UNIVERSIDAD MAYOR DE SAN ANDRÉS

FACULTAD DE CIENCIAS PURAS Y NATURALES

CARRERA DE INFORMÁTICA

PROYECTO DE GRADO

"APLICACIÓN MÓVIL PARA EL RASTREO, CONTROL Y COMUNICACIÓN DE DISPOSITIVOS INTELIGENTES CASO: TOTAL RADIO SYSTEMS LTDA"

PARA OPTAR AL TÍTULO DE LICENCIATURA EN INFORMÁTICA

MENCIÓN: INGENIERÍA DE SISTEMAS INFORMÁTICOS

POSTULANTE: VLADIMIR BONY CHACOLLA CONDORI

TUTOR METODOLÓGICO: LIC. RODRIGUEZ RAMIREZ GROVER

ASESOR: LIC. JUAN GONZALO CONTRERAS CANDIA

LA PAZ – BOLIVIA

2020

UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE CIENCIAS PURAS Y NATURALES
CARRERA DE INFORMÁTICA

LA CARRERA DE INFORMÁTICA DE LA FACULTAD DE CIENCIAS PURAS Y NATURALES PERTENECIENTE A LA UNIVERSIDAD MAYOR DE SAN ANDRÉS AUTORIZA EL USO DE LA INFORMACIÓN CONTENIDA EN ESTE DOCUMENTO SI LOS PROPÓSITOS SON ESTRICTAMENTE ACADÉMICOS.

LICENCIA DE USO

El usuario está autorizado a:

a) visualizar el documento mediante el uso de un ordenador o dispositivo móvil.
b) copiar, almacenar o imprimir si ha de ser de uso exclusivamente personal y privado.
c) copiar textualmente parte(s) de su contenido mencionando la fuente y/o haciendo la referencia correspondiente respetando normas de redacción e investigación.

El usuario no puede publicar, distribuir o realizar emisión o exhibición alguna de este material, sin la autorización correspondiente.

Dedicado a:

Dios quien siempre me guio y acompaño en todo este recorrido que es la vida y siempre espero estar bajo su resguardo.

Mis Padres por siempre apoyarme en los momentos más difíciles de mi vida quienes si no fuera por ellos no llegaría a ser la persona que ahora soy, ambos son los pilares de mi vida y la inspiración de todo mi esfuerzo gracias a dios por darme la guía de mi Papá Bonifacio y la instrucción de mi mamá Ursula espero tenerlos mucho tiempo a mi lado.

AGRADECIMIENTOS

No podría haber logrado este proyecto sin la colaboración, guía y apoyo de muchas personas en mi vida entre ellas amigos y familiares.

A mi Papá Bonifacio por su guía, enseñanza, apoyo en todo este tiempo quien me enseño que andar detrás de alguien no es ir a su sombra, si no aprender mucho más de esa persona y no cometer sus errores.

A mi Mamá quien me inculco valores desde pequeño, quien instruyo y me dio un motivo para superarme cada día más en la vida y llegar a ser un profesional y un buen Hombre de bien.

A mis hermanas Ana, Pamela, Cony quienes me apoyaron y aconsejaron siempre.

A mis amigos quienes me guiaron y apoyaron con la finalidad de terminar este proyecto, Michael amigo desde Escuela quien me aconsejo y apoyo por el cual no habría tenido la oportunidad de realizar este proyecto, Ingeniero Héctor que me guío y oriento, Sofía la persona que me ayudo y acompaño en este transcurso de mi vida y amigos de la universidad que me colaboraron con su guía y consejos.

Agradecer a mi tutor Lic. Grover Rodríguez Ramírez quien con sus observaciones y correcciones logró sacar lo mejor de mí para la culminación de este proyecto.

A mi asesor Lic. Juan Gonzalo Contreras Candia que más que mi asesor fue un gran amigo que me impulso a terminar este proyecto.

A la Ingeniera Adriana Vaca quien me abrió las puertas de su empresa, quien me dio la oportunidad de realizar este proyecto y desenvolverme laboralmente, así también a los amigos que ahí conocí, Total Radio Systems gracias.

Y por último y no menos importante a la carrera de Informática en donde crecí como persona y profesionalmente, en especial al licenciado Ramiro Flores quien fue mi docente y apoyo en la Universidad.

INDICE

Índice de Tablas

Índice de Figuras

Resumen

El presente proyecto de grado titulado Aplicación móvil para el rastreo, control y comunicación de dispositivos inteligentes. Caso Total Radio Systems. Ltda elaborado para el departamento de sistemas cuya misión es la comunicación de dispositivos inteligentes mediante notificaciones, eventos y así también el registro de usuarios y su geolocalización en tiempo real para las necesidades de la empresa.

El proyecto de grado encara el problema de los gastos excesivos en el servicio que la empresa proporciona al tener el servicio de monitoreo trunking de vehículos, para el desarrollo del proyecto se tomó como referencia la metodología Mobile–D pues brinda un enfoque de desarrollo de software específico para dispositivos inteligentes y así también para recursos web, que con la implementación de Servicios Rest Api se logra un acceso desde cualquier plataforma que trabaje con http conectándose con la base de datos de Sql Server y consumiendo desde una Aplicación hecha en Flutter que brinda trabajo en código nativo 100% para plataformas Android y iOS. Posteriormente se realizó la estimación de los costos y beneficios para ello se usó el modelo COCOMO II (Constructive Cost Model) para estimar el coste, esfuerzo y tiempo del desarrollo del mismo y los indicadores VAN (Valor Actual Neto), C/B (Costo-Beneficio) y TIR (Tasa Interna de Retorno) para predecir beneficios futuros, Para la parte de seguridad se tomó en cuenta las recomendaciones de OWASP (Open Web Application Security Project) para determinar el grado de riesgo a vulnerabilidades que puede llegar a tener el Sistema.

Palabras Clave: Trunking, Servicios Rest Api, Flutter, Recursos Web.

Capítulo I

Marco Referencial

1. Introducción

El avance de la tecnología es abrumador, tanto en nuevas tendencia como así también en dispositivos móviles y el acceso de la población a estos, todo este movimiento tecnológico permite a las personas estar rodeadas de este avance, debido a lo cual se proponen soluciones a problemas que en circunstancias diferentes serian difíciles de resolver, en la actualidad dichos dispositivos se han convertido en un ordenador personal, el cual es útil en el manejo diario al transportar con toda comodidad convirtiendo se así en dispositivos inteligentes.

Los actuales dispositivos inteligentes en diversos usos tienen una amplia tecnología que proporcionan información ya sea para la localización de personas como el GPS, lo cual incluye lo que es la georreferenciación y geolocalización, que ofrece un control mediante puntos exactos de su ubicación.

Por otro lado, la tecnología GPS abre camino a un ámbito de nuevas tecnologías, debido al uso de la tecnología GPS y Api Rest, tenemos la información de los presentes que son emisores y receptores en los teléfonos móviles.

El uso de tecnologías avanzadas (google maps) permite el acceso a la información de nuestras calles, contando así con estas herramientas se puede mantener un estricto control del usuario móvil tanto que puede servir para la geolocalización en tiempo real de un transportista en un caso más específico un radio taxi, el hecho de que no exista una plataforma móvil que brinde información actualizada y que permita a las compañías de radio taxis mantenerse informados acerca de las rutas que siguen dentro de nuestra ciudad de La Paz, también dar una clara respuesta de si el empleado vehicular esta libre o en carrera con un pasajero o quizá fuera de servicio, tomando también la disposición de un medio de comunicación chat, da lugar al presente proyecto.

Se ha decidido desarrollar de una aplicación para dispositivos inteligentes, la cual ofrezca información al momento, manteniéndolos un histórico de ellos y la capacidad de comunicarse de notificaciones en tiempo real.

1.2. Antecedentes del tema

Actualmente, la empresa cuenta con una plataforma Web que, entre sus diversas funciones, proporciona servicios de gestión, localización e información del estado actual de los vehículos como ser: posición actual, último reporte, ubicación geográfica, reportes en general, entre otros.

La necesidad de geolocalización de vehículos se hace en base a módems que la empresa compra por lo que llega ser un gasto enorme aparte que requiere de energía y la conexión y configuración a un vehículo.

Por consiguiente, se ve la necesidad de desarrollar una aplicación Android para la empresa Total Radio Systems Ltda., que sea de acceso rápido, además de ofrecer información en tiempo real del estado actual, la ubicación, el recorrido, que tenga la capacidad de notificar al cliente de los eventos de alta prioridad para que éstos tengan un mayor control sobre sus vehículos y la comunicación entre los usuarios.

1.2.1. Antecedentes institucionales

Total Radio Systems nació en el Grupo Chasqui Comunicaciones hace 20 años con el objetivo de brindar servicios de radio comunicación troncalizada en las principales ciudades de Bolivia. El Grupo Chasqui Comunicaciones fundado en 1987 instaló cientos de sistemas de comunicación a lo largo del país, implementó entre otras cosas el primer y único sistema buscapersonas en Bolivia

Actualmente Total Radio Systems es la única empresa en el país que a partir del año 2002 ofreció el servicio de AVL (Localización Vehicular Automática) Rastreo Vehicular sobre radio, utilizando la plataforma de comunicación Troncalizada.

1.2.2. PROYECTOS SIMILARES

A continuación, se describirán algunos trabajos relacionados que se tendrán como referencia durante el desarrollo del presente proyecto:

Tabla 1 Tabla De proyectos afines

Fuente: Repositorio U.M.S.A – Informática

Lugar:	La Paz, Bolivia Año: 2013
Institución académica:	Universidad Mayor de San Andrés (UMSA)
Título:	Sistema de monitoreo y control vehicular
Autor:	Henry Coarite Mamani
Descripción:	El proyecto consiste en la conexión de un microcontrolador PIC16F876A con un dispositivo móvil con sistema operativo Android v2.3.5 o superior vía bluetooth, esto para enviar los datos (posición, temperatura, encendido o apagado del motor, entre otros) a un servidor e interpretar estos datos en una página Web.

Tabla 2 Tabla de Proyectos afines

Fuente: Repositorio U.M.S.A – Informática

	Tabla de Proyectos Similares
Lugar:	La Paz, Bolivia Año: 2016
Institución académica:	Universidad Mayor de San Andrés (UMSA)
Título:	Internet de las cosas, control y seguimiento de un automóvil
Autor:	Iván Quispe Choque
Descripción:	El proyecto se basa en la implementación de una placa Arduino junto a un Shield SIM908 (ranura para chip GSM con GPS incluido) en un vehículo, además de una aplicación Android que permita realizar el rastreo del mismo. El vehículo puede enviar mensajes en caso de alejarse del dispositivo, además de recibir mensajes para cortar la alimentación de la energía o del combustible vía SMS

Tabla 3 Tabla de Proyectos Similares

Fuente: Repositorio Universidad de Aquino – Carrera de Sistemas

	Tabla de Proyectos Similares
Lugar:	Santa Cruz, Bolivia Año: 2011
Institución académica:	Universidad de Aquino – Bolivia
Título:	Sistema de monitoreo con bloqueo digital de vehículos y servicios avanzados para empresas del transporte público interprovincial e interdepartamental.
Autor:	Laura María Sarmiento Becerra
Descripción:	El proyecto tiene el fin de desarrollar una aplicación Android de rastreo vehicular para el transporte interdepartamental, además de procesos de control de conducta del chofer (nivel de alcohol, control de ruta, entre otros) para bloquear el vehículo en caso de ser necesario. Esta aplicación tiene cobertura a nivel nacional

En la ciudad de La Paz actualmente hay una cifra de 96 empresas funcionando como radio taxis y solo de ellas 76 cumplieron con los requisitos y están brindando servicios legalmente en el Municipio de La Paz, los usuarios pidieron a la alcaldía un listado de estas empresas para tener un mayor control y seguridad al usar este servicio, por tanto como muchas de estas empresas están compitiendo en un macro distrito como el municipio de La Paz y el Alto se requiere de un servicio completo y de confianza en donde cada uno está creciendo en base a su calidad de servicio por lo tanto la competencia va creciendo con las nuevas tecnologías optan por contratar los servicios de Total Radio Systems teniendo los siguientes requerimientos o problemas:

- Teniendo varios empleados vehiculares no se puede hacer un seguimiento de su ubicación de cada uno de ellos en tiempo real sin el coste de módems.
- Al contar las empresas operadoras de radio taxi con un volumen de empleados y clientes es difícil saber si un empleado está en servicio, libre, fuera de servicio para el operador a la hora de designar un taxi para un cliente.

- Las limitaciones de los componentes hacen que sea difícil una comunicación y el reporte de su ubicación, así también los costos por cada empleado aumentan.
- Las operadoras de radio taxi requieren una respuesta inmediata para el buen servicio de la operadora al no contar con ello el cliente queda insatisfecho.

1.3. Planteamiento del problema

Con lo expuesto anteriormente se define el planteamiento del siguiente problema central dentro de la empresa:

¿Cómo hacer un seguimiento de geolocalización a los clientes de las operadoras de radio taxi, gestionar los estados de cada vehículo en tiempo real, así también reducir el costo para este servicio y brindar un servicio más óptimo y eficiente?

1.4. Objetivos

1.4.1. Objetivo general

Desarrollar una aplicación móvil para la geolocalización, registro de ubicación y notificación de eventos en tiempo real para la empresa Total Radio Systems Ltda.

1.4.2. Objetivos específicos

- Implementar un servicio para la actualización en tiempo real de la ubicación del dispositivo en el Servidor.
- Establecer un historial de eventos de situación para el dispositivo para los casos, en carrera, fuera de servicio, libre, etc.
- Elaborar un módulo de notificaciones por el cual el usuario pueda controlar y almacenar los estados del dispositivo.
- Desarrollar una interfaz para la autentificación de usuarios y su registro respectivo.

1.5. Justificación

1.5.1. Justificación técnica

El presente proyecto se justifica técnicamente ya que los requerimientos para el desarrollo, implementación y mantenimiento de la aplicación en Flutter no presentan problema alguno, pues no se precisan grandes requisitos del hardware y software del ordenador, teniendo los siguientes casos:

Tabla 4 Tabla de requerimientos mínimos de desarrollo

Fuente: Jaime Hernández Instalación y desarrollo de aplicaciones móviles (2018)

Servidor:	Servidor dedicado (propiedad de la empresa).
Procesador:	Core (TM) i5 de 3.2GHz o superior
Memoria RAM:	4Gb. RAM o superior.
Espacio:	30 Gb. Mínimo de espacio en disco para el software de desarrollo.
Software:	● Sistema operativo Microsoft Windows 7 o superior. ● SQL Server edición Express 2008 o superior. ● Android Studio con SDK 29 mínimo. ● Flutter 1.17,5 ● Navegador Web cualquiera. ● Python 3 ● Flask ● SQLAlchemy
Hardware:	Hardware básico de un ordenador.
Requerimientos dispositivo móvil:	Dispositivo móvil o emulador con sistema operativo Android 4.2.2 (Jelly Bean) o superior.

1.5.2. Justificación económica

Para el desarrollo del presente proyecto se tomará en cuenta las actuales herramientas con las que trabaja la empresa, por lo que se tendrá en cuenta los siguientes

aspectos: la base de datos, la aplicación Flutter, y el Rest API teniendo el siguiente detalle de costos:

Tabla 5 Tabla de costos para el desarrollo del proyecto

Fuente: Empresa Total Radio Systems (2019)

HERRAMIENTA	DETALLE	PRECIO
Android Studio	Entorno de desarrollo proporcionado por Google de manera gratuita para la programación de aplicaciones Android.	Gratis (https://developer.android.com/studio)
Visual Studio Code	Es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código	Gratis (https://code.visualstudio.com/
Flutter	Flutter es un framework de código abierto desarrollado por Google para crear aplicaciones nativas de forma fácil, rápida y sencilla. Su principal ventaja radica en que genera código 100% nativo para cada plataforma, con lo que el rendimiento y la UX es totalmente idéntico a las aplicaciones nativas tradicionales.	Gratis (https://flutter-es.io/)
SQL Server Express	Sistema de gestión y manejo de bases de datos donde se gestionará la información de la aplicación.	Gratis (https://www.microsoft.com/en-us/sql-server/sql-server-editions-express)
Play Store	Plataforma de distribución digital de aplicaciones móviles para los dispositivos con sistema operativo Android (optativo).	$25 USD. - 174,25 Bs.

	(https://play.google.com/apps/publish/signup/)
TOTAL	$25 USD. - 174,25 Bs.

1.5.3. Justificación social

La aplicación Flutter de geolocalización, control y notificación de eventos vehiculares será para el beneficio de todos los clientes de la empresa Total Radio Systems Ltda. ya que será una plataforma de acceso rápido que proporcionará información en tiempo real de los eventos vehiculares registrados, además de la ubicación en donde se registre dicho evento, de manera que los clientes de la empresa tengan un mejor control de sus vehículos.

1.6. Alcances

La presente aplicación Android proporcionará a los clientes lo siguientes servicios:

- Ubicación en tiempo real del dispositivo inteligente
- Notificación de eventos prioritarios por falta de señal o desconexión
- Manejo de eventos desde el dispositivo, para estados ocupado en carrera fuera de servicio, etc.
- Registro y Autentificación en el dispositivo.
- Medio de comunicación segura mediante chats entre usuarios de la aplicación
- Instalación tanto en Android como iOS

Capítulo II

Marco Teórico

2.1. Marco Institucional

La empresa Total Radio Systems Ltda. nace en el grupo Chasqui comunicaciones con el objetivo de brindar servicios de radio comunicación troncalizada (Trunking), localización satelital de vehículos (AVL), telecontrol y telemedida (SCADA) y seguridad electrónica (implementación de paneles, alarmas y sensores) a nivel nacional.

El grupo Chasqui Comunicaciones fue fundado el año 1987, esta instaló una variedad de sistemas de comunicación a lo largo del país, implementó entre otras cosas el primer y único sistema buscapersonas en Bolivia.

El año 1996 es la primera empresa que introduce el concepto de comunicación troncalizada en las principales ciudades de Bolivia ofreciendo equipos profesionales que respondan a las necesidades de comunicación fiable, privada, segura y con alta calidad de audio debido a que trabaja en la frecuencia de 800 MHz a diferencia de otros sistemas convencionales de voz.

En el transcurso del año 2002 se empieza a incorporar el servicio de rastreo vehicular sobre radio, utilizando la plataforma de comunicación troncalizada. Con el avance de la tecnología en los operadores celulares, desde agosto del 2006 el servicio de localización vehicular automatizada o AVL migró a la plataforma de celular GPRS con cobertura nacional e internacional.

Posteriormente se implementa el servicio de telecontrol telemedida (SCADA), el cual realiza el monitoreo remoto de diferentes sensores y maquinas centralizando la información en la plataforma de la empresa, esto permite al cliente monitorear de forma automática, estadística y por control a distancia.

Ampliando los servicios y productos que ofrece la empresa, el año 2009 se lanza el servicio de seguridad electrónica con una variedad de soluciones para sistemas CCTV,

además de otras soluciones electrónicas incluyendo paneles de seguridad con capacidad de administrar distintos sensores.

El año 2014 se incursiona en el monitoreo y detección de incendios brindando el servicio, ingeniería y equipos que cumplen con normas internacionales.

Finalmente, el año 2016 se implementa por primera vez en Bolivia el servicio mejorado de radio comunicación troncalizada digital de la mano de Kenwood mediante el novedoso sistema Nexedge, servicio que trabaja bajo su licencia de operaciones y apuesta a revolucionar las comunicaciones digitales en el país.

Actualmente la empresa Total Radio Systems Ltda. cuenta con oficinas en el eje troncal del país, siendo el caso de La Paz, Cochabamba y Santa Cruz para ofrecer estos servicios a las distintas personas en general.

2.1.1. Misión

Brindar servicios en telecomunicaciones con soluciones integrales, innovadoras y de calidad, con tecnologías de vanguardia, personal capacitado, contribuyendo al control total de las operaciones y logística de nuestros clientes para así consolidarnos como empresa líder en el país.

2.1.2. Visión

Liderar el mercado nacional de telecomunicaciones en los servicios que se ofrece actualmente a través del desarrollo humano, la investigación y el avance tecnológico optimizando los servicios actuales y ofertando nuevos servicios en beneficio del sector empresarial y público de Bolivia.

2.1.3. Organigrama de la empresa

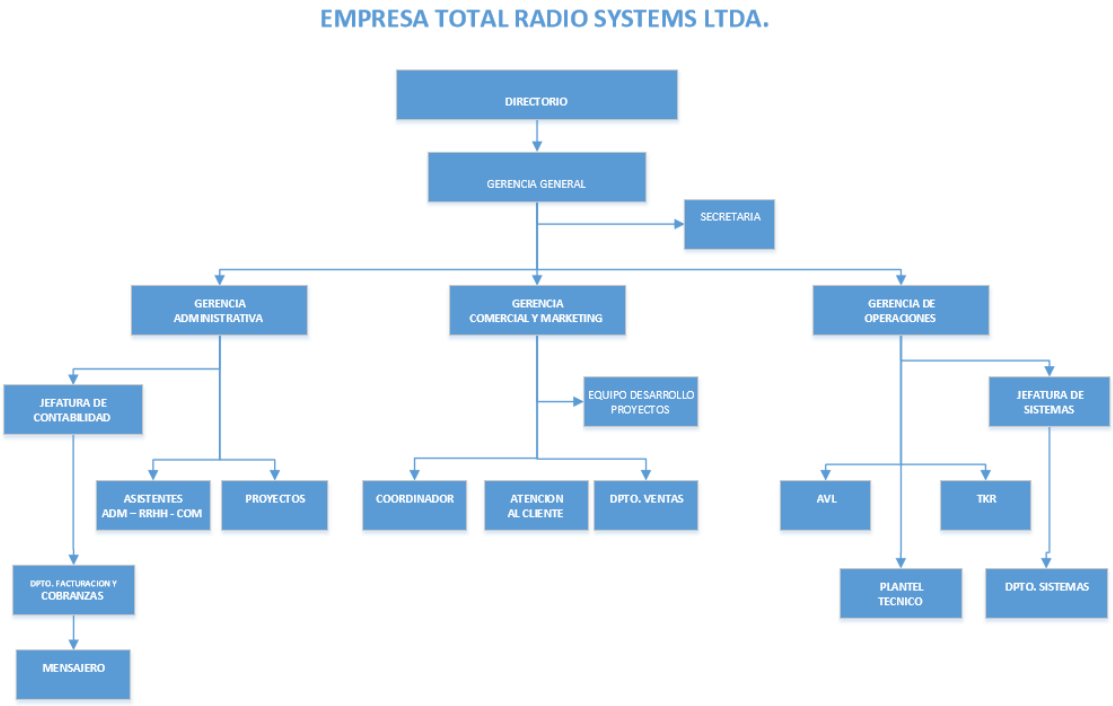


Figura 1 Organigrama de la empresa

Fuente: Organigrama empresa Total Radio System (2019) https://totalradios.com/pagina-2/

Los involucrados para el manejo de la aplicación son los siguientes:

- Jefatura de sistemas: Es el encargado de todo lo que comprende el área de sistemas, éste se encarga de coordinar los proyectos entre la gerencia de operaciones con el departamento de sistemas.
- Departamento de sistemas: Comprende a todos los encargados del área de sistemas en la empresa.
- Clientes de la empresa: Son todos aquellos clientes que requerirán contratar los servicios de la empresa.

2.2. Metodologías

A continuación, se describe las metodologías que se utilizarán en el desarrollo del presente proyecto de grado.

2.2.1. Metodología Ágil Mobile D

Metodología de desarrollo de aplicaciones móviles, parte como creación del proyecto “ICARUS” en el 2004, posee cualidades de muchas otras metodologías como ser eXtreme Programming, Crystal Methodologies y Rational Unified Process. (Agile, 2008) 34 Las ventajas de esta metodología son las siguientes: -Un costo bajo al realizar un cambio en el proyecto. - Entrega resultados de manera rápida. -Asegura el software adecuado en el momento adecuado. La metodología también cuenta con las siguientes desventajas: -No sirve para grupos de desarrollos grandes y segmentados. - Depende de buena comunicación entre los miembros del equipo. Mobile-D tiene el objetivo de ser una metodología de resultados rápidos, con mira a grupos de pocas personas o pequeños grupos, los integrantes del grupo deben poseer una habilidad y capacidad similar entre todos. (Alipknot, 2014) Posee iteraciones en la fase de producto donde la entrada a la segunda iteración de la fase de producto es el resultado de la iteración 0 y todo está controlado bajo un control de versión para el proyecto.

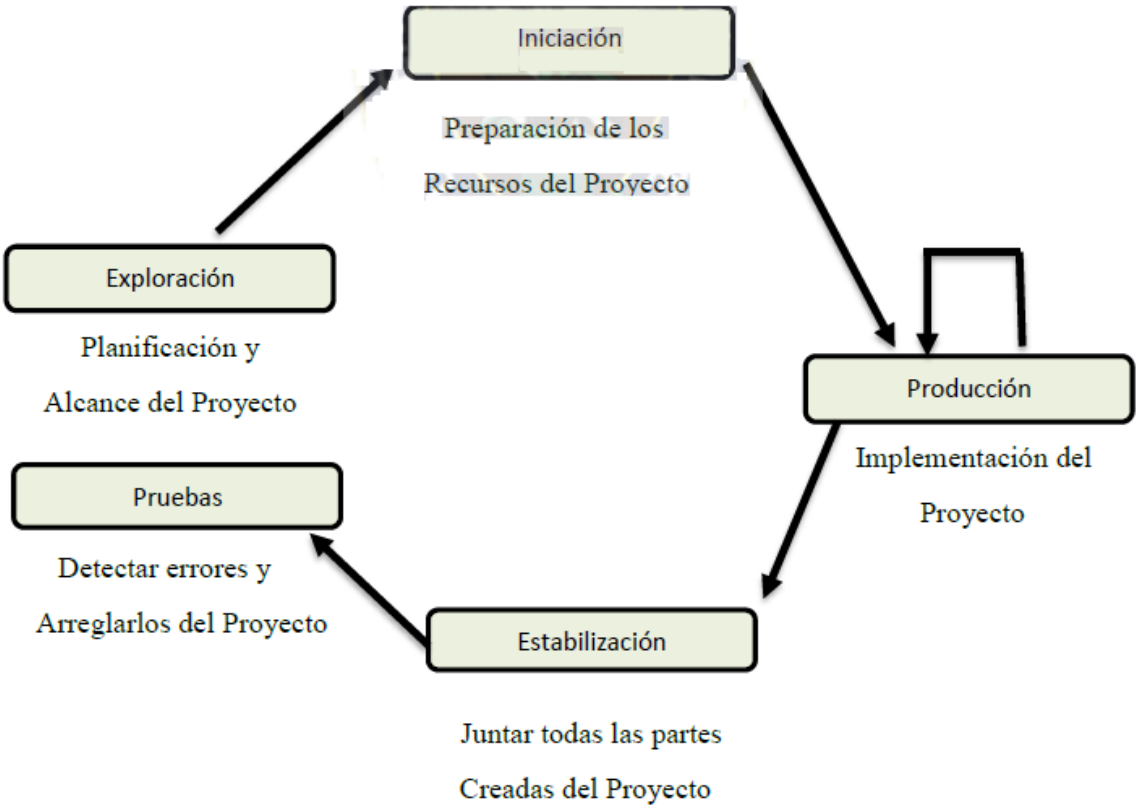


Figura 2 Diagrama Movile-D

Fuente: Metodología Para El Desarrollo De Aplicaciones Móviles "Mobile-D" David Felipe Hernández Zuleta Luego. Edición 2016

2.2.1.1. Fases De La Metodología

La metodología cuenta con 5 fases por las cuales pasa el producto a realizarse, la línea de producción empieza con la fase de exploración, después pasa a la fase de Iniciación, luego pasa a la fase de producto posteriormente a la fase de estabilización y la fase de pruebas. (Agile, 2008)

a) **FASE DE EXPLORACIÓN** Se centra la atención en la planificación y a los conceptos básicos del proyecto. Aquí es donde hacemos una definición del alcance del proyecto y su establecimiento con las funcionalidades donde queremos llegar.

b) **FASE DE INICIACIÓN** Configuramos el proyecto identificando y preparando todos los recursos necesarios y en esta fase la dedicaremos un día a la planificación y el resto al trabajo y publicación.

c) **FASE DE PRODUCCIÓN** Se repiten iterativamente las subfases, antes de iniciar el desarrollo de una funcionalidad debe existir una prueba que verifique su funcionamiento. En esta fase podemos decir que se lleva a acabo toda la implementación.

d) **FASE DE ESTABILIZACIÓN** Después de la fase de producto llega la fase de estabilización en la que se realizan las acciones de integración para enganchar los posibles módulos separados en una única aplicación.

e) **FASE DE PRUEBAS** Una vez parado totalmente el desarrollo se pasa una fase de testeo hasta llegar a una versión estable según lo establecido en las primeras fases por el cliente. Si es necesario se reparan los errores, pero no se desarrolla nada nuevo.

2.3. Lenguaje Unificado De Modelado UML

El lenguaje unificado de modelado (Unified Modeling Language – UML) es un lenguaje gráfico que permite a los creadores de sistemas generar diseños que capturen las ideas en forma convencional y fácil de comprender para visualizar, especificar, construir, documentar y comunicarlas a otras personas.

UML ofrece un estándar para describir un plano o modelo del sistema, incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados.

2.3.1. Conceptos orientados a objetos en UML

Los objetos en UML son entidades del mundo real que existen a nuestro alrededor, en el desarrollo de software, los objetos se pueden usar para describir, o modelar, el sistema que se está creando en términos que sean pertinentes para el dominio.
Los objetos también permiten la descomposición de sistemas complejos en componentes comprensibles que permiten que se construya una pieza a la vez.
Algunos conceptos fundamentales son:

- Objetos: representan una entidad y el componente básico.
- Clase: plano de un objeto.
- Abstracción: comportamiento de una entidad del mundo real.
- Encapsulación: mecanismo para enlazar los datos y ocultarlos del mundo exterior.
- Herencia: permite crear nuevas clases a partir de una existente.
- Polimorfismo: propiedad por la que es posible enviar mensajes iguales a objetos de tipos distintos.

2.3.2. Tipos de diagramas UML

Los diagramas en UML pueden clasificarse en diagramas estructurales y diagramas de comportamiento, dentro de esta clasificación podemos mencionar los siguientes diagramas:

2.3.2.1. Diagramas UML estructurales

- Diagrama de clases: es el diagrama más usado y la base principal de toda solución orientada a objetos; en este diagrama se indican las clases dentro de un sistema, sus atributos, operaciones, y la relación entre cada clase.

- Diagrama de componentes: muestra la relación estructural de los elementos del sistema, son frecuentemente empleados cuando se trabaja con sistemas complejos de componentes múltiples. Los componentes se comunican por medio de interfaces.
- Diagrama de estructura compuesta: los diagramas de estructura compuesta se usan para mostrar la estructura interna de una clase.
- Diagrama de implementación: ilustra el hardware del sistema y su software, es de utilidad cuando se implementa una solución de software en múltiples máquinas con configuraciones únicas.
- Diagrama de objetos: muestra la relación entre objetos por medio de ejemplos del mundo real e ilustra cómo se verá un sistema en un momento dado.
- Diagrama de paquetes: existen dos tipos especiales de dependencias que se definen entre paquetes: la importación de paquetes y la fusión de paquetes. Los paquetes pueden representar los diferentes niveles de un sistema para revelar la arquitectura, se pueden marcar las dependencias de paquetes para mostrar el mecanismo de comunicación entre niveles.

2.3.2.2. Diagramas UML de comportamiento

- Diagramas de actividades: son utilizados para mostrar una secuencia de actividades, muestran el flujo de trabajo desde el punto de inicio hasta el punto final detallando muchas rutas de decisiones que existen en el progreso de eventos contenidos en la actividad.
- Diagrama de secuencia: es una forma de diagrama de interacción que muestra los objetos como líneas de vida a lo largo de la página y con sus interacciones en el tiempo representadas como mensajes dibujados como flechas desde la línea de vida origen hasta la línea de vida destino.
- Diagrama de comunicación: similar a los diagramas de secuencia, pero el enfoque está en los mensajes que se pasan entre objetos, la misma información se puede representar usando un diagrama de secuencia y objetos diferentes.

- Diagrama de estados: similar a los diagramas de actividades, describen el comportamiento de objetos que se comportan de diversas formas en su estado actual.
- Diagrama de temporización: Al igual que en los diagramas de secuencia, se representa el comportamiento de los objetos en un período de tiempo dado.

Diagrama de caso de uso: captura los requisitos del sistema, los casos de uso son un medio de comunicación con los usuarios y otros interesados acerca de lo que se piensa hacer del sistema.

- **Diagrama de Casos de Uso**: Los diagramas de casos de uso documentan el comportamiento de un sistema desde el punto de vista del usuario. Por lo tanto, los casos de uso determinan los requisitos funcionales del sistema, es decir, representan las funciones que un sistema puede ejecutar.
- **Actores**: El actor es una entidad externa al sistema que de alguna manera participa en la historia del caso de uso. Por lo regular, estimula el sistema con eventos de entrada o recibe algo de él. Los actores están representados por el papel que desempeñan en el caso: cliente, técnico u otro. Conviene escribir su nombre con mayúscula en la narrativa del caso para facilitar la identificación. En la figura 3. se muestra la representación gráfica del actor.

Figura 3 Representación de un actor

Fuente: (Larman, 2004)

- **Caso de uso:** "El caso de uso es un documento narrativo que describe la secuencia de eventos de un actor (agente externo) que utiliza un sistema para completar un proceso", Booch, Jacobson & Rumbaugh (2012) los casos de uso son historias o casos de utilización de un sistema, no son exactamente los requerimientos, ni las especificaciones funcionales, si no que ejemplifican e

incluyen tácticamente los requerimientos en las historias que narran. Generalmente se los representa mediante un óvalo, como se muestra en la figura 4.

Figura 4 Representación Gráfica caso de uso

Fuente: (Booch, Jacobson & Rumbaugh, 2012)

- **Relación**: Si un caso de uso inicia o contiene el comportamiento de otro se dice que usa el segundo caso, eso es una relación unidireccional. Esta relación puede presentar uno de los siguientes tipos:
  - Include: Se puede incluir una relación entre dos casos de uso de tipo "include" si se desea especificar comportamiento común en dos o más casos de uso. En la figura 5 se puede ver la forma de representar esta relación.

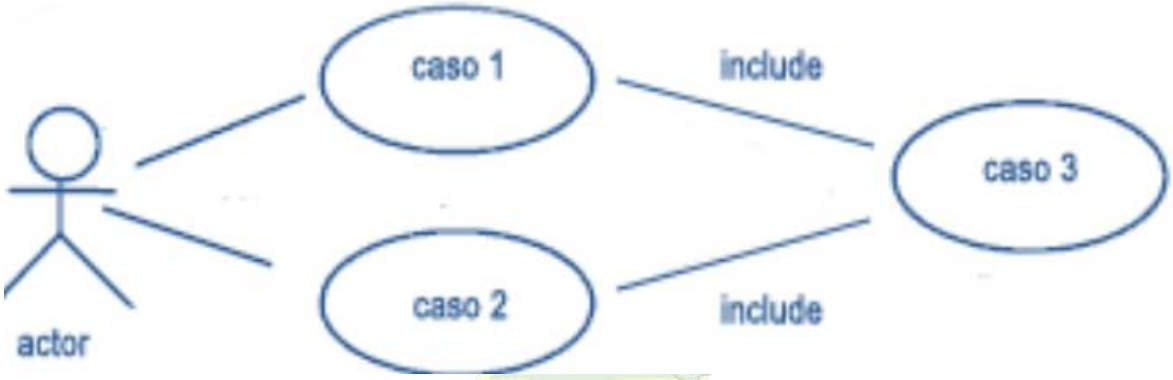


Figura 5 Representación gráfica de relación include

Fuente: (Booch, Jacobson & Rumbaugh, 2012)

  - **Extend:** Se puede incluir una relación entre dos casos de uso de tipo "include" si se desea especificar diferentes variantes del mismo caso de uso; es decir, esta relación implica que el comportamiento de un caso de uso es diferente dependiendo de ciertas circunstancias. En principio esas variaciones pueden también mostrarse como diferentes descripciones de escenarios asociadas al

mismo caso de uso. Un ejemplo de este tipo de relación se puede ver en la figura 6,

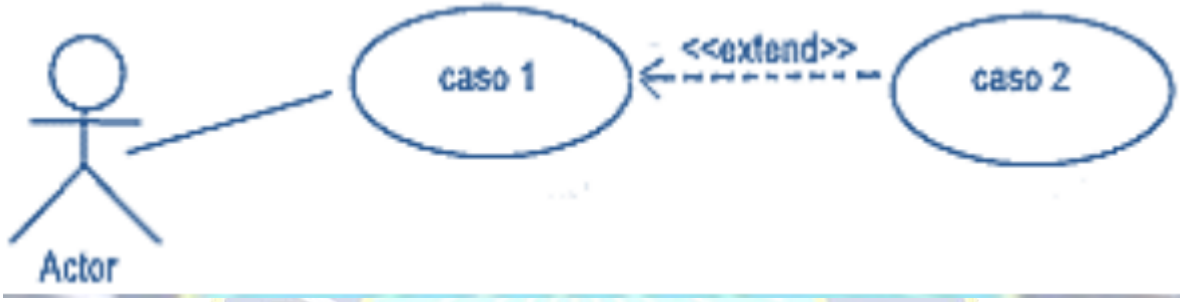


Figura 6 Representación de la relación extend

Fuente: (Booch, Jacobson & Rumbaugh, 2012)

- **Prototipo de Interfaz de Usuario:** implica la creación de un modelo o modelos operativos del trabajo de un sistema, en el que analistas y clientes deben estar de acuerdo. (Dinámico/ los usuarios se hacen participantes activos en el desarrollo).
- **Revisión del diseño preliminar /Análisis y Diseño Preliminar:** En esta fase a partir de cada caso de uso se obtendrán una ficha de caso de uso, está formada por un nombre, una descripción, una precondición que debe cumplir antes de iniciarse, una pos condición que debe cumplir al terminar si termina correctamente.
- **Diagrama de Robustez**: Según Fernández & Sumano (2004) representa el flujo básico y alterno de cada uno de los casos de uso que conforman el sistema, permite pasar del análisis al diseño del mismo. Esta herramienta permite capturar el Que hacer y a partir de eso él Como hacerlo. Facilita el reconocimiento de objetos y hace más sencilla la lectura del sistema. En la figura 7 se muestra los componentes que conforman un diagrama de robustez.

Figura 7 Elementos del diagrama de robustez

Fuente: (Fernández & Sumano, 2004)

El diagrama de robustez se divide en:

- Objetos fronterizos: usado por los actores para comunicarse con el sistema, son objetos con los cuales puede interactuar el usuario – interfaz de usuario.
- Objetos entidad: son objetos del modelo del dominio.
- Objetos de Control: es la unión entre la interfaz y los objetos de entidad.

La figura 8 muestra la forma como está conformado un diagrama de robustez.

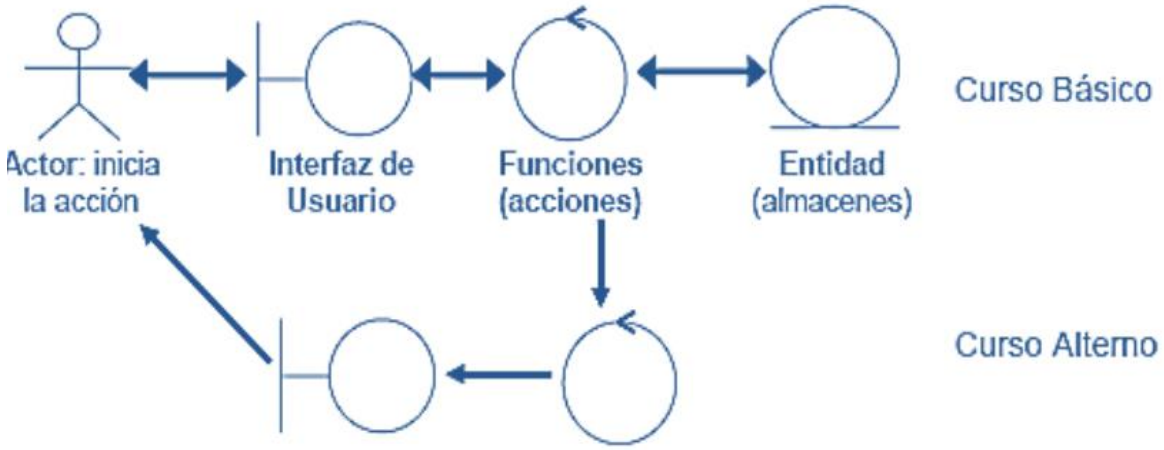


Figura 8 Representación gráfica de un diagrama de robustez

Fuente: (Fernández & Sumano, 2004)

- Implementación: Según Rosenberg & Collins (2005) en esta etapa se debe entregar una vista preliminar del sistema a través de la utilización de los diagramas de despliegue, de estados, componentes, así también se debe escribir el código, realizar las pruebas unitarias y las de integración, para que al final de todo esto se pueda proceder a su entrega.

2.4. Flutter

Flutter es un framework de código abierto desarrollado por Google para crear aplicaciones nativas de forma fácil, rápida y sencilla. Su principal ventaja radica en que genera código 100% nativo para cada plataforma, con lo que el rendimiento y la UX es totalmente idéntico a las aplicaciones nativas tradicionales.

En el mercado de desarrollo de aplicaciones móviles para iOS y Android hay una gran cantidad de frameworks o herramientas que permiten utilizar un mismo código fuente para ambas plataformas. Pero ninguna genera código 100% nativo.

La principal y más importante ventaja de Flutter es que desarrollas un solo proyecto para todos los sistemas operativos, lo que significa una reducción de costes y tiempo de producción.

2.4.1. Funcionalidades de Flutter

Calidad nativa: Las aplicaciones nativas se desarrollan específicamente para un sistema operativo, Flutter utiliza todas las ventajas de las aplicaciones nativas para conseguir calidad en el resultado final.

Experiencia de usuario: Flutter incluye Material Design de Google y Cupertino de Apple, con lo que la experiencia de usuario es óptima y los interfaces de usuario idénticos a los de las aplicaciones desarrolladas por las propias compañías.

Tiempo de carga: Una de las principales causas de abandono de una aplicación es el tiempo que tarda en cargar, con Flutter se experimentan tiempos de carga por debajo de un segundo en cualquiera de los soportes iOS o Android.

Desarrollo ágil y rápido: Gracias a la característica hot-reload, puedes programar y ver los cambios en tiempo real en tu dispositivo o en los simuladores.

Otra de las ventajas de utilizar este framework es que da igual el sistema operativo que utilices, ya que puedes descargarlo para Windows, Mac o Linux desde este enlace.

Seguimos contando beneficios al utilizar Flutter, el framework tiene una comunidad en la que podrás compartir tus conocimientos, experiencias y/o buscar ayuda para tus dudas o problemas puntuales.

Flutter al ser un sistema operativo diseñado principalmente para dispositivos inteligentes tiene características y especificaciones que van acorde al funcionamiento de éstos, como ser:

Tabla 6 Características del Framwork Flutter.

Fuente: Api Flutter documentación - https://api.flutter.dev/javadoc/

Diseño de dispositivo	La plataforma es adaptable a pantallas de resolución VGA, gráficos 2D y 3D.

Almacenamiento	Base de almacenamiento de datos para dispositivos móviles SQLite.
Conectividad	GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, HSDPA, HSPA+, NFC y WiMAX, GPRS, UMTS y HSDPA+.
Mensajería	SMS, MMS, además del servicio de Firebase Cloud Messaging (FCM) siendo la nueva versión de Google Cloud Messaging (GCM) con los nuevos SDK para realizar el desarrollo de mensajería en la nube mucho más sencillo.
Navegador web	Está basado en el motor de renderizado de código abierto WebKit, emparejado con el motor JavaScript V8 de Google Chrome (se usa el navegador de Ice Cream Sandwich).
Soporte de Java	Aunque la mayoría de las aplicaciones están escritas en Java, no hay una máquina virtual Java en la plataforma. El bytecode Java no es ejecutado, sino que primero se compila en un ejecutable Dalvik y corre en la Máquina Virtual Dalvik8.
Soporte multimedia	3GP, MPEG-4, AMR, AAC, MP3, MIDI, WAV, JPEG, PNG, GIF, BMP, ente otros.
Soporte para streaming	Streaming RTP/RTSP (3GPP PSS, ISMA), descarga progresiva de HTML5. Adobe Flash Streaming (RTMP) es soportado mediante el Adobe Flash Player.
Soporte para hardware adicional	Android soporta cámaras de fotos, de vídeo, pantallas táctiles, GPS, acelerómetros, giroscopios, magnetómetros, sensores de proximidad y de presión, sensores de luz, gamepad, termómetro, aceleración por GPU 2D y 3D.
Entorno de desarrollo	Actualmente se considera a Android Studio como entorno oficial de desarrollo.
Google Play	Es un catálogo de aplicaciones gratuitas o de pago en el que pueden ser descargadas en dispositivos Android.
Multi-táctil	Android tiene soporte nativo para pantallas capacitivas con soporte multi-táctil.
Bluetooth	El soporte para A2DF y AVRCP fue agregado en la versión 1.5, el envío de archivos (OPP) y la exploración del directorio telefónico fueron agregados en la versión 2.0; y el marcado por voz junto con el envío de contactos entre teléfonos lo fueron en la versión 2.2.
Video llamada	Android soporta videollamada a través de Hangouts (ex-Google Talk).
Multitarea	Las aplicaciones que no estén ejecutándose en primer plano reciben ciclos de reloj.
Características basadas en voz	La búsqueda en Google a través de voz está disponible como "Entrada de Búsqueda".

Tethering	Android soporta tethering, que permite al teléfono ser usado como un punto de acceso alámbrico o inalámbrico. Permite a un PC usar la conexión de datos del móvil Android (se podría requerir la instalación de software adicional).

2.4.2. Versiones de Android

En el desarrollo de aplicaciones nativas, Flutter trabaja con el código nativo de Android creando un app 100% nativo, por lo cual toma las mismas versiones de Android en su SDK de Android.

Actualmente se han desarrollado múltiples versiones de este sistema operativo en busca de corregir bugs o fallos de versiones anteriores, éstas se han caracterizado ya que a lo largo de su versionamiento han recibido nombres de postres ordenados alfabéticamente, teniendo los siguientes casos: (andro4all.com/2018/08/versiones-android-historia | 2018)

Tabla 7Versiones del sistema operativo Android

Fuente: andro4all.com/2018/08/versiones-android-historia (2018)

VERSIÓN	NOMBRE CÓDIGO	TRADUCCIÓN	FECHA DISTRIBUCIÓN	API
1	Apple Pie	Tarta de manzana	23 de septiembre de 2008	1
1.1	Banana Bread	Pan de plátano	9 de febrero de 2009	2
1.5	Cupcake	Panque	27 de abril de 2009	3
1.6	Donut	Rosquilla	15 de septiembre de 2009	4
2.0 - 2.1	Eclair	Pepito (pastel glaseado)	26 de octubre de 2009	5 - 7
2.2	Froyo	Yogurt helado	20 de mayo de 2010	8
2.3	Gingerbread	Pan de jengibre	9 de febrero de 2011	10
4	Ice Cream Sandwich	Sandwich de helado	16 de diciembre de 2011	14
4.1	Jelly Bean	Gominola	9 de julio de 2012	16
4.2	Jelly Bean	Gominola	13 de noviembre de 2012	17
4.3	Jelly Bean	Gominola	24 de julio de 2013	18
4.4	Kit Kat	Kit Kat (barra de chocolate)	31 de octubre de 2013	19

5	Lollipop	Piruleta (caramelo)	3 de noviembre de 2014	21
5.1	Lollipop	Piruleta (caramelo)	6 de abril de 2015	22
6	Marshmallow	Malvavisco	5 de octubre de 2015	23
7.0 - 7.1	Nougat	Turrón	22 de agosto de 2016	24 - 25
8.0 - 8.1	Oreo	Oreo (galleta)	21 de agosto de 2017	26 - 27
9.0	Pie	Pastel	6 de agosto de 2018	28
10.0	Android 10	Android 10	3 de septiembre de 2019	29

2.5. Entorno de Desarrollo Integrado – IDE

Un entorno de desarrollo integrado o también llamado entorno de desarrollo interactivo (en inglés Integrated Development Environment – IDE) es una aplicación visual informática que proporciona servicios integrales para facilitarle al programador el desarrollo de software, este consiste de un editor de código fuente, herramientas de construcción automáticas y un depurador.

La mayoría de los entornos de desarrollo están diseñados para poder desarrollar aplicaciones en más de un lenguaje de programación, sin embargo, existen también los que están creados para un único lenguaje. (Fernando Garcia – fergarciac.wordpress.com | 2013)

2.6. Android Studio

Es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Flutter y se basa en IntelliJ IDEA9, es considerado como el sustituto de Eclipse Android Development Tools (ADT), que anteriormente era el entorno principal para el desarrollo de aplicaciones para Android.

Android Studio se anunció por primera vez en la conferencia Google I/O el 16 de mayo de 2013 para reemplazar a Eclipse, donde se mostró una vista previa de la versión 0.1 y entró en la etapa beta hasta la versión 0.8 y comenzó a lanzarse en junio de 2014. La primera compilación estable, la versión 1.0, fue lanzada en diciembre de 2014. La última versión

estable es la 3.0, y fue lanzada en octubre de 2017.

(https://developer.android.com/studio/intro?hl=es-419 | 2019)

2.6.1. Versiones de Android Studio

Actualmente se cuenta con las siguientes versiones de Android Studio:

Tabla 8. Historial de versiones de Android Studio.
Fuente:https://developer.android.com/studio/intro?hl=es-419 | 2019

VERSIÓN	FECHA DE LANZAMIENTO
0.1.x	Mayo de 2013
0.2.x	Julio de 2013
0.3.2	Octubre de 2013
0.4.2	Enero de 2014
0.4.6	Marzo de 2014
0.5.2	Mayo de 2014
0.8.0	Junio de 2014
0.8.6	Agosto de 2014
0.8.14	Octubre de 2014
1.0	Diciembre de 2014
1.0.1	Diciembre de 2014
1.1.0	Febrero de 2015
1.2.0	Abril de 2015
1.2.1	Mayo de 2015
1.2.2	Junio de 2015
1.3.0	Julio de 2015
1.3.1	Agosto de 2015
1.3.2	Agosto de 2015
1.4.0	Septiembre de 2015
1.4.1	Octubre de 2015
1.5.0	Noviembre de 2015
1.5.1	Diciembre de 2015
2.0.0	Abril de 2016
2.1.0	Abril de 2016
2.1.1	Mayo de 2016
2.1.2	Junio de 2016
2.1.3	Agosto de 2016
2.2.0	Septiembre de 2016
2.2.1	Octubre de 2016
2.2.2	Octubre de 2016
2.2.3	Diciembre de 2016
2.3.0	Marzo de 2017
2.3.1	Abril de 2017
2.3.2	Abril de 2017
2.3.3	Junio de 2017

3.0	Octubre de 2017
3.1.4	Agosto de 2018

2.7. Características

Además del entorno de desarrollo y las herramientas para programadores de IntelliJ, Android Studio ofrece aún más funciones que aumentan la eficiencia y productividad durante la compilación de aplicaciones, como las siguientes:
Un sistema de compilación basado en Gradle flexible.

- Un emulador rápido con varias funciones de los dispositivos inteligentes.
- Un entorno que permite desarrollar aplicaciones para todos los dispositivos con sistema operativo Android.
- Instant Run permite aplicar cambios mientras la aplicación se ejecuta sin la necesidad de compilar un nuevo APK
- Integración de plantillas de código y GitHub para ayudar a compilar funciones comunes de las aplicaciones e importar ejemplos de código.
- Gran cantidad de herramientas y frameworks de prueba.
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versión, etc.
- Compatibilidad con C++ y NDK.
- Soporte incorporado para Google Cloud Platform, lo que facilita la integración de Google Cloud Messaging y App Engine.

(https://developer.android.com/studio/intro?hl=es-419 | 2019)

2.8. Aplicación Flutter

Una aplicación Flutter puede definirse como una aplicación nativa de Android o un programa informático (con características especiales) diseñada para ser ejecutada en dispositivos inteligentes, por lo general se encuentran disponibles en plataformas de distribución que son de las compañías propietarias de los sistemas operativos móviles como Android, iOS, Web, entre otros; estas pueden ser gratuitas y otras son de pago.

Las aplicaciones Flutter están escritas en algún lenguaje de programación Dart, que trabaja con el código nativo de cada sistema operativo sea Android (java-Kottlin) o iOS (Objective-C) se encuentran residentes en los dispositivos inteligentes, su funcionamiento y recursos se encaminan a aportar ventajas como:

- Un acceso más rápido y sencillo a la información necesaria sin necesidad de los datos de autenticación en cada acceso.
- Un almacenamiento de datos personales de manera interna en el dispositivo de una forma segura.
- Una gran versatilidad en cuanto a su utilización en los dispositivos.
- Mejora la capacidad de conectividad y disponibilidad de servicios y productos (usuario-usuario, usuario-proveedor de servicios, etc.).
- Son más dinámicas que los programas tradicionales.

(https://www.consumidor.ftc.gov/articulos/s0018-aplicaciones-moviles-que-son-y-como-funcionan | 2011)

2.9. Visual Studio Code

Visual Studio Code es un editor de programación multiplataforma desarrollado por Microsoft. Es un proyecto de software libre que se distribuye bajo la licencia MIT, aunque los ejecutables se distribuyen bajo una licencia gratuita no libre.
La página oficial de Visual Studio Code es https://code.visualstudio.com/. El código fuente se encuentra en GitHub https://github.com/Microsoft/vscode
La primera versión beta de Visual Studio Code se publicó en noviembre de 2015 y la primera versión estable, Visual Studio Code 1.0, se publicó en abril de 2016. Desde su aparición, Visual Studio Code ha mantenido un ritmo de desarrollo muy rápido, y se publica una nueva versión a principios de cada mes (salvo en enero). Además, casi todos los meses se publican versiones secundarias que corrigen fallos de última hora.
Actualmente (julio de 2020), la última versión publicada de Visual Studio Code es la versión 1.47, publicada el 9 de julio de 2020
https://www.mclibre.org/consultar/informatica/lecciones/vsc.html

2.9.1. Características

Visual Studio Code proporciona diversas herramientas e interfaces para el desarrollo de aplicaciones, así como diversos lenguajes de programación para su codificación. Algunos de éstos se describen de la siguiente manera:

Tabla 8 Tecnologías y lenguajes de Visual Studio Code

Fuente: Visual Studio Code
https://www.mclibre.org/consultar/informatica/lecciones/vsc.html

TIPOS DE LENGUAJES	DESCRIPCIÓN
Go	Go es un lenguaje de programación concurrente y compilado, desarrollado por los ingenieros de Google. Go vio la luz en el año 2009, esto hace a Go un lenguaje relativamente nuevo, pero que esto no nos engañe, Go es un lenguaje maduro, con el cual se han desarrollado miles de proyectos alrededor del mundo, inclusive, versiones actuales de Go están escritas con el mismo Go.
Visual C#	Visual C# (C Sharp) está diseñado para compilar una variedad de aplicaciones que se ejecutan en .NET Framework. Visual C# es simple y eficaz, ofrece seguridad de tipos y está orientado a objetos. Con sus muchas innovaciones, Visual C# permite desarrollar aplicaciones rápidamente, mantiene la expresividad y elegancia de los lenguajes tipo C.
Visual C++	Visual C++ es un lenguaje eficaz que está diseñado para proporcionar un gran control en detalle al compilar aplicaciones nativas para Windows (COM+) o aplicaciones Windows administradas mediante .NET Framework.
TypeScript.	TypeScript es un lenguaje de programación de código abierto con herramientas de programación orientada a

	objetos, muy favorable si se tienen proyectos grandes.
JavaScript	JavaScript es la siguiente generación de una implementación por parte de Microsoft del lenguaje ECMA 262. Hay disponibles muchos tipos diferentes de aplicaciones.
JSON	JSON (JavaScript Object Notation - Notación de Objetos de JavaScript) es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo. Está basado en un subconjunto del Lenguaje de Programación JavaScript, Standard ECMA-262 3rd Edition - Diciembre 1999. JSON es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos.
Python	Python es un lenguaje de scripting independiente de plataforma y orientado a objetos, preparado para realizar cualquier tipo de programa, desde aplicaciones Windows a servidores de red o incluso, páginas web. Es un lenguaje interpretado, lo que significa que no se necesita compilar el código fuente para poder ejecutarlo, lo que ofrece ventajas como la rapidez de desarrollo e inconvenientes como una menor velocidad.
PHP	PHP (acrónimo recursivo de *PHP: Hypertext Preprocessor*) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.
Dockerfile	Un Dockerfile es un archivo de texto plano que contiene una serie de instrucciones necesarias para crear una

	imagen que, posteriormente, se convertirá en una sola aplicación utilizada para un determinado propósito. Similar a lo explicado anteriormente, y la base del funcionamiento de Docker es mediante Dockerfiles
HTML	HTML es un lenguaje de marcado que se utiliza para el desarrollo de páginas de Internet. Se trata de la siglas que corresponden a HyperText Markup Language, es decir, Lenguaje de Marcas de Hipertexto

2.10. SQL Server

Microsoft SQL Server es un sistema de administración, análisis y manejo de bases de datos del modelo relacional, fue desarrollado por la empresa Microsoft el cuál propone soluciones de comercio electrónico, línea de negocio y almacenamiento de datos. El código fuente original de SQL Server que fue utilizado en las versiones previas a la versión 7.0 habría sido comprado de Sybase, pero fue actualizado en las versiones 7.0 y 2000, y reescrito en la versión 2005.

Generalmente, cada 2 a 3 años, una nueva versión es lanzada y, entre estos lanzamientos, se proponen services packs con mejoras y correcciones de bugs, por problemas urgentes en el sistema de seguridad o bugs críticos.

(https://searchdatacenter.techtarget.com/es/definicion/SQL-Server por Margaret Rouse | 2015)

2.10.1. Características

SQL Server, a comparación de otros sistemas de manejo de base de datos, posee diversas características por la que muchos se deciden a implementarlo dentro de sus empresas, entre las características más importantes podemos mencionar:

- Permite almacenamiento y gestión de datos.
- Soporte de transacciones.
- Soporta procedimientos almacenados, de manera que se puedan almacenar consultas para ser utilizadas repetidamente.

- Incluye también un entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y los terminales o clientes de la red sólo acceden a la información.
- Permite administrar información de otros servidores de datos.

(SQL SERVER VS MySQL Autores: Jose Santamaría y Javier Hernández 2010)

2.11. Calidad del software

Una vez terminada la etapa de desarrollo, el siguiente paso consiste en realizar pruebas de la aplicación Android con el objetivo de comprobar que ésta funcione correctamente.
Según Pressman (2005), la prueba basada en escenario descubrirá errores que ocurren cuando cualquier actor interactúa con el software y cuando ocurren errores asociados con una especificación incorrecta, el producto no hace lo que el cliente quiere. Puede hacer lo correcto u omitir funcionalidad importante.
La prueba de software consiste en verificar el prototipo destinado al usuario final para conocer sus cualidades, probar su eficiencia, su eficacia, verificar su funcionalidad y cómo este reacciona, o qué resultados produce. (Ingenieria del Software de Roger S. Pressman - 2011)
Para el presente proyecto de grado se utilizará el estándar de calidad denominado ISO 9126 o (ISO/IEC 9126). Mismo que se lo describe a continuación.

2.11.1. ISO 9126

El estándar ISO 9126 se desarrolló con la intención de identificar los atributos clave del software de cómputo. Este sistema identifica seis atributos clave de la calidad (Pressman, 2010):

- Funcionalidad. Grado en el que el software satisface las necesidades planteadas según las establecen los atributos siguientes: adaptabilidad, exactitud, interoperabilidad, cumplimiento y seguridad.

- Confiabilidad. Cantidad de tiempo que el software se encuentra disponible para su uso, según lo indican los siguientes atributos: madurez, tolerancia a fallas y recuperación. - Usabilidad. Grado en el que el software es fácil de usar, según lo indican los siguientes subatributos: entendible, aprendible y operable. 30
- Eficiencia. Grado en el que el software emplea óptimamente los recursos del sistema, según lo indican los subatributos siguientes: comportamiento del tiempo y de los recursos.
- Facilidad de recibir mantenimiento. Facilidad con la que pueden efectuarse reparaciones al software, según lo indican los atributos que siguen: analizable, cambiable, estable, susceptible de someterse a pruebas.
    - Portabilidad. Facilidad con la que el software puede llevarse de un ambiente a otro según lo indican los siguientes atributos: adaptable, instalable, conformidad y sustituible. Igual que otros factores de la calidad del software estudiados en las subsecciones anteriores, los factores ISO 9126 no necesariamente conducen a una medición directa. Sin embargo, proporcionan una base útil para hacer mediciones indirectas y una lista de comprobación excelente para evaluar la calidad del sistema.

2.12. Pruebas de Software

La prueba es un proceso de individualización, y el número de tipos diferentes de pruebas varía tanto como los diferentes acercamientos para su desarrollo. Como menciona (Pressman, 2006) durante muchos años, la única defensa contra los errores de programación fue el diseño cuidadoso y la inteligencia natural del programador. Ahora estamos en una era en la que modernas técnicas de diseño (y revisiones técnicas) ayudan a reducir el número de errores 57 iniciales que son inherentes al código. La prueba es un conjunto de actividades que deben de realizarse de manera sistemática.

2.13. Costo y Beneficio

Para realizar dicho estudio se utilizará la metodología COCOMO II, misma que se detalla a continuación.

2.13.1. Cocomo II

El Modelo Constructivo de Costes (COCOMO, por su acrónimo del inglés Constructive Cost Model) es un modelo matemático de base empírica utilizado para la estimación de costes de software, incluye tres submodelos, cada uno ofrece un nivel de detalle y aproximación, cada vez mayor, a medida que avanza el proceso de desarrollo del software básico, intermedio y detallado, como lo menciona Boehm, Brown y Chulani (2009).

Este modelo fue desarrollado por Barry W. Boehm a finales de los años 70 y comienzos de los 80, exponiéndolo detalladamente en su libro "Software Engineering Economics". Pertenece a la categoría de modelos de subestimaciones basados en estimaciones matemáticas. Está orientado a la magnitud del producto final, midiendo el "tamaño" del proyecto, en líneas de código principalmente.

2.13.2. Modelo de Estimación

La ecuación que se utiliza es la siguiente:

$$E = a(KI)_b * m(x), \text{ en persona mes}$$

$$Tdev = c(E)_d, \text{ en meses}$$

$$P = E / Tdev, \text{ en personas}$$

Donde:

- E es el esfuerzo requerido por el proyecto, en persona – mes.
- Tdev es el tiempo requerido por el proyecto, en meses.
- P es el número de personas requerido por el proyecto.
- a, b, c y d son constantes con valores definidos en una tabla, según cada submodelo.
- Kl es la cantidad de líneas de código, en miles.
- m(x) es un multiplicador que depende de 15 atributos.

A la vez, cada submodelo también se divide en modos que representan el tipo de proyecto, y puede ser:

- modo orgánico: un pequeño grupo de programadores experimentados desarrollan software en un entorno familiar. El tamaño de software varía desde unos pocos miles de líneas (tamaño pequeño) a unas decenas de miles (medio).
- modo semi - libre o semi – encajado: corresponde a un esquema intermedio entre el orgánico y el rígido, el grupo de desarrollo puede incluir una mezcla de personas experimentadas y no experimentadas.
- modo rígido o empotrado: el proyecto tiene fuertes restricciones, que pueden estar relacionadas con la funcionalidad y/o pueden ser técnicas. El problema a resolver es único y es difícil basarse en la experiencia, puesto que puede no haberla según Boehm, Brown y Chulani (2009).

2.13.3. Modelo de Base de Estimación

El modelo base de COCOMO II corresponde al esfuerzo de desarrollo estimado una vez que se ha fijado la arquitectura del sistema (estimación del proceso post-arquitectura). Después de presentar tal modelo base veremos cómo se ajusta el modelo para:

- Estimaciones más tempranas, correspondiente al diseño temprano (Pre-arquitectura).
- Mantenimiento.
- Estimación de número de defectos esperados.

Paso 1: Estimación de puntos de función.

Se debe considerar que cada pantalla o reporte a generar equivale a 8 puntos de función (de ahora en adelante pf).

Nota 1: Modelo más detallado.

El modelo más detallado de N, Callaos (y más cónsono con la literatura sobre puntos de función, técnicamente “feature points”) es:

- Cada pantalla de consulta vale 4pf.
- Cada pantalla que carga datos vale 5pf.
- Cada reporte vale 5pf.
- Cada tabla de una base de datos vale 10pf.

- Cada interfaz con otro sistema vale 6pf.

La simplificación de N. Callaos consiste en considerar que toda pantalla o reporte vale 5pf y que solo el 60% de los puntos de función provienen de pantalla, el resto vienen de tablas o archivos lógicos.

Paso 2: Aplicación de fórmulas básicas de esfuerzo, tiempo calendario y personal requerido.

La fórmula básica de esfuerzo ($PM_{nominal}$, person-months):

$PM_{nominal} = 2.94 * Tamaño_E * \Pi Emi$

Donde:

Tamaño se mide en KDLC.

Emi corresponde a los factores de ajuste de costo.

E es el exponente no lineal, calculando según:

$E = 0.91 + 0.01 * \Sigma WEi$

Donde:

WEi son parámetros de costo (cost-drivers)

2.13.4. Valor Actual Neto

También conocido como valor actualizado neto, cuyo acrónimo es VAN (en inglés NPV), es un procedimiento que permite calcular el valor presente de un determinado número de flujos de caja futuros, originados por una inversión. La metodología consiste en descontar al momento actual (es decir, actualizar mediante una tasa) todos los flujos de caja futuros del proyecto. A este valor se le resta la inversión inicial, de tal modo que el valor obtenido es el valor actual neto del proyecto, así lo definen Boehm, Brown y Chulani (2009).

El método de valor presente es uno de los criterios económicos más ampliamente utilizados en la evaluación de proyectos de inversión. Consiste en determinar la equivalencia en el tiempo 0 de los flujos de efectivo futuros que genera un proyecto y comparar esta equivalencia con el desembolso inicial. Cuando dicha equivalencia es mayor que el desembolso inicial, entonces, es recomendable que el proyecto sea aceptado.

La fórmula que nos permite calcular el Valor Actual Neto es:

$$VAN = \Sigma \frac{Ganancias}{(1+12\%)^n} - \Sigma \frac{Costos}{(1+22\%)^n}$$

Donde:

n: representa los flujos de caja en cada periodo.

Ganancias: es el beneficio neto actualizado, es el valor actual del flujo de caja.

Costos: es los gastos por la nueva inversión

- **Tasa Interna de Retorno. -** Es la tasa de descuento de un proyecto de inversión que permite que el valor neto actualizado sea igual a la inversión. La tasa interna de retorno es la máxima tasa de descuento que puede tener un proyecto para que sea rentable. Esta dada por la fórmula siguiente.

$$TIR = \frac{\Sigma\, Ganancias - Costos}{(1 + 3\%)^n}$$

2.14. Seguridad

La seguridad del software es una actividad que garantiza la calidad del software, se centra en la identificación y evaluación de riesgos potenciales y los mismos pueden llegar a producir impactos negativos en el sistema.

Algunas de estas pueden ser:

- Inyección SQL
- Script entre sitios
- Modificación de ingreso
- Reemplazo de identidad
- Ataques XSS
- Desencriptación del App

A la hora de desarrollar una Aplicación móvil hay que tomar en cuenta dos aspectos importantes con relación a su seguridad, y esta es la física y la otra es la lógica. A continuación, se hará una descripción de las mismas.

2.14.1. Seguridad Física

La seguridad física de los sistemas informáticos consiste en la aplicación de barreras físicas y procedimientos de control como medidas de prevención y contramedidas contra las amenazas a los recursos y la información confidencial de la Institución. La posibilidad de acceder físicamente a una computadora, en general a cualquier sistema operativo hace inútiles casi todas las medidas de seguridad que haya aplicado sobre él. Es necesario garantizar la seguridad global de la red y los sistemas conectados a ella, evidentemente el nivel de seguridad física depende completamente del entorno donde se ubiquen los puntos a proteger. Pero ¿Cómo hacerlo? ¿Cómo prevenir un acceso físico no autorizado a un determinado punto? hay soluciones de diferente tipo desde la instalación de videocámaras, pasando por tarjetas inteligentes o control con las llaves que abren determinada puerta, así también los biométricos, los más recomendados estos últimos.
Cuando la prevención es difícil por cualquier motivo (técnico, económico, humano, etc.) es deseable que un potencial ataque sea detectado cuanto antes, para minimizar así sus efectos, para ello, es importante concienciar a todo el personal su papel en la política de seguridad del entorno, si por ejemplo un usuario autorizado detecta presencia de alguien de quien sospecha que no tiene autorización para estar en una determinada estancia debe avisar inmediatamente al administrador o al responsable de los equipos, que a su vez puedan avisar al servicio de seguridad si es necesario, Hernández (2007).

2.14.2. Seguridad Lógica

Hernández (2007) sostiene que la seguridad lógica consiste en la aplicación de barreras y procedimientos que resguarden el acceso a los datos y solo se permita acceder a ellos a las personas autorizadas para ello.

2.14.2.1. Seguridad a nivel del Sistema Operativo.

Entre las medidas que se deben tomar a nivel del sistema operativo son:

- Restringir el acceso al arranque (desde el BIOS) al S.O. los programas y archivos.
- Otorgar a los usuarios permisos de solo lectura para los directorios necesarios.

- Denegar el acceso de forma predeterminada.
- Proteger el archivo de configuración de registro.
- Asegurarse de que los servicios son actuales comprobando con frecuencia si hay actualizaciones de seguridad.
- Reducir el nivel de permisos de acceso para los servicios de red.

2.14.2.2. Seguridad a nivel del Sistema de Gestión de Base de Datos

A continuación, se describen las medidas de seguridad a tomar en cuenta para la base de datos.

- Restringir los permisos a los usuarios.
- Cambiar el usuario root.
- Eliminar la cuenta de prueba y la base de datos de prueba.
- Revisar periódicamente los usuarios y la base de datos para asegurar que tienen los permisos otorgados en su momento.

2.14.2.3. Seguridad a nivel de Aplicación

Entre las medidas que se tomarán a nivel de aplicación están.

- Asignación de roles de usuario.
- Encriptación de las contraseñas de usuario.
- Asignación de privilegios.
- Generación de un log de sucesos.
- Asegurarse que esté deshabilitada la función de autocompletado en el navegador
- Creación de un Api Rest

2.12.3. Recomendaciones OWASP

El Open Web Application Security Project (OWASP) es un espacio abierto de la comunidad dedicada a la búsqueda y la lucha contra las causas del software inseguro. Todas las herramientas, documentos y los capítulos de OWASP son gratuitos y abiertos. (Wiesmann, Curphey, Stock y Stirbei, 2017).

Así también estos autores mencionan que OWASP es un nuevo tipo de entidad en el mercado de la seguridad. “Nuestra libertad de las presiones comerciales nos permite brindar información imparcial, práctica y rentable sobre seguridad de las aplicaciones”, no está afiliado con ninguna compañía de tecnología, sin embargo, apoya la utilización de tecnología de seguridad.

Las aplicaciones seguras no se dan por si mismas – son en cambio el resultado de una organización decidiendo que va a producir aplicaciones seguras. OWASP no desea forzar un enfoque particular o requerir a la organización el cumplimiento de leyes que no la afectan – cada organización es diferente.

Sin embargo, a los fines de obtener una aplicación segura, se requiere como mínimo:

Una gestión organizacional que abogue por la seguridad políticas de seguridad documentadas y apropiadamente basadas en estándares nacionales.

Una metodología de desarrollo con adecuados puntos de control y actividades de seguridad

Gestión segura de versiones y configuración.

Muchos de los controles contenidos en la Guía OWASP se encuentran influenciados por requerimientos incluidos en estándares nacionales o marcos de control tales como ISO normalmente los controles seleccionados de la guía satisfacen los requerimientos relevantes de ISO 27001 o ISO 31000 (Wiesmann, Curphey, Stock y Stirbei, 2017).

2.12.3.1. Riesgos en la Seguridad de las Aplicaciones

Los atacantes pueden, potencialmente, utilizar diferentes rutas a través de su aplicación para perjudicar su negocio u organización. Cada uno de estos caminos representa un riesgo que puede o no ser suficientemente grave como para merecer atención. En la figura 10 se muestran estas posibles amenazas.

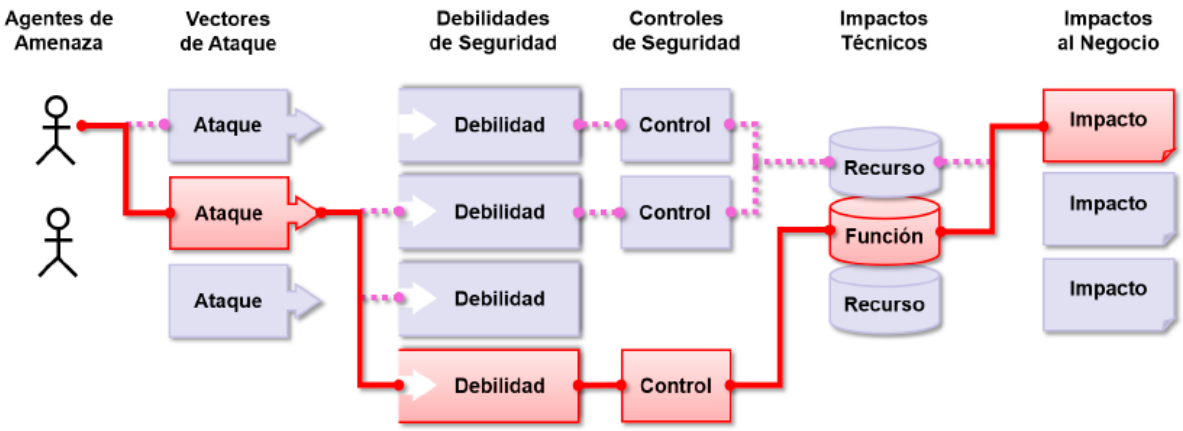


Figura 9 Riesgos en la Seguridad de las Aplicaciones

Fuente: (Wiesmann, Curphey, Stock y Stirbei, 2017)

Algunas veces, estos caminos son fáciles de encontrar y explotar, mientras que otras son extremadamente difíciles. De la misma manera, el perjuicio ocasionado puede no tener consecuencias, o puede dejarlo en la quiebra. A fin de determinar el riesgo para su organización, puede evaluar la probabilidad asociada a cada agente de amenaza, vector de ataque, debilidad de seguridad y combinarlo con una estimación del impacto técnico y de negocio, juntos, estos factores determinan su riesgo general, como se puede observar en la figura 11

Agente de Amenaza	Explotabilidad	Prevalencia de Vulnerabilidad	Detección de Vulnerabilidad	Impacto Técnico	Impacto de Negocio
Específico de la Aplicación	Fácil 3	Difundido 3	Fácil 3	Severo 3	Específico del Negocio
	Promedio 2	Común 2	Promedio 2	Moderado 2	
	Difícil 1	Poco Común 1	Difícil 1	Mínimo 1	

Figura 10 Factores que Determinan el Riesgo de Amenazas

Fuente: (Wiesmann, Curphey, Stock y Stirbei, 2017)

A continuación se describen los riesgos en la seguridad de las aplicaciones (Top 10) (Wiesmann, Curphey, Stock y Stirbei, 2017).

- **A1:2017 (Inyección)** Las fallas de inyección, como SQL, NoSQL, OS o LDAP ocurren cuando se envían datos no confiables a un intérprete, como parte de un comando o consulta. Los datos dañinos del atacante pueden engañar al intérprete para que ejecute comandos involuntarios o acceda a los datos sin la debida autorización.
- **A2:2017 (Pérdida de Autenticación)** Las funciones de la aplicación relacionadas a la autenticación y gestión de sesiones son implementadas incorrectamente, permitiendo a los atacantes comprometer usuarios y contraseñas, token de sesiones, o explotar otras fallas de implementación para asumir la identidad de otros usuarios (temporal o permanentemente).
- **A3:201 (Exposición de Datos Sensibles)** Muchas aplicaciones web y APIs no protegen adecuadamente datos sensibles, tales como información financiera, de

salud o información personalmente identificable (PII). Los atacantes pueden robar o modificar estos datos protegidos inadecuadamente para llevar a cabo fraudes con tarjeta de crédito, robos de identidad u otros delitos. Los datos sensibles requieren métodos de protección adicionales, como el cifrado en almacenamiento y tránsito.

- **A4:2017 (Entidades Externas XML (XXE))** Muchos procesadores XML antiguos o mal configurados evalúan referencias a entidades externas en documentos XML. Las entidades externas pueden utilizarse para revelar archivos internos mediante la URI o archivos internos en servidores no actualizados, escanear puertos de la LAN, ejecutar código de forma remota y realizar ataques de denegación de servicio (DoS).
- **A5:2017 (Pérdida de Control de Acceso)** Las restricciones sobre lo que los usuarios autenticados pueden hacer no se aplican correctamente. Los atacantes pueden explotar estos defectos para acceder, de forma no autorizada, a funcionalidades y/o datos, cuentas de otros usuarios, ver archivos sensibles, modificar datos, cambiar derechos de acceso y permisos, etc.
- **A6:2017 (Configuración de Seguridad Incorrecta)** La configuración de seguridad incorrecta es un problema muy común y se debe en parte a establecer la configuración de forma manual, ad hoc o por omisión (o directamente por la falta de configuración). Son ejemplos, S3 buckets abiertos, cabeceras HTTP mal configuradas, mensajes de error con contenido sensible, falta de parches y actualizaciones, frameworks, dependencias y componentes desactualizados, etc.
- **A7:2017 (Secuencia de Comandos en Sitios Cruzados (XSS))** Los XSS ocurren cuando una aplicación toma datos no confiables y los envía al navegador web sin una validación y codificación apropiada, o actualiza una página web existente con datos suministrados por el usuario utilizando una API que ejecuta JavaScript en el navegador. Permiten ejecutar comandos en el navegador de la víctima y el atacante puede secuestrar una sesión, modificar (defacement) los sitios web, o re direccionar al usuario hacia un sitio malicioso.
- **A8:2017 (Deserialización Insegura)** Estos defectos ocurren cuando una aplicación recibe objetos serializados dañinos y estos objetos pueden ser manipulados o borrados por el atacante para realizar ataques de repetición,

inyecciones o elevar sus privilegios de ejecución. En el peor de los casos, la deserialización insegura puede conducir a la ejecución remota de código en el servidor.

- **A9:2017 (Componentes con Vulnerabilidades Conocidas)** Los componentes como bibliotecas, frameworks y otros módulos se ejecutan con los mismos privilegios que la aplicación. Si se explota un componente vulnerable, el ataque puede provocar una pérdida de datos o tomar el control del servidor. Las aplicaciones y API que utilizan componentes con vulnerabilidades conocidas pueden debilitar las defensas de las aplicaciones y permitir diversos ataques e impactos.

A10:2017 (Registro y Monitoreo Insuficientes) El registro y monitoreo insuficiente, junto a la fatal de respuesta ante incidentes permiten a los atacantes mantener el ataque en el tiempo, pivotear a otros sistemas y manipular, extraer o destruir datos. Los estudios muestran que el tiempo de detección de una brecha de seguridad es mayor a 200 días, siendo típicamente detectado por terceros en lugar de por procesos internos.

Capítulo III

Marco Aplicativo

3.1. Introducción

Una vez realizado el análisis sobre la situación actual de la empresa Total Radio Systems Ltda., se pudo observar que la empresa cuenta con una plataforma Web que, entre sus diversas funciones, ofrece información en general del estado actual de los vehículos, esto resulta ser de poca utilidad ya que el cliente sólo podrá tener acceso a dicha información mientras éste se encuentre en la plataforma ocasionando molestias y pérdida de confianza hacia la empresa.

Por esta razón surge la necesidad de desarrollar una aplicación en flutter que proporcione información en tiempo real del estado actual, los eventos prioritarios, la ubicación geográfica de los vehículos, además que sea de fácil acceso, mediante el uso de tecnología

Android implementando Cloud Computing para los clientes de la empresa Total Radio Systems Ltda.

Para llevar a cabo el presente proyecto se tomará en cuenta diversos puntos, estos permitirán desarrollar la aplicación Flutter acorde a las necesidades del cliente como también utilizar las herramientas disponibles de la empresa, estos puntos son los siguientes:

- Los recursos disponibles de la empresa en cuanto a hardware y software.
- Los requerimientos solicitados por el cliente.
- Los requerimientos para la gestión y supervisión de la información.

El proyecto tomará como referencia a la metodología (MOVILE-D) junto al Lenguaje Unificado de Modelado (UML), la cual posee una estructura dinámica permitiendo que éste sea un proceso iterativo, esta metodología implica el desarrollo de cuatro fases principales:

- Fase de inicio.
- Fase de elaboración.
- Fase de desarrollo.
- Fase de cierre.

También se implementará el modelo en cascada[13], este es un enfoque metodológico que ordena rigurosamente las etapas del proceso durante el desarrollo de software, estas son las siguientes:

- Análisis de requisitos.
- Diseño del sistema.
- Diseño del programa.
- Codificación.
- Pruebas.
- Implementación del programa.
- Mantenimiento.

3.2. Fase de inicio

La empresa Total Radio Systems Ltda. cuenta con una plataforma Web que ofrece información en general del estado de los vehículos, esto resulta de poca utilidad para el

cliente ya que implica que éste permanezca constantemente pendiente de la plataforma para saber el estado actual de sus vehículos, por esto surge la necesidad de desarrollar una aplicación Flutter que sea de fácil acceso para el cliente, además proporcione información en tiempo real de los distintos eventos que reportan los vehículos registrados en dispositivos GPS que tienen incorporados. Los puntos que describen el alcance y objetivos que se pretende lograr con el proyecto han de ser especificados de manera más detallada en el documento de Visión del proyecto.

3.2.1. Especificación de requerimientos

La especificación de requerimientos nos permite analizar las necesidades que tienen los usuarios finales de la aplicación Flutter, estas deberán ser recopiladas por medio de herramientas como ser: observación, entrevistas y encuestas a un grupo aleatorio de Clientes, ya que éstos serán los que le den uso a esta aplicación.
Una vez obtenidos se procede a la elaboración del documento de especificación de requerimientos éstos se clasifican como requerimientos funcionales y no funcionales:

- Requerimientos funcionales: describe los requerimientos que el cliente solicita en cuanto a función, estos deberán ser implementados en la aplicación y en sus componentes, estos son los siguientes:

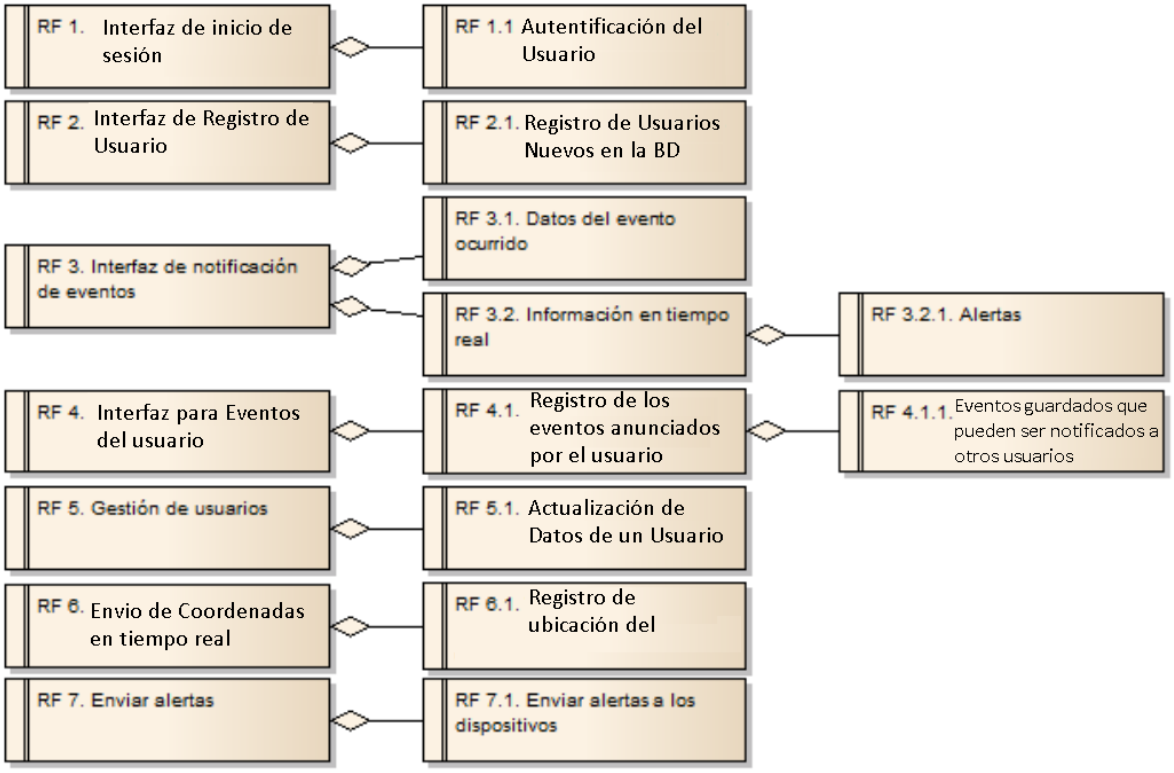


Figura 11 Diagrama de requerimientos funcionales

Fuente: Elaboración propia.

- Requerimientos no funcionales: los requerimientos no funcionales definen las restricciones o condiciones que el cliente necesita en la aplicación Flutter, éste precisa contar con los siguientes:

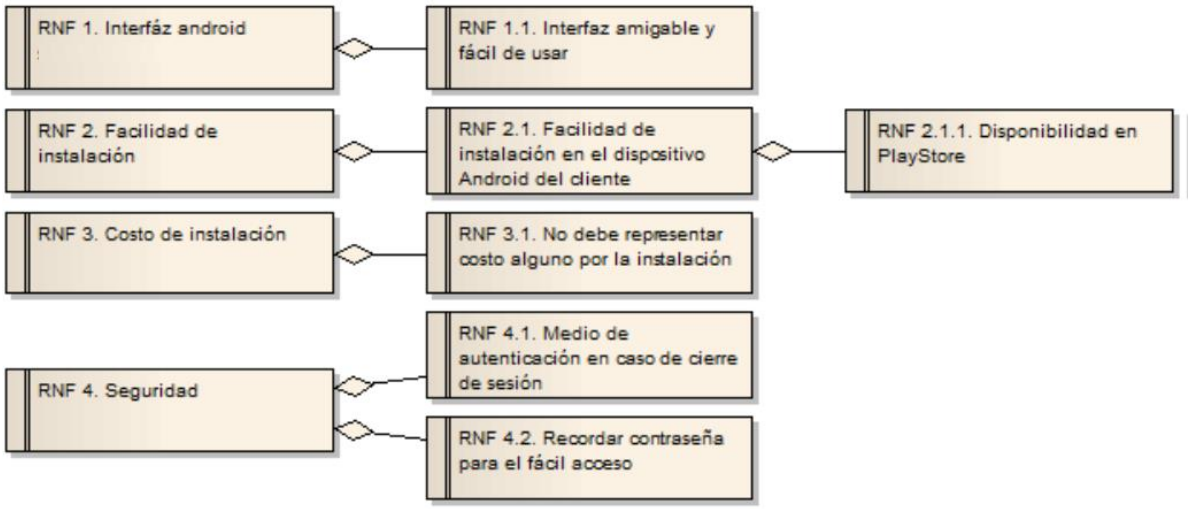


Figura 12 Diagrama de requerimientos no funcionales.

Fuente: Elaboración propia.

3.2.2. Diagrama de Casos de Uso

Teniendo definidos los requerimientos de la empresa y del usuario final o cliente, podemos realizar la elaboración de los diagramas de casos de uso, estos nos permiten obtener una descripción gráfica de puntos como ser:

- Los actores involucrados, destacando los siguientes:
- El Usuario: el usuario es un miembro de una empresa cliente que solicita el servicio de monitoreo vehicular.
- Departamento de sistemas: son los encargados de gestionar la información que se visualizará en la aplicación Android.
- Las actividades que necesitan llevar a cabo dentro de la aplicación.

A continuación, se tienen los siguientes diagramas de casos de uso:

- Diagrama general de casos de uso: El diagrama de casos de uso general muestra una descripción general de las actividades que realizarán los usuarios involucrados en el desarrollo de la aplicación Android.

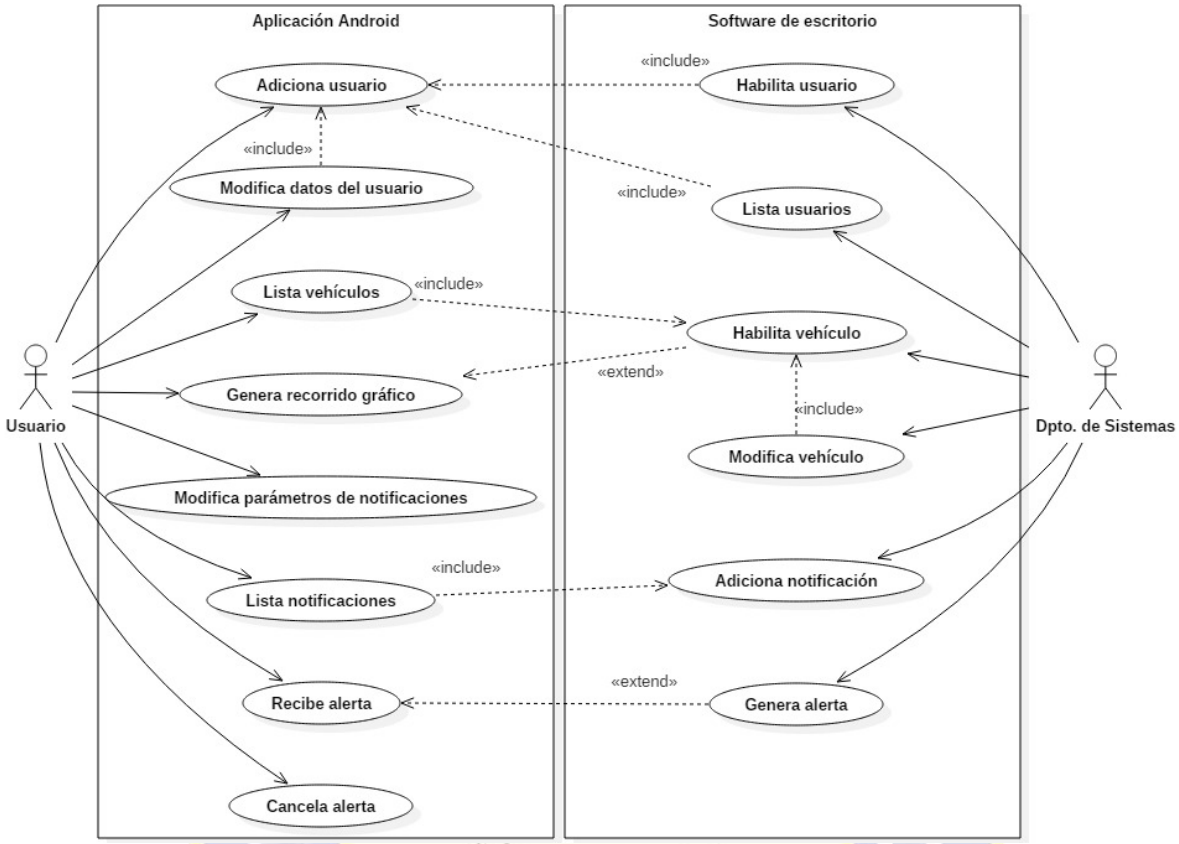


Figura 13 Diagrama general de casos de uso.

Fuente: Elaboración propia.

- Gestión de usuario: Este diagrama muestra los pasos de la gestión de usuarios donde el usuario se registra o adiciona en el sistema mientras el departamento de sistemas se encarga de habilitarlo y listar los usuarios existentes.

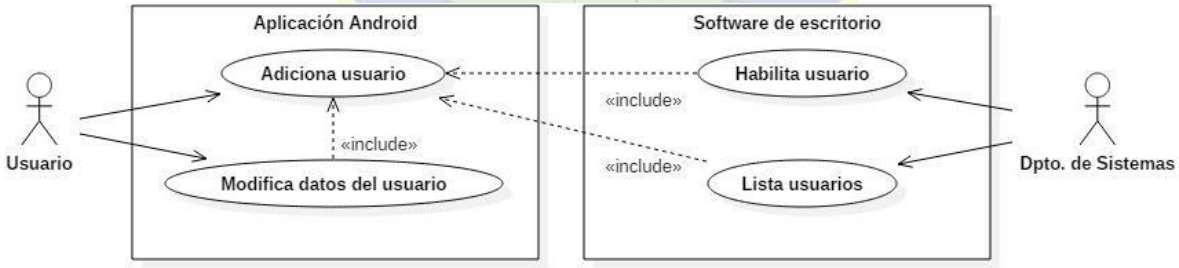


Figura 14 Caso de uso: gestión de usuario

Fuente: Elaboración propia.

- Gestión de vehículos: Este diagrama muestra lo referido a la gestión de vehículos donde el departamento de sistemas es quien realiza los cambios a la información

de los vehículos para que el usuario pueda verlos por medio de la interfaz Android.

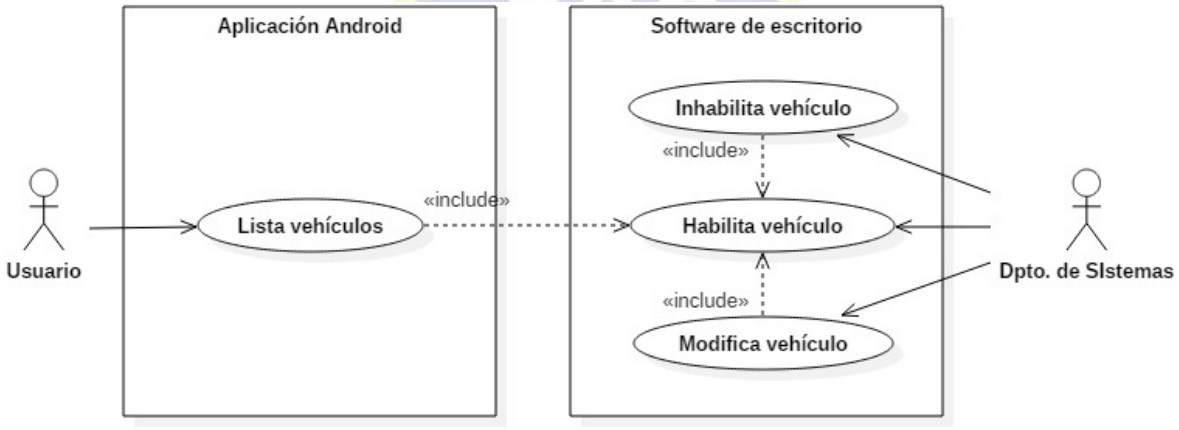


Figura 15 Caso de uso: gestión de vehículos

Fuente: Elaboración propia

- Gestión de notificaciones: El usuario puede configurar los parámetros de las notificaciones y con estos datos el departamento de sistemas le envía la información de los eventos vehiculares.

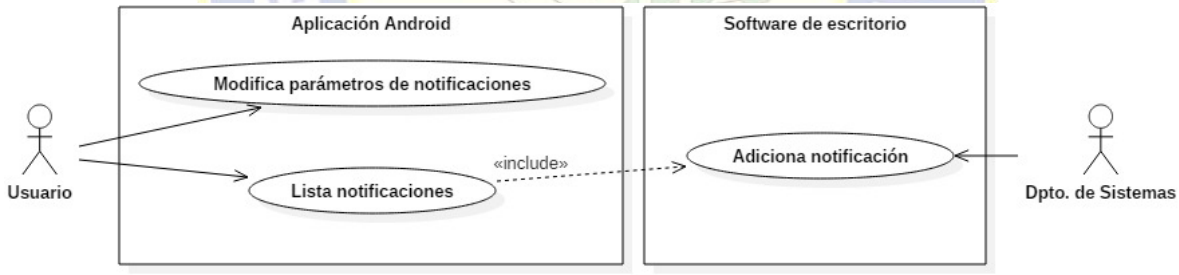


Figura 16 Caso de uso: gestión de notificaciones

Fuente: Elaboración propia.

- Gestión de alertas: Una vez configurados los parámetros de las notificaciones las alertas se generan de forma automática por medio de un desencadenador en la base de datos para que éstas sean enviadas cuando se registren en el sistema.

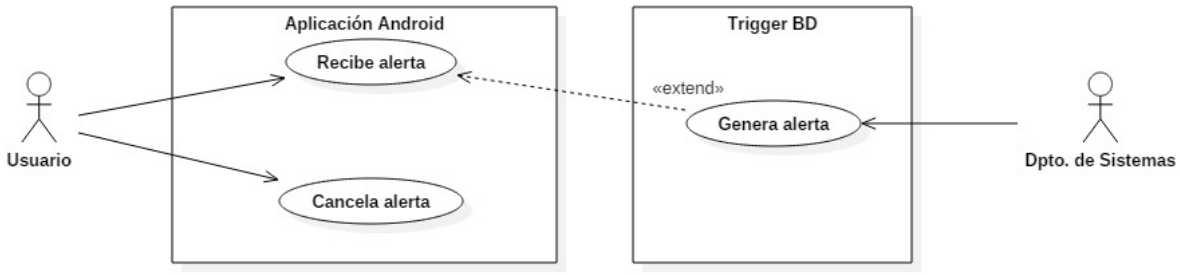


Figura 17 Caso de uso: gestión de alertas

Fuente: Elaboración propia.

- Gestión de recorrido: El departamento de sistemas cuando asigna un vehículo a un usuario, éste empieza a recibir los datos en el sistema de los lugares en donde circula, esto se conoce como el recorrido del vehículo. El usuario puede visualizar el recorrido de sus vehículos para saber en dónde han reportado.

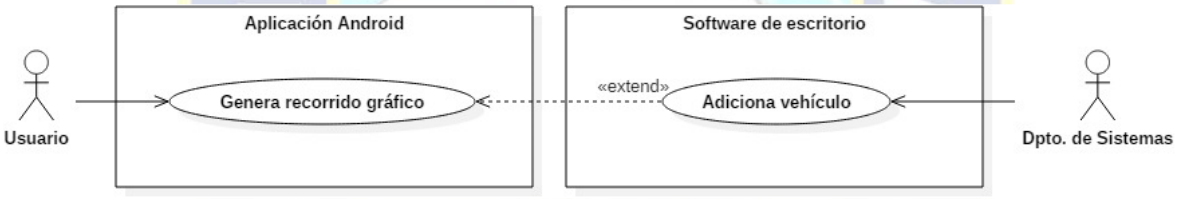


Figura 18 Caso de uso: gestión de recorrido

Fuente: Elaboración propia.

3.3. Fase de elaboración

La fase de elaboración consiste en analizar los casos de uso diseñados anteriormente para poder definir la arquitectura base, el diseño de la aplicación y la especificación de lo que deben hacer cada una de sus partes, así como la manera en que se combinan unas con otras dentro de su funcionamiento, en esta fase se implementará los artefactos del Lenguaje Unificado de Modelado citados a continuación:

3.3.1. Diagrama de Clases

El diagrama de clases es un diagrama con el propósito de describir la estructura que tendrá la aplicación Android, en este caso nos permitirá realizar una descripción de todas las clases que se tomarán en cuenta para lograr el desarrollo de esta aplicación.

De acuerdo a los requerimientos y necesidades que precisa el cliente dentro de la aplicación se pudo realizar el desarrollo de las siguientes clases:

3.3.1.1. Clases propias de la aplicación

- Servidor: Contiene las direcciones de todos los servidores con los que se conectará la aplicación.
- Versión APP: Sirve de registro para las versiones de la aplicación para informar al usuario en caso de necesitar su actualización.
- Datos Usuario: Contiene todos los datos de conexión con los servidores direccionando al usuario a su servidor correspondiente.
- Usuario Móvil: Contiene todos los datos del dispositivo inteligente y del usuario propietario de éste.
- Notificación: Contiene la información de todos los eventos prioritarios que hayan reportado los vehículos del cliente.
- Alerta: Contiene la misma información de la clase Notificacion, pero ésta es enviada a los dispositivos inmediatamente ocurra algún evento.
- Alerta Evento: Contiene la descripción de los tipos de eventos con los que cuenta la aplicación.
- Alerta Correo: Igualmente contiene la información de los eventos reportados, pero éstos son enviados al usuario vía correo electrónico.
- Alerta Dispositivo: Contiene la información de los parámetros que tiene cada dispositivo con respecto a las alertas que desea recibir.

3.3.1.2. Clases del sistema actual de monitoreo

- **Cliente**: Se encarga de guardar toda la información con respecto a los clientes de la empresa.
- **Usuario**: Es la que contiene toda la información de los usuarios que tiene cada empresa cliente.

- **Cuadrilla**: Podemos definir una cuadrilla como un conjunto o un grupo de vehículos que realizan una actividad en común. La clase **Cuadrilla** se utiliza para agrupar un grupo determinado de vehículos de un usuario.
- **Usuario Cuadrilla**: Se utiliza para asignar una cuadrilla a un usuario en el sistema con el fin de que éste pueda observar la información de los vehículos que le corresponden.
- **Vehículo**: Contiene toda la información de los vehículos del cliente.
- **Recorrido**: Es la que contiene todos los datos que se reciben de los dispositivos GPS instalados en los vehículos.

3.3.1.3. Gráfico del diagrama de Clases

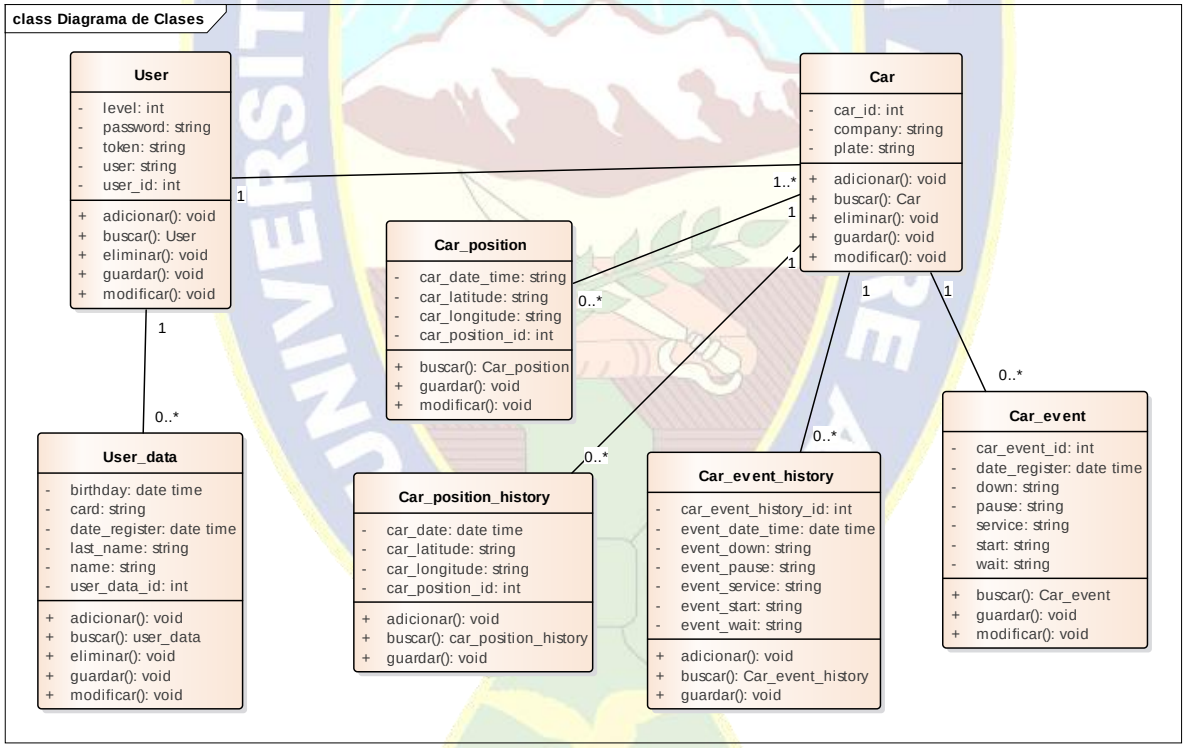


Figura 19 Diagrama de clases

Fuente: Elaboración propia.

En el diagrama anterior se pudo observar todas las clases que se tomarán como referencia para el desarrollo de la aplicación Android, éstas cuentan con sus respectivos atributos o cualidades como también con las operaciones que éstas desempeñan.

Para el diseño de este diagrama se tuvo en cuenta las clases que tiene el sistema actual de la empresa Total Radio Systems Ltda., además se diseñaron clases acorde al desarrollo del proyecto en general.

3.3.2. Modelo entidad relación

El modelo Entidad Relación (E-R) es una herramienta utilizada para el modelado de datos que permite representar las entidades relevantes de un sistema, así como sus interrelaciones y propiedades.

Una vez obtenidas todas las clases con las que contará la aplicación Android para su funcionamiento se procede a realizar el diseño del diagrama Entidad Relación indicando cómo las entidades más relevantes dentro de la aplicación se relacionan entre sí.

3.3.2.1. Gráfico del modelo entidad – relación

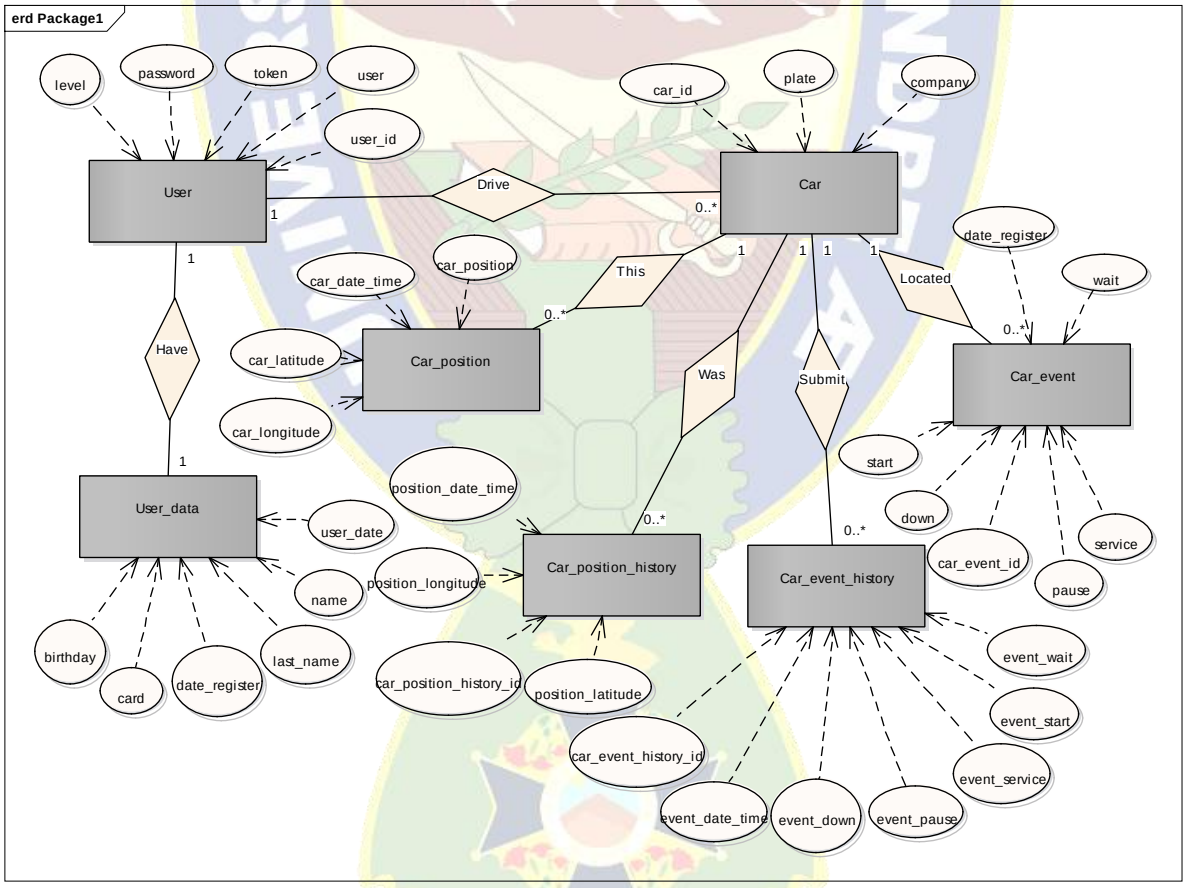
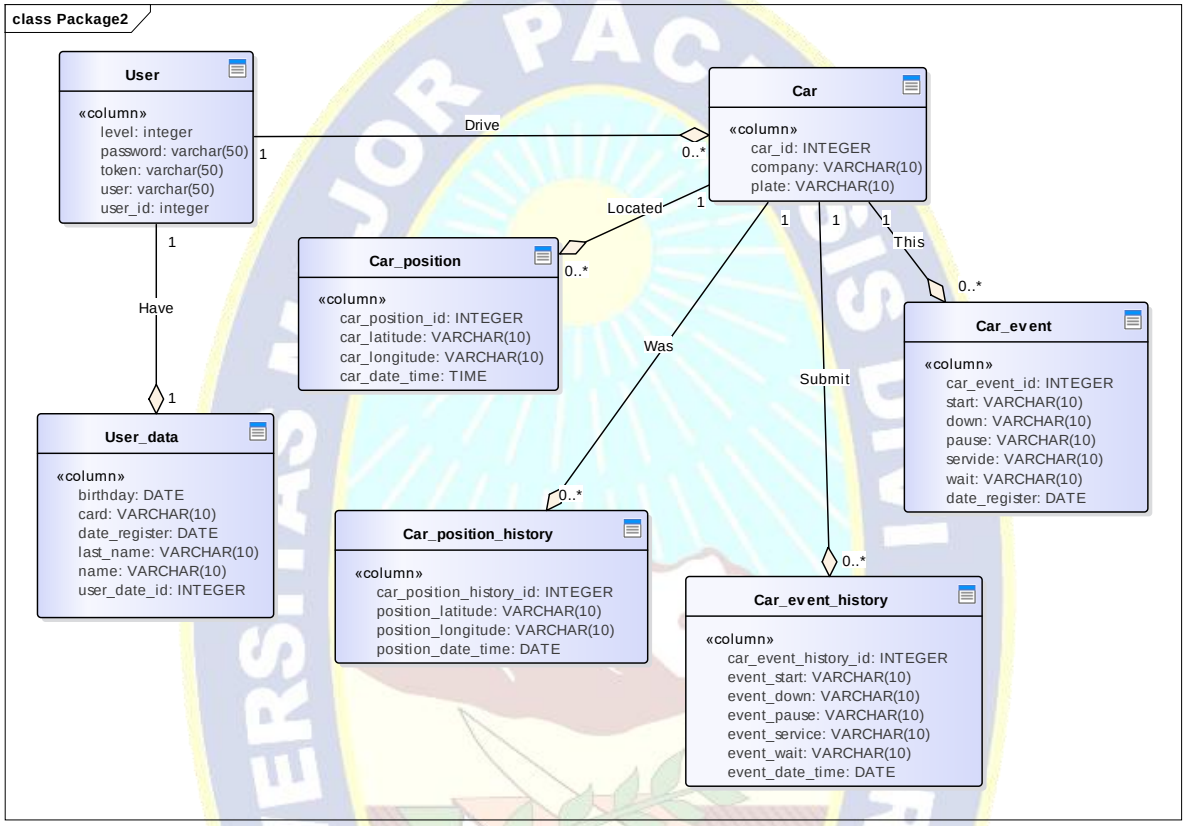


Figura 20 Modelo entidad – relación

Fuente: Elaboración propia.

3.3.3. Diagrama Físico



3.3.4. Diagrama de secuencia

El diagrama de secuencia es un tipo de diagrama que se usa para modelar la interacción entre los objetos de un sistema según el lenguaje unificado de modelado.

En este caso según lo especificado en el diagrama de casos de uso se especificará la interacción que tendrán los principales actores (el usuario y el departamento de sistemas) dentro de la aplicación Android.

3.3.4.1. Usuario

- Adicionar usuario: Al autenticarse en la aplicación, esta verifica si es que el dispositivo se encuentra registrado en la base de datos: si el usuario se encuentra registrado, se ingresa con normalidad a la pantalla principal; en caso de no estar

registrado, el usuario pasa a una pantalla donde éste deberá ingresar sus datos para habilitar el dispositivo en la aplicación Android.

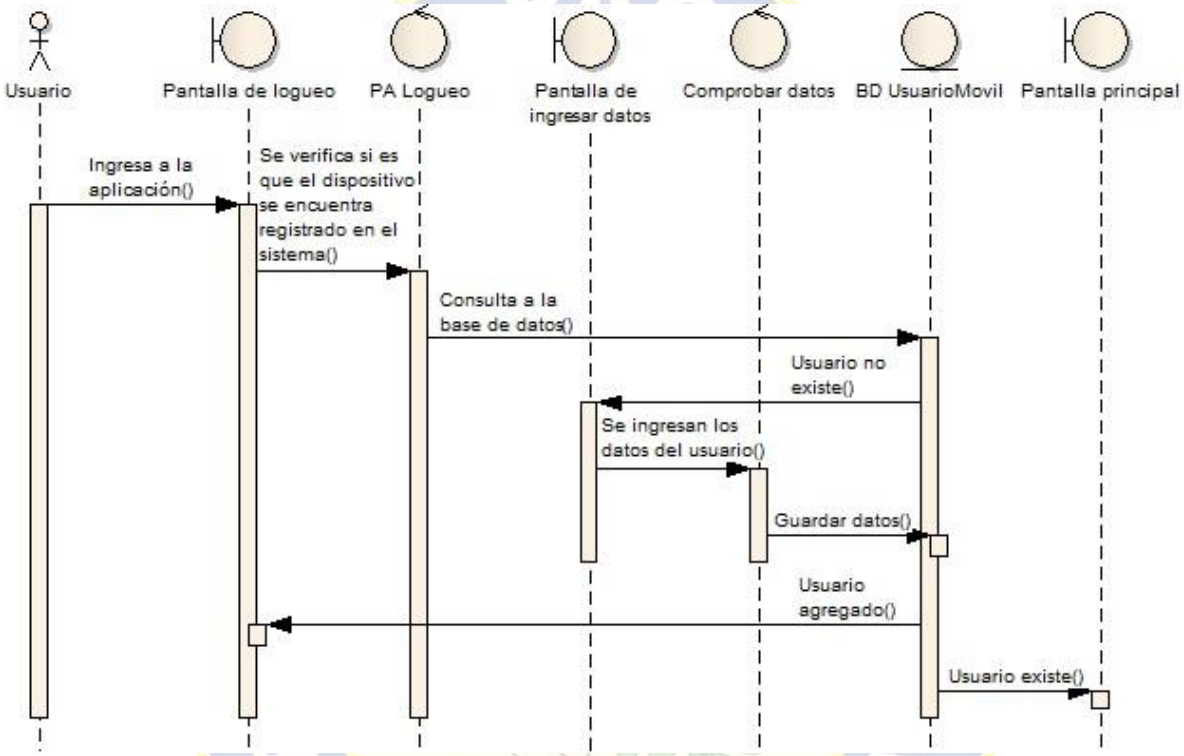


Figura 21 Secuencia: adicionar usuario

Fuente: Elaboración propia.

- Modificar usuario: En la pantalla principal el usuario cuenta con una opción donde puede modificar la información con la que se encuentra registrado en la base de datos.

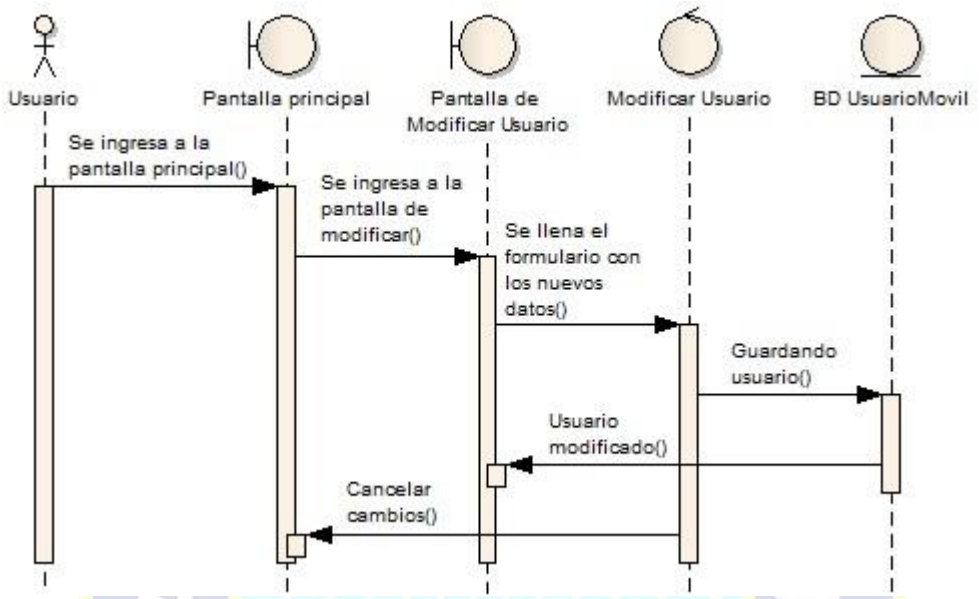


Figura 22 Secuencia: modificar usuario

Fuente: Elaboración propia.

- Listar vehículo: El usuario puede deslizarse desde la pantalla principal hacia la pantalla de vehículos donde éste puede observar la información de los vehículos que tiene asignados dentro de la plataforma de monitoreo.

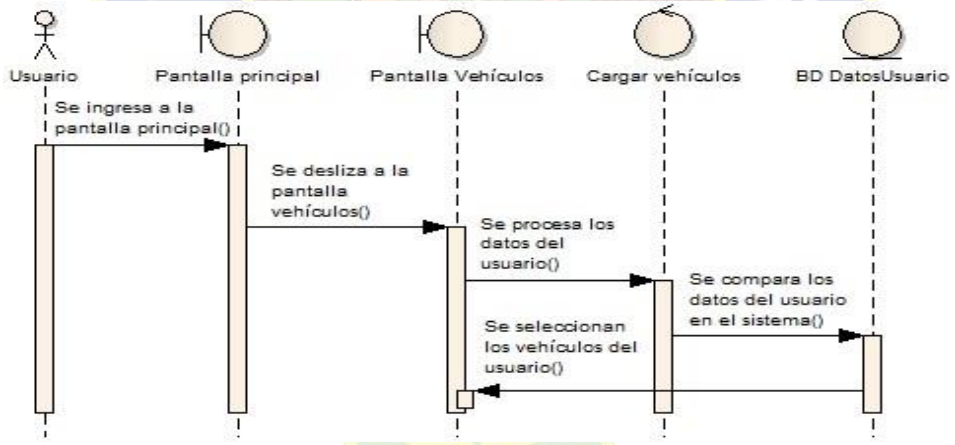


Figura 23 Secuencia: listar vehículo

Fuente: Elaboración propia.

- Listar recorrido: En la pantalla de vehículos se encuentra el botón de vehículos donde ingresando en la opción **recorrido por hora** el usuario puede observar la

ruta por la cual reportó su vehículo previamente ingresando los parámetros de la ruta.

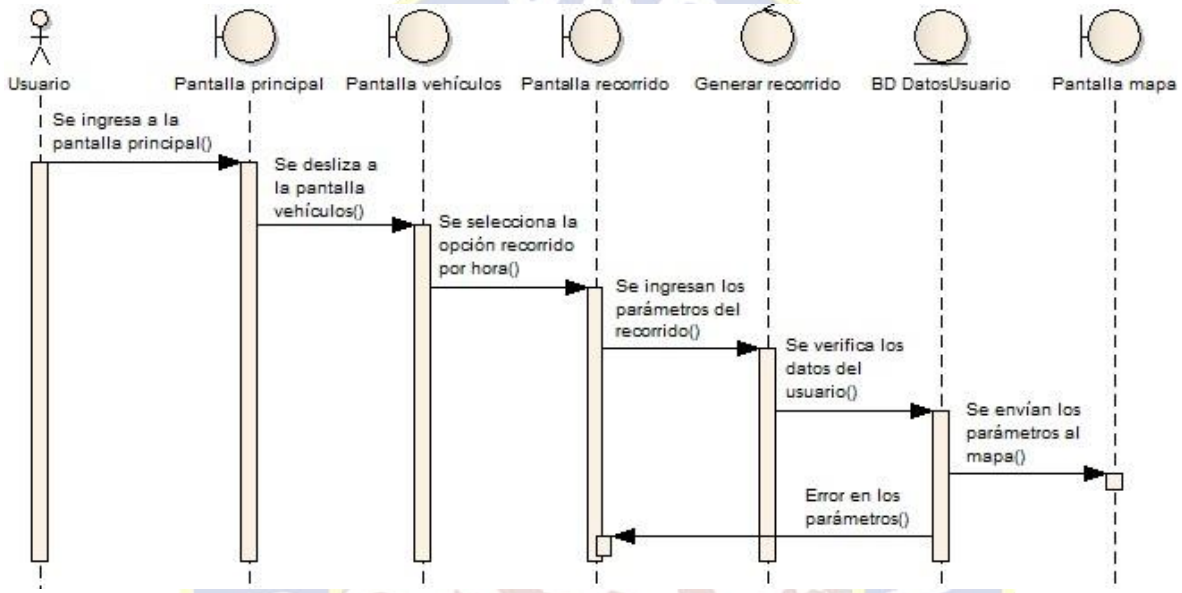


Figura 24 Secuencia: listar recorrido

Fuente: Elaboración propia.

- Modificar notificación: El usuario puede modificar los parámetros de las notificaciones que recibe ingresando en el menú del lado superior izquierdo en la opción **Alertas** donde confirma si es que quiere recibir o no las alertas y en caso de querer recibirlas, éste ingresa a una pantalla donde puede modificar los parámetros de las notificaciones.

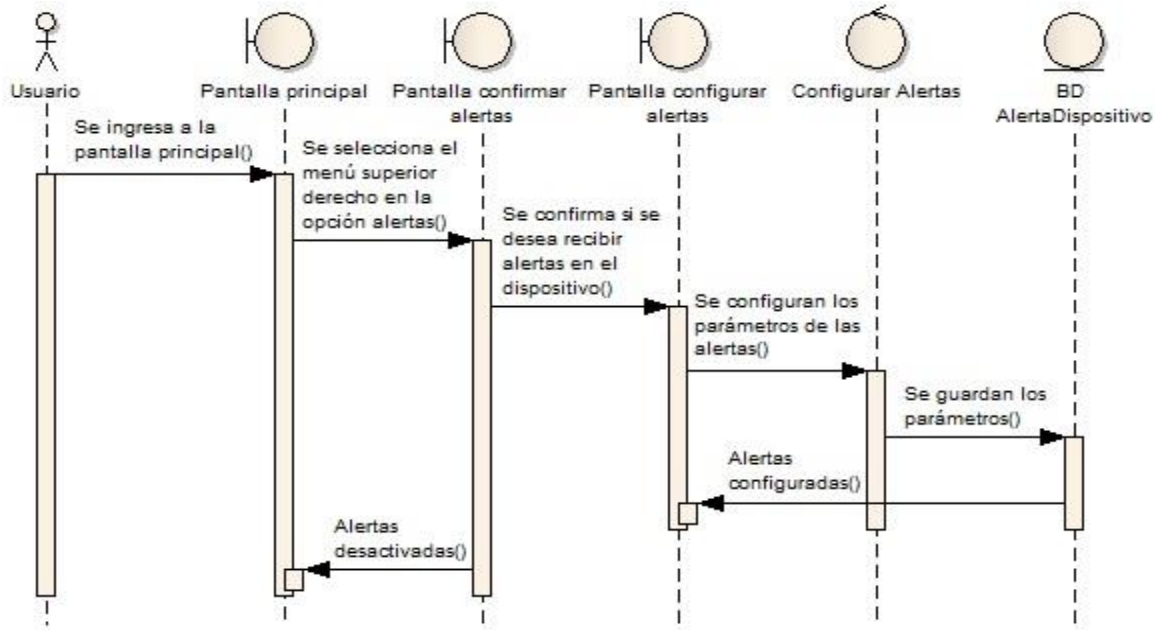


Figura 25 Secuencia: modificar notificación

Fuente: Elaboración propia.

3.3.3.2 Departamento De Sistemas

- Adiciona notificación: Se tiene configurado un procedimiento almacenado que verifique los datos recibidos de los vehículos para poder generar las notificaciones de cada usuario, este procedimiento se ejecuta constantemente como tarea de SQL Server.

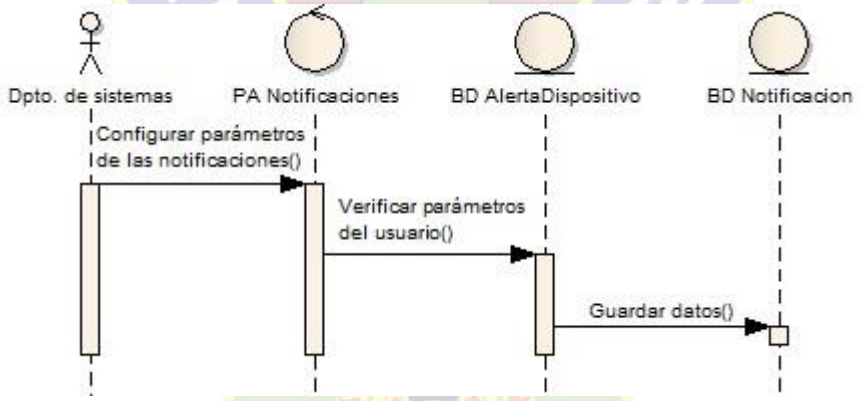


Figura 26 Secuencia: adiciona notificación

Fuente: Elaboración propia.

- Genera alerta: Cuando se genera una notificación, un desencadenador verifica si es que el usuario tiene habilitadas o no las alertas y en caso de tenerlas activadas la información de la notificación pasa a ser almacenada y enviada a los dispositivos móviles vía Firebase Cloud Messaging por la ventana Mensajes, como también es enviada al correo electrónico por la ventana Alertas Correo del software de escritorio.

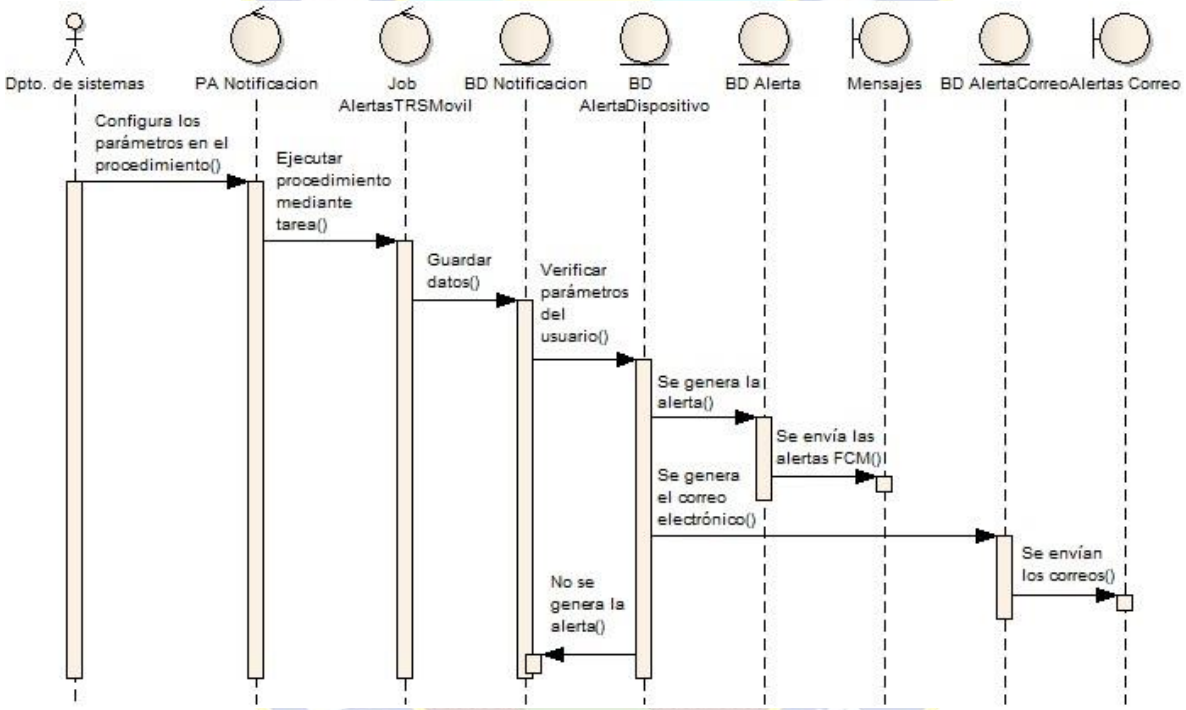


Figura 27 Secuencia: genera alerta

Fuente: Elaboración propia.

- Habilitar usuario: En la pantalla de usuarios del software de escritorio el departamento de sistemas cuenta con la opción de modificar el estado de los usuarios de la aplicación Android.

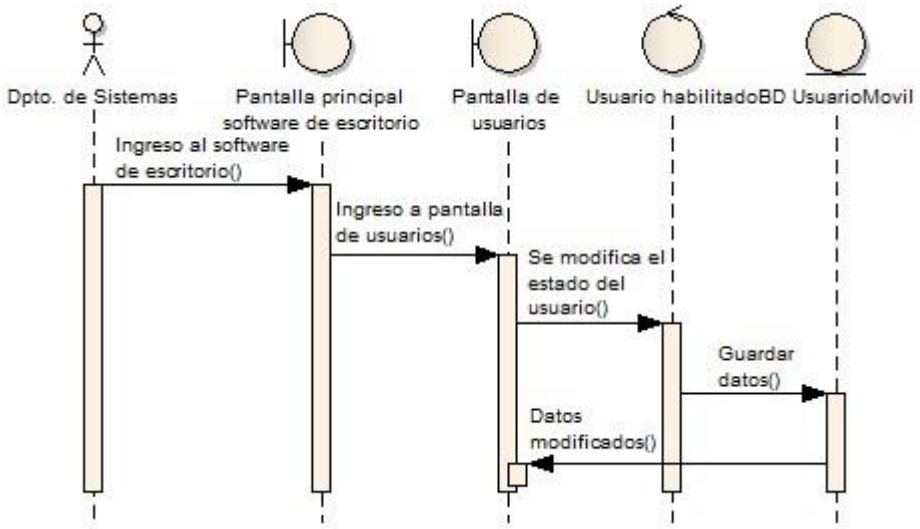


Figura 28 Secuencia: habilitar usuario

Fuente: Elaboración propia.

- Lista usuarios: En la ventana de Usuarios del software de escritorio el departamento de sistemas puede obtener el listado de los usuarios registrados en la aplicación Android.

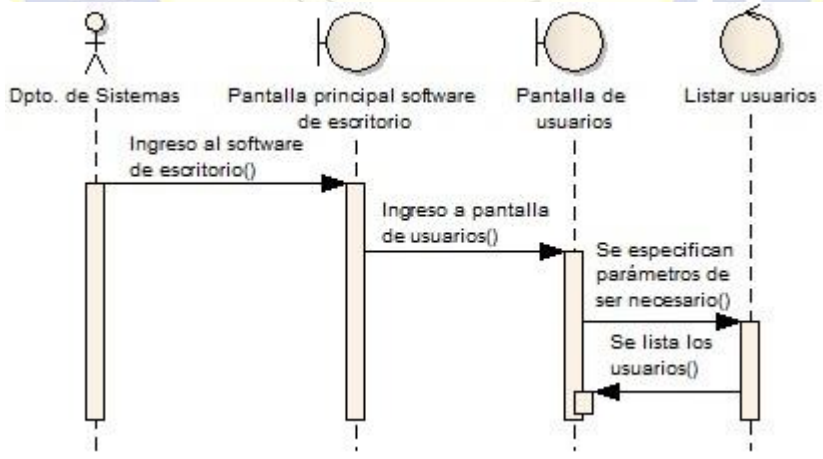


Figura 29 Secuencia: lista usuarios

Fuente: Elaboración propia.

- Habilitar vehículo: En la pantalla de vehículos el departamento de sistemas puede modificar el estado de los vehículos para que éstos sean visibles o no por el usuario de la aplicación.

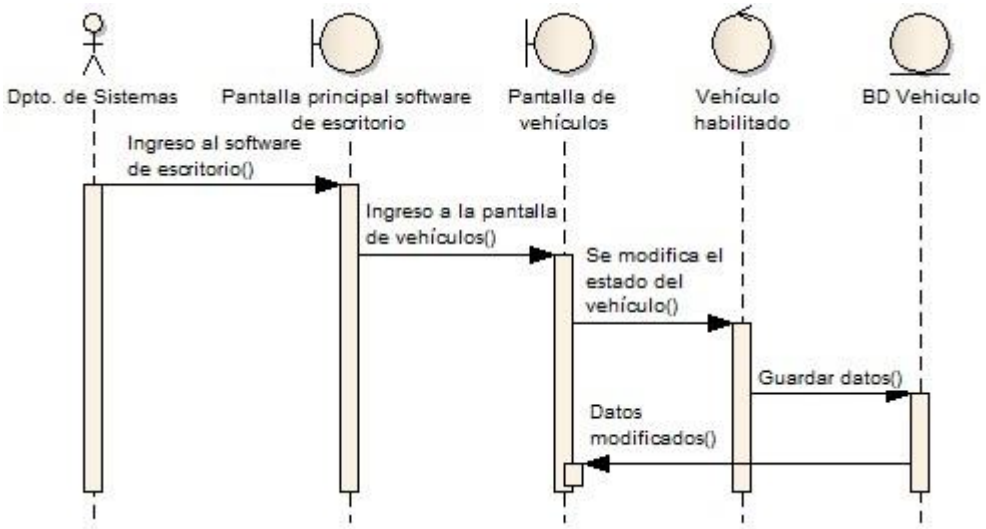


Figura 30 Secuencia: habilitar vehículo

Fuente: Elaboración propia.

- Modifica vehículo: El departamento de sistemas puede modificar la información de los vehículos según las necesidades del usuario.

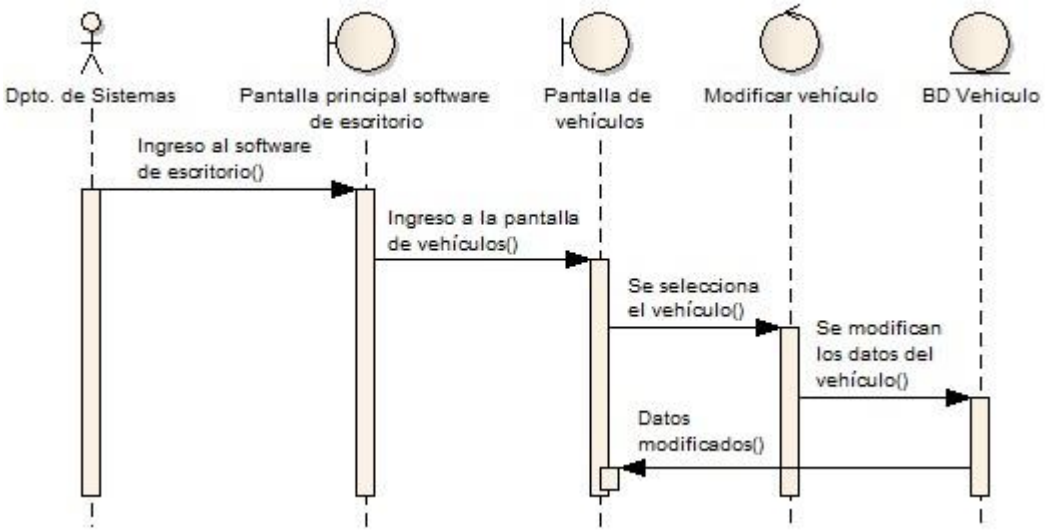


Figura 31 Secuencia: modifica vehículo

Fuente: Elaboración propia.

3.3.5. Diagrama de estados

El diagrama de estados es un método que explica todos los estados posibles de un objeto particular y la manera en que se modifica el estado del objeto, determina cada una de las

rutas o caminos que puede tomar un movimiento de información luego de ejecutarse cada proceso.

En este caso se utilizarán los diagramas de estados para describir cada una de las rutas que tomarán los procesos dentro de la aplicación Android.

3.3.5.1. Usuario

- Adicionar usuario: La aplicación Android verifica si es que el usuario se encuentra registrado en la base de datos: si el usuario está registrado se ingresa a la pantalla principal; si no se encuentra el usuario, éste debe ingresar sus datos para ser habilitado en la aplicación Android.

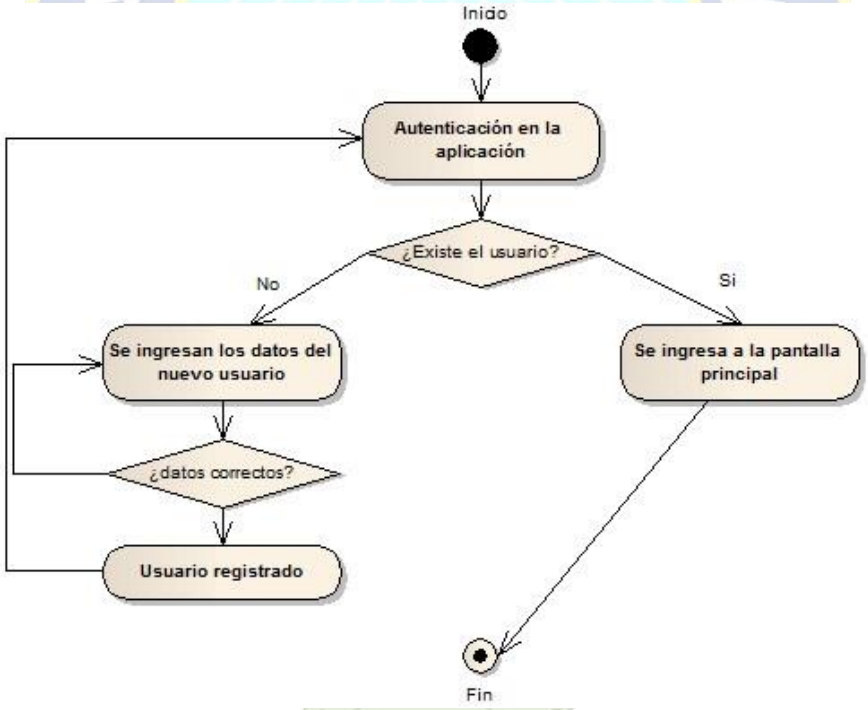


Figura 32 Diagrama de estados: adicionar usuario

Fuente: Elaboración propia.

- Modificar usuario: Si el usuario se encuentra autenticado correctamente en la aplicación Android, éste puede realizar los cambios de los datos con los que se encuentra registrado en la base de datos.

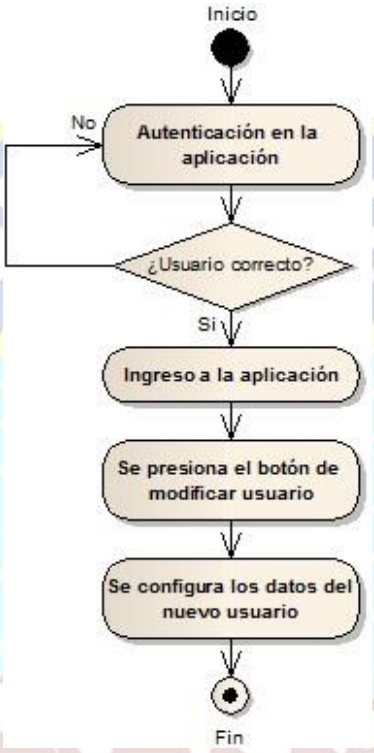


Figura 33 Diagrama de estados: modificar usuario

Fuente: Elaboración propia.

- Listar vehículo: Si el usuario tiene acceso a la aplicación Android éste puede ver el listado de los vehículos que tiene asignados.

Figura 34 Diagrama de estados: listar vehículo

Fuente: Elaboración propia.

- Listar recorrido: El usuario puede obtener el gráfico del recorrido sus vehículos previamente indicando parámetros como la fecha y hora, si existen datos en esas fechas se visualiza el recorrido; en caso de no existir datos se emite el mensaje: **No hay registros para mostrar**.

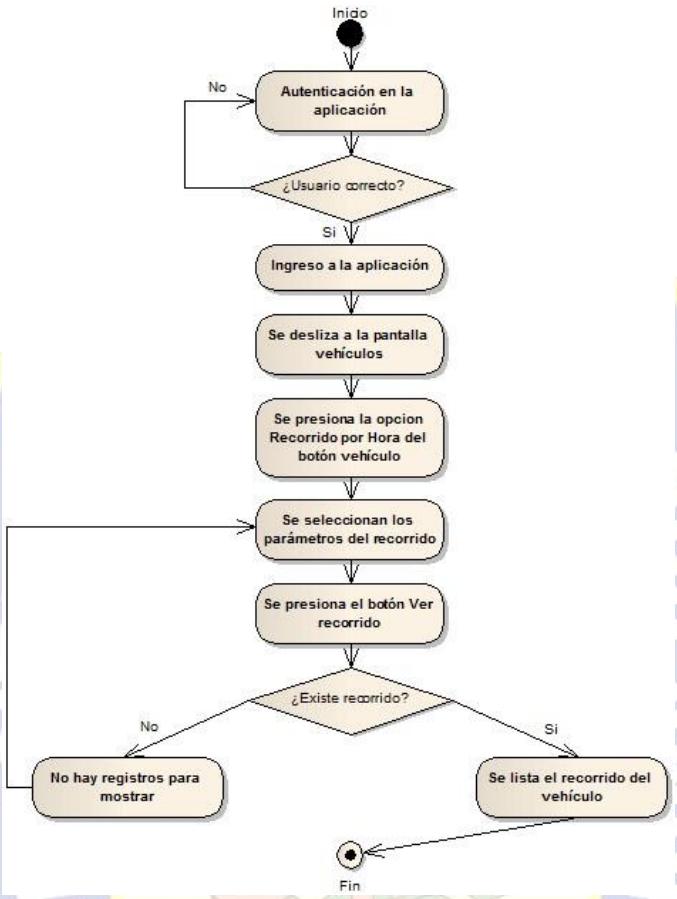


Figura 35 Diagrama de estados: listar recorrido

Fuente: Elaboración propia

- Listar notificación: Si el usuario se encuentra autenticado correctamente en la aplicación, éste puede visualizar los eventos reportados por los vehículos en la pantalla de Notificación.

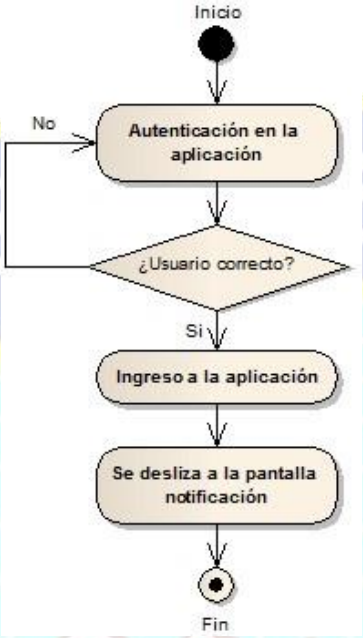


Figura 36 Diagrama de estados: listar notificación

Fuente: Elaboración propia.

- Modificar notificación: El usuario puede decidir si recibir o no las alertas en su dispositivo móvil, en caso de querer recibirlas éste pasa a una pantalla donde configura los parámetros de las notificaciones y por ende de las alertas que recibirá en su dispositivo; si no desea recibir las alertas, éstas no llegarán a su dispositivo, pero se mostrarán en la pantalla de Notificación.

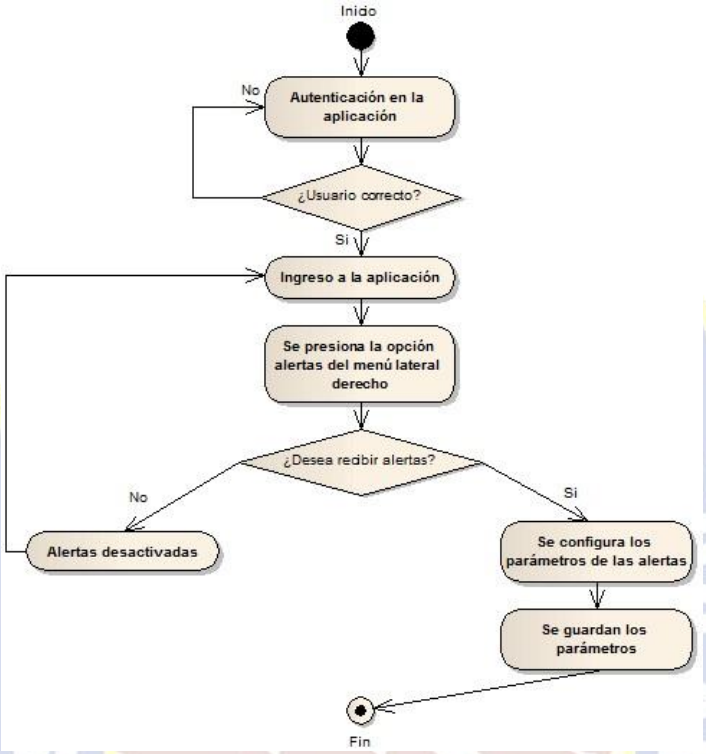


Figura 37 Diagrama de estados: modificar notificación

Fuente: Elaboración propia

3.3.5.2. Departamento de sistemas.

- Adiciona notificación: Se verifican los registros que se reciben de los vehículos y si existe alguno que cumpla con los parámetros de las alertas, este se ingresa a la tabla Notificación.

Figura 38 Diagrama de estados: adiciona notificación

Fuente: Elaboración propia.

- Genera alerta: Al ingresar un registro a la tabla Notificacion un trigger verifica si es que el usuario tiene habilitadas las alertas, en caso de tener habilitadas las alertas los registros de la notificación pasan a ser ingresados en las tablas Alerta y AlertaCorreo y son enviados en forma de alerta y correo electrónico.

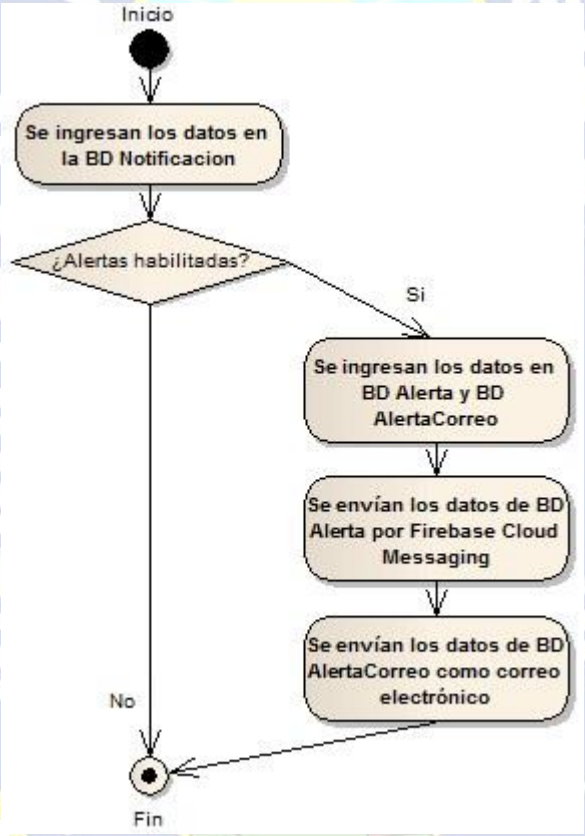


Figura 39 Diagrama de estados: genera alerta

Fuente: Elaboración propia.

- Habilitar usuario: El departamento de sistemas puede decidir si habilitar o no a los usuarios de la aplicación, esto por diversas razones.

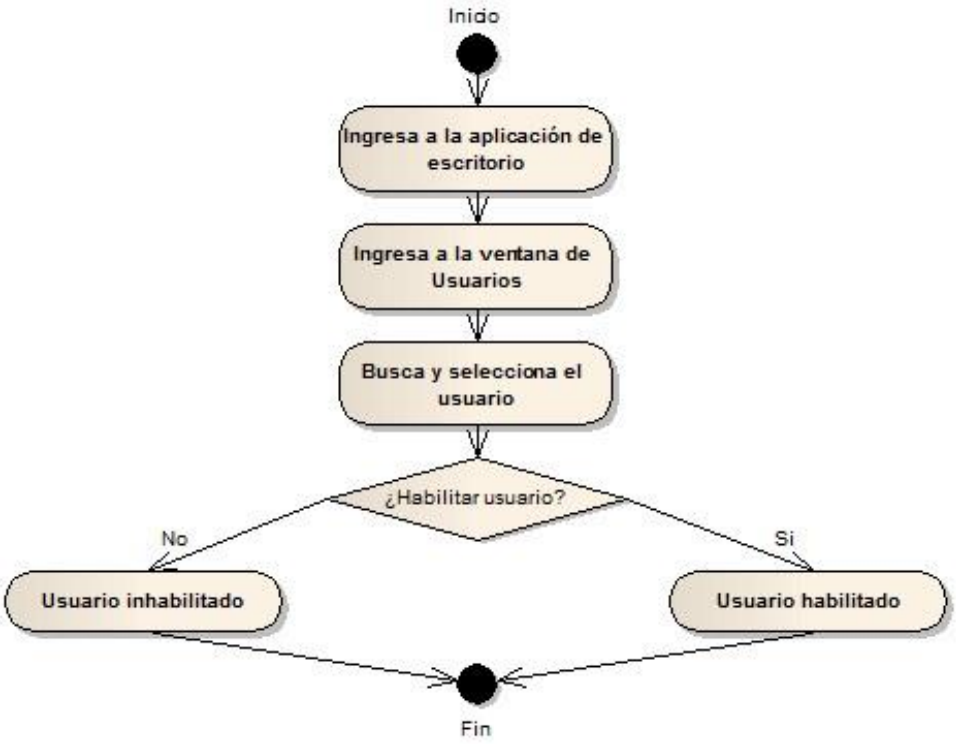


Figura 40 Diagrama de estados: habilitar usuario

Fuente: Elaboración propia.

- Lista usuarios: Si el departamento de sistemas desea puede listar a los usuarios de la aplicación en la interfaz de usuarios del software de escritorio.

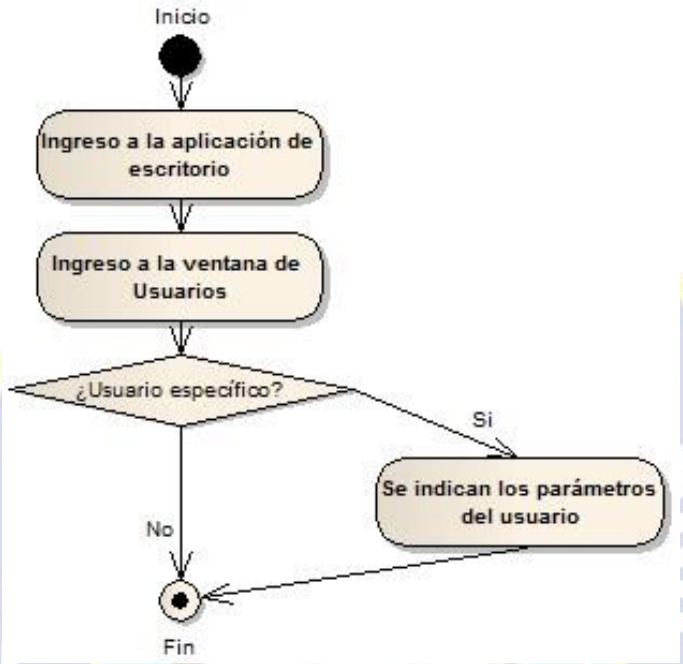


Figura 41 Diagrama de estados: lista usuarios

Fuente: Elaboración propia.

- Habilitar vehículo: El departamento de sistemas puede habilitar o inhabilitar los vehículos dentro del sistema por temas distintos para que sean visibles o no para el usuario.

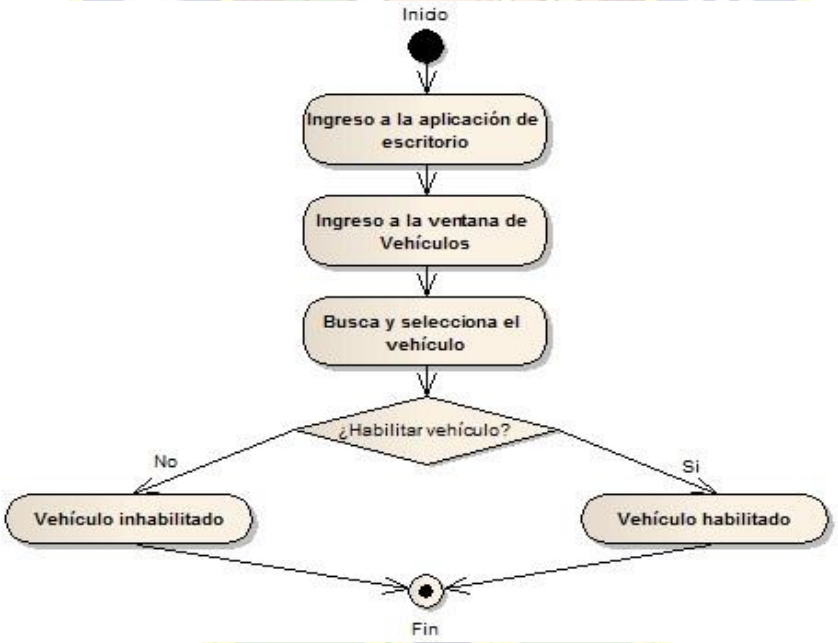


Figura 42 Diagrama de estados: habilitar vehículo

Fuente: Elaboración propia.

- Modifica vehículo: Si el usuario lo desea, el departamento de sistemas puede modificar la información de los vehículos dentro del sistema (alias, placa, entre otros), esto para su visualización en la aplicación.

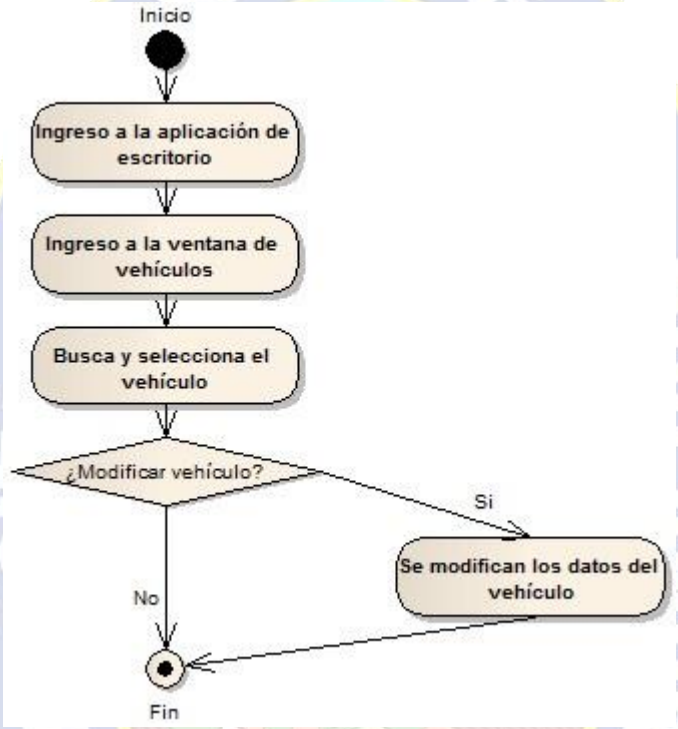


Figura 43 Diagrama de estados: modifica vehículo

Fuente: Elaboración propia.

3.3.6. Diagrama de colaboración

El diagrama de colaboración (también llamado diagrama de comunicación, aunque es una versión más simplificada) es esencialmente un diagrama que muestra la interacción entre los roles.

A diferencia de los diagramas de secuencia, estos muestran las relaciones de los roles y modelan las interacciones entre objetos o partes en términos de mensajes en secuencia, representan una combinación de información de los diagramas de clases, secuencia, y casos de uso describiendo tanto la estructura estática como el comportamiento dinámico de un sistema.

Los diagramas de comunicación y de secuencia describen información similar, y con algunas variantes, pueden ser transformados unos en otros sin dificultad.

Los diagramas de colaboración se utilizarán para describir la interacción de los roles y secuencias dentro de la aplicación Android.

3.3.6.1. Usuario

- Adicionar usuario: El usuario ingresa a la pantalla de logueo, éste se autentica y es verificado por el sistema; si se encuentra registrado pasa a la pantalla principal, en caso de no estar registrado pasa a otra pantalla donde deberá ingresar sus datos para ser habilitado en el sistema.

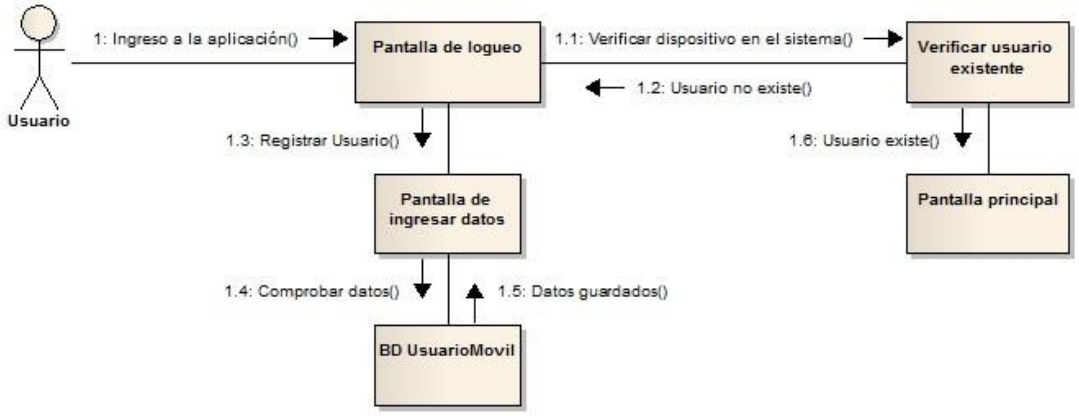


Figura 44 Diagrama de colaboración: adicionar usuario

Fuente: Elaboración propia.

- Modificar usuario: El usuario ingresa a la pantalla principal y posteriormente a la pantalla de modificar usuario donde éste puede modificar los datos con los que se encuentra registrado en el sistema.

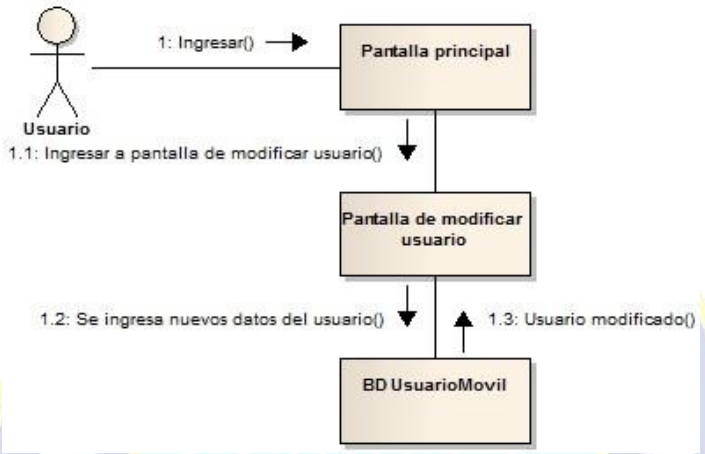


Figura 45 Diagrama de colaboración: modificar usuario

Fuente: Elaboración propia.

- Listar vehículo: El usuario puede listar los vehículos que tiene asignados en la pantalla vehículo.

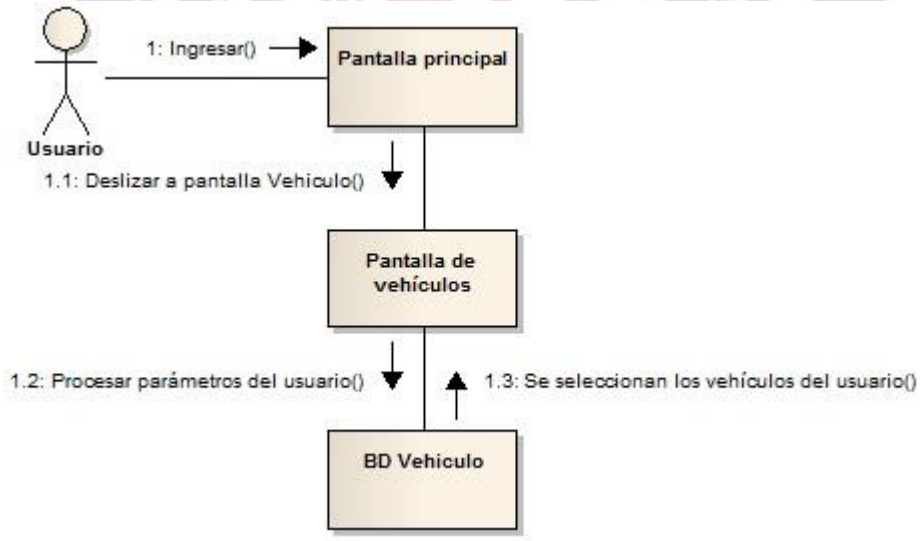


Figura 46 Diagrama de colaboración: listar vehículo

Fuente: Elaboración propia.

- Listar recorrido: El usuario puede ver la ruta por la cual han transitado sus vehículos en la pantalla Recorrido previamente ingresando los parámetros del mismo.

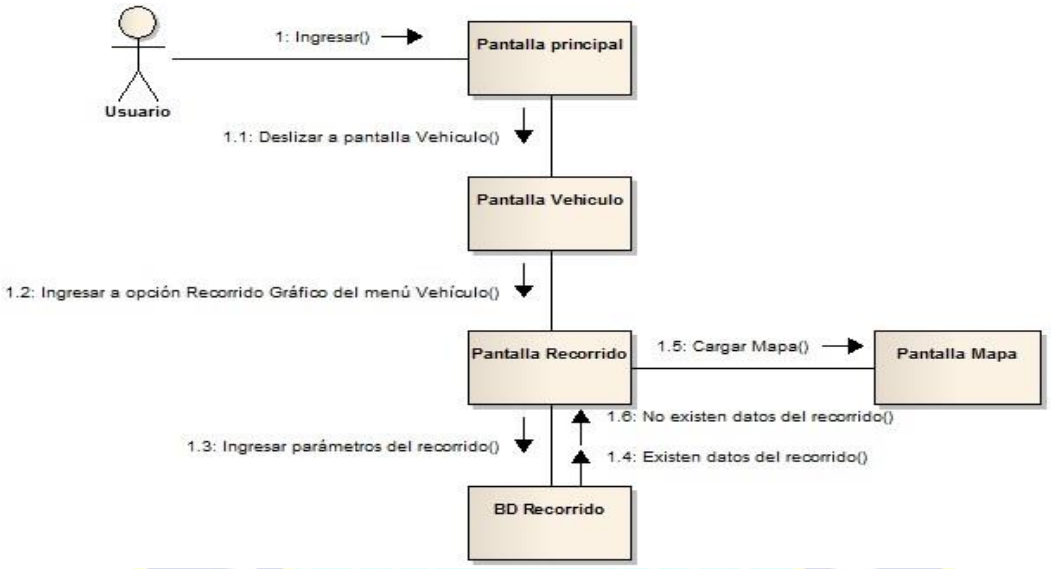


Figura 47 Diagrama de colaboración: listar recorrido

Fuente: Elaboración propia.

- Listar notificación: Las notificaciones pueden ser listadas por el usuario desde la pantalla de Notificación.

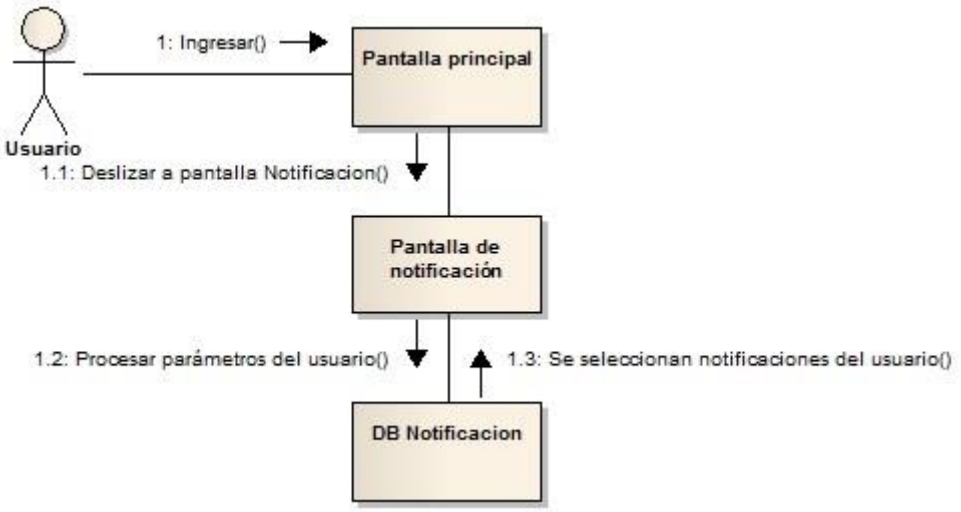


Figura 48 Diagrama de colaboración: listar notificación

Fuente: Elaboración propia.

- Modificar notificación: El usuario puede modificar los parámetros de las notificaciones (y por ende las alertas) ingresando a la pantalla de Alertas.

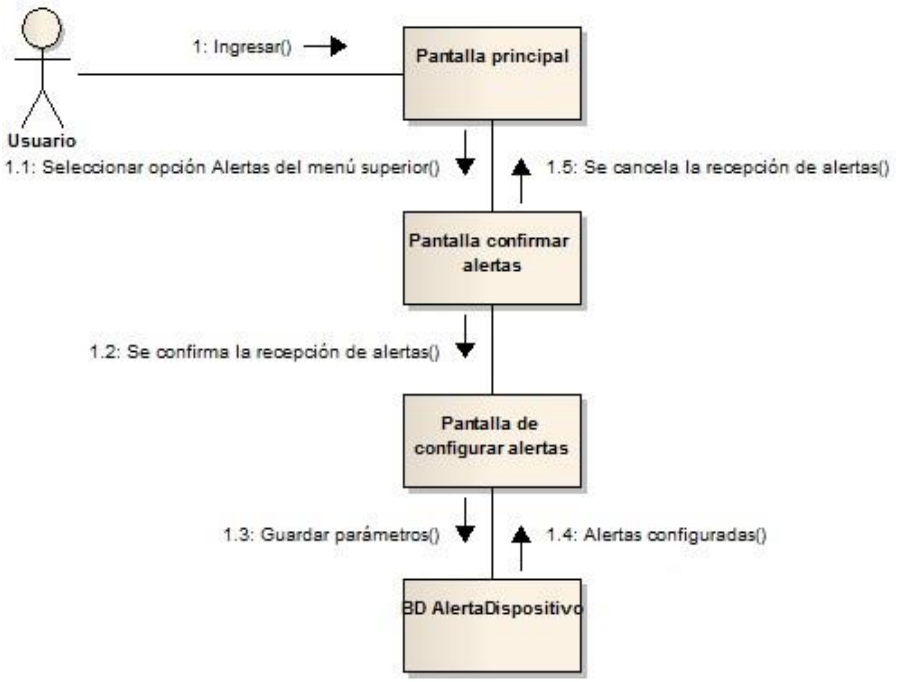


Figura 49 Diagrama de colaboración: modificar notificación

Fuente: Elaboración propia.

3.3.6.2. Departamento de sistemas

- Adiciona notificación: El encargado de sistemas puede ingresar una notificación para enviarla desde la pantalla Notificacion de la aplicación de escritorio.

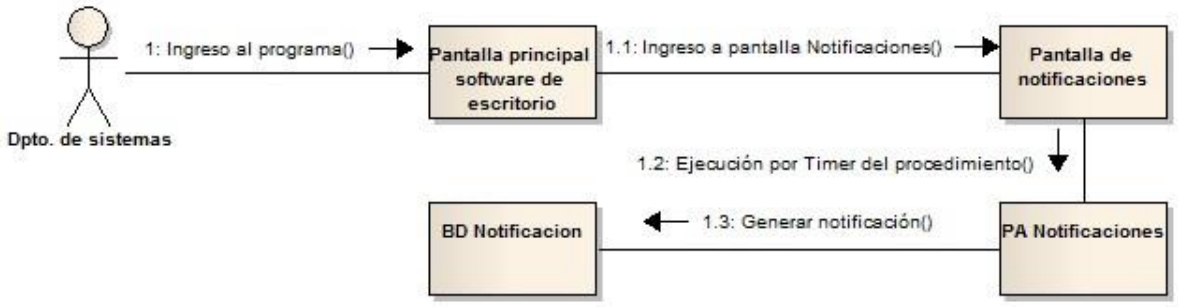


Figura 50 Diagrama de colaboración: adiciona notificación

Fuente: Elaboración propia.

- Genera alerta: El departamento de sistemas configura el trigger de la tabla Notificación para que éste verifique cuando ingrese un registro a la tabla, éste genere una alerta para ser enviada a la aplicación y por correo electrónico.

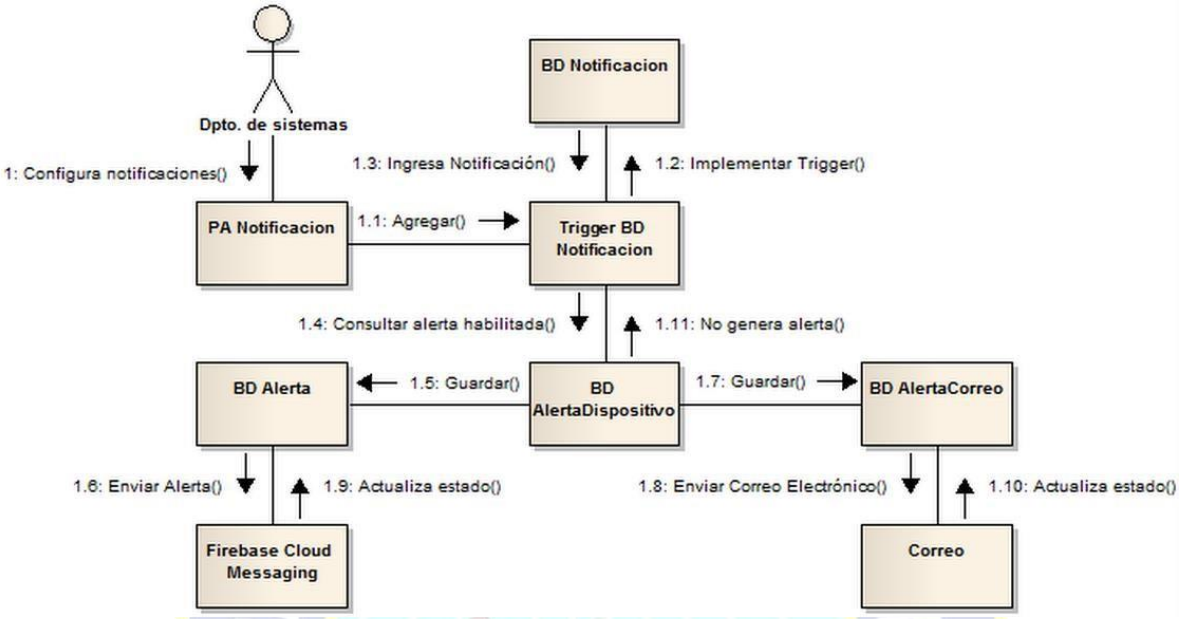


Figura 51 Diagrama de colaboración: genera alerta

Fuente: Elaboración propia.

- Habilitar usuario: En la pantalla de Usuarios del software de escritorio el departamento de sistemas puede habilitar o no a los usuarios de la aplicación Android.

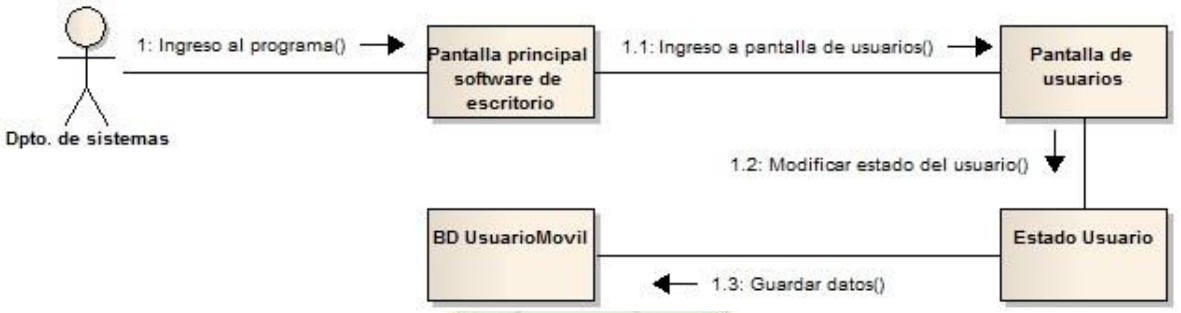


Figura 52 Diagrama de colaboración: habilitar usuario

Fuente: Elaboración propia.

- Lista usuarios: El encargado de sistemas puede listar a todos los usuarios registrados en la aplicación o ingresar un parámetro para listar a un grupo de usuarios.

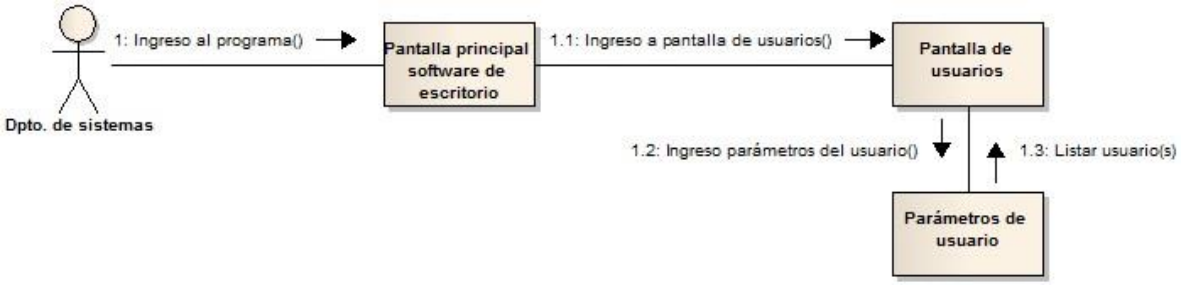


Figura 53 Diagrama de colaboración: lista usuario

Fuente: Elaboración propia.

- Habilitar vehículo: En la pantalla vehículo el departamento de sistemas puede habilitar o deshabilitar los vehículos del sistema.

Figura 54 Diagrama de colaboración: habilitar vehículo.

Fuente: Elaboración propia.

- Modifica vehículo: El departamento de sistemas puede modificar los datos de los vehículos que se encuentran en el sistema en caso de ser necesario.

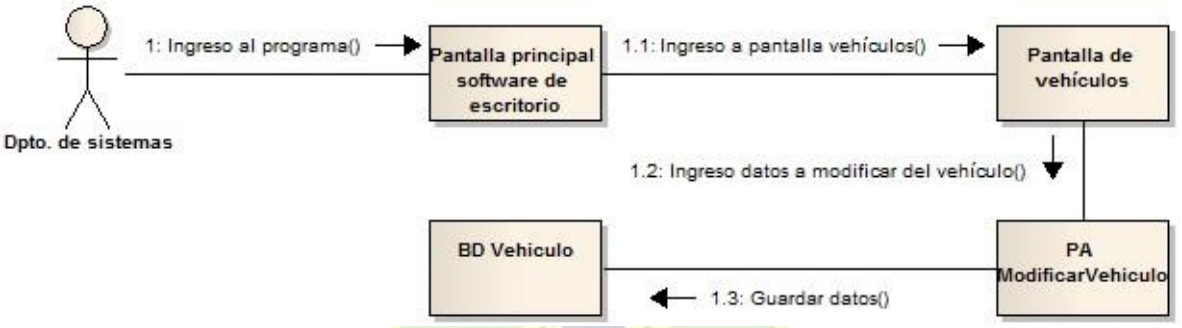


Figura 55 Diagrama de colaboración: modifica vehículo

Fuente: Elaboración propia.

3.3.7. Mapa de comportamiento a nivel de hardware

Para describir el comportamiento a nivel de hardware que tiene la aplicación Android se debe hacer el uso del diagrama de despliegue que proporciona la metodología del Proceso Unificado Racional implementada en el desarrollo actual.

El diagrama de despliegue describe el despliegue físico del software en los componentes de hardware, en este caso se tendrá estructurado el siguiente diagrama:

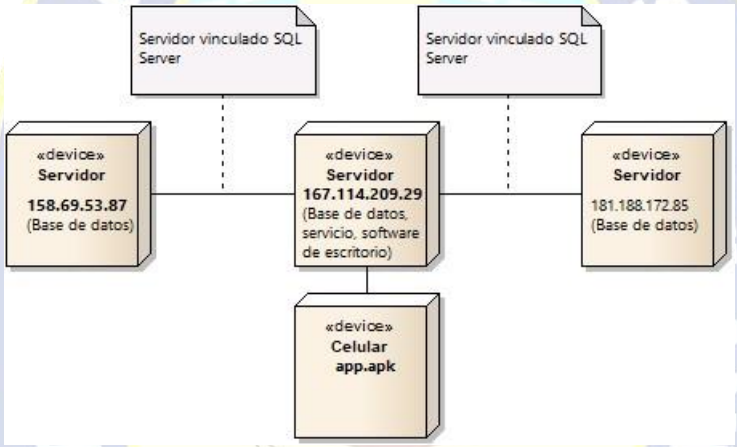


Figura 56 Mapa de comportamiento a nivel de hardware

Fuente: Elaboración propia.

Actualmente la empresa Total Radio Systems Ltda., cuenta con tres servidores donde se realiza el guardado de la información de los clientes.

En este caso la aplicación Android se conectará a un servicio Web y éste se conectará al servidor principal donde se encuentra su base de datos, esta base de datos se conectará a otras bases de datos de los servidores mediante la vinculación de los servidores para que la aplicación Android pueda trabajar con los tres servidores.

3.4. Fase de construcción

La fase de construcción consiste en el desarrollo de la aplicación Android de forma que ésta cumpla con todos los requerimientos y necesidades de los involucrados identificados en las fases previas de análisis y diseño.

Es esta fase se implementarán las herramientas de desarrollo seleccionadas para realizar el diseño de las interfaces de usuario y la codificación de la aplicación para el cumplimiento de los requerimientos del usuario.

3.4.1. Análisis y selección de los colores

La aplicación Android será desarrollada con una paleta de colores que está basada en el tono de colores que usa actualmente la empresa:

Figura 57 Logotipo actual de la empresa
Fuente: Total Radio Systems Ltda.

En este caso se implementarán los siguientes tonos de colores:

Tabla 9 Colores implementados en la aplicación

Fuente: Elaboración propia

NRO.	NOMBRE	#HEX	#RGB	COLOR
1	Verde	#07725E	7,114,94	
2	Blanco	#FFFFFF	255,255,255	

Según estudios referentes a la psicología del color en cuanto a los colores que usa actualmente la empresa podemos destacar que éstos destacan: juventud, crecimiento, esperanza, tranquilidad (verde) como también paz, humildad, modestia (blanco) intentando reflejar que ésta es una empresa joven, que busca crecer en el mercado de la innovación tecnológica en el país que garantice a sus clientes un servicio de calidad.

3.4.2. Diseño de interfaces y codificación

A continuación, se procede a plasmar la arquitectura y los casos de uso indicados anteriormente dentro del desarrollo e interfaces necesarias para cumplir con el objetivo indicado:

3.4.2.1. Rest Api

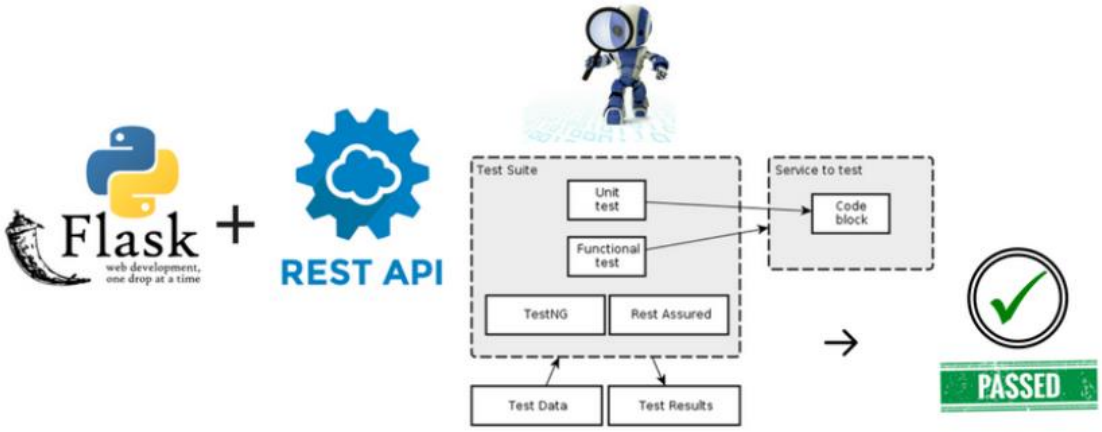


Figura 58 Interfaz de Rest API

Fuente: https://medium.com/hackernoon/writing-unit-tests-for-rest-api-in-python-web-application-2e675a601a53.

3.4.2.2. Aplicación Flutter

El funcionamiento de la aplicación es bastante intuitivo, la persona únicamente debe digitar su usuario y contraseña con los que cuenta actualmente para ingresar a la plataforma web y presionar el botón CONECTAR para acceder, después se guardara en un token para que los datos de su cuenta queden guardados y ya no realizar la autenticación la próxima vez que ingrese a la aplicación. Las pantallas serán descritas con más detalle en el documento de Manual de Usuario

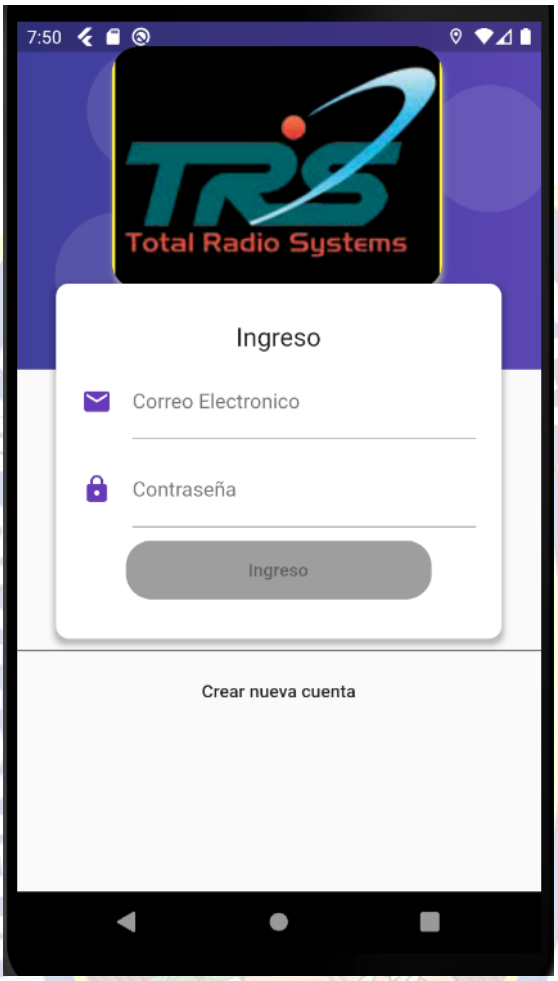


Figura 59 Pantalla de autenticación del Usuario

Fuente: Elaboración propia.

Una vez iniciado sesión, el Usuario entrará en la interfaz de eventos de usuario donde el podrá elegir el estado para su Usuario los cuales estarán entre:

- En Servicio
- Fuera de Servicio
- Ocupado
- En Espera

Tabla 10 Pantalla de Eventos de Usuario

Fuente: Elaboración propia.

En la pantalla de Notificaciones, el usuario puede listar notificaciones detalladas de los eventos vehiculares registrados como Señal Perdida, Envió de Ubicación, entre otras.

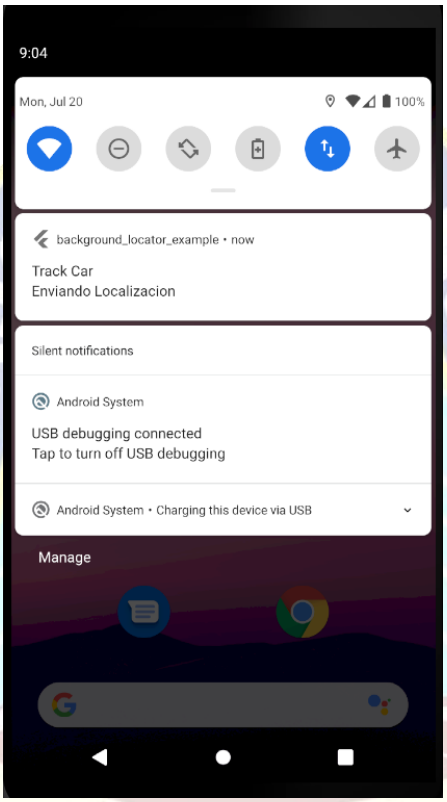


Figura 60 Pantalla de notificaciones

Fuente: Elaboración propia.

Capítulo IV

Métricas De Calidad

4.1. Introducción

Una vez terminada la etapa de desarrollo, el siguiente paso consiste en realizar pruebas de la aplicación Android con el objetivo de comprobar que ésta funcione correctamente.

Según Pressman (2005), la prueba basada en escenario descubrirá errores que ocurren cuando cualquier actor interactúa con el software y cuando ocurren errores asociados con una especificación incorrecta, el producto no hace lo que el cliente quiere. Puede hacer lo correcto u omitir funcionalidad importante.

La prueba de software consiste en verificar el prototipo destinado al usuario final para conocer sus cualidades, probar su eficiencia, su eficacia, verificar su funcionalidad y cómo este reacciona, o qué resultados produce.

4.1.1. Características Propuestas por WEB site QEM

WEB site QEM plantea un modelo normalizado que permite evaluar y comparar productos sobre la misma base.

Aquí la calidad se define por seis características:

- **Funcionalidad:** En este grupo se conjunta una serie de atributos que permiten calificar si un producto de software maneja en norma adecuada el conjunto de fondones que satisfagan las necesidades para las cuales fue diseñado.
- **Fiabilidad:** Capacidad del software de mantener su nivel de ejecución bajo condiciones normales en un periodo de tiempo establecido.
- **Usabilidad:** Consiste de un conjunto de atributos que permiten evaluar el esfuerzo necesario que deberá invertir el usuario para utilizar el sistema.
- **Portabilidad:** Evalúa la oportunidad para adaptar el software a diferentes ambientes sin necesidad de aplicarle modificaciones.
- **Mantenibilidad:** Es el esfuerzo necesario para diagnosticar las deficiencias o causas de fallas, o para identificar las partes que deberán ser modificadas.
- **Eficiencia:** Esta característica permite evaluar la relación entre el nivel de funcionamiento del software y la cantidad de recursos usados

Tabla 11 resultados de la calidad del sistema

Fuente: Elaboración Propia

CÓDIGO	ATRIBUTO	CRITERIO ELEMENTAL	IE i(%)	PESO	VALOR
1.	FUNCIONALIDAD				91.47
1.1	Notificaciones en tiempo real			0.8	90

1.1.1	Mecanismos de Geolocalización				90
1.1.1.1	Búsqueda De Vehículos	CB	100	0.8	90
1.1.1.2	Búsqueda Eventos	CB		0.4	0
1.2	Aspectos de Navegación eh interfaz			0.4	92.95
1.2.1	Navegabilidad			0.35	97
1.2.1.1	Orientación			0.7	100
1.2.1.1.1	Indicador del Camino	CB	100	0.6	60
1.2.1.1.2	Etiqueta de Posición actual		100	0.4	40
1.2.1.2	Promedio de históricos			0.3	90
1.2.2*	Objetos de control de Navegación			0.35	100
1.2.2.2	Nivel de ubicación exacta				100
1.2.2.2.1	Desplazamiento Vertical	CB	100	0.5	50
1.2.2.2.2	Desplazamiento Horizontal	CB	100	0.5	50
1.2.3*	Predicción navegacional	CB		0.3	80
1.2.3.1	Enlace con titulo	CDP	80	0.3	24
1.2.3.2	Calidad de Frase de Enlace	CDP	80	0.7	56

De acuerdo a la tabla 11 se puede ver que el grado de funcionalidad del sistema es de un 91.47% es decir funcione debidamente**.**

4.1.2. Fase de transición

4.1.3. Pruebas finales de aceptación

Una vez programada la aplicación Android y el software de escritorio, el siguiente paso es probar éstos para verificar que funcionen correctamente, además de buscar y corregir todos los errores antes de ser entregado al usuario final.
En este caso se realizarán las pruebas correspondientes a la aplicación mediante la implementación de técnicas de: caja negra y caja blanca, prueba de seguridad e inyección SQL, con el objetivo de garantizar la calidad, el buen funcionamiento de la aplicación Android y que ésta cumpla con todas las expectativas del usuario final.

4.1.4. Prueba de caja negra

La prueba de caja negra es una técnica de pruebas de software en la cual la funcionalidad se verifica sin tomar en cuenta la estructura interna de código, detalles de implementación o escenarios de ejecución internos en el software.
En las pruebas de caja negra, nos enfocamos solamente en las entradas y salidas del sistema, sin preocuparnos en tener conocimiento de la estructura interna del programa de software. Para obtener el detalle de cuáles deben ser esas entradas y salidas, nos basamos en los requerimientos de software y especificaciones funcionales.

4.1.4.1. Prueba de funcionalidad del prototipo

Durante la prueba de funcionalidad del prototipo se analiza que la aplicación cumpla con los casos de uso planteados por los involucrados en el desarrollo de la aplicación teniendo los siguientes resultados:
Donde:

- B: Bueno, aceptable, cumple con las expectativas.
- R: Regular.
- M: Malo, poco aceptable.

- Muestra 4: área del departamento de sistemas.
- Muestra 10: muestra de clientes de la empresa.

Tabla 12 Resultado del caso de uso: Gestión de usuario

Fuente: Elaboración propia.

CASO DE USO: GESTIÓN DE USUARIO

NRO.	ESCENARIO	DETALLE	INVOLUCRADO	PRE REQUISITO	OBSERVACIÓN	MUESTRA	ESTADO B	R	M
1	Habilita usuario	Habilita los usuarios de la aplicación Android.	Dpto. de Sistemas	Ingresar a la ventana de Usuarios del escritorio.	Debe tener acceso al software de software.	4	4	0	0
2	Lista usuario	Lista todos los usuarios registrados en el sistema.	Dpto. de Sistemas	Ingresar a la ventana de Usuarios del software.	Debe tener acceso al software de escritorio.	4	4	0	0
3	Adiciona usuario	Adiciona su usuario en el sistema.	Usuario	Ser cliente de la empresa.	Debe ser cliente de la empresa.	10	8	2	0
4	Modifica datos del usuario	Modifica sus datos en el sistema.	Usuario	Estar logueado en la aplicación.	Ninguna.	10	8	2	0

Resultado del caso de uso: Gestión de vehículos.

CASO DE USO: GESTIÓN DE VEHÍCULOS

NRO.	ESCENARIO	DETALLE	INVOLUCRADO	PRE REQUISITO	OBSERVACIÓN	MUESTRA	ESTADO B	R	M
1	Inhabilita vehículo	Inhabilita los vehículos del sistema.	Dpto. de Sistemas	Ingresar a la ventana de Vehículo del software.	Debe tener acceso al software de escritorio.	4	3	1	0

2	Habilita vehículo	Habilita los vehículos del sistema.	Dpto. de Sistemas	Ingresar a la ventana de Vehículo del software.	Debe tener acceso al software de escritorio.	4	3	1	0
3	Modifica vehículo	Gestiona la información de los vehículos del sistema	Dpto. de Sistemas	Ingresar a la ventana de Vehículo del software.	Debe tener acceso al software de escritorio.	4	4	0	0
4	Lista vehículos	Lista los vehículos que tiene asignados	Usuario	Estar logueado en la aplicación.	Ninguna.	10	7	2	1

Resultado del caso de uso: Gestión de notificaciones.

CASO DE USO: GESTIÓN DE NOTIFICACIONES									
NRO.	**ESCENARIO**	**DETALLE**	**INVOLUCRADO**	**PRE REQUISITO**	**OBSERVACIÓN**	**MUESTRA**	**ESTADO**		
							B	**R**	**M**
1	Adiciona notificación	Adiciona notificaciones para los usuarios.	Dpto. de Sistemas	Ingresar a la ventana de Notificación del software.	Debe tener acceso al software de escritorio.	4	3	1	0
2	Modifica parámetros de notificaciones	Modifica los parámetros de las notificaciones recibidas.	Usuario	Estar logueado en la aplicación.	Ninguna.	10	9	1	0
3	Lista notificaciones	Lista las notificaciones recibidas.	Usuario	Estar logueado en la aplicación.	Ninguna.	10	10	0	0

Resultado del caso de uso: Gestión de alertas.

Fuente: Elaboración propia.

CASO DE USO: GESTIÓN DE ALERTAS									
NRO.	**ESCENARIO**	**DETALLE**	**INVOLUCRADO**	**PRE REQUISITO**	**OBSERVACIÓN**	**MUESTRA**	**ESTADO**		
							B	**R**	**M**

1	Genera alerta	Genera las alertas en el sistema.	Dpto. de Sistemas	Ingresar al trigger de Notificaciones de la BD.	Conocimientos en SQL.	4	1	2	1
2	Recibe alerta	Recibe las alertas de las notificaciones.	Usuario	Estar logueado en la aplicación.	Ninguna.	10	10	0	0
3	Cancela alerta	Cancela la recepción de alertas.	Usuario	Estar logueado en la aplicación.	Ninguna.	10	9	1	0

Resultado del caso de uso: Gestión de recorrido.

CASO DE USO: GESTIÓN DE RECORRIDO									
NRO.	**ESCENARIO**	**DETALLE**	**INVOLUCRADO**	**PRE REQUISITO**	**OBSERVACIÓN**	**MUESTRA**	**B**	**ESTADO R**	**M**
1	Adiciona Vehículo	Agrega el vehículo al sistema para obtener el recorrido.	Dpto. de Sistemas	Ingresar el vehículo al sistema.	Ninguna.	4	3	0	1
2	Genera recorrido gráfico	Visualiza los puntos por donde pasó el vehículo.	Usuario	Estar logueado en la aplicación.	Se deben indicar los parámetros del recorrido.	10	9	1	0

De acuerdo a la tabla 12 se puede ver que el grado de eficiencia del sistema es de un 90.47%, esto quiere decir que existe un buen tiempo de respuesta y accesibilidad en la información.

4.1.4.2. Prueba del diseño del prototipo

En esta etapa se analiza el diseño y las respuestas que la aplicación ofrece en cuanto al uso que tiene el sistema, para esto se tomarán los siguientes aspectos:

Tabla 13 Aspectos de referencia para la prueba de caja negra

Fuente: Elaboración propia.

ASPECTO	**CARACTERÍSTICAS**
FACILIDAD DE USO E INSTALACIÓN	Software amigable, sencillo y de uso intuitivo. Instalación sencilla de manera fácil, rápida y transparente.
CALIDAD DEL ENTORNO GRÁFICO	Diseño estético, entendible y visualmente atrayente en las interfaces. Fácil navegación en: servicio Web, software de escritorio y aplicación Android.
NAVEGACIÓN E INTERACCIÓN	Buena estructuración que permita usar y acceder a todos los contenidos y prestaciones generales del sistema. Alta velocidad entre funcionalidad, conexión y respuesta a las peticiones.

RESPUESTA A ERRORES	Respuestas a errores validadas, entendibles y de fácil comprensión.

Los aspectos mencionados en la tabla anterior son esenciales para ver el nivel de respuesta y aceptación que tiene la aplicación, el servicio Web y el software de escritorio con respecto al uso que le darán los usuarios involucrados teniendo los siguientes resultados: Donde:

- Usuario 1 – 4: área del departamento de sistemas.
- Usuario 5 – 14: muestra de clientes de la empresa.

Tabla 14 Promedio de aceptación en cuanto a los aspectos de la prueba de caja negra

Fuente: Elaboración propia.

ASPECTO	FACILIDAD DE USO E INSTALACIÓN	CALIDAD DEL ENTORNO GRÁFICO	NAVEGACIÓN E INTERACCIÓN	RESPUESTA A ERRORES	PROMEDIO
Usuario 1	10	8	9	10	9.25
Usuario 2	9	10	8	9	9
Usuario 3	10	8	9	8	8.75
Usuario 4	10	9	9	9	9.25
Usuario 5	9	8	8	9	8.5
Usuario 6	8	9	9	9	8.75
Usuario 7	9	8	9	8	8.5
Usuario 8	10	7	8	9	8.5
Usuario 9	8	9	10	10	9.25
Usuario 10	9	10	9	8	9
Usuario 11	8	8	8	8	8
Usuario 12	7	8	9	7	7.75
Usuario 13	8	9	8	8	8.25
Usuario 14	10	8	8	8	8.5
	PROMEDIO TOTAL DE ACEPTACIÓN DEL USUARIO:				8.66

Teniendo los datos recogidos de los usuarios se puede concluir que entre un puntaje de nivel bajo (0) y un puntaje de nivel alto (10) los resultados en cuanto a los aspectos referentes de la prueba de caja negra se obtuvo un promedio de aceptación de 8.66 de 10, teniendo un nivel de eficiencia de 86.6% de 100% siendo este considerado bueno, aceptable.

4.1.5. Prueba de caja blanca

La prueba de caja blanca (también conocidas como prueba de caja de cristal o prueba estructural) se centra en los detalles procedimentales del software, por lo que su diseño está fuertemente ligado al código fuente.

La prueba de caja blanca consiste en probar la función interna del software y verificar que todo funcione correctamente, para esto se tomará en cuenta el flujo principal del funcionamiento que tiene el software (acceso al sistema) ya que hacer un análisis de todos los flujos existentes dentro del desarrollo resulta poco práctico, para esto se establece el conjunto básico de caminos donde se identifiquen los nodos y aristas principales del grafo para recorrer todos los caminos posibles.

4.1.5.1. Diseño de la prueba de caja blanca

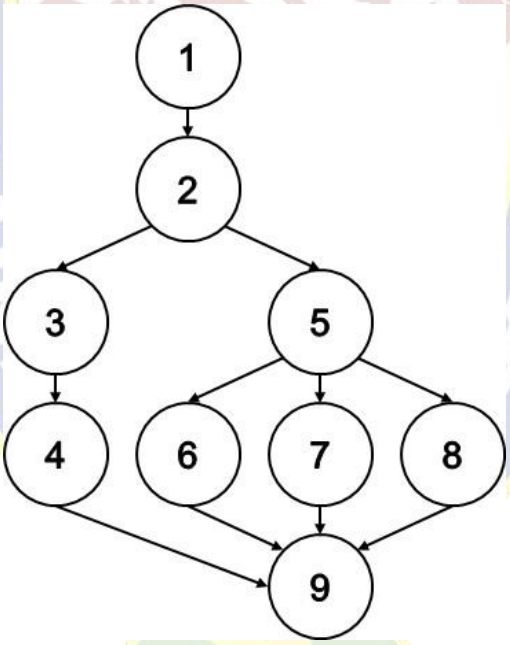


Figura 61 Grafo de flujo del flujo principal de la aplicación

Fuente: Elaboración propia.

- Se tienen los siguientes aspectos en el grafo:
- Aristas: 11
- Nodos: 9

- Una ruta independiente es una ruta distinta cada vez que se pasa por una arista nueva:

R1: 1, 2, 3, 4, 9

R2: 1, 2, 5, 6, 9

R3: 1, 2, 5, 7, 9

R4: 1, 2, 5, 8, 9

- El número de regiones es la suma de todos los caminos posibles en el grafo:

$$\sum R\,(n)$$

$$\sum R\,(n) = 4$$

- La complejidad ciclomántica V(G) de un grafo de flujo G se define como:

$$V(G) = \text{Aristas} - \text{nodos} + 2$$

$$V(G) = 11 - 9 + 2\ V(G) = 4$$

- La complejidad ciclomántica V(G) de un grafo de flujo G también se define como:

$$V(G) = \text{nodos de predicado } (p) + 1$$

$$V(G) = 3 + 1$$

$$V(G) = 4$$

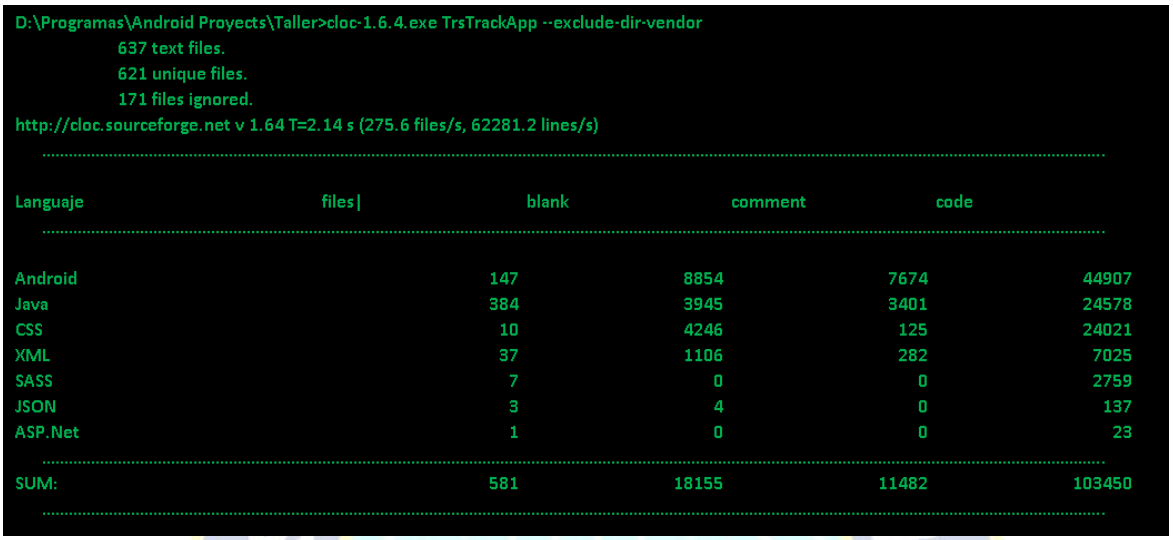
Una vez obtenido el valor de la complejidad ciclo mantica se puede identificar el número de caminos independientes.

Capítulo V

Evaluación Costo-Beneficio

5.1. Estudio de costo: aplicación del modelo COCOMO

Para el desarrollo del estudio costo-beneficio se usó el modelo COCOMO. Como primer paso se debe determinar el tipo de proyecto, para esto se utilizó la herramienta CLOC para calcular las líneas de código del proyecto obteniendo los resultados de la figura 63



```
D:\Programas\Android Proyects\Taller>cloc-1.6.4.exe TrsTrackApp --exclude-dir-vendor
      637 text files.
      621 unique files.
      171 files ignored.
http://cloc.sourceforge.net v 1.64 T=2.14 s (275.6 files/s, 62281.2 lines/s)
-------------------------------------------------------------------------------
Languaje                     files|       blank        comment           code
-------------------------------------------------------------------------------
Android                        147           8854           7674          44907
Java                           384           3945           3401          24578
CSS                             10           4246            125          24021
XML                             37           1106            282           7025
SASS                             7              0              0           2759
JSON                             3              4              0            137
ASP.Net                          1              0              0             23
-------------------------------------------------------------------------------
SUM:                           581          18155          11482         103450
-------------------------------------------------------------------------------
```

Figura 62 Reporte LDC realizado con CLOC

Fuente: Elaboración propia – Reporte de CLOC

Nos enfocaremos en los archivos con código JAVA y clasificaremos al proyecto como un proyecto orgánico al contar con 24578 líneas de código. Se utilizará el modelo intermedio de COCOMO utilizando conductores de costes. A partir de conductor de costes se seleccionan los multiplicadores mostrados en la Tabla 14:

Tabla 15 Resultado de multiplicadores

Fuente: Elaboración propia

CONDUCTORES DE COSTE	VALORACIÓN					
	Muy bajo	Bajo	Nominal	Alto	Muy alto	Extr. alto
Fiabilidad requerida del software	0,75	0,88	1.00	1,15	1,40	-
Tamaño de la base de datos	-	0,94	1.00	1,08	1,16	-
Complejidad del producto	0,70	0,85	1.00	1,15	1,30	1,65
Restricciones del tiempo de ejecución	-	-	1.00	1,11	1,30	1,66
Restricciones del almacenamiento principal	-	-	1.00	1,06	1,21	1,56
Volatilidad de la máquina virtual	-	0,87	1.00	1,15	1,30	-

Tiempo de respuesta del ordenador	**-**	**0,87**	**1.00**	**1,07**	**1,15**	**-**
Capacidad del analista	**1,46**	**1,19**	**1.00**	**0,86**	**0,71**	**-**
Experiencia en la aplicación	**1,29**	**1,13**	**1.00**	**0,91**	**0,82**	**-**
Capacidad de los programadores	**1,42**	**1,17**	**1.00**	**0,86**	**0,70**	**-**
Experiencia en S.O. utilizado	**1,21**	**1,10**	**1.00**	**0,90**	**-**	**-**
Experiencia en el lenguaje de programación	**1,14**	**1,07**	**1.00**	**0,95**	**-**	**-**
Prácticas de programación modernas	**1,24**	**1,10**	**1.00**	**0,91**	**0,82**	**-**
Utilización de herramientas software	**1,24**	**1,10**	**1.00**	**0,91**	**0,83**	**-**
Limitaciones de planificación del proyecto	**1,23**	**1,08**	**1.00**	**1,04**	**1,10**	**-**

El cálculo de los puntos de función se basa en la fórmula:

PF = Cuenta total* Factor de Ajuste

PF = 273 * 1.16

PF = 316.68

Conversión de los puntos de función a KDLC.

*LDC = PF * Factor LDC / PF LDC* = 316.68 * 29

LDC = 9183.72

KLDC= 91.83

Aplicando las fórmulas básicas del esfuerzo, tiempo calendario y personal requerido. Las ecuaciones del COCOMO tienen la siguiente forma:

E=ab(KLCD)db

D=cb(E) db

Dónde:

E: Esfuerzo aplicado en personas por mes.

D: Tiempo de desarrollo en meses.

KLDC: Número estimado de líneas de código distribuidas (en miles)

Tabla 16Coeficiente Ab y Bb

Fuente: Elaboración Propia

Proyecto de software	Ab	Bb	Cb	Db
Orgánico	2.4	1.05	2.5	0.38
Semi-acoplado	3	1.12	2.5	0.35

Empotrado	3.6	1.2	2.5	0.32

En la Tabla 16 se muestra los tipos de proyecto de software. Como este es un proyecto intermedio, en tamaño y complejidad, se elige Semi-acoplado.

E=3 *$(11.40)^{1.12}$

E= 45.79

D= 2.5 *$(45.79)^{0.35}$

D=9.53

El personal requerido se obtiene con la siguiente fórmula: Numero Programadores=E/D

Numero Programadores = 45.79/9.53

Numero Programadores = 4.80 = 5

EL salario de un programador aproximadamente es de 400 USD, cifra que se tornará en cuenta para la estimación siguiente:

Costo Software= Numero Programadores * Salario Programador

Costo Software= 5*400

Costo Software= 2000 USD

Teniendo así:

$$FAE = 0.94 * 1.07 * 0.86 * 1.13 * 0.86 * 0.9 * 0.95 * 0.91 * 0.91 * 1.08$$

$$FAE = 0.6428$$

Aplicando las fórmulas del modelo y utilizando las constantes de la Tabla 1 se obtienen los siguientes resultados:

$$E = a * KDLC^b * FAE(persona * mes)$$

$$E = 3.2 * 24.578^{1.05} * 0.6428\ (persona\ x\ mes)$$

$$E = \mathbf{59.333}(\boldsymbol{persona\ x\ mes})$$

$$T = c * Esfuerzo^d\ (meses)$$

$$T = 2.5 * 59.333^{0.38}\ (meses)$$

$$T = 11.79\ (meses)$$

$$p = \frac{E}{T}(personas)$$

$$p = \frac{59.333}{11.79}(personas)$$

$$p = \mathbf{5.03}(\boldsymbol{personas})$$

5.2. Estudio de beneficio

El producto de la empresa Total Radio Systems.ltda es la ubicación en tiempo real de vehículos, este es un producto intangible y sobre el cual el sistema desarrollado tiene influencia sobre las empresas de radio taxi. Con esto decimos que el sistema trae beneficios financieros a la institución.

Una de las principales falencias era el gasto que con llevaba instalar un modem en el vehículo que conducía a invertir más tiempo y gasto en la obtención especifica de información y la generación de diferentes tipos de reportes.

En la tabla 15 se describe los gastos antes y ahora:

Tabla 17 Comparación de tiempos de ejecución de tareas

FUENTE: ELABORACIÓN PROPIA – CÁLCULO DE TIEMPOS POR TAREA

TAREA	**Antes de la implementación**	**Después de la implementación**	**% de mejoría en tiempo**
REPORTE DE UBICACIÓN DE UN PROGRAMA	**2 minutos**	**1 minutos**	**50%**
REPORTE DE UBICACIÓN AL USUARIO	**2 minutos**	**0.1 minuto**	**75%**
ACCESO A LA INFORMACIÓN	**2 minutos**	**0.2 minuto**	**83.3%**
ACCESO A INFORMACIÓN PERSONAL DEL USUARIO	**1.5 minutos**	**0.1 minuto**	**66%**
RECOPILACIÓN DEL HISTORIAL DE UBICACIÓN	**1 minutos**	**0.5 minutos**	**50%**
OBTENCIÓN DE REGISTROS DE UBICACIÓN	**-**	**1 minuto**	
ELABORACIÓN DE FORMULARIO PARA LA OBSERVACIÓN DEL REGISTRO	**2 minutos**	**0.5 minutos**	**80%**
RESULTADOS AL USUARIO	**1 minutos**	**0.1 minutos**	**96.6%**

PROMEDIO DE TIEMPO DE MEJORA	**67.27%**

Además de estos beneficios tenemos el hecho de tener un resguardo de toda la información del registro de ubicación ejecutados desde el móvil que puede ser utilizado en la toma de decisiones al momento de la definición de nuevos programas.

5.2.1. Valor Neto Actual

El Valor neto Actual (VAN) es un indicador financiero que mide los flujos de los futuros ingresos y egresos que tendrá un proyecto, para determinar, si luego de descontar la inversión inicial, nos quedaría alguna ganancia. Si el resultado es positivo, el proyecto es viable.

Basta con hallar VAN de un proyecto de inversión para saber si dicho proyecto es viable o no. El VAN también nos permite determinar cuál proyecto es el más rentable entre varias opciones de inversión. Incluso, si alguien nos ofrece comprar nuestro negocio, con este indicador podemos determinar si el precio ofrecido está por encima o por debajo de lo que ganaríamos de no venderlo.

La fórmula del VAN es: VAN = BNA - Inversión

Donde el beneficio neto actualizado (BNA) es el valor actual del flujo de caja o beneficio neto proyectado, el cual ha sido actualizado a través de una tasa de descuento. La tasa de descuento (TD) con la que se descuenta el flujo neto proyectado, es ella tasa de oportunidad, rendimiento o rentabilidad mínima, que se espera ganar; por lo tanto, cuando la inversión resulta mayor que el BNA (VAN negativo o menor que 0) es porque no se ha satisfecho dicha tasa. Cuando el BNA es igual a la inversión (VAN igual a 0) es porque se ha cumplido con dicha tasa. Y cuando el BNA es mayor que la inversión es porque se ha cumplido con dicha tasa y además, se ha generado una ganancia o beneficio adicional.

VAN > 0 –> el proyecto es rentable.

VAN = 0 –> el proyecto es rentable también, porque ya está incorporado ganancia de la TD,

VAN < 0 –> el proyecto no es rentable.

Entonces para hallar el VAN se necesitan:

- Tamaño de la inversión.
- Flujo de caja neto proyectado.
- Tasa de descuento.

Veamos un ejemplo:

El proyecto tiene una inversión de 2000 USD y una tasa de descuento (TD) de 10%, por defecto.

Hallando el VAN: VAN = BNA - Inversión

$$VAN=1800 / (1 + 0.4)_{1} + 1800/(1+ 0.4)_{2} + 1800/(1 + 0.4)_{3} + 1800/(1 + 0.4)_{4} + 1800/(1 + 0.4)_{5} - 1800$$

$$VAN=583.54$$

Entonces V.A.N. > 0 por lo tanto esta en el rango de aceptable.

5.2.2. Tasa interna de retorno

La TIR es la tasa de descuento (TD) de un proyecto de inversión que permite que el BNA sea igual a la inversión (VAN igual a 0). La TIR es la máxima TD que puede tener un proyecto para que sea rentable, pues una mayor tasa ocasionarla que el BNA sea menor que la inversión (VAN menor que 0).

Entonces para hallar la TIR se necesitan:

- Tamaño de inversión.
- Flujo de caja neto proyectado.

Veamos un ejemplo: El proyecto t iene una inversión de 1200.

Para hallar la TIR hacemos uso de la fórmula del VAN, sólo que en vez de hallar el VAN (el cual reemplazarnos por 0), estaríamos hallando la tasa de descuento:

VAN= BNA - Inversion

$$0= 2000/ (1 + i)_{1} +2000 *1(1+i)_{2}+2000/ (l + i)_{3} + 2000\ 1(l+i)_{4} +20001(l + i)_{5} -2000$$

i = 10% T. I. R. = 10%

Si esta tasa fuera mayor, el proyecto empezaría a no ser rentable, pues el BNA empezaría a ser menor que la inversión. Y si la tasa fuera menor (como en el caso del ejemplo del VAN donde la tasa es de 10%), a menor tasa, el proyecto sería cada vez más rentable, pues el BNA sería cada vez mayor que la inversión.

CAPITULO VI

SEGURIDAD

La seguridad de un sistema, se lo define como un conjunto de medidas preventivas que se aplican para el resguardo y la protección de la información, buscando siempre mantener la confidencialidad, la disponibilidad y la integridad del mismo.
Así también, se tomó en cuenta aspectos como; la seguridad física (Infraestructura), la seguridad lógica, misma que es catalogada por niveles y se hace uso de las recomendaciones que propone la metodología OWASP.

6.1. Seguridad Física

La empresa Total Radio Systems Ltda cuenta con un circuito cerrado de cámaras de video vigilancia, mismas que están instaladas tanto en el exterior como en el interior de la mencionada Institución. Así también se cuenta con un sistema de control biométrico.

6.2. Seguridad Lógica

En esta parte, se tomó en cuenta los siguientes aspectos, la seguridad a nivel del sistema operativo, seguridad a nivel del sistema de gestión de la base de datos y a nivel de la aplicación desarrollada.

6.2.1. Seguridad a Nivel del Sistema Operativo

Las medidas de seguridad que se tomaron para evitar vulnerabilidades a nivel del sistema operativo fueron las siguientes:

- Colocación de un pasword al BIOS.
- Se bloqueó la línea de comandos del sistema operativo (cmd).
- Se bloqueó la opción de ejecutar del sistema operativo solamente el administrador puede hacer uso del mismo.

- Mediante el uso de la herramienta bitlocker se encriptó los discos duros de la empresa, al cual solamente el administrador del sistema tiene acceso.
- Solamente el administrador del sistema tiene acceso al servidor para la modificación de las contraseñas del usuario y otros.

Cabe mencionar que todas las computadoras de empresa cuentan con un sello de garantía.

6.2.2. Seguridad a Nivel del Sistema de Gestión de la Base de Datos

En este tipo de nivel las medidas que se tomaron fueron las siguientes:

- Se deshabilitó cualquier acceso vía http tanto a la parte de administración como a la API Rest.
- Se creó el usuario admin con el role “root” en la base de datos admin, para que solo este pueda tener acceso a la base de datos.
- Se tiene un plan de backups y recuperación de la base de datos.
- Las conexiones entrantes y salientes utilizan la configuración TLS / SSL.

6.2.3. Seguridad a Nivel de la Aplicación Desarrollada

Para este nivel lo que se hizo fue utilizar las recomendaciones de la metodología OWASP, las cuales se las describen a continuación:

6.3. Ataque Externo

Las pruebas para este punto hacen caso a las recomendaciones de la tabla 6.1

Tabla 18 Pruebas ciegas

Fuente: Metodo Prueba Ciega OWASP	
Código	**Tipo de Riesgo**
A1, A3,A4,A7	Validación de Entradas
A2	Administración de Contraseñas

Esta fase fue realizada mediante el uso de test de penetración, los cuales sirven para evaluar la seguridad de los equipos y las redes de comunicación.

- **Pentest Externo:** El objetivo de este tipo de test, es acceder en forma remota a los equipos de la institución y posicionarse como administrador del sistema. Se compone de un número considerable de pruebas.
- **Validación de Entradas:** Las entradas de las diversas interfaces de captura de datos fueron validadas mediante:
- **Input Validation:** Se verificó que la aplicación valida los datos ingresados en los distintos formularios y campos de ingreso de la aplicación.
- **Link Transversal:** Se intentó identificar URLs no utilizadas por la aplicación, enumerando las mismas o bien obteniéndolas del código fuente de una URL activa.
- **Path Truncation:** Se intentó truncar las URLs activas para obtener el listado de los archivos fuente desprotegidos.
- **Common File Checks:** Mediante la alteración de URLs, se intentó identificar y acceder a los archivos ocultos o no disponibles para los usuarios.
- **Hidden Web Paths:** Se intentó identificar paths, referencias dentro del código fuente del sitio o líneas comentadas con la finalidad de identificar URLs no autorizadas.

Como conclusión de esta fase, se pudo determinar que el nivel de probabilidad de riesgo es medio.

- **Administración de Sesión:** Se intentó realizar la contaminación de cookies y variables de sesión mediante el uso de la prueba Session Hijacking. El portal permite intentar un número determinado de intentos de ingreso al sistema, por lo que este ataque no es latente y sujeto a programas que intenten descubrir la contraseña por fuerza bruta. No se identificaron vulnerabilidades respecto a las restricciones a nivel de autenticación para el acceso al sistema. El nivel de probabilidad de riesgo es bajo. De igual forma los inicios de sesión de un determinado usuario en diversas terminales están controlados, el sistema no permite sesiones simultáneas, de una misma o de diferentes IP sean estas

internas o externas. Por lo que se determinó que el nivel de probabilidad de riesgo es bajo.

- **Infraestructura:** Se verificó que el sistema no es vulnerable a ataques Cross-Frame-Scripting (XFS). Es decir que los atacantes no pueden aprovecharse de las fallas que tengan algunos navegadores para acceder a los datos privados de un tercer sitio web. Se concluyó que su nivel de riesgo es bajo.
- **Pérdida de Control de Acceso:** A través de este tipo de prueba lo que se hizo fue buscar la elevación de privilegios, es decir, actuar como un usuario sin iniciar sesión o actuar como un administrador con una cuenta estándar. Así también se verifico el acceso forzado a páginas autenticadas como un usuario no autenticado o con un privilegió que no le corresponde, para ello se hizo uso del método de inyección Json. Esta prueba corresponde a la recomendación OWASP A5. Al finalizar las pruebas se concluyó que tiene una probabilidad de riesgo baja.
- **Revelación de la Información:** En esta parte se intentó revelar información relacionada a la administración de contraseñas del usuario. Este tipo de prueba corresponde a la recomendación A6 de OWASP. Cabe mencionar que el sistema cuenta con el tipo de cifrado de datos denominado SHA1. Al final de la prueba se pudo determinar que el nivel de probabilidad de riesgo es bajo.
- **Uso de Componentes con vulnerabilidades Conocidas:** Mediante el uso del analizador retire.js se hizo pruebas para determinar las vulnerabilidades en los componentes que conforman el sistema y si estos están actualizados y configurados correctamente, esta prueba corresponde a la recomendación A9 de OWASP. Al final de las pruebas realizadas se determinó que tiene una probabilidad de riesgo media.
- **Registro y Monitoreo Insuficientes:** Se realizó una serie de pruebas para determinar el nivel de vulnerabilidad del sistema cuando no tiene un monitoreo y registro adecuado, se verifico los inicios de sesión, los fallos a la hora de iniciar sesión, las advertencias, los umbrales de alerta y las pruebas de penetración que se realizaron con diferentes herramientas, se pudo verificar

que no existe una monitorización y alerta efectivos acerca de actividades que se realizan en el sistema, por lo cual se determinó que el grado de probabilidad de riesgo es alto, pero el impacto a nivel del negocio es mediano. Esta prueba corresponde a las recomendaciones OWASP A10.

La tabla 6.2 presenta un resumen de los factores de riesgo para las recomendaciones A1, A2, A3, A4 y A7, según la metodología de evaluación de riesgo de OWASP, además de una escala de riesgos en la cual para cada factor utiliza el rango de menos de 3 es bajo, a menos de 6 es mediano y de 6 a 9 es alto.

Tabla 19 Factore4s de Riesgos para la Validación de Entradas

Fuente: Propia								
Riesgo	Factores del agente de amenaza				Factores de vulnerabilidad			
A1, A2, A3, A4, A7	Nivel de Habilidad	Motivo	Oportunidad	tamaño	Facilidad De descubrimiento	Facilidad de explotación	Conciencia	Detección de intrusos
	5	2	7	1	3	6	9	2
Probabilidad general = 4.375 (mediano)								
Impacto Técnico					**Impacto de negocios**			
Pérdida de confidencialidad	Pérdida de Integridad	**Pérdida de disponibilidad**	**Perdida de responsabilidad**		**Daño financiero**	**Daño de reputación**	**Incumplimiento**	Violación de la privacidad
9	7	**5**	8		**1**	**2**	**1**	5
Impacto Técnico general= 7.25 (Alto)					Impacto general del negocio= 2.25 (Bajo)			

En la tabla 17 se puede ver que la probabilidad de vulnerabilidad es mediano, el impacto técnico es alto y el de negocio es bajo.

En la tabla 18 se puede observar que la probabilidad para una vulnerabilidad en el control de acceso es bajo como así también en el impacto técnico y de negocio.

Tabla 20 Factores de Riesgo de Componentes con Vulnerabilidad

Fuente: Propia								
Riesgo	Factores del agente de amenaza				Factores de vulnerabilidad			
A9	Nivel de Habilidad	Motivo	Oportunidad	tamaño	Facilidad De descubrimiento	Facilidad de explotación	Conciencia	Detección de intrusos
	5	2	7	1	4	6	7	4
Probabilidad general = 4.5 (mediano)								
Impacto Técnico					**Impacto de negocios**			
Pérdida de confidencialidad	Pérdida de Integridad	**Pérdida de disponibilidad**	**Perdida de responsabilidad**		**Daño financiero**	**Daño de reputación**	**Incumplimiento**	Violación de la privacidad
3	4	**5**	8		**1**	**2**	**3**	7
Impacto Técnico general= 5 (Mediano)					Impacto general del negocio= 3.25 (Mediano)			

La tabla 19 muestra que existe una probabilidad alta de que ocurra alguna vulnerabilidad en el sistema, así también en el impacto técnico y no así en el negocio cuyo riesgo de que ocurra una intrusión es mediano.

Tabla 21 Factores de Riesgo para el Registro y Monitoreo

Fuente: Propia		
Riesgo	Factores del agente de amenaza	Factores de vulnerabilidad

A9	Nivel de Habilidad	Motivo	Oportunidad	tamaño	Facilidad De descubrimiento	Facilidad de explotación	Conciencia	Detección de intrusos
	7	3	8	6	8	6	9	6
Probabilidad general = 6.625 (Alto)								
Impacto Técnico					**Impacto de negocios**			
Pérdida de confidencialidad	Pérdida de Integridad	**Pérdida de disponibilidad**	**Perdida de responsabilidad**		**Daño financiero**	**Daño de reputación**	**Incumplimiento**	Violación de la privacidad
9	7	**5**	8		**3**	**6**	**6**	8
Impacto Técnico general= 7.25 (Alto)					Impacto general del negocio=5.75 (Mediano)			

6.4. Ataque Interno

En esta etapa lo que se hizo fue utilizar pentest internos para determinar las vulnerabilidades y la seguridad interna.

Se estableció que puede hacer un atacante interno y hasta donde es capaz de penetrar en el sistema siendo un usuario con privilegios bajos. Este test también incluyó pruebas de:

- Análisis de protocolos internos y sus vulnerabilidades
- Autenticación de usuarios
- Verificación de permisos y recursos compartidos
- Test de los servidores principales (WWW, DNS, FTP, SMTP, etc.)
- Nivel de detección de la intrusión de los sistemas
- Análisis de la seguridad de las estaciones de trabajo
- Verificación de reglas de acceso
- Ataques de Denegación de Servicio.

Capitulo VII

Conclusiones y Recomendaciones

7.1. Conclusiones

Con el proyecto terminado y de acuerdo a las actividades definidas para el análisis e implementación de la aplicación en Flutter Track Car para la empresa Total Radio Systems Ltda dando como resultado Aplicacón fácil de comprender y de usar, dando así cumplimiento al objetivo principal del presente proyecto de grado. Respecto a los objetivos específicos se desarrolló e implemento el módulo de registro de usuarios, envio de eventos, comunicación por notificaciones y envió de geolocalización, mediante una interfaz intuitiva y de fácil manejo. El sistema logró mejorar los procesos de registro y monitoreo de vehículos por parte de la empresa, además del seguimiento de los mismos brindando información confiable y actualizada. Así también se logró cumplir con otro de los objetivos puesto que se desarrolló e implemento los módulos para la actualización y la baja de usuarios con lo cual se logra mejorar los procesos de monitoreo en la Institución. Con respecto al módulo de notificación de eventos se lo implementó con la particularidad de poder enviarlos a cualquier hora del día y recibirlo en todos los dispositivos. Se logra dar escalabilidad al sistema gracias al manejo de permisos (roles de usuario) a perfiles de acceso Se logra tener un sistema seguro y confiable. Respecto a la metodología utilizada para el desarrollo del sistema, se realizaron y cumplieron cada una de las fases de la metodología MOVILE-D que fueron de mucha ayuda para poder completar de manera ordenada y muy bien estructurada metódicamente el desarrollo del mismo, así también se realizaron con éxito las pruebas pertinentes a cada módulo. Finalmente se concluye que el sistema funciona correctamente y da cumplimiento a los objetivos planteados en un principio.

7.2. Recomendaciones

- Con la finalización del presente proyecto se realizan las siguientes recomendaciones para trabajos futuros:

- Mejorar el consumo de la aplicación con respecto a memoria y batería
- Aumentar más interfaces para control de usuario con respecto a sus eventos
- Aumentar un Módulo de contacto directo con la operadora de taxis
- Es recomendable actualizar la aplicación con respecto a las nuevas versiones que vayan saliendo de flutter y en base a su documentación
- Se recomienda crear módulo de captcha en la app a la hora de crear un usuario

Bibliografía

- https://www.qualitydevs.com/2019/07/05/que-es-flutter/
- http://devops.mediainline.com/que-es-visual-studio-code/#:~:text=En%20Visual%20Studio%20Code%2C%20tenemos,los%20principales%20lenguajes%20de%20programaci%C3%B3n.&text=Entre%20lo%20slenguajes%20soportados%20por,%E2%80%93%20T%2DSQL%20%E2%80%93%20TypeScript.
- https://codigofacilito.com/articulos/que-es-go
- https://devcode.la/blog/que-es-typescript/
- https://www.json.org/json-es.html
- https://www.php.net/manual/es/intro-whatis.php
- https://openwebinars.net/blog/que-es-dockerfile/#:~:text=Un%20Dockerfile%20es%20un%20archivo,de%20Docker%20es%20mediante%20Dockerfiles.
- https://codigofacilito.com/articulos/que-es-html
- Coarite, H. (2013) Sistema de monitoreo y control vehicular. Tesis de grado. Universidad Mayor de San Andrés.
- Quispe, I. (2016) Internet de las cosas, control y seguimiento de un automóvil. Tesis de grado. Universidad Mayor de San Andrés.

- Sarmiento, L (2011) Sistema de monitoreo con bloqueo digital de vehículos y servicios avanzados para empresas del transporte público interprovincial e interdepartamental. Proyecto de Grado. Universidad de Aquino – Bolivia.
- Santiago, R. et al. (2015) Mobile learning: nuevas realidades en el aula. Grupo Océano.
- Total Radio Systems Ltda. (2017) Historia. Recuperado de: https://totalradios.com/sobre-nosotros/
- Coelho, F. & Kuo Chen, C. (2013 – 2018) Significados – Marco Teórico. Recuperado de:https://www.significados.com/marco-teorico/
- Malavida (2017 – 2018) Android: La historia de Android de Apple Pie 1.0 a Pie 9. Recuperado de: https://www.malavida.com/es/analisis/la-historia-de-android

- Developers (2018) Versiones de la plataforma. Recuperado de https://developer.android.com/about/dashboards/?hl=es-419
- MSN (2017) ¿Qué es y para qué sirve Visual Studio? Recuperado de: https://www.msn.com/es-cl/noticias/microsoftstore/%C2%BFqu%C3%A9-es-y-para-qu%C3%A9-sirve-visual-studio-2017/ar-AAnLZL9
- Microsoft Docs (2011) Tecnologías y lenguajes de Visual Studio. Recuperado de: https://msdn.microsoft.com/es-es/library/bb514232(v=vs.100).aspx
- Microsoft Developer Nerwork (2018) ASP.NET y Visual Studio para Web. Recuperado de: https://msdn.microsoft.com/es-es/library/dd566231(v=vs.120).aspx

GLOSARIO

API: *Apllication Programming Interfaces* (interface de programación de aplicaciones), son un conjunto de comandos, funciones y protocolos que permiten crear programas específicos.

REST: *Representational State Transfer*, es un tipo de arquitectura de desarrollo web que se apoya totalmente en el estándar HTTP. Permite crear servicios y aplicaciones que pueden ser usados en cualquier dispositivo.

BSON: Es un formato de intercambio de datos usado para el almacenamiento y transferencia de datos en la base de datos MongoDB.

JSON: Acrónimo de JavaScript Object Notation es un formato de texto ligero para el intercambio de datos.

FRAMEWORK: Es un entorno de trabajo, es una estructura conceptual y tecnológica de asistencia definida, normalmente con artefactos o módulos concretos de software.

JQUERY: Es una biblioteca multiplataforma de JavaScript, permite simplificar la manera de interactuar con los documentos HTML.

BACKBONE: Es una línea de transmisión más grande que transporta los datos recogidos de líneas pequeñas que con las que se interconecta.

URL: Es una sigla del idioma inglés correspondiente a *Uniform Resource Locator* (Localizador Uniforme de Recursos). Se trata de una secuencia de caracteres que sigue un estándar y que permite denominar recursos dentro del entorno de Internet para que puedan ser localizados.

ROUTER: Es un direccionador el cual permite interconectar computadoras que funcionan en el marco de una red, se encarga de establecer una ruta en esta.

DOM: Es una abreviatura de *Document Object Model*, es la estructura de objetos que genera el navegador cuando se carga un documento y se puede alterar mediante javascript para cambiar los contenidos y aspecto de la página.

GRAFO: Es un conjunto de objetos llamados vértices o nodos unidos por enlaces llamados aristas que permiten representar relaciones entre elementos de un conjunto.

ASINCRONIA: Hace referencia al suceso que no tiene lugar en total correspondencia temporal con otro suceso.

BOWER: Es una herramienta que nos permite encontrar proyectos y herramientas de javascript y administrarlas dentro de nuestro proyecto.

NPM: Es el manejador de paquetes por defecto para Nodejs, un entorno de ejecución para JavaScript de Nodejs.

XSS: Del inglés *Cross Site Scripting*, es un tipo de inseguridad informática o agujero de seguridad típico de las aplicaciones web.

XFS: Es un sistema de archivos de alto rendimiento, de 64 bits y con soporte de jounaling. Se lo utiliza en servidores que manejan una gran carga de lectura y escritura a disco, bases de datos y sistemas de Hosting.

SHA1: Del inglés *Secure Hash Algorithm*, es un algoritmo con muchos otros algoritmos existente de encriptación o cifrado.

SALT: Es una biblioteca de JavaScript basada en el estado que ofrece una organización de código y un control de flujo, no importa cómo se escribe el código, el bucle de evento lo ve como una secuencia de funciones.

XML: Del inglés *Extensible Markup Language*, es un meta lenguaje de marcado extensible que permite definir lenguaje de marcas.

HTTP: Del inglés *Hypertext Transfer Protocol*, es el protocolo de comunicación que permite las transferencias de información en la *World Wide Web*.

SERVIDOR: Es una aplicación en ejecución capaz de atender las peticiones de un cliente y devolverle una respuesta en concordancia.

THREAD: Es una secuencia de tareas encadenadas muy pequeñas que pueden ser ejecutadas por un sistema operativo.

PUNTO FUNCION: Se la define como una función comercial de usuario final, mide la aplicación desde una perspectiva del usuario, dejando de lado los detalles de codificación.

MOCHA: Es un framework de pruebas de JavaScript que se ejecuta en Nodejs. Da la posibilidad de crear tanto tests asíncronos como síncronos.

CHAI: Es una librería de aserciones, la cual se puede emparejar con cualquier marco de pruebas de JavaScript, tiene varias interfaces que permiten al desarrollador elegir el estilo que le resulte más legible y cómodo a la hora de desarrollar sus test.

SERIALIZACION: Es un proceso de codificación de un objeto en un medio de almacenamiento con el fin de transmitirlo a través de una conexión en red como una serie de bytes o en un formato humanamente más legible como XML o JSON.

LINK: Es un elemento de un documento electrónico que hace referencia a otro recurso.

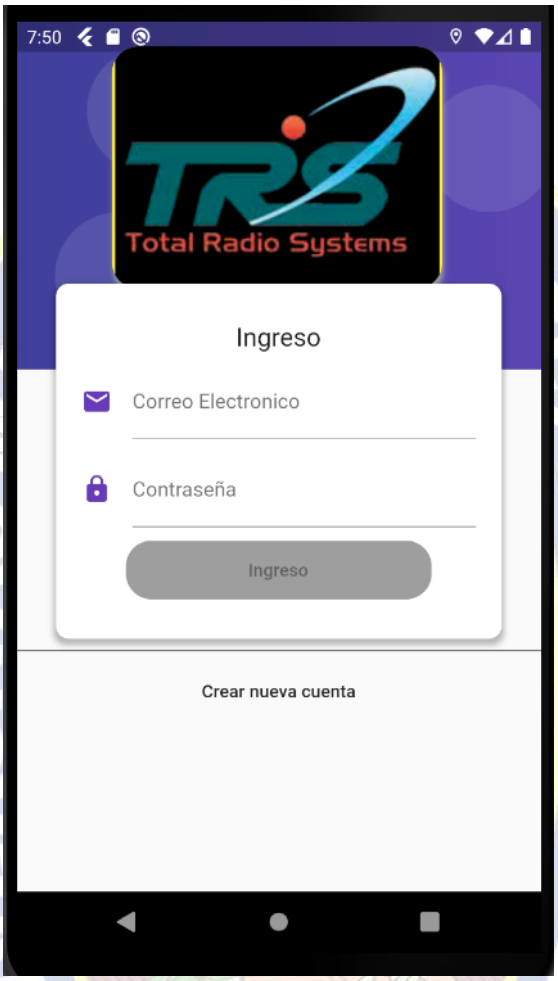
COMPLEJIDAD CICLOMATICA: Es una métrica de software que proporciona una medición cuantitativa de la complejidad lógica de un programa.

KARMA: Es una herramienta que monta un servidor web en donde es posible ejecutar código de fuente en diferentes clientes que se conectan a el. Un cliente es un navegador web

(Chrome, Firefox, etc.). Los resultados de cada test se examinan y se muestran al desarrollador a través de la terminal.

Anexos

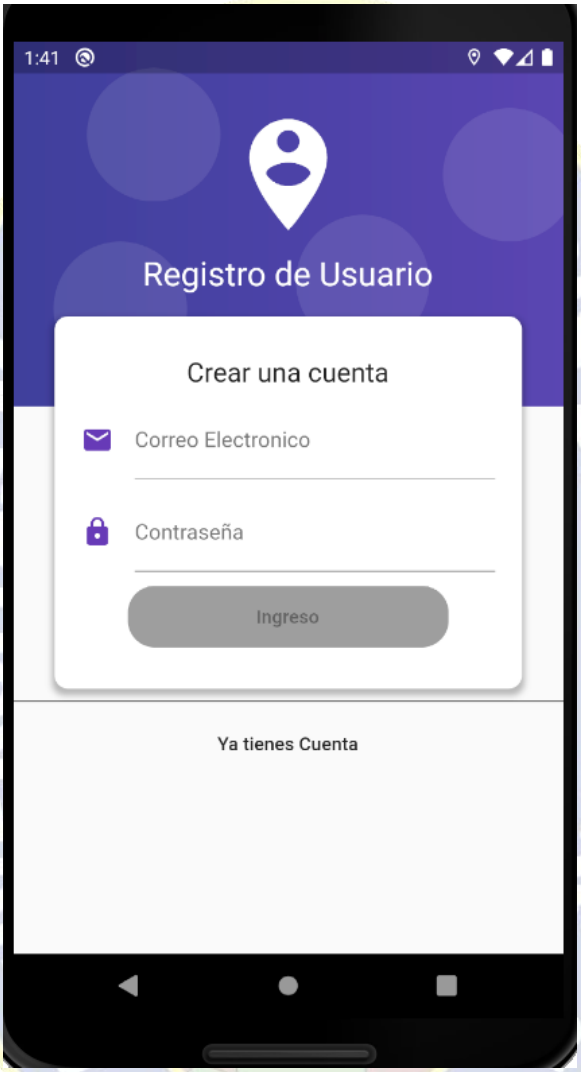
Anexo A – Pantalla de Inicio de la aplicación



Anexo B – Pantalla de Eventos de Usuario


8:55
Eventos de usuario
Estado Del Movil
En Servicio
Fuera de Servicio
En Espera
Ocupado

Anexo C – Registrod de Usuario



DOCUMENTACIÓN