UNIVERSIDAD TÉCNICA NACIONAL

SEDE CENTRAL

CARRERA DE INGENIERÍA DEL SOFTWARE

Análisis de las tecnologías de desarrollo *cross-platform* existentes en el mercado y propuesta de un proceso de desarrollo automatizado utilizando herramientas DevOps aplicable a una pyme enfocada en el desarrollo de aplicaciones multiplataforma

Trabajo final de graduación como requisito para optar por el grado académico de
LICENTIATURA EN INGENIERIA DEL SOFTWARE

Marni Andrei López López

Alajuela, Costa Rica

Noviembre, 2023

TRIBUNAL EXAMINADOR

Licda. Ana Cecilia Odio Ugalde
Director de carrera

Msc. Michael Jiménez Palacios
Tutor

Msc. Sergio Quesada Espinoza
Lector

Msc. Helberth Roman Garita
Lector

DEDICATORIA

A mis padres, mi hermano y mi esposa por su apoyo incondicional en todos los procesos de mi vida, por siempre estar a mi lado a lo largo de los años, en los momentos difíciles y en los momentos de mayor felicidad.

AGRADECIMIENTOS

A mi directora de carrera, quien siempre ha apoyado a todos sus estudiantes en todos los caminos que estos toman y siempre ha estado dispuesta a escucharnos y mejorar los procesos universitarios para todos nosotros, a todos los profesores de la Universidad Técnica Nacional que se esfuerzan día a día para dar lo mejor y transferir sus conocimientos a todos sus estudiantes, quiero agradecerles, ya que sus palabras fueron sabias, y sus conocimientos rigurosos y precisos. A ustedes, mis profesores queridos, les debo mis conocimientos. Muchas gracias por su paciencia y por compartir con todos nosotros de manera profesional e invaluable su saber.

A mis padres, quienes siempre han sido el motor que impulsa mis sueños y esperanzas y estuvieron a mi lado en los días y las noches más difíciles durante mis horas de estudio. Hoy, al concluir mis estudios, les dedico a ustedes este logro, amados padres, como una montaña más que he logrado conquistar con su apoyo.

Por último, un aprecio general a todas las demás personas que me han acompañado durante esta etapa de mi vida de formación profesional, mi esposa y mis amigos, el apoyo y la guía brindados han de ser un pilar sustentable para el éxito de mis años profesionales.

Gracias de todo corazón a todos ustedes.

DECLARACIÓN JURADA

Yo, **Marni Andrei López López**, mayor, casado, estudiante de la Carrera de Ingeniería del Software, de la Universidad Técnica Nacional, domiciliado en Ciudad Colón, Costa Rica, portador de la cédula de identidad número 1 - 1623 0677, en este acto, debidamente apercibido y entendido de las penas y consecuencias con las que se castiga, en el Código Penal, el delito de perjurio, ante quienes se constituyen en el Tribunal Examinador de mi trabajo de Tesis para optar por el título de licenciatura en Ingeniería del Software, juro solemnemente que mi trabajo de investigación titulado: **"Análisis de las tecnologías de desarrollo *cross-platform* existentes en el mercado y propuesta de un proceso de desarrollo automatizado utilizando herramientas DevOps aplicable a una pyme enfocada en el desarrollo de aplicaciones multiplataforma"** es una obra original que ha respetado todo lo preceptuado por las Leyes Penales así como la Ley de Derechos de Autor y Derechos Conexos, número 6683 del 14 de octubre de 1982 y sus reformas, publicada en la Gaceta número 226 del 25 de noviembre de 1982; incluyendo el numeral 70 de dicha ley que advierte: artículo 70: Es permitido citar a un autor transcribiendo los pasajes pertinentes siempre que estos no sean tantos y seguidos, que puedan considerarse como una reproducción simulada y sustancial, que redunde en perjuicio del autor y de la obra original. Asimismo, estoy advertido que la Universidad Técnica Nacional se reserva el derecho de protocolizar este documento ante Notario Público. En fe de lo anterior firmo en la ciudad de Alajuela, el día 30 del mes de Octubre del año 2023.

MARNI ANDREI LOPEZ LOPEZ (FIRMA)

Digitally signed by MARNI ANDREI LOPEZ LOPEZ (FIRMA)
Date: 2023.10.30 21:32:11 -06'00'

Marni Andrei López López

Cédula de identidad: 1-1623 0677

ACTA DE APROBACION DEL TRIBUNAL

Licenciatura en Ingeniería del Software

Trabajo Final de Graduación

Acta No. 005 - 2023

Acta de la sesión **No. 004**, del día miércoles 22 de noviembre de 2023 a partir de las 18:00 horas, en periodo del tercer cuatrimestre 2023, y en la que el Tribunal Evaluador recibe la sustentación del proyecto de graduación, realizado por el estudiante**: Marni Andrei López López**, portador de la cédula: 1-1623-0677, quien opta por el Grado Académico de Licenciatura en Ingeniería del Software, sita, en la Universidad Técnica Nacional, presentando el trabajo final de graduación con el tema:

"Análisis de las tecnologías de Desarrollo cross-platform existentes en el mercado y propuesta de un proceso de desarrollo automatizado utilizando herramientas DevOps aplicable a una pyme enfocada en el desarrollo de aplicaciones multiplataforma".

Preside el Tribunal la señora Directora de Carrera de Ingeniería del Software, MGt. Ana Cecilia Odio Ugalde, junto con la participación del académico **Maikol Jiménez Palacios,** tutor del trabajo final de graduación, **Sergio Adrián Quesada Espinoza y Helberth Román Garita** lectores del trabajo final de graduación.

La señora presidenta del Tribunal manifiesta que los miembros del mismo leyeron el informe, que los estudiantes acogieron las recomendaciones indicadas y en consecuencia se procede a recibir la sustentación correspondiente, en la que los estudiantes realizan su exposición, sujeto al tiempo establecido. Terminada la misma, se procede a externar comentarios pertinentes al trabajo presentado, se formulan preguntas que fueron respondidas por parte de los estudiantes de manera exitosa.

Concluida la sustentación, el Tribunal solicita a los estudiantes retirarse de la reunión para proceder a la votación secreta. La votación da como resultado: **Aprobado**, **c**on nota de: **9.6**

De nuevo en la reunión, la señora presidenta les comunica el resultado declarando que ya son: Licenciados en Ingeniería del Software, a la vez indica que, conforme a la normativa existente, debe revisar el Reglamento de Trabajos Finales de Graduación de la Universidad Técnica Nacional (disponible en la página web de la UTN), específicamente en el Capítulo

IV, Artículos 38 y 40, donde se indica el procedimiento a seguir para efectuar la entrega de los ejemplares físicos y digitales. Se les recuerda también la obligación de presentarse al ACTO DE GRADUACIÓN, al que serán convocados oportunamente. Se cierra la sesión a las: 19:00 horas del 22 de noviembre de 2023.

MARNI ANDREI LOPEZ LOPEZ (FIRMA)
Digitally signed by MARNI ANDREI LOPEZ LOPEZ (FIRMA)
Date: 2023.11.27 02:36:21 -05'00'

Marni Andrei López López
Estudiante

SERGIO ADRIAN QUESADA ESPINOZA (FIRMA)
PERSONA FISICA, CPF-02-0537-0023.
Fecha declarada: 27/11/2023 10:20:43 AM
Esta es una representación gráfica únicamente, verifique la validez de la firma.

Sergio Adrián Quesada Espinoza
Miembro del Tribunal
Evaluador
Lector

MAIKOL JIMENEZ PALACIOS (FIRMA)
Digitally signed by MAIKOL JIMENEZ PALACIOS (FIRMA)
Date: 2023.11.27 11:36:43 -06'00'

Maikol Jiménez Palacios

Miembro del
Tribunal Evaluador
Tutor

HELBERTH ROMAN GARITA (FIRMA)
Firmado digitalmente por HELBERTH ROMAN GARITA (FIRMA)
Fecha: 2023.11.23 08:15:19 -06'00'

Helberth Román Garita
Miembro del Tribunal
Evaluador
Lector

ANA CECILIA ODIO UGALDE (FIRMA)
PERSONA FISICA, CPF-02-0359-0099.
Fecha declarada: 22/11/2023 07:42:01 PM
Razón: UTN
Lugar: ISW Contacto: aodio@utn.ac.cr

Ana Cecilia Odio Ugalde
Directora de Carrera

CARTA DE AUTORIZACIÓN DEL TUTOR

Alajuela, 4 de agosto de 2023

Sr.

Lic. Ana Cecilia Odio Ugalde

Director de la Carrera de Ingeniería del Software

Universidad Técnica Nacional

Estimado señor director:

El estudiante **Marni Andrei Lopez Lopez**, portador de la cédula de identidad No. 1 - 1623 0677, ha presentado para revisión el Proyecto de Graduación denominado: **"Análisis de las tecnologías de desarrollo Cross-Platform existentes en el mercado y propuesta de un proceso de desarrollo automatizado utilizando herramientas DevOps aplicable a una PYME enfocada en el desarrollo de aplicaciones multiplataforma."** En calidad de Tutor, se ha revisado y corregido todos los aspectos referentes a este documento. Por lo tanto, se hace constar, que se encuentra listo para ser presentado a la Universidad Técnica Nacional, como trabajo de graduación.

Atentamente,

MAIKOL JIMENEZ PALACIOS (FIRMA)

Digitally signed by MAIKOL JIMENEZ PALACIOS (FIRMA)
Date: 2023.08.04 12:58:21 -06'00'

Msc. Michael Jimenez Palacios

Tutor

CARTA DE AUTORIZACIÓN DEL LECTOR

Alajuela, 27 de julio de 2023

Sra.
Mgt. Ana Cecilia Odio Ugalde
Directora de la Carrera de Ingeniería del Software
Universidad Técnica Nacional

Estimada señora Directora:

Sirva la presente para saludarle y hacer de su conocimiento mi aprobación, en calidad de lector, del Proyecto de Graduación realizado por el estudiante Marni Andrei López López, portador de la Cédula de Identidad No. 1 – 1623 0677, titulado: **"Análisis de las tecnologías de desarrollo Cross-Platform existentes en el mercado y propuesta de un proceso de desarrollo automatizado utilizando herramientas DevOps aplicable a una PYME enfocada en el desarrollo de aplicaciones multiplataforma."** Hago constar que se ha revisado y corregido todos los aspectos referentes a este documento; por lo que manifiesto que el mismo se encuentra listo para ser presentado a la Universidad Técnica Nacional, como trabajo final de graduación.

Atentamente,

SERGIO ADRIAN QUESADA ESPINOZA (FIRMA)
PERSONA FISICA, CPF-02-0537-0023.
Fecha declarada: 27/07/2023 03:52:02 PM
Esta es una representación gráfica únicamente,
verifique la validez de la firma.

Msc Sergio Quesada Espinoza
Lector

CARTA DE AUTORIZACIÓN DEL LECTOR

Alajuela, 28 de Julio de 2023

Sr.

Lic. Ana Cecilia Odio Ugalde

Director de la Carrera de Ingeniería del Software

Universidad Técnica Nacional

Estimado señor director:

Sirva la presente para saludarle y hacer de su conocimiento mi aprobación, en calidad de lector, del Proyecto de Graduación realizado por el estudiante **Marni Andrei Lopez Lopez**, portador de la Cédula de Identidad No. 1 1623 0677, titulado: **"Análisis de las tecnologías de desarrollo Cross-Platform existentes en el mercado y propuesta de un proceso de desarrollo automatizado utilizando herramientas DevOps aplicable a una PYME enfocada en el desarrollo de aplicaciones multiplataforma."** Hago constar que se ha revisado y corregido todos los aspectos referentes a este documento; por lo que manifiesto que el mismo se encuentra listo para ser presentado a la Universidad Técnica Nacional, como trabajo de graduación.

Atentamente,

HELBERTH ROMAN GARITA (FIRMA)

Firmado digitalmente por HELBERTH ROMAN GARITA (FIRMA)
Fecha: 2023.07.28 17:03:18 -06'00'

Ing. Helberth Román Garita Msc.

Lector

CARTA DE REVISIÓN FILOLÓGICA

Alajuela, 3 de octubre de 2023

Sra.
Licda. Ana Cecilia Odio Ugalde
Directora de la Carrera de Ingeniería del Software
Universidad Técnica Nacional

Estimada señora directora:

Sirva la presente para saludarle y expresar que en mi calidad de licenciada en Filología Española he revisado la redacción, ortografía y estilo del proyecto de graduación titulado: **"Análisis de las tecnologías de desarrollo *cross-platform* existentes en el mercado y propuesta de un proceso de desarrollo automatizado utilizando herramientas DevOps aplicable a una pyme enfocada en el desarrollo de aplicaciones multiplataforma"**, realizado por el estudiante **Marni Andrei López López**, para optar por el grado académico de Licenciatura en Ingeniería del Software de la Universidad Técnica Nacional, por lo que se puede dar fe del correcto español que este contiene.

Atentamente,

DAHIANA PRISCILA JIMENEZ PICADO (FIRMA)

DAHIANA PRISCILA JIMENEZ PICADO (FIRMA)
2023.10.03 22:46:16 -06'00'

Licda. Dahiana Jiménez Picado
Carné de Colypro n.º 94334
Filóloga

Tabla de contenidos

Índice de tablas

Índice de figuras

CAPÍTULO I

INTRODUCCIÓN

1.1 Introducción

Las aplicaciones móviles no son aplicaciones de escritorio adaptadas para dispositivos con pantallas pequeñas, son, por el contrario, aplicaciones completamente diferentes por su uso y arquitectura con respecto a las aplicaciones de escritorio que los usuarios están acostumbrados a utilizar.

La revolución de los dispositivos para comunicaciones móviles tiene menos de 20 años. De acuerdo con Logan y Scolari (2014), para finales de 2007, una de cada dos personas tenía un teléfono celular. En Europa, la cifra alcanza el 100% de la población. En África hay una de cada cuatro personas, y en Asia, una de cada tres. La alta competitividad y el abaratamiento de los costos ha reducido la barrera digital del teléfono celular en grandes medidas.

Durante este tiempo, la tecnología ha evolucionado desde la voz hasta la información inalámbrica, y el uso de dispositivos móviles se ha convertido en algo cotidiano. Gracias a ello, en la actualidad existe la posibilidad de comunicación con cualquier persona, en cualquier momento y desde casi cualquier lugar. Dentzel (2013) manifiesta que:

> La tendencia de uso de internet de los usuarios ya no es pasar horas conectados delante de un ordenador después de las clases o de trabajar, sino estar conectados en todo momento y en cualquier lugar a través de dispositivos móviles. (párr. 11)

Los orígenes de este desarrollo datan del matemático escocés James Clerk Maxwell, quien formuló, en el año 1860, un par de ecuaciones cuya solución predijo la propagación de las ondas electromagnéticas a la velocidad de la luz. Se necesitaron 20 años para comprobar dicha predicción en un laboratorio y otros 20 años para que se llevara a cabo la primera aplicación móvil. El sitio Itedamza (2021) señala que:

> Buscando ahondar en el problema, Maxwell demostró que sus ecuaciones predecían la existencia de ondas de campos eléctricos/magnéticos oscilantes, que viajaban por el vacío a una velocidad que era posible predecir sobre la base de experimentos eléctricos simples; empleando los medios y datos disponibles en la época, Maxwell obtuvo una velocidad de 310 740 000 metros por segundo. En su artículo “A Dynamical Theory of the Electromagnetic Field”, de 1864, declara que: “Este acuerdo de resultados parece mostrarnos que la luz y el magnetismo son efectos de la misma sustancia, y que la luz es una perturbación electromagnética propagada a través de un campo de acuerdo con las leyes electromagnéticas”. (p. 4)

Actualmente, las nuevas generaciones consideran al Internet como el medio de comunicación más importante, por ello se ha dado el surgimiento o *boom* de las aplicaciones móviles. Todas las empresas han creado su propia aplicación con el objetivo de mantener el interés de los usuarios y las ventas en esta nueva era de la comunicación.

Debido a esto, se han desarrollado una serie de metodologías para el modelado de aplicaciones móviles que apuntan a resolver distintos problemas existentes en el desarrollo de este tipo de *software*. Se debe recordar que, como señala Aguirre (2018):

> Al momento de elegir la tecnología con la que desarrollar una aplicación entran en juego múltiples factores: tiempos del proyecto, costo, capacidad técnica del equipo y las ventajas y desventajas intrínsecas de cada tecnología. Cada uno ejerce presión en una dirección diferente, y es importante encontrar un punto de equilibrio en el que confluyen todos en pos de las necesidades reales del proyecto y las expectativas del cliente. (párr. 1)

Las pequeñas y medianas empresas (pymes) se pueden ver envueltas en un problema, ya que esta elección es la que dicta el resto del proyecto y la calidad del producto final que se presenta al cliente. La elección del entorno de trabajo (*framework*) es el proceso más importante en cuanto a arquitectura del proyecto que se plantea. De acuerdo con Muente (2021):

> El *framework* es una especie de plantilla, un esquema conceptual, que simplifica la elaboración de una tarea, ya que solo es necesario complementarlo de acuerdo con lo que se quiere realizar.
>
> A pesar de que su uso más común es en la informática, este concepto es también utilizado en el *marketing*. (párr. 10-11)

Cabe mencionar que en el mercado existen tecnologías nativas, híbridas y progresivas. Esto agrega una capa adicional de complejidad al proceso de desarrollo móvil, y en el caso de una pyme que desee ingresar en este campo decidir entre una tecnología nativa, híbrida o progresiva.

Una de las mayores confusiones que existen en el momento de desarrollar en móvil es pensar que si no se usa Swift o Java (los lenguajes oficiales de iOS y Android) se trata de un desarrollo híbrido, pero no es así. La mayor diferencia entre una aplicación nativa y una híbrida no es la forma en que se programan, sino cómo se ejecutan para el usuario final. El sitio Platzi (2016) específica que:

> Si requiere algo adicional para ejecutarse como un navegador o un contenedor, son híbridas. Es el caso de Apache Cordova, Ionic y Unity. Si se ejecutan directamente en el OS, son nativas, tal como lo hace ReactNative, NativeScript y Xamarin. (párr. 2)

La existente oferta tecnológica tan extensa, sumada a la ya enorme cantidad de metodologías de administración de proyectos y arquitectura de aplicaciones aplicables al desarrollo móvil, complica el proceso de selección de tecnologías, mucho más para las pymes. El sitio de Lucidchart (2019) señala que:

> Si bien estas metodologías tienen diferencias significativas, es importante reconocer que, en última instancia, cada metodología de gestión de proyectos tiene el mismo objetivo: facilitar la finalización de los proyectos. Con ese fin, cada metodología ayuda a gestionar los procesos de trabajo de su equipo a través de la estructura y la comunicación. Aunque cada

> metodología se implementa diferente, Agile, Waterfall, KANBAN y SCRUM tienen mucho en común. (párr. 5)

A estos conceptos hay que agregar que para comenzar con el desarrollo de aplicaciones móviles es importante conocer sobre la arquitectura de los sistemas operativos, los cuales difieren en gran parte entre los grandes del mercado, Android y iOS. Mientras que Android está formado por varios niveles o capas, iOS no está formado por múltiples capas, lo que hace el proceso de desarrollo más tedioso, ya que, en el momento de acceder a capacidades nativas del sistema operativo, en ocasiones se pierde control sobre el comportamiento de la aplicación.

Es necesario siempre tomar en cuenta el proceso de desarrollo y la administración del proyecto como un todo, ya que estos, aunque normalmente ignorados, son puntos importantes en el proceso. Para esto existen los DevOps, que son considerados una especie o conjunto de prácticas de trabajo con el objetivo principal de automatizar e integrar los procesos entre el desarrollo de *software* y los equipos de TI, para que puedan construir, probar y lanzar *softwares* de manera más rápida y confiable. De acuerdo con el sitio Ambit (2018):

> La principal característica del movimiento DevOps es defender enérgicamente la automatización y el monitoreo en todos los pasos de la construcción del *software*, desde la integración, las pruebas, la liberación hasta la implementación y la administración de la infraestructura. DevOps apunta a ciclos de desarrollo más cortos, mayor frecuencia de

> implementación, lanzamientos más confiables, en estrecha alineación con los objetivos comerciales. (párr. 2)

El término DevOps se formó combinando las palabras “desarrollo” y “operaciones”, y significa un cambio cultural que cierra la brecha entre los equipos de desarrollo y operación, que históricamente funcionaron en silos.

Debido a la naturaleza continua de DevOps, los profesionales utilizan el bucle infinito para mostrar cómo las fases del ciclo de vida de DevOps se relacionan entre sí. A pesar de que parece fluir secuencialmente, el ciclo simboliza la necesidad de colaboración constante y mejora iterativa a lo largo de todo el ciclo de vida.

Se debe recordar que el fin de DevOps es la reducción del tiempo de cada fase del proceso de desarrollo, así que DevOps como cultura tiene la característica principal de defender la automatización de los procesos.

Aunque esta cultura se ha dado desde el nacimiento de DevOps, continúa siendo un tema que no se ha explorado en Latinoamérica de manera amplia y los casos de éxito de referencia encontrados pertenecen en su mayoría a grandes empresas de Europa y Estados Unidos. Esto conlleva a poca documentación sobre métodos, técnicas, estrategias y lineamientos a seguir para adoptar.

En la investigación de Wettinger *et al*. (2016), de acuerdo con Muñoz, Ordóñez y Buchelli (2020), se define a DevOps como un paradigma emergente cuyo objetivo es la estrecha integración de desarrolladores con el personal de operaciones. Esto se ha establecido como una ventaja competitiva crítica para

responder rápidamente a los cambios en la cultura y la organización, y para impulsar de manera constante nuevas herramientas, enfoques y artefactos de código, para así lograr implementar aplicaciones de una manera más eficiente.

Por otra parte, Muñoz, Ordóñez y Buchelli (2020) afirman que, pese al creciente interés y la variedad de definiciones adoptadas por el concepto DevOps en la implementación práctica, se sabe poco sobre la comprensión que tienen los profesionales de tecnologías de la información sobre los caminos que en realidad son exitosos para la correcta adopción de DevOps.

Aunque hay un consenso generalizado en la literatura sobre la existencia de diferencias operacionales entre las grandes y pequeñas empresas, nunca se han llevado a cabo iniciativas para verificarlo y proponer modelos de trabajo especializados en el desarrollo ágil desde el punto de vista de las pequeñas empresas.

Dybå (2003) realizó un estudio para determinar si el tamaño de una organización puede afectar a la estrategia de implementación de un programa de mejora de procesos y el grado de éxito que se alcance. Seleccionó una muestra muy amplia de empresas y de personal de las TIC (tecnologías de la Información y de las comunicaciones) de Noruega. Los hallazgos muestran que las organizaciones pequeñas informaron que implementan los SPI (*software process improvement*) con la misma eficacia que las organizaciones grandes y, a su vez, logran un alto rendimiento organizacional.

Según Wiegers y Stutzenberger (2000), el mayor error en la implementación de programas DevOps en empresas de pequeño y mediano trabajo es causado por la falta o nulo seguimiento que se les da a los planes implementados, lo que se debe, principalmente, a los costos que conlleva realizar su control.

Otro problema añadido para las pymes, mencionado en Calvo, Manzano, Villalón, Feliu, Amescu, Sánchez y Pérez (2002), es que, aunque el retorno de la inversión de un proceso de desarrollo automatizado se produce desde la planificación inicial, este no es notorio hasta un medio largo plazo, lo que representa un desafío considerable para una empresa con un personal y recursos económicos reducidos.

Así pues, la dificultad de aplicar los modelos DevOps más extensos con múltiples pasos y un proceso de implementación considerable a los procedimientos definidos por las pymes se debe principalmente a los costos asociados directamente con su implementación y al considerable tiempo necesario para obtener resultados de estos modelos.

1.2 Justificación

En la actualidad, uno de los aspectos vitales y en crecimiento de los negocios está constituido por las tecnologías de la información, que apoyan a todo tipo de organización mejorando su eficacia y eficiencia en los procesos propios de la actividad, en la toma de decisiones gerencial y en la colaboración entre grupos de trabajo sin importar su situación geográfica.

En el siglo XXI, se ha originado un enorme cambio en la forma de interactuar de las personas y de manejar los negocios, esto, por consiguiente, ha generado un cambio en la manera en que los sistemas de información apoyan la toma de decisiones. Este cambio se ha producido en gran medida debido al *boom* de Internet y las tecnologías móviles, que han impulsado la transición de los procesos y negocios del papel a la web y las aplicaciones móviles.

Es innegable que la forma en que la sociedad humana se relaciona e interactúa ha experimentado cambios significativos, lo cual ha tenido un impacto notable en las actividades de las organizaciones, incluidas las empresas. En este contexto, el auge de dispositivos móviles, como *smartphones* y *tablets*, ha creado nuevas oportunidades tanto para las empresas de desarrollo como para la sociedad en general.

Esta nueva etapa en la historia de la humanidad y las tecnologías de la información se caracteriza por dos grandes líderes de la industria de la movilidad. Por un lado, se sitúa el creador de los famosos iPhone y iPad, la multinacional tecnológica Apple, Inc. que incorporan el sistema operativo iOS; por otro, se encuentra Google con su sistema operativo Android. Cada uno de estos sistemas operativos tiene su propio lenguaje de programación y su propio set de mejores prácticas, lo que ha causado a su vez el surgimiento de una nueva forma de desarrollo llamada plataformas cruzadas o *cross-platform*, cuyo objetivo es englobar el desarrollo de aplicaciones en una sola tecnología con un solo set de mejores prácticas.

Se debe tomar en cuenta que la existencia de distintas arquitecturas y tecnologías ha causado el surgimiento de muchos enfoques distintos que se le pueden dar a los procesos de desarrollo, que se denominan DevOps, una combinación de filosofías, prácticas y herramientas culturales que aumenta la capacidad de una organización, sin importar su tamaño, para entregar aplicaciones y servicios a alta velocidad y gran calidad.

La presente investigación se centra en la entrega continua y la agilidad de los procesos, ya que, al existir tantas tecnologías y metodologías de desarrollo de aplicaciones *cross-platform* en el mercado, una empresa pyme que recién ingrese al mercado de desarrollo podría verse en dificultades con respecto a la elección de la mejor tecnología y el mejor proceso de desarrollo aplicable. Más adelante, en este trabajo de tesis, se examinará esta diversidad tecnológica y se ofrecerán recomendaciones prácticas para guiar a las pymes en su elección de enfoques óptimos para el desarrollo de aplicaciones multiplataforma.

A partir del estudio de trabajos relacionados, se observa que la mayoría de estos se centran en las herramientas y la integración de la colaboración y automatización DevOps como un todo, y a pesar de que las prácticas y herramientas son importantes en el entorno de desarrollo ninguno se interesa por definir un proceso de implementación de esas herramientas, dedicadas a pymes.

El propósito de la investigación es, a partir del análisis de los avances tecnológicos de desarrollo *cross-platform*, así como de las mejores prácticas de desarrollo aplicadas por profesionales del área en el territorio costarricense,

establecer un marco de trabajo de entrega continua, que dependa del menor esfuerzo de los colaboradores para la creación de productos de *software* aplicable a una empresa pyme que se dedique al desarrollo de aplicaciones *in-house* para la venta de estos como SaaS.Objetivos

1.3 Objetivos

1.3.1 Objetivo general

Analizar las tecnologías de desarrollo móvil *cross-platform*, así como las prácticas de desarrollo automatizado existentes en el mercado nacional e internacional, por medio de investigación de campo consultando a profesionales del sector de desarrollo de nivel nacional, para la definición de un proceso de desarrollo y entrega continua de aplicaciones móviles aplicable a una empresa pyme que se dedique al desarrollo de *software*.

1.3.2 Objetivos específicos

- Describir la evolución del sector de desarrollo con tecnologías *cross-platform*, por medio de investigación de la documentación tecnológica para elaborar una base de conocimientos.
- Identificar las prácticas de desarrollo existentes y sus características en las empresas del territorio costarricense por medio de encuestas a profesionales del área, así como identificar las dificultades que experimentaron al implementar estas prácticas en sus proyectos de desarrollo.

- Proponer un marco de trabajo de entrega basado en las mejores prácticas de proyectos de desarrollo de *software* en aplicaciones móviles dirigido a equipos de desarrollo de empresas pyme.

1.4 Problema

Debido al *boom* del Internet y a la constante y creciente adopción de las tecnologías de la información de la cual el mercado nacional se ha visto beneficiado, se ha producido que los usuarios adquieran cada vez más equipos móviles. Esto, a la vez, ha abierto un nuevo mercado de desarrollo: el de las aplicaciones móviles. Al abrirse este mercado, se han creado nuevas tecnologías de desarrollo *cross-platform* por las grandes empresas a nivel mundial; la existencia de estas tecnologías tiene como objetivo principal agilizar el proceso de creación y desarrollo de nuevas aplicaciones móviles.

Considerando que las empresas pymes no siempre disponen del personal necesario para llevar a cabo los procesos mencionados, se debe planificar la implementación de un set de mejores prácticas que agilicen el desarrollo de nuevas versiones de las aplicaciones móviles que la pyme cree para sus clientes o el público en general.

Estas tecnologías pueden ser aplicadas por las micro y pequeñas empresas que deseen entrar en el mercado de la comercialización de aplicaciones móviles. El problema es que no existe un camino claro a seguir en la implementación de estas tecnologías en el lugar de trabajo, que tome en cuenta las limitaciones de personal y recursos monetarios de las pymes. Por lo tanto, se pretende determinar:

¿Cómo las pymes dedicadas al desarrollo de aplicaciones móviles *cross-platform* pueden resolver las limitaciones técnicas y operativas del ámbito costarricense, mediante el aprovechamiento de arquitecturas de *software* y mejores prácticas de automatización de procesos?

1.5 Hipótesis

Dado que el fundamento de esta investigación es la presentación de un marco de trabajo, se entiende que algunas micro y pequeñas empresas pueden encontrarlo un tanto complicado de incorporar o aceptar en su totalidad. Sin embargo, el objetivo principal es buscar mejoras. Por lo tanto, se espera que cualquier empresa pyme en Costa Rica que desee orientar sus esfuerzos de desarrollo hacia el ámbito de las aplicaciones móviles al menos considere las recomendaciones que surjan al concluir la investigación.

Con base en lo anterior, se espera que las pymes que se dediquen a la venta y distribución de *software* aplicativo móvil puedan implementar las prácticas de desarrollo automatizado y las tecnologías recomendadas durante la investigación, y obtengan una mejora en los ámbitos comercial, financiero y tecnológico para con sus clientes.

Una micro o pequeña empresa que se dedica al desarrollo de aplicaciones móviles *cross-platform* podrá resolver sus limitaciones técnicas y operativas mediante el aprovechamiento de arquitecturas de *software* y mejores prácticas de automatización de procesos de desarrollo basados en estándares a nivel internacional.

1.6 Alcances

- Se estudian las tecnologías *cross-platform* existentes en el momento de la realización de la investigación, para asegurar que los resultados y las recomendaciones obtenidas de este trabajo sean acordes con la realidad tecnológica del sector de desarrollo.
- Se impulsa el uso de las tecnologías *cross-platform* en lugar de las nativas como medio de mejora para las micro y pequeñas empresas en sus procesos de desarrollo, al explicar los beneficios que se pueden obtener de estas nuevas tecnologías.
- Se documenta y explica el proceso de creación de una aplicación para dispositivos móviles *cross-platform*, con el objetivo de crear una mentalidad de incursión hacia el mundo digital y los beneficios que puede traer para las empresas.
- Se describen las mejores prácticas de un proceso de desarrollo automatizado aplicable a una empresa pyme, con el objetivo de crear la base para un marco de trabajo enfocado en la mejora continua de los procesos, sin importar el tamaño de la empresa que lo implemente.
- Se delimita un marco de trabajo de desarrollo móvil *cross-platform* para las micro y pequeñas empresas, incluido dentro de la temática del estudio, que brinde soporte a estas empresas en sus proyectos de desarrollo móvil.

1.7 Limitaciones

La cooperación de los propietarios de las micro y pequeñas empresas representa un factor clave, ya que son el contacto primordial y se necesitará de su permiso y ayuda para la recopilación de datos e información comercial y financiera, con el fin de realizar la investigación pertinente. Al respecto, puede existir temor e inseguridad sobre el uso que se le pueda dar a los datos sensibles.

El apoyo por parte de la comunidad de informáticos a nivel nacional es importante para la investigación, ya que algunos instrumentos se aplican por medio de redes sociales a profesionales del área de la informática, con el objetivo de obtener información de la adopción de las tecnologías de desarrollo *cross-platform* en el país, así como sobre sus costos de implementación, los desafíos asociados a su uso y sus ventajas y desventajas.

Por último, cabe mencionar que existe poca información en español sobre el tema en estudio, lo que hace necesario consultar material en inglés.

1.8 Matriz de congruencia

Tabla 1

Matriz de congruencia

Título	Análisis de las tecnologías de desarrollo *cross-platform* existentes en el mercado y propuesta de un proceso de desarrollo automatizado utilizando herramientas DevOps aplicable a una pyme enfocada en el desarrollo de aplicaciones multiplataforma.
Problema	¿De qué manera una micro o pequeña empresa que se dedica al desarrollo de aplicaciones *cross-platform* podrá resolver las limitaciones técnicas y operativas del medio costarricense, mediante el aprovechamiento de arquitecturas de *software* y mejores prácticas de automatización de procesos?
Pregunta general	¿Cómo las pymes dedicadas al desarrollo de aplicaciones móviles *cross-platform* pueden resolver las limitaciones técnicas y operativas del medio costarricense, mediante el aprovechamiento de arquitecturas de *software* y mejores prácticas de automatización de procesos?

Objetivo general	Objetivos específicos	Preguntas específicas
Análisis de las tecnologías de desarrollo *cross-platform* existentes en el mercado y propuesta de un proceso de desarrollo automatizado utilizando herramientas DevOps aplicable a una pyme enfocada en el desarrollo de aplicaciones multiplataforma.	● Describir la evolución del sector de desarrollo con tecnologías *cross-platform*, por medio de investigación de la documentación tecnológica para elaborar una base de conocimientos. ● Identificar las prácticas de desarrollo existentes y sus características en las empresas del territorio costarricense por medio de encuestas a profesionales del área, así como identificar las dificultades que experimentaron al implementar estas prácticas en sus proyectos de desarrollo. ● Proponer un marco de trabajo de entrega y mejora continua basado en las mejores prácticas para proyectos de desarrollo de *software* para aplicaciones móviles para equipos de desarrollo de empresas pyme.	● ¿Qué tecnologías de desarrollo *cross-platform* existen en el mercado actualmente? ● ¿Qué procesos e implicaciones legales se deben tomar en cuenta al momento de desarrollar una aplicación móvil? ● ¿Qué ventajas y desventajas presentan las tecnologías de desarrollo *cross-platform* para una empresa pyme? ● ¿Qué mejores prácticas de desarrollo aplicables al *cross-platform* existen en el mercado actualmente?

		● ¿Qué es y cómo se implementa un proceso de desarrollo automatizado? ● ¿Cuáles son las mejores prácticas de entrega continua disponibles en el mercado actualmente?

Fuente: Elaboración propia.

CAPÍTULO II

MARCO TEÓRICO REFERENCIAL

2.1 Marco teórico

El marco teórico consiste en la recopilación de información proveniente de investigaciones previas y consideraciones teóricas desde donde se sustentará el proyecto de investigación, permitiendo de esta manera la correcta interpretación de los resultados, así como la formulación de una serie de recomendaciones.

El marco teórico, también llamado marco de referencia, es el soporte conceptual de una teoría o de los conceptos teóricos, que se utilizan para el planteamiento del problema de un proyecto o una tesis de investigación (Significados, 2018, párr. 2).

En el siguiente apartado, se procede a definir los diferentes elementos concernientes al tema en desarrollo, con la finalidad de esclarecer las diversas terminologías por emplear y crear una base para que el lector conozca los términos utilizados durante la investigación.

2.2 El mercado de los dispositivos y aplicaciones móviles

Los dispositivos y las aplicaciones móviles han estado en constante evolución desde que aparecieron en el mercado en 1983. Estos medios de comunicación han permitido mantener en movimiento toda la sociedad y se han convertido en algo imprescindible.

Al inicio de su desarrollo, los teléfonos móviles no tuvieron la mayor acogida de parte del consumidor, principalmente por el gran tamaño y peso con que fueron

diseñados, además de que tenían un alto valor económico, por lo que muchas personas no tenían la posibilidad de adquirirlos.

Por lo antes mencionado, los desarrolladores han estado en constante actualización, reduciendo y estilizando el diseño de los teléfonos móviles, así como mejorando sus sistemas operativos.

Como mencionan Ranchal (2014) y Murgo (2019), la evolución de las generaciones de los teléfonos celulares ha sido la siguiente:

- **1G. La primera generación:** Para la década de los años ochenta, estos aparatos representaban una evolución sin precedentes dentro de las comunicaciones móviles, y significaron un gran avance, ya que, a partir de la denominada primera generación, las terminales se volvieron más pequeñas, permitiendo a los usuarios trasladarse con sus equipos de comunicación.
- **2G. La segunda generación:** Para la década de los años noventa, se incorporaron diferentes tecnologías para mejorar las comunicaciones móviles. Se destaca que el cambio de 1G a 2G significó un importante paso en el mundo de la telefonía móvil, pues las comunicaciones lograron alcanzar una calidad destacada.
- **2.5G. La generación de transición:** Con la llegada de la 2.5G, en los dispositivos móviles se incluyeron dos nuevos servicios: en primer lugar, el sistema EMS, que básicamente se trataba de un servicio de mensajería mejorado, que permitía incluir dentro de los mensajes algunas melodías e iconos, lo que evolucionó luego a los SMS. Se incorporó el servicio de MMS

(sistema de mensajería multimedia), mensajes que ofrecían la posibilidad de incluir imágenes, sonidos, texto y videos, utilizando para ello la tecnología GPRS.

- **3G. La tercera generación:** En esta generación se buscó aumentar la capacidad de transmisión y recepción de datos, y apareció la conexión a internet. Se desarrolló el sistema UMTS (*universal mobile telecommunications system*) sucesor de la tecnología GSM.
- **4G. La cuarta generación:** Tecnología utilizada actualmente en la mayoría de los teléfonos celulares, la cual permite realizar cosas inimaginables que antes solamente se podían hacer en una computadora, y en la que los usuarios pueden hacer uso de diferentes aplicaciones que mejoran a través de la conexión de wifi.
- **5G. La quinta generación:** Sistema de conexión de red sin cables, que está disponible desde el 2020 en países como Corea del Sur y también en Europa con un proyecto de 50 millones de euros. Su principal avance radica en la mayor velocidad de transferencia, con velocidades de descarga de hasta 10 Gbps (1,25 GB/s). Se espera que la introducción de esta tecnología en el mundo occidental se produzca a finales de 2022 e inicios de 2023.

Como se ha mencionado, por la evolución de los sistemas con los cuales operan, los teléfonos celulares también han cambiado sus formas y diseños, con el objetivo de brindar una mayor comodidad a los usuarios.

Sánchez (2018) menciona los teléfonos más destacados en la evolución de la telefonía celular:

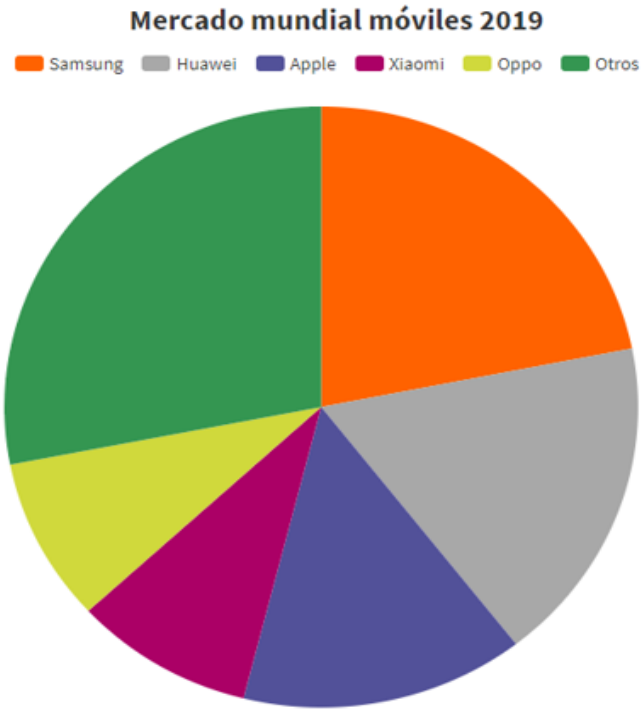
- 1984. Motorola DynaTAC 8000X – Primera generación
- 1996. Nokia 2146
- 2000. Sharp J. sh04 – Segunda generación
- 2004. Motorola RAZR V3
- 2007. iPhone – Tercera generación
- 2008. Blackberry Bold 9000
- 2009. Samsung Galaxy Note – Cuarta generación
- 2017. Samsung – Galaxy S8

El mercado de la telefonía celular no deja de evolucionar: a partir del año 2000 se ha multiplicado pasando de aproximadamente 1000 millones de líneas a 7000 millones, según datos de The World Bank mencionado en sitio web En Naranja.ING (2020).

Ese crecimiento del mercado de móviles ha impactado las economías a nivel mundial, como se menciona en el sitio web En Naranja.ING (2020): “Según datos de GSMA, en 2018 supuso nada más y nada menos que el 4,6 % del PIB mundial (3,9 billones de dólares), y se estima que alcanzará los 4,8 billones de dólares en 2023”. Dicho crecimiento se muestra en la siguiente figura:

Figura 1

Mercado mundial móviles 2019



Fuente: Naranja ING (2020).

2.3 Sistemas operativos para dispositivos móviles

Los sistemas operativos con los cuales funcionan los móviles se han convertido en algo bastante importante en materia de comunicación, en donde los desarrolladores de los sistemas se concentran en la demanda de aplicaciones de parte de los usuarios.

Como sistema operativo se puede señalar la definición de Pedrozo (2012):

> Un sistema operativo móvil o SO móvil es un sistema operativo que controla un dispositivo móvil al igual que las computadoras más grandes utilizan Windows, Linux o Mac OS entre otros. Sin embargo, los sistemas operativos móviles son mucho más simples y están más orientados a la conectividad inalámbrica, los formatos multimedia para móviles y las diferentes maneras de introducir información en ellos. (p. 3)

Como señala Pedrozo (2012), las compañías desarrolladoras de sistemas operativos han evolucionado significativamente “desde los inicios en los años 90 con las versiones de EPOC32 para PDA’s hasta los más avanzados y sofisticados como Android, IOS, BlackBerry” (p. 3).

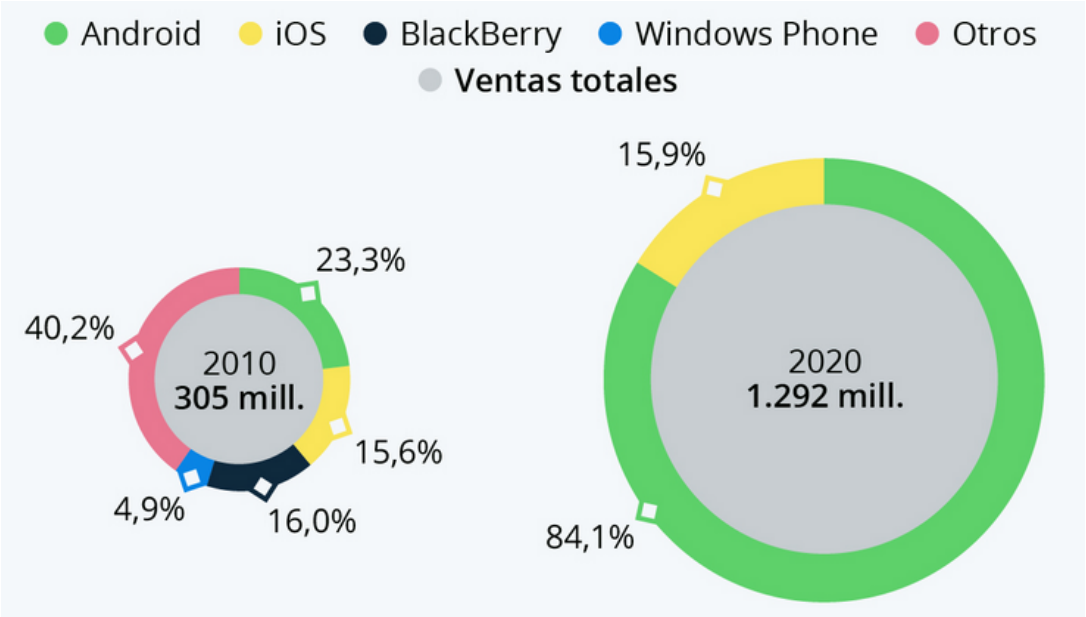
Al respecto, se pueden mencionar algunos datos comparativos de ventas totales entre los años 2010 a 2020 como señala Mena (2021):

> … los dispositivos Android representaron algo más del 84% de las unidades enviadas en 2020, y los iOS de Apple casi el 16% restante. En cambio, en 2010, hace poco más de una década, la cuota de mercado combinada de Android e iOS era inferior al 40%, con BlackBerry, Windows Phone y otros sistemas operativos compartiendo el resto del mercado. (párr. 3)

En la siguiente figura se muestra la cuota de mercado de los sistemas operativos móviles:

Figura 2

Cuota de mercado de smartphones por sistema operativo (por unidad), en el año 2021



Fuente: Mena (2021).

Como funciones básicas de los sistemas operativos se pueden mencionar tres:

1. Ofrecen una interfaz de usuario a través de una CLI o una interfaz gráfica de usuario.
2. Lanzan y gestionan la ejecución de la aplicación.
3. Identifican y exponen los recursos de *hardware* del sistema a esas aplicaciones, por lo general, a través de una API estandarizada.

De los sistemas operativos para móviles, los más utilizados por los desarrolladores son el Android y el iOS, por esta razón, enfocaremos nuestros esfuerzos de investigación en estos.

2.4 Android vs. iOS

En la literatura se puede encontrar una serie de opiniones respecto a las diferencias entre los dos sistemas operativos más utilizados por los desarrolladores de *smartphones*, por lo que se presenta a continuación un cuadro resumen de las principales diferencias entre iOS y Android.

Tabla 2

Cuadro comparativo, Android vs. iOS

Aspecto	iOS	Android
Programación	OS X, UNIX (Código cerrado)	Linux (Código abierto)
Personalización	Las opciones de personalización son limitadas. Sin embargo, una de las ventajas de iOS 14 es que incluye nuevos *widgets*.	Puedes personalizar casi todo, desde la interfaz de usuario, la pantalla de inicio, los *widgets* e incluso aplicaciones de terceros.

Actualizaciones	Apple es el fabricante tanto del *software* como del *hardware*, por eso, es más sencillo desarrollar nuevas actualizaciones y parches de seguridad.	Al estar disponible para muchos fabricantes, las actualizaciones dependen en gran medida de los modelos que saquen al mercado.
Marcas	Solo en iPhone y otros dispositivos móviles de Apple.	Diversos fabricantes como Google, Samsung, Huawei, Xiaomi, Motorola, entre otros
Precio	A pesar del esfuerzo de Apple para entrar a la gama media, los precios siguen siendo elevados.	Es posible conseguir celulares con Android en todas las gamas y precios.
Seguridad	La vigilancia de Apple sobre las aplicaciones desarrolladas da una garantía extra de seguridad y privacidad.	Al estar desarrollado en código abierto, puede que algunas aplicaciones representen un riesgo de entrada de *malware*, sobre todo si son descargadas fuera de la Play Store.

Privacidad	Apple recopila datos de forma anónima. Sin embargo, puedes deshabilitar el seguimiento en los ajustes.	Google también recopila datos cifrados y tiene la opción para deshabilitarlo.
Asistente virtual	Los dispositivos iPhone cuentan con Siri.	En los dispositivos Android se tiene la asistencia de Google Now.

Fuente: Elaboración propia.

Otras diferencias encontradas entre los sistemas Android y iOS se mencionan en el sitio web Quality devs (2020):

- Con Android se puede descargar cualquier tipo de archivo.
- Apple es mucho más restrictivo a la hora de las descargas.
- Cada sistema operativo tiene un sistema de voz propio, en el caso de Android es el asistente de voz de Google y el de iOS es Siri.
- Android vincula y gestiona todo con Google.
- iOs puede gestionar y vincular todas las aplicaciones con Google, pero lo más recomendable es hacerlo con sus aplicaciones nativas.
- Quienes utilizan todo el entorno Apple no suelen tener problemas con la seguridad, ya que iOs utiliza un entorno cerrado precisamente para evitar fallos de seguridad.

En cuanto a precios y ventas de los dispositivos móviles, Android está a la cabeza superando las ventas de *smartphones* Apple. En el sitio web RedNew (2020), se mencionan los siguientes datos:

- En el tope de la lista está el **iPhone XR**, con 3% de ventas. Pero, en el quinto lugar aparece el iPhone 11, con 1,6%.
- Los **Samsung Galaxy A10** y **A50** ocupan el segundo y el tercer lugar, con 2,6% y 1,9% de participación, respectivamente. Mientras, el **Galaxy A20** aparece en la sexta plaza (1,4%).
- **Oppo** colocó tres modelos en este top ten. El A9 de esta marca china está en la cuarta posición, superando ligeramente en ventas al iPhone 11. El A5s está en el sexto puesto (1,5%), y el A5, en el octavo (1,3%).
- Por fin, en noveno y décimo lugar están el **Redmi 7A** de **Xiaomi** (1,2%) y P 30 de **Huawei** (1,1%). (párrs. 11-14)

En cuanto a marcas, la cuota de mercado se reparte como menciona Alcántara (2021): Samsung fue la marca que más móviles vendió (19%)**,** seguida de Apple (15%), Huawei (14%), Xiaomi (11%), OPPO (8%) y Vivo (8%).

2.4.1 Ventajas y desventajas de Android y iOS

En el apartado anterior se han mencionado las principales características de los dos sistemas operativos mayormente utilizados en *smartphones*, de los cuales el usuario elige según sus necesidades y preferencias.

A continuación, se presentan las ventajas y desventajas de cada uno de esos sistemas operativos, según García (2021):

Tabla 3

Ventajas y desventajas del sistema operativo Android

Ventajas	Desventajas
Código abierto	Baja calidad de aplicaciones
Gran variedad de aplicaciones gratis	Mayor riesgo de vulnerabilidad
Diversidad y versatilidad	Complejidad en configuración avanzada
Sistema multitarea	Posibles problemas de sincronización
Buena usabilidad y opciones de personalización	Disponibilidad de aplicaciones exclusivas
Buen rendimiento de la batería	
Mapas de ubicación	

Fuente: García (2021).

Tabla 4

Ventajas y desventajas del sistema operativo iOS

Ventajas	Desventajas
Mejor rendimiento	Pocas opciones de personalización
Mayor seguridad	Precio alto
Sincronización en iCloud	Bajo rendimiento de la batería
Excelente usabilidad	Almacenamiento externo
Actualizaciones de *software* constantes	

Fuente: García (2021).

2.5 Desarrollo *cross-platform*

Debido al constante desarrollo de las plataformas y sistemas operativos utilizados en la fabricación de *smartphones*, es necesario abarcar un mayor número de ellas. Por esta razón, se han creado nuevas aplicaciones que pueden ejecutarse en varias de las plataformas predefinidas.

El *cross-platform* o multiplataformas, como también se le conoce, se define como:

> *Cross-platform* es una metodología de desarrollo de aplicaciones para dispositivos móviles, con un coste y tiempo menor que un desarrollo nativo gracias a la capacidad de poder exportar el código fuente a las diferentes

> plataformas del mercado. A pesar de no garantizar las mismas prestaciones a nivel visual y en concretas funcionalidades que las desarrolladas en el lenguaje nativo, permite ahorrarnos el desarrollo por cada plataforma. Idóneas para el *testing* de ideas o MPV en el mercado. (Moberest, 2015, párr. 3)

Estas multiplataformas se caracterizan por ser creadas bajo un único lenguaje de programación, que facilita al desarrollador una mejor exportación y visualización de los datos desde cualquier dispositivo independientemente del sistema operativo que se esté utilizando.

Hay dos maneras o formas de desarrollar aplicaciones multiplataforma o *cross-platform*, como menciona sitio web ABAMOBILE (2020):

- Por un lado, a través del uso de un lenguaje de desarrollo web como puede ser HTML5, CSS o JavaScript. De esta forma se estaría desarrollando una aplicación como si fuera una web con capacidad para adaptarse a cualquier dispositivo.
- Por otro lado, la creación de aplicaciones móviles multiplataforma también se puede realizar con herramientas de *rendering* a nativo. En este caso, existen herramientas como Flutter o React Native que son *frameworks* que generan código nativo para cada sistema operativo. Esto hace que la experiencia de usuario sea igual que una *app* nativa. (párrs. 9-10)

Según las características de la aplicación que se quiera desarrollar, se pueden presentar una serie de alternativas, como indica Santana (2020):

- Un alto nivel de procesado de datos y uso intensivo de memoria, así como un óptimo ajuste de la UX (*user experience design*) a los estándares de cada plataforma son factores que siguen favoreciendo el desarrollo nativo como solución adoptada.
- La integración del *hardware* de los dispositivos móviles en la aplicación es limitada en soluciones no nativas, por lo que será necesario tener en cuenta las necesidades de la aplicación en este ámbito, para tomar una decisión.
- La reducción del tiempo de desarrollo cuando se quiere tener presencia en más de una plataforma y, por tanto, la reducción también del *time to market*, es el principal punto fuerte de las soluciones web y las híbridas.
- Las soluciones web se imponen en cuanto a velocidad de despliegue, ya que no necesitan pasar por los procesos de validación para entrar en las *stores* oficiales de cada plataforma, pero también debe tenerse en cuenta la confianza que puede producir en el cliente final el hecho de haber superado estas validaciones.
- Las distintas plataformas en las que se quiere tener presencia es también un factor por tener en cuenta, ya que no todas las herramientas de desarrollo *cross-platform* soportan todas las plataformas. Sin embargo, cuanto mayor número de plataformas se quiera abarcar, mayor sería el beneficio, a nivel de desarrollo, ofrecido por una solución *cross-platform*.

2.5.1 Ventajas y desventajas del desarrollo de aplicaciones *cross-platform* o multiaplicaciones

Cada vez que se desarrolla una aplicación es recomendable realizar una revisión del mercado, visualizando las ventajas y desventajas que tiene en función de las necesidades y objetivos individuales.

A continuación, se presentan una serie de ventajas y desventajas de las aplicaciones móviles mencionadas en sitio web ABAMOBILE (2020):

- Ventajas de las *apps* multiplataforma
  - La principal ventaja es que son compatibles con todo tipo de dispositivos y cualquier sistema operativo. Da igual que se visualice en un *smartphone*, *tablet* o PC, o que sea para sistemas Android o iOS. Las aplicaciones multiplataforma se adaptan a todo sin necesidad de crear diferentes *apps*.
  - Al ser desarrolladas bajo un mismo lenguaje, el ahorro de tiempo, de costes y de recursos es una realidad, no disminuyendo con ello la calidad. Las *apps* multiplataforma tienen un gran rendimiento que no tiene nada que envidiar a las aplicaciones nativas.
  - No requieren el uso del navegador, ya que se pueden descargar y crear así un acceso directo desde el dispositivo; estas *apps* tienen una integración completa tanto con el *hardware* como con el *software* de cada dispositivo.

- Al utilizar lenguajes de programación tan conocidos como HTML5 o herramientas como React Native hacen que encontrar profesionales con conocimientos de estos códigos sea más sencillo. Además, son lenguajes muy utilizados lo que hace que la experiencia y el conocimiento de ellos sea mayor.
- La filosofía Mobile First es una de las claves en el desarrollo de *apps* multiplataforma. El hecho de que se puedan adaptar a todo tipo de dispositivo, en especial a los móviles, hace que sean una herramienta mucho más atractiva para los clientes. En la era móvil en la que vivimos, es esencial que cualquier tipo de contenido se pueda visualizar desde un teléfono.

- Desventajas de las aplicaciones multiplataforma
  - A pesar de que las *apps* multiplataforma se adaptan a todos los dispositivos, las pruebas y los test no pueden faltar nunca. Por eso, se debe comprobar que la *app* se visualiza correctamente. Además, cuando se realice cualquier cambio se tiene que volver a comprobar que todo funciona bien en el resto de dispositivos.
  - Aunque las aplicaciones sean multiplataformas, hay que generar una APK (Android) e IPA (iOS) para que funcione bien, por lo que los SDK (*software development kit*) para cada plataforma serán necesarios.

2.5.2 Tipos de herramientas para desarrollo de *cross-platform* o multiplataforma

Existe una variedad de herramientas para el desarrollo de aplicaciones móviles personalizadas, y los desarrolladores encuentran en la multiplataforma la mejor opción para competir en el mercado.

Las personas desarrolladoras de aplicaciones buscan tener presencia en el mercado, pero a menudo se enfrentan a la necesidad de realizar inversiones monetarias significativas. Por lo tanto, encuentran en las plataformas multiplataforma la opción más económica para desarrollar diversas aplicaciones nativas, lo que ha llevado al desarrollo de varias plataformas cruzadas alternativas.

De acuerdo con Ruchir (2020), algunas de las herramientas para el desarrollo móvil multiplataforma son las siguientes:

- **React native:** Permite crear fácilmente aplicaciones nativas y utiliza JavaScript como lenguaje de programación principal para desarrollar aplicaciones. El lado más importante de esto es que puede escribir módulos en diferentes lenguajes como C, Java y Swift. Esta herramienta en particular puede trabajar fácilmente en el procesamiento de video y la edición de imágenes, lo que en realidad no es posible con diferentes marcos de API.
- **Xamarin:** Lanzado por Microsoft, permite desarrollar aplicaciones para varias plataformas como Windows, iOS y Android utilizando un solo código .NET. Esta herramienta multiplataforma es construida como aplicaciones nativas y esto en realidad aparece debido al uso de las interfaces nativas que

funcionan de la misma manera que el usuario desea. Xamarin trabaja en un solo código mediante su identificación y acelera el proceso para todo el desarrollo de aplicaciones móviles multiplataforma.

- **Ionic:** Se encuentra entre los SDK de HTML5 más potentes, que generalmente le permite desarrollar fácilmente aplicaciones móviles con una sensación nativa mediante el uso de tecnologías avanzadas como JavaScript, HTML y CSS. Este SDK en particular se centra en el aspecto y en la interacción de la interfaz de usuario de la aplicación.
- **Flutter:** Es un SDK o kit de desarrollo de *software* que permite al desarrollador crear aplicaciones de alto rendimiento en diferentes plataformas como web, Android, iOS y escritorio desde una única base de código. Además, Google ha creado debidamente este SDK de interfaz de usuario de código abierto en particular.

2.6 Procesos de creación de aplicaciones

2.6.1 Metodologías de administración de proyectos de desarrollo de *software*

Para la ejecución de un proyecto se precisa de una serie de elementos que se deben cumplir de forma óptima. En el caso de la administración para el desarrollo de *software*, es preciso contar con una metodología. Según Maida y Pacienzia (2015):

> La metodología para el desarrollo de *software* es un modo sistemático de realizar, gestionar y administrar un proyecto para llevarlo a cabo con altas posibilidades de éxito. Una metodología para el desarrollo de *software*

> comprende los procesos a seguir sistemáticamente para idear, implementar y mantener un producto *software* desde que surge la necesidad del producto hasta que cumplimos el objetivo por el cual fue creado. (p. 12)

Por esa razón, se precisa aplicar de forma ordenada y sistemática la ejecución de un proyecto, ya que se debe manejar una secuencia para asegurar la correcta funcionalidad de un *software* (Maida y Pacienzia, 2015). En la aplicación de la ingeniería del *software* podemos destacar por medio de la metodología:

- Optimización del proceso y el producto *software*.
- En la planificación y el desarrollo del *software* se guía por métodos.
- Definición de las interrogantes: qué hacer, cómo y cuándo durante todo el desarrollo y mantenimiento de un proyecto.

2.6.1.1 Metodologías tradicionales

2.6.1.1.1 Waterfall

Se aplica en la ingeniería de *Software* y se denomina *waterfall* (cascada) debido a la posición de las fases, que por su ubicación se parece a una cascada “por gravedad” hacia las siguientes fases. Para Maida y Pacienzia (2015):

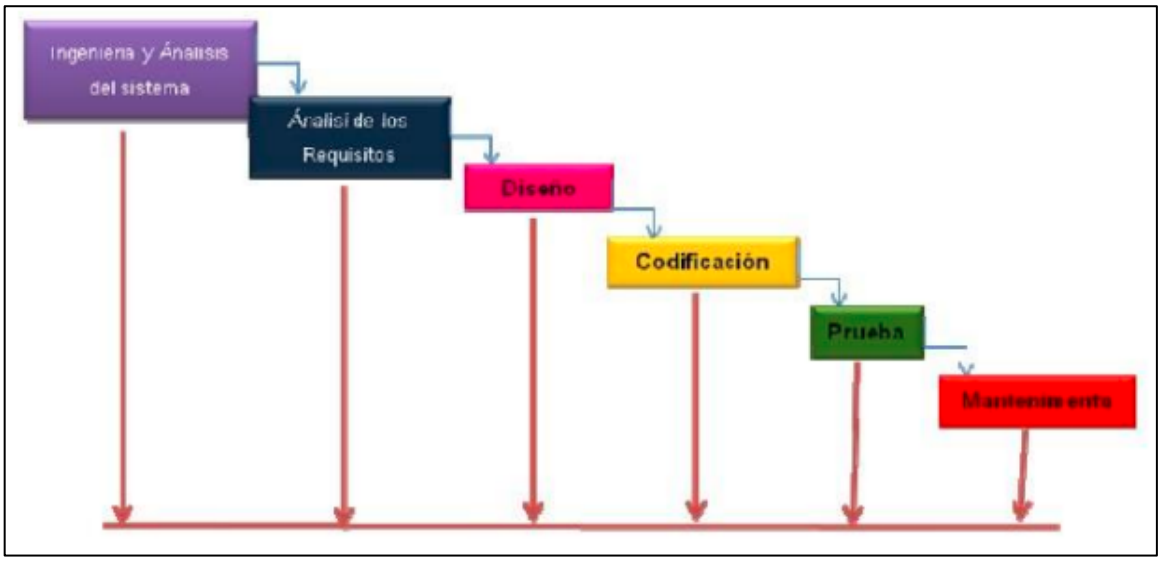
> Es el enfoque metodológico que ordena rigurosamente las etapas del proceso para el desarrollo de *software*, de tal forma que el inicio de cada etapa debe esperar a la finalización de la etapa anterior. Al final de cada etapa, el modelo está diseñado para llevar a cabo una revisión final, que se

encarga de determinar si el proyecto está listo para avanzar a la siguiente fase. (p. 39)

Cada fase debe finalizar su proceso de forma independiente para transferirse a la siguiente; se puede afirmar que el progreso de una fase depende de la efectividad de la anterior.

Figura 3

Modelo del ciclo de vida en cascada (waterfall)



Fuente: Maida y Pacienzia (2015).

Según se muestra en la figura 1, cada fase cumple una funcionalidad para permitir la ejecución de la fase continua. Maida y Pacienzia (2015) detallan al respecto:

Análisis de los requisitos del *software*: El proceso de recopilación de los requisitos se centra e intensifica especialmente en el *software*. El ingeniero

de *software* (analista) debe comprender el ámbito de la información del *software*, así como la función, el rendimiento y las interfaces requeridas.

Diseño: El diseño del *software* se enfoca en cuatro atributos distintos del programa: la estructura de los datos, la arquitectura del *software*, el detalle procedimental y la caracterización de la interfaz. El proceso de diseño traduce los requisitos en una representación del *software* con la calidad requerida antes de que comience la codificación.

Codificación: El diseño debe traducirse en una forma legible para la máquina. El paso de codificación realiza esta tarea. Si el diseño se elabora de manera detallada, la codificación puede darse mecánicamente.

Prueba: Una vez que se ha generado el código comienza la prueba del programa. La prueba se centra en la lógica interna del *software* y en las funciones externas, realizando pruebas que aseguren que la entrada definida produce los resultados que realmente se requieren.

Mantenimiento: El *software* sufrirá cambios después de que se entrega al cliente. Los cambios ocurrirán debido a que se han encontrado errores, a que el *software* debe adaptarse a cambios del entorno externo (sistema operativo o dispositivos periféricos), o a que el cliente requiere ampliaciones funcionales o del rendimiento. (p. 40)

Cada fase es necesaria en el desarrollo de un *software*, y la organización es descendiente de lo general a lo específico.

2.6.1.2 Metodologías ágiles

Su implementación está basada en la ejecución de un proyecto sin entorpecer la parte presupuestaria, con el objetivo de cumplir con las expectativas de los clientes.

Luján (2021) menciona que “las metodologías ágiles se definen como un conjunto de tareas y procedimientos dirigidos a la gestión de proyectos; permiten adaptar la forma de trabajo a las condiciones del proyecto”. Cuando se precisa desarrollar un proyecto, estas metodologías subsanan todos los aspectos que requieren un cambio repentino en cuanto a su aplicación. Asimismo, Maida y Pacienzia (2015) mencionan:

En el "Manifiesto ágil" se definen los cuatro valores principales por los que se deberían guiar las metodologías ágiles.

1. Al individuo y sus interacciones más que al proceso y las herramientas.
2. Desarrollar *software* que funciona más que obtener una documentación exhaustiva.
3. La colaboración con el cliente más que la negociación de un contrato.
4. Responder a los cambios más que seguir una planificación. (p. 51)

Los valores mencionados aseguran el éxito del proyecto una vez finalizado, debido a que adaptarse a cualquier cambio permite la corrección de un riesgo a futuro que limite las necesidades y la funcionalidad del proceso planificado.

2.6.1.2.1 SCRUM

El desarrollo de SCRUM se realiza por medio de trabajo en equipo, y su aplicación se basa en la productividad que se genere durante la ejecución del proceso. Maida y Pacienzia (2015) mencionan que:

> SCRUM es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos. (p. 73)

Por su parte, Luján (2021) afirma que SCRUM define tres roles:

> **SCRUM master:** Lidera el equipo asegurándose de que cumpla las reglas y procesos. Es el encargado de garantizar que el equipo adopte las teorías, prácticas y reglas de la metodología SCRUM.
>
> **Dueño del producto:** Representante de los accionistas y clientes que usan el *software*. Es el responsable de transmitir al equipo de desarrollo la visión del producto que se desea crear; aporta la perspectiva de negocio.
>
> **Equipo de desarrollo:** Grupo de profesionales encargados de convertir la lista de requerimiento en funcionalidades del *software*. Mantiene una organización horizontal en la que cada miembro del equipo se autogestiona y se organiza libremente en la definición y ejecución de los distintos *sprints.* (p. 6)

La ejecución de un rol permite establecer responsabilidades dentro de un grupo de trabajo o delega funciones a cada miembro del equipo dando oportunidad a la innovación en algunos casos.

2.6.1.2.2 KANBAN

KANBAN realiza una serie de aportes que benefician el desarrollo de un proyecto. De acuerdo con Maida y Pacienzia (2015):

> El aporte principal de KANBAN a las metodologías ágiles es que, a través de la implementación de tarjetas visuales, proporciona transparencia al proceso, ya que su flujo de trabajo expone los cuellos de botella, colas, variabilidad y desperdicios a lo largo del tiempo y todas las cosas que impactan al rendimiento de la organización en términos de la cantidad de trabajo entregado y el ciclo de tiempo requerido para entregarlo. Proporciona a los miembros del equipo y a las partes interesadas visibilidad sobre los efectos de sus acciones o falta de acción. De esta forma, cambia el comportamiento y motiva a una mayor colaboración en el trabajo. La visibilidad de los cuellos de botella, desperdicios y variabilidades y su impacto también promueve la discusión sobre las posibles mejoras, y hace que los equipos comiencen rápidamente a implementar mejoras en su proceso. (p. 91)

Dentro de un proyecto se trabaja esperando un progreso en secuencia o con dificultades menores. Cada miembro del equipo de trabajo debe estar dispuesto a presentar cualidades de mejoras o apoyar a quien lo requiera para evitar problemas que caractericen de forma negativa el desarrollo del proyecto.

2.6.1.2.3 XP-Extreme Manufacturing

Según Meléndez, Gaitán y Pérez (2016), XP-Extreme Manufacturing “es una metodología ligera de desarrollo de aplicaciones que se basa en la simplicidad, la comunicación y la realimentación del código desarrollado”. Es una metodología basada en prueba y error para obtener un *software* que realmente funcione; entre sus características, los autores destacan:

- Fundamentada en principios.
- Está orientada hacia quien produce y usa *software* (el cliente participa muy activamente).
- Reduce el costo del cambio en todas las etapas del ciclo de vida del sistema.
- Combina las que han demostrado ser las mejores prácticas para desarrollar *software*, y las lleva al extremo.
- Cliente bien definido.
- Los requisitos pueden cambiar.
- Grupo pequeño y muy integrado (2-12 personas).
- Equipo con formación elevada y capacidad de aprender. (p. 26)

Tabla 5

Comparativo entre metodologías de administración de proyectos

Metodologías de gestión de proyectos tradicionales	Metodologías de gestión de proyectos ágiles
Desarrollo en estilo “cascada” o en línea recta.	Desarrollo en estilo “cíclico” o repetitivo.
El proyecto se planifica y se levantan requerimientos, lo cual toma tiempo; estos requerimientos se aprueban y se da inicio.	Se planifica la visión general del proyecto, así como el rumbo que este debe tomar.
Se crea una línea de tiempo del proyecto definida de inicio a fin.	Se comienza el proyecto sin una línea de tiempo en firme; los requerimientos base se convierten en “historias” y de ahí se comienza a trabajar.
Se comienza el proyecto y se recibe retroalimentación hasta el final de este.	El proyecto recibe constante retroalimentación de los usuarios finales, lo que le permite cambiar los requerimientos con el paso del tiempo.
Al haber acabado el proyecto, se cierra y se entrega con respecto a los requerimientos recibidos al inicio.	El proyecto se cierra cuando los requerimientos iniciales son cumplidos, pero no necesariamente se cierra, ya que los requerimientos cambian con el paso del tiempo.

Fuente: Elaboración propia con información tomada de nextop.es.

Las metodologías buscan la obtención de beneficios a partir de su aplicación y según la finalidad del proyecto. Al inicio de un proyecto, es importante definir el tipo de metodología que se aplicará para evitar riesgos de algún tipo posteriores al inicio del proceso. La tabla 5 muestra una comparación entre las metodologías de desarrollo de proyectos generadas según su funcionalidad, las cuales no presentan similitudes en los distintos puntos del proceso.

2.6.2 Herramientas de administración de proyectos de desarrollo

Se utilizan para dar seguimiento al proyecto por medio de una serie de actividades que delimiten el avance del proceso. Según Guzmán (2019):

> Una herramienta de gestión de proyectos es un *software* con el que se puede realizar un seguimiento a todas las actividades, tareas, recursos, entre otros, de uno o varios proyectos, tanto de manera estratégica como operativa. El objetivo es que se lleve a cabo una gestión exitosa de proyectos. (p. 1)

Las herramientas de gestión de proyectos son importantes porque gestionan la utilización de los recursos y aseguran los tiempos de entrega pactados, o bien mejoran los tiempos de entrega, considerando que el proceso se realice de manera óptima y efectiva.

2.6.2.1 Azure DevOps

Se utiliza para asignar la colaboración conjunta de personas, procesos y tecnología para ofrecer valor de manera constante a los clientes. Para Microsoft (2021):

> Permite que los roles que antes estaban aislados (desarrollo, operaciones de TI, ingeniería de la calidad y seguridad) se coordinen y colaboren para producir productos mejores y más confiables. Al adoptar una cultura de DevOps junto con prácticas y herramientas de DevOps, los equipos adquieren la capacidad de responder mejor a las necesidades de los clientes, aumentar la confianza en las aplicaciones que crean y alcanzar los objetivos empresariales en menos tiempo. (p. 1)

Es por ello por lo que su utilidad resulta importante, ya que conduce a los miembros del equipo bajo una misma línea que permite un seguimiento continuo y en tiempo real de cada ajuste, cambio, problema, entre otros.

2.6.2.2 Jira

En la búsqueda de *softwares* que permitan la agilidad en los trabajos que se desarrollan en grupo, según Atlassian (2021):

> Jira Software forma parte de una gama de productos diseñados para ayudar a equipos de todo tipo a gestionar el trabajo. En principio, Jira se diseñó como un gestor de incidencias y errores. Sin embargo, se ha convertido en una potente herramienta de gestión de trabajo para todo tipo de casos de uso, desde la gestión de requisitos y casos de prueba hasta el desarrollo de *software* ágil. (párr. 1)

Se intenta generar un ambiente de trabajo con variedad de complementos que permitan conocer el control de calidad del proyecto por parte de sus integrantes y, en algunos casos, del cliente final.

Tabla 6

Comparativo de funcionalidades entre herramientas de administración de proyectos

Azure DevOps	**Jira**
Tableros SCRUM	Tableros SCRUM y KANBAN
Administración de pruebas	Administración de pruebas y hojas de ruta del proyecto
Control de versiones nativas en Azure Repos (costo adicional)	Integración con gitlab, Github y SVN para control de versiones
Automatización de procesos	Integración con Gitlab, Github y SVN
Administración de configuración de implementaciones en Azure Pipelines (costo adicional)	
Integración con suite Office (costo adicional)	*Plug-ins* para agregar funcionalidad con herramientas de comunicación como Slack y GitHub desktop
Aplicación como servicio	Aplicación como servicio y hospedaje propio

Creación de informes	Creación de informes
Precio de $6 por usuario	Versión comunitaria para hospedar localmente y cuenta gratuita para 10 usuarios, después $7,50 por usuario

Fuente: Elaboración propia con información tomada de Microsoft y Atlassian.

2.7 Marcos de trabajo DevOps

2.7.1 ¿Qué es DevOps?

Las empresas actualmente enfocan parte de sus recursos para crear opciones rápidas y confiables a sus clientes a través de la tecnología. Netapp (2021) menciona que:

> Esta relación estrecha entre «Dev» y «Ops» se extiende a cada una de las fases del ciclo de vida de DevOps: desde la planificación inicial del *software* a las fases de codificación, compilación, pruebas y publicación, y en la puesta en marcha, las operaciones y la supervisión continua. Esta relación impulsa un bucle de retroalimentación continua con los clientes sobre las mejoras, el desarrollo, las pruebas y la puesta en marcha. Uno de los resultados de todos estos esfuerzos puede ser la publicación continua y más rápida de las adiciones y los cambios que se necesitan en las funciones. (párr. 2)

Es decir, DevOps incluye todos los puntos a trabajar dentro del desarrollo de un proyecto determinado. Asimismo, los DevOps en las empresas pymes juegan un papel importante. Al respecto, Netapp (2021) menciona las siguientes ventajas:

- Una mejor y más rápida entrega de productos.
- Resolución de problemas en menos tiempo y con menor complejidad.
- Mejor escalabilidad y disponibilidad.
- Entornos de funcionamiento más estables.
- Mejor utilización de los recursos.
- Mayor automatización.
- Mayor visibilidad de resultados del sistema.
- Mayor innovación. (p. 2)

Para las empresas pymes la práctica de DevOps proporciona que cada empresa ofrezca un servicio con mejoras constantes, en menos tiempo, con mayor calidad y seguridad para sus clientes finales.

2.7.2 ¿Cuál es la importancia de los DevOps?

En las empresas se busca de forma continua las mejoras en cuanto al desarrollo del trabajo con la finalidad de llegar a la meta. Al respecto, Amazon (2021) menciona que:

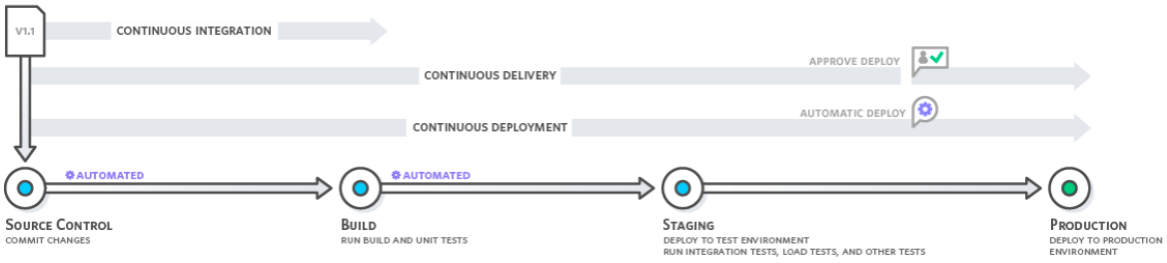
> La integración continua es una práctica de desarrollo de *software* mediante la cual los desarrolladores combinan los cambios en el código en un repositorio central de forma periódica, tras lo cual se ejecutan versiones y

pruebas automáticas. La integración continua se refiere en su mayoría a la fase de creación o integración del proceso de publicación de *software* y conlleva un componente de automatización. (párr. 1)

Mediante el uso de la integración continua es posible identificar errores y corregirlos con mayor agilidad, en menos tiempo, e informar a los involucrados de actualizaciones de *software* (Amazon, 2021). Con la entrega continua, se amplía la integración continua al implementar todos los cambios en el código en un entorno de pruebas y/o de producción después de la fase de creación.

Figura 4

Ciclo de integración continua



Fuente: Amazon (2021) tomado de Continous Integration.

La figura 4 muestra en qué consiste la integración continua. Además, Amazon (2021) menciona que:

Con la integración continua, los desarrolladores envían los cambios de forma periódica a un repositorio compartido con un sistema de control de versiones

> como Git. Antes de cada envío, los desarrolladores pueden elegir ejecutar pruebas de unidad local en el código como medida de verificación adicional antes de la integración. Un servicio de integración continua crea y ejecuta automáticamente pruebas de unidad en los nuevos cambios realizados en el código para identificar inmediatamente cualquier error. (párr. 2)

La importancia de la comprobación delimita la utilidad del *software*, por esa razón es necesaria la integración continua, la cual también genera una serie de beneficios como:

- **Mejoramiento de la productividad de desarrollo:** Libera a los desarrolladores de tareas manuales y reduce por medio de comportamientos la cantidad de errores y *bugs* enviados a los clientes.
- **Encuentra y arregla los errores con mayor agilidad:** El equipo puede descubrir y arreglar los errores antes de que se conviertan en problemas más graves, gracias a la realización de pruebas más frecuentes.
- **Entrega las actualizaciones con mayor agilidad:** Permite a su equipo entregar actualizaciones a los clientes con mayor rapidez y frecuencia por medio de la integración continua. (Amazon, 2021)

2.7.3 Herramientas utilizadas en DevOps

Las herramientas que se utilizan en DevOps forman una parte fundamental y están relacionadas con el enfoque de la aplicación, ya que permiten agilizar el proceso y planteamiento desde el inicio hasta el final de la entrega del *software*. Al respecto, Amazon (2021) menciona las siguientes herramientas:

- **Herramientas de gestión de proyectos:** Se encargan de la administración y seguimiento de las tareas del proyecto. Entre las de código abierto, destacan GitHub Issues y Jira.
- **Repositorios de código fuente colaborativo:** Gracias a estos entornos, integrados con las herramientas de CI/CD, testeo y seguridad, los desarrolladores pueden trabajar en la misma base de código. Los repositorios de código abierto más destacados son GiHub y GitLab.
- **CI/CD pipelines:** Herramientas DevOps para la integración continua, es decir, la automatización de todo el ciclo de vida de un desarrollo. Jenkins y ArgoCD son algunas de las más populares en código abierto.
- **Test de la automatización de los *frameworks*:** Se refiere a las herramientas de prueba continua que graban y reproducen la funcionalidad de la aplicación. Se usan frecuentemente Selenium, Appium o Serenity como algunas de código abierto.
- **Herramientas de gestión de la configuración:** Se encargan de configurar y gestionar la infraestructura como código facilitando la labor de los ingenieros en la ejecución de un *script*. Las de código abierto más usadas son Ansible (Red Hat), Chef, Puppet y Terraform.
- **Herramientas de monitorización:** Recopilan datos en tiempo real y detectan posibles problemas que puedan afectar al rendimiento de las aplicaciones y la experiencia del usuario. Algunas de estas herramientas de código abierto son Datadog, Nagios, Prometheus y Splunk.
- **Medición del impacto:** Se implementan después del lanzamiento del producto. Su objetivo es registrar el comportamiento y la satisfacción de los

usuarios, mediante mapas de calor, encuestas o anotación de incidencias. (p. 2)

2.7.3.1 GitHub

El almacenamiento de información ha ido evolucionando en cuánto a su mantenimiento. Bustos (2021) menciona que:

> GitHub es un servicio basado en la nube que aloja un sistema de control de versiones (VCS) llamado Git. Este permite a los desarrolladores colaborar y realizar cambios en proyectos compartidos, a la vez que mantienen un seguimiento detallado de su progreso. (párr. 2)

Git tiene una historia particular que comenzó con un poco de destrucción creativa y una gran polémica. Chacón y Straub (2014) afirman que:

> El kernel de Linux es un proyecto de *software* de código abierto con un alcance bastante amplio. Durante la mayor parte del mantenimiento del kernel de Linux (1991-2002), los cambios en el *software* se realizaban a través de parches y archivos. En el 2002, el proyecto del kernel de Linux empezó a usar un DVCS propietario llamado BitKeeper.
>
> En el 2005, la relación entre la comunidad que desarrollaba el kernel de Linux y la compañía que desarrollaba BitKeeper se vino abajo y la herramienta dejó de ser ofrecida de manera gratuita. Esto impulsó a la comunidad de desarrollo de Linux (y en particular a Linus Torvalds, el creador de Linux) a desarrollar

su propia herramienta basada en algunas de las lecciones que aprendieron mientras usaban BitKeeper. (párrs. 2-3)

La importancia de descubrir una herramienta como GIT ha representado un hallazgo que ha agilizado el trabajo para los desarrolladores de manera que se busca resolver con lo disponible y en algunos casos innovar y generar nuevas opciones.

Chacón y Straub (2014) destacan objetivos como:

- Velocidad.
- Diseño sencillo.
- Gran soporte para desarrollo no lineal (miles de ramas paralelas).
- Completamente distribuido.
- Capaz de manejar grandes proyectos (como el kernel de Linux) eficientemente (velocidad y tamaño de los datos).

Desde su nacimiento en 2005, Git ha evolucionado y madurado. Chacón y Straub (2014) afirman que "es tremendamente rápido, muy eficiente con grandes proyectos y tiene un increíble sistema de ramificación (*branching*) para desarrollo no lineal" (párr. 6). Debido a su agilidad para la ejecución de trabajos presenta las siguientes funcionalidades que favorecen a los ingenieros de *software*:

- **Crear un repositorio de GitHub:** Considerado el eje central que contiene los archivos (código, imágenes, texto).

- **Crear ramas en GITHUB:** Genera diferentes versiones. Al realizar cambios en el proyecto (en la rama), el desarrollador visualiza las afectaciones o cambios en el proyecto.
- **Los *commits* de GITHUB:** Permite realizar cambios y ejecutar un *commit* para mantener el archivo de la rama de característica.
- **Crear solicitudes de extracción en GITHUB:** Estas facilitan el trabajo conjunto en los proyectos, ya que son la principal herramienta de colaboración en GitHub. Permiten observar la versión original y la que contiene cambios.

2.7.3.2 Slack

Para la comunicación por medio de mensajería se utilizan distintos medios que permiten actualizar al equipo de trabajo sobre todo lo relacionado con el proyecto. Tilman y Betters (2021) menciona:

> Slack es básicamente una aplicación de mensajería con variadas opciones. Está diseñado para equipos y lugares de trabajo que se pueden usar en múltiples dispositivos y plataformas, y está equipado con funciones sólidas que le permiten no solo chatear uno a uno con asociados, sino también en grupos. También puede cargar y compartir archivos con ellos, así como integrarse con otras aplicaciones y servicios, y puede controlar de forma granular casi todas las configuraciones, incluida la capacidad de crear emojis personalizados. (p. 2)

Este medio de comunicación es un método de transmisión para la información sobre el proyecto que se está ejecutando, que permite comunicarse en distintos lugares en tiempo real. Histografías.com (2021) menciona que:

> Slack fue fundada por Stewart Butterfield junto con Eric Costello, Cal Henderson y Serguei Mourachov en 2009. Sin embargo, la empresa no siempre se llamó Slack Technologies, se llamaba Tiny Speck y desarrollaban videojuegos online como Glitch. (párr. 2)

Debido a una necesidad surge una funcionalidad; en este caso, la contribución es realizada por parte de la tecnología que facilita la comunicación grupal o individual. Algunas de las funcionalidades de Slack son:

- Notificaciones constantes de cambios ejecutados en el proceso del proyecto que se está desarrollando.
- Distinción con estrellas de datos relevantes.
- Creación de canales que permite chatear en grupos.
- Envío de mensajería de forma individual o directa.
- Invitar nuevos ingresos a participar en el desarrollo del proyecto.
- Cargar y compartir archivos importantes, o bien para manejar un control de toda la documentación que se recibe o envía.

2.7.3.3 Github Actions

El control de lo indispensable en un proyecto es manejado y vigilado de manera automatizada. De acuerdo con Canorea (2018):

> GitHub Actions es una herramienta que permite reducir la cadena de acciones necesarias para la ejecución de código, mediante la creación de un flujo de trabajo encargado del *pipeline*. Es configurable para que GitHub reaccione a ciertos eventos de forma automática según nuestras preferencias. (p. 3)

La importancia de la automatización radica en que las configuraciones se realizan según eventos que representan cambios importantes dentro del desarrollo de un proyecto. Según Canorea (2018), “permite crear *workflows* que se puedan utilizar para compilar, testear y desplegar código. Además, da la posibilidad de crear flujos de integración y despliegue continuo dentro de nuestro repositorio” (párr. 8).

De acuerdo con Turrado (2020):

> Hace muchos años que aparecieron plataformas con las que se pueden gestionar el código fuente en internet. Los más veteranos en la sala recordarán como Codeplex o SourceForge se llevaban el mercado en los tiempos de Subversion (SVN). De hecho, aunque Codeplex está actualmente en modo archivo, SourceForge sigue funcionando y hospedando miles de proyectos como repositorio de control de versiones. (párr. 4)

La creación de plataformas a partir de un *software* generalmente surge como una necesidad que previene algún tipo de riesgo a futuro en la realización de un proyecto. Turrado (2021) menciona que:

> Plataformas como Github son sistemas web para control de código fuente que utilizan Git como sistema de versionado subyacente y son compatibles con sus protocolos. Es decir, aunque parezca una obviedad, es necesario aclarar que todos los repositorios de Github utilizan Git, pero no todos los repositorios Git están en Github. (p. 2)

GitHub Actions ofrece las siguientes funcionalidades que nos permiten controlar los despliegues:

- Activar flujos de trabajo con eventos diversos.
- Configurar ambientes para establecer reglas antes de que un *job* pueda proceder y para limitar el acceso a los secretos.
- Utilizar la concurrencia para controlar la cantidad de despliegues que se ejecutan al mismo tiempo.

2.7.3.4 Docker

El manejo de datos con amplia variedad de aspectos suele presentar problemas cuando se ejecuta en equipos que no tienen la capacidad necesaria. Al respecto, Docker busca “crear contenedores ligeros y portables para las aplicaciones *software* que puedan ejecutarse en cualquier máquina con Docker instalado, independientemente del sistema operativo que la máquina tenga por debajo, facilitando así también los despliegues” (Garzas, 2015, p. 5).

La creación de estos contenedores surgió como una necesidad de segmentar. Ramírez (2020) detalla que “surgió por primera vez en el año 2000 como

FreeBSD jail, una tecnología que permite la partición de un sistema FreeBSD en varios subsistemas o 'jaulas' (*jails*)", y permite la conexión grupal de forma segura y específica evitando el ingreso de usuarios no permitidos. Ramírez (2020) agrega que:

> En 2001, se introdujo en Linux la implementación de un entorno aislado, a través del proyecto VServer de Jacques Gélinas. Una vez que se estableció para múltiples espacios de usuario controlados en Linux, comenzó a tomar forma lo que hoy es un contenedor de Linux. (párr. 2)

> En 2008, Docker apareció en escena con su tecnología de contenedores que lleva el mismo nombre. La tecnología Docker incorporó una serie de conceptos y herramientas nuevos: una interfaz de línea de comandos sencilla para ejecutar y diseñar imágenes nuevas en capas, un daemon de servidor, una biblioteca de imágenes en contenedores prediseñadas y el concepto de un servidor de registros. (párr. 4)

Compartir entre usuarios fue uno de los beneficios que proporcionó el uso de Docker, ya que se realiza de forma segura y sin ninguna dificultad, garantizando la interoperabilidad de las tecnologías de contenedores. Para Garza (2015), algunas funcionalidades del Docker son:

- Beneficia desarrolladores, *testers*, donde se ejecutan las aplicaciones *software*, los procesos de despliegue, como administradores de sistemas, en relación con las máquinas.

- Desarrolla su código en la máquina donde se ejecutará sin preocuparse si dicho código funcionará.
- Es funcional para entornos de pruebas (*testing*).

Docker es una herramienta *open source*, que puede ser implementada por cualquier organización para así reducir el número de máquinas necesarias en ambientes de producción y a la vez estandarizar el ambiente de desarrollo para los desarrolladores y encargados de pruebas de calidad con el objetivo de volver el proceso de desarrollo más ágil.

CAPÍTULO III

MARCO CONTEXTUAL

3.1 Marco contextual

Para la ejecución de un trabajo con carácter investigativo es necesario implementar un marco contextual, ya que este amplía aspectos o características del entorno en el que se aplicará el estudio.

Según Castillo (2018):

> El marco contextual en un proyecto o tesis es el escenario físico, condiciones temporales y situación general que describen el entorno de un trabajo investigativo. De forma general, este puede contener aspectos sociales, culturales, históricos, económicos y culturales que se consideren relevantes para hacer una aproximación al objeto del estudio. (párr. 1)

A continuación, se describe la institución en la que se aplicarían los resultados de la presente investigación, con el objetivo de clarificar el porqué de las recomendaciones que se brindan.

3.2 Tipo de empresa

Todas las recomendaciones emitidas al final de la investigación se enfocarán en las necesidades de una empresa de desarrollo pyme enfocada en el desarrollo de sistemas y aplicaciones *in-house*, sea esto por requerimiento de sus clientes, o por necesidad propia del negocio que mantienen.

3.3 Ubicación geográfica

La recolección de los datos para la investigación se realizará en la gran área metropolitana del país, con el objetivo de recoger la mayor cantidad de datos posibles, ya que la mayoría de las empresas de desarrollo se encuentran en esta zona en el país. No obstante, las recomendaciones serán aplicables a cualquier empresa, sin importar su localización geográfica, siempre y cuando esta sea considerada una pyme.

CAPÍTULO IV

MARCO METODOLÓGICO

4.1 Marco metodológico

La formulación del marco metodológico en la investigación permite el tipo de investigación y la recolección de datos que luego serán comentados en los resultados obtenidos.

4.2 Enfoque de investigación

La presente investigación está enmarcada dentro del enfoque mixto, que analiza a profundidad diferentes variables que permitan determinar, tal como lo dice el título de la investigación, el "análisis de las tecnologías de desarrollo *cross-platform* existentes en el mercado y propuesta de un proceso de desarrollo automatizado aplicable a una pyme enfocada en el desarrollo de aplicaciones multiplataforma".

Según Sánchez (2013), en la investigación mixta el investigador combina métodos cuantitativos y cualitativos, por lo que se caracteriza por su pluralismo metodológico o eclecticismo, que a menudo resulta en la investigación superior.

Por ello, desde una perspectiva filosófica, la investigación mixta emplea el método pragmático y el sistema de la filosofía; es un método incluyente y plural. La meta de la investigación mixta no es remplazar a la investigación cuantitativa ni a la investigación cualitativa, sino utilizar las fortalezas de ambos tipos de indagación combinándolas y tratando de minimizar sus debilidades potenciales (Hernández, Fernández y Baptista, 2014).

El objeto de la investigación de métodos mixtos es obtener de ambos métodos sus fortalezas y minimizar sus debilidades. Los métodos de la investigación mixta son también un intento de legitimar el uso de múltiples enfoques para responder a las preguntas de investigación, en lugar de restringir o limitar las opciones de los investigadores y así contrarrestar el dogmatismo (Sánchez, 2013).

Los aspectos cuantitativos se desarrollan con el objetivo de obtener la información numérica acerca de una propiedad o cualidad del objeto o fenómeno, y comparar magnitudes medibles y conocidas. Es decir, es la atribución de valores numéricos a las propiedades de los objetos (p. 15). En el presente caso se trata de una encuesta aplicada a desarrolladores, mediante la cual se establece la medición de una serie de criterios y opiniones; se trata de llegar a cifras de los datos recopilados.

En este caso, la investigación cualitativa profundiza en aspectos medidos de los que se necesitan apreciaciones y razonamientos, tal es el caso de entrevistas a encargados de tecnología de desarrollo móvil y arquitectos; aquí no se trata de la medición sino de la comprensión de una serie de expresiones y razonamientos, y de profundizar, por sobre el análisis estadístico. También, aquí se incluye el análisis de una serie de documentos relativos al tema, sobre una selección de artículos que serán sometidos a un análisis documental.

De esta manera se pretende contar con resultados tangibles, producto de los instrumentos aplicados a la población muestra, que permitan comprender la situación actual respecto a la implementación de las tecnologías de desarrollo *cross-*

platform y las mejores prácticas de automatización del proceso de desarrollo de *software* ya implementadas en empresas de nivel nacional, con el objetivo de obtener recomendaciones para el sector pyme del país.

Los datos y su análisis son los elementos utilizados con el objetivo de elaborar un plan o protocolo para situaciones futuras, buscando mantener una continuidad de los entrenamientos técnicos de sus colaboradores y que estos puedan sentirse no solamente satisfechos, sino también ser competitivos tanto en su empresa actual como en otro mercado laboral a futuro.

4.3 Fuentes y sujetos de información

Se realiza la inclusión de varias fuentes primarias, ya que estas son concebidas como la primera y más importante fuente de búsqueda, al englobar la información actual sin modificarse ni mezclarse con otras ideas, fundamentos ni conceptos. Las fuentes primarias principales de esta investigación son las siguientes:

- Entrevistas.
- Encuestas.
- Documentación de tecnologías *cross-platform*.

Además de las fuentes primarias, existen las fuentes secundarias, las cuales se encargan del procesamiento de la información de primera mano. En esta investigación, se consideran las siguientes:

- Libros.

- Tesis.
- Artículos generales.
- Blogs.
- Videotutoriales sobre desarrollo móvil.
- Informes de implementación de tecnologías.

4.4 Definición de la población

Dentro del desarrollo de una investigación y según el tipo que sea, es necesario definir los involucrados o participantes que vienen a ser las personas sobre las cuales se recopilan los datos y muestran un panorama del resultado.

Hernández *et al*. (2014) expresan que se entiende por población al conjunto total de individuos, objetos o medidas que poseen algunas características comunes observables en un lugar y en un momento determinado. Debido a esto, es importante delimitar la población a estudiar, ya que una definición inadecuada puede extender la investigación al punto de no completarse o incluso arrojar resultados erróneos.

Para esclarecer más este punto, es importante acotar que, como dicen Hernández *et al*. (2014):

> La delimitación de las características de la población no solo depende de los objetivos de la investigación, sino de otras razones prácticas. Un estudio no será mejor por tener una población más grande; la calidad de un trabajo investigativo estriba en delimitar claramente la población con base en el

> planteamiento del problema. Las poblaciones deben situarse claramente por sus características de contenido, lugar y tiempo. (p. 174)

Con base en lo anterior, y teniendo siempre claro el objetivo de la investigación, se pretende identificar tres tipos de población:

- Arquitectos de aplicaciones.
- Administradores de proyectos de desarrollo móvil.
- Desarrolladores de tecnologías móviles que tengan o hayan tenido experiencia en el uso de tecnologías *cross-platform*.

4.5 Definición de la muestra

Una vez definida la población se procede a definir la muestra para esta investigación. De acuerdo con López (2014), la muestra:

> Es un subconjunto o parte del universo o población en que se llevará a cabo la investigación. Hay procedimientos para obtener la cantidad de los componentes de la muestra como fórmulas, lógica y otros … La muestra es una parte representativa de la población. (p. 1)

La muestra es un subconjunto de aspectos vistos desde casos o individuos de una población que la conforman, pero que mantienen características representativas entre ellos, o comparten diversos elementos.

Las primeras dos poblaciones están abarcadas en la metodología cualitativa, por lo tanto la muestra, como indica Corral (2012), lo que requiere es un muestreo

de criterio, es decir, la escogencia de las personas que se consideren más representativas de una base de datos debido a su actividad o al tipo de empresas. Entonces la muestra queda de la siguiente manera:

Tabla 7

Definición de la población

Definición de la población de la empresa 3Pillar Global de C. R.	
Cargo	Número de personas
Arquitectos de aplicaciones móviles en la empresa 3Pillar Global de Costa Rica	121
Administradores de proyectos de desarrollo móvil en la empresa 3Pillar Global de Costa Rica	111
Desarrolladores de aplicaciones móviles de 3Pillar Global de Costa Rica	132

Fuente: Elaboración propia.

Debido a que la población se ha definido como los profesionales de tecnologías de la información con enfoque en desarrollo de aplicaciones móviles que forman parte de la empresa 3Pillar Global de Costa Rica (121 arquitectos, 132 desarrolladores y 111 administradores de proyectos de desarrollo), se ha decidido aplicar la fórmula estadística de cálculo de muestras para así obtener una muestra con un nivel de confianza del 95% y un margen de error máximo del 5%.

Figura 5

Cálculo de la muestra

$$\text{Tamaño de la muestra} = \frac{\frac{z^2 \times p\,(1-p)}{e^2}}{1 + \left(\frac{z^2 \times p\,(1-p)}{e^2 N}\right)}$$

Fuente: Elaboración propia.

Donde:

- N = tamaño de la población
- e = margen de error (porcentaje expresado con decimales)
- z = puntuación z

La puntuación z se refiere a la cantidad de desviaciones estándar en que una proporción determinada se aleja de la media. En el caso del nivel de confianza deseado en la presente investigación (95%), esta variable toma un valor de 1,64.

En este caso, se toman las poblaciones mencionadas anteriormente, todas pertenecientes a la empresa 3Pillar Global de Costa Rica, y se procede a calcular la muestra para cada una de ellas con un margen de error del 5%.

Tabla 8

Distribución de la muestra del estudio

Distribución de la muestra del estudio		
Estratos del estudio	**Población**	**Muestra**
Arquitectos de aplicaciones móviles en la empresa 3Pillar Global de Costa Rica	121	93
Desarrolladores de aplicaciones móviles de 3Pillar Global de Costa Rica	132	99
Administradores de aplicaciones móviles de 3Pillar Global de Costa Rica	111	87

Fuente: Elaboración propia.

4.6 Métodos de recolección

Es importante indicar que, para abarcar los temas y realizar una investigación que cumpla con los objetivos, se debe contar con un marco de datos que permita la

obtención y recopilación de la información. Básicamente, se busca una relación entre las fuentes y los métodos a utilizar, además de todo lo que conlleva.

Dado el tipo de información que se pretende recopilar, la mejor manera para recolectar la información es por medio de consultas a los diferentes actores definidos en la investigación, junto con la lectura y el análisis de las publicaciones que se consideren necesarias para tener un panorama más claro.

Para esta investigación, los actores son los desarrolladores, administradores de proyectos y arquitectos de *software* de aplicaciones móviles de la empresa 3Pillar Global de Costa Rica. Cada uno de ellos puede agregar a la investigación información de calidad que nos permitirá desarrollar de manera más profunda los temas analizados y brindar un resultado claro y conciso sobre el cual pueda desarrollarse un proceso de desarrollo de aplicaciones móviles aplicable a una empresa pyme.

4.6.1 Análisis documental

De acuerdo con Vera (2017), este tipo de investigación es la que se realiza, como su nombre lo indica, apoyándose en fuentes de carácter documental, esto es, en documentos de cualquier tipo. Como subtipos de esta investigación se encuentran la investigación bibliográfica, la hemerográfica y la archivística; la primera se basa en la consulta de libros, la segunda en artículos o ensayos de revistas y periódicos, y la tercera en documentos que se encuentran en los archivos, como cartas, oficios, circulares, expedientes, entre otros.

Se caracteriza, además, porque busca la aplicación o utilización de los conocimientos que se adquieren. En una investigación empírica, lo que le interesa al investigador, primordialmente, son las consecuencias prácticas. En este caso, se toman en cuenta artículos especializados sobre el tema que aportan información útil.

4.6.2 Cuestionarios

Esta técnica será aplicada con los diferentes sujetos, con el objetivo de definir las tecnologías presentes en el mercado actual y las mejores alternativas con respecto al desarrollo o manejo de procesos de desarrollo. Para este trabajo se seleccionó el cuestionario (apéndices b, c y d), ya que puede ser aplicado por medio de una encuesta.

Según Mills *et al*. (2010), un cuestionario consiste en un conjunto de preguntas respecto de una o más variables a medir. Hernández *et al*. (2014) afirman que los cuestionarios se utilizan en encuestas de todo tipo (por ejemplo, para calificar el desempeño de un gobierno, conocer las necesidades de hábitat de futuros compradores de viviendas o evaluar la percepción ciudadana sobre ciertos problemas como la inseguridad). Pero también se implementan en otros campos: por ejemplo, un ingeniero en minas usó un cuestionario como herramienta para que expertos de diversas partes del mundo aportarán opiniones calificadas con el fin de resolver ciertas problemáticas de producción.

4.7 Elaboración de instrumentos

Para cada investigación es necesario la recopilación de datos y para cada recopilación de datos se requiere de instrumentos que faciliten la tarea al investigador, pero estos deben tener un propósito claro y no desarrollarse sin un objetivo. Para lograr el cometido de esta investigación se utilizarán múltiples instrumentos, entre ellos:

4.7.1 Revisión y análisis documental

Se analizarán artículos y publicaciones que se relacionen con el tema de la investigación de manera directa o indirecta.

4.7.2 Cuestionarios a los arquitectos de *software*

Se realizarán cuestionarios a arquitectos de *software*, con el objetivo de identificar las mejores prácticas y tecnologías que han sido aplicadas en el país y su proceso de implementación (ver Apéndice B – Entrevista a los arquitectos).

4.7.3 Encuestas a los profesionales de desarrollo

Se llevarán a cabo entrevistas y encuestas a profesionales del área de desarrollo de aplicaciones móviles, con el objetivo de obtener la realidad del mercado laboral hoy en día (ver Apéndice C - Encuesta a los desarrolladores de tecnologías móviles).

4.7.4 Entrevistas a los administradores de proyectos de desarrollo

Se entrevistarán administradores de proyectos profesionales especializados en el área de desarrollo, con el objetivo de identificar las mejores prácticas que han sido aplicadas en el país y su proceso de implementación (ver Apéndice D – Entrevista a los administradores de proyectos).

4.8 Tabulación y manejo de información

Con la finalidad de centrar la información recopilada, se busca realizar una separación con base en temas de diferentes índoles que involucren aspectos claves según cada actor consultado, sobre cada una de las variables, para entregar un producto tangible que permita conocer cómo implementar una solución práctica, eficiente y funcional. Esta entrega se hará principalmente por medio de gráficos, con la intención de manejar visualmente los datos para identificar características en ellos. Esta es la parte cuantitativa de la investigación.

Los gráficos tienen la particularidad de dar cuenta de dichas relaciones y de dichos escenarios de un modo muy elocuente, que fácilmente podría ser entendido por cualquiera como para conocer las conclusiones.

Dado el mecanismo utilizado y la herramienta de recolección de datos, los gráficos entregados con las respuestas de los actores encuestados facilitan visualizar la información de una manera más sencilla para la investigación, permitiendo así contar con una clasificación rápida, fácil y sencilla de interpretar.

En lo cualitativo, que abarca los artículos a estudiar, se empieza con el análisis de contenido.

Hernández *et al.* (2014) lo definen como una “técnica objetiva y a la vez sistemática, que cuantifica los contenidos en categorías y subcategorías y los somete a un análisis estadístico” (p. 251). Es una manera particular de analizar los documentos de una investigación.

Consiste en seleccionar las ideas relevantes que se recopilan del documento con la finalidad de ordenarlas de acuerdo con variables y métricas. El análisis puede adquirir la forma de un sumario, un resumen, un índice alfabético de materias o códigos sistemáticos. Luego viene el análisis de contenido que, según Terán (2019), es una técnica que se utiliza para investigaciones sobre temas de comunicación, literatura, educación, entre otros; el propósito de esta técnica es analizar y estudiar la información de una manera objetiva, cuantitativa y sistemática. Al final lo que se pretende es que se relacione la información de los artículos de cada variable de manera que se muestren similitudes y diferencias que conduzcan a una síntesis.

4.9 Variables

La obtención de la información y el análisis se centran en una serie de unidades de análisis con sus respectivas variables de estudio que se describen en la siguiente tabla.

Tabla 9

Distribución de variables a medir

Distribución de variables a medir	
Unidad de análisis	**Variable**
Uso de tecnologías de desarrollo *cross-platform* existentes en el mercado	Tecnologías de desarrollo existentes
	Utilidad
	Mercado de tecnologías
	Aspectos orientadores del proyecto
Uso de metodologías de administración de proyectos de *software* existentes en el mercado	Técnicas de administración de equipos DevOps
	Metodologías de administración existentes
	Prácticas DevOps

Fuente: Elaboración propia.

4.10 Matriz metodológica

Tabla 10

Matriz metodológica: objetivo específico 1

Objetivo específico	Describir la evolución del sector de desarrollo con tecnologías *cross-platform*, por medio de investigación de la documentación tecnológica para elaborar una base de conocimientos.			
Variable	**Conceptualización**	**Dimensión**	**Indicadores y métrica**	**Instrumentos**
Tecnologías de desarrollo existentes	Tecnología enfocada al desarrollo de aplicaciones compatibles con múltiples plataformas	Descripción de los diferentes tipos	Madurez Uso de las tecnologías de desarrollo *cross-platform* por desarrolladores	Revisión documental
Balance de uso	Descripción de los beneficios y desventajas de cada uno de los tipos con respecto a otros	Funciones y rendimiento de las opciones existentes	Tipos de aplicaciones, web o escritorio Sistemas operativos soportados	Revisión documental

Fuente: Elaboración propia.

Tabla 11

Matriz metodológica: objetivo específico 2

Objetivo específico	Identificar las prácticas de desarrollo existentes y sus características en las empresas del territorio costarricense por medio de encuestas a profesionales del área, así como identificar las dificultades que experimentaron al implementar estas prácticas en sus proyectos de desarrollo.			
Variable	**Conceptualización**	**Dimensión**	**Indicadores y métricas**	**Instrumentos**
Mercado de tecnologías Prácticas DevOps	Se refiere a todas las prácticas que se aplican en una empresa para agilizar el proceso de desarrollo.	Experiencia profesional de las prácticas Utilización de herramientas de automatización	● Mercado, web o móvil ● Experiencia de los arquitectos ● Tecnologías utilizadas previamente por el arquitecto en proyectos ● Criterio del arquitecto (vale la pena trabajar híbrido o nativo) ● Mayor reto cuando se desarrolla una aplicación	Entrevista a arquitectos de *software* Entrevista a desarrolladores

		Trabajo en DEVS Metodologías y herramientas DEV Uso de tecnologías	● Mayor reto al implementar una aplicación ● Arquitectura preferida por los arquitectos de *software* ● Experiencia de usuario importante ● Herramientas de automatización existentes utilizadas ● Utilización de las herramientas ● Finalidad de uso de las herramientas ● Curva de aprendizaje en el uso de herramientas ● Trabajo con *apps* en el mercado ● Relevancia de la UX de los devs ● Tiempo que se considera adecuado para hacer un mvp	Entrevista a administradores de proyectos

			● Tecnologías de desarrollo *cross-platform* usadas en los devs ● Tecnologías de *cross-platform* más fáciles de aprender ● Tecnologías preferidas por el desarrollador común ● *Apps* de desarrollo *cross-platform* existentes ● Consideración de nativo o *cross-platform* ● Metodologías trabajadas en el dev ● Herramientas de automatización usadas en el dev en los proyectos ● Metodologías más usadas ● Facilidad de implementación de metodologías ● Herramientas más usadas por administradores	

			● Porcentaje de proyectos híbridos ● Tecnologías usadas en los proyectos ● Tecnología más fácil de aprender para los devs ● Decisión de la empresa de invertir en los devs aprendiendo esas tecnologías ● Importancia de automatizar procesos	

Fuente: Elaboración propia.

Tabla 12

Matriz metodológica: objetivo específico 3

Objetivo específico	Proponer un marco de trabajo de entrega y mejora continua basado en las mejores prácticas para proyectos de desarrollo de *software* para aplicaciones móviles para equipos de desarrollo de empresas pyme.			
Variable	**Conceptualización**	**Dimensión**	**Indicadores y métrica**	**Instrumentos**
Aspectos orientadores del proyecto	Definir los ejes que guían el marco	● Objetivos ● Justificación ● Aspectos contextuales (breve reseña de la necesidad)		Con base en objetivos anteriores

Técnicas de administración de equipos DevOps	Todas las prácticas que engloba la administración de un equipo de desarrollo DevOps ágil	Recursos Organización	● Actividades de calidad ● Técnicas por aplicar ● Herramientas ● Capital humano ● Conocer el tiempo de duración del proyecto ● Tener definidos los roles y responsabilidades ● Tener identificados los riesgos potenciales que puedan afectar la ejecución de las actividades de calidad ● Actividades principales	Con base en objetivos anteriores

Fuente: Elaboración propia.

CAPÍTULO V

ANÁLISIS DE RESULTADOS

5.1 Análisis de resultados

En este apartado se presenta el análisis de la revisión integrativa referente al tema de estudio: “Análisis de las tecnologías de desarrollo *cross-platform* existentes en el mercado y propuesta de un proceso de desarrollo automatizado aplicable a una pyme enfocada en el desarrollo de aplicaciones multiplataforma”. Para ello se consideran referentes de bases de datos nacionales e internacionales en las cuales se encontraron estudios especializados sobre el tema y relacionados con los objetivos planteados en la presente investigación.

5.2 Describir la evolución del sector de desarrollo con tecnologías *cross-platform*, por medio de investigación de la documentación tecnológica para elaborar una base de conocimientos

5.2.1 Tecnologías *cross-platform*

5.2.1.1 Comprensión de la arquitectura React Native (2020)

Autor: Jun Kaneko.

En el artículo escrito para el sitio web Medium, el autor busca dar una explicación general de la arquitectura React Native para lograr una mejor comprensión de esta. Native React constituye un excelente punto de partida para el desarrollo de aplicaciones como desarrollador web, pero para ello es necesario tener conocimiento sobre el lenguaje JavaScript y comprender de manera general plataformas como iOS y Android.

En el entorno de ejecución de React Native, se pueden identificar tres subprocesos: el subproceso de JavaScript, el subproceso principal nativo y el subproceso de fondo para manejar Shadow Node.

React, tal como indica el autor, es “bastante intuitivo, ya que el concepto clave, el DOM virtual, implica la representación del típico DOM HTML que utilizan los navegadores” (párr. 10). Proporciona una abstracción para la interfaz de usuario declarativa, cuya representación virtual de una UI es guardada en la memoria y se puede sincronizar con las bibliotecas de UI externas, lo que se conoce como reconciliación.

Se indica en el artículo de Jun Kaneco que:

> React Native proporciona su propia capa de abstracción de UI en plataformas iOS y Android. React Native core y los componentes nativos invocan las vistas nativas para que pueda escribir la interfaz de usuario de la aplicación del teléfono inteligente con JavaScript, en lugar de Kotlin / Java o Swift / Objective-C. (párr. 18)

React Native constituye un tipo de biblioteca de JavaScript en la cual esta no incluye ningún código nativo. Se basa en bibliotecas como Reanimated, Gesture Handler y Screens, con el objetivo de implementar patrones de navegación de aplicaciones comunes. A la vez “proporciona la mejor práctica sobre cómo estructurar y navegar por las pantallas de la aplicación, que es una de las partes más confusas cuando se tiene experiencia en desarrollo web” (párr. 11).

A pesar de que pueden existir diferencias entre la implementación de interfaces de usuario, Jun Kaneco indica que no cambia el proceso de pensamiento para la creación de aplicaciones, en el cual se debe seguir el “Pensar en React”:

- Empezar con un simulacro.
- Dividir la interfaz de usuario en una jerarquía de componentes.
- Crear una versión estática en React.
- Identificar la representación mínima (pero completa) del estado de la interfaz de usuario.
- Identificar dónde debería vivir su estado.
- Agregar flujo de datos inverso. (párr. 24)

React presenta como característica la composición de los componentes, en los que lo más importante es agregar funcionalidad a cada componente sin que se dé cambio alguno en la base del código.

5.2.1.2 React Native vs. Flutter: ¿cuál elegir para el desarrollo multiplataforma? (2021)

Autora: Anna Dziuba.

La autora del artículo traza como objetivo brindar una serie de diferencias entre Flutter y React Native basándose en ocho aspectos que se resumen a continuación:

1. Lenguaje de programación:
    a. React Native usa JavaScript.
    b. Flutter usa Dart. (párr. 6)
2. Arquitectura:
    a. React Native permite la comunicación entre JavaScript y el lenguaje de programación nativo a través del puente JavaScript.
    b. Flutter no requiere un puente para comunicarse con los componentes nativos, ya que compila a código nativo. (párrs. 10-11)
3. Instalación y configuración inicial:
    a. El proceso de configuración de React Native se basa en CLI globalmente a través de la línea de comando, tomando en cuenta que también se necesitará NodeJS y Yarn instalados como administrador de paquetes.
    b. Para instalar Flutter se hace descargando el repositorio de Flutter desde Github, luego se ejecuta el Flutter doctor. El siguiente paso es agregarlo al PATH; esto se puede hacer a través de la línea de comandos. Sin embargo, con esta configuración adicional, Flutter pierde puntos frente a React Native. Su instalación no es tan sencilla. (párrs. 13-14)
4. Herramientas de desarrollo y documentación:
    a. React Native tiene documentación decente.
    b. Flutter tiene una rica documentación; ofrece guías detalladas acompañadas de gráficos y tutoriales en video. (párrs. 15-16)

5. Interfaz de usuario:
    a. Gracias al puente de JavaScript, React Native renderiza los componentes nativos para cada plataforma. Esta característica permite crear la apariencia de las aplicaciones nativas de Android y iOS.
    b. Flutter convierte todo en *widgets* de interfaz de usuario, que a su vez compilan a código nativo al ser empaquetados en la aplicación, lo que significa que crear aplicaciones de tipo nativo es fácil. (párrs. 17,19)
6. Productividad de los desarrolladores:
    a. React Native tiene una capacidad de reutilización de código muy alta en todas las plataformas.
    b. Flutter, aparte de la capacidad de reutilización al ser estructurado como un *widget*, también tiene una función de recarga en caliente, lo que significa que puede realizar cualquier iteración rápidamente y recibir comentarios de inmediato. (párrs. 20-21)
7. Soporte comunitario:
    a. React Native se caracteriza por tener una amplia comunidad de colaboradores. Dicho marco fue desarrollado en 2015 por Facebook en donde contaron con un amplio apoyo de desarrolladores.
    b. Flutter es un marco de desarrollo más nuevo, creado en 2017 por Google; eso no significa que no haya ido creciendo y ganando popularidad entre los desarrolladores de manera exponencial en los últimos años.

8. Soporte CI/CD:
    a. React Native no tiene una solución CI/CD para la entrega de App Store o Google Play. Desafortunadamente, solo se documenta la implementación manual para Google Play
    b. Flutter tiene una guía bien documentada para crear e implementar aplicaciones de iOS y Android. Se puede implementar una aplicación en diferentes plataformas simplemente usando CLI (interfaz de línea de comando). (párr. 24)

Concluye la autora del artículo que el escoger entre React Native o Flutter va a depender de las necesidades y gustos de los desarrolladores. React Native es una herramienta para crear aplicaciones multiplataforma o si el proyecto a realizar es algo grande; por otro lado, si se quiere desarrollar una aplicación para web o escritorio se debe elegir Flutter.

5.2.1.3 Flutter vs. React Native en 2021: ¿cuál es mejor para su proyecto? (2021)

Autor: Serhii Osadchuk.

En su artículo, Serhii Osadchuk hace mención de los principales desarrolladores de aplicaciones: Flutter, React Native, Cordova, Ionic y Xamarin. Asimismo, presenta las principales ventajas y desventajas de las dos arquitecturas más utilizadas: Flutter y React Native.

Además de Flutter y React Native, en el mercado se pueden encontrar otras arquitecturas para el desarrollo de aplicaciones, producto de la evolución del desarrollo de las multiplataformas a nivel mundial. Actualmente, se pueden encontrar los siguientes marcos para el desarrollo de aplicaciones: React Native, Flutter, Cordova, Ionic y Xamarin.Forms.

Serhii Osadchuk menciona que evidentemente las características de Flutter y React Native son bastante evidentes, por lo que a continuación se resumen los principales componentes de las herramientas de desarrollo multiplataforma.

- **Xamarin**: Es una extensión de la plataforma de desarrollo .NET, una herramienta multiplataforma de código abierto para crear aplicaciones iOS, Android y Windows a partir de una única base de código compartida. Xamarin se puede adaptar mejor al ecosistema .NET y Windows por ser desarrollado por Microsoft. El lenguaje de Xamarin marcado personalizado XAML o C # para crear interfaces de usuario habilita el acceso a características específicas de la plataforma como el área segura de iOS, la elevación de Android o Windows ListView. (párrs. 6-8)
- **Ionic**: Es un kit de herramientas de interfaz de usuario móvil de código abierto para crear experiencias de aplicaciones web y nativas multiplataforma, escritas en JS. Las aplicaciones React.JS, Vue.JS y Angular.JS pueden incorporar componentes iónicos para implementar interfaces adaptativas de apariencia ordenada y fácilmente programables. Las aplicaciones que se

ejecutan en JS puro tampoco tendrían problemas para usar Ionic. (párrs. 9-10)

- **Cordova**: Es una herramienta de desarrollo multiplataforma de código abierto, excepto que es la más controvertida y la menos popular. Cordova no es autosuficiente para construir una interfaz atractiva. Sus complementos respaldan las capacidades de Ionic para trabajar con *hardware*, mientras que los elementos Ionic dan a las aplicaciones un aspecto nativo. Entonces estas dos herramientas son codependientes. Cordova usa WebView basado en navegador, que empeora notablemente el rendimiento en dispositivos de nivel medio y bajo. (párrs. 12-14)

En el artículo, el autor menciona los pros y los contras de los principales desarrolladores de aplicaciones, Flutter y React Native, a saber:

- Ventajas de Flutter:
    - Desarrollo e implementación más rápidos gracias a las funciones integradas, como “recarga en caliente”.
    - Documentación de calidad, indispensable para proyectos de código abierto.
    - Interfaces de usuario ricas en funciones que son totalmente personalizables hasta el último píxel.
    - La compatibilidad con dispositivos más antiguos garantiza una representación y funcionalidad adecuadas en las versiones de Android a partir de la 5.1.1, y en las versiones 8 de iOS y posteriores.

- La interfaz de usuario de Flutter está separada de la interfaz de usuario nativa. (párrs. 24-27)

- Desventajas de Flutter:

  - Edad marco: Flutter es bastante joven por lo que sus usuarios no han tenido la capacidad de proponer una gran variedad de casos de uso un equipo está en aguas profundas cuando se trata del desarrollo de proyectos altamente complejos o de nicho.
  - Evolución dinámica: No es una desventaja definida del marco, sino más bien un desafío. Los cambios frecuentes en el entorno de desarrollo significan que un producto se comportará de manera diferente después de cada actualización.
  - Tamaño del proyecto: Los archivos de proyecto ocupan más espacio en comparación con los creados con otras herramientas. (párrs. 29-31)

- Ventajas de React Native:

  - Componentes nativos específicos de la plataforma: Reconoce la importancia de importar componentes nativos específicos de la plataforma.
  - Bibliotecas web y ReactXP: Son una adición útil al marco si se considera lanzar una versión web de una aplicación determinada.
  - Popularidad: JavaScript es muy popular entre los desarrolladores; es el tercer lenguaje más popular basado en datos PYPL.

- Rendimiento: Los puntos de referencia demuestran cómo RN a veces es un poco mejor que Swift para iOS.
- Recarga en vivo: La recarga en vivo/en caliente (básicamente idéntica a la función correspondiente de Flutter) evita muchos dolores de cabeza a los desarrolladores, permitiéndoles ver los cambios sin la reconstrucción del proyecto
- CodePush: Un servicio en la nube de App Center que permite implementar actualizaciones de aplicaciones móviles directamente en los dispositivos de los usuarios finales. (párrs. 34-39)

- Desventajas de React Native:
  - Mala documentación.
  - Muchas bibliotecas nativas interconectadas con el SDK nativo.
  - Independencia parcial de una plataforma nativa. (párrs. 43-45)

Del artículo se concluye que hay dos marcos de desarrollo principales para el desarrollo de aplicaciones híbridas, y que React Native y Flutter ocupan esos lugares. React es más fácil de aprender, ya que se basa en JavaScript, pero Flutter, al ser respaldado por Google, se está convirtiendo en una alternativa bastante viable a React Native, la cual se debe considerar al momento de elegir el marco de desarrollo del proyecto.

5.2.1.4 Síntesis

JavaScript viene a conformarse como un multiparadigma de programación orientada a objetos y de programación funcional, y React ha heredado la fuerza de

ambos. El desarrollador JavaScript, React y React Native constituyen un marco muy útil para aprender, en el que se saca provecho de la capacidad de JavaScript al máximo nivel, y se pueden realizar variadas prácticas.

Al elegir entre la arquitectura Flutter o React Native, el desarrollador debe considerar su presupuesto. Si se cuenta con un presupuesto limitado o se quiere solamente hacer una aplicación sencilla se debe escoger la arquitectura Flutter, mientras que si se cuenta con suficiente presupuesto y se desea crear una aplicación algo más compleja se puede elegir React Native.

Elegir un marco adecuado para una futura aplicación se puede determinar por la productividad demostrada y por su conjunto de características. Cabe mencionar que una aplicación mal diseñada puede llegar a sobrecalentar un dispositivo en uso, lo que ocasiona retrasos y mal funcionamiento. Es por ello por lo que en ciertas ocasiones se origina una confrontación de larga data entre la elección de un desarrollador nativo y multiplataforma; esa confrontación particularmente se da entre las multiplataformas Flutter y React Native.

De la bibliografía consultada se desprende que Flutter tiene su propio motor de renderizado lo que permite la creación de diseños únicos. React Native está restringido a cierto uso de componentes nativos que requieren de una mayor personalización. En Flutter las aplicaciones pueden “portarse” (implementarse de manera nativa) al entorno de escritorio de manera inmediata, mientras que una aplicación de React debe usar una interfaz web como intermediaria para esa situación.

5.2.2 Metodologías de administración de proyectos

5.2.2.1 ¿Qué es la gestión de proyectos? (2022)

Autor: Project Management Institute.

Cualquier proyecto que se desee llevar a cabo requerirá de algún tipo de esfuerzo. Dicho proyecto contará con una etapa de inicio y una etapa final, y tendrá resultado la creación de un valor u obra determinada.

En el artículo definen la gestión de proyectos como “el uso de conocimientos, habilidades, herramientas y técnicas específicas para ofrecer algo de valor a las personas” (párr. 1). Como ejemplos de proyectos se pueden mencionar el desarrollo de un *software*, la construcción de un edificio, entre otros.

Efectos de la globalización y avances tecnológicos obligan a las empresas a establecer nuevas modalidades de trabajo en las cuales para lograr organizar los trabajos se requieren que estos sean desarrollados por medio de proyectos. Para ello, algunas empresas desarrolladoras se han establecido en el mercado con el objetivo de orientar a otras por medio de profesionales que brindan herramientas, técnicas y enfoques con el propósito de lograr los objetivos.

El tema de gestión de proyectos se está viendo en el mercado como una oportunidad para nuevos profesionales y para que algunas empresas surjan como especializadas en el área; como se menciona en el artículo:

> Las habilidades de gestión de proyectos pueden ayudar a un joven estudiante que trabaja en un proyecto de ciencias a lograr el éxito, o a un ejecutivo

> corporativo a resolver disputas de personalidad. Estas habilidades pueden ayudar a una enfermera a optimizar los cambios de turno para mejorar los tiempos de respuesta de los pacientes en su sala. Pueden ayudar a un profesional de TI a entregar *software* innovador en un tiempo récord o ayudar a una agencia gubernamental a mejorar los servicios que brindan de una manera más económica. (párr. 13)

A nivel mundial los profesionales pueden adquirir una certificación Project Management Professional (PMP)®, mediante la cual se impulsa a nuevos líderes en proyectos y se desarrollan habilidades para el trabajo más inteligente y con mejor desempeño.

5.2.2.2 ¿Quiénes son los directores de proyectos? (2022)

Autor: Project Management Institute.

Para desarrollar proyectos se requiere de líderes y ese campo lo vienen a llenar los directores de proyectos, quienes tienen diversos conjuntos de habilidades que les permiten abordar cada tarea de una manera única y estratégica. Lo más importante es que entienden cómo aprovechar sus habilidades de gestión de proyectos para fomentar la capacidad de una organización para aprender, tener éxito y evolucionar con un proyecto.

Los directores de proyectos son personas que deben trabajar bajo presión y deben saber hacer frente a los cambios y al entorno complejo en el cual deben trabajar. “Adaptan su enfoque al contexto y las limitaciones de cada proyecto,

sabiendo que ningún “talla única” puede adaptarse a toda la variedad de proyectos. Y siempre están mejorando sus propias habilidades y las de sus equipos a través de revisiones de lecciones aprendidas al finalizar el proyecto” (párr. 26).

5.2.2.3 Agilidad amplificada. El último pulso de la profesión de PMI revela poderes gimnásticos (2022)

Autor: Project Management Institute.

En el artículo se desarrolla cómo las empresas actualmente deben desarrollar cierto tipo de habilidades para hacer frente a la realidad actual; dichas empresas se les conoce con el nombre de “empresas gimnásticas”, las cuales “combinan estructura, forma y gobierno con la capacidad de flexionarse y pivotar, bajo demanda” (párr. 2).

A continuación, se enlista una serie de características de las empresas gimnásticas con datos gráficos:

- Presentan ventaja competitiva.

Figura 6

Ventaja competitiva de una empresa ágil

	Gymnastic	Traditional
Digital transformation	73%	62%
Business strategy	70%	57%
Organizational adaptability	70%	53%
Operational efficiency	65%	52%
Innovation approach	65%	46%
Diversity, equity and inclusion initiatives	60%	51%
Organizational culture	60%	48%
Project management maturity	57%	41%
Talent management	48%	36%

Gymnastic enterprises also have a project performance advantage.

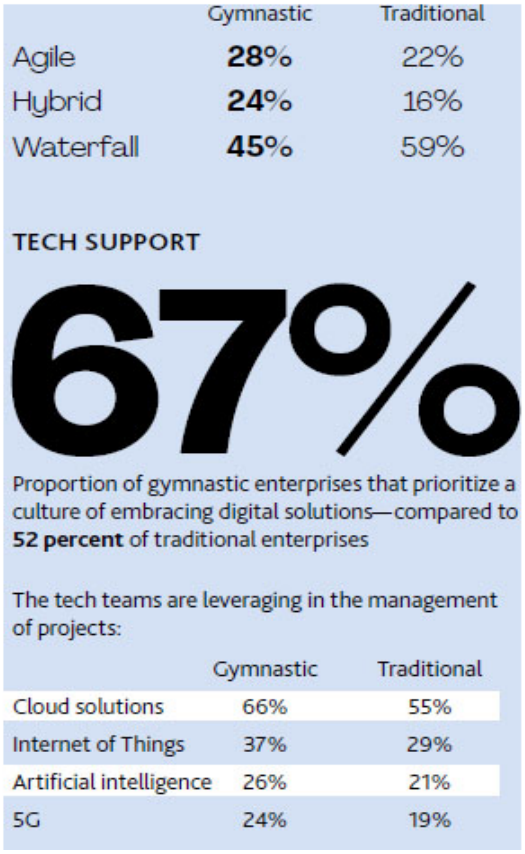
	Gymnastic	Traditional
Met original goals/business intent	75%	72%
Completed within original budget	64%	59%
Completed on time	58%	52%
Experience scope creep	33%	37%
Failed project, budget lost	36%	37%
Deemed failures	11%	13%

Fuente: Project Management Institute (2022c).

- Seleccionan mejor las formas de trabajar.

Figura 7

Método de trabajo de empresas ágiles



	Gymnastic	Traditional
Agile	**28%**	22%
Hybrid	**24%**	16%
Waterfall	**45%**	59%

TECH SUPPORT

67%

Proportion of gymnastic enterprises that prioritize a culture of embracing digital solutions—compared to **52 percent** of traditional enterprises

The tech teams are leveraging in the management of projects:

	Gymnastic	Traditional
Cloud solutions	66%	55%
Internet of Things	37%	29%
Artificial intelligence	26%	21%
5G	24%	19%

POWER TO PEOPLE

Gymnastic enterprises are more likely to prioritize power skills throughout the organization.

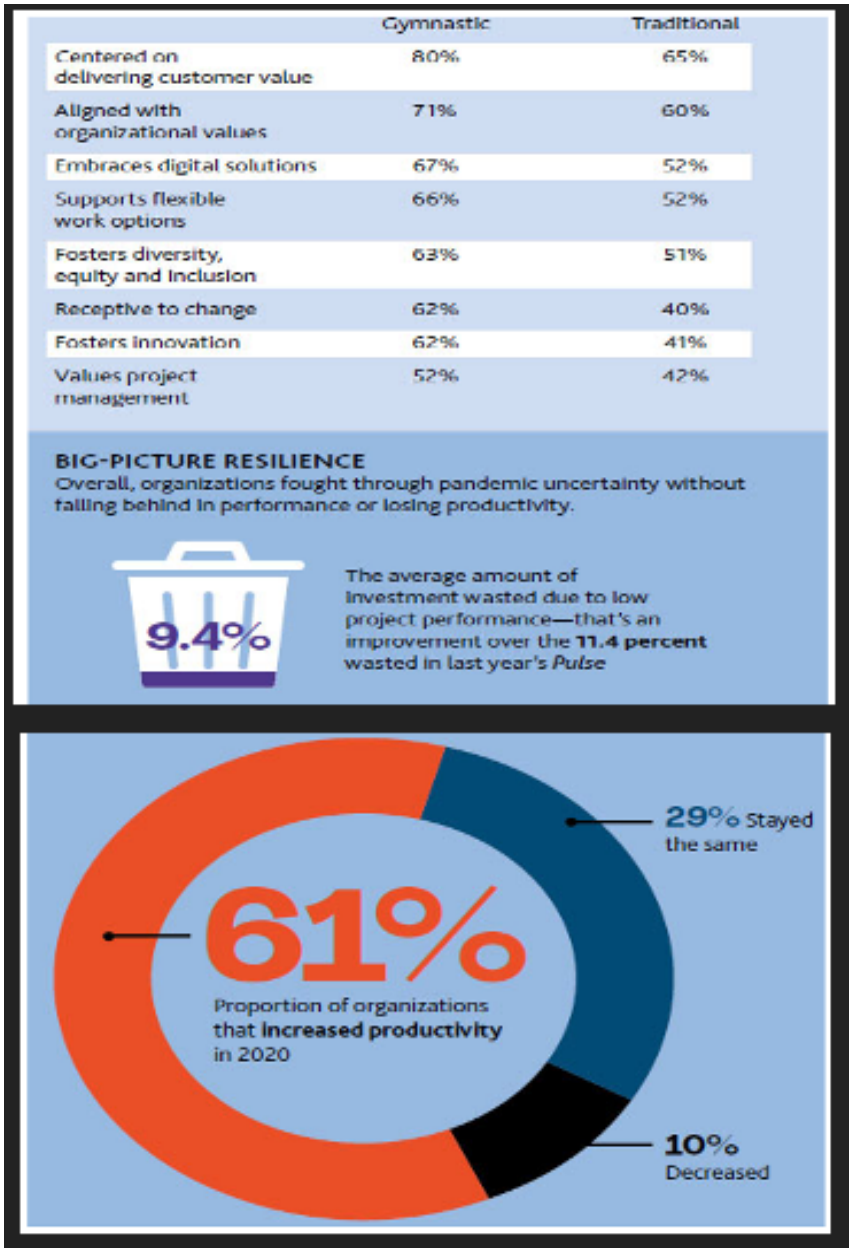
	Gymnastic	Traditional
Collaborative leadership	54%	42%
Adaptability	50%	40%
Innovative mindset	46%	32%
Empathy	21%	17%

Fuente: Project Management Institute (2022c).

- Fomentan la cultura organizacional y cumplen una misión crítica.

Figura 8

Misión crítica de una empresa ágil



Fuente: Project Management Institute (2022c).

5.2.2.4 El próximo despertar ágil. Cuatro líderes ágiles analizan nuevas posibilidades en un mundo de cambios repentinos (2022)

Autor: Project Management Institute.

Las empresas en la actualidad están apostando por implementar dentro de sus organizaciones la cultura organizacional basada en cambios tecnológicos y digitales, adoptando metodologías ágiles con el propósito de optimizar los costos y los tiempos de trabajo.

Las metodologías ágiles se deben entender como un tipo de estrategia que opera de manera integral impulsando a las organizaciones a gestionar los proyectos con una mayor rapidez y flexibilidad.

> *Disciplined Agile* permite que los equipos agilicen el tamaño para adaptarse a sus necesidades: tomar prestado de una amplia gama de enfoques ágiles, como SCRUM, SAFe, KANBAN y otros, y adaptarlos a un híbrido que tenga más sentido para la situación actual. En otras palabras, los equipos ya no necesitan sentirse atrapados en una sola forma de trabajar. (párr. 3)

Producto de la pandemia a nivel mundial las empresas se vieron forzadas a cambiar su estilo de trabajo, en donde algunas adoptaron metodologías ágiles las cuales dieron frutos positivos para ellas, como indican algunas de ellas a nivel mundial:

- **Raczka:** Seguro. Agile nos ayudó a adaptarnos a la pandemia casi de inmediato. Decidimos abandonar algunos proyectos y, en cambio, asumir nuevos proyectos que ayudarían a nuestros clientes a tener una banca segura y confiable. Las iteraciones cortas y los ciclos de retroalimentación de Agile nos ayudaron a saber que estábamos haciendo lo correcto.
- **Molina:** Agile ya formaba parte del ADN de mi organización. Trabajamos en equipos pequeños que se enfocan en productos mínimos viables para brindarles a nuestros clientes mejores soluciones más rápidamente. Pero definitivamente aumentamos nuestro uso de Agile durante la pandemia.
- **Sudame:** Nuestro uso también aumentó. Nuestra empresa desarrolla aplicaciones de *software* para diferentes clientes, como las aseguradoras, que es el área comercial que dirijo, y nuestros clientes estaban bajo mucho estrés para optimizar el tiempo de lanzamiento y acelerar sus transformaciones digitales. Y para eso, necesitaban ágiles. (párr. 5-7)

La metodología Agile ha sido útil para las empresas como mencionan los expertos en el artículo, esto porque se evidencia que el sistema es bastante flexible, que se adapta a las necesidades individuales; SCRUM o Scaled Agile Framework (SAFe) son citados como ejemplo. Además, se indica que Agile “toma grandes ideas de una amplia gama de fuentes (Lean, SCRUM, KANBAN, tradicional) y las pone en contexto: esta es la situación en la que se encuentra y estas son las opciones disponibles” (párr. 16).

Se les hizo la consulta a los expertos sobre cómo Agile ha ayudado a las empresas a mantener la resiliencia y convertirse en empresas gimnásticas, a lo que indicaron:

- **Ambler:** Los equipos que usan técnicas tradicionales pueden sobrevalorar estar a tiempo y dentro del presupuesto, lo que puede afectar las decisiones tomadas sobre la calidad o el alcance, o cumplen con especificaciones que ya no son realmente lo que el cliente quiere. Los equipos ágiles ayudan a las empresas a centrarse en los resultados y el valor para el cliente.
- **Molina:** Con los enfoques tradicionales, puedes dedicar mucho tiempo solo a diseñar un producto, por lo que, cuando termines, es posible que no sea lo que el cliente quiere. Con Agile, desarrollamos el producto junto con la persona que lo quiere, el dueño del producto. Nos enfocamos en los resultados, no solo en el plan o el cronograma.
- **Raczka:** A través de Agile, nos volvimos más capaces de adoptar e implementar cambios a un ritmo mucho más rápido. (párrs. 20-22)

Como conclusión del artículo, se puede mencionar que con Agile las empresas reducen su ciclo de comentarios y logran aumentar la colaboración, visibilidad y comunicación, aspectos clave para abordar el riesgo. Las personas y empresas agilistas no pueden ocultar lo que están haciendo, por lo que les resulta fácil detectar cualquier problema.

5.2.2.5 Síntesis

Actualmente, la gestión de proyectos se está desarrollando como una carrera profesional en la cual las personas se pueden certificar a nivel mundial, y se crean habilidades como la colaboración, la investigación y la educación de las personas profesionales; además con la certificación se asegura a las organizaciones y a las personas una forma de trabajar más inteligente en la nueva realidad de un mundo más dinámico y en constante cambio.

En cualquier proyecto que se lleve a cabo se requerirá de personal que lidere. Estos profesionales deben estar bien capacitados para hacer frente a las demandas, y deben cultivar relaciones interpersonales en la organización. Es por ello por lo que dicho profesional es una persona estratégica para el éxito de los objetivos de una organización o de un proyecto en sí.

Debido a la globalización, las empresas que quieran permanecer deben adoptar nuevas formas de trabajo, y en el mercado destacan las empresas gimnásticas, las cuales adoptan medidas y formas de trabajar más flexibles que les están permitiendo alcanzar ventajas competitivas ante las otras organizaciones.

Por medio de la metodología Agile se puede mejorar el desarrollo de proyectos, los cuales requieren de rapidez y flexibilidad para cubrir las necesidades del cliente, sin dejar de lado los resultados esperados. La metodología Agile se diferencia del sistema tradicional en la forma de gestionar los proyectos, pues no requiere definir al inicio de un proyecto la totalidad del alcance. Las empresas que adoptan la metodología Agile poseen ventajas competitivas antes las demás.

5.2.3 Tecnologías de automatización de proyectos

5.2.3.1 ¿Cómo ayuda DevOps a los desarrolladores? (2020)

Autor: Tarun Saini.

El presente artículo tiene como objetivo señalar los principales puntos en los que DevOps les facilita el trabajo a los desarrolladores, al permitir realizar actividades a mayor velocidad.

Los puntos principales que se mencionan en el artículo sobre DevOps son:

- **Ayuda a los desarrolladores y programadores a pensar y usar "Automatización"**: Con DevOps es posible tener una perspectiva mucho más amplia que se obtiene al hacer clic en "Confirmar". Hay varias configuraciones en el entorno donde se ejecuta el código. Cuando algo parece no funcionar bien en el sistema de CI, el desarrollador puede utilizar DevOps para depurarlo y reproducir el contexto que se encontró defectuoso. DevOps ayuda a ahorrar tiempo para desarrollar el entorno y reparar cualquier defecto. El uso de DevOps permite al desarrollador escribir scripts que pueden crear automáticamente un entorno gemelo que se asemeje a un servidor de CI. El desarrollador no necesita necesariamente un gran conocimiento para este.
- **Pruebas de emisión de integración continua (CI):** En DevOps, al depurar antes de confirmar el código, el desarrollador debe tener en cuenta las discrepancias entre sus propias máquinas y los servidores de CI. La carga

adicional provoca retrasos a medida que varios procesos pasan al segundo plano y salen de él. Esta forma de variación temporal revela vulnerabilidades en códigos no probados. Los desarrolladores ejecutan experimentos a través de un proxy para replicar estos retrasos de tiempo particulares, agregando artificialmente retrasos cercanos a los que enfrenta el servidor CI. Esto revela más situaciones de IC de la vida real y fomenta la escritura de experimentos rigurosos.

- **Mejora el rendimiento:** El mayor rendimiento se suma a la velocidad y evita la operación complicada. Las actividades de DevOps se pueden optimizar. El proceso de verificación del código está automatizado por servidores de integración continua, lo que minimiza el esfuerzo manual necesario. Los ingenieros de *software* se concentrarán entonces en realizar tareas que no se pueden programar.
- **Interacción mejorada:** DevOps fortalece la cultura de creación de aplicaciones. Ya no se trata de “intentar dar la vuelta” a la aplicación y mirar para ver qué sucede. Los procesos no tienen que esperar hasta que otro equipo pueda resolver un problema y resolverlo. Como todas las personas trabajan para un propósito común, el proceso parece cada vez más fluido.
- **La invención más rápida, ciclos de producción rápidos:** Las aplicaciones se pueden usar incluso más rápido para equipos mixtos de desarrollo y operación. Esto es fundamental porque las empresas pueden innovar más rápidamente que sus rivales.

Se desprende del artículo que, si una empresa u organización adopta la cultura de DevOps junto con las prácticas y las herramientas de DevOps, estos equipos adquirirán la capacidad de responder y mejorar las necesidades de los clientes, aumentar la confianza en las aplicaciones creadas y alcanzar los objetivos en menos tiempo.

5.2.3.2 ¿Qué es Jenkins? Jenkis para la integración continua (2021)

Autor: Saurabh.

Para DevOps, tener una integración continua resulta un aspecto importante, en el que las diferentes etapas se integran. Jenkins se define en el artículo como:

> … una herramienta de automatización de código abierto escrita en Java con complementos creados para fines de integración continua. Jenkins se utiliza para crear y probar sus proyectos de *software* continuamente, lo que facilita a los desarrolladores la integración de cambios en el proyecto y facilita a los usuarios obtener una nueva compilación. También le permite entregar su *software* continuamente al integrarse con una gran cantidad de tecnologías de prueba e implementación. (párr. 2)

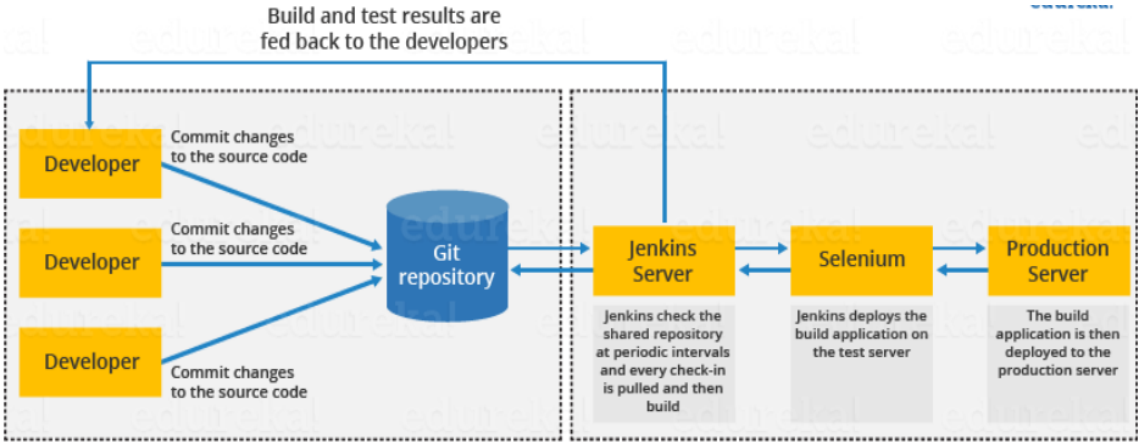
Las organizaciones que trabajan con Jenkins logran acelerar los procesos de desarrollo de *software*, en donde se integran procesos en el ciclo de vida incluyendo la compilación, documentación, prueba, paquete, etapa, implementación, análisis estático, entre otros.

Cuando se menciona Jenkins, se hace referencia a la integración continua, la cual se puede definir como “una práctica de desarrollo en la que los desarrolladores deben realizar cambios en el código fuente en un repositorio compartido varias veces al día o con mayor frecuencia” (párr. 12).

Para comprender la integración continua de Jenkins, en el artículo se indica de manera gráfica dicho proceso:

Figura 9

Diagrama de flujo genérico de integración continua



Fuente: Saurabh (2021).

- El desarrollador envía un código al repositorio de código fuente. Al mismo tiempo, el servidor Jenkins verifica el repositorio a intervalos regulares en busca de cambios.

- Luego de que se produzca una confirmación, el servidor Jenkins detecta los cambios ocurridos en el repositorio del código fuente. Jenkins extraerá esos cambios y comenzará a preparar una nueva compilación.
- Si la compilación falla, se notificará al equipo en cuestión.
- Si la compilación es exitosa, entonces Jenkins implementa la compilación en el servidor de prueba.
- Después de la prueba, Jenkins genera una retroalimentación y luego notifica a los desarrolladores sobre los resultados de la compilación y la prueba.
- Jenkins continuará verificando el repositorio del código fuente para ver si hay cambios realizados en el código fuente y todo el proceso continúa repitiéndose. (párrs. 16-20)

5.2.3.3 Arquitectura Docker y sus componentes para principiantes (2019)

Autor: Avi Sysadmin.

El artículo explica de forma sencilla la arquitectura Docker en DevOps y sus principales componentes. Años atrás se creaban máquinas virtuales, cada una de ellas con un sistema operativo que requería de mucho espacio, lo cual lo hacía muy pesado. Actualmente, al crear una máquina se le integra un solo sistema operativo, en el cual los recursos son compartidos, lo que lo hace más liviano y permite iniciarlo en segundos.

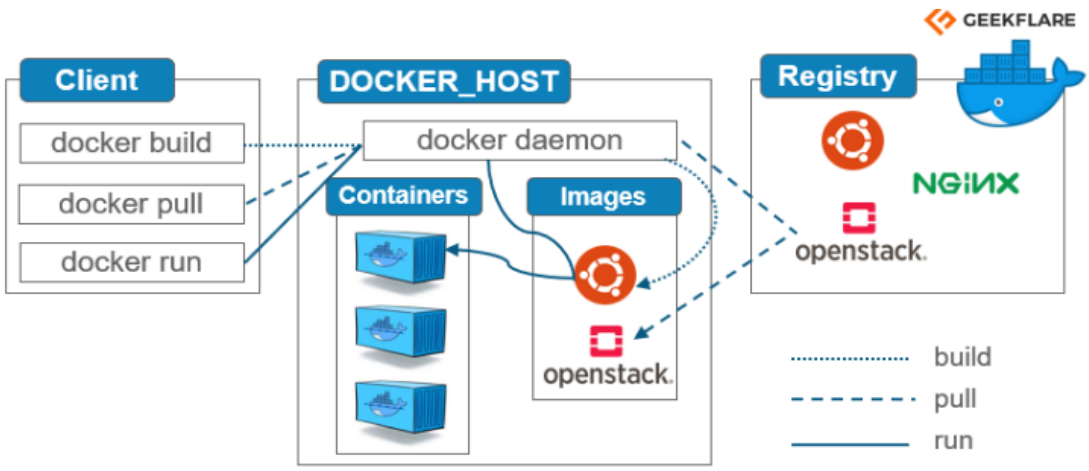
Entre los componentes de un sistema Docker se encuentran los siguientes:

- **Servidor:** Es el demonio Docker llamado Dockerd. Puede crear y administrar imágenes de la ventana acoplable. Contenedores, redes, etc.
- **Rest API:** Se utiliza para indicar al demonio Docker qué hacer.
- **Interfaz de línea de comandos (CLI):** Es un cliente que se utiliza para ingresar comandos de la ventana acoplable. (párrs. 8-9)

En la figura 10 se muestra un diagrama de la arquitectura Docker:

Figura 10

Diagrama arquitectura Docker



Fuente: Avi Sysadmin.

El autor del artículo explica cada una de las partes de la figura anterior de la siguiente manera:

- Cliente Docker: Los usuarios de Docker pueden interactuar con Docker a través de un cliente. Cuando se ejecuta cualquier comando de Docker, el

cliente los envía al demonio de Dockerd, que los ejecuta. Los comandos de Docker utilizan la API de Docker. El cliente de Docker puede comunicarse con más de un demonio.

- Registros de Docker: Es la ubicación donde se almacenan las imágenes de Docker. Puede ser un registro de ventana acoplable público o un registro de ventana acoplable privado. Docker Hub es el lugar predeterminado de las imágenes de la ventana acoplable, el registro público de sus tiendas. También puede crear y ejecutar su propio registro privado. Cuando ejecuta los comandos *docker pull* o *docker run*, la imagen de la ventana acoplable requerida se extrae del registro configurado. Cuando ejecuta el comando *docker push*, la imagen de la ventana acoplable se almacena en el registro configurado.
- Objetos acoplables: Cuando trabaja con Docker, usa imágenes, contenedores, volúmenes, redes; todos estos son objetos de Docker. (párrs. 10-12)

Se concluye del artículo que Docker utiliza los contenedores en máquinas virtuales muy livianas y modulares, obteniendo una flexibilidad necesaria para crearlos, copiarlos, implementarlos y trasladarlos de un entorno a otro, permitiendo optimizar sus aplicaciones en la nube.

5.2.3.4 Síntesis

Diseñar aplicaciones a la medida es un tema que se ha vuelto crítico, pues es importante brindar la mejor experiencia posible sin inconvenientes. DevOps

vincula los servicios a un determinado subproceso de DevOps, que no es un nuevo programa o sistema de aprendizaje por construir. Los equipos que adoptan DevOps mejoran el rendimiento y se pueden crear productos de mayor calidad en menor tiempo, aumentando la satisfacción de los clientes.

Adoptar prácticas de DevOps logra automatizar y optimizar los procesos con tecnología dentro de la cultura organizacional, involucrando a las personas que participan en ella. Como desafío de la cultura DevOps se encuentra que requiere realizar cambios profundos en la forma en que las personas trabajan. No obstante, si se logra un compromiso a nivel organizacional de cultura de DevOps, se pueden crear entornos que facilitan el desarrollo de equipos de alto rendimiento.

Jenkins se constituye como una herramienta de automatización utilizada para compilar y probar los proyectos de *software* de forma continua, facilitando a los desarrolladores la integración de cambios en un proyecto y entregar nuevas versiones a los usuarios.

Otro sistema DevOps con el cual se pueden desarrollar sistemas es el conocido Docker, por medio del cual el cliente puede “hablar” con el servidor a través de un API con el cual gestiona el ciclo de los contenedores, y con ello lograr construirlos, distribuirlos y ejecutarlos, en donde estos contenedores se encuentran aislados de todo el sistema, y a nivel de la red, así cada contenedor tendrá su propia *stack de net* y sus propios puertos.

Los desarrolladores pueden encontrar el conocido GitHub, el cual es una plataforma de organización y gestión de proyectos basada en la nube,

en que se incorporan funciones de control de versiones de Git. Esto quiere decir que todos los usuarios de GitHub pueden gestionar y rastrear los cambios realizados en el código fuente en tiempo real.

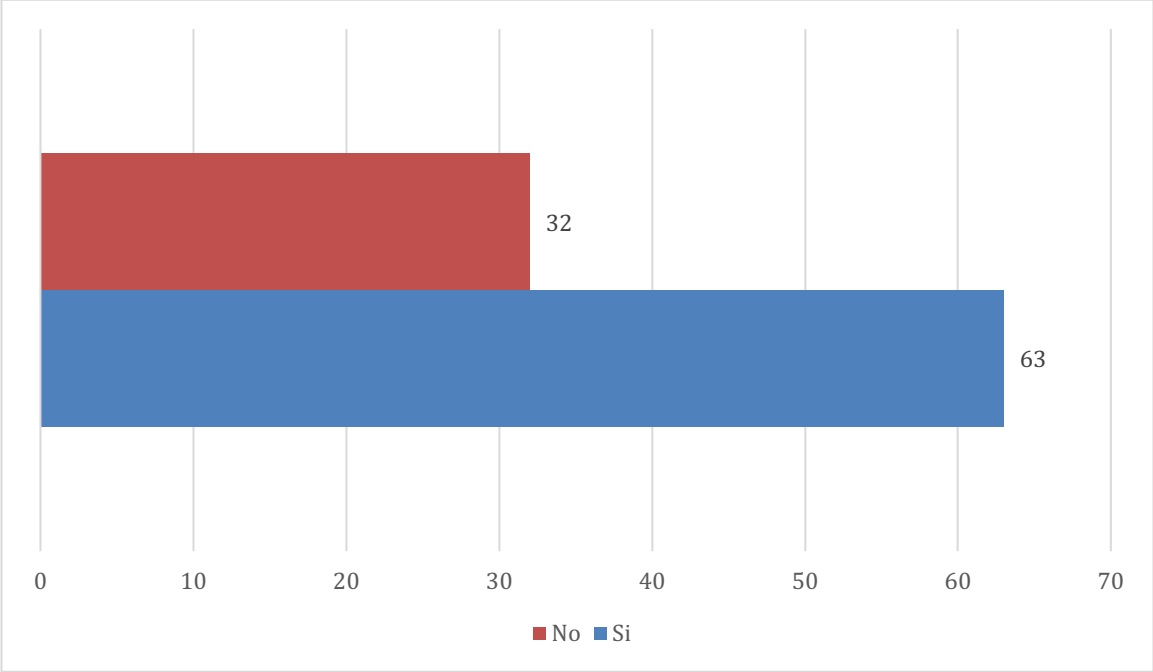
5.3 Identificar las prácticas de desarrollo existentes y sus características en las empresas del territorio costarricense por medio de encuestas a profesionales del área, así como identificar las dificultades que experimentaron al implementar estas prácticas en sus proyectos de desarrollo

Las entrevistas y encuestas aplicadas permiten la recolección de datos relacionados con el tema “Análisis de las tecnologías de desarrollo *cross-platform* existentes en el mercado y propuesta de un proceso de desarrollo automatizado aplicable a una pyme enfocada en el desarrollo de aplicaciones multiplataforma”. La aplicación del instrumento se realizó a desarrolladores, administradores y arquitectos; las respuestas a las preguntas cuestionadas se relacionan con la experiencia y conocimiento de cada participante.

En el caso de los desarrolladores, la información extraída demuestra que, con respecto a las prácticas DevOps, el uso de la metodología se relaciona con la utilización previa en proyectos y según la preferencia del desarrollador. El objetivo principal de las herramientas utilizadas es acelerar el proceso de desarrollo de la empresa.

Figura 11

Gráfico de número de personas con conocimiento sobre aplicaciones móviles informáticas creadas en Costa Rica. Empresa 3Pillar Global de C. R., noviembre de 2017

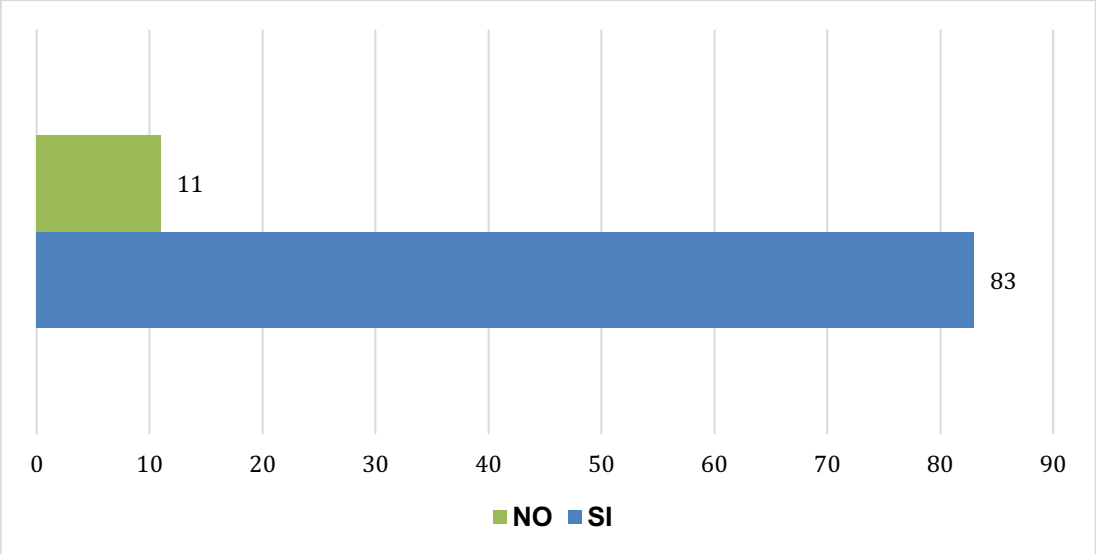


Fuente: Elaboración propia.

Según el gráfico anterior, el instrumento demuestra que existe un conocimiento generalizado de aplicaciones desarrolladas en Costa Rica que sean utilizadas tanto por usuarios nacionales como internacionales, lo que se traduce en un total de 63 respuestas “Sí” y 32 respuestas “No”.

Figura 12

Gráfico numérico de uso de tecnologías cross-platform por parte de empresas pymes. Empresa 3Pillar Global de C. R., noviembre de 2017

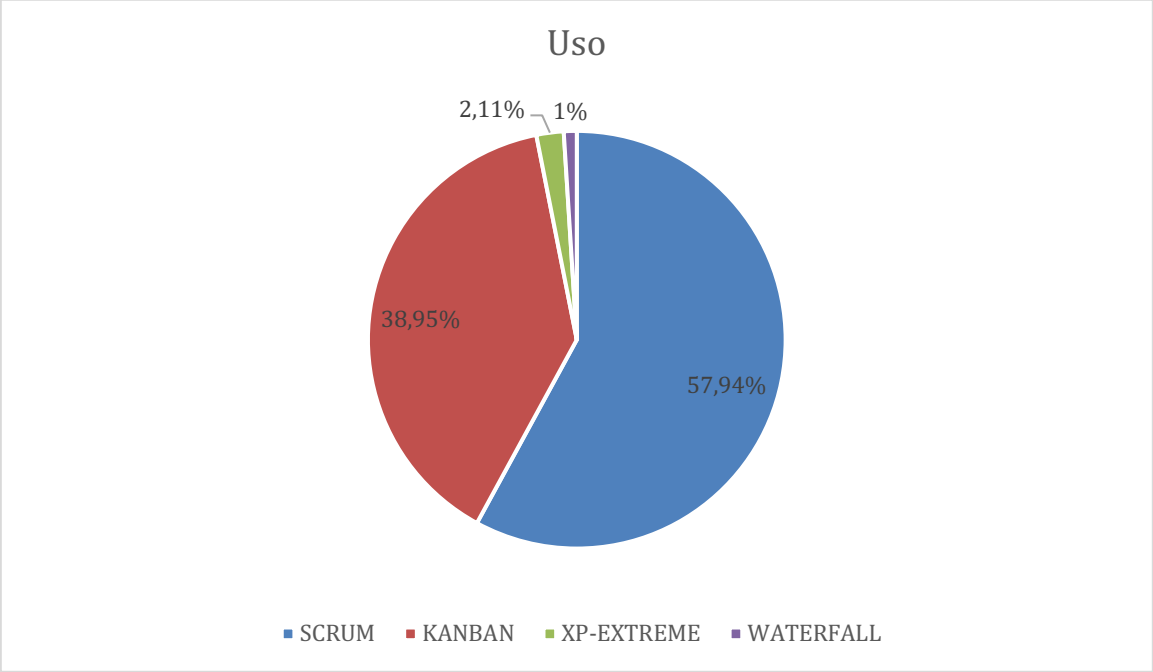


Fuente: Elaboración propia.

Al analizar el gráfico anterior, se observa que el uso de tecnologías *cross-platform* en el desarrollo de aplicaciones por parte de las empresas pyme se considera fundamental para agilizar el proceso de desarrollo. Este enfoque representa un desafío significativo al crear una aplicación. Los datos reflejan un total de 83 respuestas afirmativas y 11 respuestas negativas en relación con esta tendencia.

Figura 13

Gráfico de porcentaje. Metodología utilizada por desarrolladores, arquitectos y administradores de proyectos en la Empresa 3Pillar Global de C. R., noviembre de 2017



Fuente: Elaboración propia.

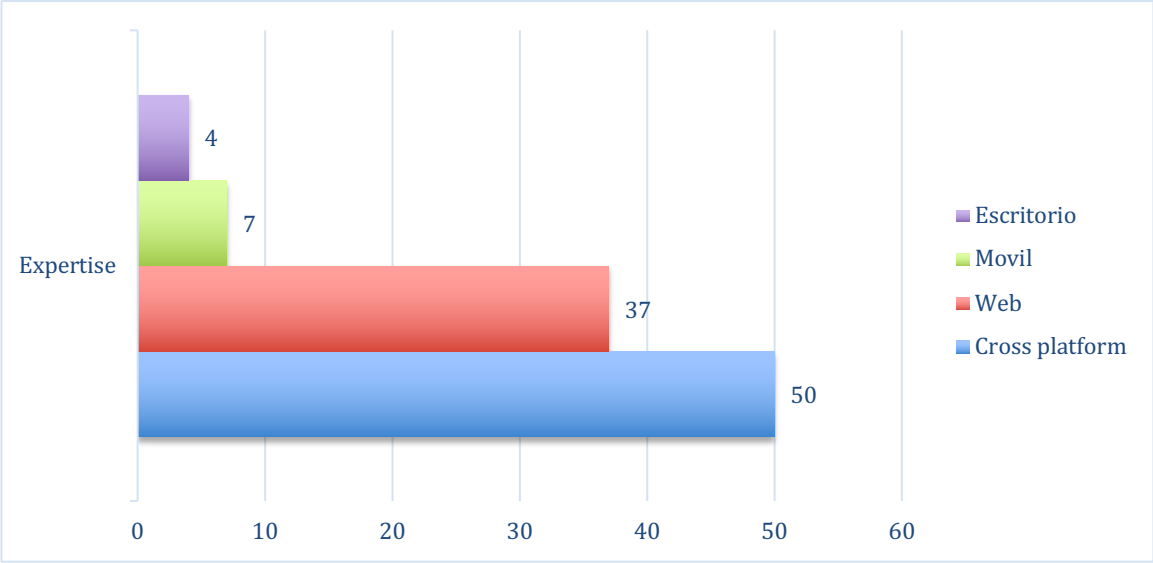
El gráfico circular anterior presenta información recopilada por el instrumento en el caso de los desarrolladores. Según los datos, el 57,94% utiliza SCRUM; el 38,95% prefiere KANBAN; el 2,11% se inclina por XP-EXTREME y el 1% restante opta por Waterfall. Estos datos reflejan que SCRUM es la metodología más destacada entre las opciones, mientras que Waterfall es la menos utilizada.

5.3.1 Análisis del instrumento aplicado a los desarrolladores

Según la aplicación del instrumento se extrae la información expuesta por parte de los administradores**,** dichos datos recopilados en el instrumento aplicado demuestran que los desarrolladores presentan experiencia variada en desarrollo móvil, web, *cross-platform* (híbridas) o de escritorio, por lo tanto, se presenta una diversificación en el desarrollo, con una mayor tendencia al desarrollo *cross-platform.*

Figura 14

Gráfico numérico de expertise tecnológica de los desarrolladores en la Empresa 3Pillar Global de C. R., noviembre de 2017



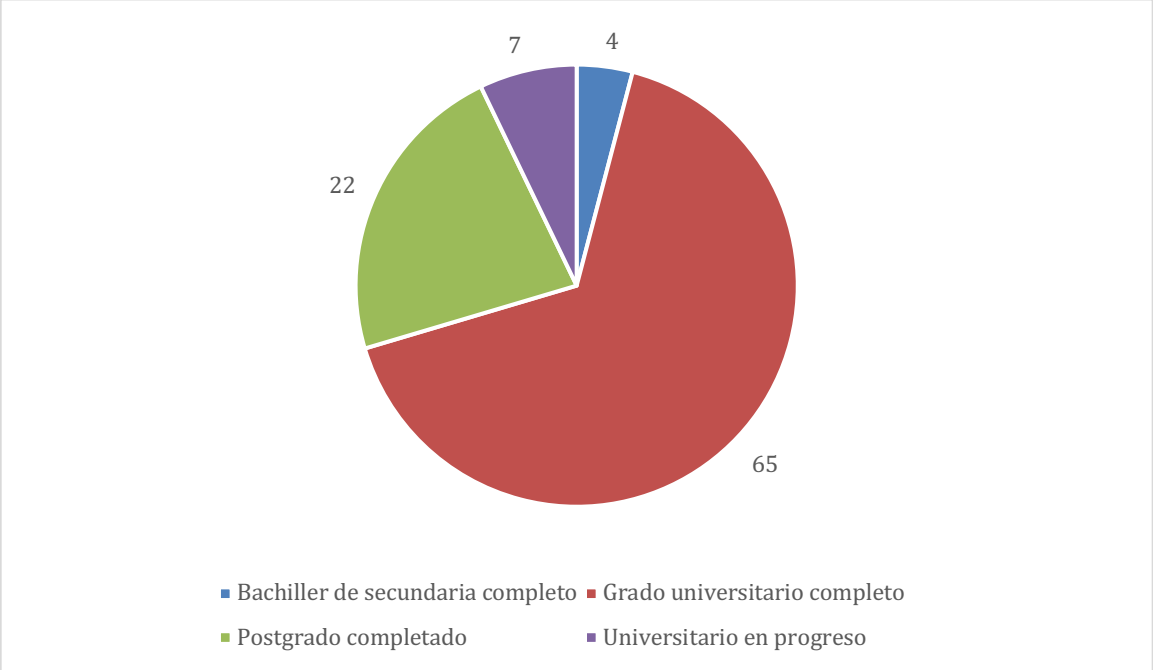
Fuente: Elaboración propia.

Las variaciones en el nivel de especialización dependen de la experiencia desarrollada o del ámbito de conocimiento que presentan los desarrolladores. Con

respecto a la experiencia profesional y al grado académico, se demostró una mayoría de desarrolladores con un grado académico universitario completado, así como una gran cantidad con un postgrado universitario (licenciatura o maestría) completado en su totalidad. Se considera que, debido al tipo de tareas y a la importancia de estas, se requiere un nivel universitario más alto.

Figura 15

Gráfico numérico: nivel académico de los desarrolladores de software en la Empresa 3Pillar Global de C. R., noviembre de 2017



Fuente: Elaboración propia.

En general, los desarrolladores consideran relevante la creación de aplicaciones amigables, de fácil comprensión y utilización, ya que proporcionan al usuario mayor agilidad en su ejecución. Las tecnologías de desarrollo *cross-*

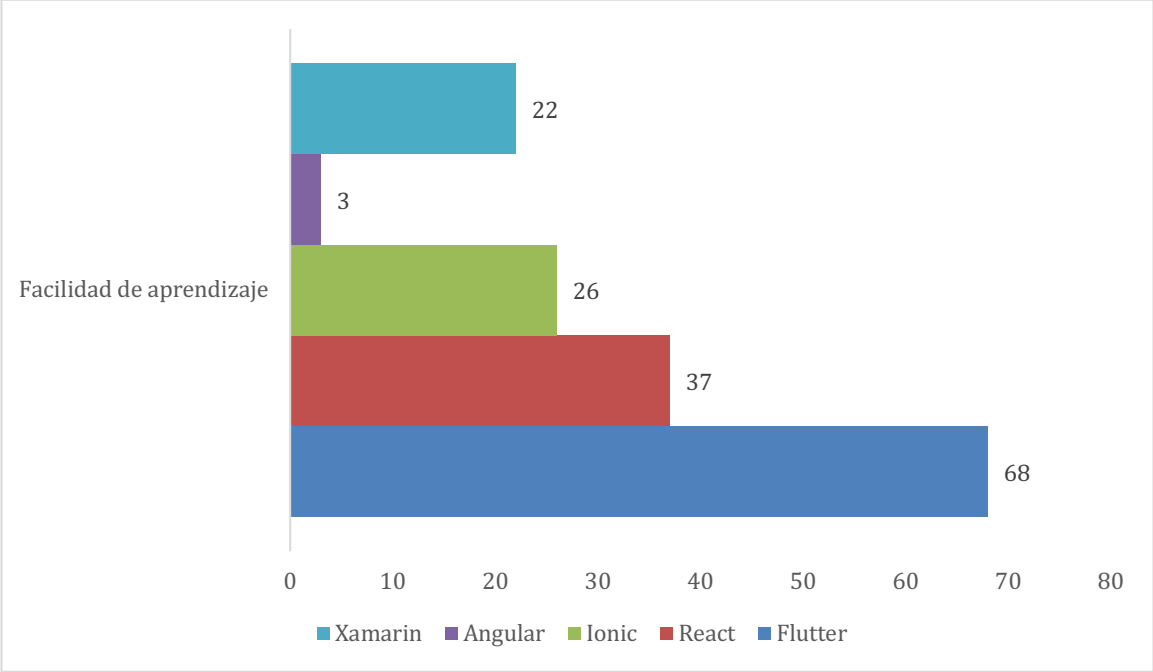
platform utilizadas por los desarrolladores, según la recopilación de información, son:

- Xamarin
- React Native
- Ionic
- Flutter
- Cordova
- Angular
- Nativescript

De las tecnologías mencionadas anteriormente, los desarrolladores consideran que para ellos fue más fácil aprender las tecnologías de *desarrollo cross-platform* Flutter y React.

Figura 16

Gráfico numérico: cantidad de votos, lenguajes de desarrollo más fáciles de aprender para los desarrolladores de la Empresa 3Pillar Global de C. R., noviembre de 2017



Fuente: Elaboración propia.

Por lo tanto, es importante la inclusión en los proyectos de desarrollo móvil de tecnologías *cross-platform* en donde intervenga una combinación de ambas tecnologías.

En cuanto al uso de herramientas de automatización de procesos, las más utilizadas según la experiencia de los desarrolladores son:

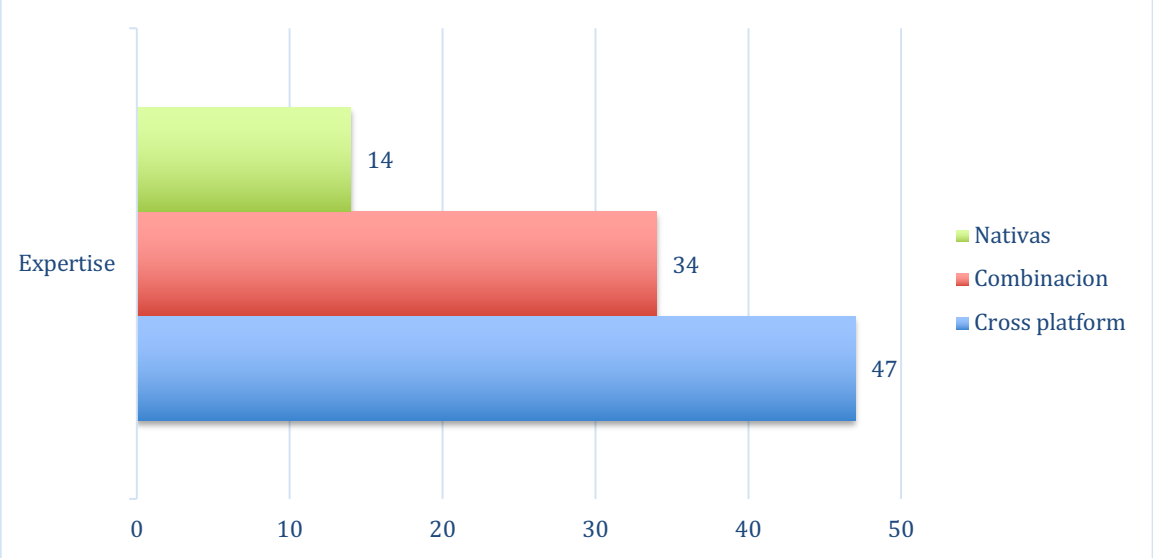
- Slack
- Jenkins

- GitHub Actions
- Docker

Según el punto de vista de los desarrolladores, las aplicaciones *cross-platform* tienen una mayor presencia en el mercado en comparación con las tecnologías completamente nativas o la combinación de tecnologías.

Figura 17

Gráfico numérico de presencia de tecnologías de desarrollo de aplicaciones según los desarrolladores de la Empresa 3Pillar Global de C. R., noviembre de 2017



Fuente: Elaboración propia.

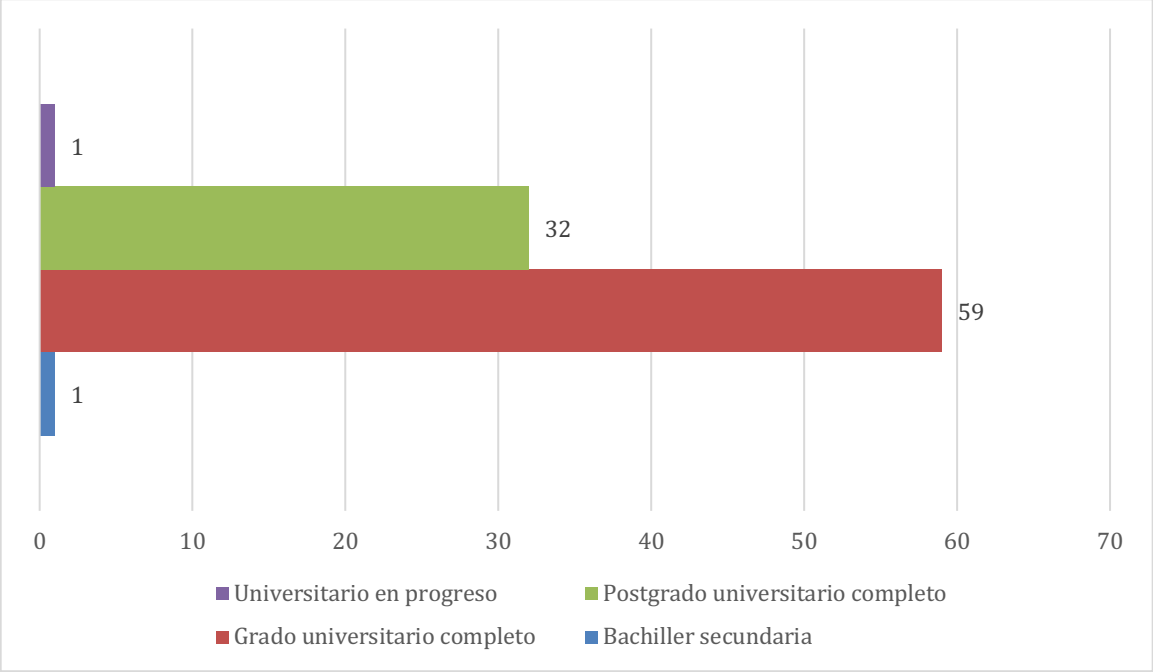
En cuanto a las tecnologías React y Flutter, estas demuestran un mayor uso en el desarrollo cross-platform de proyectos. Según el criterio de los desarrolladores, es importante que las empresas PYME opten por desarrollar tecnologías *cross-platform* en lugar de tecnologías nativas.

5.3.2 Análisis del instrumento aplicado a los arquitectos

El rango de experiencia presentado oscila entre 1-5 años de conocimientos varios por parte de los arquitectos. Dentro de su nivel de formación académica se indican los siguientes:

Figura 18

Gráfico numérico: nivel académico de los arquitectos de software en la Empresa 3Pillar Global de C. R., noviembre de 2017



Fuente: Elaboración propia.

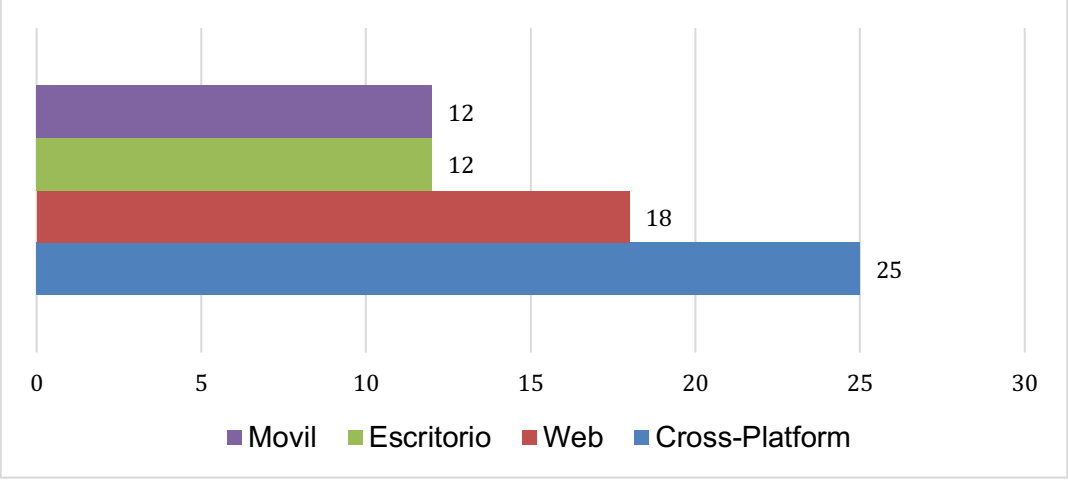
Los niveles expuestos se presentan como mayoría entre los sujetos entrevistados; solo se presenta un caso en donde el arquitecto cuenta con “Bachiller

de secundaria completado". Todos los sujetos han tenido experiencia creando la arquitectura para aplicaciones móviles o híbridas.

En lo que respecta a la experiencia en el desarrollo de aplicaciones *cross-platform*, se evidencia una sólida experiencia en el ámbito de las aplicaciones híbridas, como se muestra en el siguiente gráfico.

Figura 19

Gráfico numérico de experiencia de los arquitectos en uso de plataformas de desarrollo de aplicaciones Empresa 3Pillar Global de C. R., noviembre de 2017



Fuente: Elaboración propia.

Se considera que los resultados presentados a continuación son bastante realistas con el mercado tecnológico actual, basándose en la experiencia mencionada por los arquitectos de *software*.

Los arquitectos consideran importante que las empresas pyme opten por desarrollar con tecnologías de desarrollo *cross-platform* en lugar de desarrollar con tecnologías nativas, opinión que comparten los desarrolladores.

También consideran que el mayor reto al implementar una aplicación con tecnologías *cross-platform* son los procesos de calidad más rigurosos y la documentación del mismo proyecto. Es importante mencionar que algunos arquitectos mencionaron que estos retos también se presentan al desarrollar una aplicación con tecnologías nativas.

Tabla 13

Retos al momento de crear una aplicación con tecnologías cross-platform según lo apreciado por los arquitectos de la Empresa 3Pillar Global de C. R., noviembre de 2017

Reto mayor al momento de implementar una aplicación hibrida	**Conteo**
Ajustarse a las diferencias entre plataformas	1
Implementación en tiendas en línea, Creación de pruebas automatizadas, Procesos de QA rigurosos en el proceso de desarrollo	1
Procesos de QA rigurosos en el proceso de desarrollo	1
Creación de pruebas automatizadas	3
Creación de pruebas automatizadas, Training para devs que piensan en las apps como nativas	3

Documentación del proyecto, Creación de pruebas automatizadas, Procesos de QA rigurosos en el proceso de desarrollo	3
Documentación del proyecto, Procesos de QA rigurosos en el proceso de desarrollo	38
Procesos de QA rigurosos en el proceso de desarrollo, Tiempo de desarrollo	43
Total	**93**

Fuente: Elaboración propia.

Se considera que la mejor arquitectura de proyecto que podría aplicarse en una aplicación desarrollada con tecnologías *cross-platform* es a través del uso de:

- Arquitectura de componentes
- Arquitecturas de capas
- Arquitectura de eventos
- Dependencia del proyecto
- Dependencia de las métricas del proyecto

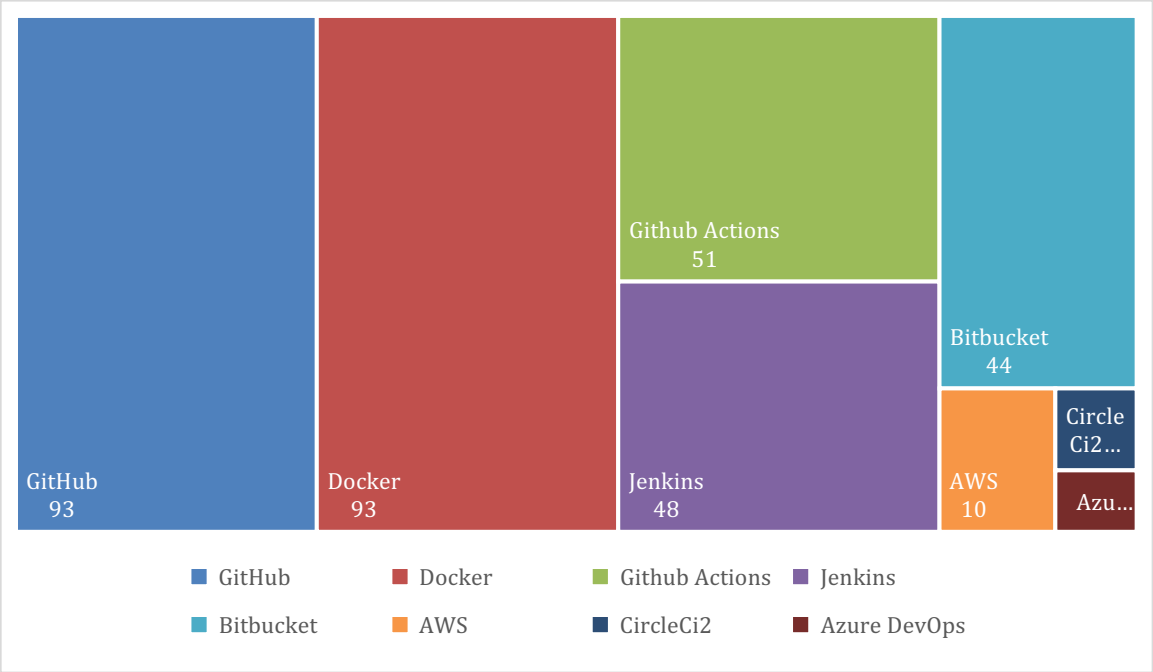
También, según los arquitectos, consideran que al desarrollar un proyecto, los equipos deben dar prioridad a ofrecer una buena experiencia de usuario. Indican que es recomendable contar con personas dedicadas a la experiencia del usuario y utilizar herramientas de automatización de procesos para ejecutar tareas repetitivas en los proyectos de forma automática.

Según los arquitectos, es una práctica bastante común utilizar herramientas para automatizar el proceso de desarrollo tanto como sea posible, especialmente

las tareas repetitivas como compilaciones, implementaciones y pruebas unitarias. Según la experiencia que han compartido, se han utilizado las siguientes herramientas de automatización de procesos en los proyectos de desarrollo.

Figura 20

Gráfico numérico de herramientas utilizadas para la automatización del proceso de desarrollo de software en la Empresa 3Pillar Global de C. R., noviembre de 2017



Fuente: Elaboración propia.

En el gráfico anterior se demuestra que las herramientas de mayor utilización para la automatización del proceso de desarrollo en los proyectos son GitHub, Docker y GitHub Actions.

5.3.3 Análisis del instrumento aplicado a los administradores de proyectos

La recopilación de información se efectuó aplicando el instrumento de la entrevista a los administradores.

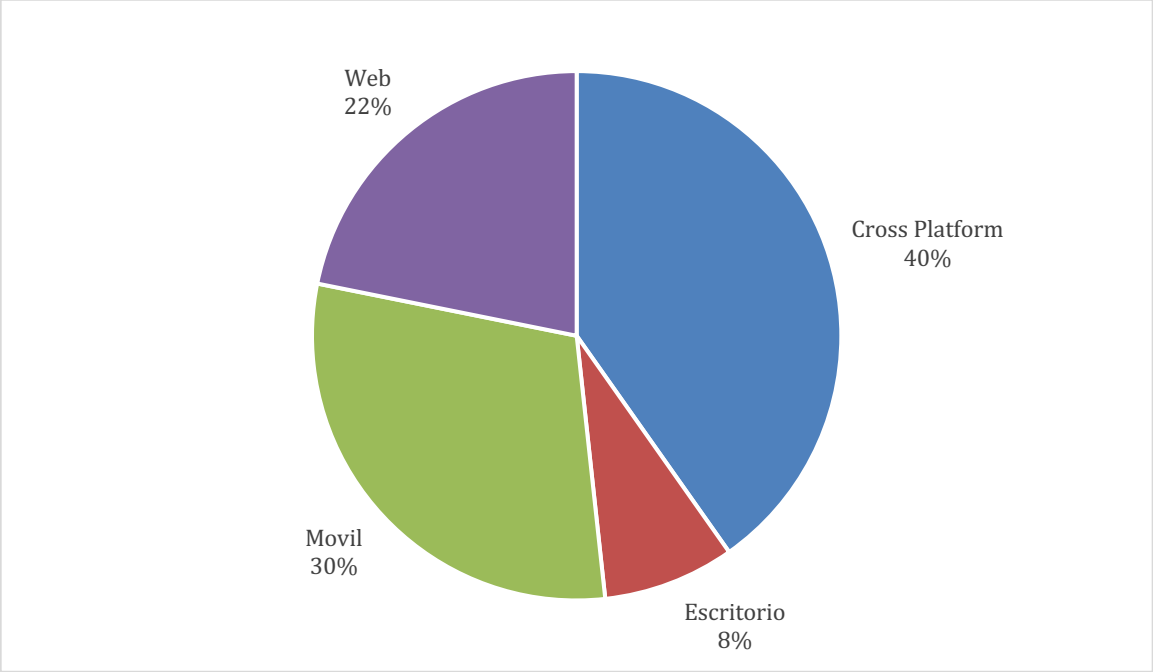
La información recopilada comparte los años de experiencia con los anteriores. El rango oscilatorio es entre 1-5 años con el nivel académico según las siguientes opciones:

- Bachiller de secundaria completado.
- Universitario en progreso.
- Grado universitario completado.
- Postgrado universitario completado.

También, con respecto a los proyectos en los que se ha desempeñado el rol de administrador de proyectos, se observa que el 40% de los administradores ha trabajado en proyectos de desarrollo híbrido y el 30% en desarrollo móvil nativo, siendo estas dos opciones las más elegidas.

Figura 21

Gráfico porcentual: experiencia en desarrollo de aplicaciones por plataforma de los administradores de proyectos de la Empresa 3Pillar Global de C. R., noviembre de 2017



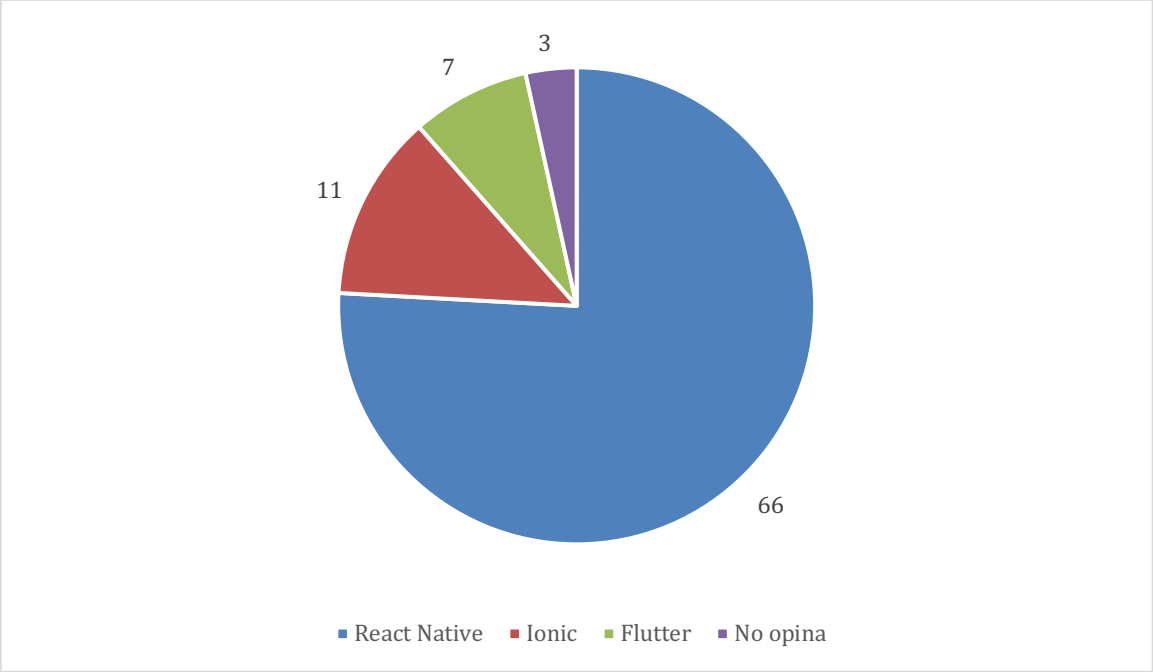
Fuente: Elaboración propia.

Para los proyectos en los que se ha ejercido como administrador de proyecto y que fueron desarrollados con tecnologías híbridas, se lograron identificar las siguientes tecnologías utilizadas en los proyectos:

- React Native
- Ionic
- Flutter
- No he trabajado ese tipo de proyectos

Figura 22

Gráfico numérico: tecnologías de desarrollo cross-platform con mayor curva de aprendizaje para los desarrolladores, según los administradores de proyectos de la Empresa 3Pillar Global de C. R., noviembre de 2017



Fuente: Elaboración propia.

Según el gráfico anterior, al utilizar estas herramientas, se considera que la curva de aprendizaje más pronunciada para los desarrolladores se encuentra en React Native, principalmente debido a su arquitectura que incluye la escritura ocasional de código de desarrollo nativo para cada plataforma.

Por otro lado, los administradores sí consideran que la empresa debe invertir tiempo, dinero y recursos en ayudar a los ingenieros de un equipo a superar la curva de aprendizaje que conlleva aprender una nueva tecnología. Según su criterio, es

importante contar internamente con la automatización de procesos de desarrollo para evitar realizar tareas repetitivas en los proyectos de forma manual.

Tabla 14

Tabla numérica: conteo de administradores de la empresa 3 Pillar Global de Costa Rica en 2017 que consideran que la empresa debe invertir en procesos de desarrollo automatizados

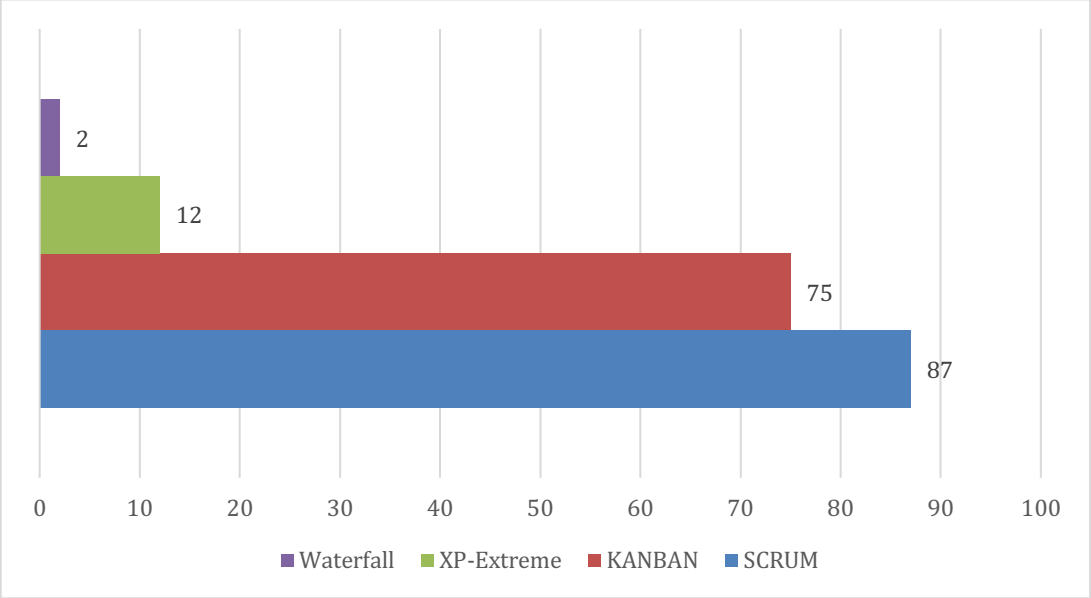
¿Considera usted que la empresa debe invertir tiempo, dinero y recursos en ayudar a los ingenieros de un equipo a superar la curva de aprendizaje que conlleva aprender una nueva tecnología?	**Conteo**
Sí	87
Total	**87**

Fuente: Elaboración propia.

De acuerdo con los datos recopilados y su análisis, se puede concluir que la mayoría de los administradores considera más relevante el uso de SCRUM y KANBAN. Este resultado se obtuvo de la misma manera en los datos recopilados anteriormente por parte de los desarrolladores y arquitectos.

Figura 23

Gráfico numérico de metodologías aplicadas por parte de los administradores de proyectos de la Empresa 3Pillar Global de C. R., noviembre de 2017



Fuente: Elaboración propia.

El gráfico anterior presenta los datos recopilados, mostrando que las principales prácticas en un proceso de desarrollo de software son Scrum y Kanban, mientras que la práctica menos utilizada es el modelo Waterfall.

CAPÍTULO VI

CONCLUSIONES Y PROPUESTA

6.1 Conclusiones

Las tecnologías más utilizadas como instrumento por los desarrolladores son React Native y Flutter. Según lo indicado por Dziuba (2021), esto se debe principalmente a las preferencias de los desarrolladores. React Native permite la comunicación entre Java Script y el puente de programación nativo, mientras que Flutter no requiere este puente de comunicación nativo al compilar a código kotlin nativo o JavaScript. Ambas son las más fáciles de utilizar, sobre todo Flutter. Se caracterizan por contar con estabilidad, rendimiento, una adecuada documentación y una comunidad activa. La principal diferencia radica en que React Native ha estado más tiempo en el mercado y ha demostrado mayor estabilidad en los productos que se conocen.

En el caso de la muestra analizada, Flutter demuestra mayor uso en el desarrollo *cross-platform* de los proyectos. Según el criterio de los desarrolladores, es importante que las empresas pymes opten por desarrollar tecnologías *cross-platform* en lugar de tecnologías nativas.

Las metodologías más utilizadas son SCRUM y KANBAN. Como se ha indicado, SCRUM es un conjunto de buenas prácticas para trabajar colaborativamente y se basa en la forma de trabajar de los equipos colaborativos. Por su parte, KANBAN es una metodología que utiliza tarjetas visuales para exponer cuellos de botella, variabilidades y desperdicios; su visualización es su gran atractivo. El estudio mostró que estas metodologías abarcan más del 90% de los casos estudiados.

En este sentido, Rehkopf (2021) señala que las metodologías KANBAN son más continuas y fluidas, mientras que SCRUM se basa en *sprints* de trabajos cortos y estructurados. Los principios son los mismos; KANBAN se centra en la eficiencia y en el flujo de trabajo, mientras que SCRUM se utiliza principalmente para entregar *softwares* en un tiempo establecido.

La comunicación es importante, al igual que la automatización de procesos, por lo que usar herramientas de automatización como GitHub Actions, Slack y otros es indispensable para que un proyecto funcione. En este contexto, GitHub reúne equipos de desarrolladores con la finalidad de escribir códigos y administrar proyectos. Según lo indicado en el sitio web Slack (2022), GitHub se puede utilizar de manera conjunta con Slack, especialmente cuando se cuenta con un plan de desarrollador, además de con las notificaciones de GitHub. Como manifiesta Leyva (2021):

> GitHub ya envía sus propias notificaciones por email cuando algo sale mal en tus acciones, pero de vez en cuando nos funciona mejor recibir esos avisos en un canal de Slack, ya sea para asegurarte de verlo o para que el resto del equipo esté enterado del problema que surgió y entre todos lo puedan resolver. (párr. 1)

En este sentido, se puede crear una configuración dentro del *workflow* de GitHub Actions, con la ayuda de *actions slack*, con el objetivo de notificar a los desarrolladores los resultados de las compilaciones.

6.2 Propuesta

A partir de la investigación de las fuentes escritas mencionadas anteriormente y las entrevistas realizadas a profesionales del área de ingeniería del *software*, se procede a presentar una propuesta de un marco de trabajo que podría ser implementado en una empresa pyme sin mayor problema.

6.2.1 Tecnología o tecnologías para desarrollo

Se recomienda utilizar Flutter o React Native para desarrollar aplicaciones móviles, de escritorio y web, ya que estas presentan ventajas considerables sobre las tecnologías de desarrollo nativo y son las más utilizadas por los desarrolladores entrevistados. Como indica el sitio Rootstack (2021), su popularidad reside en que son aplicaciones móviles atractivas. Flutter se lanzó en el año 2017 y React Native en el 2015.

Según Rootstack, las ventajas de Flutter son las siguientes:

Una base de código: Este *framework* te permite desarrollar aplicaciones para iOS y Android, debido a que renderiza todo por sí mismo; deja al desarrollador correr todo dentro de una sola base de código ahorrando así una considerable cantidad de tiempo.

Una vistosa interfaz en poco tiempo: En Flutter, la interfaz de usuario se construye con *widgets*, estos son pequeños bloques de construcción UI utilizando una técnica llamada Composición. Todo el proceso para crear la interfaz es parecido a usar componentes de React.

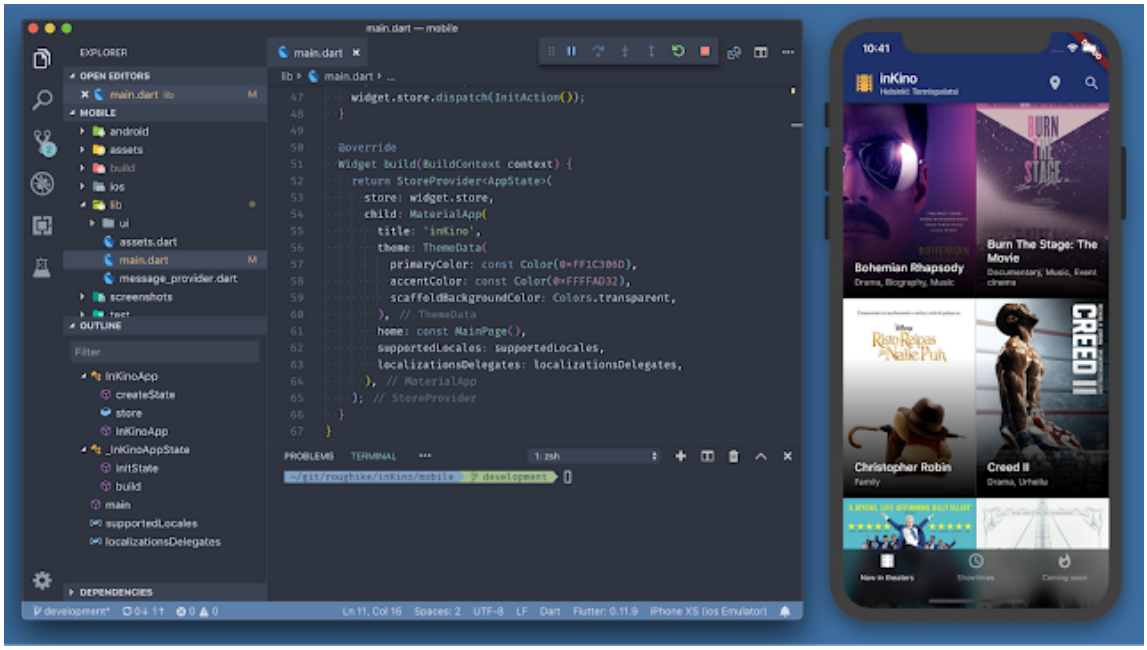
La representación de píxeles: Debido a que Flutter administra cada píxel en la pantalla de la aplicación, los *widgets* creados se verán iguales en todos los dispositivos móviles, incluyendo acá los modelos más antiguos. Esto elimina de raíz los problemas de soporte de dispositivo y permite crear interfaces de usuario que se ven exactamente igual tanto en Android como en iOS.

Desarrollo rápido: Una de las funciones de Flutter que realmente lo ha hecho brillar entre sus usuarios es la función de recarga en caliente brindando la capacidad de introducir cambios sobre la marcha, lo que le permite verlos inmediatamente durante el desarrollo.

Desarrollo de aplicaciones multiplataforma: Como ya se mencionó, Flutter es una herramienta multiplataforma que nos permite desarrollar para escritorio, dispositivos móviles y la web utilizando una única base de código. (párr. 6)

Figura 24

Flutter



Fuente: Google Developers (diciembre de 2018).

En síntesis, es la base de código que permite ahorrar tiempo, junto con la vistosidad de la interfaz, representación de pixeles, rapidez en el desarrollo y aplicaciones multiplataforma, lo que la hace que muchas personas la escojan. Por su parte, sobre React Native, Rootstack (2021) indica:

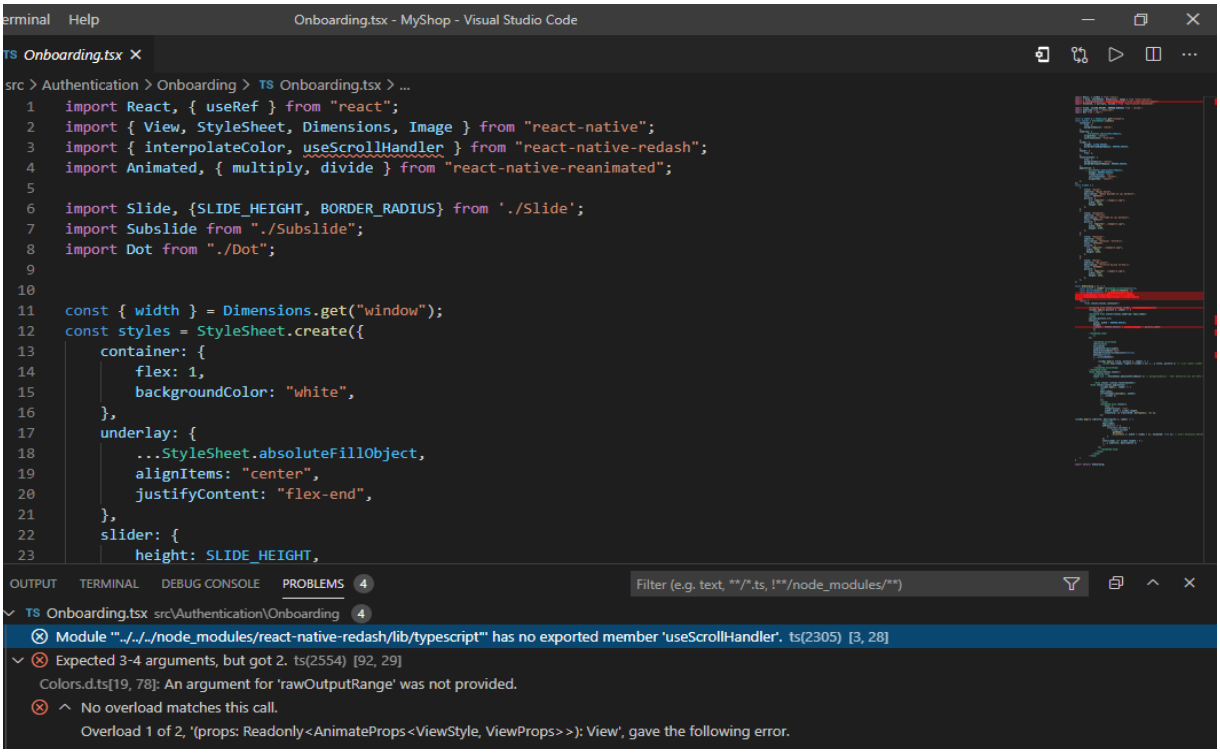
> **API Estable y respaldo de Facebook:** Es un *framework* con una API bastante estable y que cuenta con el respaldo de Facebook, más una comunidad bastante extensa de desarrolladores.
>
> **Todo desarrollador web lo puede aprender rápidamente:** Todo desarrollador que tenga conocimientos de React y JavaScript puede

aprender React Native bastante rápido, aprovechando las bibliotecas, herramientas, *frameworks* de diseño UI y tutoriales existentes de React.

Desarrollo rápido con solo una base de código: Al igual que Flutter, permite un desarrollo rápido para iOS, Android y web con una base de código. Una de sus ventajas principales es que React Native te permite agregar código nuevo a una aplicación en ejecución, lo que reduce el riesgo de perder algunas funcionalidades durante una recarga completa o la reconstrucción de la aplicación. (párr. 8)

Figura 25

React Native



Fuente: Stack Oveflow (2021). React Native - Typescript issue.

Ahora bien, de acuerdo con Rootstack, la escogencia de una u otra tecnología dependerá de la labor de los desarrolladores. Flutter se orienta más a la Internet y los dispositivos móviles, mientras que React Native se orienta más a computadoras y equipos integrados en proyectos.

6.3 Tecnología o tecnologías para automatización de procesos

Las tecnologías para la automatización de los procesos son variadas y su escogencia dependerá de lo que se requiere para el proyecto y de la capacidad adquisitiva del equipo.

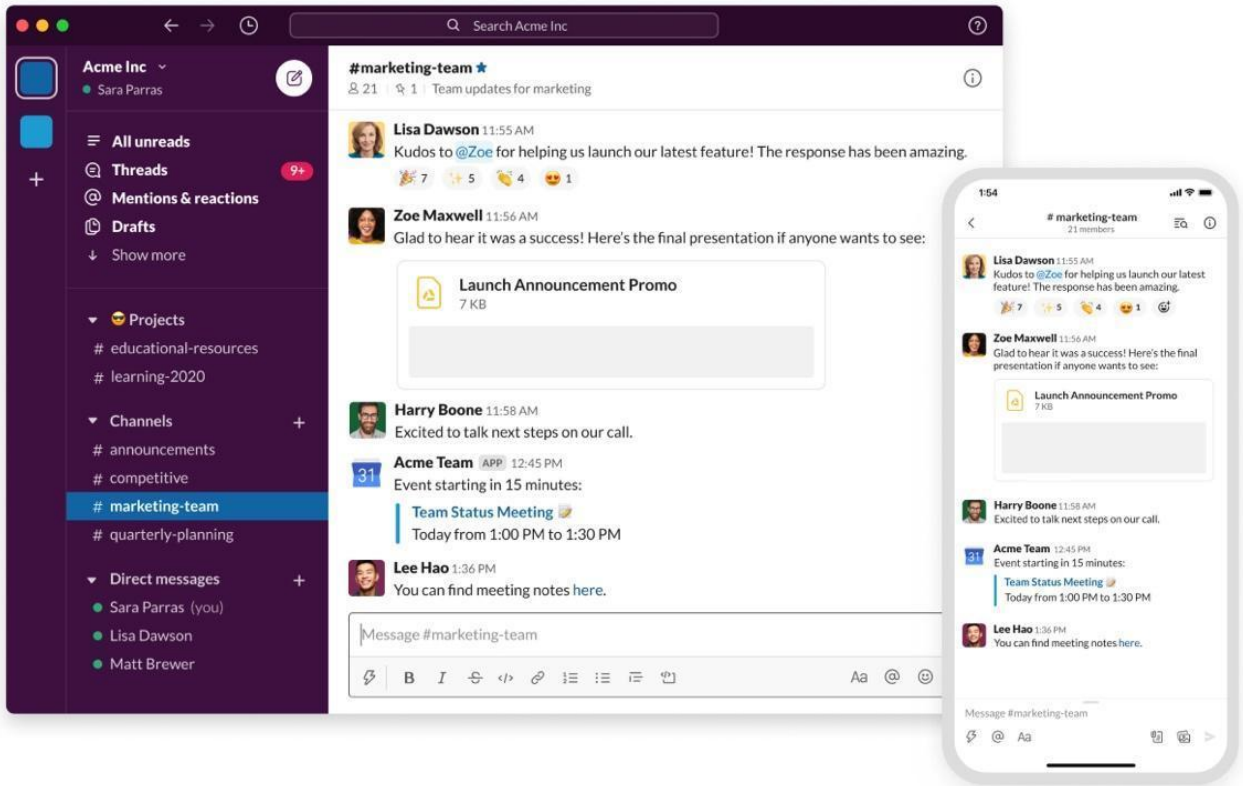
Para la comunicación se recomienda utilizar Slack, porque se dirige a las organizaciones, permitiendo la comunicación de sus empleados y logrando así un flujo de trabajo más rápido. Cebra (2021) manifiesta:

> La plataforma permite dividir este espacio general de trabajo en subcanales, que corresponden a grupos de chat independientes. Estos pueden estar compuestos, por ejemplo, por un área de la empresa donde los colaboradores pueden generar una conversación más fluida e intercambiar ideas. (párr. 7)

Cebra (2021) también la recomienda porque permite usar diferentes recursos a la vez, tales como imágenes, enlaces externos, archivos adjuntos, videos, audios y llamadas, entre otros.

Figura 26

Interfaz de Slack



Fuente: Slack (2022).

También se piensa en el uso de GitHub Projects para administrar las tareas del equipo. Se considera que esta herramienta puede ayudar de forma significativa cuando se trata de la gestión conjunta de proyectos. Souza (2020) expresa:

> Sin embargo, lo que realmente llama la atención es el hecho de que el equipo puede trabajar al mismo tiempo, desde diferentes lugares, no solo en la misma ciudad, sino también en el mundo.

> Hoy en día, para cualquier negocio, la automatización de los flujos de trabajo es esencial y GitHub lo hace posible. Los recursos encontrados en la plataforma ayudan en el desarrollo de proyectos, facilitando el crecimiento de la empresa como un todo. (párrs. 24-25)

GitHub se puede usar mediante una cuenta personal o una cuenta de equipo, y existen aplicaciones gratuitas para personas individuales o las que usan código abierto. Además, tiene el Hello World Guide para las personas que lo empiezan a usar por primera vez, para aprender en especial como crear un *branch*, o hacer un *pull request*, que es cuando se informa a los otros miembros que se incorpora un *branch* propio. Asimismo, Souza comenta la importancia de los repositorios:

> En Git, tampoco hay problema con el reemplazo del código y la pérdida de información, ya que las versiones se guardan en el repositorio, que es un directorio donde se almacenan todos los archivos del proyecto de desarrollo.
>
> Es posible almacenar este repositorio en tu computadora o, si utilizas una plataforma en línea como GitHub, también puede almacenarse allí. (párrs. 5-6)

Por último, se recomienda utilizar también la herramienta GitHub Actions, para realizar implementación continua, ya que, como señala el sitio Plain Concepts (2021):

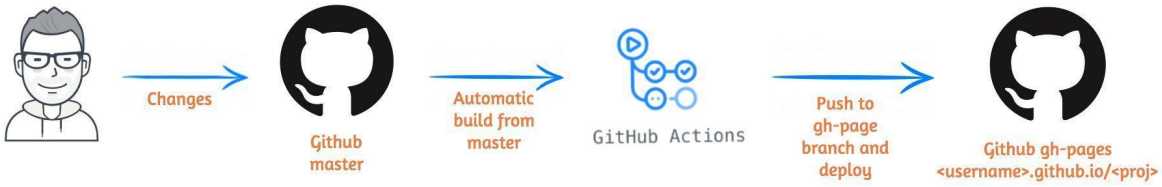
> GitHub Actions es una herramienta que permite reducir la cadena de acciones necesaria para la ejecución de código, mediante la elaboración de

un de flujo de trabajo (o Pipeline), configurable para que GitHub reaccione a ciertos eventos de forma automática según nuestras preferencias. (párr. 1)

La herramienta se utiliza para crear *workflows* que permiten compilar, testear y hacer implementación de todo el código, utilizando contenedores de paquetes Docker que están implementados en los servidores GitHub. Además, brinda la posibilidad de utilizarla en cualquier lenguaje de programación.

Figura 27

GitHub Actions



Fuente: Automate your Deployment with GitHub Actions.

6.4 Proceso de desarrollo propuesto

Se recomienda utilizar un híbrido de la metodología KANBAN con SCRUM (llamada SCRUMBAN) al iniciar los proyectos, lo que permite crear un entregable rápido y salir al mercado de forma más expedita. Esto se puede combinar con el uso de GitHub issues para llevar un control de los errores (*bugs*) que se encuentren en la aplicación durante el proceso de desarrollo.

Debe recordarse que como indica Asana (2022) la metodología KANBAN se caracteriza por su agilidad y una filosofía basada en la planificación adaptativa, metodología temprana y mejora continua. Asana (2022) expone que “Cuando se menciona KANBAN en la gestión de proyectos, lo más común es que se haga referencia a los tableros KANBAN. Estos representan las etapas del trabajo con columnas que incluyen las tareas individuales para cada etapa” (párr. 16).

Por su parte, SCRUM también tiene un marco ágil, pero se usa más como una herramienta para visualizar el trabajo. Asana (2022) señala que:

> SCRUM es un marco completo que puedes implementar para administrar equipos. Taiichi Ohno desarrolló esta metodología, que ofrece un modelo de valores, pautas y roles para ayudar a tu equipo a centrarse en la mejora continua y la iteración. (párr. 22)

Asana afirma que SCRUM presenta menos flexibilidad que KANBAN, pero ayuda mucho a la colaboración y proyectos de alto impacto. Como indica Roche (2021), lo comentado hace que tanto KANBAN como SCRUM tengan similitudes en planteamientos y objetivos, y por ello se haya tratado de unirlos.

El *framework* SCRUM está cerrado a modificaciones pero abierto a extensiones, lo cual se refleja en esta nueva guía llamada “Guía KANBAN para equipos SCRUM”. KANBAN se considera una extensión de SCRUM que agrega las siguientes prácticas:

- Visualización del flujo de trabajo.

- Limitación del WIP.
- Gestión activa de las partidas en curso.
- Inspección y adaptación del flujo de trabajo.

Con el fin de adaptar el flujo de trabajo para maximizar el flujo de valor para los clientes, es necesario añadir al menos las siguientes métricas:

- WIP.
- Duración del ciclo.
- Edad de ítem de trabajo.
- Rendimiento.

Esta es una aplicación de KANBAN en el contexto de SCRUM y, como tal, tiene una definición que puede ser distinta de otras implementaciones de KANBAN

Resulta evidente que lo que se busca es tomar lo mejor de las dos metodologías, uniendo la visualización y los flujos de trabajo para su uso en equipos de personas, con la flexibilidad requerida. Roche (2021) indica que aquí se trata de incorporar el flujo Lean/KANBAN en el contexto SCRUM. También expone que, aunque SCRUM es bastante bueno, KANBAN puede mejorar los flujos durante el *sprint*. En este punto debe indicarse que, como menciona SCRUM.org (2021), SCRUM es un marco de trabajo, por lo que funciona bien para contener otras técnicas, metodologías y prácticas.

Se propone al mismo tiempo utilizar GitHub Issues para llevar un control de los problemas (*bugs*) que se encuentren en la aplicación durante el proceso de

desarrollo. Los *bugs* son errores informáticos que se producen de forma inesperada. Al respecto, el sitio de documentos GitHub (2022) expone que:

- Los problemas se pueden manejar de varias maneras, por lo que puede elegir el método más conveniente para su flujo de trabajo.
- Los miembros de una empresa con usuarios administrados solo pueden hacer cambios en los repositorios que sean parte de ella.
- Para los problemas se puede realizar un seguimiento de *bugs*, mejoras u otras solicitudes. (párrs. 1-3)

El repositorio, por su parte, usa plantillas de problemas que permiten especificar el tipo de problema que se desea abrir, o también se puede optar por crear un problema en blanco. Los administradores de repositorio pueden inhabilitar las propuestas de un repositorio.

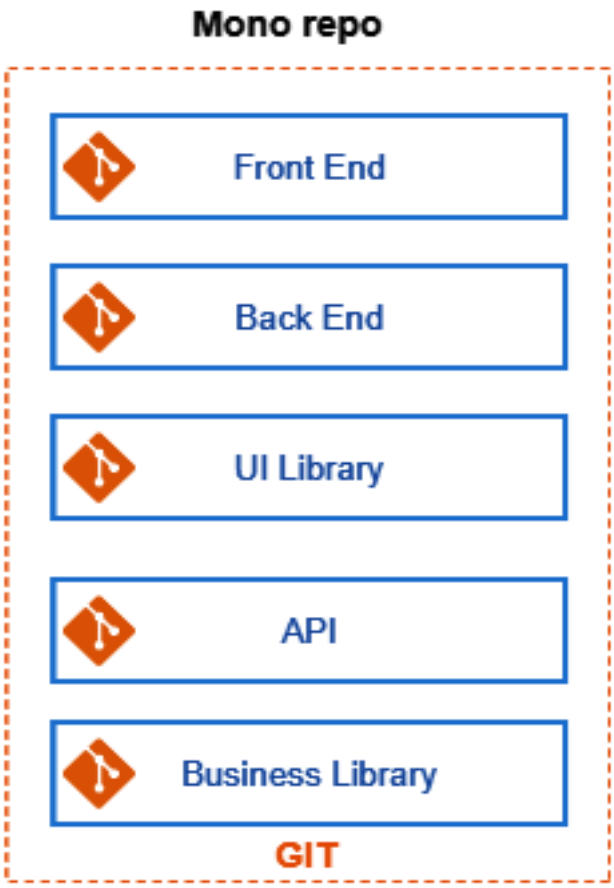
6.5 Configuración de repositorios y ambientes para el desarrollo

Se recomienda utilizar repositorios estilo mono-repo, pues, como reseña el sitio JC con Software (2020), la administración del código es mucho más fácil en este tipo de repositorios. También, se debe considerar que, por lo general, una pyme no contará con suficiente personal para hacerse cargo de diversos repositorios.

Con los repositorios estilo mono-repo se obtienen beneficios relacionados con el mantenimiento del código, la integración, la reutilización y refactorización, y el manejo de cambios a gran escala.

Figura 28

Estructura de un mono-repo



Fuente: JC con Software. (30 de junio de 2020). Mono-repo vs. multi-repo.

Sobre el mono-repo, el sitio Geekflake (2021) afirma que son muchas sus ventajas, entre ellas:

- Un solo lugar para almacenar todo el código del proyecto, y todos los miembros del equipo pueden acceder a él.
- Fácil de reutilizar y compartir código, colaborar con el equipo.

- Fácil de comprender el impacto de su cambio en todo el proyecto.
- La mejor opción para la refactorización de código y grandes cambios en el código.
- Los miembros del equipo pueden obtener una vista general de todo el proyecto.
- Dependencias fáciles de administrar. (párr. 22)

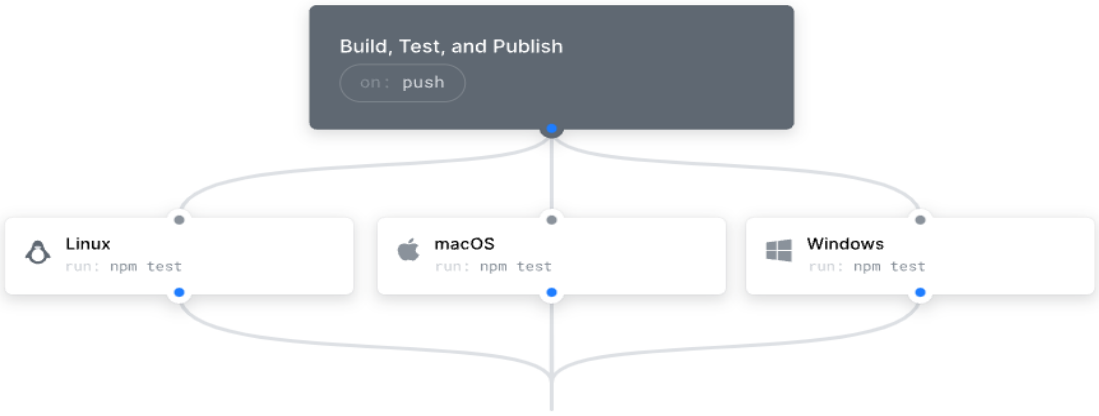
En otras palabras, un mono-repo es fácil de usar, pero requiere de la colaboración y comunicación constante del equipo; también tiene desventajas, y la principal de ellas es que el proceso de pruebas y compilación se vuelve más lento a medida que se agregan más archivos.

Se recomienda utilizar herramientas de desarrollo automatizadas, tales como GitHub Actions, para realizar las pruebas del código de manera automatizada cada vez que se realice un cambio en los repositorios. Al respecto, el sitio GitHub.com (2022) comenta que en los flujos de trabajo se pueden llevar a cabo eventos como envíos, creación de problemas o nuevas versiones. Estas acciones se actualizan y son mantenidas por la comunidad. En GitHub.com se comenta que:

> Ya sea que desee crear un contenedor, implementar un servicio web o automatizar la bienvenida a nuevos usuarios a sus proyectos de código abierto, hay una acción para eso. Combine GitHub packages con Actions para simplificar la administración de paquetes, incluidas las actualizaciones de versiones, la distribución rápida con nuestra CDN global y la resolución de dependencias, utilizando su GITHUB_TOKEN existente. (párr. 2)

Figura 29

Flujo de trabajo en GitHub



Fuente: GitHub (2022).

Debe recordarse que el flujo de trabajo, como manifiesta Atlassian (2020), es una recomendación sobre cómo GitHub debe llevar a cabo su trabajo de la manera más homogénea y productiva.

CAPÍTULO VII

REFERENCIAS

7.1 Referencias bibliográficas

ABAMobile. (2021, 18 de mayo). *Apps multiplataforma. Qué son y características*. https://abamobile.com/web/apps-multiplataforma-que-son-y-caracteristicas/.

Aguirre, G. (2018, 10 de septiembre). *Por qué React Native es una excelente alternativa en América Latina*. Medium. https://medium.com/underscopeio/por-qu%C3%A9-react-native-es-la-mejor-alternativa-para-am%C3%A9rica-latina-97cf3041d6ae.

Alcántara, B. (2021, 17 de abril). *Estas son las marcas de móviles más vendidas en cada continente.* Andro4all. https://andro4all.com/noticias/moviles/estas-son-las-marcas-de-moviles-mas-vendidas-en-cada-continente.

Amazon. (2021). *Integración continua del software | Pruebas automatizadas | AWS*. https://aws.amazon.com/es/devops/continuous-integration/.

Andreu, I. (2021, 15 de julio). *Lean Manufacturing: ¿qué es y cuáles son sus principios?* APD España. https://www.apd.es/lean-manufacturing-que-es/.

Asana, T. (2022, 19 de enero). *¿Cuál es la diferencia entre Waterfall, Agile, Kanban y Scrum?* Atlassian. https://www.atlassian.com/es/git/tutorials/comparing-workflows#:~:text=Un%20flujo%20de%20trabajo%20de,de%20forma%20eficaz%20y%20estable.

Atlassian. (2020). *Comparar workflows*. https://www.atlassian.com/es/git/tutorials/comparing-workflows#:~:text=Un%20flujo%20de%20trabajo%20de,de%20forma%20eficaz%20y%20estable.

Atlassian. (s. f.). *¿Para qué se utiliza Jira Software?* https://www.atlassian.com/es/software/jira/guides/use-cases/what-is-jira-used-for#Jira-for-requirements-&-test-case-management.

B., G. (2021, 8 de marzo). *¿Qué es GitHub y Cómo Usarlo?* Tutoriales Hostinger. https://www.hostinger.es/tutoriales/que-es-github#Que-es-GitHub-y-por-que-es-tan-popular.

Caballero, J. (2021, 29 de abril). *Desarrollo de aplicaciones híbridas: ¿Cuándo son buena opción?* Armadillo Amarillo. https://www.armadilloamarillo.com/blog/desarrollo-de-aplicaciones-hibridas-cuando-son-buena-opcion/.

Castillo, I. (2018, 13 de diciembre). *Marco contextual: característica, cómo se hace y ejemplo.* Lifeder. https://www.lifeder.com/marco-contextual/.

Cebra. (2021). Slack: La herramienta que optimizará el trabajo remoto. https://www.cebra.cl/blog/slack-la-herramienta-que-optimizara-el-trabajo-remoto/.

Chacón, S., y Straub, B. (2014). *Git - Una breve historia de Git. GIT SCM*. https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Una-breve-historia-de-Git.

Dybå, T. (2003). Factors of software process improvement success in small and large organizations. *ACM SIGSOFT Software Engineering Notes*, *28*(5), 148-157. https://doi.org/10.1145/949952.940092.

Dziuba, A. (2021). *React Native vs. Flutter: Which to Choose for Cross-platform Development?* Relevant. https://relevant.software/blog/react-native-vs-flutter-which-to-choose-for-cross-platform-development/.

Esparza, E. (2020, 12 de febrero). *Así ha evolucionado la industria móvil desde 1973*. En Naranja. https://www.ennaranja.com/economia-facil/evolucion-industria-movil/.

Fernández, Y. (2019). *Qué es Github y qué es lo que le ofrece a los desarrolladores*. Xataka. https://www.xataka.com/basics/que-github-que-que-le-ofrece-a-desarrolladores.

García, M. (2021, 14 de octubre). *Android vs. iOS: ¡Batalla de sistemas operativos 2022!* Crehana. https://www.crehana.com/cr/blog/desarrollo-web/android-vs-ios/.

Garzas, J. (2015, 24 de julio). *¿Qué es Docker? ¿Para qué se utiliza? Explicado de forma sencilla.* https://www.javiergarzas.com/2015/07/que-es-docker-sencillo.html.

Geek Flake. (2021). *Mono-repositorio vs. multi-repositorio: arrojar luz sobre las estrategias de repositorio de código*. https://geekflare.com/es/code-repository-strategies/#:~:text=Ventajas%20de%20Mono%2Drepo,cambio%20en%20todo%20el%20proyecto.

GitHub.com. (2022). Automatice su flujo de trabajo desde la idea hasta la producción. https://github.com/features/actions.

Google Developers. (2018, 6 de diciembre). *Flutter 1.0: Kit de herramientas portátil de la IU de Google*. https://developers-latam.googleblog.com/2018/12/flutter-10-kit-de-herramientas-portatil.html.

Hernández, R., Fernández, C., y Baptista, P. (2014). *Metodología de Investigación* (3.ª ed.). Mc Graw Hill.

Hernández Hermosillo, S. (2013). *Población y muestra*. Universidad Autónoma del Estado de Hidalgo. https://www.uaeh.edu.mx/docencia/VI_Lectura/maestria/documentos/LECT86.pdf.

Histografias. (2021). *Historia de Slack, la mayor adquisición de Salesforce $27.700M*. https://histografias.com/historia_slack_infografia_salesforce.html.

Itedamza. (2021). *Maxwell. Transferencia educativa viajeros de la luz. Cuadernillo 3.* http://itedamza.frm.utn.edu.ar/wp-content/uploads/2020/05/MAXWELL.pdf.

JC con Software. (2020, 30 de junio). *Monorepo vs. Multirepo*. https://jconsoftwares.com/blog/article/18.

Jiménez-Hernández, E. y Orantes-Jiménez, Sandra. (2012). *Metodología híbrida para desarrollo de software en México*. https://www.iiis.org/CDs2012/CD2012IMC/CICIC_2012/PapersPdf/CB153YB.pdf.

Kaneko, J. (2020, 27 de agosto). *Understanding React Native Architecture*. DEV Community. https://dev.to/goodpic/understanding-react-native-architecture-22hh.

Kata Software. (2020, 30 de noviembre). *Docker - Breve historia de los contenedores*. https://kata-software.com/es/publicaciones/docker-historia-de-los-contenedores.

Leyva, A. (2021, 21 de enero). *GitHub Actions con Notificaciones de Slack*. https://alanleyvaweb.medium.com/github-actions-con-notificaciones-de-slack-e1cbb67df5e8.

Logan, R. K., y Scolari, C. (2014). El surgimiento de la comunicación móvil en el ecosistema mediático. *Letra. Imagen. Sonido: Ciudad Mediatizada*, *11*, 67-82. https://dialnet.unirioja.es/servlet/articulo?codigo=5837855.

LucidChart. (2019, 9 de octubre). *Agile vs. Waterfall vs. Kanban vs. Scrum: What's the Difference?* Lucidchart Blog. https://www.lucidchart.com/blog/agile-vs-waterfall-vs-kanban-vs-scrum.

Maida, E. y Pacienzia, J. (2015). *Metodologías de desarrollo de software* [Tesis de licenciatura]. Universidad Católica de Argentina. https://repositorio.uca.edu.ar/bitstream/123456789/522/1/metodologias-desarrollo-software.pdf.

Massimini, A. (2021, 7 de julio). *Herramienta de gestión de proyectos: ¿Cómo elegirla?* Project COR. https://projectcor.com/es/blog/herramienta-de-gestion-de-proyectos-como-elegirla/.

Maxwell, J. C. (2021). *A Dynamical Theory of the Electromagnetic Field*. Rough Draft Printing.

Meléndez Valladarez, S. M., Gaitán, M. E. y Pérez Reyes, N. N. (2016). *Metodología ágil de desarrollo de software programación extrema* [Tesis]. Universidad Nacional Autónoma de Nicaragua. https://repositorio.unan.edu.ni/1365/1/62161.pdf.

Moberest | Servicio Desarrollo de apps Nativas y Cross. (2021). *Moberest - App Marketing Solutions*. http://www.moberest.com/servicios-desarrollo-apps/.

Monja, M. L. (2021, 19 de julio). *Metodologías ágiles: definición, manifiesto, principios, SCRUM, KANBAN. Innovar o morir*.

http://innovaromorir.com/metodologias-agiles-definicion-manifiesto-principios-SCRUM-KANBAN/.

Muente, G. (2021, 12 de febrero). *Guía completa del framework: qué es, cuáles tipos existen y por qué es importante en Internet.* Rock Content - ES. https://rockcontent.com/es/blog/framework/.

Muñoz, D. A., Ordóñez, H., y Bucheli, V. (2021). Guideline to implement the CALMS model (DevOps) at mipymes in the software development organizations of the South of Colombia [Lineamientos para la implementación del modelo CALMS de DevOps en mipymes desarrolladoras de software en el contexto surcolombiano]. *Revista Guillermo de Ockham*, *18*(1). https://doi.org/10.21500/22563202.4270.

Murgo, E. (2019, 14 de junio). *Historia y evolución de los teléfonos celulares: ¿con cuál empezaste?* Unidiversidad. https://www.unidiversidad.com.ar/historia-y-evolucion-de-los-telefonos-celulares-con-cual-empezaste.

Netapp. (2019). *¿Qué es DevOps? - explicación de prácticas y beneficios*. https://www.netapp.com/es/devops-solutions/what-is-devops/.

OpenMindBBVA. (s. f.). *El impacto de internet en la vida diaria*. https://www.bbvaopenmind.com/articulos/el-impacto-de-internet-en-la-vida-diaria/.

Osadchuk, S. (2021, 12 de diciembre). *Flutter vs. React Native en 2021: ¿cuál es mejor para su proyecto?* DOIT Software. Recuperado de https://doit.software/blog/flutter-vs-react-native#screen2.

Pedrozo Petrazzini, G. O. (2012). *Sistemas operativos en dispositivos móviles* [Monografía]. Universidad Nacional del Nordeste. https://studylib.es/doc/2435595/sistemas-operativos-en-dispositivos-m%C3%B3viles.

Plain Concepts. (2021, 6 de abril). *GitHub Actions | Introducción*. https://www.plainconcepts.com/es/que-es-github-actions/#:~:text=GitHub%20Actions%20es%20una%20herramienta,forma%20autom%C3%A1tica%20seg%C3%BAn%20nuestras%20preferencias.

Platzi. (s. f.). *Aplicación híbrida o nativa: ¿Cuál es mejor?* https://platzi.com/blog/hibrida-o-nativa/.

Project Management Institute. (2022a). What is Project Management? https://www.pmi.org/about/learn-about-pmi/what-is-project-management.

Project Management Institute. (2022b). Who are Project Managers? https://www.pmi.org/about/learn-about-pmi/who-are-project-managers.

Project Management Institute. (2022c). Agility Amplified. PMI's Latest Pulse of The Profession Reveals Gymnastic Powers. https://www.pmi.org/learning/library/pulse-of-the-profession-reveals-gymnastic-powers-13075.

Project Management Institute. (2022d). The Next Agile Awakening:Four Agile Leaders Discuss New Possibilities in a World of Sudden Change. Recuperado de https://www.pmi.org/learning/library/agile-leaders-discuss-possibilities-post-pandemic-13077.

Quality Devs. (2020, 16 de julio). *Diferencias entre iOS y Android | Quality Devs | Aplicaciones*. https://www.qualitydevs.com/2020/07/14/diferencias-entre-ios-android/.

Ramírez, U. (2016). *Platzi: Cursos online profesionales de tecnología*. Platzi. https://platzi.com/blog/hibrida-o-nativa/.

Ranchal, J. (2018, 25 julio). *Inicios, evolución y futuro del teléfono móvil*. MuyCanal. https://muycanal.com/2014/01/31/futuro-del-telefono-movil#:~:text=La%20primera%20generaci%C3%B3n%201G%20fue,y%20segu%C3%ADa%20utilizando%20canales%20anal%C3%B3gicos.&text=La%20segunda%20generaci%C3%B3n%202G%20lleg%C3%B3,europeo%20de%20telefon%C3%ADa%20m%C3%B3vil%20digital.

RedNew. (2020, 18 de abril). *Datos sobre el uso de iPhone vs. Android en el mundo*. https://rednew.es/iphone-vs-android/.

Rehkopf, M. (2021). *Kanban frente a Scrum: ¿qué metodología ágil prefieres?* Atlassian. https://www.atlassian.com/es/agile/kanban/kanban-vs-scrum#:~:text=Resumen%3A%20el%20dilema%20de%20%E2%80%9Ckanban,de%20trabajo%20cortos%20y%20estructurados.

Roa, M. M. (2021, 30 de agosto). *Android y iOS dominan el mercado de los smartphones.* Statista Infografías. https://es.statista.com/grafico/18920/cuota-de-mercado-mundial-de-smartphones-por-sistema-operativo/.

Roche, J. (2021). *Scrum con Kanban: la visión de Scrum.org. https*://www2.deloitte.com/es/es/pages/technology/articles/scrum-kanban-vision-scrumorg.html.

Rootstack. (2021, 17 de diciembre). *Flutter vs. React Native: cuál es la mejor opción para desarrollar aplicaciones móviles*. https://www.rootstack.com/es/blog/react-native-vs-flutter/.

Ruchir, R. (s. f.). *¿Cuáles son los diferentes tipos de herramientas de desarrollo móvil multiplataforma para 2020?* Cyber Infrastructure, "CIS". https://www.cisin.com/coffee-break/es/enterprise/what-are-the-different-types-of-cross-platform-mobile-development-tools-for-2020.html.

Saini, T. (2020). *DevOps is helpful to Developers?* TechTarget. https://www.datasciencecentral.com/profiles/blogs/how-devops-helps-the-developers.

Sánchez, C. (2018). *Evolución de los teléfonos móviles desde su invención* [Mensaje en un blog]. InformaticaDirecto. https://www.informaticadirecto.com/blog/evolucion-los-telefonos-moviles-desde-invencion/.

Sánchez Valtierra, J. (2013). *Práctica docente. Métodos de investigación mixto: un paradigma de investigación cuyo tiempo ha llegado*. Blogger. http://www.blogger.com/profile/13155690421517949845.

Santana, X. (2014, 7 de febrero). *Por qué desarrollar tu app en cross-platform*. SlashMobility. https://slashmobility.com/blog/2014/02/desarrollo-de-apps-cross-platform/.

Saurabh. (2021). *¿Qué es Jenkins? Jenkins para la integración continua*. [Mensaje en un blog]. https://www.edureka.co/blog/what-is-jenkins/.

Scrum.org. (2021). *La guía Kanban para Scrum Teams*. https://scrumorg-website-prod.s3.amazonaws.com/drupal/2021-03/2021-Kanban-Guide-Spanish-European.pdf.

Sitio de documentos GitHub. (2022). *Creating an issue*. https://docs.github.com/es/enterprise-cloud@latest/issues/tracking-your-work-with-issues/creating-an-issue.

Slack.com. (2022). *GitHub para Slack*. https://slack.com/intl/es-cr/help/articles/232289568-GitHub-para-Slack.

Souza, I. (2020, 20 de marzo). *Entiende a detalle qué es GitHub y su importancia para un negocio*. Rockcontent. https://rockcontent.com/es/blog/github/#:~:text=GitHub%20es%20una%20excelente%20herramienta,es%20conocida%20como%20GitHub%20Enterprise.

Stack Overflow. (2021). *React Native - Typescript issue.* https://stackoverflow.com/questions/63933592/react-native-typescript-issue.

Statista. (2021, 29 de junio). *Market share of mobile operating systems worldwide 2012–2021*. https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/#statisticContainer.

Sysadmin, A. (2019). *Docker Architecture and its Components for Beginner*. Geekflare. https://geekflare.com/docker-architecture/.

Tillman, M. (2021, 7 de abril). *¿Qué es Slack y cómo funciona? Además de muchos consejos y trucos de Slack*. Pocket-lint. https://www.pocket-lint.com/es-es/aplicaciones/noticias/150925-que-es-holgura-y-como-funciona-consejos-trucos.

Turrado, J. (2020, 6 de mayo). *Github: mucho más que un simple almacén de código fuente.* campusMVP. https://www.campusmvp.es/recursos/post/github-mucho-mas-que-un-simple-almacen-de-codigo-fuente.aspx.

Vargas-Hernández, J. G., Muratalla-Bautista, G., y Jiménez-Castillo, M. (2016). Lean Manufacturing ¿una herramienta de mejora de un sistema de producción? *Ingeniería Industrial. Actualidad y Nuevas Tendencias*, *5*(17), 153-174. https://www.redalyc.org/pdf/2150/215049679011.pdf.

Wiegers, K., y Sturzenberger, D. (2000). A modular software process mini-assessment method. *IEEE Software*, *17*(1), 62-69. https://doi.org/10.1109/52.819970.

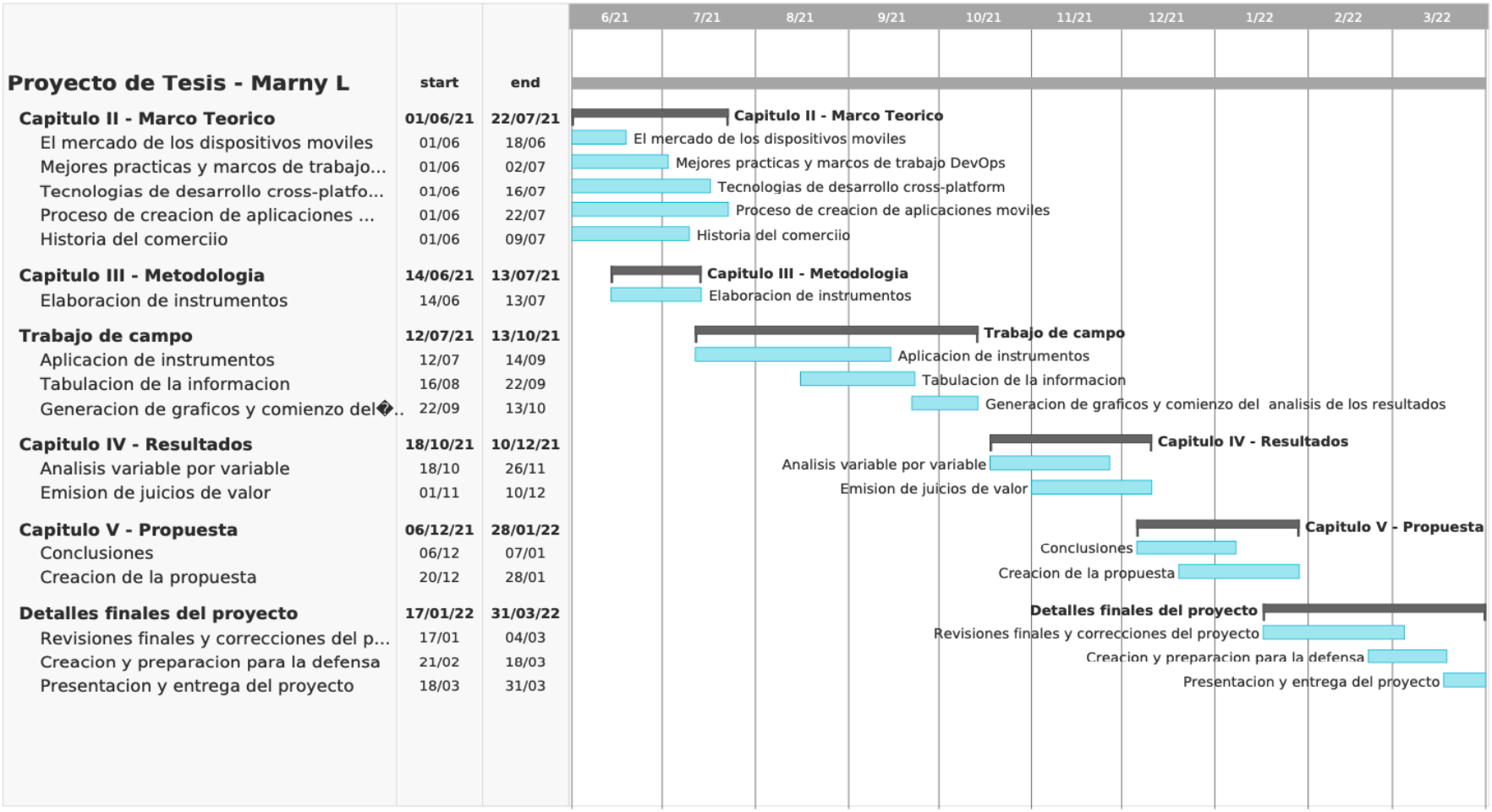
Zamorano García, J. (2018). *Tercera fase investigativa*. Universidad Autónoma del Estado de Hidalgo. https://www.uaeh.edu.mx/scige/boletin/prepa4/n2/m4.html.

CAPÍTULO VIII

APÉNDICES

8.1 Apéndices

8.1.1 Apéndice A – Cronograma del proyecto



8.1.2 Apéndice B – Entrevista a los arquitectos

ENTREVISTA A ARQUITECTOS DE SOFTWARE

Buen día. Mi nombre es Marni Andrei López López, y estoy realizando el proyecto de investigación denominado “Análisis de las tecnologías de desarrollo *cross-platform* existentes en el mercado y propuesta de un proceso de desarrollo automatizado aplicable a una pyme enfocada en el desarrollo de aplicaciones multiplataforma”, como requisito de graduación de la carrera de Ingeniería del Software.

Agradezco su participación para responder. La información suministrada será tratada bajo los principios de confidencialidad y con fines exclusivos de la investigación.

Instrucciones: Proceda a contestar cada pregunta según su experiencia.

1. ¿Según su punto de vista qué tipo de aplicación tiene mayor fuerza en el mercado?
    - Aplicaciones móviles.
    - Aplicaciones web.
    - Aplicaciones de escritorio.
    - Aplicaciones híbridas (*cross-platform*).
2. ¿Consideraría su *expertise* primariamente en desarrollo móvil, en *cross-platform* o de escritorio?
    - Móvil.
    - Escritorio.
    - *Cross-platform.*

3. ¿Ha tenido experiencia creando la arquitectura para aplicaciones móviles o híbridas?
    - Sí.
    - No.
4. De las siguientes, ¿cuáles tecnologías de desarrollo *cross-platform* ha utilizado en un proyecto?
    - React Native.
    - Xamarin.
    - Ionic.
    - Flutter.
    - Ninguna de las anteriores.
5. De las siguientes, ¿cuáles tecnologías de desarrollo *cross-platform* considera que son las más fáciles de implementar en un proyecto?
    - React Native.
    - Xamarin.
    - Ionic.
    - Flutter.
    - Ninguna de las anteriores.
6. ¿Considera usted que una empresa pyme debe optar por desarrollar con tecnologías de desarrollo *cross-platform* en lugar de desarrollar con tecnologías nativas?
    - Sí.
    - No.
7. De las siguientes opciones, ¿cuáles considera que son el mayor reto al momento de desarrollar una aplicación *cross-platform*?
    - Tiempo de desarrollo.
    - Definición de una arquitectura del proyecto.
    - Curva de aprendizaje de las tecnologías a utilizar.
    - Automatización correcta de los procesos de desarrollo.
    - Ninguna de las anteriores.

8. De las siguientes opciones, ¿cuáles considera que son el mayor reto al momento de IMPLEMENTAR una aplicación *cross-platform*?
    - Documentación del proyecto.
    - Implementación en tiendas en línea.
    - Creación de pruebas automatizadas.
    - Procesos de QA rigurosos en el proceso de desarrollo.
    - Ninguna de las anteriores.
9. Con respecto a su *expertise*, ¿cuál considera que es la mejor arquitectura de proyecto que podría aplicarse en una aplicación desarrollada con tecnologías *cross-platform*?
    - Arquitectura de capas.
    - Arquitectura de eventos.
    - Arquitectura de componentes.
    - Ninguna de las anteriores.
10. ¿Considera usted que al desarrollar un proyecto de desarrollo los equipos deben darle prioridad a brindar una buena experiencia de usuario?
    - Sí.
    - No.
    - No responde.
11. ¿Considera usted como arquitecto que los equipos de desarrollo deben contar con personas específicamente dedicadas a la experiencia de usuario?
    - Sí.
    - No.
    - No responde.
12. ¿Considera usted que se deben utilizar herramientas de automatización de procesos para realizar procesos repetitivos en los proyectos de manera automática?
    - Sí.
    - No.
    - No responde.

13. De las siguientes, ¿qué herramientas de automatización de procesos ha utilizado en sus proyectos de desarrollo?
    - GitHub.
    - GitHub Actions.
    - Slack.
    - Docker.
    - Jenkins.
    - Otra: _____.
14. ¿Para qué ha utilizado estas herramientas de automatización de procesos ha utilizado en sus proyectos?
    - Implementaciones automatizadas.
    - *Testing* de código automatizado.
    - Comunicación del equipo.
    - Estandarización de la base de datos de código
    - Ninguna de las anteriores.
15. ¿Al utilizar estas herramientas considera que la curva de aprendizaje para los desarrolladores del proyecto fue muy pronunciada?
    - Sí.
    - No.
    - No responde.
16. ¿Cuál es su experiencia en desarrollo de aplicaciones móviles?
    - Más de 5 años.
    - Entre 1 y 5 años
    - Menos de 1 año.
17. ¿Cuál es su nivel de formación académica?
    - Postgrado universitario completado.
    - Grado universitario completado.
    - Universitario en progreso.
    - Bachiller completado.

Adicionalmente, si usted así lo desea, puede compartir sus datos personales a continuación, o bien, finalizar esta encuesta, esto en caso de que, de ser necesario, le pueda contactar con más preguntas sobre el tema.

Nombre: _____

Correo electrónico: _____

8.1.3 Apéndice C – Encuesta a los desarrolladores de tecnologías móviles

ENCUESTA A DESARROLLADORES

Buen día. Mi nombre es Marni Andrei López López, y estoy realizando el proyecto de investigación denominado “Análisis de las tecnologías de desarrollo *cross-platform* existentes en el mercado y propuesta de un proceso de desarrollo automatizado aplicable a una pyme enfocada en el desarrollo de aplicaciones multiplataforma”, como requisito de graduación de la carrera de Ingeniería del Software.

Agradezco su participación para responder. La información suministrada será tratada bajo los principios de confidencialidad y con fines exclusivos de la investigación.

Instrucciones: Proceda a contestar cada pregunta según su criterio.

1. ¿Consideraría su *expertise* primariamente en desarrollo móvil, en *cross-platform* o de escritorio?
    - Móvil.
    - Escritorio.
    - *Cross-platform*.
2. ¿Cuál es su experiencia en desarrollo de aplicaciones móviles?
    - Más de 5 años.
    - Entre 1 y 5 años.
    - Menos de 1 año.

3. ¿Cuál es su nivel de formación académica?
    - Postgrado universitario completado.
    - Grado universitario completado.
    - Universitario en progreso.
    - Bachiller completado.
4. ¿Ha trabajado usted en alguna aplicación que actualmente se utiliza en el mercado?
    - Sí.
    - No.
    - No responde.
5. ¿Qué tan relevante considera que es crear aplicaciones amigables, de fácil comprensión y utilización?
    - Muy relevante.
    - Relevante.
    - Indiferente.
    - Poco relevante.
    - Nada relevante.
    - No responde.
6. ¿Cuánto considera que es el tiempo adecuado que debería durar un proyecto en presentar un MVP (producto mínimo viable) a los usuarios finales?
    - Menos de 3 meses.
    - Más de 3 meses y menos de 6 meses.
    - Más de 6 meses y menos de 1 año.
    - Otro: _____

7. De las siguientes, ¿cuáles tecnologías de desarrollo *cross-platform* ha utilizado en un proyecto?
    - React Native.
    - Xamarin.
    - Ionic.
    - Flutter.
    - Ninguna de las anteriores.
8. De las siguientes, ¿cuáles tecnologías de desarrollo *cross-platform* considera que son las más fáciles de aprender para un desarrollador?
    - React Native.
    - Xamarin.
    - Ionic.
    - Flutter.
    - Ninguna de las anteriores.
9. De las siguientes, ¿cuáles tecnologías de desarrollo *cross-platform* elegiría para desarrollar su propio proyecto de desarrollo de *software*?
    - React Native.
    - Xamarin.
    - Ionic.
    - Flutter.
    - Ninguna de las anteriores.
10. ¿Conoce alguna aplicación que se haya creado en Costa Rica y sea utilizada a nivel nacional e internacional con tecnologías *cross-platform*?
    - Sí.
    - No.
    - No responde.

11. ¿Considera que los proyectos de desarrollo móvil deben desarrollarse con tecnologías *cross-platform*, tecnologías nativas principalmente o una combinación de ambas?
    - Tecnologías *cross-platform* principalmente.
    - Tecnologías nativas principalmente.
    - Una combinación con ambos tipos de tecnologías.
    - No responde.
12. ¿Considera que las empresas pyme deben considerar el desarrollo de aplicaciones con tecnologías *cross-platform*?
    - Siempre.
    - Usualmente.
    - De vez en cuando.
    - A veces.
    - Nunca.
    - No responde.
13. De las siguientes, ¿qué metodologías de trabajo ha utilizado en sus proyectos de desarrollo?
    - SCRUM.
    - KANBAN.
    - XP-Extreme.
    - Waterfall.
14. De las siguientes, ¿qué metodología de trabajo prefiere utilizar en sus proyectos de desarrollo?
    - SCRUM.
    - KANBAN.
    - XP-Extreme.
    - Waterfall.
    - ¿Por qué? _____

15. De las siguientes, ¿qué herramientas de automatización de procesos ha utilizado en sus proyectos de desarrollo?
    - GitHub.
    - GitHub Actions.
    - Slack.
    - Docker.
    - Jenkins.

Adicionalmente, si usted así lo desea, puede compartir sus datos personales a continuación, o bien, finalizar esta encuesta, esto en caso de que, de ser necesario, le pueda contactar con más preguntas sobre el tema.

Nombre: _____

Correo electrónico: _____

8.1.4 Apéndice D – Entrevista a administradores de proyectos

ENTREVISTA A ADMINISTRADORES DE PROYECTOS

Buen día. Mi nombre es Marni Andrei López López, y estoy realizando el proyecto de investigación denominado “Análisis de las tecnologías de desarrollo *cross-platform* existentes en el mercado y propuesta de un proceso de desarrollo automatizado aplicable a una pyme enfocada en el desarrollo de aplicaciones multiplataforma”, como requisito de graduación de la carrera de Ingeniería del Software.

Agradezco su participación para responder. La información suministrada será tratada bajo los principios de confidencialidad y con fines exclusivos de la investigación.

Instrucciones: Proceda a contestar cada pregunta según su criterio.

1. De las siguientes, ¿qué metodologías de administración de proyectos ha aplicado en los proyectos en los que ha trabajado?
    - SCRUM.
    - KANBAN.
    - WATERFALL.
    - XP-Xtreme.

2. ¿Cuál de las siguientes prácticas considera más fácil de implementar en un proceso de desarrollo de *software*?
    - SCRUM.
    - KANBAN.
    - WATERFALL.
    - XP-Xtreme.
3. ¿Qué herramientas de administración de proyectos de desarrollo utiliza?
    - Azure DevOps.
    - Jira.
    - Otra: _____.
4. De los proyectos en los que ha ejercido como administrador de proyecto, ¿qué porcentaje estima que eran proyectos de desarrollo móvil híbrido?
    - _____%.
5. De los proyectos en los que ha ejercido como administrador de proyecto, y que fueron desarrollados con tecnologías híbridas, ¿qué tecnología(s) se utilizaron para trabajar en ellos?
    - React Native.
    - Xamarin.
    - Ionic.
    - Flutter.
    - Ninguna de las anteriores.
6. De las siguientes, ¿cuáles tecnologías de desarrollo *cross-platform* considera usted que tienen la curva de aprendizaje más pronunciada para un equipo de desarrollo con ingenieros de nivel intermedio que no han utilizado nunca una de estas tecnologías?
    - React Native.
    - Xamarin.
    - Ionic.
    - Flutter.
    - Ninguna de las anteriores.

7. ¿Considera usted que la empresa debe invertir tiempo, dinero y recursos en ayudar a los ingenieros de un equipo a superar la curva de aprendizaje que conlleva aprender una nueva tecnología?
    - Sí.
    - No.
    - No responde.
    - _____%
8. De acuerdo con su criterio, ¿cree que es importante contar de forma interna con automatización de procesos de desarrollo, como implementaciones y otros?
    - Sí.
    - No.
    - No responde.
9. ¿Considera usted que se deben utilizar herramientas de automatización de procesos para realizar procesos repetitivos en los proyectos de manera automática?
    - Sí.
    - No.
    - No responde.
10. ¿Qué herramientas de automatización de procesos ha utilizado en sus proyectos?
    - _____.
11. ¿Cuál es su experiencia en desarrollo de aplicaciones móviles?
    - Más de 5 años.
    - Entre 1 y 5 años
    - Menos de 1 año.
12. ¿Cuál es su nivel de formación académica?
    - Postgrado universitario completado.
    - Grado universitario completado.
    - Universitario en progreso.
    - Bachiller completado.

Adicionalmente, si usted así lo desea, puede compartir sus datos personales a continuación, o bien, finalizar esta encuesta, esto en caso de que, de ser necesario, le pueda contactar con más preguntas sobre el tema.

Nombre: _____

Correo electrónico: _____

CARTA DE AUTORIZACIÓN DE USO

Universidad Técnica Nacional

Anexo III

CARTA DE AUTORIZACIÓN PARA USO Y MANEJO DE LOS TRABAJOS FINALES DE GRADUACIÓN UNIVERSIDAD TÉCNICA NACIONAL

(Trabajo Individual)

Alajuela, Costa Rica

27 de Noviembre 2023

Señores/as: Vicerrectoría de Investigación. Sistema Integrado de Bibliotecas y Recursos Digitales

Estimados señores/as:

Yo Marni Andrei Lopez Lopez portador (a) de la cédula de identidad número 1 – 1623 0677. En mi calidad de autor (a) del trabajo de graduación titulada:

Análisis de las tecnologías de desarrollo Cross-Platform existentes en el mercado y propuesta de un proceso de desarrollo automatizado utilizando herramientas DevOps aplicable a una PYME enfocada en el desarrollo de aplicaciones multiplataforma.

El cual se presenta bajo la modalidad de, marque una opción:

_____ Proyecto de Graduación

___X___ Tesis de Graduación

Presentado en la fecha 22/11/2023, autorizo a la Universidad Técnica Nacional, sede Central, para que mi trabajo pueda ser manejado de la siguiente manera:

AUTORIZO Ver capítulo V, disposiciones finales, artículo 41 (O aquel que refiera a derechos patrimoniales)	
Marque con una X o un ✓	
Conservación de ejemplares para préstamo y consulta física en biblioteca.	✓
Inclusión en el catálogo digital del SIBIREDI (Cita catalográfica).	✓
Comunicación y divulgación a través del Repositorio Institucional.	✓
Resumen (Describe en forma breve el contenido del documento)	✓
Consulta electrónica con texto protegido	✓
Descarga electrónica del documento en texto completo protegido	✓
Inclusión en bases de datos y sitios web que se encuentren en convenio con la Universidad Técnica Nacional contando con las mismas condiciones y limitaciones aquí establecidas.	✓
Divulgación del resumen en el Repositorio UTN, con una cantidad de 200 a 500 palabras	✓

Por otra parte, declaro que el trabajo que aquí presento es de plena autoría, es un esfuerzo realizado de forma personal, académica e intelectual con plenos elementos de originalidad y creatividad. Garantizo que no contiene citas, ni transcripciones de forma indebida que puedan devenir en plagio, pues se ha utilizado la normativa vigente de la American Psychological Association (APA). Las citas y transcripciones utilizadas se realizan en el marco de respeto a las obras de terceros. La responsabilidad directa en el diseño y presentación son de competencia exclusiva, por tanto, eximo de toda responsabilidad a la Universidad Técnica Nacional.

Consciente de que las autorizaciones no reprimen mis derechos patrimoniales como autor del trabajo. Confío en la que Universidad Técnica Nacional respete y haga respetar mis derechos de propiedad intelectual.

Firma del estudiante: MARNI ANDREI LOPEZ LOPEZ (FIRMA) Digitally signed by MARNI ANDREI LOPEZ LOPEZ (FIRMA) Date: 2023.11.27 03:09:12 -05'00'

Cédula: 1 – 1623 0677

Día: 27 de Noviembre de 2023

(Reformado mediante Acuerdo 9-3-2021, tomado por el Consejo Universitario en la Sesión Ordinaria No. 3-2021, celebrada el jueves 11 de febrero de 2021, a las nueve horas, según el Artículo 12. Publicado en el diario oficial La Gaceta No. 39 del 25 de febrero del 2021, sección de Reglamentos).