UNIVERSIDAD NACIONAL

SISTEMA DE ESTUDIOS DE POSGRADOS

MAESTRIA EN ADMINISTRACIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN

PROPUESTA DE PROYECTO DE APLICACIÓN PRÁCTICA DE TECNOLOGÍA DE LA INFORMACIÓN

Adaptación de prácticas DevOps para aumentar el 'Scrum Velocity' del proyecto SecurityOps en Componentes Intel Costa Rica S.A.

Autor

Mauricio Piedra Hidalgo

Heredia, Costa Rica

Febrero, 2019

TABLA DE CONTENIDOS

Agosto 2019

Universidad Nacional
Facultad de Ciencias Exactas y Naturales
Escuela de Informática
Posgrado en Gestión de la Tecnología de Información y Comunicación (ProgesTIC)

FORMULARIO DE DEPÓSITO LEGAL, AUTORIZACIÓN DE USO DE DERECHOS PATRIMONIALES DE AUTOR E INCORPORACIÓN A REPOSITORIOS INSTITUCIONALES DE INFORMACIÓN DE ACCESO PÚBLICO

La persona abajo firmante, en condición de estudiante de la maestría en Administración de Tecnología de la Información (MATI) y autor del Trabajo final de graduación titulado: "Adaptación de prácticas DevOps para aumentar el 'Scrum Velocity' del proyecto SecurityOps en Componentes Intel Costa Rica S.A." para optar al grado académico de Máster en: Administración de Tecnología de la Información con énfasis en Administración de Proyectos, de conformidad con lo establecido en el documento de "Lineamientos generales para la realización del trabajo final de graduación" y demás normativa universitaria relacionada con estos trabajos de graduación, DECLARO BAJO FE DE JURAMENTO conociendo la responsabilidad civil, penal o administrativa en que podría(amos) incurrir al no decir la verdad, lo siguiente:

1. El documento, producto, obra audiovisual, software, resultado del trabajo final de graduación referido anteriormente es original, inédito y ha cumplido con todo el proceso de aprobación académico que confiere el grado académico postulado con esta obra.
2. El trabajo final de graduación referido anteriormente constituye una producción intelectual propia de la persona abajo firmante y a esta fecha no ha sido divulgado a terceros(as) de forma pública, por ningún medio de difusión impreso o digital.
3. Autorizo el depósito de un ejemplar en formato impreso y otro en formato digital (entregado en soporte de disco compacto), en la colección de trabajos finales de graduación del ProGesTIC de la Universidad Nacional, así como la realización de copias electrónicas adicionales para fines exclusivos de seguridad y conservación de la información.
4. En caso de que el trabajo final de graduación haya sido elaborado como obra en colaboración -bien se trate de obras en las que los autores(as) tienen el mismo grado de participación o aquellas en las que existe una persona autora principal y una o varias personas autoras secundarias-, todos(as) ellos(as) han contribuido intelectualmente en la elaboración del documento y en este acto, libero de responsabilidad a las autoridades del posgrado y a los funcionarios que custodian la colección del ProGesTIC, en relación con el

reconocimiento que se realiza respecto de los niveles de participación asignados por el propio autor del proyecto.

5. En caso de que el trabajo final de graduación haya sido elaborado como obras en colaboración (conforme a lo dispuesto en el punto 4), el autor abajo firmante Mauricio Piedra Hidalgo es el encargado de recibir comunicaciones y hace constar que el presente trabajo final no fue elaborado como obra en colaboración con otros autores.

6. Reconozco que la colección de trabajos finales del ProGesTIC no emite criterios ni valoraciones académicas sobre lo planteado en el producto final del trabajo de graduación y autorizo a esta dependencia para que proceda a poner a disposición del público la obra en mención, a través de los espacios físicos o virtuales que se posea, así como a través del Repositorio Institucional; a partir del cual los usuarios de dichas plataformas puedan acceder al documento y hacer uso de este en el marco de los fines académicos, no lucrativos y de respeto a la integridad del contenido del mismo así como la mención del autor o poseedor de sus derechos.

7. Manifiesto que todos los datos de citas dentro de texto y sus respectivas referencias bibliográficas, así como las tablas y figuras (ilustraciones, fotografías, dibujos, mapas, esquemas u otros) tienen la fuente y el crédito debidamente identificados y se han respetado los derechos de autor.

8. Autorizo la licencia gratuita no exclusiva de los derechos patrimoniales de autor para reproducir, traducir, distribuir y poner a disposición pública en formato electrónico, el documento depositado, para fines académicos, no lucrativos y por plazo indefinido en favor de la Universidad Nacional, que incluye además los siguientes actos:

a. La publicación y reproducción íntegra de la obra o parte de esta, tanto por medios impresos como electrónicos, incluyendo Internet y cualquier otra tecnología conocida o por conocer.

b. La traducción a cualquier idioma o dialecto de la obra o parte de esta.

c. La adaptación de la obra a formatos de lectura, sonido, voz y cualquier otra representación o mecanismo técnico disponible, que posibilite su acceso para personas invidentes parcial o totalmente, o con alguna otra forma de capacidades especiales que le impida su acceso a la lectura convencional del proyecto.

d. La distribución y puesta a disposición de la obra al público, de tal forma que el público pueda tener acceso a ella desde el momento y lugar que cada quien elija, a través de los mecanismos físicos o electrónicos de que disponga.

e. Cualquier otra forma de utilización, proceso o sistema conocido o por conocerse que se relacione con las actividades y fines académicos a los cuales se vincula la maestría, la colección de trabajos finales del ProGesTIC, la Escuela de Informática y la Universidad Nacional.

9. Reconozco que la colección de trabajos del ProGesTIC manifiesta actuar con diligencia para evitar la existencia en su sitio web de contenidos ilícitos y en caso de que tenga conocimiento efectivo de la existencia de infracciones a los derechos de propiedad intelectual, se reserva el derecho de proceder a bloquear el acceso durante el trámite del debido proceso para comprobar el incumplimiento y en caso de verificarse la falta, retirar definitivamente el acceso al proyecto depositado.

10. Acepto que la publicación y puesta a disposición del público del trabajo final de graduación, así como la presente autorización de uso de la obra, se regirá por la normativa institucional de la Universidad Nacional y la legislación de la República de Costa Rica. Adicionalmente, en caso de cualquier eventual diferencia de criterio o disputa futura, acepto que esta se dirimirá de acuerdo con los mecanismos de Resolución Alterna de Conflictos y la Jurisdicción Costarricense.

Autor: Mauricio Piedra Hidalgo

Firma: 

Fecha de entrega: 20 de agosto de 2019

Correo electrónico: mau.piedra.h@gmail.com

RESUMEN EJECUTIVO

SecurityOps es un proyecto utilizado como repositorio para investigaciones de casos de seguridad informática de la empresa Intel® y se encuentra ubicado en el departamento de Seguridad Informática. La herramienta es utilizada por más de cien usuarios en la empresa y tiene una gran relevancia para esta, pues en ella se encuentra información sensible que ocupa un alto grado de disponibilidad y un constante manejo de las mejoras que los clientes solicitan frecuentemente.

Los integrantes del proyecto han identificado problemas con respecto a la cantidad de trabajo que se entrega al cliente final, la cual tiene un grado de estancamiento y no se está cumpliendo con la expectativa de entrega de los clientes, afectando su modelo del negocio.

Por lo tanto, la investigación consiste en una estrategia para implementar un marco de trabajo que permita mejorar esos números de entrega, utilizando la misma cantidad de miembros en el equipo y evitando la contratación de más plazas para el mismo. El marco de trabajo propuesto es DevOps, el cual procura combinar las actividades de desarrollo y operaciones, donde no existan miembros del equipo específicos para cada tarea, sino que todos tengan las capacidades y herramientas para ejecutar cualquier actividad en el equipo.

A partir de los hallazgos identificados en el escenario previo a la investigación, se logra evidenciar una marcable deuda técnica en el equipo, lo que significa que no todos los miembros del mismo pueden trabajar ciertas tareas, lo cual crea un sobrecargo en ciertos recursos, por ende, retrasos en la entrega. Además, se evidencian otras falencias que afectan la velocidad de la productividad del proyecto.

Finalmente, el desarrollo de la propuesta solución se basa en la implementación del marco de trabajo DevOps, implementación que se divide en tres grandes pilares, los cuales están compuestos por actividades específicas que sumadas y ejecutadas dan como resultado la obtención del objetivo principal de cada uno de esos pilares. Se plantean las conclusiones y recomendaciones del caso, así como las acciones futuras que deben tomarse en consideración para la implementación de otros procesos según las prácticas del marco de trabajo seleccionado.

CAPÍTULO 1: EL PROBLEMA Y SU IMPORTANCIA

1.1 Antecedentes

En este apartado se presentan los principales antecedentes que motivan este proyecto. El apartado inicia con una contextualización del entorno del proyecto para desarrollar luego aquellos elementos que permitirán al lector comprender el problema que se atenderá en el desarrollo del proyecto.

SecurityOps es un programa utilizado como repositorio para investigaciones de casos de seguridad informática de la empresa. Este proyecto se encuentra ubicado en el departamento de Seguridad Informática de la empresa. SecurityOps es un programa utilizado por más de cien usuarios en la empresa. Está información fue extraída de sesiones realizadas con la Product Owner del equipo entre el 13 y 19 de abril del 2019.

Intel® actualmente está transformando su estrategia de negocio para migrar de ser una empresa enfocada en “Hardware”, mediante una intensiva creación de microcomponentes, y abarcar más espacios de desarrollo y convertirse en una empresa de data, según las palabras del CEO de la empresa, Bob Swan (Fortune, 2019). Es por eso que varios equipos dentro de la compañía han migrado su forma de trabajar, incluido el equipo de trabajo del proyecto SecurityOps, es por esa razón que el equipo trabaja bajo una práctica ágil: Scrum.

El equipo a cargo de esta plataforma tiene que lidiar con varios temas, por ejemplo, el desarrollo y programación de nuevos módulos de SecurityOp, así como la implementación de la base de datos y la conexión del servicio a la red interna de la empresa, siguiendo las pautas de la misma. El equipo actualmente trabaja bajo la metodología ágil Scrum, se compone de una Scrum Master, una Product Owner y cuatro implementadores, los cuales tienen roles específicos, un líder técnico, un programador, una analista de sistemas y una encargada de hacer las pruebas de calidad (A estas cuatro personas se les referirá en este documento como Scrum Team). Está información fue extraída de sesiones realizadas con la Product Owner del equipo entre el 13 y 19 de abril del 2019.

El soporte de SecurityOps supone todo un reto para el Scrum Team, pues muchas de las tareas a realizar en futuras iteraciones, están totalmente fuera de las funciones normales del equipo. Supondrá abarcar nuevos campos de la tecnología de la información, puesto que se tendrá que configurar redes, sistemas de fallo, replicación, seguranza de la información, entre otros desafíos que le esperan al equipo.

El Scrum Team se encarga también de soportar las incidencias diarias de los usuarios de la aplicación, las cuales pueden ser fallas en el sistema en producción o dudas con respecto al uso del mismo, estas peticiones que realizan los usuarios se les conoce como tiquetes.

Muchos de estos usuarios también solicitan nuevas funcionalidades en la aplicación, las cuales conllevan a la programación e integración de módulos para suplir sus necesidades, por ejemplo, sistemas de notificaciones de correo electrónico, recordatorios, formularios, entre otras funcionalidades. Este proyecto busca que todos los miembros del Scrum Team se encarguen no sólo de soportar tiquetes, sino también que sean capaces de agregar y programar nuevas funcionalidades a la herramienta que está en producción y a la que se estará migrando.

DevOps tiene como gran ventaja aumentar la productividad y eficiencia de un equipo de Scrum, pues no se presentan casos donde algún miembro deba esperar a otro terminar una tarea asignada (Riungu-Kalliosaari, Mäkinen, Lwakatare, Tiihonen, Männistö, 2016), sin embargo, se presenta un gran desafío ya que no todos en el Scrum Team tienen experiencia como programadores o en caso contrario, no todos tienen la práctica necesaria para atender las incidencias de los clientes de la herramienta.

1.2 Descripción y delimitación del problema

Actualmente, el Scrum Team, en el cual se quiere integrar las prácticas de DevOps, desempeña sus tareas bajo la práctica ágil Scrum. El Scrum Team lo componen seis miembros: Una Scrum Master, una Product Owner, un líder técnico, un desarrollador, una administradora de sistemas y una analista de calidad. (Cohen, 2010)

Como señala el párrafo anterior, solo se cuenta con un analista de calidad, el cual muchas veces se ve sobrecargado con la cantidad de historias de usuario que debe revisar y aprobar durante el Sprint, esta sobrecarga hace que muchas historias de usuario no se puedan efectuar en el Sprint planeado, y se deban trasladar a otro, lo que significa un atraso en la puesta en producción de la misma. Este escenario se repite en el caso del desarrollador y líder técnico, quienes se saturan con tareas que solo ellos tienen el conocimiento para realizar dentro del equipo.

Este problema afecta directamente al cliente final de la aplicación, el cual no ve las historias de usuario propuestas ser entregadas y en producción en el tiempo deseado. El impacto asociado al retraso en la entrega de las historias de usuario es directamente al negocio del cliente, pues estas tareas buscan integrar nuevas funciones o corregir defectos de la

aplicación SecurityOps, procurando agilizar el día a día de sus clientes. Indirectamente, la imagen del Scrum Team se ve afectada, puesto no es capaz de entregar el trabajo comprometido en el tiempo establecido.

1.3 Justificación

SecurityOps actualmente reside en una plataforma cuestionada por los clientes, con retrasos en la entrega de requerimientos propuestos por los clientes de la aplicación y con un equipo de soporte limitado en presupuesto y capacidades técnicas. Está información fue extraída de sesiones realizadas con la Product Owner del equipo entre el 13 y 19 de abril de 2019. Migrar SecurityOps a una nueva plataforma procura solucionar los problemas indicados anteriormente.

Es en ese proyecto de migración de la plataforma a una nueva, donde la metodología DevOps permite maximizar los recursos del Scrum Team, donde no se ocuparía contratar nuevo personal y es el complemento perfecto para la metodología ágil utilizada por el equipo para entregar de manera más rápida y oportuna cada una de las historias planteadas para completar de manera exitosa la migración de la plataforma. La velocidad en que se pueden cumplir los entregables de la migración utilizando DevOps sería clave para evitar más reclamos por parte de los clientes de la plataforma. Intel® ganaría con la utilización de DevOps en este proyecto evitar empezar de cero con una nueva plataforma que dé solución a los problemas actuales, sino también por formar al equipo existente, en uno capaz de soportar la migración y la futura mantención de la herramienta sin invertir en la contratación de recurso técnico.

1.4 Objetivos generales y específicos

1.4.1 Objetivo general

Proponer una estrategia de implementación de la práctica DevOps para aumentar el 'Scrum Velocity' del proyecto SecurityOps, con el fin de aumentar la entrega de historias de usuario en el Scrum Team.

1.4.2 Objetivos específicos

En este apartado se presentan los objetivos específicos del proyecto sometido a consideración.

***Objetivo específico 1*:** Determinar los conceptos que la práctica DevOps propone para agilizar proceso de desarrollo y ejecución de las historias de usuario en un equipo Scrum, mediante una revisión de la literatura asociada a lo tratado en este proyecto para contar con referentes conceptuales y teóricos sólidos a la hora de crear un plan de implementación de DevOps en el problema a tratar.

***Objetivo específico 2*:** Diagnosticar el estado del Scrum Team en cuanto a la cantidad de historias de usuario que son capaces de finalizar en un Sprint mediante la revisión y comparación de las estadísticas que arrojan los históricos de los Sprints para reconocer los cuellos de botella que generan los atrasos descritos en el problema en el proyecto SecurityOps.

Objetivo específico 3: Proponer una estrategia para aumentar el 'Scrum Velocity' del proyecto SecurityOps por medio de la aplicación de los fundamentos de la práctica DevOps durante la ejecución de los Sprints para aumentar la entrega de historias de usuario que los clientes tienen.

***Objetivo específico 4*:** Probar la implementación de la práctica DevOps en el Scrum Team mediante un plan piloto para recolectar retroalimentación que permita afinar "la solución" propuesta en el proyecto SecurityOps.

***Objetivo específico 5*:** Proponer una hoja de ruta de la implementación pragmática de la práctica DevOps mediante las lecciones aprendidas del plan piloto para resolver los atrasos en las entregas de las historias de usuario que presenta en el Scrum Team.

1.5 Alcances y beneficios del proyecto

En este apartado, para cada uno de los objetivos planteados se detallan las metas por alcanzar.

Objetivos	Metas Preliminares
OP	· Presentar una solución factible al Scrum Team para la solución del problema descrito anteriormente.
OE1	· Lograr un conocimiento sólido de los referentes conceptuales y teóricos de la solución a implementar. · Lograr un conocimiento sólido de los referentes conceptuales y teóricos del problema a tratar.
OE2	· Lograr una medición tangible de los principales índices que el Scrum Team cuenta, para poder tener referencia sólida para partir de las mismas en la estrategia a seguir y en las futuras comparaciones después de implementada la solución.
OE3	· Aumentar el 'Scrum Velocity' del equipo. · Elevar la cantidad de Releases (Cohen, 2010) del proyecto SecurityOps que el 'Scrum Team' trabaja.
OE4	· Contar con la retroalimentación necesaria que brinda una prueba de la solución en la práctica diaria del Scrum Team.
OE5	· Proponer una hoja de ruta para implementar DevOps como práctica que aumentara el número de historias de usuario que el Scrum Team realiza en un Sprint.

Tabla 1.5.1 (Elaboración propia)

CAPÍTULO 2: MARCO TEÓRICO O CONCEPTUAL

2.1 Definiciones

- ***Metodología**:* Enfoque documentado para realizar actividades en una manera coherente, consistente, contable y que permita ser reproducida. (Draffin, 2010)
- *Marco de Trabajo (Framework):* Estructura lógica para clasificar y organizar información compleja. Es una guía que no exige ser reproducida fielmente. (Draffin, 2010)
- ***Práctica**:* Aplicación o uso de una idea, creencia o método. (Oxford, 2019)
- ***Técnica**:* "Conjunto de procedimientos o recursos que se usan en un arte, en una ciencia o en una actividad determinada, en especial cuando se adquieren por medio de su práctica y requieren habilidad". (Press, 2018)
- ***Cultura Organizacional**:* Patrón general de conductas, creencias y valores compartidos por los miembros de una organización (Salazar, Guerrero, Machado, Cañedo, 2009).
- ***Conocimiento**:* Combinación experiencia, valores, información contextual y perspectiva experta que proporciona un marco para evaluar e incorporar nuevas experiencias e información. En las organizaciones, a menudo se captura en documentos, repositorios, rutinas organizativas, procesos, prácticas y normas. (Davenport & Prusak, 1998)
- ***Equipo Multifuncional**:* Un equipo multifuncional está compuesto idealmente por desarrolladores, analistas, evaluadores e individuos de otras disciplinas que pueden contribuir al éxito de un proyecto a través del intercambio activo de conocimientos dentro del mismo. (Dorairaj, Noble & Malik 2012)
- ***Metodologías Tradicionales**:* Las metodologías tradicionales de desarrollo de software son orientadas por planeación. Inician el desarrollo de un proyecto con un riguroso proceso de elicitación de requerimientos, previo a etapas de análisis y diseño. Se concibe un solo proyecto, de grandes dimensiones y estructura definida; se sigue un proceso secuencial en una sola dirección y sin marcha atrás. El proceso es rígido y no cambia. (Khurana, Sohal, 2011)
- ***Extreme Programming (XP)**:* XP es una parte de la familia de metodologías ágiles de desarrollo de software. XP está diseñada para entregar el software que los clientes necesitan en el momento en que lo necesitan. XP alienta a los desarrolladores a responder a los requerimientos cambiantes de los clientes, aún en fases tardías del ciclo de vida del desarrollo. (Beck, 2002)
- ***Kanban**:* Los enfoques Lean y Kanban se introdujeron en la industria manufacturera japonesa en la década de 1950. Kanban es una palabra japonesa que significa un letrero, y se usa en la fabricación como un sistema de programación. (Ahmad, Markkula, Oivo, 2013)

- ***Programador**:* Persona que elabora programas de computadora. (RAE, 2019)
- ***Tester (QA):*** Persona que lleva a cabo análisis informáticos. (RAE, 2019)
- ***Diseñador de UI**:* Persona que elabora las interfaces de usuario de un programa.
- ***Definition of Done**:* Criterio de aceptación de una historia de usuario.
- ***Clientes finales**:* Usuarios no pertenecientes al equipo de trabajo que utilizan la aplicación y sus funcionalidades.
- ***Producto**:* Cambios o funcionalidades nuevas que se deben ejecutar en el ambiente de producción como parte de un requerimiento del cliente final.
- ***KTBR**:* Abreviación en inglés para "Keep The Business Running", significan las operaciones y tiquetes que se reciben por parte de los clientes para mantener los sistemas de producción y sus aplicaciones en ejecución.
- ***Actividad de Operaciones:*** Actividades de KTBR.
- ***Deuda Técnica**:* Indica la cantidad de trabajo de desarrollo adicional que se acumula cuando se implementa una mala solución (Vega, 2017). Las siguientes son otras causas comunes de deuda técnica:
    - Falta de colaboración.
    - Falta de documentación.
    - Falta de comprensión.
- ***CI**:* Integración continua. Se refiere a integrar lo más prontamente posible los cambios al producto.
- ***CD**:* Entrega continua. Es una estrategia que busca que cualquier cambio en el código que haya pasado satisfactoriamente los controles de calidad, sea automáticamente puesto en el ambiente de producción.

2.2 Metodologías Ágiles

La entrega de producto de manera ágil y eficiente en el mercado es una demanda actual de varias empresas de tecnología, las cuales están migrando a las metodologías ágiles para satisfacer esa necesidad (Reifer, 2002). Las metodologías tradicionales se han visto seriamente cuestionadas en su eficacia a la hora de entregar producto de manera rápida en sus proyectos, eso lo constata un estudio realizado por Standish Group en 2007, el cual determina que el 46% de los proyectos de software excedieron sus tiempos de entrega originales. (Cerpa & Verner, 2009)

A diferencia de los proyectos basados en metodologías tradicionales, las Metodologías Ágiles (Agile) generan constantes entregas del producto en fases tempranas del proyecto. Existen varias metodologías y Marcos de Trabajo que pertenecen a la familia Agile, por ejemplo, Scrum, Extreme Programming (XP) o Kanban (Matharu, Mishra, Singh & Upadhyay, 2015).

Estas metodologías utilizan técnicas que fomentan la colaboración del equipo, tales como Pair Programming o Refactoring, conceptos profundizados más adelante en este documento. Además de integrar al cliente en las reuniones diarias del equipo (Reifer, 2002). Por esta razón, las metodologías ágiles sacan ventaja sobre las tradicionales y son preferidas para proyectos cuyo enfoque se basa en muestras inmediatas del producto final.

2.3 Scrum

Scrum es un Marco de Trabajo que pertenece a la familia Agile, que se utiliza para el control de proyectos de Software y su desarrollo es por medio de prácticas iterativas e incrementales (Advanced Development Methods, 2007). Scrum premia la velocidad y agilidad a la hora de entregar producto, por lo tanto, la documentación requerida en este Marco de Trabajo es poca, y es el mismo ciclo de vida del proyecto el que genera los datos que se utilizan para documentar el trabajo (Schwaber, 2004). Existe un dilema entre varios autores del tema sobre si Scrum es una metodología o un Marco de Trabajo. Algunos autores consideran Scrum como una metodología puesto que tiene herramientas y roles determinados (Schwaber & Beedle, 2002), sin embargo, otros autores le dan a Scrum el título de Marco de Trabajo, puesto que no sigue una secuencia de instrucciones de manera estricta para definir las fases de vida de un proyecto, permitiendo modificar los conceptos de acuerdo a la situación y ambiente en que se desarrolla el proyecto. (Schwaber & Sutherland, 2013).

2.4 Roles de Scrum

Scrum se compone de roles con diferentes responsabilidades definidas para poder lograr su propuesta. Mahalakshmi & Sundararajan (2016) citan los siguientes roles:

- *Product Owner*: Encargado de capturar los requerimientos del proyecto por parte del cliente, también crea las llamadas historias de usuario y los objetivos del retorno de inversión (Schwaber, 2004). Es el rol que trabaja directamente con el cliente final, mantiene el Product Backlog y prioriza las historias de usuario.
- *Scrum Team (Equipo de Trabajo Scrum)*: Implementa las funcionalidades descritas en los requerimientos planteados por el Product Owner (Schwaber, 2004). Los miembros del equipo pueden ser programadores, testers, diseñadores de UI, entre otros.
- *Scrum Master*: Está a cargo del manejo del equipo, balancear cargas de trabajo, remover obstáculos que los miembros del equipo tengan, negociar con el Product Owner situaciones puntuales entre el trabajo por realizar y el Scrum Team (Schwaber, 2004). También es el puesto encargado de darle el seguimiento y almacenaje requerido de la documentación que Scrum genera en el proyecto trabajado.

2.5 Ceremonias de Scrum

Son reuniones críticas que se realizan en distintos momentos de las iteraciones con un sentido específico cada una. Mahalakshmi & Sundararajan (2016) mencionan que Scrum cuenta con las siguientes reuniones o ceremonias:

- *Daily Stand Up Meeting:* Reuniones diarias con una duración aproximada de 15 minutos donde cada miembro del equipo menciona en lo que trabajó el día antes, lo que trabaja en el presente día y si existe algún impedimento que imposibilite la continuación de la historia de usuario en la que esté trabajando.
- *Reunión de Retrospectiva:* En esta ceremonia, se analiza la manera en que se trabajó durante la iteración y resultados obtenidos. Estas iteraciones son conocidas como Sprints y es el espacio de tiempo donde se inician y entregan las historias de usuario definidas.
- *Sprint Planning:* Ceremonia en la cual se asignan las historias de usuario a trabajar en el Sprint a los miembros del Scrum Team.
- *Grooming:* En esta reunión, el equipo de trabajo junto al Scrum Master le da un valor de esfuerzo, reflejado por medio de un sistema de puntaje, a cada una de las historias de usuario creadas por el Product Owner. Cada punto de valor refleja el tiempo y dificultad que tomará lograr completar la historia de usuario. (Harvie & Agah, 2016)

2.6 Artefactos de Scrum

Los artefactos en Scrum se refieren a las herramientas con las que cuenta este Marco de Trabajo y a la postre son parte fundamental de la documentación que se genera en un proyecto de Scrum. Mahalakshmi & Sundararajan (2016) mencionan los siguientes artefactos:

- *Product Backlog:* Es una lista ordenada por prioridad con todas las historias de usuario del proyecto creadas por el Product Owner, basados en los requerimientos capturados mediante sus reuniones con los clientes finales del proyecto. Se permite priorizar nuevamente el orden de esas historias de acuerdo a las condiciones del proyecto.
- *Sprint Backlog*: Es una lista que contiene las historias de usuario que se van a trabajar durante la iteración presente del proyecto. Se consideran como las historias de usuario activas y tienen un miembro del equipo asignado para su ejecución.
- *Burndown Chart:* Es una gráfica que se actualiza diariamente y que muestra el trabajo concluido y restante del Sprint. Es importante para mostrar de manera sencilla y visual el progreso del proyecto.

- *Historias de Usuario:* Son los requerimientos del proyecto, recopiladas por el Product Owner por medio de sus entrevistas y reuniones con los clientes del proyecto. Una historia de usuario tiene un ciclo de vida: No iniciada, iniciada y aceptada (Finalizada).

2.7 Métricas de Scrum

Las métricas que arroja un proyecto que trabaja bajo el Framework Scrum, da a conocer la salud del proyecto, la capacidad de trabajo y los puntos débiles y fuertes del equipo. La tabla siguiente muestra diferentes métricas de Scrum.

Métrica	Descripción	Fórmula
Scrum Velocity	Suma de todos los estimados de las historias aceptadas en una iteración	V=∑ Historias de Usuario aceptadas
Capacity	Suma de todos los estimados de las historias trabajadas en una iteración, sin importar si se concluyeron o no	C=∑ Historias de Usuario
Factor de Enfoque	Capacidad de entrega de un equipo	V÷C
Porcentaje de Trabajo Iniciado		∑(Estimado Original de Trabajo Iniciado) ÷ (Trabajo original previsto para el Sprint)
Incremento de Valor entregado		Velocity del Sprint actual ÷ Velocity del primer Sprint
Sprint Ganado/Perdido	Un Sprint se considera ganado si un mínimo del 80% de las historias de usuario previstas son aceptadas.	

Tabla 2.7.1 (Downey & Sutherland, 2013)

2.8 Ventajas de utilizar Scrum

Scrum es una práctica muy conveniente para equipos pequeños, de ocho a diez miembros, y con proyectos con plazos cortos de entrega y anuentes a cambios constantes en los requerimientos planteados (Rising & Jannoff, 2000). Las siguientes son algunas de las ventajas que la aplicación de este Framework genera en un proyecto:

- La metodología propone una serie de tareas realizables en cada Sprint (Rising & Jannoff, 2000), lo que genera una entrega e implementación rápida de las mismas para la consecución del proyecto. En cierto punto, el manejar tareas realizables, genera en el equipo un sentimiento de éxito, puesto que no se estanca en la misma durante mucho tiempo.
- Aun teniendo constantes cambios en los requerimientos por parte del cliente, el progreso en el proyecto es visible y continuo.
- Scrum da una mejor perspectiva con respecto al avance del proyecto a los clientes y al equipo de trabajo (Rising & Jannoff, 2000). Esto se debe a las ceremonias propias de la metodología, que visibiliza a todos los involucrados el estado de cada una de las tareas en las que se está trabajando en el momento.
- Mejora la comunicación dentro del equipo (Rising & Jannoff, 2000). Enfatizan prácticas para compartir conocimiento, por ejemplo, Pair Programming, donde entre los miembros del equipo, se ayudan mutuamente en áreas donde exista una oportunidad de mejora entre los miembros del proyecto.
- Se reconoce el éxito y las falencias del equipo. Scrum propone mediante sus instrumentos y reuniones, un análisis constructivo de las oportunidades de mejora con respecto al trabajo realizado previamente, lo cual visibiliza problemas y soluciones en el proyecto.
- Los clientes obtienen constantes notificaciones del estado del producto, además, el marco de trabajo Scrum permite a los clientes finales avanzar en sus propias tareas en las fases entregadas del proyecto y puestas en producción.
- Se crea una cultura de confianza y conocimiento entre clientes y el equipo (Rising & Jannoff, 2000). La constante intervención entre los involucrados en el proyecto, sumado a las funciones propias del Scrum Master y Product Owner, convergen en un conocimiento más amplio del negocio por parte de todos los actores.

2.9 Desventajas de utilizar Scrum

Segun Mona Singh, en su artículo “U-SCRUM: An Agile Methodology for Promoting Usability” (2008), Scrum falla a la hora de la formulación adecuada de las historias de usuario. Singh (2008) menciona las siguientes razones:

- Un Product Owner suele complacer las demandas del cliente y no dedica el tiempo adecuado a crear historias de usuario enfocadas en usabilidad.
- No es una obligación del Product Owner tener un amplio conocimiento técnico a la hora de desarrollar historias de usuario, lo cual puede afectar el diseño de las mismas.
- Las metodologías ágiles le dan poco espacio a la especificación de las historias de usuario, las cuales pueden tener problemas de coherencia a la hora de implementarlas.
- Al contar con roles específicos en el equipo (Programadores, encargado de base de datos o control de calidad) muchas historias de usuario pueden quedar estancadas en alguna etapa.
- Juyun Cho (2008) menciona en su artículo “Issues and challenges of Agile Software Development with Scrum”, el desafío que los miembros del Scrum Team, en especial aquellos con poca experiencia, encuentran a la hora de tomar una historia de usuario sin una documentación fuerte y detallada, puesto que no se tiene muy claro la dificultad de la misma y las estimaciones del esfuerzo que esa historia conlleva no es acertada.
- Tener un miembro del Scrum Team dedicado a una sola tarea, por ejemplo, un programador o un analista de calidad, no permite tener una perspectiva global de todas las historias de usuario del proyecto, cayendo en un individualismo que construye por separado sin entender el objetivo general. (Cho, 2008)
- Si el equipo no está al tanto de las historias que sus compañeros están trabajando, resulta complejo entender las historias de usuario aún no trabajadas que están en el backlog y que dependen o se integran con las historias ya completadas.
- Dependencias con otros equipos fuera del Scrum Team repercuten en la velocidad de entrega del mismo. Por ejemplo, depender de un equipo de Network que habilita puertos de red, o un equipo al cargo del soporte y monitoreo de los sistemas y el hardware, son cosas que no se pueden controlar bajo un proyecto de Scrum y se cuenta como un riesgo.
- Al existir esta inexperiencia en ciertos miembros del grupo, se retrasa la entrega de historias de usuario y finalmente la entrega del proyecto, puesto que los miembros que sí entiendan las mismas, tendrán que dedicar tiempo para explicar las historias a los miembros que no las dominan. (Cho, 2008)

- Un punto primordial de reducir la documentación en Agile es mantener a todos los miembros de un equipo en un nivel balanceado sobre el conocimiento del sistema y de las habilidades que se ocupen, por lo tanto, si una persona abandona el equipo, esta no será difícil de reemplazar. (Cho, 2008)

2.10 DevOps

Como propuesta para atacar las desventajas previamente mencionadas cuando se utiliza Scrum como Marco de Trabajo, la práctica DevOps reduce el tiempo entre ejecutar un cambio al producto y ponerlo en producción, asegurando una alta calidad. (Ebert, Gallardo, Hernandez & Serrano, 2016)

DevOps es una cultura organizacional que fusiona las tareas de desarrollo y operaciones (Development y Operations, lo cual da origen a su nombre) mediante la automatización, entrega y monitoreo de infraestructura. Se desprende completamente de la idea de tener silos separados de trabajo que desarrollan tareas específicas, y propone una organización múltiple que trabaja en la entrega de producto de manera continua. (Ebert, Gallardo, Hernandez & Serrano, 2016)

La práctica de tener silos completamente separados para cada labor supone facilitar el manejo de los equipos, pues, al estar más divididos, es muy simple notar los problemas en la cadena de entrega de un producto, que equipo hace bien su trabajo y cuál debe reforzarse. Es ahí donde se termina dividiendo el trabajo en actividades relacionadas al desarrollo, diseño de interfaz, diseño de base de datos y pruebas. Por otra parte, existen otros miembros del equipo soportando las actividades de operaciones diarias, tales como redes, seguridad, almacenamiento o soporte al cliente que crecen de manera separada a través del desarrollo del proyecto, aumentando su complejidad una vez implementado e iniciado el mismo. (Hutterman, 2012)

Cuando existe esta separación, es donde chocan estos dos conceptos, puesto que las actividades de Desarrollo ocupan introducir cambios al proyecto, mientras las actividades de Operaciones no están al tanto de cómo enfrentar esos nuevos cambios (Need For Change vs. Fear of Change). Esta situación ocasiona un estancamiento en el movimiento continuo del proyecto, y este conflicto es algo que las Metodologías Ágiles no contemplan (Hutterman, 2012), en el particular el Marco de Trabajo Scrum no brinda una solución propia a este conflicto. Por esta razón, DevOps funciona como complemento perfecto a Scrum para solventar los desafíos previamente descritos.

DevOps es una práctica que promueve una rápida entrega de historias de usuario y mejora la comunicación entre el equipo, además de actuar bajo un lema de “Fallar Rápido” para una pronta detección y resolución de problemas (Ebert, Gallardo, Hernandez & Serrano, 2016). Compañías como Amazon y Google son líderes en la implementación de DevOps en sus equipos de trabajo. (Ebert, Gallardo, Hernandez & Serrano, 2016)

2.11 Principios y conceptos de DevOps

DevOps se basa en una serie de principios orientados a la entrega de producto final, donde su principal foco es la velocidad de entrega, según Virmani (2015) estos principios son:

- El control de calidad continuo en ambientes similares al ambiente de Producción.
- La movilidad en el proyecto, contar con historias de usuario que puedan cambiar de estado en su proceso de ejecución diariamente (Inicio, desarrollo y entrega).
- Retroalimentación continua del equipo y del proyecto.
- Habilidad de reaccionar al cambio de manera rápida y conformar un equipo orientado a la resolución de metas en vez de tareas.

DevOps extiende los principios de Agile a cada una de las fases de vida de un proyecto de software, permitiendo a un equipo convivir sanamente entre operaciones y desarrollo. (Virmani, 2015)

Los siguientes son conceptos fundamentales de acuerdo a Virmani (2015) en la aplicación de DevOps a las fases del desarrollo de un software.

- *Planeación Continua:* Es difícil para un equipo que desarrolla y también controla la calidad de los cambios a producción estar al tanto de las nuevas condiciones que imponen los cambios continuos en los requerimientos del cliente. DevOps permite enfrentar esos cambios mediante un Product Backlog priorizado y un canal de comunicación continuo con los clientes. Se propone un ciclo de ejecución, recibir retroalimentación, reaccionar a la retroalimentación y ajustar el planeamiento del trabajo cuando sea necesario.
- *Integración Continua*: Se refiere a integrar lo más prontamente posible los cambios al producto. Compartir el cambio con el equipo y monitorear cómo se comporta continuamente dentro de un sistema similar a producción. Este concepto de DevOps también hace alusión a la automatización, ya que, al incorporar rápidamente un cambio, las alertas de un posible mal funcionamiento deben activarse para indicar la falencia y proceder con una satisfactoria solución.

- *Entrega Continua*: Es una estrategia que busca que cualquier cambio en el código que haya pasado satisfactoriamente los controles de calidad, sea automáticamente puesto en el ambiente de producción. Es crucial contar con el hardware ideal y herramientas necesarias para poder garantizar este movimiento.
- *Testing Continuo*: Propone recibir lo más pronto posible los cambios en el código que deban ser pasados a un control de calidad. Se prioriza el control de calidad automatizado, que permita mover el cambio una vez haya sido aprobado a los sistemas de producción.
- *Monitoreo Continuo*: Al estar los cambios aprobados en producción, el monitoreo de los mismos debe ser continuo y automatizado. Es una oportunidad para valorar qué tan efectivo son los sistemas utilizados en la fase de control de calidad.

Al conocer los principios en los que DevOps se basa, la siguiente pregunta lógica es saber el tipo de equipo y herramientas que se deben tener para lograr un ambiente de DevOps óptimo. DevOps no propone una arquitectura ideal para lograr la correcta aplicación de esta cultura en un proyecto, pues cada proyecto presenta su propio ambiente con parámetros diferentes, y deja la elección de esa arquitectura y herramientas totalmente a discreción del proyecto. (Hutterman, 2012)

2.12 Generalistas, Especialistas y Equipos Multifuncionales

DevOps no especifica contar con un equipo generalista o especialista, definiendo el primero como un equipo multidisciplinario, donde cada miembro tiene la capacidad para realizar cualquier tarea que esté en el alcance del proyecto (Sohaib & Khan, 2010), y el segundo, los especialistas, donde cada miembro tiene un rol específico y se especializa en una función (Microsoft, 2014). Las metodologías ágiles suelen preferir los equipos generalistas (Parsons, Lal, Ryu, Lange, 2017), puesto que brindan una mejor respuesta al cambio y reduce el tiempo de entrega. (Parsons, Lal, Ryu, Lange, 2017)

Un problema con los equipos especialistas es el factor de disponibilidad, ya que al combinarse con un Framework como Scrum, que prioriza la entrega continua mediante historias manejables, el tener la necesidad de que un rol especialista se encargue de la revisión, guianza o desarrollo en general de un algoritmo, representa un posible retraso en el sistema propuesto por Scrum y su complementación con DevOp. (Dorairaj, Noble & Malik 2012)

Para desarrollar el potencial de un equipo Scrum, el concepto de equipo generalista apoya en gran medida los resultados finales. Dorairaj, Noble & Malik (2012) mencionan que un equipo Agile es multifuncional, el éxito del desarrollo del producto es mediante la colaboración y comunicación efectiva entre los miembros del equipo y también de los clientes.

Los miembros del equipo Scrum comparten de manera frecuente conocimiento específico que es esencial para entregar a los clientes, es por eso que el enfoque en depurar la manera en que se comparte el conocimiento es imperativo para el éxito de los proyectos basados en Metodologías Ágiles. (Melnik & Maurer, 2004)

2.13 Construcción de conocimiento un Equipo Multifuncional

Los siguientes conceptos son un recopilado de técnicas y actividades para generar conocimiento entre los miembros de un equipo de trabajo.

- *Talleres de Partida:* Los Talleres de Partida es una serie interactiva de talleres estructurados que ofrece oportunidades para que los clientes cristalicen sus ideas en colaboración con los equipos de desarrollo (Smith, 2005). Estos talleres concluyen con una visión compartida para el proyecto y prepara al equipo de desarrollo para las iteraciones de desarrollo Agile. (Coram & Bohner, 2005)
- *Cliente Colaborador*: Las responsabilidades de un cliente incluyen impulsar el proyecto de desarrollo de software, proporcionar los requisitos del proyecto y realizar pruebas de aceptación (Martin, Biddle & Noble, 2004). El cliente colabora con los equipos de proyecto para mejorar la colaboración comercial y técnica en un proyecto y establece la dirección del proyecto para determinar qué y cuándo construir. (Martin, Biddle & Noble, 2009)
- *Sesiones de Entrenamiento:* Mediante capacitaciones formales, los gerentes pueden estandarizar el contenido y las prácticas de múltiples equipos en una organización. Las sesiones de entrenamiento son beneficiosas para difundir el conocimiento técnico y de procesos a miembros principiantes, en particular sobre los requisitos de dominio y sistema, y las tecnologías que se utilizan en los proyectos. (Chau, Maurer, & Melnik, 2003)
- *Comunidades de Práctica*: Las comunidades de práctica son grupos autoorganizados que consisten en individuos que comparten información, conocimiento, experiencia y habilidades técnicas en una disciplina especializada, y colaboran en desafíos comunes o estimulan nuevas ideas. (McDermott, 1999)
- *Autoaprendizaje:* Los equipos ágiles de autogestión se involucran en el autoaprendizaje cuando un conocimiento específico parece ser importante y necesario para las actividades del proyecto. El aprendizaje puede suceder a través de la interacción y la colaboración con los miembros del equipo, el acceso y la comprensión de la información disponible en los repositorios de conocimientos o la participación en actividades de la comunidad. (Dorairaj, Noble & Malik 2012)

2.14 Transferencia de conocimiento en un equipo Multifuncional

Para construir un equipo multifuncional, es necesario que los miembros del mismo puedan recibir y compartir su conocimiento entre todos para evitar silos de especialidades y de este modo evitar los riesgos mencionados previamente. Entre las técnicas y actividades para la transferencia de conocimiento se encuentran las siguientes:

- *Daily Stand Up Meeting:* Está reunión propia del Marco de Trabajo Scrum, involucra a todos los miembros del equipo y sirve como una actividad de formación del mismo que genera un sentimiento de pertenencia, además de escuchar lo que otros miembros del mismo han hecho y descubierto en sus historias de usuario. (Rising & Janoff, 2000)
- *Programación Dual (Pair Programming)*: Promueve la integración del conocimiento colaborativo dentro de los programadores del equipo mediante la dedicación entre ambas partes a una sola historia de usuario. Proporciona un medio eficaz de difusión y retención de conocimiento en una organización (Palmieri, 2002). La idea de la programación dual es trabajar ambas partes juntas en una misma tarea, donde un miembro de la pareja es el primario, que crea activamente el código y lleva el control. El secundario revisa constantemente los datos codificados para identificar deficiencias tácticas y estratégicas, que incluyen sintaxis y lógica erróneas, faltas de ortografía e implementaciones que no se corresponden con el diseño. Después de un período de tiempo designado, los miembros invierten sus roles. El código producido por un solo miembro se descarta o se revisa de forma conjunta antes de integrarlo. (McDowell, C, Werner, Bullock, H. E, Fernald, 2006)
- *Herramientas y Repositorios:* Estas plataformas son importantes para conservar y distribuir el conocimiento recopilado por los miembros del equipo en diferentes experiencias. Además, es esencial para un proyecto contar con una herramienta donde se comparta el estado y la velocidad del mismo (Dorairaj, Noble & Malik 2012).
- *Visitas:* La transferencia de conocimiento puede ocurrir durante las reuniones diarias a través de la comunicación mediada por la tecnología, como la conferencia de audio o video. Sin embargo, se prefiere la interacción cara a cara, especialmente cuando hay un alto nivel de complejidad y ambigüedad dentro de un proyecto. (Melnik & Maurer, 2004)
- *Rotación:* La rotación en las historias de usuario asignadas promueve la distribución del negocio y el dominio del conocimiento. (Dorairaj, Noble & Malik 2012)
- *Discusión:* Las discusiones facilitan la apertura y la comunicación entre los miembros del equipo. Las discusiones con expertos en la materia sobre temas específicos

brindan oportunidades para refinar, reorganizar requerimientos y soluciones (Cohan, 2016).

- *Shadowing:* En Shadowing, un miembro secundario (Quien busca adquirir conocimiento) se encarga de observar y seguir sin intervenir en la actividad a un miembro primario (Quien tiene el conocimiento) mientras éste ejecuta. Este mecanismo es fácil de implementar, da resultados rápidos y no requieren equipo especial. Sin embargo, a menudo es difícil comprender lo que hace un ingeniero de software experimentado, ya que la idea del Shadowing es que el miembro primario trabaje de manera casual, esto propicia el uso atajos de teclado o comandos para trabajar de forma rápida. (Lethbridge, Timothy & Sim, Susan & Singer, Janice, 2005)

2.15 State of DevOps Report

'State of DevOps Report 2016' es un reporte entregado por el centro de Investigaciones Puppet y DevOps Research and Assessment (DORA), con patrocinio de empresas como HP y Attalasian. Este esfuerzo ha encuestó más de 25,000 profesionales técnicos alrededor del mundo para explorar las relaciones entre rendimiento de TI, prácticas de DevOps, cultura, desempeño organizacional y otros elementos que afectan los resultados del negocio dentro de las tecnologías de información (PUPPET-DORA, 2016).

Este reporte muestra importantes hallazgos con respecto a la implementación de DevOps para aumentar la frecuencia de entregas en un equipo de trabajo (Aumento del Scrum Velocity). Dentro de esos hallazgos, se extraen los siguientes:

- Las organizaciones con un proceso maduro de DevOps superan a las que no tienen procesos de este marco de trabajo implementado en términos de rendimiento. Las organizaciones que aplican DevOps son capaces de hacer cambios en producción con una frecuencia 200 veces superior a las que no aplican DevOps. También son 24 veces más rápidas para recuperarse de una falla en sus operaciones.

Gráfico 2.15.1 (PUPPET-DORA, 2016)

- Las organizaciones que aplican DevOps dedican un 22 por ciento menos de tiempo a actividades no planificadas trabajo y retrabajo. Como resultado, pueden usar 29% más de tiempo en nuevos trabajos, como nuevas funciones o código.

Gráfico 2.15.2 (PUPPET-DORA, 2016)

- Se descubrió que las organizaciones que aplican DevOps gastan un 50% menos de tiempo en remediar problemas relacionados a fallas de seguridad que las organizaciones que no lo utilizan.

Gráfico 2.15.3 (PUPPET-DORA, 2016)

CAPÍTULO 3: MARCO METODOLÓGICO

3.1 Enfoque de la Investigación

En el presente trabajo se utilizó un enfoque mixto de recolección de información, mezclando el método cualitativo y cuantitativo. La siguiente presenta una descripción detallada de ambos paradigmas de investigación.

Paradigma cualitativo	Paradigma cuantitativo
● Aboga por el empleo de los métodos cualitativos. ● Fenomenologismo y verstehen (comprensión) "interesado en comprender la conducta humana desde el propio marco de referencia de quien actúa". ● Observación naturalista y sin control ● Subjetivo. ● Próximo a los datos; perspectiva "desde dentro". ● Fundamentado en la realidad, orientado a los descubrimientos, exploratorio, expansionista, descriptivo e inductivo. ● Orientado al proceso. ● Válido: datos "reales", "ricos" y "profundos. ● No generalizable: estudios de casos aislados. ● Asume una realidad dinámica.	● Aboga por el empleo de los métodos cuantitativos. ● Positivismo lógico; "busca los hechos o causas de los fenómenos sociales, prestando escasa atención los estados subjetivos de los individuos". ● Medición penetrante y controlada. ● Objetivo. ● Al margen de los datos; perspectiva" desde fuera". ● No fundamentado en la realidad, orientado a la comprobación confirmatorio, reduccionista, inferencial e hipotético. ● Orientado al resultado. ● Fiable: datas "sólidos" y repetibles. ● Generalizable: estudios de casos múltiples. ● Particularista. ● Asume una realidad estable.

Tabla 3.1.1 Cook T.D & Reichardt (1986)

A continuación, se presenta los productos metodológicos para cada objetivo de la investigación.

Objetivo	Actividad	Instrumentos	Metas
Objetivo general Proponer una estrategia de implementación de la práctica DevOps para aumentar el 'Scrum Velocity' del proyecto SecurityOps, con el fin de aumentar la entrega de historias de usuario en el Scrum Team.	- Diseño de la estrategia para la implementación de técnica. - Realizar un focus group. con el Scrum Master del equipo para recolectar los datos de Scrum Velocity de pasadas iteraciones. - Llevar a cabo un piloto de la estrategia de implementación. - Plantear observaciones y mejoras a la estrategia. - Estimación del coste y beneficio del proyecto.	- Análisis de los datos. - Lista de estimación de costos. - Resultados del focus group. - Análisis cuantitativo de resultados del focus group. - Análisis cualitativo de resultados del focus group.	- Definir los procedimientos y metodología de desarrollo ajustadas según la técnica propuesta. - Definir las herramientas y componentes necesarios para la implementación del nuevo Marco de Trabajo.
Objetivo específico 1 Determinar los conceptos que la práctica DevOps propone para agilizar proceso de desarrollo y ejecución de las historias de usuario en un equipo Scrum, mediante una revisión de la literatura asociada a lo tratado en este proyecto para contar con referentes conceptuales y teóricos sólidos a la hora de crear un plan de implementación de DevOps en el problema a tratar.	- Realizar una detallada revisión de literatura sobre los temas de Metodologías Ágiles, DevOps y equipos multifuncionales.	- Revisión de literatura.	- Contar con una fuerte documentación de los temas propuestos en el trabajo para un desarrollo idóneo de las estrategias a implementar.
Objetivo específico 2 Diagnosticar el estado del Scrum Team en cuanto a la cantidad de historias de usuario que son capaces de finalizar en un Sprint mediante la revisión y	- Realizar un focus group con el Scrum Master del equipo para recolectar los datos arrojados en las iteraciones pasadas del equipo.	- Análisis de datos.	- Contar con el criterio del Scrum Team acerca de su ambiente de trabajo.

comparación de las estadísticas que arrojan los históricos de los Sprints para reconocer los cuellos de botella que generan los atrasos descritos en el problema en el proyecto SecurityOps.	- Realizar una encuesta con los miembros del Scrum Team para recolectar las virtudes y falencias del equipo de acuerdo a sus perspectivas.		
Objetivo específico 3 Proponer una estrategia para aumentar el 'Scrum Velocity' del proyecto SecurityOps por medio de la aplicación de los fundamentos de la práctica DevOps durante la ejecución de los Sprints para aumentar la entrega de historias de usuario que los clientes tienen.	- Definir una estrategia para la implementación de DevOps como complemento al Scrum Team para aumentar los puntos de Scrum Velocity.	- Diseño de estrategia.	- Entregar una guía con los procedimientos necesarios para la implementación de DevOps en el equipo. - Proponer una arquitectura de sistemas para la implementación de DevOps.
Objetivo específico 4 Probar la implementación de la práctica DevOps en el Scrum Team mediante un plan piloto para recolectar retroalimentación que permita afinar "la solución" propuesta en el proyecto SecurityOps.	- Diseño de la experiencia para la validación de la técnica. - Llevar a cabo un piloto del diseño de la experiencia. - Plantear Observaciones y mejoras al diseño de la experiencia.	- Diseño de la Experiencia. - Ejecución de la Experiencia. - Recolección de datos.	- Entregar una hoja de ruta validada sobre la experiencia y resultados del Scrum Team al trabajar bajo DevOps.

Tabla 3.1.2 (Elaboración propia)

3.2 Tipo de Investigación

Este trabajo combina los enfoques de investigación cualitativos y cuantitativos. Los datos cualitativos están basados en las experiencias del Scrum Team y sus opiniones. La perspectiva cuantitativa se basa en los datos arrojados y acumulados por parte del equipo durante los últimos doce Sprints de trabajo.

La revisión de literatura se basa en artículos de revistas académicas en su mayoría, por ejemplo, se consultaron artículos pertenecientes a IEEE y Springer. También se consultaron libros de reconocidos expertos de los temas investigados. Google Scholar fue el motor de búsqueda utilizado para conseguir dichas referencias mediante palabras claves, también se utilizó una técnica de análisis recursivo de referencias de los artículos consultados.

3.3 Variables del estudio

La siguiente tabla relaciona los objetivos específicos del trabajo, las variables contempladas y la manera en que se recolectan los datos.

Objetivo Específico	Variable	Técnica de Recolección
Objetivo específico 1 Determinar los conceptos que la práctica DevOps propone para agilizar proceso de desarrollo y ejecución de las historias de usuario en un equipo Scrum, mediante una revisión de la literatura asociada a lo tratado en este proyecto para contar con referentes conceptuales y teóricos sólidos a la hora de crear un plan de implementación de DevOps en el problema a tratar.	- Conocimiento general sobre Metodologías Ágiles. - Conocimiento general sobre DevOps.	- Revisión de Literatura. - Focus Group. - Revisión de Literatura. - Focus Group.
Objetivo específico 2 Diagnosticar el estado del Scrum Team en cuanto a la cantidad de historias de usuario que son capaces de finalizar en un Sprint mediante la revisión y comparación de las estadísticas que arrojan los históricos de los Sprints para reconocer los cuellos de botella que generan los atrasos descritos en el problema en el proyecto SecurityOps.	- Scrum Velocity. - Scrum Capacity. - Dificultades que presenta el equipo.	- Focus Group. - Observación. - Histórico del Scrum Team. - Focus Group. - Observación. - Histórico del Scrum Team. - Cuestionario.
Objetivo específico 3 Proponer una estrategia para aumentar el 'Scrum Velocity' del proyecto SecurityOps por medio de la aplicación de los fundamentos de la práctica DevOps durante la ejecución de los Sprints para aumentar la entrega de historias de usuario que los clientes tienen.		
Objetivo específico 4 Probar la implementación de la práctica DevOps en el Scrum Team mediante un plan piloto para recolectar retroalimentación que permita afinar "la solución" propuesta en el proyecto SecurityOps.		

Tabla 3.3.1 (Elaboración Propia)

3.4 Sujetos y Fuentes de Información

A continuación, se mencionan los sujetos y fuentes de estudio en las que se basa el presente trabajo. Se detallan los tipos de fuentes y métodos utilizados.

Sujetos: Los sujetos de estudio son los miembros del Scrum Team, Product Owner del proyecto y Scrum Master.

Fuentes: En este trabajo se utilizan fuentes primarias para la recolección de datos, estas son las estadísticas históricas del Scrum Team y entrevistas directas con los miembros del equipo. Entre las fuentes secundarias utilizadas, se enlistó un total de 42 referencias bibliográficas para construir una base sólida alrededor de los conceptos de Metodologías Ágiles, Scrum y DevOps.

Fuentes Primarias	Fuentes Secundarias
● Observación de las estadísticas históricas del Scrum Team, por medio del Burndown Chart y las métricas históricas del Scrum Team ● Aplicación y análisis de cuestionarios, focus group y entrevistas a los miembros del Scrum Team.	● 49 referencias bibliográficas para construir una base sólida alrededor de los conceptos de Metodologías Ágiles, Scrum y DevOps.

Tabla 3.4.1 (Elaboración Propia)

3.5 Población y Muestra

Para determinar los participantes en la recolección de datos se utilizó un tipo de muestreo dirigido, enfocándose directamente en el equipo estudiado. Este se compone de una Scrum Master, una Product Owner y cuatro miembros del Scrum Team, estos miembros tienen roles específicos: Un líder técnico, un programador, una analista de sistemas y una encargada de hacer las pruebas de calidad. Todos son profesionales en el área de Tecnologías de Información, tienen más de cuatro años laborando para la organización y más de un año trabajando bajo el Marco de Trabajo Scrum.

3.6 Instrumentos y técnicas utilizadas para la recolección de datos

La siguiente tabla describe las técnicas utilizadas, proceso y el correspondiente análisis de datos y gestión.

Técnica	Descripción	Proceso	Análisis y Gestión
Observación	Segun Batista y Hernandez (2014), la Observación es un método de recolección de datos que por medio de un registro sistemático de comportamiento permite obtener información de las actividades individuales, colectivas, artefactos y técnicas que usa un equipo. Se analizó las métricas de Scrum Capacity, Velocity y Factor de Enfoque del Scrum Team durante los últimos doce Sprints, siguiendo el proceso Levy y Ellis (2006) de Observación de los datos suministrados por el Scrum Team desde su herramienta de control (Rally) el 30 de marzo de 2019.	Se obtuvo las métricas históricas del Scrum Team durante los últimos doce Sprints. Se realizó una comparativa y relación de historias de usuario atrasadas y las opiniones mencionados por parte de los miembros del Scrum Team.	**Proceso de Lévy y Ellis (2006)** Conocer: Listar, definir, describir e identificar las métricas históricas proporcionadas por el Scrum Team. Comprender: Resumir, diferenciar, interpretar y contrastar las métricas históricas proporcionadas por el Scrum Team. Aplicar: Demostrar, ilustrar, resolver, relatar, clasificar y analizar cada métrica con los puntos sugeridos por el Marco de Trabajo Scrum. Analizar: Se mencionan las mejoras posibles al implementar las recomendaciones de DevOps a las métricas proporcionadas por el Scrum Team. Sintetizar: Se validan los cambios propuestos con el Scrum Team. Evaluar: Se entrega una propuesta para la implementación de DevOps en el proyecto SecurityOps del Scrum Team.
Autoevaluación	Según Hernández y Batista (2014), un cuestionario consiste en un conjunto de preguntas relacionadas con	Se formularon preguntas para cada una de las variables. Se realizó un cuestionario que se envió al correo	Se realizó un análisis cuantitativo de los datos, mostrando las tendencias de cada una de las variables.

	una o más variables definidas en la investigación. En el presente estudio se utilizó una encuesta conformada por 10 preguntas cerradas usando la escala Likert (Albaum, 1997).	electrónico de cada uno de los miembros del Scrum Team y que devolvieron con las respectivas respuestas. La encuesta es de elaboración propia, y tiene como objetivo determinar puntos débiles y altos de la operación del equipo de trabajo.	
Focus Group	Segun Hernandez y Batista (2014), Focus Group consiste en reuniones pequeñas en un ambiente relajado e informal. En esta dinámica, se obtienen perspectivas cualitativas del grupo.	Se reunió al Scrum Team en un conversatorio virtual donde se tocaron temas sobre su experiencia utilizando Scrum y oportunidades de mejora que perciben.	Se analizaron los resultados y se realizó una comparativa y correlación entre los resultados obtenidos en la observación y encuestas realizadas previamente.

Tabla 3.6.1 (Elaboración propia)

3.6.2 Criterios para clasificación de Historias de Usuario

Es necesario para la fase de diagnóstico contar con un instrumento que permita detectar la naturaleza de cada una de las historias de usuario que el equipo tiene en el backlog. Teniendo este instrumento, se puede observar las oportunidades de mejora que existen en temas específicos y aprovechar a la postre esa información para la elaboración de la solución. Para la clasificación de las historias de usuario, se definen los siguientes tres grupos:

- **Historias de Usuario técnica:** Se considera una Historia de Usuario técnica aquella que toque temas relacionados a scripting, bases de datos, instalación y configuración de sistemas operativos y networking.
- **Historia de Usuario no técnica:** Se considera una Historia de Usuario no técnica aquella que toque temas relacionados a configuraciones propias de la aplicación, tareas administrativas y documentación.
- **KTBR:** “Keep the Business Running”, tiquetes diarios que generan los usuarios de la aplicación. KTBR también incluye actividades que se deben ejecutar para que el ambiente de producción no falle.

CAPÍTULO 4: DIAGNÓSTICO Y ANÁLISIS DE RESULTADOS

En este capítulo se presentan los resultados obtenido luego de aplicar las técnicas y herramientas descritas en el Marco Metodológico. Se muestra un diagnóstico sobre la operación actual del Scrum Team que realiza el proyecto Security Operations. Este análisis captura la información necesaria con respecto a los puntos obtenidos en los últimos doce Sprints, la valoración del equipo con respecto a áreas puntuales del día a día del proyecto y perspectivas cualitativas de mejoras posibles para aumentar el Scrum Velocity.

Se detalla el proceso de aplicación del diagnóstico, cómo se obtuvieron los resultados comparativos correspondientes, así como las variables que se utilizaron para concluir con los principales hallazgos encontrados.

4.1 Situación previa a la aplicación de Instrumentos de Investigación

De acuerdo con comunicaciones personales entre el 13 al 19 de abril con la Product Owner, Scrum Master y el equipo de trabajo, se obtuvieron los siguientes datos:

- El equipo de trabajo tiene más de dos años de trabajar bajo el framework Scrum.
- El equipo de trabajo utiliza una aplicación (Agile Software Development) para el seguimiento de los Sprints, historias de usuario, gráficas, backlog y monitoreo del trabajo realizado y futuro por realizar.
- El equipo de trabajo no cuenta con una aplicación para “Continuous Integration”, sin embargo, continúan un proceso basado en ITIL para “Change Management”.
- Los cambios de código son realizados de manera manual a través de cada uno de los ambientes con los que el equipo cuenta. Esto significa que se deben realizar los cambios desde cero en cada ambiente al igual que las pruebas.
- El equipo de trabajo no cuenta con una aplicación para “Configuration Management”, sus configuraciones en servidores son realizadas manualmente.
- El equipo de trabajo se compone por una Product Owner, una Scrum Master, una analista de sistemas, una QA, un desarrollador y un líder técnico.
- En el departamento donde se desarrolla el equipo de trabajo existen otros equipos que actualmente se desenvuelven bajo las prácticas DevOps, sin embargo, el equipo en cuestión no tiene visualizado su implementación futuramente.
- El equipo de trabajo también soporta tareas de operaciones de una segunda aplicación que soportan fuera de Security Operations, por lo tanto, dentro de sus historias de usuario, se incluyen las relacionadas a la segunda herramienta.
- Con respecto al punto previo, el equipo está enfocado en el proyecto de Security Operations, por lo tanto, se le asigna un 10% del Scrum Capacity a las tareas relacionadas a una segunda herramienta que es administrada por el grupo.

- El equipo cuenta con una solución de repositorios basada en Git, sin embargo, solo el desarrollador y el líder técnico la utilizan, y no tienen definido qué tipo de cambios almacenar ahí. Lo utilizan como respaldo de los códigos que desarrollan y consideran a criterio personal importantes.
- Entre los planes del equipo para aumentar el rendimiento del mismo, con el sistema actual que utilizan, se contempla la contratación de un nuevo miembro de perfil técnico para el proyecto.
- En cuanto a las arquitecturas de los sistemas en las que se almacena el actual proyecto de SecurityOps, el equipo cuenta con un ambiente de producción, con un servidor con un espacio de disco de 5TB, 16GB de memoria y un sistema operativo UNIX, este sistema en producción cuenta con un servidor para Disaster Recovery, con las mismas características del servidor primario. También cuenta con un ambiente de Test, con un espacio de 500GB, 8GB de memoria, con un sistema operativo UNIX, además de un sistema de Development. El equipo cuenta con un sistema de repositorios en GitLab y un sistema de monitoreo para cada uno de sus servidores, el cual se encuentra aislado del sistema de administración de operaciones, donde los clientes suben tiquetes cuando existen problemas en los diferentes ambientes de la aplicación.
- El ciclo de vida de una historia de usuario dentro del equipo está estructurado de la siguiente manera:

    1. El PO captura el requisito del cliente y redacta la historia de usuario.
    2. La historia de usuario ingresa al Backlog del equipo.
    3. La historia de usuario es definida en la sesión de Grooming para participar dentro de un Sprint. En esa misma sesión, se le da una puntuación equivalente al esfuerzo que conlleva la historia de usuario.
    4. En la sesión de Planning, se le es asignada a un miembro del equipo. El estado de la historia pasa a ser “Iniciada”.
    5. La historia de usuario es iniciada por el miembro designado, su estado pasa a ser “En desarrollo”. La historia se trabaja en el ambiente de Development.
    6. Cuando el designado considera lista su solución, debe recrear el cambio en el ambiente de Test.
    7. Una vez concluida la historia, tiene que pasar por el proceso de análisis de calidad que realiza el analista del equipo.
    8. Si la historia es aceptada, se inicia el proceso para llevarla al ambiente de producción. Se debe abrir una solicitud de cambio con el analista de sistemas

del equipo. Está solicitud incluye la descripción del cambio y el plan de contingencia si el mismo llegara a impactar el ambiente de producción.

9. Una vez que la solicitud es aprobada, el designado realiza el cambio en el ambiente de Producción.
10. Cuando el cambio está en Producción, la historia de usuario se cierra y se considera concluida por parte del PO del equipo.

4.2 Instrumentos de Recolección de Datos: Observación de datos de los Sprints

Se analizaron los últimos diez Sprints del Scrum Team, durante el periodo comprendido entre noviembre de 2018 y abril de 2019, información extraída entre el 13 y 19 de abril junto al Scrum Master del equipo en cuestión y basados en la herramienta para el seguimiento de los Sprints. La duración de los Sprints del equipo es de dos semanas. Se muestra la cantidad de puntos aceptados durante la iteración, después de la iteración y los que no fueron ejecutados. Así también se exhiben los valores promediados de las iteraciones, y se puede deducir los Sprints que se encuentran por debajo de esa línea. También se muestra el Scrum Velocity de cada Sprint individualmente.

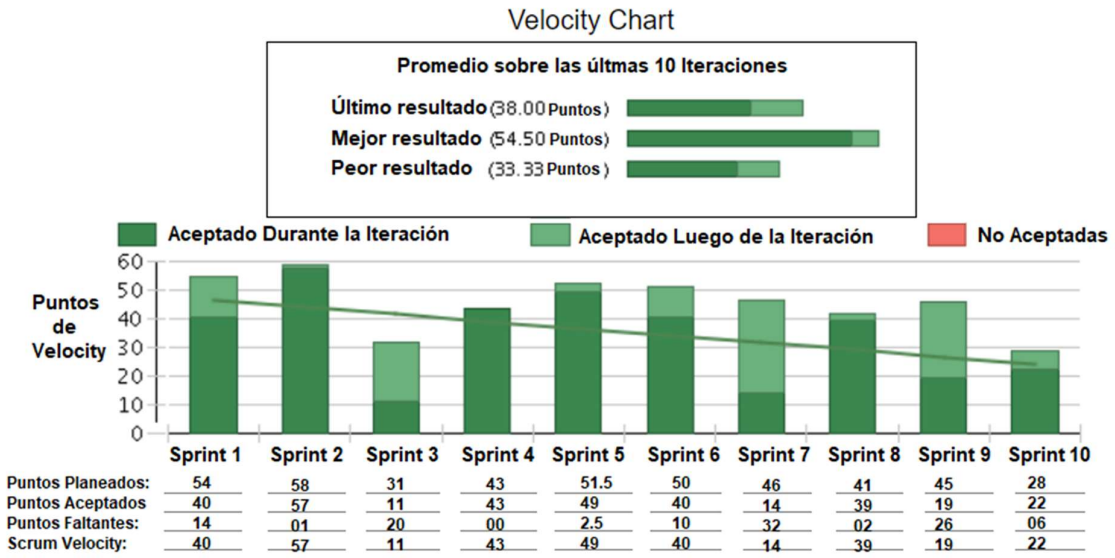

Figura 4.2.1 (Elaboración propia)

La tabla refleja la constancia del equipo con respecto al Scrum Velocity, hay cinco Sprints de los diez evaluados que se encuentran por debajo de la línea promedio. Se procede a evaluar individualmente esos cuatro Sprints, para demostrar el detalle de las historias de usuario que no fueron aceptadas durante el Sprint. Se usan tres criterios para clasificar las Historias de Usuario: Técnica, No Técnica, KTBR (Operación diaria). Estos criterios utilizados fueron previamente detallados en el Marco Metodológico.

En las siguientes tablas se especifica el Sprint correspondiente y los puntos que no fueron aceptados durante la iteración, clasificados de la manera previamente mencionada.

	Técnicas	No Técnicas	KTBR
Sprint 1	**10**	**02**	**02**
Sprint 3	**18**	**02**	**00**
Sprint 7	**25**	**05**	**02**
Sprint 9	**18**	**06**	**02**
Sprint 10	**06**	**00**	**00**

Tabla 4.2.1 (Elaboración Propia)

La pasada tabla permite desarrollar el siguiente gráfico, donde observa claramente que un 79% de las historias de usuario presentes en los Sprints que no alcanzaron el promedio de entregas del equipo eran historias clasificadas como técnicas.

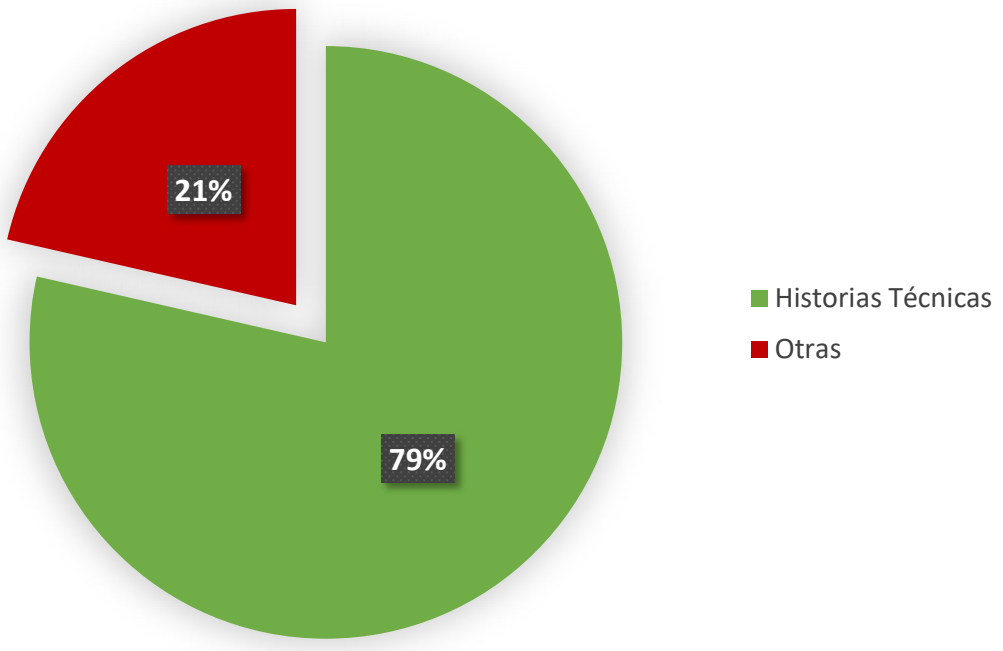

Gráfico 4.2.1 (Elaboración Propia)

4.3 Instrumentos de Recolección de Datos: Autoevaluación

Como parte del soporte para la investigación, se realizó entre el 13 al 19 de abril una evaluación de diferentes aspectos del equipo entorno a la implementación de Agile en el mismo, para determinar puntos altos y bajos de la operación diaria bajo las prácticas ágiles.

Las preguntas, puntuaciones y formato de la evaluación están inspirado en la propuesta de SAFe Team Self-Assessment, en su versión V4.0.4. Se escogió esta propuesta ya que es utilizada en varios departamentos de la empresa.

Se evaluaron cuatro áreas, que incluyen el desempeño del Product Owner y la elaboración de las historias de usuario, el desempeño propio del equipo y del Scrum Master, las iteraciones y la perspectiva técnica del equipo.

Se considera muy mala las evaluaciones cuyo promedio se encuentre entre 1 y 2 puntos, mala las que se encuentren entre 2 y 3 puntos, regular las que se encuentren entre 3 y 4 puntos y positivas las que se encuentren entre 4 y 5 puntos. Aquellas evaluaciones que se encuentren entre los rangos malos y muy malos son las que se toman como base para la realización de la propuesta de solución.

Se aplicó la siguiente encuesta a tres miembros del Scrum Team (Líder Técnico, desarrollador y QA. La analista de sistemas no participó en el equipo). La puntuación que se muestra en la tabla es el resultado de aplicar el promedio simple en las seis encuestas aplicadas. Se describe también la importancia de los datos arrojados por la encuesta.

Área / Pregunta	**Puntuación**
Evaluación del Product Owner	
El Product Owner facilita el desarrollo de historias de usuarios, priorización y negociación.	4.3
El Product Owner colabora de forma proactiva con Product Management y otras partes interesadas.	4.3
Las historias de usuarios son pequeñas, estimadas, funcionales, verticales y se ajustan a una iteración.	3.3
El Product Owner facilita el desarrollo de criterios de aceptación que se utilizan en la planificación, revisión y aceptación de la historia.	3.6
El equipo refina el Team Backlog en cada iteración.	3.6
Total Puntuación de la Evaluación del Product Owner	**3.82**

Tabla 4.3.1 (Elaboración Propia)

Según la puntuación total de esta categoría, la labor del Product Owner es aceptada y se puede definir como regular. Sin embargo, el margen muestra oportunidades de mejora, siendo la estimación, función y definición de las historias el trabajo realizado sobre la formulación de las historias de usuario.

Área / Pregunta	Puntuación
Evaluación las Iteraciones	
El equipo planea la Iteración de manera colaborativa, efectiva y eficiente.	4.6
El equipo siempre tiene objetivos de iteración claros, en apoyo de los objetivos de la empresa, y se compromete a cumplirlos.	4.3
El equipo aplica los criterios de aceptación y aplica Definition of Done a la aceptación de la historia.	4.3
El equipo tiene una velocidad predecible y normalizada que se utiliza para estimar y planificar	4.3
El equipo cumple regularmente sus objetivos de iteración.	4.3
Total Puntuación de la Evaluación las Iteraciones	**4.36**

Tabla 4.3.2 (Elaboración Propia)

Según la puntuación total de esta categoría, la evaluación de las iteraciones por parte de los encuestados es positiva.

Área / Pregunta	Puntuación
Evaluación del Equipo	
Los miembros del equipo son autoorganizados, se respetan mutuamente, se ayudan mutuamente a cumplir los Objetivos de iteración, gestionan las interdependencias y se mantienen sincronizados entre sí.	4.6
El Scrum Master asiste a Scrum of Scrums e interactúa con RTE según corresponda.	4.6
Las historias se completan a lo largo de la Iteración con múltiples ciclos de definir-construir-prueba (es decir, la Iteración no es catastrófica).	4.6
El equipo se reúne todos los días a la misma hora y lugar para el Stand-up diario para coordinar su trabajo y resolver los impedimentos.	5.0
El equipo realiza una retrospectiva después de cada Iteración y realiza cambios incrementales para mejorar continuamente su rendimiento.	5.0
Total Puntuación de la Evaluación del Equipo	**4.70**

Tabla 4.3.3 (Elaboración Propia)

Según la puntuación total de esta categoría, la evaluación de las iteraciones por parte de los encuestados es positiva.

Área / Pregunta	Puntuación
Evaluación del Técnica del Equipo	
El equipo reduce activamente la deuda técnica en cada iteración.	1.6
El equipo tiene una guía y comprensión claras de la arquitectura de los proyectos, pero es lo suficientemente libre y flexible para permitir que el diseño emergente respalde la implementación óptima	2.0
Se realizan pruebas automatizadas de aceptación y pruebas unitarias son parte de la historia y su Definition of Done	2.0
La refactorización es constante en las iteraciones.	2.6
Constante mejora en infraestructura de automatización de construcción y prueba	2.6
El equipo genera nuevas hipótesis y las prueba continuamente.	2.3
El equipo está desplegando continuamente a producción.	2.6
El equipo diseña su trabajo para permitir el despliegue continuo, el lanzamiento bajo demanda y la recuperación	2.3
Total Puntuación de la Evaluación Técnica del Equipo	**2.25**

Tabla 4.3.4 (Elaboración Propia)

Según la puntuación total de esta categoría, la evaluación de la deuda técnica del equipo por parte de los encuestados tiene una nota negativa. El reducir la deuda técnica en cada iteración es el área evaluada peor puntuada en toda la encuesta.

El siguiente gráfico permite visualizar mejor los hallazgos una vez aplicado el instrumento de la autoevaluación.

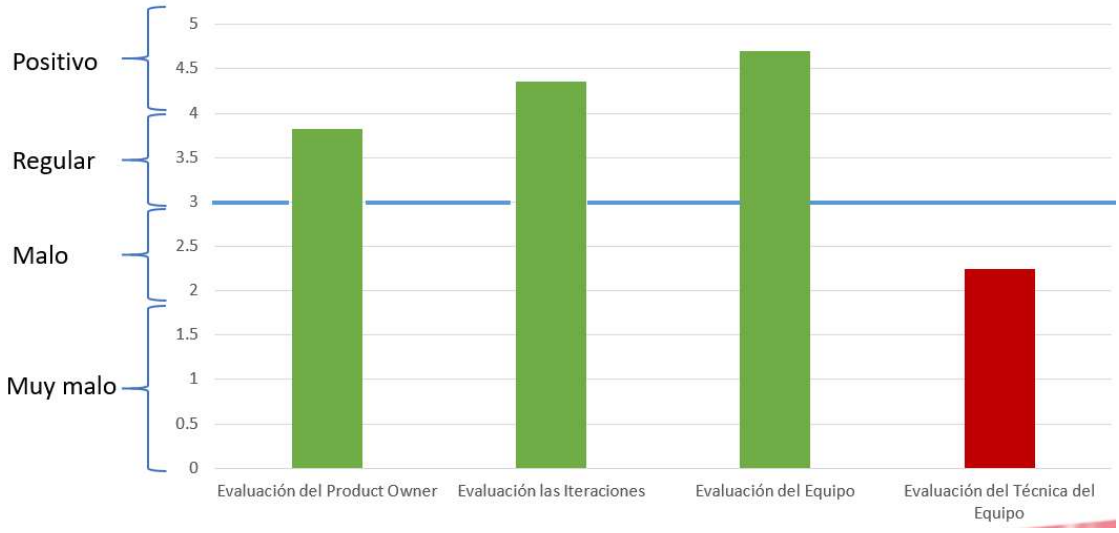

Gráfico 4.3.1 (Elaboración Propia)

4.5 Instrumentos de Recolección de Datos: Focus Group

Se realizó el instrumento de Focus Group el 15 de abril junto al Product Owner, Scrum Master y el equipo de trabajo. Está sesión se llevó a cabo en las instalaciones de Intel en Folsom California, donde se obtuvieron las siguientes conclusiones al consultarles los obstáculos principales que no permiten al equipo ir más rápido a la hora de entregar las historias de usuario:

- Existencia de una deuda técnica en el equipo.
- Las historias de usuarios no están dimensionadas correctamente. El peso que se les da a las historias suele ser diferente una vez que se trabaja en ellas, lo que conlleva a retrasos a la hora de la entrega presupuestada.
- Falta de herramientas para la aplicación de “Continuous Integration” y “Continuous Delivery”. Los cambios hacia producción se realizan paso a paso por el encargado de llevar la historia de usuario al ambiente de producción.
- Falta de herramientas para la Gestión de Configuración. Al no contar con este tipo de herramientas, procesos técnicos que se pueden reducir mediante la automatización se deben realizar manualmente.
- Los cuatro miembros del equipo de trabajo realizan roles distintos, lo que no permite tener un buen respaldo cuando alguno de los miembros se ausenta.
- La analista de calidad menciona sobrecarga en su rol. Indica que existen ciertas historias de usuario que podrían pasar facilitar su inspección si los implementadores hicieran pruebas antes de enviarlas al análisis.
- El equipo depende de servicios prestados por otros equipos, por ejemplo, los relacionados a la configuración de reglas de Firewall, Hosting o equipos de monitoreo. Además, los servicios profesionales que provee la plataforma de Security Operations no responde de manera pronta los tiquetes y dudas puestos por el Scrum Team.
- Las historias de usuario relacionadas a las operaciones diarias (KTBR) aunque se le den un 10% de la capacidad del Sprint, no se pueden definir de buena manera, y la resolución de las mismas puede tardar más de lo esperado. Cuando es un tiquete de operación técnico, uno de los dos miembros técnicos debe enfocarse en la solución de la misma, y dejar de lado las otras historias de usuario asignadas relacionadas a Security Operations. Sin embargo, este tipo de actividades son realizadas sin problema por cualquiera de los miembros del equipo.
- Los ambientes de Desarrollo y Test de la aplicación están contaminados y desfasados, esto significa que Desarrollo no refleja los últimos cambios que se encuentran en producción, ya que, en muchas ocasiones, se realizan los cambios y pruebas directamente en el ambiente de Test.

CAPÍTULO 5: SOLUCIÓN DEL PROBLEMA

En este capítulo se presenta el desarrollo de los procedimientos que conforman la propuesta de solución para la adaptación de DevOps dentro del proyecto SecurityOps. Se describe el procedimiento para cada uno de los pasos y se finaliza con las pruebas y los resultados de la aplicación de la experiencia controlada, en el contexto de un Sprint del Scrum Team. La propuesta busca cumplir los objetivos planteados en el Capítulo I tomando en cuenta los hallazgos encontrados en la fase de diagnóstico para contar con una propuesta que se ajuste a las necesidades y condiciones del equipo.

5.1 Desarrollo de la solución

Se plantea utilizar las buenas prácticas propuestas por DevOps para aumentar el Scrum Velocity del equipo, estas prácticas previamente explicadas en este documento promueven eficiencia y eficacia en los hallazgos del diagnóstico realizado. El procedimiento involucra nueva infraestructura, entrenamiento en áreas del equipo y procesos.

En un panorama amplio de la solución propuesta, se deben construir tres pilares en el equipo, basados en las ideas conceptuales que presenta DevOps y en la situación actual del equipo de trabajo. El primer pilar busca crear una responsabilidad compartida en el mismo (Evitando trabajar en silos separados, como menciona Ebert, Gallardo, Hernandez & Serrano (2016) y atacando la dependencia técnica que se encontró el diagnóstico realizado al Scrum Team. El segundo pilar, basado en los principios dictados por Virmani (2015), promueve desarrollar procesos para escalar la autonomía del equipo ya que, de acuerdo al Focus Group realizado en la fase de diagnóstico, no se cuenta con una sólida base en proceso de Integración y Entrega Continua, y por último, un tercer pilar formulado para crear una herramienta de retroalimentación que permita ver el avance de las prácticas en el Scrum Team, enfatizando la importancia de la retroalimentación para DevOps. (Virmani, 2015)

Estos tres pilares tienen por objetivo justificar debilidades halladas en las evaluaciones realizadas. Con respecto a crear un sentimiento de responsabilidad compartida, se encontró que el equipo de desarrollo trabaja en silos, lo que lleva a un desinterés en el mantenimiento operacional del sistema. Se debe fomentar el trabajo conexo entre operaciones y desarrollo en cada una de las iteraciones, donde los miembros del equipo sean capaces y estén anuentes a tomar historias de usuario de los dos tipos, tanto operación como desarrollo. Según la teoría que describen las metodologías ágiles, las cuales busca optimizar DevOps, se suele preferir los equipos generalistas (Parsons, Lal, Ryu, Lange, 2017) donde no haya una división entre operaciones y desarrollo, puesto que brindan una mejor respuesta al cambio y reduce el tiempo de entrega. (Parsons, Lal, Ryu, Lange, 2017)

La autonomía del equipo busca que los desarrolladores, así como operaciones, puedan tomar decisiones sin seguir un proceso complicado y extremadamente burocrático. Los equipos pueden trabajar en un entorno favorable donde no hay discrepancias en el alcance. En lugar de esperar una firma manual para que se implemente el código en las pruebas, puede confiar en un sistema de control de versiones que puede auditarse fácilmente. Sin la necesidad de una aprobación manual, el equipo puede automatizar las implementaciones y acelerar el ciclo de pruebas. En el diagnóstico realizado, se identificó un proceso de Integración Continua y de Entrega Continua, el cual no está debidamente depurado, al faltar la determinación en ciertos ambientes de trabajo. Virmani (2015) define la importancia de contar con herramientas que permitan la correcta ejecución de los procesos de Integración y Entrega continua, además de acelerar procesos mediante automatización.

Por último, un proceso de retroalimentación es importante para mejorar continuamente la colaboración entre el desarrollo y las operaciones. La supervisión de cómo se comporta una aplicación o un sistema en producción crea un entorno de propiedad y captura las áreas de mejora potencial, además de hacer visible al grupo los objetivos cumplidos.

5.2 Desarrollo descriptivo de la Propuesta de Solución

Como se menciona en los párrafos anteriores, la propuesta de solución se basa en tres pilares: Responsabilidad Compartida, Autonomía y Retroalimentación. A continuación, se desgranan esos pilares en cada una de las actividades necesarias para llevarlos a cabo.

5.2.1 Responsabilidad Compartida

A continuación, se desarrollan cada una de las actividades programadas para cumplir el pilar de Responsabilidad Compartida, se especifica el nombre, justificación de acuerdo a los hallazgos del Diagnóstico, se compara contra la teoría recolectada en el marco teórico y la implementación propuesta de la actividad.

5.2.1.1 Entrenamientos en DevOps

Es necesario introducir a todos los miembros del equipo a la metodología DevOps y sus fundamentos. Para ello, se deben agendar los cursos de capacitación de DevOps para preparar al equipo a una carrera en DevOps.

Se busca comprender en teoría, para poder llevarlo a la práctica, los principios de desarrollo e implementación continuos, automatización de la gestión de la configuración, colaboración entre equipos y agilidad de los servicios de TI.

Para la implementación de esta actividad se debe seguir lo siguiente:

- Programar como historias de usuario las sesiones de entrenamiento asignadas.
- Proponer opciones de entrenamiento para aprender las prácticas DevOps.
- Participantes: Scrum Team, Scrum Master, Product Owner, Mánager.

5.2.1.2 Definición del rol deseado para los miembros del Scrum Team

De acuerdo con los datos encontrados en el diagnóstico realizado al Scrum Team, la deuda técnica es una de las razones que han impedido al equipo completar de manera más expedita las historias de usuario que tienen, esto debido a los roles asignados que se tiene en cada miembro del equipo. Se cuenta con personas dedicadas a desarrollo, análisis de calidad y tareas de administración de sistemas. Dependiendo del Sprint y las historias que se están realizando, se crea un estancamiento en las mismas, pues se sobrecargan alguno de los miembros del equipo mientras otros pueden tener inclusive una menor cantidad de puntos de las que pueden entregar realmente. Es necesario esta definición puesto que los miembros del equipo, y futuros miembros, tendrán un nuevo rol, con un mayor alcance y responsabilidades. Es necesario contar con ese rol definido por cuestiones netamente de administración y documentación por parte del mánager del equipo para el desarrollo de carrera, rangos salariales y reportes al departamento de recursos humanos.

La unidad organizativa conceptual inherente a un paradigma DevOps se extiende naturalmente a la interoperabilidad de las herramientas de desarrollo de software y operaciones, para garantizar el acceso máximo a los datos, la difusión de conocimientos y la automatización. (C. A. Cois, J. Yankel and A. Connell, 2014). Por eso, se precisa definir el perfil ideal que debe cumplir un miembro del Scrum Team para ser apto de tomar cualquier historia de usuario. Para la implementación de esta actividad se debe seguir lo siguiente:

- Reunión junto al Manager del equipo, Scrum Master y Product Owner del mismo para definir el perfil necesario acorde al negocio.
- Formular un perfil de acuerdo a las áreas técnicas diagnosticadas previamente en este trabajo, entre las que destacan Java Scripting, desarrollo de API’s, soporte de Infraestructura en RedHat, soporte de infraestructura en Intel® Cloud/Prem Hosting y Load Balancing. Además de manejo de protocolos de seguridad y sistemas de monitoreo. Por último, experiencia en ambientes de operaciones y troubleshooting.
- La definición de este perfil se debe tener presente para la transformación de los actuales roles y de futuras contrataciones.
- Participantes: Scrum Master, Product Owner, Manager.

5.2.1.3 Sesiones de Shadowing

Las historias de usuario relacionadas con programación y soporte de infraestructura son tomadas por los miembros más técnicos del Scrum Team. Existen iteraciones donde se sobrecarga de historias de este tipo, por ende, se sobrecarga a un miembro del equipo y se atrasan historias. Según el diagnóstico realizado, los miembros del equipo se sienten cómodos trabajando historias relacionadas a operaciones, por lo tanto, se descartan las sesiones de Shadowing para historias de esta índole.

Shadowing proporciona un método rápido y sin necesidad de adquirir ningún entrenamiento, programa o licenciamiento asociado para la transferencia de conocimiento (Lethbridge, Timothy & Sim, Susan & Singer, Janice, 2005). Para la implementación de esta actividad, se debe seguir lo siguiente:

- Para las historias de usuario relacionadas a Java Scripting, desarrollo de APIs, Phyton o Bash en procesos de automatización, se asignará un titular para la historia, y también un secundario para la misma.
- Se escoge en los Sprint Plannings el titular de la historia como el secundario convendrán un horario para realizar la tarea asignada.
- Este tipo de tarea significa un esfuerzo extra para el titular, sin embargo, la idea de la misma es realizar de tres a historias de usuario bajo esta técnica (Según referencia de Shadowing).
- Pasadas las sesiones de Shadowing, se prosigue en futuras iteraciones con programación dual, suponiendo que el secundario ha mejorado sus habilidades de programación.
- Participantes: Scrum Team, Scrum Master.
- Beneficios: Reducción de la deuda técnica, más historias de usuario

5.2.1.4 Programación Dual (Pair Programming)

La idea de aplicar Pair Programming en el equipo es reafirmar la responsabilidad compartida del mismo, convertir el desarrollo de software, una de las áreas donde existe mayor deuda técnica, en un esfuerzo grupal. En lugar de que una persona escriba el código, dos personas colaboran en tiempo real. Se sugiere utilizar Programación dual en las historias de usuario relacionadas a Java Scripting, Python, APIs y en las de Soporte de Infraestructura donde esta modalidad amerite ser utilizada. Se ha determinado en experimentos que los equipos que aplican Programación Dual tienen un desempeño alto en funcionalidad y legibilidad del programa, brindan una mayor satisfacción con el proceso de resolución de problemas, tienen mayor confianza en sus soluciones y completan mayor cantidad de tareas de programación

que un programador independiente (McDowell, C, Werner, Bullock, H. E, Fernald, 2006). Para la implementación de esta actividad se debe seguir lo siguiente:

- Pasados los Sprints donde un miembro haya tenido las suficientes sesiones de Shadowing, se asignan historias bajo la modalidad de Programación Dual.
- Las historias de usuario deberán ser divididas en tareas, de ser posible, en las ceremonias de Grooming.
- En los Sprint Plannings, se escoge para cada historia un desarrollador principal, que funcionará como controlador de la historia y lleva la mayor parte del desarrollo de código. También se asigna un secundario, que se encargará de otras tareas de la historia.
- Ambos desarrolladores deben calibrar sus avances para la entrega efectiva de la historia.
- Participantes: Scrum Team, Scrum Master.

5.2.1.5 Rotación de roles: Operaciones y Desarrollo

La rotación de roles ayuda al Scrum Team descubrir y solucionar proactivamente los problemas del sistema antes de que afecten al cliente. Se busca evitar la separación de los miembros del equipo entre historias de desarrollo y las de operaciones. Según el diagnóstico realizado, los miembros del equipo evitan ciertas historias en las que no se sienten cómodos. La rotación de roles promueve que todos aprendan a trabajar cualquier historia del backlog. Con menos tickets de soporte para resolver, los miembros del equipo tienen más tiempo para entregar valor. Como resultado, se brindan mejores servicios con mayor velocidad y eficiencia.

La rotación en las historias de usuario asignadas promueve la distribución del negocio y el dominio del conocimiento (Dorairaj, Noble & Malik 2012). Para la implementación de esta actividad se debe seguir lo siguiente:

- Cuando se demuestra una madurez en el equipo de trabajo, donde cada miembro se acerque al perfil elaborado previamente (Para alcanzar este perfil, se habrán realizado las suficientes sesiones de Shadowing y Programación Dual).
- En el Sprint Planning, se asignan las historias de usuario de manera voluntaria, sugiriendo que exista una rotación de los dueños de las historias de desarrollo y operaciones con respecto al Sprint terminado.
- Participantes: Scrum Team, Scrum Master, Product Owner.

5.2.1.6 Entrenamientos técnicos

Las capacitaciones formales permiten estandarizar el contenido y las prácticas de múltiples equipos en una organización. Las sesiones de entrenamiento son beneficiosas para difundir el conocimiento técnico (Chau, Maurer, & Melnik, 2003). Según el diagnóstico realizado, las áreas donde se debe buscar entrenamientos son las siguientes: Java Scripting, desarrollo de API's, soporte de Infraestructura en RedHat, soporte de infraestructura en Intel® Cloud/Prem Hosting, Load Balancing y fundamentos básicos para análisis y ejecución de pruebas de calidad. Para la implementación de esta actividad se debe seguir lo siguiente:

- Asignar entrenamientos a los miembros no técnicos del equipo, ya que se determinó en el diagnóstico que esta es el área donde se necesita realizar una mejoría en el equipo.
- Asignar entrenamientos a los miembros del equipo, excluyendo a la analista de calidad, sobre fundamentos básicos para el análisis y la ejecución de pruebas de calidad.
- Programar como historias de usuario las sesiones de entrenamiento asignadas.
- Las sesiones de entrenamiento deben ser relativas a Java Scripting, desarrollo de API's, soporte de Infraestructura en RedHat, soporte de infraestructura en Intel® Cloud/Prem Hosting y Load Balancing.
- Se debe incluir una sesión de entrenamientos sobre fundamentos de análisis de calidad, para que el trabajo de la analista de calidad sea más expedito.
- PO, Scrum Master y el Scrum Team propondrán las opciones de entrenamientos relativos a los temas previamente mencionados y se seleccionarán.
- Participantes: Scrum Team.

5.2.1.7 Entrenamientos: Resistencia al cambio

Una vez ejecutado el plan piloto, se detectan comentarios por parte de los participantes que demarcan la existencia de una resistencia al cambio, en este caso, asumir nuevas responsabilidades en el equipo. Por esta razón, se propone entrenar a la Product Owner y Scrum Master para lidiar con este tema. Para la implementación de esta actividad se debe seguir lo siguiente:

- Programar como historias de usuario las sesiones de entrenamiento asignadas.
- Participantes: Scrum Master y Product Owner

5.2.2 Autonomía

A continuación, se desarrolla cada una de las actividades programadas para cumplir el pilar de Autonomía, se especifica el nombre, justificación de acuerdo con los hallazgos del Diagnóstico, se compara contra la teoría recolectada en el marco teórico y la implementación propuesta de la actividad.

5.2.2.1 Definición de una herramienta de CI/CD

El proceso actualmente no está siendo acompañado por una herramienta especializada de CI/CD. Existe una dinámica establecida por el equipo de trabajo para llevar los cambios a producción, la cual se podría potenciar con el uso de una herramienta de esta índole.

La idea de una aplicación para llevar el proceso de CI/CD es fusionar el código individual producido por los desarrolladores diariamente la mayor cantidad de veces posible y hacer los controles de calidad y pruebas continuamente para evitar problemas posteriores.

El CD lleva esto un paso más allá para garantizar que todo el código combinado esté siempre en un estado listo para la producción. Estas aplicaciones no son las encargadas de automatizar un proceso en sí (De eso se encarga Ansible, Chef, Puppet…), sino de orquestar la ejecución de estos procesos para llevar un cambio a producción, funcionando como un semáforo para aplicar los cambios entre ambientes para concluir en producción. (Virmani, 2015)

Herramientas de CI/CD como Jenkins o TeamCity son capaces de realizar cientos de tareas necesarias para asegurar la calidad del software y facilitar su despliegue o construcción. Entre las tareas que pueden realizar destacan:

- Análisis de la calidad y pruebas de código.
- Supervisar criterios de calidad de los cambios del código en la aplicación establecidas por el equipo de trabajo.
- Ejecutar las modificaciones de la aplicación, una vez pasadas todas las validaciones, a un repositorio principal.
- Automatizar la compilación del código y su despliegue, una vez se hayan integrado nuevos cambios en el proyecto.
- Notificar al equipo de trabajo el aseguramiento de la calidad cuando se encuentra falencias y errores, ya sea en base a las pruebas del programa o a las métricas de calidad definidas.
- Generar o visualizar la documentación del proyecto

Para la implementación de esta actividad se debe seguir lo siguiente:

- Reunión entre PO, Scrum Master, Manager y Scrum Team para la selección de un programa para la administración de CI/CD (Jenkins, Team City...).
- Configuración del servidor, contenedor o máquina virtual donde se almacenará la aplicación.
- Instalación de la aplicación.
- Configurar los proyectos y definir los estados de los mismos dentro de la aplicación.
- Scrum Master debe asignar esta aplicación como una historia de usuario.
- Participantes: Scrum Team, Scrum Master, Product Owner.

5.2.2.2 Estrategia de Ramificación de repositorios

Aunque el Scrum Team cuenta con un repositorio para subir código, no existe un estándar para su utilización, esto significa que muchos de los cambios que actualmente llegan a producción no pasan por una rama del repositorio. Además, se necesita alinear la estrategia de ramificación junto a la aplicación de CI/CD.

Los sistemas de control de versiones distribuidos brindan a los miembros de un equipo de trabajo una gran flexibilidad en la forma para manejar el proceso de control de versiones y administración del código fuente. Estas herramientas permiten al equipo pasar menos tiempo administrando el versionamiento y más tiempo desarrollando. La automatización de procesos es una clave del marco de trabajo DevOps. Para la implementación de esta actividad se debe seguir lo siguiente:

- Definición del repositorio a utilizar (GitLab vs. GitHub).
- Definición del código que se espera subir y respaldar en los repositorios por parte del equipo técnico.
- Instalación de los plugins o breaches necesarios para la conexión entre la herramienta de CI/CD y el repositorio.
- Scrum Master debe asignar esta configuración como una historia de usuario.
- Participantes: Scrum Team, Scrum Master, Product Owner.

5.2.2.3 Definición de Ambientes de desarrollo

Actualmente, el Scrum Team cuenta con tres ambientes de trabajo: Desarrollo, Test y Producción, incluyendo su ambiente replicado para Disaster Recovery. A esta estrategia se le debe limitar las funciones que se pueden y deben realizar en cada una de ellas para evitar

la contaminación a la cual se refieren los miembros del Scrum Team en la sesión de Focus Group realizada. Además, definir las pruebas que se deben hacer en el ambiente de Test antes de pasar la historia de usuario a la analista de sistemas. Para la implementación de esta actividad se debe seguir lo siguiente:

- Integración de los ambientes existentes a la aplicación utilizada para CI/CD.
- Definición y limitación de las actividades que se van a realizar en los diferentes ambientes, para evitar la contaminación y control de versiones innecesario entre instancias.
- Definición de las pruebas a realizar en el ambiente de Test, antes de iniciar el proceso formal de análisis de calidad.
- Scrum Master debe asignar esta configuración como una historia de usuario.
- Participantes: Scrum Team, Scrum Master, Product Owner.

5.2.2.4 Automatización de Procesos

Muchos de los procesos del equipo pueden ser automatizados mediante herramientas como Ansible. Estos procesos automáticos reducen tiempo en tareas de operaciones diarias. Es importante reconocer estas tareas y aplicar los templates necesarios para aumentar la eficiencia del grupo. Algunos procesos que se pueden automatizar son reinicio de servidores, instalación de aplicaciones en los servidores, apagado de los servidores y notificaciones. Para la implementación de esta actividad se debe seguir lo siguiente:

- Reunión entre PO, Scrum Master, Manager y Scrum Team para la selección de un programa para la automatización de procesos (Ansible, Puppet...).
- Configuración del servidor, contenedor o máquina virtual donde se almacenará la aplicación.
- Instalación de la aplicación.
- Reunión con el Scrum Team y Scrum Master para la detección de procesos que pueden ser automatizados.
- Creación del scripting necesario para la automatización de procesos dentro de la aplicación.
- Scrum Master debe asignar esta configuración como una historia de usuario al igual que los procesos que pueden ser automatizados y su debido desarrollo.
- Participantes: Scrum Team, Scrum Master, Product Owner.

5.2.2.5 Monitoreo de Infraestructura

Prevenir futuros problemas en los sistemas tanto de producción como de test se logra mediante la implementación correcta de los sistemas de monitoreo. Aunque el equipo de trabajo cuenta con un sistema de monitoreo, ese encuentra aislado del resto de sistemas, por lo tanto, varias alertas no son tomadas en cuenta hasta que se abre un tiquete por parte de los clientes mencionando el problema, cuando se pudo haber automatizado ciertos procesos entre los sistemas para evitar estos casos. Para la implementación de esta actividad, se debe seguir lo siguiente:

- Instalación de la herramienta de monitoreo en todos los ambientes de la aplicación.
- Integración de la herramienta de automatización con la de monitoreo utilizada.
- Scrum Master debe asignar esta configuración como una historia de usuario al igual que los procesos que pueden ser automatizados y su debido desarrollo.
- Reunión con el Scrum Team y Scrum Master para la creación de dashboards y debidas alertas para reducir los tiquetes operativos.
- Participantes: Scrum Team, Scrum Master, Product Owner.

5.2.3 Retrospectiva

A continuación, se desarrolla cada una de las actividades programadas para cumplir el pilar de Retrospectiva, se especifica el nombre, justificación de acuerdo a los hallazgos del Diagnóstico, se compara contra la teoría recolectada en el marco teórico y la implementación propuesta de la actividad.

5.2.3.1 Depurar los indicadores de avance para el Scrum Master

Se debe medir el avance de la implementación de DevOps en el equipo para conocer el progreso del mismo. Con este marco de trabajo, se presume mejorar varios de los aspectos a mejorar hallados en la sección de diagnóstico. Virmani (2015) menciona en la aplicación de DevOps la importancia de la Integración Continua, Entrega Continua y la retroalimentación del equipo. Es necesario por parte del Scrum Master medir estos aspectos y transmitirlos al equipo de manera cuantitativa. Para la implementación de esta actividad, se debe seguir lo siguiente:

- Contar con un formulario para comparar cuantitativamente y cualitativamente Sprints. Los aspectos a comparar son:
    - Entregas en el ambiente de Producción.
    - Automatizaciones realizadas.
    - Reducción de la Deuda Técnica (Cuantas historias consideradas técnicas

fueron tomadas por diferentes miembros del equipo).
    - Historias de usuario propuestas por el Scrum Team para mejorar el desempeño del sistema.
- Este ejercicio se debe llevar a cabo al finalizar cada Sprint por el Scrum Master y mostrar los avances en las reuniones de retrospectiva.
- Participantes: Scrum Master, Product Owner

5.3 Representación gráfica de la Propuesta de Solución

El siguiente gráfico representa un resumen visual de la propuesta de solución. También especifica la cantidad necesaria y orden de los Sprints para cada una de las fases de los tres Pilares (Responsabilidad Compartida, Autonomía y Retrospectiva). Al lado izquierdo se especifica los participantes necesarios para cada una de las etapas. (M: Manager, SM: Scrum Master, PO: Product Owner. ST: Scrum Team)

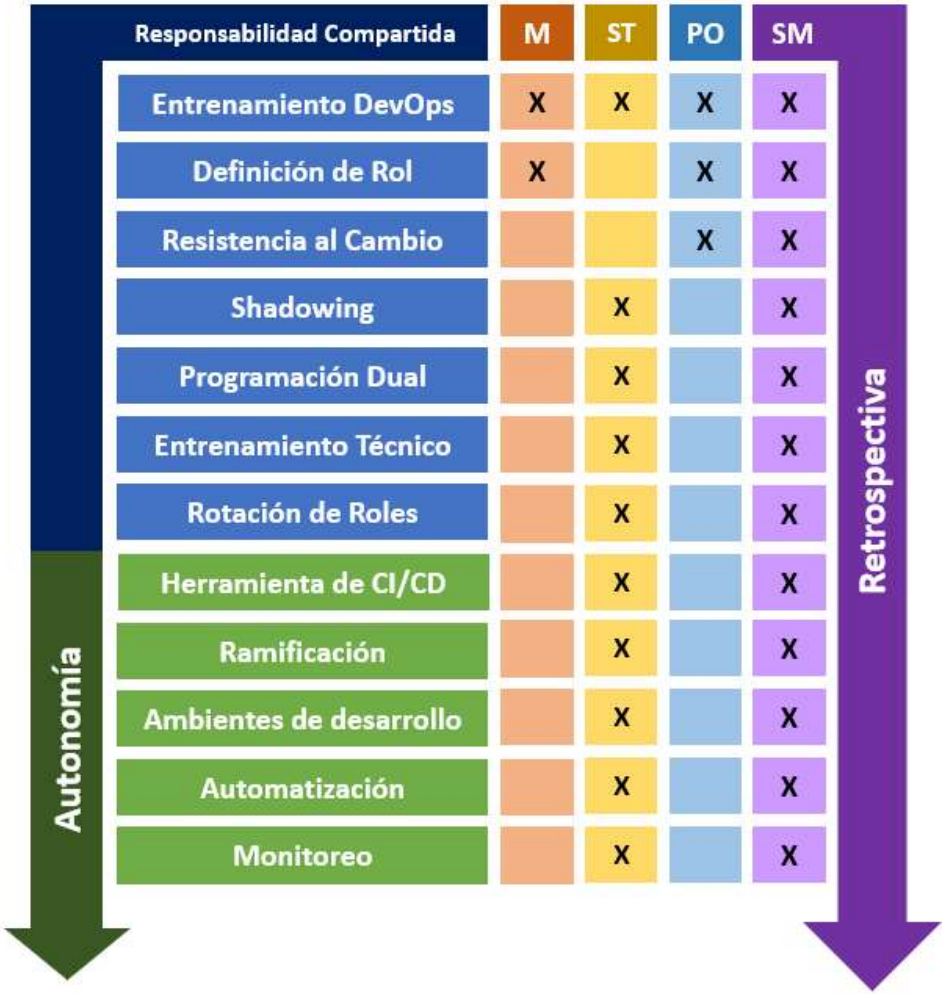

Figura 5.3.1 (Elaboración Propia)

5.4 Plan Piloto

Debido a la gran escala que compone la implementación de los tres pilares en los que se funde la solución de la propuesta, se escoge trabajar en un plan piloto basado en uno de los componentes de un pilar, el de Responsabilidad Compartida, desarrollando durante un Sprint un proceso de Shadowing. Se escoge implementar un proceso de Shadowing debido que se ajustaba a los tiempos propuestos por el equipo de trabajo para trabajar en un plan piloto, además no requería adquisición de equipo o licencias para desarrollarla.

La prueba fue efectuada en la Iteración comprendida entre el 26 de junio al 10 de Julio del 2019. Se tomó del Backlog una historia planeada para ese Sprint de carácter técnico (Desarrollo de un formulario en JavaScript para Security Operations), esta historia de usuario se le dieron 3 puntos de esfuerzo durante el Sprint Planning.

La Scrum Master del equipo asignó como dueño primario de la historia al Líder Técnico del equipo, quien tiene gran experiencia en este tipo de historias de usuario. Y se nombró como secundario a un miembro no técnico del Scrum Team, quien tomaría el rol de visor en la ejecución de la historia.

El miembro secundario fue el encargado de programar sesiones junto al primario para poder observar la elaboración del formulario en JavaScript. Cabe mencionar que el miembro secundario no tiene una formación netamente en programación, sino que funge como analista de calidad en el grupo.

La terminación de la historia tomó tres sesiones de una hora, efectuadas en tres días diferentes durante el Sprint mencionado previamente. Las sesiones no incluyeron el movimiento a producción del cambio, ni tampoco un demo con el cliente, como parte de los criterios de aceptación de la historia impuestos por la PO del equipo. Las tres sesiones se enfocaron en el desarrollo y programación en JavaScript del formulario requerido.

Antes de terminar el Sprint, y una vez entregada y aceptada la historia de usuario, se tuvo una reunión de media hora con los involucrados en la ejecución de la misma, también se invitó a la reunión a la Scrum Master y PO del equipo. La reunión se realizó el 9 de Julio del 2019, vía Skype.

El resultado de la aplicación del plan piloto fueron los siguientes:

- Existió una confusión entre la mecánica de trabajo que existe para el Shadowing y para la Programación Dual, pues el primario de la historia de usuario le dio ciertas partes de la programación del formulario al miembro secundario, con un constante monitoreo, sin embargo.
- El miembro primario menciona que al tener que compartir y explicar lo que estaba trabajando, el tiempo de completación de la historia de usuario tomó más de tres horas, según su criterio, de no haber tenido que hacer una sesión de Shadowing, hubiese tomado solo una.
- El miembro secundario menciona la importancia de las habilidades blandas necesarias para el Shadowing por parte de ambos participantes, entre ellas se mencionan: Capacidad para explicar, paciencia, disposición para aprender, comunicación y un buen ambiente entre los miembros del equipo.
- Los dos miembros mencionan que la historia de usuario desarrollada permitió una buena sesión de Shadowing, puesto que no presentaba una gran complejidad.
- Se le preguntó al miembro secundario si estaría cómodo en aceptar una historia de usuario similar para una futura iteración, a lo cual contestó que no se siente cómodo y evita este tipo de historias técnicas.
- Se le preguntó al miembro primario si estaría cómodo en aceptar una sesión de Programación Dual o Shadowing, a lo cual contestó que estaría de acuerdo.
- Se le preguntó al miembro secundario si estaría cómodo en aceptar una sesión de Programación Dual o Shadowing, a lo cual contestó que estaría de acuerdo.
- Se le preguntó al miembro secundario si estaría de acuerdo en transformar su rol actual y convertirlo en un rol más técnico, a lo que contestó que sí estaría de acuerdo, pero no es algo en lo que se halla especializado y que tampoco disfrutaría de realizar.
- La Scrum Master ve la importancia de tener miembros polifuncionales en el equipo, pero también está en contra de que se tengan que transformar los roles en los que trabajan actualmente y no fueron originalmente contratados.
- La Product Owner del equipo concuerda con la Scrum Master. Rescata el beneficio que tendría para acelerar la entrega de historias si se pudiese contar con más miembros técnicos en el equipo.

La historia fue aceptada por la PO y llevada a producción antes de finalizar el Sprint. En el Sprint Planning siguiente, no se programó ninguna sesión de Shadowing ni de Programación Dual para la siguiente iteración. La siguiente figura resume el proceso que se siguió en el plan piloto y sus resultados.

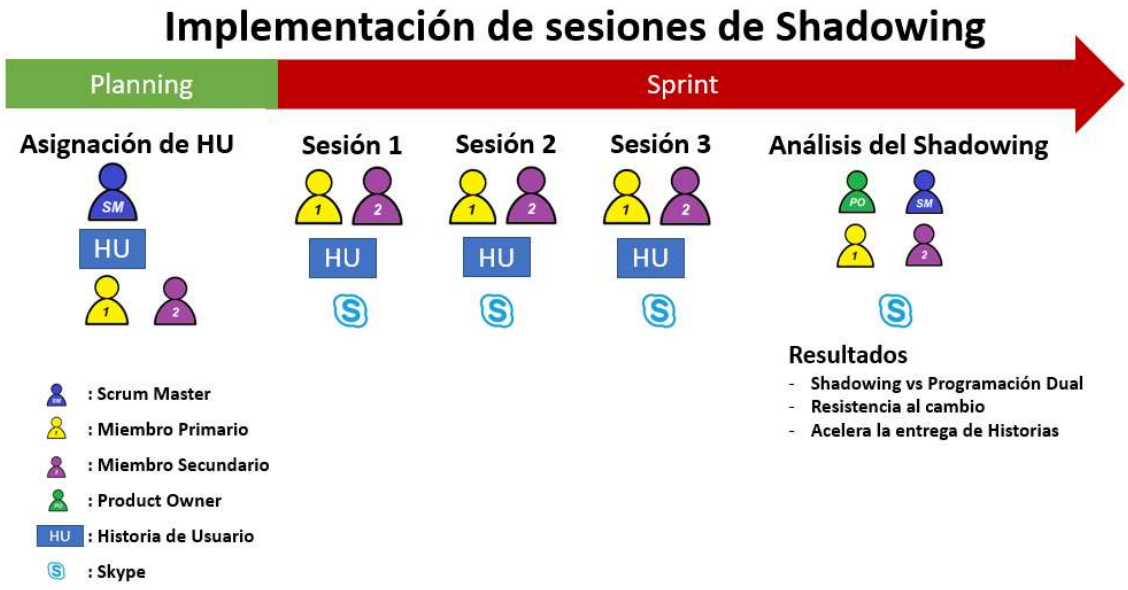

Figura 5.4.1 (Elaboración Propia)

CAPÍTULO 6: ANÁLISIS FINANCIERO

Este capítulo presenta el análisis financiero de la investigación planteada y desarrollada, donde se muestran los costos y los beneficios esperados por el proyecto SecurityOps.

Cabe destacar que no se pretende adquirir activos fuera de las estructuras actuales que provee el Departamento de IT de Intel®, por ejemplo, máquinas virtuales, contenedores o programas que cuenta con licencias empresariales. Aunque la adquisición de una máquina virtual, entrenamientos, como Lynda o Udemy, o de una licencia ya aprobada de una aplicación, no se cobra al centro de costos del equipo, estas tienen un costo para el departamento de IT. Al tratarse de contratos con alto grado de confidencialidad, la información relacionada al costo de este tipo de servicios no se pudo obtener por parte de la empresa, por lo tanto, se utilizan supuestos basados en los costos de esos servicios en empresas especializadas, por ejemplo, el costo de una máquina virtual en un contrato con AWS.

En este ejercicio, no se va a contar con herramientas con las que el equipo ya cuenta, por ejemplo, CA Technologies (Rally) para el seguimiento de las historias de usuario, o GitLab como repositorio de código.

Con la aplicación de este análisis, se pretende obtener el beneficio económico que trae la aplicación de DevOps en el equipo: Reducción de horas de trabajo y evitar la contratación de personal.

6.1 Estimación de los costos del proyecto

Para obtener el costo por hora de cada uno de los miembros del equipo, se toma como base el índice salarial del segundo semestre del 2018 de la Caja Costarricense de Seguro Social para un perfil de Analista en Sistemas 4 en TIC (Se toma este perfil puesto que es el más alto entre los puestos técnicos), el salario base es de ₡800,150 colones (Índice Salarial Julio. CCSS. 2018).

Se considera el mes de 30 días, se toma la referencia de anualidad y dedicación exclusiva como los bonos y promociones anuales que la empresa da a sus colaboradores. Además, intentando ajustar lo mayormente posible a la realidad de la empresa, se agrega un 57% más al salario base, tomando como referencia el artículo publicado por la revista Summa titulado “En Costa Rica profesionales bilingües ganan hasta un 57% más de salario”. (2017)

Una vez considerados los detalles previamente mencionados, se concluye la siguiente gráfica:

Rubros	Detalle
Salario base	₡1,256,235.50
Anualidad	₡00.00
Dedicación exclusiva 35%	₡00.00
Cargas Sociales 26.33%	₡330,766.80
Aguinaldo 8.33%	₡104,644.41
Meses laborados al año	12
Semanas laboradas al año	52
Horas laboradas por semana	40
Horas laboradas por día	8
Salario Mensual	**₡1,691,646.71 ($2,967.49)**
Costo por hora	**₡7,048.52 ($12.36)**

Tabla 6.1.1 (Elaboración Propia)

Para obtener el costo por hora de trabajo de los colaboradores, se divide el salario mensual entre 30 días para obtener el costo de un día y este a su vez se divide entre las horas laboradas por día, con lo cual se obtiene que la hora para el perfil de Analista en Sistemas 4 tiene un costo de 7,048.52 colones. Se utiliza el tipo de cambio de dólar de ₡571.00 de acuerdo al Banco Central el tres de agosto del 2019. Se usa dólares para representar los costos debido a que los precios que ofrecen las soluciones de almacenaje, entrenamientos y monitoreo proporcionan su costo en esa moneda.

En el caso de los entrenamientos para aprender sobre DevOps, se toma como base el costo mensual de la plataforma Lynda, plataforma utilizada por la empresa para la adquisición de conocimiento de sus colaboradores. El costo mensual de Lynda es de $29.99 de acuerdo con su página web el 3 de agosto del 2019. Se divide entre 30 días para obtener el costo por día de la plataforma ($0.99). Es importante aclarar que estos costos cubren a un equipo de más de diez personas durante el tiempo establecido.

Para los entrenamientos técnicos, al igual que el de resistencia al cambio, la plataforma de Lynda cuenta con gran cantidad de tutoriales de múltiples temas de scripting, bases de datos

o networking, sin embargo, también se cuentan los entrenamientos de ServiceNow Scripting, con un costo de $2,400 por persona, según su página web el 3 de agosto de 2019 y MariaDB for DBA's, con un costo de $2,400 por persona. Estos dos entrenamientos los llevaran los dos miembros no técnicos del equipo. Es importante mencionar que estos entrenamientos y plataformas son sugeridas, puesto que parte de las actividades es una definición en conjunto de cuáles son los entrenamientos, cual proveedor y modalidad es la que el equipo considera mejor para sus necesidades. Con respecto a las actividades de Shadowing, Programación Dual, Definición de Rol y Rotación, no se toma ningún valor externo (Como entrenamientos o equipo), pues para dichas actividades se ocupa solamente el costo por hora de los miembros del equipo. En el caso especial de Rotación de Roles, no se considera un costo asociado ya que son actividades que el Scrum Team actualmente realiza, no son actividades propias de este proyecto.

En el segundo pilar, Automatización, se considera para la herramienta de CI/CD el uso de Jenkins o TeamCity, el cual no tiene un costo asociado por instalación o uso de licencias. Aunque se menciona al inicio de este capítulo, el equipo ya cuenta con GitLab como repositorio para ramificación de código, este también es un programa Open Source sin un costo asociado. En el caso de automatización Ansible, Puppet y Chef son programas libres sin costo asociado. Para monitoreo, se utiliza el precio mensual de New Relic, sugerencia de herramienta de monitoreo, el cual es de $42. Es parte de la historia de usuario definir cuál es la mejor herramienta de monitoreo para los sistemas del equipo.

Para el hospedaje de estas aplicaciones, se utiliza como base los precios sugeridos de Amazon Web Services, se utiliza la calculadora que AWS ofrece para crear la estimación. La especificación sugiere utilizar una instancia de Amazon EC2, con 1 TB de espacio de almacenaje, con 4 CPU's y 16GiB de memoria, utilizando una escala t3a.xlarge con una reservación de tres años (Duración propuesta del proyecto), esta arquitectura propuesta tiene un costo mensual de $150 mensuales, de acuerdo con los datos extraídos el 3 de agosto de 2018. Cabe mencionar que la empresa cuenta con servicios propios para almacenaje, pero no se brindaron los datos, es por eso que se utiliza como referencia las soluciones de AWS.

Para el pilar de Retrospectiva se toma solamente el costo de las horas que invierte el colaborador del equipo en la creación y uso de la herramienta de retrospectiva.

El proyecto tiene una duración de tres años (156 Semanas). Un Sprint constituye dos semanas en total (78 Sprints en el periodo de tres años). Los gastos asociados al proyecto se presentan con respecto al desglose de cada uno de los tres pilares propuestos en la

propuesta de solución, se muestra el nombre de la actividad, la cantidad de Sprints necesarios para desarrollarla, una columna “Costo Externo” que muestra el costo asociado a un entrenamiento, licencia o programa que debe ser contratado externamente. Se agrega una columna de “Hardware” que muestra costos asociados al almacenaje de las aplicaciones, “Miembros” especifica la cantidad de miembros del equipo necesarios para llevar a cabo la actividad y junto a esa columna está “Salario por hora”, que muestra el costo del salario que reciben los miembros del equipo involucrados por hora laborada, en esa casilla se refleja el salario por hora y una estimación de la cantidad de horas necesarias para completar la actividad. La última columna “Total” captura el costo de la actividad sumando la cantidad de miembros, Sprints y resto de la información mencionada.

Pilar	Actividad	Sprints	Costo Externo	Miembros	Salario x H	Total
RC	**Entrenamiento DevOps**	**2**	$9.99 x Sprint	7	$12.36 x 8	$1,404.30
RC	**Entrenamiento Resistencia**	**1**	$9.99 x Sprint	2	$12.36 x 20	$504.39
RC	**Entrenamiento Técnico**	**1**	$9.99 x Sprint	2	$12.36 x 20	$504.39
RC	**Entrenamiento QA**	**1**	$9.99 x Sprint	3	$12.36 x 20	$380.79
RC	**Entrenamiento Scripting**	**1**	$2,400.00	2	$12.36 x 20	$5,294.4
RC	**Entrenamiento DBA**	**1**	$2,400.00	2	$12.36 x 20	$5,294.4
RC	**Definición Rol**	**1**	$00.00	3	$12.36 x 2	$74.16
RC	**Shadowing**	**2**	$00.00	3	$12.36 x 4	$296.64
RC	**Programación Dual**	**2**	$00.00	3	$12.36 x 4	$296.64
RC	**Rotación de Roles**	**67**	$00.00	4	$00.00	$00.00
AU	**Instalar / Ejecutar CI/CD**	**67**	$00.00	1	$12.36 x 10	$8,281.20
AU	**Instalar / Ejecutar Ramificación**	**67**	$00.00	1	$12.36 x 10	$8,281.20
AU	**Instalar / Ejecutar Ambientes**	**67**	$00.00	1	$12.36 x 10	$8,281.20
AU	**Instalar / Ejecutar Automatización**	**67**	$00.00	1	$12.36 x 10	$8,281.20
AU	**Instalar / Ejecutar Monitoreo**	**67**	$21.00 x Sprint	1	$12.36 x 10	$9,688.20
AU	**Hardware Almacenaje**	**67**	$75 x Sprint	0	$00.00	$5,475.00
RT	**Retrospectiva**	**67**	$00.00	1	$12.36 x 1	$902.28
						$62,338.11

Tabla 6.1.2 (Elaboración Propia)

6.2 Obtención de Beneficios

El beneficio monetario asociado al proyecto plantea la estrategia de evitar la contratación de un nuevo miembro técnico al equipo. Este ejercicio financiero se enfoca en la supuesta contratación de solamente un colaborador nuevo, el cual devenga un salario con todas las cargas sociales asociadas. Esto bajo el caso hipotético de que el equipo ocupe solamente un nuevo recurso, puesto que está fuera del alcance del proyecto determinar cuántos recursos son realmente necesarios para aumentar el Scrum Velocity del equipo.

Aunque no es parte de este análisis financiero, el cual busca mostrar la rentabilidad del proyecto, al implementar DevOps la reducción de horas en el esfuerzo dedicado a una tarea baja drásticamente, al automatizar procesos y evitar cuellos de botella. Esas horas que se ahorran mediante las prácticas que propone DevOps no son parte del alcance de este análisis.

La siguiente tabla muestra el costo en el que incurriría el equipo durante tres años suponiendo que se debe contratar un nuevo recurso, el cual ganaría hipotéticamente el mismo salario planteado en esta solución. A este salario se le suma un 2% en cada año debido al factor de inflación. Para capturar estas cifras, se utiliza el salario mensual calculado previamente, se le agrega un aumento de 2% y se multiplica por 12 meses.

Detalle	Primer Año	Segundo Año	Tercer Año
Salario Anual	**$35,609.88**	**$36,321.96**	**$37,048.32**
Total devengado durante los tres períodos			**$108,980.16**

Tabla 6.2.1 (Elaboración Propia)

6.3 Evaluación Financiera

En este apartado se analizan los conceptos de costos, ahorros y utilidad que el proyecto propone.

En el primer año se suman los costos de las actividades relacionadas al pilar de Responsabilidad Compartida ya que son las primeras actividades en realizarse, y se llevan a cabo el primer año (Entrenamiento DevOps, Entrenamiento Técnico, Entrenamiento Scripting, Entrenamiento DBA, Definición Rol, Shadowing y Programación Dual, para un total de $14,050.11). A este primer periodo también se agrega el costo asociado al hardware

necesitado ($5,475.00). Esta suma es la inversión inicial para la realización del proyecto ($19,525.11).

Las actividades correspondientes a los pilares de Autonomía y Retrospectiva (Y la actividad de Rotación de Roles, perteneciente al pilar de Responsabilidad Compartida), al ser iterativos, se dividen entre los tres periodos, para un total de $14,271.00 por periodo. En el caso del segundo y tercer año, solo cuentan con esta cifra, puesto que gran parte de la inversión se lleva a cabo en el primer año. En la fila de utilidad, se muestra la resta entre los costos y los ahorros generados.

En la siguiente tabla se muestran los valores en la diferencia entre los ingresos y los ahorros de los tres periodos analizados. Se describe el cálculo del valor presente neto y la tasa interna de retorno del proyecto.

Detalle	Primer Año	Segundo Año	Tercer Año
Costos	**$33,796.11**	**$14,271.00**	**$14,271.00**
Beneficios	**$35,609.88**	**$36,321.96**	**$37,048.32**
Utilidad	**$1,813.77**	**$22,050.96**	**$22,777.32**
Valor Actual Neto (VAN)			**$21,028.47**
Tasa Interna de Retorno (TIR)			**43.98%**

Tabla 6.3.1 (Elaboración Propia)

Para la evaluación de la viabilidad del proyecto se utiliza el valor actual neto (VPN) también conocido como VAN (Valor Actual Neto), el cual es una técnica para elaboración del presupuesto de capital que se calcula restando la inversión inicial de un proyecto del valor presente de sus flujos de entrada de efectivo descontados a una tasa equivalente al costo de capital de la empresa. (Gitman & Zutter, 2012)

El VPN se obtiene restando la inversión inicial de un proyecto (FE0) del valor presente de sus flujos de entrada de efectivo (FEt) descontados a una tasa (k) equivalente al costo de capital de la empresa. (Gitman & Zutter, 2012)

Se extrae el 7 de agosto de 2019 del sitio web del Banco Central de Costa Rica (http://www.bccr.fi.cr) la tasa básica pasiva (TBP) 5.9%, esta es la tasa de descuento utilizada en este cálculo. La inversión inicial es de $19,032.95 con una tasa de retorno fijada en 5.9%.

La figura siguiente muestra la fórmula utilizada para el cálculo del VPN.

$$\text{VPN} = \text{Valor presente de las entradas de efectivo} - \text{Inversión inicial}$$

$$\text{VPN} = \sum_{t=1}^{n} \frac{FE_t}{(1+k)^t} - FE_0$$

Figura 6.1.3 (Fuente: Gitman & Zutter)

Basado los resultados de la Tabla 6.3.1, se concluye la viabilidad del proyecto dado el valor positivo del VPN.

Adicionalmente, se considera la tasa interna de rendimiento o retorno (TIR), la cual según Gitman & Zutter (2012): Es la tasa de descuento que iguala el VPN de una oportunidad de inversión con $0 (debido a que el valor presente de las entradas de efectivo es igual a la inversión inicial); es la tasa de rendimiento que ganará la empresa si invierte en el proyecto y recibe las entradas de efectivo esperadas.

La tasa interna de rendimiento que se muestra en la Tabla 6.3.1, de 32.45% es mayor que la tasa de descuento definida en 5.9%.

CAPÍTULO 7: CONCLUSIONES Y RECOMENDACIONES

7.1 Conclusiones

Las siguientes son las conclusiones obtenidas después del desarrollo de este documento:

1. Al finalizar este proyecto, se propone y entrega una estrategia de implementación de la práctica DevOps para el proyecto SecurityOps. Se determina que DevOps aumenta la frecuencia de historias de usuario que un equipo puede entregar debido a sus principios, entre los que destacan la integración y entrega continua, aunado a la alta automatización de procesos.
2. Se concluye que el marco de trabajo DevOps es un excelente complemento para las metodologías ágiles, en el caso estudiado fue Scrum, puesto que maximiza la entrega de historia de usuario.
3. Por medio de una amplia revisión de lectura se determinan los conceptos que la práctica DevOps propone para agilizar proceso de desarrollo y ejecución de las historias de usuario en un equipo Scrum, con este marco teórico desarrollado se cuenta con referentes conceptuales sólidos para la propuesta de solución para la implementación de DevOps en el problema a tratar.
4. Se realizó un diagnóstico del estado del Scrum Team en cuanto a la cantidad de historias de usuario, por medio de este ejercicio se logran capturar las áreas de mejora donde las prácticas DevOps son ideales para el mejoramiento de estas.
5. Se realizó un plan piloto para recolectar retroalimentación del equipo con respecto a la propuesta de solución. Por los alcances de la propuesta, se desarrolló solamente una actividad (Shadowing) la cual mostró dos conclusiones muy relevantes:
    a. PO y la Scrum Master del equipo piensan que la actividad de Shadowing es de gran beneficio para la reducción de la deuda técnica, la cual mejoraría la entrega de historias de usuario.
    b. Se encuentra una resistencia al cambio por parte de los miembros del Scrum Team al tener que hacer actividades fuera de su área experta.
6. La implementación de las prácticas DevOps no está sujetas a límites de tiempo, no existe un final establecido donde se pueda concluir el proyecto, puesto que la naturaleza propia de DevOps hace que exista una mejora continua, iterativa e incremental, concept como la automatización de procesos son constantes en el diario hacer de las operaciones y desarrollo. La madurez del equipo en DevOps es pieza clave para la correcta implementación del mismo.
7. Es una gran ventaja que el equipo de trabajo haya trabajado durante mucho tiempo bajo las metodologías ágiles, en este caso Scrum, pues muchas de las prácticas de Scrum facilitan el entendimiento de los principios DevOps, por ejemplo las ceremonias

de Retrospectiva permiten detectar falencias en un proceso, el cual con DevOps puede ser rápidamente solucionado.

8. El análisis financiero demuestra la viabilidad del proyecto solamente comparando el gasto en el que debe incurrir SecurityOps para implementar DevOps versus el gasto que supondría contratar un nuevo recurso. Sin embargo, un punto sumamente importante que no se menciona en ese análisis, es la cantidad de historias de usuario que el equipo puede entregar, las cuales benefician al cliente en su negocio, ahorrando tiempos o también recursos. Estos esfuerzos sumados significan una ganancia para la compañía.
9. Se destaca la comprensión de la situación por parte del equipo del proyecto SecurityOps. El entender sus debilidades, fortalezas y oportunidades de mejora es muestra de la madurez que tiene el equipo, en gran medida por los principios que Scrum dicta. Esta madurez es fundamental y crítica para una adopción integral y satisfactoria de DevOps.
10. Es importante recalcar que el perfil de un ingeniero DevOps es directamente proporcional a su experiencia en varios ambientes de trabajo. En otras palabras, no existe un perfil de ingeniero DevOps “Junior”, ya que para llegar a ese punto es necesaria la experiencia.

7.2 Recomendaciones

Una vez finalizado este documento, se detectan varias situaciones y escenarios que permiten formular las siguientes notas y sugerencias.

1. Durante la fase de diagnóstico, se detectaron y mencionaron varios aspectos de mejora, por ejemplo, la elaboración de una mejor redacción de las historias de usuario, así como su “Definition of Done”. Esto es necesario para una fácil comprensión de las tareas por parte del Scrum Team.
2. Detectar las dependencias con equipos terceros lo antes posible por parte de la Product Owner, pues muchas historias de usuario se ven retrasadas por estas dependencias.
3. De aplicar el proyecto, es fundamental hacer un enfoque en el factor humano. DevOps propone un cambio para la mayoría de los roles en el equipo, lo cual genera resistencia por parte del mismo. Por este motivo, es esencial por parte de la Product Owner y Scrum Master, trabajar internamente la gestión del cambio cultural para aclarar dudas y superar miedos a posibles riesgos.
4. Se recomienda para el equipo del proyecto SecurityOps enfocar la capacitación entre los miembros del equipo, así como la documentación de procesos. Un grupo DevOps

maduro puede verse afectado por la salida de miembros y la llegada de nuevos integrantes, la cual puede ser minimizada mediante la polifuncionalidad del equipo para cubrir ciertas áreas, aunque signifique bajar el Scrum Velocity, mientras el nuevo integrante del grupo asimila el ritmo del mismo.

5. En el análisis financiero se hace un ejercicio para la estimación del costo de capacitaciones de acuerdo con sugerencias planteadas por el autor. Sin embargo, el determinar cuáles son las mejores fuentes de entrenamiento tienen que venir de una decisión grupal, no impuesta por un tercero. Ya que esto fomenta la determinación e involucramiento del grupo.
6. De igual manera, sucede con la propuesta de arquitectura para los sistemas de automatización y ambientes de desarrollo. El análisis financiero muestra sugerencias de la arquitectura a seguir, sin embargo, puesto que no entra dentro del alcance del proyecto, no se definen temas como escalabilidad o redundancia. Se le recomienda a la organización tener en cuenta que aplicar DevOps, es una oportunidad idónea para hacer un refrescamiento de la estructura de la solución que soporta el equipo.
7. Este no es el primer caso de implementación de DevOps en la empresa. Bajo esta premisa, se recomienda al Scrum Master y Product Owner realizar sondeos, comparaciones, compartir criterios y experiencias para mejorar la implementación del marco de trabajo.
8. Se recomienda al Scrum Master como parte de la fase de retrospectiva, incluir un análisis financiero de la implementación de DevOps dentro del equipo de trabajo. Se sugiere usar el indicador TIR para considerar el valor económico actual de una decisión tomada.
9. Son muchas las herramientas que ofrece el mercado para la implementación de las prácticas propuestas por DevOps. Se recomienda al proyecto SecurityOps realizar una amplia investigación de las mismas, con sus fabricantes, y adecuar la realidad del grupo con las soluciones que estas herramientas pueden brindar. En el análisis financiero, se sugieren muchos programas y librerías “Open Source”, así como para monitoreo se sugiere una alternativa con un costo por licencia.
10. Se sugiere al proyecto SecurityOps, realizar un estudio de cuántos puntos aspira el equipo aumentar el Scrum Velocity. Este documento brinda una solución para aumentar este indicador, mas no se definen una cantidad específica.

CAPÍTULO 8: ANÁLISIS RETROSPECTIVO

La realización de este proyecto es el resultado de un proceso de más de dos años cursando la Maestría en Administración de Tecnologías de la Información de la Escuela de Informática de la Universidad Nacional. Significó balancear y priorizar actividades de vida personal y trabajo junto a los cursos de esta Maestría, para lo cual fue necesario la constancia y paciencia por parte de mi persona para poder alcanzar este punto.

Durante la construcción de este documento, viví el cierre de operaciones del departamento donde laboraba en Costa Rica por parte de la empresa. Los plazos para conseguir la información restante se acortaron, lo cual significó redoblar esfuerzos para concluir este proyecto mientras buscaba una nueva oportunidad laboral. El equipo de trabajo en el que se basó este proyecto continuó brindando todo el apoyo necesario para la entrega de la propuesta final. Logré mantenerme dentro de la misma empresa, en un nuevo grupo, lo cual facilitó la comunicación con el equipo de trabajo y permitió continuar con el proyecto.

Una vez concluido este proyecto, con los hallazgos determinados en la fase de diagnóstico y de la propuesta de solución, considero que el enfoque hubiese sido más puntual si el objetivo por alcanzar fuese reducir la deuda técnica del mismo y no el de aumentar el Scrum Velocity del grupo, ya que la mayor parte de la propuesta describe esfuerzos para reducir la deuda técnica, que como consecuencia, trae un aumento en la cantidad de entregas de historias de usuario, por ende, un aumento en el Scrum Velocity.

El plan piloto fue un ejercicio que disfruté mucho, la disposición que mostró el equipo para hacerlo y la dinámica de grupo durante el mismo fue muy agradable, aunque arroja comentarios que demarcan una resistencia al cambio, sin embargo, considero que el nivel de profesionalismo del equipo es muy alto al igual que el grado de compromiso que tienen los miembros de SecurityOps, características que considero fundamentales para la aplicación de DevOps, y si se determinará aplicar este proyecto, no tengo dudas de que el equipo lo implementará de la mejor manera y con la mejor actitud.

Finalmente, mencionar que aplicar DevOps en un proyecto no tiene una fecha de finalización ni un estado donde se puede considerar dominado. Al adoptar DevOps como marco de trabajo, se está adoptando una cultura integral, constante e incremental, que tiene como objetivo depurarse más en cada iteración, y no solo hace al equipo entregar más historias de usuario, también convierte a los miembros del equipo en mejores y más capaces profesionales.

CAPÍTULO 9: REFERENCIAS BIBLIOGRÁFICAS

1. Roche J. (2013) Adopting DevOps Practices in Quality Assurance. Merging the art and science of software development. Volume 11, issue 9.

2. Sacks M. (2012) DevOps Principles for Successful Web Sites. In: Pro Website Development and Operations. Apress, Berkeley, CA

3. Lwakatare, L.E., Mäkinen, S., Männistö, T., Riungu-Kalliosaari, L., & Tiihonen, J. (2016). DevOps Adoption Benefits and Challenges in Practice: A Case Study. PROFES.

4. C. Ebert, G. Gallardo, J. Hernantes and N. Serrano, "DevOps," in IEEE Software, vol. 33, no. 3, pp. 94-100, May-June 2016.doi: 10.1109/MS.2016.68

5. M. Singh, "U-SCRUM: An Agile Methodology for Promoting Usability," Agile 2008 Conference, Toronto, ON, 2008, pp. 555-560. doi: 10.1109/Agile.2008.33

6. Cho, J. (2008). Issues and Challenges of agile software development with SCRUM. Issues in Information Systems, 9(2), 188-195.

7. Advanced Development Methods, Inc. (2007). Scrum, Retrieved March 19, 2008, from http://www.chaos.com.

8. Schwaber, K. (2004). Agile project management with Scrum. Redmond, WA: Microsoft Press.

9. Schwaber, K., & Beedle, M. (2002). Agile software development with Scrum, Upper Saddle River, NJ: Prentice Hall.

10. L. Rising and N. S. Janoff, "The Scrum software development process for small teams," in IEEE Software, vol. 17, no. 4, pp. 26-32, July-Aug. 2000.

11. C. Ebert, G. Gallardo, J. Hernantes and N. Serrano, "DevOps," in IEEE Software, vol. 33, no. 3, pp. 94-100, May-June 2016.

12. Hüttermann, M. (2012). DevOps for developers. Apress.

13. M. Virmani, "Understanding DevOps & bridging the gap from continuous integration to continuous delivery," Fifth International Conference on the Innovative Computing Technology (INTECH 2015), Pontevedra, 2015, pp. 78-82.

14. Meier, J. (2013, August 30). Generalists vs. Specialists [Web log post]. Retrieved April 4, 2019, from https://blogs.msdn.microsoft.com/jmeier/2013/08/30/generalists-vs-specialists/

15. Parsons, D., Lal, R., Ryu, H., & Lange, M. (2007). Software development methodologies, agile development and usability engineering. ACIS 2007 Proceedings, 21.

16. O. Sohaib and K. Khan, "Integrating usability engineering and agile software development: A literature review," 2010 International Conference On Computer Design and Applications, Qinhuangdao, 2010, pp. V2-32-V2-38.

17. D. J. Reifer, "How good are agile methods?," in IEEE Software, vol. 19, no. 4, pp. 16-18, July-Aug. 2002.

18. Cerpa, N., & Verner, J. (2009). Why did your project fail? Verner Communications of the ACM, 130-134.

19. Pressman, A. (2019, January 31). Intel Makes Surprise CEO Pick After Seven-Month Search. Fortune.

20. Gurpreet Singh Matharu, Anju Mishra, Harmeet Singh, and Priyanka Upadhyay. 2015. Empirical Study of Agile Software Development Methodologies: A Comparative Analysis. SIGSOFT Softw. Eng. Notes 40, 1 (February 2015), 1-6. DOI: https://doi.org/10.1145/2693208.2693233

21. Mahalakshmi, M., & Sundararajan, M. (2013). Traditional SDLC Vs Scrum Methodology–A Comparative Study. International Journal of Emerging Technology and Advanced Engineering, 3(6), 192-196.

22. D. P. Harvie and A. Agah, "Targeted Scrum: Applying Mission Command to Agile Software Development," in IEEE Transactions on Software Engineering, vol. 42, no. 5, pp. 476-489, 1 May 2016.

23. S. Downey and J. Sutherland, "Scrum Metrics for Hyperproductive Teams: How They Fly like Fighter Aircraft," 2013 46th Hawaii International Conference on System Sciences, Wailea, Maui, HI, 2013, pp. 4870-4878.

24. Draffin, A. (2010). Methodology vs framework: why waterfall and agile are not methodologies.

25. Salazar EJG, Guerrero PJC, Machado RYB, Cañedo AR Clima y cultura organizacional: dos componentes esenciales en la productividad laboral Revista Cubana de Información en Ciencias de la Salud (ACIMED) 2009; 20 (4)

26. S. Dorairaj, J. Noble and P. Malik, "Knowledge Management in Distributed Agile Software Development," 2012 Agile Conference, Dallas, TX, 2012, pp. 64-73.

27. G. Melnik and F. Maurer. Direct verbal communication as a catalyst of Agile knowledge sharing. In Proceedings of the Agile Development Conference, pages 21-31, June 2004.

28. T. H. Davenport and L. Prusak. Working Knowledge: How Organizations Manage What They Know. Harvard Business School Press, Boston, MA, USA, 1998

29. J. Smith. An approach to developing a performance brief at the project inception stage. Architectural Engineering and Design Management, 1(1):3-20, 2005.

30. M. Coram and S. Bohner. The impact of Agile methods on software project management. In The 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems, pages 363-370, April 2005.

31. A. Martin, R. Biddle, and J. Noble. The XP customer role in practice: Three studies. In Proceedings of the Agile Development Conference, pages 42-54, Washington DC, USA, 2004. IEEE Computer Society.

32. A. Martin, R. Biddle, and J. Noble. The XP customer team: A grounded theory. In Proceedings of the AGILE, pages 57-64, 2009.

33. T. Chau, F. Maurer, and G. Melnik. Knowledge sharing: Agile methods vs. Tayloristic methods. In Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. WET

ICE 2003. Proceedings. Twelfth IEEE International Workshops on, pages 302-307, June 2003.

34. R. McDermott. Learning across teams: The role of communities of practice in team organization. Knowledge Management Review, 2, 1999.

35. L. Rising and N. S. Janoff. The scrum software development process for small teams. Software, IEEE, 17(4):26-32, July/Aug 2000.

36. D. W. Palmieri. Knowledge management through pair programming. Master's thesis, North Carolina State University, 2002.

37. G. Melnik and F. Maurer. Direct verbal communication as a catalyst of Agile knowledge sharing. In Proceedings of the Agile Development Conference, pages 21-31, June 2004.

38. S. Cohan. Successful customer collaboration resulting in the right product for the end user. In AGILE Conference, pages 284-288, 2008.

39. Cook T.D & Reichardt, Ch. Métodos cualitativos y cuantitativos en la investigación evaluativa. Madrid. Edit. Moranta (1986)

40. Albaum, G. (1997). The Likert Scale Revisited. Market Research Society. Journal., 39(2), 1–21. https://doi.org/10.1177/147078539703900202

41. C. A. Cois, J. Yankel and A. Connell, "Modern DevOps: Optimizing software development through effective system interactions," 2014 IEEE International Professional Communication Conference (IPCC), Pittsburgh, PA, 2014, pp. 1-7.
doi: 10.1109/IPCC.2014.7020388

42. Lethbridge, Timothy & Sim, Susan & Singer, Janice. (2005). Studying Software Engineers: Data Collection Techniques for Software Field Studies. Empirical Software Engineering. 10. 311-341. 10.1007/s10664-005-1290-x.

43. McDowell, C., Werner, L., Bullock, H. E, & Fernald, J. (2006). Pair programming improves student retention, confidence, and program quality. Communications of the ACM, 49(8), 90-95. http://dx.doi.org/10.1145/1145287.1145293 Retrieved from https://escholarship.org/uc/item/1s49s13f

44. Gitman, L. J., & Zutter, C. J. (2012). Principios de Administración Financiera. México: Pearson Educación de México.

45. Khurana, H. & Sohal, J.S. Agile: The necessitate of contemporary software developers. International Journal of Engineering Science & Technology, 3(2), 1031-1039, 2011

46. Beck, K.. "Extreme Programming Explained. Embrace Change", Pearson Education, 1999. Traducido al español como: "Una explicación de la programación extrema. Aceptar el cambio", Addison Wesley, 2000.

47. M. O. Ahmad, J. Markkula and M. Oivo, "Kanban in software development: A systematic literature review," 2013 39th Euromicro Conference on Software Engineering and Advanced Applications, Santander, 2013, pp. 9-16. doi: 10.1109/SEAA.2013.28

48. (2018, August 14). En Costa Rica profesionales bilingües ganan hasta un 57% más de salario. Revista Summa.

49. Puppet Labs and IT Revolution Press. 2018 state of devops report (2018). https://puppet.com/system/files/2016-06/2016%20State%20of%20DevOps%20Report_0.pdf (Accedido el 1 de agosto de 2019)

ANEXOS

ANEXO I. Encuestas Realizadas para la autoevaluación

Autoevaluación: Líder Técnico

Autoevaluación del Equipo

Fecha: 04/17/2019

Puntuación: 0 - Nunca, 1 - Rara vez 2 - Ocasionalmente 3 - Seguido 4 - Muy Seguido 5 - Siempre

Área / Pregunta	Puntuación
El Product Owner facilita el desarrollo de historias de usuarios, priorización y negociación	4
El Product Owner colabora de forma proactiva con Product Management y otras partes interesadas	4
Las historias de usuarios son pequeñas, estimadas, funcionales, verticales y se ajustan a una iteración	4
El Product Owner facilita el desarrollo de criterios de aceptación que se utilizan en la planificación, revisión y aceptación de la historia	4
El equipo refina el Team Backlog en cada iteración	4
Total Puntuación de la Evaluación del Product Owner	**0.0**
El equipo planea la Iteración de manera colaborativa, efectiva y eficiente.	4
El equipo siempre tiene objetivos de iteración claros, en apoyo de los objetivos de la empresa, y se compromete a cumplirlos	4
El equipo aplica los criterios de aceptación y aplica Definition of Done a la aceptación de la historia.	4
El equipo tiene una velocidad predecible y normalizada que se utiliza para estimar y planificar	4
El equipo cumple regularmente sus objetivos de iteración.	4
Total Puntuación de la Evaluación las Iteraciones	**0.0**
Los miembros del equipo son autoorganizados, se respetan mutuamente, se ayudan mutuamente a cumplir los Objetivos de iteración, gestionan las interdependencias y se mantienen sincronizados entre sí.	4
El Scrum Master asiste a Scrum of Scrums e interactúa con RTE según corresponda	4
Las historias se completan a lo largo de la Iteración con múltiples ciclos de definir-construir-prueba (es decir, la Iteración no es catastrófica)	4
El equipo se reúne todos los días a la misma hora y lugar para el Stand-up diario para coordinar su trabajo y resolver los impedimentos.	5
El equipo realiza una retrospectiva después de cada Iteración y realiza cambios incrementales para mejorar continuamente su rendimiento	5
Total Puntuación de la Evaluación del Equipo	**0.0**

El equipo reduce activamente la deuda técnica en cada iteración.	1
El equipo tiene una guía y comprensión claras de la arquitectura de los proyectos, pero es lo suficientemente libre y flexible para permitir que el diseño emergente respalde la implementación óptima	1
Se realizan pruebas automatizadas de aceptación y pruebas unitarias son parte de la historia y su Definition of Done	1
La refactorización es constante en las iteraciones.	3
Constante mejora en infraestructura de automatización de construcción y prueba	3
El equipo genera nuevas hipótesis y las prueba continuamente.	2
El equipo está desplegando continuamente a producción.	3
El equipo diseña su trabajo para permitir el despliegue continuo, el lanzamiento bajo demanda y la recuperación	2
Total Puntuación de la Evaluación Técnica del Equipo	**0.0**
Total Puntuación del Equipo	

Autoevaluación: Desarrollador

Autoevaluación del Equipo

Fecha: 04/17/2019

Puntuación: 0 - Nunca, 1 - Rara vez 2 - Ocasionalmente 3 - Seguido 4 - Muy Seguido 5 - Siempre

Área / Pregunta	**Puntuación**
El Product Owner facilita el desarrollo de historias de usuarios, priorización y negociación	4
El Product Owner colabora de forma proactiva con Product Management y otras partes interesadas	4
Las historias de usuarios son pequeñas, estimadas, funcionales, verticales y se ajustan a una iteración	3
El Product Owner facilita el desarrollo de criterios de aceptación que se utilizan en la planificación, revisión y aceptación de la historia	3
El equipo refina el Team Backlog en cada iteración	3
Total Puntuación de la Evaluación del Product Owner	**0.0**
El equipo planea la Iteración de manera colaborativa, efectiva y eficiente.	4
El equipo siempre tiene objetivos de iteración claros, en apoyo de los objetivos de la empresa, y se compromete a cumplirlos	4
El equipo aplica los criterios de aceptación y aplica Definition of Done a la aceptación de la historia.	4
El equipo tiene una velocidad predecible y normalizada que se utiliza para estimar y planificar	4
El equipo cumple regularmente sus objetivos de iteración.	4
Total Puntuación de la Evaluación las Iteraciones	**0.0**

Los miembros del equipo son autoorganizados, se respetan mutuamente, se ayudan mutuamente a cumplir los Objetivos de iteración, gestionan las interdependencias y se mantienen sincronizados entre sí.	5
El Scrum Master asiste a Scrum of Scrums e interactúa con RTE según corresponda	5
Las historias se completan a lo largo de la Iteración con múltiples ciclos de definir-construir-prueba (es decir, la Iteración no es catastrófica)	5
El equipo se reúne todos los días a la misma hora y lugar para el Stand-up diario para coordinar su trabajo y resolver los impedimentos.	5
El equipo realiza una retrospectiva después de cada Iteración y realiza cambios incrementales para mejorar continuamente su rendimiento	5
Total Puntuación de la Evaluación del Equipo	**0.0**
El equipo reduce activamente la deuda técnica en cada iteración.	2
El equipo tiene una guía y comprensión claras de la arquitectura de los proyectos, pero es lo suficientemente libre y flexible para permitir que el diseño emergente respalde la implementación óptima	2
Se realizan pruebas automatizadas de aceptación y pruebas unitarias son parte de la historia y su Definition of Done	2
La refactorización es constante en las iteraciones.	2
Constante mejora en infraestructura de automatización de construcción y prueba	3
El equipo genera nuevas hipótesis y las prueba continuamente.	2
El equipo está desplegando continuamente a producción.	3
El equipo diseña su trabajo para permitir el despliegue continuo, el lanzamiento bajo demanda y la recuperación	2
Total Puntuación de la Evaluación Técnica del Equipo	**0.0**
Total Puntuación del Equipo	

Autoevaluación: QA

Autoevaluación del Equipo

Fecha: 04/17/2019

Puntuación: 0 - Nunca, 1 - Rara vez 2 - Ocasionalmente 3 - Seguido 4 - Muy Seguido 5 - Siempre

Área / Pregunta	**Puntuación**
El Product Owner facilita el desarrollo de historias de usuarios, priorización y negociación	5
El Product Owner colabora de forma proactiva con Product Management y otras partes interesadas	5
Las historias de usuarios son pequeñas, estimadas, funcionales, verticales y se ajustan a una iteración	3

El Product Owner facilita el desarrollo de criterios de aceptación que se utilizan en la planificación, revisión y aceptación de la historia	4
El equipo refina el Team Backlog en cada iteración	4
Total Puntuación de la Evaluación del Product Owner	**0.0**
El equipo planea la Iteración de manera colaborativa, efectiva y eficiente.	5
El equipo siempre tiene objetivos de iteración claros, en apoyo de los objetivos de la empresa, y se compromete a cumplirlos	5
El equipo aplica los criterios de aceptación y aplica Definition of Done a la aceptación de la historia.	5
El equipo tiene una velocidad predecible y normalizada que se utiliza para estimar y planificar	5
El equipo cumple regularmente sus objetivos de iteración.	5
Total Puntuación de la Evaluación las Iteraciones	**0.0**
Los miembros del equipo son autoorganizados, se respetan mutuamente, se ayudan mutuamente a cumplir los Objetivos de iteración, gestionan las interdependencias y se mantienen sincronizados entre sí.	5
El Scrum Master asiste a Scrum of Scrums e interactúa con RTE según corresponda	5
Las historias se completan a lo largo de la Iteración con múltiples ciclos de definir-construir-prueba (es decir, la Iteración no es catastrófica)	5
El equipo se reúne todos los días a la misma hora y lugar para el Stand-up diario para coordinar su trabajo y resolver los impedimentos.	5
El equipo realiza una retrospectiva después de cada Iteración y realiza cambios incrementales para mejorar continuamente su rendimiento	5
Total Puntuación de la Evaluación del Equipo	**0.0**
El equipo reduce activamente la deuda técnica en cada iteración.	2
El equipo tiene una guía y comprensión claras de la arquitectura de los proyectos, pero es lo suficientemente libre y flexible para permitir que el diseño emergente respalde la implementación óptima	3
Se realizan pruebas automatizadas de aceptación y pruebas unitarias son parte de la historia y su Definition of Done	3
La refactorización es constante en las iteraciones.	3
Constante mejora en infraestructura de automatización de construcción y prueba	2
El equipo genera nuevas hipótesis y las prueba continuamente.	3
El equipo está desplegando continuamente a producción.	2
El equipo diseña su trabajo para permitir el despliegue continuo, el lanzamiento bajo demanda y la recuperación	3
Total Puntuación de la Evaluación Técnica del Equipo	**0.0**
Total Puntuación del Equipo	

ANEXO II. SAFE Team Self Assessment

Team Self-Assessment

Team Name: xxxxxxxxxxxxxxxx Date: xx/x/20xx

Scoring: 0 - Never, 1 - Rarely 2 - Occasionally, 3 - Often, 4 - Very Often, 5 - Always

Área / Question	Score		Comments
Product Ownership Health			
Product Owner facilitates user story development, prioritization and negotiation	3.0		
Product Owner collaborates proactively with Product Management and other stakeholders	3.0		
User Stories are small, estimated, functional and vertical	3.0		
Product Owner facilitates development of acceptance criteria which are used in planning, review and story acceptance	3.0		
Team refines the Team Backlog every Iteration	3.0		
Total Product Health Score	**15.0**	**60%**	
PI Health			
Team participates fully in PI Planning and Inspect and Adapt	3.0		
Team proactively interacts with other teams on the Train as necessary to resolve impediments	3.0		
Team participates in System Demo every two weeks, illustrating real progress towards objectives	3.0		
Team reliably meets 80-100% of committed PI Objectives	3.0		
Total PI Health Score	**12.0**	**60%**	
Iteration Health			
Team plans the Iteration collaboratively, effectively and efficiently	3.0		
Team always has clear Iteration Goals, in support of PI objectives, and commits to meeting them	3.0		
Team applies acceptance criteria and Definition of Done to story acceptance	3.0		
Team has a predictable, normalized velocity which is used for estimating and planning	3.0		
Team regularly delivers on their Iteration goals	3.0		
Total Iteration Health Score	**15.0**	**60%**	
Team Health			
Team members are self-organized, respect each other, help each other complete Iteration Goals, manage interdependencies and stay in-sync with each other	3.0		

Scrum Master attends Scrum of Scrums and interacts with RTE as appropriate	3.0		
Stories are iterated through the Iteration with multiple define-build-test cycles (i.e., the Iteration is not waterfalled)	3.0		
Team meets each day at the same time and place for the Daily Stand-up to coordinate their work and resolve impediments	3.0		
Team holds a retrospective after each Iteration and makes incremental changes to continuously improve its performance	3.0		
Total Team Health Score	**15.0**	**60%**	
Technical Health			
Team actively reduces technical debt in each Iteration	3.0		
Team has clear guidance and understanding of intentional architecture guidance, but is free and flexible enough to allow emergent design to support optimal implementation	3.0		
Automated acceptance tests and unit tests are part of story DoD	3.0		
Refactoring is always underway	3.0		
CI, build and test automation infrastructure is improving	3.0		
Total Technical Health Score	**15.0**	**60%**	
Total Team Score	**72.0**	**60%**	
Radar Chart Data			
Product Ownership Health	**60%**		
PI Health	**60%**		
Iteration Health	**60%**		
Team Health	**60%**		
Technical Health	**60%**		

ANEXO III. Carta de aceptación y entrega de proyecto en la empresa

Folsom California, May 27, 2019

Commission of Graduation Final Works,

I am the manager for Mauricio Piedra at Intel® Corporation. I would like to express my support for his graduate project titled "Adaptation of the DevOps practice to increase the 'Scrum Velocity' at Intel® Corporation". As our organization makes this transformation to DevOps, Mauricio's project will be valuable to the agile persistent team he is on. We look forward to incorporating his insights and recommendations into the team's transformation. Thank you for your consideration.

Regards,

Matthew Joseph White
Intel® Corporation
Chief Product Owner, Security Incident Response
SAFe®4 Agilist, CISSP

San Jose, August 16, 2019

Commission of Graduation Final Works,

I am the manager for Mauricio Piedra at Intel Corporation. I would like to express that I received Mauricio's graduate project titled "Adaptation of the DevOps practice to increase the 'Scrum Velocity' at Intel Corporation". As our organization makes this transformation to DevOps, Mauricio's project will be valuable to the agile persistent team. We appreciate his effort and recommendations into the team's transformation. Thank you.

Regards,

Sanjeev Rana

IT | Enterprise Applications Development | Product Engineering Solutions Delivery | - Costa Rica
iNet: 8 227 3550

A quien interese:

Yo, Astrid Quirós Granados, Filóloga de la Universidad de Costa Rica; con cédula de identidad 3-438-182, inscrita en el Colegio Licenciados y Profesores, con el carné N° 80791 y en la Asociación Costarricense de Filólogos, con el carné N° 0096, hago constar que he revisado el trabajo. Y he corregido en él, los errores encontrados en redacción, ortografía, gramática y sintaxis. El trabajo se titula:

ADAPTACIÓN DE PRÁCTICAS DEVOPS PARA AUMENTAR EL 'SCRUM VELOCITY' DEL PROYECTO SECURITYOPS EN COMPONENTES INTEL COSTA RICA S.A

MAURICIO PIEDRA HIDALGO

Se extiende la presente certificación a solicitud del interesado, en la ciudad de San José a los catorce días del mes de agosto del dos mil diecinueve. La filóloga no se hace responsable de los cambios que se le introduzcan al trabajo posterior a su revisión.

Teléfono: 8415 95 27 Correo: asqui24@hotmail.es