Frameworks, *2024 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, Vienna, Austria, 2024, pp. 626-633, doi: 10.1109/EuroSPW61312.2024.00076.

Anexos

- Anexo 1

Cuestionario para la recopilación de datos necesarios para el desarrollo del objetivo específico 1. El cuestionario fue validado mediante juicio de expertos en desarrollo de *software* del sector bancario costarricense.

Maestría en tecnologías de información y comunicación para la gestión organizacional
"Propuesta de un modelo metodológico para la gestión y adopción de prácticas seguras, en los equipos de desarrollo de software, en una entidad bancaria costarricense"

La presente encuesta pretende analizar el estado actual de las prácticas de desarrollo de *software* seguro en el sector bancario público costarricense, mediante la rrecopilación de información cualitativa sobre la percepción de prácticas actuales en desarrollo seguro.

Los datos aquí recopilados serán utilizados con fines académicos.

Sección 1: Información general del entrevistado

1. ¿Cuál es su posición en la entidad bancaria?
    a. Desarrollador de *software*
    b. Líder de equipo
    c. Jefe de desarrollo
    d. Especialista en seguridad informática
    e. Otro _____

2. ¿Cuánto tiempo lleva trabajando en desarrollo de *software* en el sector bancario costarricense?
    a. Menos de 1 año
    b. 1 a 3 años
    c. 4 a 6 años
    d. Más de 6 años

Sección 2: Percepción sobre la importancia de la seguridad en el desarrollo

3. En una escala del 1 al 5, ¿qué tan importante considera la seguridad en el proceso de desarrollo de *software* en su entidad?
    a. 1 = Nada
    b. 2 = Poco importante
    c. 3 = Moderadamente importante
    d. 4 = Muy importante

   e. 5 = Extremadamente importante
4. ¿En qué fases del ciclo de desarrollo se integran actualmente las prácticas de seguridad? (Seleccione todas las que apliquen)
   a. Diseño y planificación
   b. Desarrollo / codificación
   c. Pruebas y validación
   d. Despliegue
   e. Mantenimiento y actualizaciones
   f. No se aplican prácticas específicas de seguridad

Sección 3: Prácticas actuales y desafíos

5. Describa brevemente qué prácticas específicas de seguridad se utilizan en su equipo durante el desarrollo de *software*.
6. ¿Qué barreras o desafíos ha identificado en la implementación de prácticas de desarrollo seguro en su entidad?
7. En su experiencia, ¿qué factores facilitan la adopción de prácticas seguras en el desarrollo de *software*?

Sección 4: Conocimiento y uso de marcos de trabajo o estándares

8. ¿Está su equipo familiarizado con algún marco internacional o estándar relacionado con desarrollo seguro de *software*?
   a. OWASP SAMM
   b. Microsoft SDL
   c. NIST
   d. Otro (especificar): _____
   e. Ninguno

9. En una escala del 1 al 5, ¿qué tanto cree que el marco o estándar que usan se adapta a la realidad y necesidades de su entidad?
   a. 1 = Nada
   b. 2 = Poco importante
   c. 3 = Moderadamente importante
   d. 4 = Muy importante
   e. 5 = Extremadamente importante

Sección 5: Capacitación y cultura organizacional

10. ¿Recibe su equipo capacitación periódica en temas de seguridad informática relacionada con el desarrollo de *software*?
    a. Sí, de forma regular
    b. Sí, pero de forma esporádica o insuficiente
    c. No

11. En su opinión, ¿qué tan alineada está la cultura organizacional de su entidad con la importancia de la seguridad en el desarrollo de *software*?
    a. 1 = Nada alineada
    b. 2 = Poco alineada
    c. 3 = Alineación media
    d. 4 = Bien alineada
    e. 5 = Totalmente alineada

Sección 6: Sugerencias y cierre

12. ¿Qué recomendaciones haría para mejorar la adopción de prácticas de desarrollo de software *seguro* en su entidad?
13. Agradecería si deja un comentario adicional sobre las prácticas de desarrollo seguro o seguridad informática en el sector bancario costarricense

Muchas gracias por su colaboración. Sus respuestas serán de vital importancia para esta investigación que promueve un entorno más seguro, en el desarrollo de *software* del sector bancario costarricense.