



Universidad Nacional de Loja

FACULTAD DE LA ENERGÍA, LAS INDUSTRIAS Y LOS
RECURSOS NATURALES NO RENOVABLES

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y
TELECOMUNICACIONES

“Desarrollo de una aplicación móvil para dispositivos con OS
Android orientada a la identificación de Operadoras Móviles.”

TESIS DE GRADO PREVIA A LA
OBTENCIÓN DEL TÍTULO DE
INGENIERO EN ELECTRÓNICA Y
TELECOMUNICACIONES

Autor:

Jayro Michael Arias Tandazo

Director:

Ing. Ramiro Miguel Agila Astudillo, Mg. Sc.

LOJA – ECUADOR
2017

CERTIFICACIÓN

Ing. Ramiro Miguel Agila Astudillo, Mg. Sc.

DIRECTOR DE TESIS

CERTIFICA:

Haber dirigido, asesorado, revisado y corregido el presente trabajo de tesis de grado, en su proceso de investigación cuyo tema versa en “**Desarrollo de una aplicación móvil para dispositivos con OS Android orientada a la identificación de Operadoras Móviles.**”, previa a la obtención del título de Ingeniero en Electrónica y Telecomunicaciones, realizado por el señor egresado: **Jayro Michael Arias Tandazo**, la misma que cumple con la reglamentación y políticas de investigación, por lo que autorizo su presentación y posterior sustentación y defensa.

Loja de 07 de Agosto



Ing. Ramiro Miguel Agila Astudillo, Mg. Sc.

DIRECTOR DE TESIS

AUTORÍA

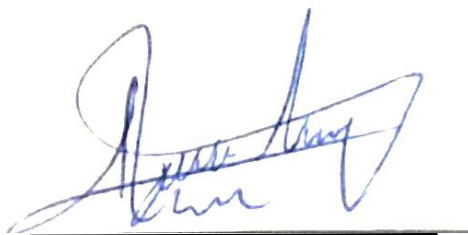
Yo **JAYRO MICHAEL ARIAS TANDAZO** declaro ser autor del presente trabajo de tesis y eximo expresamente a la Universidad Nacional de Loja y a sus representantes jurídicos de posibles reclamos o acciones legales por el contenido de la misma.

Adicionalmente acepto y autorizo a la Universidad Nacional de Loja, la publicación de mi tesis en el Repositorio Institucional – Biblioteca Virtual.

Firma:

Cédula: 1400846208

Fecha: 17 de noviembre del 2017



Jayro Michael Arias Tandazo

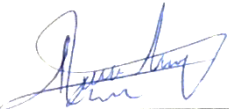
CARTA DE AUTORIZACIÓN DE TESIS POR PARTE DEL AUTOR, PARA LA CONSULTA, REPRODUCCIÓN PARCIAL O TOTAL Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO.

Yo, **JAYRO MICHAEL ARIAS TANDAZO**, declaro ser autor de la tesis titulada: **“DESARROLLO DE UNA APLICACIÓN MÓVIL PARA DISPOSITIVOS CON OS ANDROID ORIENTADA A LA IDENTIFICACIÓN DE OPERADORAS MÓVILES”**, como requisito para optar al grado de: **INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES**; autorizo al Sistema Bibliotecario de la Universidad Nacional de Loja para que, con fines académicos, muestre al mundo la producción intelectual de la Universidad, a través de la visibilidad de su contenido de la siguiente manera en el Repositorio Digital Institucional:

Los usuarios pueden consultar el contenido de este trabajo en el RDI, en las redes de información del país y del exterior, con las cuales tenga convenio la Universidad.

La Universidad Nacional de Loja, no se responsabiliza por el plagio o copia de la tesis que realice un tercero.

Para constancia de esta autorización, en la ciudad de Loja, a los diecisiete días del mes de noviembre del dos mil diecisiete.

Firma: 

Autor: Jayro Michael Arias Tandazo.

Cédula: 1400846208

Dirección: Loja (Argentina y Chile)

Correo Electrónico: jmariast@unl.edu.ec

Teléfono: 2580496 **Celular:** 0958775101

DATOS COMPLEMENTARIOS

Director de Tesis: Ing. Ramiro Miguel Agila Astudillo, Mg. Sc.

Tribunal de Grado: Ing. Juan Manuel Galindo Vera, Mg. Sc.

Ing. Antonio Arquímides Ramírez Gonzales, Mg. Sc.

Ing. Ángel José Ordoñez Mendieta, Mg. Sc.

DEDICATORIA

Dedico el presente trabajo de titulación al ser más influyente de mi vida, quien es mi motor, mi motivación, padre, amigo y consejero, mi Dios y Señor quien por medio de su hijo Jesucristo, me dado su perdón y día tras día en su infinito amor guía mis pasos, quien me ha concedido la sabiduría y entendimiento necesario para culminar con éxito esta carrera Universitaria.

A mi familia, mi madre una mujer luchadora y fuerte quien con su amor y comprensión ha sido un apoyo incondicional cada día de mi vida, a mi papá y mis hermanos por su compañía y palabras de apoyo siempre que las necesito.

A mi amigo y pastor David Ludeña quien con sus sabios consejos y amistad ha sabido inculcar en mí, un carácter disciplina y convicción ante las diferentes circunstancias de la vida.

Jayro Michael Arias Tandazo

AGRADECIMIENTO

Agradezco a mi Señor y Dios quien con amor ha llenado mi vida de fortaleza, predisposición y voluntad, con su apoyo incondicional desde los inicios de mi carrera me ha motivado a avanzar modulo a módulo esforzándome y dando lo mejor de mí.

A mis padres por su apoyo emocional y económico, por el aliento diario que día a día me saben darme, por su amor y comprensión durante toda mi carrera universitaria.

A mis compañeros por su amistad y ayuda incondicional, por todos los momentos vividos en el aula de clases.

A mis amigos que con gracia y paciencia me motivaron a seguir adelante a pesar de los momentos difíciles que como personas tenemos que afrontar.

A mis jefes por brindarme las facilidades necesarias para culminar con éxito la redacción del presente proyecto de titulación.

Y por último a una persona que marco de manera positiva mi vida, llegando inesperadamente, con su energía y alegría me motivo a acercarme a quien hoy en día es la razón de mi vida, mi Padre celestial, por sus sinceros consejos y el cariño que supo darme cuando yo más lo necesitaba.

ÍNDICE DE CONTENIDOS

CERTIFICACIÓN.....	II
AUTORÍA.....	III
CARTA DE AUTORIZACIÓN DE TESIS POR PARTE DEL AUTOR, PARA LA CONSULTA, REPRODUCCIÓN PARCIAL O TOTAL Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO.....	IV
DEDICATORIA	V
AGRADECIMIENTO	VI
1. TÍTULO.....	1
2. RESUMEN.....	2
ABSTRACT	3
3. INTRODUCCIÓN.....	4
4. REVISIÓN LITERARIA	6
4.1. Android.....	6
4.1.1. Generalidades.....	6
4.1.2. Plataforma.....	6
4.1.3. Arquitectura	6
4.1.3.1. Aplicaciones	7
4.1.3.2. Marco de aplicación.....	7
4.1.3.3. Librerías nativas y entorno de ejecución	7
4.1.3.4. Versiones.....	8
4.1.4. Android studio	9
4.1.4.1. Bibliotecas.....	9
4.1.4.2. Herramientas de desarrollo	10
4.1.4.3. Entorno de desarrollo	10
4.1.4.4. Interfaz de usuario	11
4.1.4.5. Sistema de compilación de gradle.....	12
4.1.4.6. Kit de desarrollo nativo (NDK).....	12
4.2. Php & Mysql	13
4.2.1. Php.....	13

4.2.1.1.	Extensiones en php	13
4.2.1.2.	Portabilidad.....	14
4.2.1.3.	Campos de aplicación	14
4.2.1.4.	Bases de datos	15
4.2.2.	Mysql	15
4.2.2.1.	Arquitectura	15
4.2.2.2.	Modelo de datos	16
4.2.2.3.	Tabla.....	16
4.2.2.3.1.	Tipos de.....	16
4.2.2.3.2.	Módulos de una tabla	18
4.3.	Servidor Web.....	20
4.3.1.	Arquitectura	20
4.3.1.1.	Funcionalidad	20
4.3.1.2.	Accesibilidad.....	21
4.3.1.2.1.	Peticiones	21
4.3.1.3.	Tipos.....	21
4.3.2.	Apache	22
4.3.2.1.	Generalidades	22
4.3.2.2.	Características	22
4.3.2.3.	Contenido.....	23
4.3.2.4.	Arquitectura	23
4.3.3.	Xampp.....	23
4.4.	Java	25
4.4.1.	Generalidades	25
4.4.1.1.	Características del lenguaje	25
4.4.1.2.	Entornos de desarrollo	26
4.4.1.3.	Aplicaciones de Escritorio.....	27
4.4.1.4.	Recursos	27
5.	MATERIALES Y MÉTODOS	29
5.1.	Materiales	29

5.2.	Métodos	29
5.2.2.	Método deductivo.....	29
5.2.3.	Método inductivo.....	31
5.3.	Técnicas.....	31
6.	DESARROLLO.....	32
6.1.	Diagnóstico de la contribución de operadoras móviles al incremento de los índices de pobreza en el país	32
6.1.1.	Operadoras móviles	32
6.1.2.	Usuarios y modalidades de telefonía móvil.....	32
6.1.3.	Nivel de pobreza en el país.....	34
6.1.4.	Telefonía móvil y su influencia en los niveles de pobreza nacional.....	34
6.3.	Propuesta.....	36
6.3.1.	Justificación	37
6.3.2.	Aplicativo móvil	37
6.3.2.1.	Interfaz de usuario.....	37
6.3.2.2.	Diagrama Plataforma IDMovil	39
6.3.2.3.	Diagrama UML IDMovil.....	40
6.5.	Desarrollo aplicación móvil.....	46
6.5.1.	Aplicación escritorio.....	46
6.5.2.	Aplicación web	53
6.5.4.	Aplicación móvil	64
7.	RESULTADOS.....	77
7.1.	Pruebas.....	77
7.1.1.	Opción llamar	77
7.1.2.	Opción Historial.....	77
7.1.3.	Opción Agenda	78
7.1.4.	Opción Favoritos.....	78
7.1.5.	Opción Sincronizar	79
8.	DISCUSIÓN.....	80
9.	CONCLUSIONES.....	82
10.	RECOMENDACIONES.....	83

11.	BIBLIOGRAFÍA	84
12.	ANEXOS	86

ÍNDICE DE FIGURAS

Figura 1. Arquitectura Android.....	7
Figura 2. Archivos del proyecto - Android Studio [12].....	11
Figura 3. Ventana principal Android Studio. [12].....	11
Figura 4. Estructura de una tabla. [14].....	16
Figura 5. Simbología de modalidad en las relaciones. [14].....	20
Figura 6. Interacción Cliente-Servidor. [24].....	21
Figura 7. Metodología Empleada [25].....	30
Figura 8. Niveles de Pobreza en el País. [18].....	34
Figura 9. Porcentaje de deciles que podrían acceder a un plan básico de telefonía. [22]..	35
Figura 10. Interacción y flujo de aplicativo IDMóvil. [26].....	38
Figura 11. Plataforma IDMóvil [26].....	39
Figura 12. Diagrama UML Interacción Aplicación de Escritorio (MisContactosMg) – Aplicación Web (MiscontactosWeb). [26].....	40
Figura 13. Diagrama UML Interacción Aplicación de Escritorio (MisContactosMg) – Aplicación Web (MiscontactosWeb) – Administrador IDMovil. [26].....	41
Figura 14. Diagrama UML Interacción Aplicación Móvil (MisContactos) – Aplicación Web (MiscontactosWeb). [26].....	42
Figura 15. Infografía servicio SMA Ecuador - junio 2017. [18].....	43
Figura 16. Interfaz Aplicación Escritorio. [26].....	47
Figura 17. Interfaz Opción Cargar Teléfonos. [26].....	47
Figura 18. Interfaz Opción Sincronizar. [26].....	47
Figura 19. Interfaz Opción Configurar Operadoras. [26].....	48
Figura 20. Interfaz Opción Configurar Operadoras - Nuevo. [26].....	48
Figura 21. Estructura lógica librerías – Aplicación de Escritorio. [26].....	48
Figura 22. Estructura lógica módulos miscontactosmng – Aplicación de Escritorio. [26]....	50
Figura 23. Estructura lógica clases - Aplicación Web. [26].....	55
Figura 24. Estructura lógica commonclases - Aplicación Web. [26].....	56
Figura 25. Estructura lógica base de datos Web. [26].....	58
Figura 26. Estructura General Base de Datos Web. [26].....	59

Figura 27. Estructura Secundaria Carga Base de Datos Web. [26].....	59
Figura 28. Estructura Secundaria Operador Base de Datos Web. [26].....	60
Figura 29. Estructura Secundaria Parámetro Base de Datos Web. [26].....	60
Figura 30. Estructura Secundaria Telefono01 Base de Datos Web. [26].....	60
Figura 31. Estructura Secundaria Telefono02 Base de Datos Web. [26].....	61
Figura 32. Estructura lógica base de datos escritorio. [26].....	61
Figura 33. Estructura General Base de Datos Escritorio. [26].....	62
Figura 34. Estructura Secundaria Carga Base de Datos Escritorio. [26].....	62
Figura 35. Estructura Secundaria Operador Base de Datos Escritorio. [26].....	63
Figura 36. Estructura Parámetro Operador Base de Datos Escritorio. [26].....	63
Figura 37. Estructura Teléfono01 Operador Base de Datos Escritorio. [26].....	64
Figura 38. Estructura Teléfono02 Operador Base de Datos Escritorio. [26].....	64
Figura 39. Estructura lógica librerías – Aplicación Móvil. [26].....	72
Figura 40. Estructura lógica Módulos – Aplicación Móvil. [26].....	75
Figura 41. Funcionalidad opción llamar aplicación móvil. [26].....	77
Figura 42. Funcionalidad opción historial aplicación móvil. [26].....	77
Figura 43. Funcionalidad opción agenda aplicación móvil. [26].....	78
Figura 44. Funcionalidad opción favoritos aplicación móvil. [26].....	78
Figura 45. Funcionalidad opción sincronizar aplicación móvil. [26].....	79
Figura 46. Interfaz de Usuario Aplicación de Escritorio. [26].....	86
Figura 47. Interfaz de Usuario – Opción Cargar Teléfonos. [26].....	86
Figura 48. Interfaz de Usuario – Opción Sincronizar. [26].....	86
Figura 49. Interfaz de Usuario – Configurar Operadoras. [26].....	87
Figura 50. Entorno de desarrollo Netbeans IDE 8.2. [26].....	88
Figura 51. Entorno de desarrollo Android Studio 2.3.1. [26].....	88

ÍNDICE DE TABLAS

Tabla 1. Evolución de Android	8
Tabla 2. Soporte de bibliotecas Android Studio	9
Tabla 3. Plataformas Soportadas. [14].....	14
Tabla 4. Usuarios de Telefonía Móvil Avanzada.....	33
Tabla 5. Usuarios de Telefonía Móvil Avanzada Registrados por Modalidad de Pago	33
Tabla 6. Tarifas prepago por minuto de voz.	35

1. TÍTULO

“DESARROLLO DE UNA APLICACIÓN MÓVIL PARA DISPOSITIVOS CON OS ANDROID ORIENTADA A LA IDENTIFICACIÓN DE OPERADORAS MÓVILES”

2. RESUMEN

El presente trabajo de titulación describe de manera práctica y con fundamentos teóricos el desarrollo por etapas de una aplicación móvil para dispositivos con OS Android orientada a la identificación de operadoras móviles con la capacidad de modificar la presentación de agenda contactos del dispositivo, identificando con una etiqueta de color rojo a la operadora móvil claro, verde a la operadora móvil movistar y azul a la operadora móvil cnt.

Cada etapa de desarrollo está orientada a un objetivo específico, partiendo por la elección de la interfaz de desarrollo de software SDK de Android como la adecuada, a criterio propio, capaz de adaptarse a las necesidades y conocimientos del desarrollador, plataforma en la que se desarrolló un algoritmo capaz de gestionar, procesar y comparar una base de datos web, con la agenda telefónica almacenada en el dispositivo y de esta manera brindar al usuario una herramienta que le permita el ahorro de recursos económicos por concepto de conexión de voz inter e intra operadoras.

ABSTRACT

The present work of the titration describes in a practical way and with theoretical foundations the development in stages of a mobile application for the devices with OS Android oriented to the identification of mobile operators with the ability to modify the device contacts presentation, identifying with a label red to mobile operator claro, green to movistar mobile operator and blue to cnt mobile operator.

Each stage of development is oriented to a specific objective, starting with the choice of the Android SDK software development interface as the appropriate, an own criteria, able to adapt to the needs and developer knowledge, an algorithm capable of managing, process and compare a web database, with the phonebook stored in the device and in this way provide the user with a tool that allows the savings of economic resources for the connection of voice inter and intra-operators.

3. INTRODUCCIÓN

La adaptación de la tecnología como herramienta de desarrollo socioeconómico en el mundo ha permitido elevar la calidad de vida de las personas que lo habitan, en la actualidad las TIC son consideradas un método de convergencia de servicios tecnológicos con el mismo propósito, en el ámbito económico el ahorro de recursos es sustancial para lograrlo.

Ante esto la Universidad Nacional de Loja como entidad estatal, propone el desarrollo de una aplicación móvil para dispositivos con OS Android orientada a la identificación de operadoras móviles, eligiendo este sistema operativo móvil por su gran acogida e imposición en la mayoría de Smartphone funcionales en el país.

Como característica esencial de la aplicación mencionada, modificará la presentación de agenda contactos del dispositivo identificando con una etiqueta de color rojo a la operadora móvil claro, verde a la operadora móvil movistar y azul a la operadora móvil cnt, esta distinción permitirá al usuario de cualquier operadora móvil elegir, en el caso de existir contactos asociados a más de una, la que más le convengan en cuanto a rubros de facturación se trate.

Por lo que se considera necesario el desarrollo de una aplicación móvil enfocada a la identificación de líneas telefónica por empresa prestadora de servicio para dispositivos Android, como un servicio universal para todos los usuarios dentro del país.

El aplicativo permitirá a los usuarios identificar a que operadora móvil pertenecen los contactos registrados en la agenda nativa del dispositivo, presentando para ello un distintivo de la operadora en la parte lateral de cada contacto.

El presente proyecto de titulación tiene como objetivo General: **Desarrollar una aplicación móvil para dispositivos con OS Android orientada a la identificación de Operadoras Móviles**, de igual manera sus objetivos específicos son:

- Desarrollar un algoritmo basado en SDK capaz de gestionar, procesar y comparar una base de datos web, con la agenda telefónica almacenada en el dispositivo con OS Android.

- Modificar por medio de la aplicación móvil la presentación de agenda contactos del dispositivo, identificando con una etiqueta de color rojo a la operadora móvil claro, verde a la operadora móvil movistar y azul a la operadora móvil cnt.
- Brindar al usuario una aplicación que le permita el ahorro de recursos económicos por concepto de conexión de voz inter e intra operadoras.
- Establecer vínculos de colaboración y cooperación entre la Superintendencia de Control del Poder de Mercado y la Universidad Nacional de Loja.

4. REVISIÓN LITERARIA

4.1. Android

4.1.1. Generalidades

Desde la aparición de dispositivos móviles con capacidades heredadas de los Handheld PC, Personal digital assistant, Web-enabled phone y el Teléfono móvil convencional, que posteriormente se denominaron teléfonos inteligentes “Smart Phone” el mercado tecnológico ha experimentado cambios sustanciales a favor de su demanda [1].

Sin embargo, no existía una estandarización del software que corre en ellos, producto de la diferenciación impuestas por los fabricantes, provocando de manera indirecta la división de la comunidad tecnológica mundial en subcomunidades orientadas en un principio al fabricante y las características que ofrece su producto, para luego pasar a orientaciones enfocadas en las características y ventajas que ofrece el software que corre en los dispositivos móviles (Smart Phone).

Actualmente la mayoría de fabricantes de Smart Phone se han rendido a la plataforma Android propietaria del gigante Google que lidera el mercado del software móvil, sin embargo aún existen algunos que apuestan a plataformas propietarias entre los más sobresaliente esta Nokia con el Windows Phone 8.1 Update 2 [2], Apple con iOS 9 [3], BlackBerry con BlackBerry 10 OS [4].

4.1.2. Plataforma

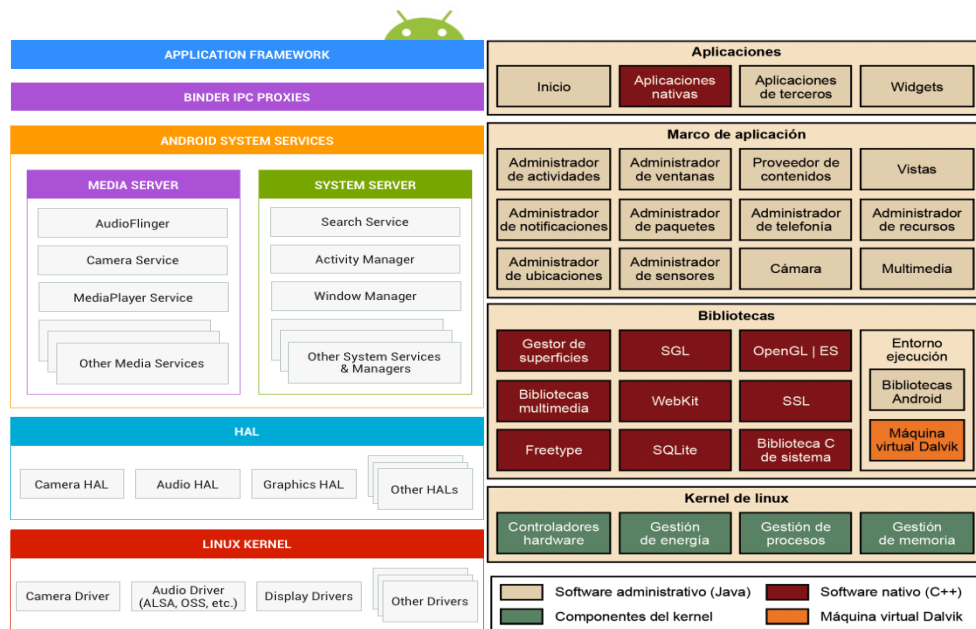
Está basada en Linux de la Open Handset Alliance (empresa dedicada al desarrollo de estándares abiertos para dispositivos móviles), usando como capa de abstracción de hardware su kernel (HAL). [1]

El desarrollo de sus aplicaciones se realizan sobre la SDK (kit de desarrollo de software) Android, que es adaptación de lenguaje Java, basada en se basa en *IntelliJ IDEA*, la API oficial de proyecto es conocido como *Android Studio*. [5].

4.1.3. Arquitectura

La plataforma de Android está concebida de manera jerárquica en torno a 4 o 5 capas diferenciadas entre las que se incluyen un sistema operativo, middleware (lógica de

intercambio de información entre aplicaciones) y aplicaciones básicas para el usuario, cada capa describe un nivel de operación, la figura 1 lo muestra de manera gráfica. [6]



a. Arquitectura de Android según [19].

b. Arquitectura de Android según [1].

Figura 1. Arquitectura Android

4.1.3.1. Aplicaciones

En este nivel se encuentran las aplicaciones instaladas en Android, siendo ejecutadas por seguridad del sistema en la máquina virtual (Dalvik o ART). Está formado por el conjunto de aplicaciones instaladas en una máquina Android. [1]

4.1.3.2. Marco de aplicación

Nivel al que se desarrollan las aplicaciones, de manera simplificada es el nivel en el que se definen las características y prestaciones para las que es concebida la aplicación que correrá en el dispositivo final. [1]

4.1.3.3. Librerías nativas y entorno de ejecución

Las librerías de Android están compiladas en código nativo C/C++ del procesador, entre las más destacables se tiene:

- Gestor de superficies.
- Scalable Graphics Library (SGL).
- OpenGL for Embedded Systems (OpenGLES).
- Bibliotecas multimedia.

- WebKit.
- Secure Sockets Layer (SSL).
- FreeType.
- SQLite.
- Biblioteca C de sistema (libc).

El concepto de entorno de ejecución de Android, hace referencia a la gestión optimizada de recursos propios del dispositivo, esto se logra a partir de una máquina virtual, dependiendo de la versión del OS existen dos máquinas virtuales que dan soporte a la ejecución de rutinas requeridas por las aplicaciones;

- **Virtual Dalvik:** para versiones inferiores a la 5.0.
- **Android Runtime:** para versiones a partir de la 5.0 y superiores.

La diferencia entre la Android Runtime sobre la Virtual Dalvik radica en el grado de optimización y eficiencia, permitiendo reducir el tiempo de ejecución del código java en un 33%. [7]

4.1.3.4. Versiones

Desde la aparición de su primera versión en el 2008 integrada en el QWERTY HTC G1 conocida como versión 1.0, hasta la actualidad, existen 13 versiones, de las cuales tan solo las 3 penúltimas son completamente operacionales, la tabla 1.0 describe un cronograma de evolución del Sistema Operativo Android. [8]

Tabla 1. Evolución de Android

Nombre	Versión	Lanzamiento	API
<i>Apple Pie</i>	v1.0	23 de septiembre de 2008	1
<i>Banana Bread</i>	v1.1	9 de febrero de 2009	2
<i>Cupcake</i>	v1.5	30 de abril de 2009	3
<i>Donut</i>	v1.6	15 de septiembre de 2009	4
<i>Éclair</i>	v2.0/v2.1	26 de octubre de 2009	5-7
<i>Froyo</i>	v2.2	20 de mayo de 2010	8
<i>Gingerbread</i>	v2.3	6 de diciembre de 2010	9-10
<i>Honeycomb</i>	v3.0/v3.1/v3.2	22 de febrero de 2011	11-13
<i>Ice Cream Sandwich</i>	v4.0	19 de octubre de 2011	14-15
<i>Jelly Bean/Gummy Bear</i>	v4.1/v4.2/v4.3	9 de julio de 2012	16-18
<i>KitKat</i>	v4.4	31 de octubre de 2013	19-20
<i>Lollipop</i>	v.5.0	3 de noviembre de 2014	21-22
<i>Marshmallow</i>	v.6.0/v.6.0.1	5 de octubre de 2015	23
<i>Nougat</i>	v.7.0/v.7.1.2	22 de agosto de 2016	24-25
<i>Oreo</i>	v.8	En desarrollo	--

4.1.4. Android studio

Es la plataforma integrada de desarrollo (IDE) de aplicaciones oficial de Android, basada en IntelliJ IDEA (Ambiente de desarrollo integrado), con características adicionales que presentan potentes herramientas de edición de código, a continuación se las enuncian [9]:

- Un sistema de compilación basado en Gradle flexible.
- Un emulador rápido con varias funciones.
- Un entorno unificado en el que puedes realizar desarrollos para todos los dispositivos Android.
- Instant Run para aplicar cambios mientras tu app se ejecuta sin la necesidad de compilar un nuevo APK.
- Integración de plantillas de código y GitHub para ayudarte a compilar funciones comunes de las apps 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.

Su versión más actual es la 2.3.1.

4.1.4.1. Bibliotecas

Está soportada según la versión de OS para la cual se diseña la aplicación. Su compatibilidad por versión se describe en la tabla 2.0. [10]

Tabla 2. Soporte de bibliotecas Android Studio

Biblioteca	Versión soportada
Biblioteca de compatibilidad v4	Android 2.3 y superior
Biblioteca de compatibilidad Multidex	Android 2.3 y superior
Bibliotecas de soporte v7	Android 2.3 y superior.
Biblioteca de compatibilidad v8	Android 2.3 y superior
Biblioteca de compatibilidad v13	Android 3.2 y superior
Biblioteca de compatibilidad Preference v14	Android 7.0 y superior
Biblioteca Preference v17 para TV	Android 7.0 y superior
Biblioteca Leanback v17	Android 7.0 y superior
Biblioteca de compatibilidad Annotations	Android 7.0 y superior
Biblioteca de compatibilidad Design	Android 7.0 y superior
Biblioteca de Custom Tabs Support Library	Android 7.0 y superior
Biblioteca de compatibilidad Percent	Android 7.0 y superior
Biblioteca de compatibilidad Recommendation para TV	Android 7.0 y superior

4.1.4.2. Herramientas de desarrollo

Hasta la actualidad existen 33 versiones de IDE Android Studio, teniendo en común herramientas como [11]:

- Entorno integrado de desarrollo IntelliJ IDE + Estudio plugin de Android.
- Herramientas del Kit de desarrollo de software Android (SDK).
- Herramientas para la plataforma Android.
- La versión de la plataforma Android.
- También incluye emulador de Android con servicios de Google Play.
- La versión 2.1.0 incluye soporte para NDK, pero requiere adaptaciones.

Las prestaciones de cada herramienta aumentan de manera ascendente dependiendo de la versión.

4.1.4.3. Entorno de desarrollo

4.1.4.3.1. Estructura del proyecto

Android Studio contiene uno o más módulos dependiendo de la funcionalidad del proyecto, que consisten en archivos de código fuente y archivos de recursos, como se ejemplo tenemos; [12]

- módulos de apps para Android.
- módulos de bibliotecas.
- módulos de Google App Engine.

Independientemente de los módulos a utilizarse Android Studio guarda un orden de presentación general con el objetivo de proporcionar un rápido acceso a los archivos de origen del proyecto. [12]

De manera específica un módulo está compuesto por las siguientes carpetas:

- **manifests:** contiene el archivo AndroidManifest.xml.
- **java:** contiene los archivos de código fuente de Java, incluido el código de prueba JUnit.
- **res:** Contiene todos los recursos, como diseños XML, cadenas de IU e imágenes de mapa de bits.

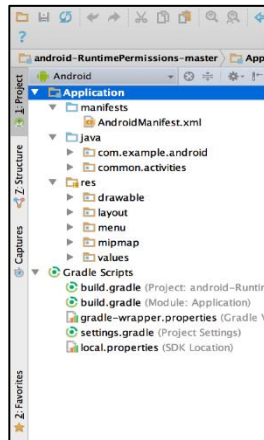


Figura 2. Archivos del proyecto - Android Studio [12]

4.1.4.4. Interfaz de usuario

Al igual que una aplicación de escritorio cualquiera, Android Studio presenta una GUI intuitiva y gráficamente bien elaborada, haciendo posible la distinción de varias áreas lógicas (Véase Figura .3.). [12]

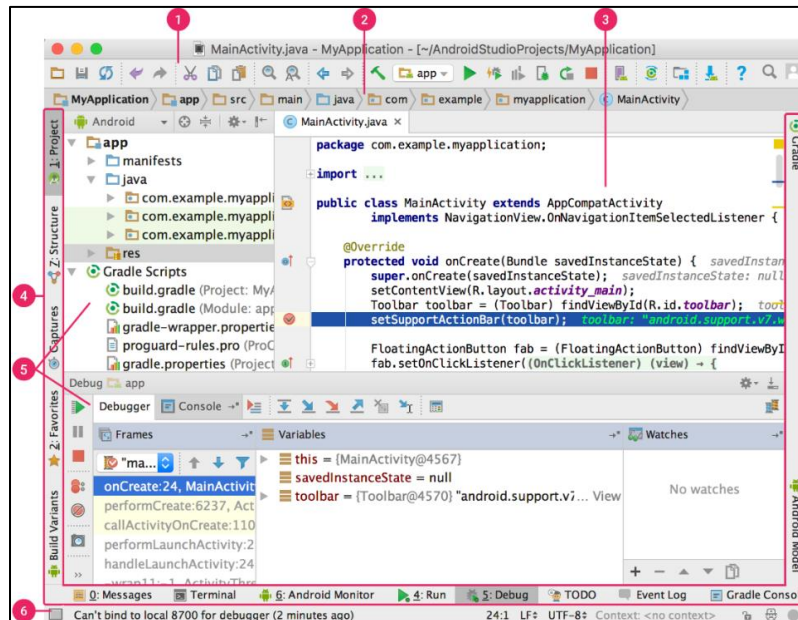


Figura 3. Ventana principal Android Studio. [12]

1. **Barra de herramientas:** permite realizar una gran variedad de acciones, como la ejecución de tu app y el inicio de herramientas de Android.
2. **Barra de navegación:** permite explorar el proyecto y abrir archivos para su edición.
3. **Ventana del editor:** área para la creación y modificación del código. Cuenta con un editor dinámico permitiendo visualizar el archivo actual.

4. **Barra ventana de herramientas:** se ubica lateral a la ventana del IDE, contiene los botones para la expansión o contracción de ventanas con herramientas individuales.
5. **Ventanas de herramientas:** su función es facilitar el acceso a tareas específicas, entre las que puede estar la administración de proyectos, las búsquedas, los controles de versión, entre otras, cuenta con características dinámicas de expansión y contracción.
6. **Barra de estado:** orientada a mostrar el estado del proyecto en desarrollo.

4.1.4.5. Sistema de compilación de gradle

Gradle es la opción que presenta Android Studio como la base del sistema de compilación, con características adicionales proporcionados por el complemento de Android, cuenta sintaxis basada en Groovy e independencia a la línea de comandos que permite usar las funciones del sistema de compilación para casos como [12]:

- Personalizar, configurar y extender el proceso de compilación.
- Crear múltiples APK para una app, con diferentes funciones utilizando el mismo proyecto y los mismos módulos.
- Reutilización de códigos y recursos entre conjuntos de archivos de origen.

4.1.4.5.1. Características de gradle

- Variantes de compilación: las características adicionales de gradle dan opciones a la creación de una aplicación en diferentes versiones a partir de un solo proyecto.
- Divisiones de APK: tiene como la finalidad la generación de varios APK en función de la densidad de la pantalla o ABI.
- Reducción de recursos: Tiene como objetivo la eliminación de recursos innecesarios en el proyecto, y así generar un App ligera y funcional.
- Administración de dependencias: gradle se encarga de la búsqueda de dependencias, tomando como referencia los nombre especificados en el archivo build.gradle, para suministrarlas en la compilación.

4.1.4.6. Kit de desarrollo nativo (NDK)

Utilizado en casos específicos en los que se requiere el uso intensivo de la CPU del equipo terminal, consiguiendo que la ejecución de la aplicación sea relativamente más rápida, dicha rapidez se consigue al ejecutarse directamente en el procesador sin pasar por un interpretador de una máquina virtual. [13]

4.2. Php & Mysql

4.2.1. Php

Hypertext Preprocessor o su acrónimo PHP, es una de las principales alternativas en el desarrollo web, considerado como un lenguaje de propósito general por la variedad de características con las que cuenta, entre las cuales podemos enunciar: [14]

- Programación de páginas dinámicas en servidores.
- Programación de aplicaciones de escritorio con GTK (PHPGTK).
- Soporte para trabajar con múltiples bases de datos.
- Soporte para múltiples plataformas.
- Soporte para múltiples servidores.
- Facilidad de aprendizaje.
- Portabilidad de código entre diferentes plataformas.
- Total, libertad para distribuir las aplicaciones.

4.2.1.1. Extensiones en php

PHP cuenta con extensiones formadas por conjuntos de funciones definidas como herramientas de programación, por su origen o residencia original es posible distinguir entre las que vienen incorporadas con el lenguaje (built in) y las que se encuentran en las bibliotecas añadidas, que debemos incluir en el sistema de manera específica. [14]

Por defecto PHP contiene, entre otras, las siguientes funciones:

- Manejo de matrices.
- Funciones matemáticas.
- BCMath.
- Manejo de clases/objetos.
- Manejo de variables de tipo carácter.
- Tratamiento de fecha/hora.
- Acceso directo a entrada/salida.
- Funciones de directorio.
- Gestión de errores y registros.
- Funciones de sistema de archivos.
- Utilización del protocolo FTP.
- Utilización del protocolo HTTP.
- Funciones de correo.

- Funciones de red.
- Ejecución de programas.
- Manejo de sesiones.
- Funciones de secuencia.
- Funciones de cadenas.
- Funciones URL.
- Manejo de variables.

4.2.1.2. Portabilidad

Al ser un lenguaje multiplataforma PHP está optimizado para trabajar sobre distintos sistemas operativos, reduciendo de manera significativa la adaptación de código al realizar una migración. [14]

Tabla 3. Plataformas Soportadas. [14]

	Windows	Mac	OSX	Linux	BSD	Unix z/OS
Adaptive Server Enterprise	SI	SI	SI	SI	SI	NO
ANT's Data Server	SI	SI	SI	SI	SI	?
DB2	SI	NO	SI	SI	NO	SI
Firebird	SI	SI	SI	SI	SI	Quizá
HSQLDB	SI	SI	SI	SI	SI	SI
Infomix	SI	SI	SI	SI	SI	NO
Ingres	SI	?	SI	SI	?	Quizá
InterBase	SI	NO	SI	SI	NO	NO
SapDB	SI	NO	SI	SI	NO	?
MaxDB	SI	NO	SI	SI	NO	?
Microsoft SQL Server	SI	NO	NO	NO	NO	NO
My SQL	SI	SI	SI	SI	SI	Quizá
Oracle	SI	SI	SI	SI	SI	SI
Postgre SQL	SI	SI	SI	SI	SI	NO
Small SQL	SI	SI	SI	SI	SI	SI
SQLite	SI	SI	SI	SI	SI	Quizá

4.2.1.3. Campos de aplicación

PHP se centra en tres campos principales, como fortalezas de su lenguaje. [14]

- **Scripts del lado del servidor:** Se impone a sus predecesores por su gran funcionalidad y recurrencia de desarrolladores. Su funcionalidad está relacionada con la interacción de un analizador de PHP (módulo CGI o servidor), un servidor web y un navegador web.

- **Scripts desde la línea de comandos:** Orientado a aplicaciones complementarias, son independientes de un servidor o navegador, necesitando solamente para su ejecución el analizador de PHP.
- **Escribir aplicaciones de escritorio:** A diferencia de lenguajes con orientación específicas al desarrollo de este tipo de aplicaciones, PHP también cuenta con complementos que pueden ayudarnos a crearlas, teniendo como condición un conocimiento avanzado del entorno PHP, la solución que presenta es un módulo externo conocido como PHP-GTK.

4.2.1.4. Bases de datos

La característica multifunción de PHP también se aplica en el soporte para un amplio abanico de bases de datos, adicionando sencillez en el desarrollo de página web con acceso a una base de datos siendo necesario únicamente una de las extensiones específicas de bases de datos (p.ej., para mysql), siendo posible también optar por una capa de abstracción como PDO, o conectarse a cualquier base de datos que admita el estándar de Conexión Abierta a Bases de Datos por medio de la extensión ODBC. [14]

4.2.2. Mysql

4.2.2.1. Arquitectura

Se basa en la arquitectura propuesta por la SI-SPARC (American National Standard Institute - Standards Planning and Requirements Committee) de tres niveles, que tiene como objetivo el separar las aplicaciones de la base de datos física, los niveles enunciados son:

- **Interno:** es el nivel más bajo de abstracción. Define todos los detalles de cómo funciona el almacenamiento en la base de datos y los métodos de acceso a la información. [14]
- **Conceptual:** este nivel se aleja de los detalles técnicos y se concentra en las necesidades de los usuarios. Se describen entidades, atributos, relaciones, operaciones de los usuarios y restricciones a sus acciones. Es una representación de los datos desde el punto de vista de una organización. [14]
- **Externo:** aquí se define qué partes de la base de datos podrán ser vistas y cuáles serán ocultas a qué usuarios. Es el nivel de mayor abstracción. [14]

4.2.2.2. Modelo de datos

Fundamentado en conceptos como; tabla (arreglo bidimensional), fila y columna, definiendo la recuperación de sus datos a través de lenguajes de consulta, de manera general definimos al modelo relacional como una propuesta para ver los datos como objetos del mundo real, diferenciables entre sí por sus características básicas (*atributos*), admitiendo, y en dependencia de del tipo de información que guardemos, tres tipos de relaciones; [14]

- Uno a uno.
- Uno a varios.
- Varios a varios.

La información de es agrupada en conjuntos de información llamados *tablas*.

4.2.2.3. Tabla

Se define como una colección de columnas y filas (entidad), que permite organizar los datos en filas (también llamadas tuplas o registros) y columnas (atributos o campos), teniendo semejanza con una matriz de dos niveles como una estructura de datos, pudiendo ser el primero el número de fila, y el segundo, el número de columna. [14]

El acceso a un valor de esta matriz, debe estar directamente relacionado con alguno de esos datos.

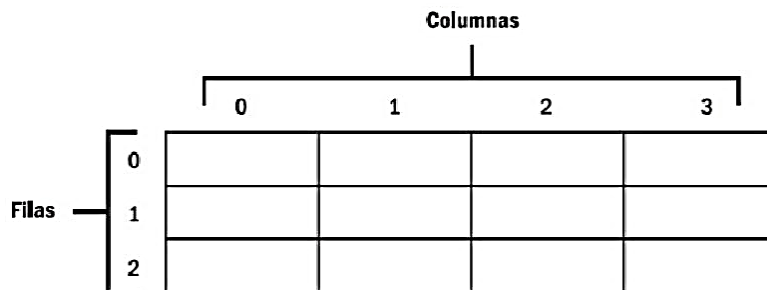


Figura 4. Estructura de una tabla. [14]

4.2.2.3.1. Tipos de tablas

- **Isam:** actualmente, se las considera en desuso, fue el primer tipo de tabla utilizado por MySQL. [14]
- **Myisam:** Tipo de tabla por defecto en MySQL se basa en su predecesor ISAM desde la versión 3.23 con mejoras significativas, entre la que destaca la posibilidad de un

almacenamiento independiente, es decir se pueden copiar tablas de una máquina a otra de distintas plataformas. [14]

Aquí algunas de sus características:

- Soportan archivos de gran tamaño (63 bits, ficheros de tablas superiores a 4 gigas) en comparación con los que soportaban las ISAM.
 - Optimizadas para sistemas operativos de 64 bits.
 - Posibilidad de indexar campos BLOB y TEXT.
 - Se permiten valores NULL en columnas indexadas.
 - Inserts concurrentes (pueden insertarse varios registros al mismo tiempo).
 - Cada tabla guarda un registro que indica si fue cerrada correctamente o no, y al iniciar MySQL existe la opción de indicarle que se verifique ese registro, y se repare la tabla de ser necesario de forma automática.
- **Merge:** Permite la gestión de un número N de tablas, por lo general cuando la tabla MyISAM original fuera de gran tamaño y acceder a su contenido llevara una cantidad considerable de tiempo y recursos. [14]

Aquí algunas de sus características:

- Solo se pueden aplicar instrucciones SELECT, DELETE y UPDATE.
 - La definición de la tabla se almacena en un archivo .FRM, y el listado de las tablas MySAM en un archivo MRG.
 - Permite de algún modo burlar el tamaño máximo de una tabla y el tamaño máximo de un archivo en un sistema operativo específico.
 - Si quisiéramos borrar una de las tablas MyISAM que forma parte de la tabla MERGE, no podríamos hacerlo bajo el sistema operativo Windows, ya que este no permite borrar archivos que estén abiertos, y la tabla al formar parte de la tabla MERGE se considera abierta.
- **Heap:** Son rápidas y puede ser utilizado por diferentes usuarios con la particularidad que son tablas en memoria, temporales y desaparecen cuando el servidor se cierra. [14]

Aquí algunas de sus características:

- No soportan columnas de tipo BLOB o TEXT.
- No soportan columnas de tipo AUTO_INCREMENT. T.

- No se permiten valores NULL en columnas indexadas (antes de MySQL versión 4.0.2).
- Siempre conviene especificar el número máximo de filas (MAX_ROWS) cuando se crea la tabla, para no usar toda la memoria disponible.
- **InnoDB:** son más lentas y ocupan más memoria, pero lo compensan con mayor seguridad frente a fallas durante la consulta esta disponibles desde la versión 4.0 de MySQL. [14]

Aquí algunas de sus características:

- Proveen la posibilidad de transacciones seguras, ACID, Atomicidad, Consistencia, Durabilidad.
- Recuperación ante fallas.
- Soporta FOREIGNKEY (claves foráneas).
- Bloqueo a nivel de fila.
- Permite realizar copias de seguridad mientras la base está funcionando.
- Alta eficacia en el procesamiento de grandes volúmenes de información.
- No permite crear claves sobre columnas de tipo BLOB o TEXT.
- Una tabla no puede tener más de 1000 columnas.
- **BerkeleyDB:** no es nativa de MySQL, pero ofrece una interfaz para trabajar con ellas como una posibilidad más. [14]

Aquí algunas de sus características:

- Soportan operaciones COMMIT y ROLLBACK.
- Es de tipo TST (Transactions Safe Tables).
- En el archivo en donde se almacenan los datos se guarda la ruta a ese mismo archivo, de modo que no es posible cambiar de directorio la base.

4.2.2.3.2. Módulos de una tabla

- **Atributo:** Diferencia una entidad de otra considerándose como características de las entidades. [14]
- **Relación:** Define vinculación entre ideas, un modo de asociar entidades/tablas para lograr el objetivo que tienen en común. [14]
- **Claves primarias:** Permite distinguir una fila de otra dentro de una tabla, su particularidad es la de no tener valores repetidos. [14]

- **Índices:** Son estructuras de datos donde se almacena información adicional acerca de una columna (una columna indexada), optimiza el proceso de búsqueda para el gestor de base de datos. [14]
- **Vistas:** Por temas relacionados con seguridad, orden entre otros motivos; las vistas nos permiten recortar una tabla y restringir el acceso a ella. [14]
- **Esquemas:** Está dotado por información referente al nombre de sus tablas, los atributos de cada una de estas, los dominios sobre los que se definen estos atributos, las claves primarias y las claves foráneas. [14]

Tipos de relaciones entre tablas

Se las puede catalogar según sus propiedades para el caso de estudio de detalla dos de ellas: la cardinalidad y la modalidad. [14]

“La cardinalidad especifica el número de instancias de una entidad que se puede relacionar con un número de instancias de otra entidad”. [14]

Tipos de cardinalidad:

- Uno a uno: haciendo referencia a dos tablas esta cardinalidad tiene lugar cuando, a cada elemento de la clave de la tabla A, se le asigna un único elemento de la tabla B y viceversa. [14]
- Uno a varios: tomando la misma consideración que el primer literal. esta cardinalidad tiene lugar cuando una clave de la tabla A posee varios elementos relacionados en la tabla B y cuando una clave de la tabla B posee un único elemento relacionado en la tabla A. [14]
- Varios a varios: Es poco común, pero el concepto de una cardinalidad varios a varios entra en ejecución cuando una clave de la tabla A posee varios elementos relacionados en la tabla B y, a su vez, una clave de la tabla B posee varios elementos relacionados en la tabla A, una adaptación de este cardinalidad es la crear una tercera tabla (C) que hace de intermediaria entre la A y la B que permanecen intactas. [14]

Modalidad

Define la intervención obligatoria o no de una instancia en una relación. Existente múltiples formas gráficas de definir la modalidad en las distintas relaciones, la forma clásica se la muestra en la figura. [14]

	Obligatoria
○	Opcional

Figura 5. Simbología de modalidad en las relaciones. [14]

Características

Entre las características sobresalientes con las que cuenta MySQL se tiene: [14]

- Rapidez.
- Posibilidad de trabajar en diferentes plataformas.
- Múltiples formatos de tablas para cada necesidad.
- Seguridad.
- Gran estabilidad.
- Administración simple.
- Soporte técnico (con el licenciamiento comercial).

4.3. Servidor Web

4.3.1. Arquitectura

Se basa en una arquitectura cliente-servidor, las solicitudes son generadas y enviadas por el equipo cliente y receptadas por el equipo servidor para dar una respuesta posterior. [15]

- **Acciones equipo cliente:** Proporciona al usuario una interfaz de comunicación y navegación con el servidor. [15]
- **Acciones equipo servidor:** Atiende las peticiones recibidas desde los navegadores o clientes web, incluyendo características de seguridad como autenticación de un usuario y una contraseña. [15]

4.3.1.1. Funcionalidad

Básicamente la funcionalidad de un servidor web tiene como objetivo proveer de páginas web a los clientes o navegadores que las solicitan, en el que interactúan tres componentes que permiten dar un accionar dinámico a la Web (Véase figura) [15]

Interacción Cliente - Servidor

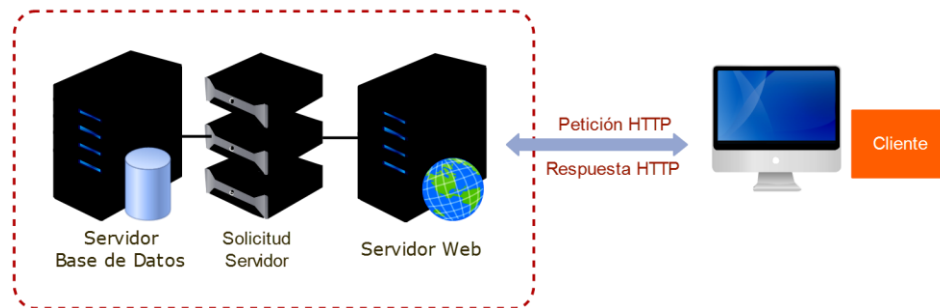


Figura 6. Interacción Cliente-Servidor. [23]

4.3.1.2. Accesibilidad

La comunicación entre el equipo cliente y el servidor se realiza por sobre el protocolo IP, las direcciones que se les asignan tanto al cliente como al servidor son almacenadas por un servidor DNS que permitirá realizar el redireccionamiento necesario para que la petición llegue al destino correcto. [15]

4.3.1.2.1. Peticiones

Cada petición que el usuario genera y es enviada por medio de un navegador contiene información acerca de acción específica que desea recibir, comúnmente se lo define como URL (Uniform Resource Locators), cada URL consta de: [15]

1. Protocolo a utilizar: http, https, ftp, ftps, etc.
2. //: hace las veces de separador.
3. Servidor: es la dirección IP o el nombre del servidor que contiene el recurso.
4. Ruta_al_recurso: directorio y subdirectorios del sitio web donde está ubicado el recurso.
5. Recurso: recurso al que se quiere acceder.

4.3.1.3. Tipos

Actualmente existen multitud de servidores web, pero los dos más conocidos e importantes, por sus niveles de implantación son Apache e IIS (Internet Information Service), distinguiéndose el uno del otro por su tipo de versión comercial, en el caso de Apache es un software libre y el caso del IIS es de carácter propietario (véase Tab.). [15]

4.3.2. Apache

4.3.2.1. Generalidades

Es un servidor web HTTP Apache2 Open Source desarrollado por la Apache Software Foundation (ASF), entre las principales fortalezas con las que cuenta, están: [15]

Apache	IIS
Es software libre	Es software propietario (Microsoft)
Es modular	Es modular a partir de la versión 6
Utilizado en los grandes servidores raíz de Internet	Utilizado en entornos corporativos
Ejecutado desde un usuario diferente del superusuario root del sistema	Ejecutado desde el propio usuario administrador del sistema hasta la versión 6

- **Estable:** es una consecuencia de su probada robustez que impide caídas o cambios en el servidor inesperados.
- **Flexible y eficiente:** es capaz de trabajar con el estándar HTTP/1.1 (RFC2616) y con la mayor parte de las extensiones web que existen en la actualidad, como son los módulos PHP, SSL, CGI, SSI, proxy.
- **Extensible:** dispone de gran cantidad de módulos que amplían su funcionalidad.
- **Multiplataforma:** ya que está disponible para diferentes plataformas como GNU/Linux, Windows, MacOS.

4.3.2.2. Características

Se incluyen las características más sobresalientes del servidor:

1. Cumple el estándar HTTP/1.1.
2. Configurable y extensible mediante módulos.
3. Proporciona el código fuente con licencia sin restricciones.
4. Multiplataforma.
5. Continua actualización y desarrollo.
6. Bases de datos DBM para autenticación de usuarios.
7. Permite crear páginas web protegidas.
8. Permite personalizar las respuestas a errores y problemas del servidor.
9. Múltiples directivas Directory Index.
10. Sin límite fijo sobre el número de Alias y redirecciones.
11. Negociación de contenido.

12. Hosts virtuales también llamados servidores multi-alojados.
13. Archivos logs configurables.

4.3.2.3. Contenido

La plataforma de apache proporciona contenidos al cliente web o navegador como: [15]

- **Páginas estáticas:** es el modo más básico y antiguo, pero también es el uso más generalizado que se hace de un servidor web. De esta forma se transfieren archivos HTML, imágenes, etc y no se requiere un servidor muy potente en lo que al hardware se refiere. [15]
- **Páginas dinámicas:** la información que muestran las páginas que sirve Apache cambia continuamente ya que se obtiene a partir de consultas a bases de datos u otras fuentes de datos. Son páginas con contenido dinámico, cambiante. [15]

4.3.2.4. Arquitectura

Su arquitectura se base en un modelo por capas, diferenciándose tres de ellas por su utilidad; [15]

1. de presentación que será la interfaz gráfica con la que interactúa el usuario.
2. para el servidor de aplicaciones que indicará cómo se ejecutan los procesos (modelo).
3. para guardar los datos (servidor de bases de datos).

4.3.3. Xampp

Es un servidor multiplataforma de código libre, y libre distribución, su principal característica es permitir instalar de forma sencilla Apache en el extremo del usuario, independientemente del sistema operativo (Linux, Windows, MAC o Solaris). [16]

Como característica adicional XAMPP incluye servidores de bases de datos entre los que encontramos, MySQL y SQLite con sus respectivos gestores phpMyAdmin y phpSQLiteAdmin, además incorpora también intérprete de PHP, el intérprete del Perl, servidores de FTP como ProFTPD o FileZilla FTP serve, entre otros componentes. [16]

4.3.3.1. Paquetes Incluidos

Paquetes básicos:

- Apache, el servidor Web más famoso.
- MySQL, una excelente base de datos de código libre.

- PHP y Perl: lenguajes de programación.
- ProFTPD: un servidor FTP.
- OpenSSL: para soporte a la capa de sockets segura. [16]

Paquetes gráficos:

- GD (Graphics Draw): la librería de dibujo de gráficos.
- libpng: la librería oficial de referencia de PNG.
- libjpeg: la librería oficial de referencia de JPEG.
- ncurses: la librería de gráficos de caracteres. [16]

Paquete de bases de datos:

- gdbm: la implementación GNU de la librería standard dbm de UNIX.
- SQLite: un motor de base de datos SQL muy pequeño y cero configuraciones.
- FreeTDS: una librería de base de datos que da a los programas de Linux y UNIX la habilidad de comunicarse con Microsoft SQL y Sybase.

Paquetes XML:

- expat: una librería parser de XML.
- Salbotron: una toolkit de XML.
- libxml: un parser C de XML y un toolkit para GNOME. [16]

Paquetes PHP:

- PEAR: la librería de PHP.
- Una clase pdf que genera documentos PDF dinámicos con PHP.
- TURCK MMCache: un potenciador de la performance de PHP. [16]

Otros paquetes:

- zlib: una librería de compresión.
- mod_perl: empotra un intérprete de Perl en Apache.
- gettext: un conjunto de herramientas que asiste a los paquetes GNU para producir mensajes multilinguales.
- mcrypt: un programa de encriptación.
- Ming: una librería de salida en Flash.
- IMAC C-Client: un API de correos. [16]

4.4. Java

4.4.1. Generalidades

Es un lenguaje de programación orientado a objetos, que toma la estructura de sintaxis (escritura de código fuente) de C y C++ incorporando un modelo de objetos más simples y eliminando las herramientas de bajo nivel (que suele inducir muchos errores como la manipulación de punteros en C y C++), por su estructura esta soportado para multiples plataformas desde Windows, Mac Os y la mayoría de las distribuciones de GNU/Linux. [17]

4.4.1.1. Características del lenguaje

- **Orientado a objetos:** es un software diseñado de forma que los distintos tipos de datos que usen estén unidos a sus operaciones, ofreciendo una base más estable para el diseño de un sistema software y conseguir que grandes proyectos sean fáciles de gestionar y manejar, mejorando como consecuencia su calidad y reduciendo el número de proyectos fallidos. [17]
- **Simple:** su lógica sencilla reduce hasta en un 50% los errores más comunes de programación con respecto a otros lenguajes eliminando la aritmética de punteros, registros, macros, definición de tipos, liberar memoria entre otros. [17]
- **Distribuido:** su lenguaje proporciona librerías y herramientas para que los programas sean distribuidos permitiendo su ejecución en varias máquinas e interactuando entre ellas. [17]
- **Multiplataforma plataforma:** su portabilidad de código está fundamentado en Java Virtual Machine (JVM) que es la máquina virtual que se encarga de interpretar y luego traducir al lenguaje máquina según la plataforma donde se ejecute. [17]
- **Robusto:** realiza verificaciones en busca de problemas desde la compilación hasta durante la ejecución detectando errores en ciclo de desarrollo: obligando la declaración explícita de métodos, también se encarga de la liberación de memoria. [17]
- **Seguro:** su enfoque depende de del nivel de aplicación;
A nivel de lenguaje: se eliminan punteros y casting implícito de los compiladores de lenguajes convencionales para prevenir el acceso ilegal a la memoria. [17]
A nivel de ejecución: el código Java pasa una serie de pruebas (test) antes de ejecutarse en la máquina, pasando el código a través de un verificador de código

bytecode que comprueba el formato de los fragmentos de código y aplica un verificador de teoremas para detectar fragmentos de código ilegal. [17]

A nivel de código fuente: el JDK proporciona un desensamblador de bytecode que permite que cualquier aplicación pueda ser convertido a código fuente, convirtiéndose en una herramienta muy potente tanto para desarrolladores éticos como para los que no lo son. [17]

- **Interpretado:** debido a que no existen JDK específicos para las diversas plataformas, java es lento respecto a otros lenguajes ya que primero deberá ser interpretado y no ejecutado como en C++. [17]
- **Multihilo:** es quizá la característica más potente con la cuenta, permite funciones simultáneas en una aplicación, obteniendo un mejor rendimiento interactivo y mejor comportamiento en tiempo real. [17]

4.4.1.2. Entornos de desarrollo

Por múltiples razones Java se ha convertido en unos de los lenguajes referentes de programación, su alcance es tal que se puede implementar en casi cualquier dispositivo o plataforma, a continuación, enumeramos sus entornos de desarrollo: [17]

- **Dispositivos Móviles y Sistemas Embebidos:** fue posible desde la creación de la especificación J2ME (una versión del entorno de ejecución Java reducido y altamente optimizado), orientado al mercado de dispositivos electrónicos, encontrando microprocesadores específicamente diseñados para ejecutar bytecode Java y software Java para tarjetas inteligentes (JavaCard), teléfonos móviles, buscapersonas, set-top-boxes, sintonizadores de TV y otros pequeños electrodomésticos. [17]
- **Navegador web:** actualmente no tiene mayores aplicaciones, está disponible desde la primera versión de java, consiste en desarrollar pequeñas aplicaciones en Java que luego pueden ser incrustadas en una página HTML para que sean descargadas y ejecutadas por el navegador web, fue un proyecto poco exitoso por diversos factores, siendo quizás el más importante la lentitud y el reducido ancho de banda de las comunicaciones en aquel entonces. [17]
- **Sistemas de servidor:** por otra parte, java ha tenido un gran despunte en esta parte, desde la aparición de la especificación de Servlets y JSP (Java Server Pages), su aporte radica: [17]

- El API de programación es sencillo, flexible y extensible. [17]
- Los servlets se ejecutan dentro del mismo proceso que la JVM mejorando notablemente el rendimiento y reduciendo la carga computacional y de memoria requeridas. [17]
- Las JSPs son páginas que se compilan dinámicamente de modo que el código que se consigue una tiene una ventaja en rendimiento substancial frente a muchos lenguajes interpretados. [17]
- A partir de su expansión entre la comunidad de desarrolladores, estas tecnologías han dado paso a modelos de desarrollo mucho más elaborados con frameworks que se sobreponen sobre los servlets y las JSPs para conseguir un entorno de trabajo mucho más poderoso y segmentado en el que la especialización de roles sea posible y se facilite la reutilización y robustez de código. [17]

4.4.1.3. Aplicaciones de Escritorio

Actualmente Java se encuentra presente en una multitud de aplicaciones gráficas de usuario, por su parte el entorno de ejecución Java (JRE) se ha convertido en un componente habitual en los PC de usuario de los sistemas operativos más usados en el mundo, las primeras versiones de la plataforma Java existían importantes limitaciones en las APIs de desarrollo gráfico (AWT), pero gracias a bibliotecas como Swing y SWT hacen que el desarrollo de aplicaciones de escritorio complejas y con gran dinamismo, sean relativamente sencillo. [17]

4.4.1.4. Recursos

- **JRE:** es el software necesario para ejecutar cualquier aplicación desarrollada para la plataforma Java. [17]
- **Bibliotecas:** Son el resultado de compilar el código fuente desarrollado por quien implementa la JRE, y que ofrecen apoyo para el desarrollo en Java incluyen: [17]
 - La API para acceso a bases de datos JDBC (Java DataBase Connectivity). [17]
 - La interfaz JNDI (Java Naming and Directory Interface) para servicios de directorio. RMI (Remote Method Invocation) y CORBA para el desarrollo de aplicaciones distribuidas. [17]

- Bibliotecas para la interfaz de usuario, que incluyen: El conjunto de herramientas nativas AWT (Abstract Windowing Toolkit), que ofrece componentes GUI (Graphical User Interface), mecanismos para usarlos y manejar sus eventos asociados. [17]
- Las Bibliotecas de Swing, construidas sobre AWT pero ofrecen implementaciones no nativas de los componentes de AWT. [17]
- **APIs:** Se definen tres plataformas globales como una manera de cubrir distintos entornos de aplicación:
 - **Java ME (Java Platform, Micro Edition) o J2ME** — orientada a entornos de limitados recursos, como teléfonos móviles, etc. [17]
 - **Java SE (Java Platform, Standard Edition) o J2SE** — para entornos de gama media y estaciones de trabajo. Aquí se sitúa al usuario medio en un PC de escritorio. [17]
 - **Java EE (Java Platform, Enterprise Edition) o J2EE** — orientada a entornos distribuidos empresariales o de Internet. [17]

5. MATERIALES Y MÉTODOS

5.1. Materiales

Conforme a su funcionalidad y área de aplicación, para llevar a cabo el desarrollo del proyecto en descripción, se seleccionaron los materiales enunciados a continuación:

- Php: Gestión y configuración de base de datos.
- MySql: Gestión de base de datos.
- Java: Desarrollo de aplicación de escritorio para gestión manual de datos y operadoras.
- Android Studio: Desarrollo de aplicación móvil para dispositivos con OS Android.
- Edraw Max: Diseño de prototipos del flujo de la aplicación móvil y de escritorio.
- Xampp: Gestión de interprete PHP, My Sql, Apache.

5.2. Métodos

5.2.1. Método científico

Nos permitió desarrollar la estructura secuencial para el desarrollo del proyecto (metodología empleada), partiendo por el planteamiento del problema, continuando por la formulación de una hipótesis como solución al problema planteado, y [18] culminar en la comprobación y comunicación de resultados obtenidos.

5.2.2. Método deductivo

Aplicable en la deducción de la estructura adecuada para la gestión y automatización de las consultas y asignaciones por parte del web service a la aplicación móvil, en la cual se tomaron las siguientes premisas:

- Desarrollo de un sistema de consultas basado en la interacción plataforma web (incluye gestión automática de actualizaciones de la base de datos, web service, procesamiento de la consulta) - aplicación móvil.
- Desarrollo de un sistema de consultas basada en la interacción aplicación escritorio (administrador) – aplicación web (incluye web service, procesamiento de la consulta) – aplicación móvil.
- Desarrollo de un sistema de consultas basada en la interacción servidor de bases de datos – aplicación móvil (procesamiento de la consulta).

5.2.2.1. Metodología

Consta de 5 etapas continuas, las cuales tienen dependencia entre sí, determinando con ello un orden no modificable ni sujeto a variaciones, se describe de manera gráfica en la siguiente figura.

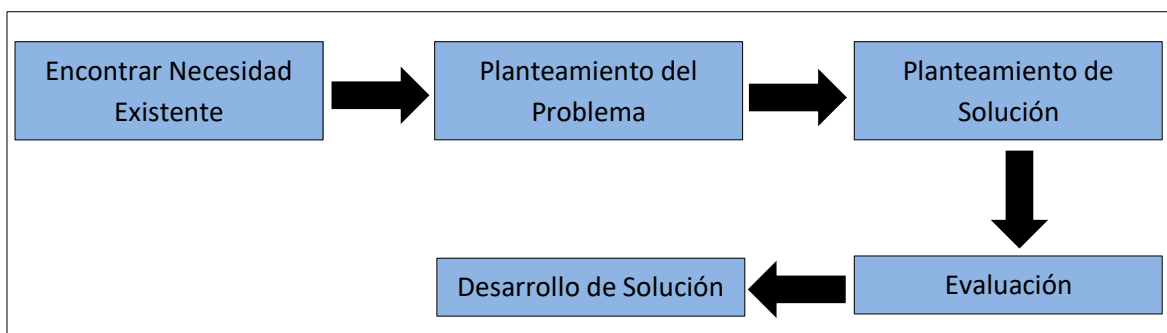


Figura 7. Metodología Empleada [24]

Los temas abordados por cada etapa se enlistan a continuación:

1. **Encontrar Necesidad Existente:** se define un sector a estudiar, y sobre el cual se determinará una necesidad relevante a ser cubierta, relacionada con las telecomunicaciones y su vinculación con la colectividad.

En esta etapa se incluye:

- Diagnóstico de la contribución de operadoras móviles al incremento de los índices de pobreza en el país.

2. **Planteamiento del Problema:** Una vez identificada la necesidad se procede a plantear el problema a resolver, tomando en cuenta para ello características relevantes a cumplir.

En esta etapa se incluye:

- Problema.

3. **Planteamiento de Solución:** Cubre cada una de las características consideradas dentro de planteamiento del problema.

En esta etapa se incluye:

- Propuesta.

4. **Evaluación:** esta etapa es fundamental, puesto es aquí donde se analiza la factibilidad del desarrollo de IDMóvil y su posterior desarrollo.

En esta etapa se incluye:

- Factibilidad.

5. Desarrollo de solución: es la etapa final del proyecto, donde se plasma la solución propuesta con anterioridad, y que pretender cubrir la necesidad existente.

En esta etapa se incluye:

- Desarrollo aplicación móvil.

5.2.3. Método inductivo

Aplicable en la redacción, discusión y evaluación de resultados obtenidos en el “Desarrollo de una aplicación móvil para dispositivos con OS Android orientada a la identificación de Operadoras Móviles”.

5.3. Técnicas

Investigación Bibliográfica: permite establecer las pautas a seguir para iniciar, desplegar y culminar satisfactoriamente la investigación del tema tratado, basándose en teorías, hipótesis, experimentos, resultados, instrumentos y técnicas usadas por otros investigadores.

6. DESARROLLO

6.1. Diagnóstico de la contribución de operadoras móviles al incremento de los índices de pobreza en el país

6.1.1. Operadoras móviles

Desde su llegada a Ecuador en agosto de 1993, de la mano de Conecel C.A., la telefonía móvil marco un precedente tecnológico a nivel nacional, imponiéndose a través de los años sobre la telefonía fija, a tal punto de llegar a considerarse en un servicio básico.

A la fecha las empresas existentes de telefonía móvil han centralizado su mercado en la comercialización de servicios desplegados sobre plataformas móviles, prevaleciendo entre todos, los de voz y datos móviles, la Agencia de Regulación y Control de las Telecomunicaciones registra la existencia de tres de las cuatro operadoras en funcionamiento con sus títulos habilitantes en vigencia:

- OTECEL (Movistar)
- CONECEL (Claro)
- CNT EP

TUENTI, aunque es considerada por la ciudadanía en conjunto como una empresa autónoma, está orientada a la prestación y comercialización de servicios de telefonía móvil avanzada sobre infraestructura de telecomunicaciones rentada, definido como un operador móvil virtual por el tipo de tecnología sobre el que corre su plataforma.

Según la ARCOTEL al cierre de enero del año 2017, el número de líneas reportadas por los prestadores del servicio de telefonía fija en Ecuador, alcanzó un total de 2.438.793 entre líneas de abonado y líneas para terminales de uso público, presentando de esta forma una densidad del 14,74%, indicador que obtuvo su valor más alto en el pasado mes de noviembre del año 2015 con 15,56%, observando a partir de este mes una estabilización del servicio, con tendencia a la baja.

6.1.2. Usuarios y modalidades de telefonía móvil

Tanto la modalidad prepago como pospago han experimentado grandes cambios a lo largo de los años, permitiendo al usuario acceder a nuevos tipos de servicios, sin embargo la demanda del servicio prepago mantiene su liderazgo con 243,76 % sobre el pospago (Tabla 4), acaparando una gran cantidad del mercado, estas cifras han mejorado notablemente, a

enero del 2017 la modalidad postpago ha tenido mayor despunte que en años anteriores atribuyéndoselo a la creciente necesidad de estar conectados (internet móvil) y comunicados (servicio de voz) entre usuarios.

El número de usuarios registrados a mayo del 2017 asciende a 15.061.858 (Véase Tabla 4).

Tabla 4. Usuarios de Telefonía Móvil Avanzada

MES/AÑO	CONECEL S.A.					TOTAL CONECEL
	CDMA	GSM	UMTS	HSPA +	LTE	
Abril 2107	0	991.865	4.139.688	2.789.011	859.164	8.779.728
May 2017	0	1.014.576	4.084.533	2.771.959	930.154	8.801.222
MES/AÑO	OTECEL S.A.					TOTAL OTECEL
	CDMA	GSM	UMTS	HSPA+	LTE	
Abril 2107	0	1.950.607	1.684.825	219.287	748.611	4.603.330
May 2017	0	2.017.118	1.550.688	148.788	820.559	4.537.153
MES/AÑO	CNT EP					TOTAL CNT
	CDMA	GSM	UMTS	HSPA +	LTE	
Abril 2107	0	161.918	0	288.623	1.270.868	1.721.409
May 2017	0	160.558	0	286.873	1.276.052	1.723.483
	TOTAL CDMA	TOTAL GSM	TOTAL UMTS	TOTAL HSPA +	TOTAL LTE	TOTAL
Abril 2107	0	3.104.390	5.824.513	3.296.921	2.878.643	15.104.467
May 2017	0	3.192.252	5.635.221	3.207.620	3.026.765	15.061.858

Tabla 5. Usuarios de Telefonía Móvil Avanzada Registrados por Modalidad de Pago

MES/AÑO	CONECEL S.A.			TOTAL CONECEL
	PREPAGO	POSPAGO	TTUP	
Abril 2017	6.323.871	2.439.808	16.049	8.779.728
May 2017	6.334.630	2.450.552	16.040	8.801.222
MES/AÑO	OTECEL S.A.			TOTAL OTECEL
	PREPAGO	POSPAGO	TTUP	
Abril 2017	3.254.403	1.314.458	34.469	4.603.330
May 2017	3.182.844	1.319.590	34.719	4.537.153
MES/AÑO	CNT EP			TOTAL CNT
	PREPAGO	POSPAGO	TTUP	

Abril 2017	1.136.078	580.313	5.018	1.721.409
May 2017	1.138.622	579.862	4.999	1.723.483
MES/AÑO	TOTAL PREPAGO	TOTAL POSPAGO	TOTAL TTUP	TOTAL
Abril 2017	10.714.352	4.334.579	55.536	15.104.467
May 2017	10.656.096	4.350.004	55.758	15.061.858

6.1.3. Nivel de pobreza en el país

Las cifras del último censo registrado por el INEC en el 2010, muestran una población aproximada de 14.483.499 ciudadanos ecuatorianos, con niveles de pobreza a diciembre de 2016, en el sector rural de 38,2% con una pobreza extrema del 17,6%, en el sector urbano se ubicó en el 15,7% con una pobreza extrema del 4,5%, a nivel nacional la pobreza se ubicó en 22,9% y la pobreza extrema en 8,7%.

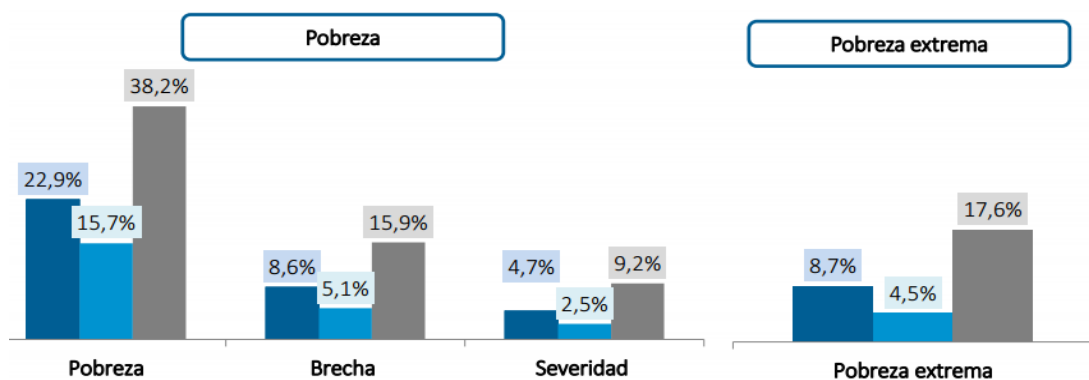


Figura 8. Niveles de Pobreza en el País. [18]

6.1.4. Telefonía móvil y su influencia en los niveles de pobreza nacional

Fundamentado en el Plan Nacional de Telecomunicaciones y Tecnologías de Información del Ecuador 2016-2021 que pretende ser el instrumento de planificación y gestión del sector de telecomunicaciones y Tecnologías de la Información y Comunicación que articule las políticas de desarrollo sectorial e intersectorial en materia de Tecnologías de la Información y Comunicación, para conseguir una mayor inclusión digital y competitividad del país, la penetración de la telefonía móvil junto con sus servicios avanzados y de voz alcanzan cada vez más usuarios, sin hacer exclusión de estatus social.

Visto desde la perspectiva de los gobiernos de turno, a más ciudadanos ecuatorianos se encuentren comunicados, su condición de vida tiende a mejorar, sin embargo, en los

sectores marginales del país esta realidad no se ve plasmada, las causas son deducibles, una mala administración de los ingresos familiares inferiores a los 100\$ mensuales (Figura 9).

Entre los servicios con más impacto en la economía de estas familias, está el servicio de voz en modalidad prepago, aquí los planes en general tienen poca acogida, resultaría poco veras el aceptar que aun en estos sectores la tecnología Smart ha tenido gran despliegue, encontrando terminales inteligentes(Smartphone) por debajo de los 50\$, pero es una realidad latente.

Datos oficiales proporcionados por SENATEL, en el 2014, mencionan que un plan básico de servicio móvil avanzado SMA resultaba inasequible para un 50% de los hogares, optando por planes libres de contrato (modalidad prepago), no obstante, el no tener establecido un límite de consumo, puede tener consecuencias negativas poco perceptibles para los usuarios, el caso más común es el de sobrepasar la tarifa del Plan Básico de Telefonía Móvil.

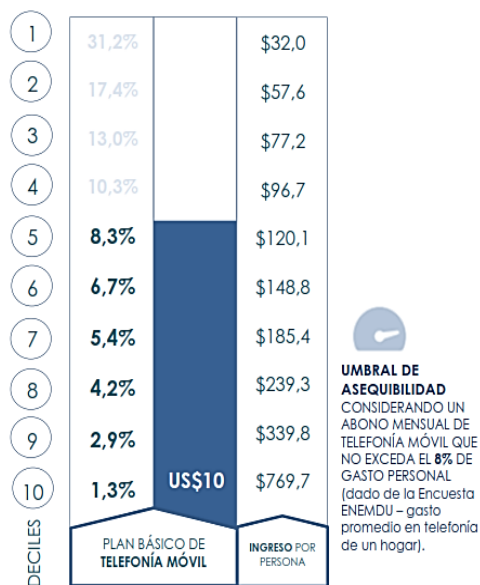


Figura 9. Porcentaje de deciles que podrían acceder a un plan básico de telefonía. [22]

Este gasto mensual del usuario promedio podría variar en relación a las políticas de OFFNET (Tabla 5) y ONNET de cada empresa (Tabla 6).

Tabla 6. Tarifas prepago por minuto de voz.

EVOLUCION TARIFARIA - PREPAGO OFF NET MAXIMAS				PREPAGO OFF NET MINIMA		
	2011	2012	2013 (jun.)	2011	2012	2013
Conecel S.A.	\$ 0,15	\$ 0,22	\$ 0,22	\$ 0,05	\$ 0,05	\$ 0,05
Otecel S.A.	\$ 0,22	\$ 0,18	\$ 0,18	\$ 0,08	\$ 0,08	\$ 0,08
CNT EP	\$ 0,18	\$ 0,22	\$ 0,22	\$ 0,05	\$ 0,05	\$ 0,05
Promedio	\$ 0,18	\$ 0,21	\$ 0,21	\$ 0,06	\$ 0,06	\$ 0,06
EVOLUCION TARIFARIA - PREPAGO ON NET MAXIMAS				PREPAGO ON NET MINIMA		
	2011	2012	2013	2011	2012	2013
Conecel S.A.	\$ 0,15	\$ 0,22	\$ 0,22	\$ 0,05	\$ 0,05	\$ 0,05
Otecel S.A.	\$ 0,16	\$ 0,18	\$ 0,18	\$ 0,08	\$ 0,08	\$ 0,08
CNT EP.	\$ 0,18	\$ 0,22	\$ 0,22	\$ 0,05	\$ 0,05	\$ 0,05
Promedio	\$ 0,16	\$ 0,21	\$ 0,21	\$ 0,06	\$ 0,06	\$ 0,06

6.2. Problema

Conforme a la información analizada en la primera etapa, se determina la necesidad de desarrollar un sistema que permita a los usuarios de telefonía móvil reducir sus gastos por concepto de tiempo aire, para el servicio Offnet y Onnet.

Siendo esto un problema relevante para familias que perciben ingresos inferiores a los 100,00 mensuales, por lo que un ahorro de al menos el 10% resultaría significativo en su calidad de vida.

Entre las prestaciones con las que deberá constar este sistema, está la de ser accesible para la mayor parte de los ciudadanos ecuatorianos y residentes en el país, incluyendo características intuitivas amigables con los usuarios y de portabilidad.

6.3. Propuesta

El tratar contrarrestar el impacto indirecto del de Plan Nacional de Telecomunicaciones y Tecnologías de Información del Ecuador 2016-2021, en los sectores con más pobreza del país, sería perjudicial para el proceso de desarrollo en el que actualmente nos encontramos, por lo que surge la necesidad de soluciones que trabajen en conjunto con sus objetivos,

bajo esta perspectiva nace la propuesta de IDMóvil, aplicación móvil para dispositivos con OS Android orientada a la identificación de Operadoras Móviles.

Su finalidad es contribuir al ahorro consiente de recursos económicos en la familia ecuatoriana, su principio de funcionamiento está basado en la identificación del operador móvil en el que se encuentre registrado cada uno de los usuarios adheridos a la agenda de contactos de un Smart Phone con sistema operativo Android.

6.3.1. Justificación

IDMóvil centra sus objetivos en el ahorro de recursos económicos en servicios voz (tiempo aire), aprovechando el conocimiento general intuitivo por parte de la ciudadanía sobre las tarifas ONNET y OFFNET que rigen en las diferentes operadoras móviles existentes en el país.

IDMóvil permitirá conocer a que operadora móvil se encuentra afiliado cada uno de los usuarios presentes en la agenda contactos del terminal inteligente con el que se cuente, esta característica pretende contribuir positivamente a la concientización en el ahorro de tiempo aire disponible en línea móvil del usuario.

El ahorro puede no ser representativo para familias con ingresos mayores a un salario básico, pero si para las que subsisten con ingresos inferiores a este, un gasto incensario repercutiría negativamente en su economía.

Otro aspecto a tomar en cuenta es que, hoy en día, por cuestiones labores y personales existen personas con más de una línea móvil registrada, IDMóvil con sus características permitirá identificar sin problema la operadora relacionada con cada una de estas líneas, aportando con esto también al ahorro por selección de operador.

6.3.2. Aplicativo móvil

6.3.2.1. Interfaz de usuario

El objetivo es concebir una aplicación intuitiva, con características básicas y comunes para los usuarios portadores de teléfonos inteligentes (Ver figura 10).

Constará de un menú con 5 opciones:

- *Opción realizar llamadas*



- *Opción historial llamadas*



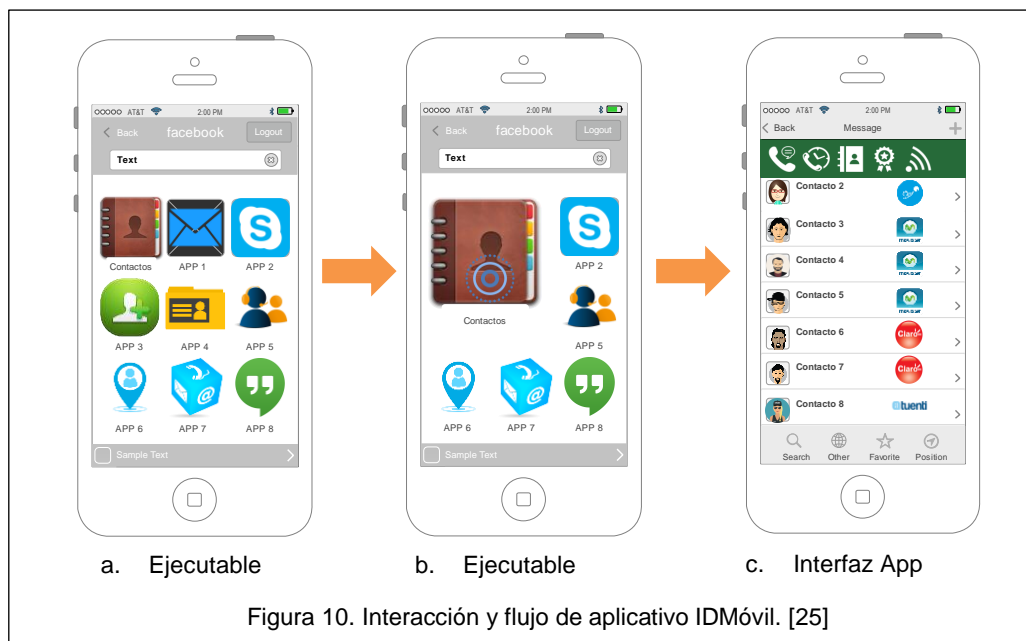
- *Opción contactos*



- *Opción favoritos*



- *Opción Sincronizar*



- Realizar Llamadas

Esta función redirecciona a la aplicación nativa dicado (marcar y llamar), su finalidad está orientada a la reducción de recursos en procesamiento.

- Historial Llamadas

Extrae el cache generado por la llamadas realizadas, rechazadas y aceptadas desde el terminal.

- Opción Contactos

Es la parte esencial de IDMóvil, es aquí donde los miles de datos numéricos serán comparados en procesos transparentes para el usuario, mostrando como resultado una agenda de contactos personalizada, donde es posible identificar la operadora móvil a la que se encuentra asociada dicho usuario.

Como característica adicional, esta agenda contendrá solo contactos con un número telefónico asociado (como mínimo), desechando cualquier otro que cumpla con esta propiedad.

La opción crear nuevo contacto no está pensada para esta interfaz, pues es posible redireccionarse a la opción llamar, y crearlo desde ahí, IDMóvil optimiza la mayor cantidad de recursos posible.

- Opción Favoritos

Permite seleccionar contactos recurrentes de nuestra agenda, obteniendo como resultado un listo de usuarios selecta y de rápido acceso.

- Opción Sincronizar

Tiene como objetivo permitir al usuario tener un control, sobre las actualizaciones de cambios y agregación de nuevos números a la plataforma de IDMóvil.

6.3.2.2. Diagrama Plataforma IDMóvil

Consta de 3 Módulos de software; Aplicación Móvil, Aplicación Web y Aplicación de Escritorio respectivamente.

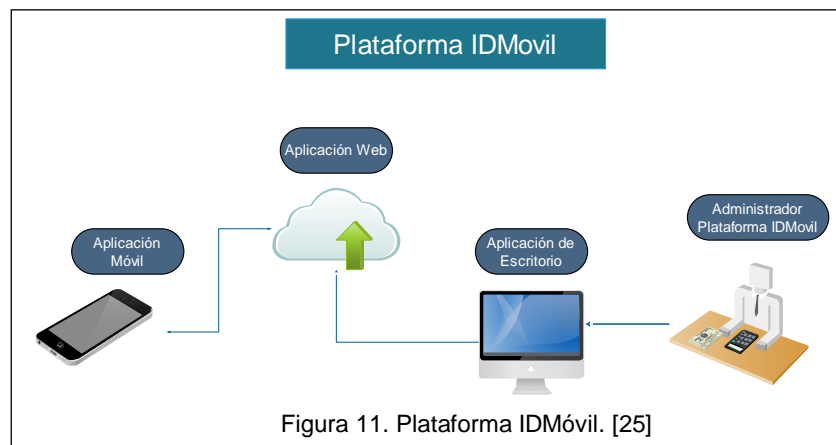


Figura 11. Plataforma IDMóvil. [25]

6.3.2.3. Diagrama UML IDMóvil

Interacción Aplicación de Escritorio (MisContactosMg) – Aplicación Web (MiscontactosWeb).

Los números de teléfonos en formato CSV se cargan a la base de datos del servidor (escritorio), a excepción de la primera vez, todos se comparan entre ellos para verificar si existen nuevos datos o modificaciones en alguno, a continuación, los números que se consideren nuevos o modificados son enviados para ser almacenados en un servidor web, esto es gestionado desde la aplicación de escritorio MisContactosMg.

Los números de teléfono enviados y almacenados en el servidor web, son comparados con la base de datos existentes, a excepción de la primera vez que se realice él envío, para efectuar modificaciones o añadir registros cualquiera sea el caso, esto es gestionado desde la aplicación de escritorio MisContactosWeb.

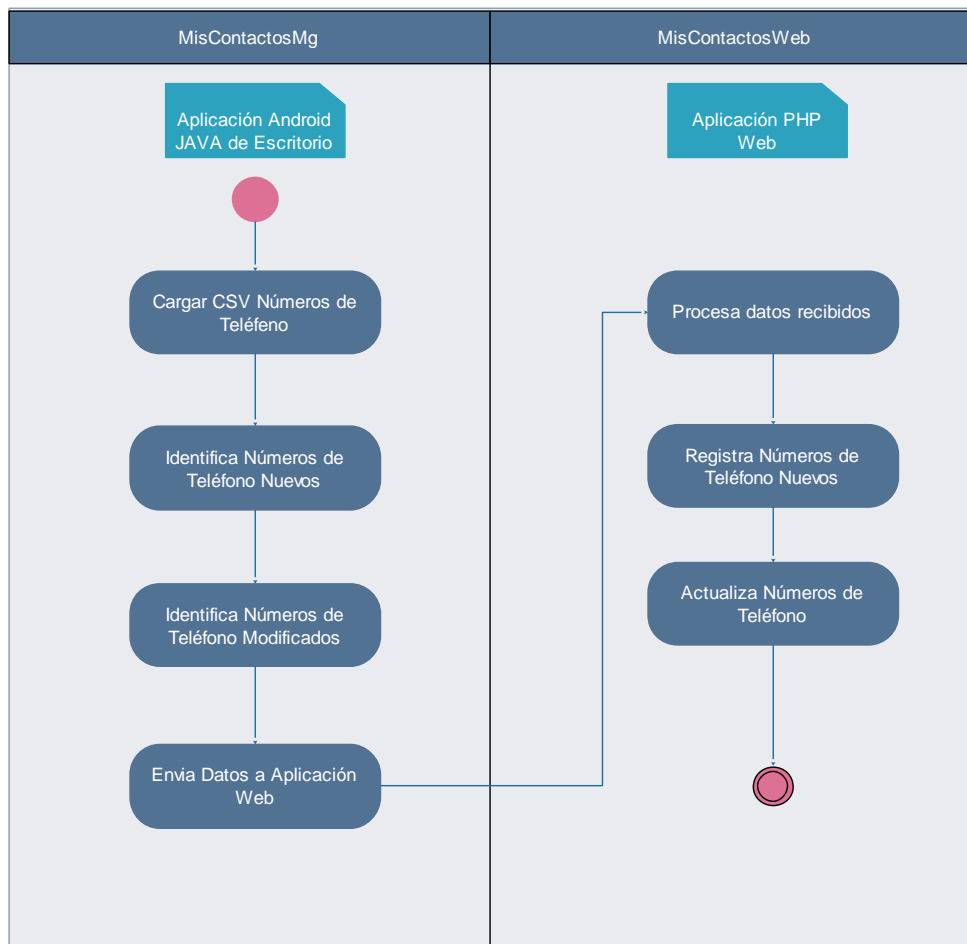


Figura 12. Diagrama UML Interacción Aplicación de Escritorio (MisContactosMg) – Aplicación Web (MiscontactosWeb). [25]

Interacción Aplicación de Escritorio (MisContactosMg) – Aplicación Web (MiscontactosWeb) – Administrador IDMóvil.

El registro, eliminación y modificación de las operadoras a las que pertenece un determinado número telefónico, se gestiona de manera independiente en otra tabla de registros, al igual que la carga de datos, el registro, eliminación y modificación también se lo realiza por medio de la aplicación de escritorio MisContactosMg.

Los registros de las operadoras modificadas, creadas o eliminadas son enviados a la base de datos web, para su respectiva actualización.

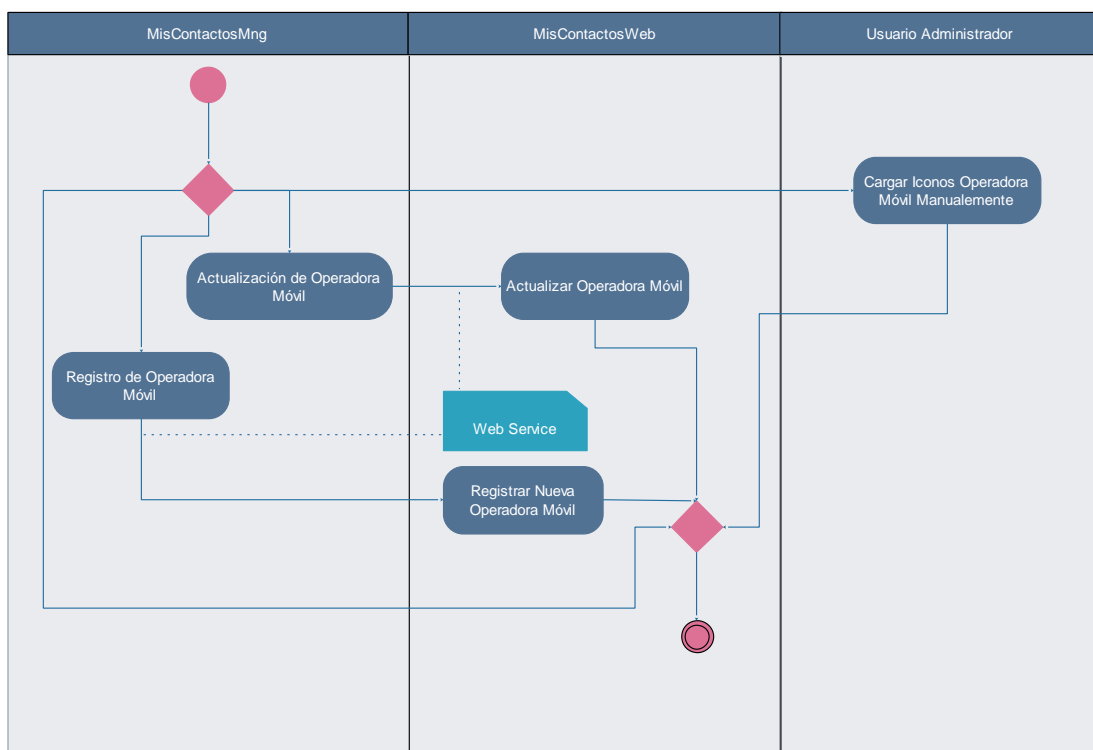


Figura 13. Diagrama UML Interacción Aplicación de Escritorio (MisContactosMg) – Aplicación Web (MiscontactosWeb) – Administrador IDMóvil. [25]

Interacción Aplicación Móvil (MisContactos) – Aplicación Web (MiscontactosWeb).

El aplicativo móvil, que corre en la plataforma de Android recolecta la información de cada contacto que incluya un número de teléfono como mínimo, descartando correos electrónicos y otro tipo de identificativo.

Esta información es enviada a la aplicación MisContactosWeb, por medio de un servicio web, el cual se encarga de consultar la información, referente a la operadora perteneciente, de cada contacto enviado.

A partir de la primera ejecución de la aplicación, la consulta solo se realizará con respecto a los nuevos usuarios o aquellos que hayan realizado portabilidad telefónica (cambio de operadora móvil conservando el mismo número), evitando de esta manera gasto innecesario de recursos de hardware en el Smart Phone.

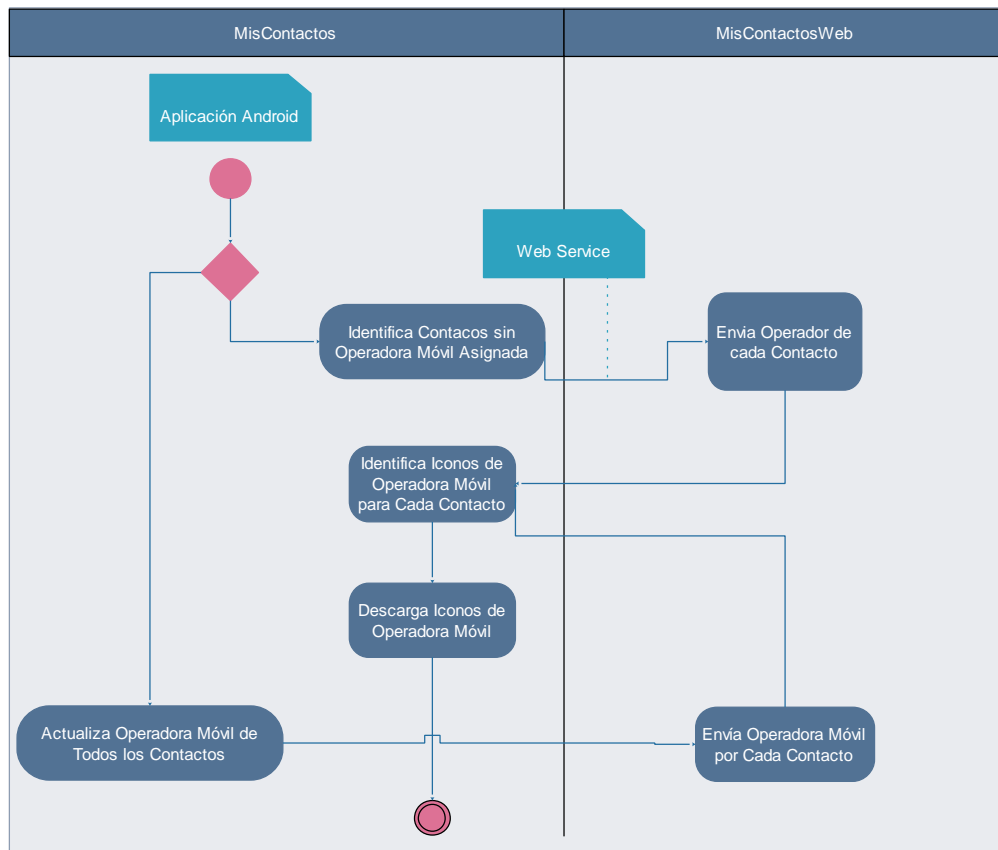


Figura 14. Diagrama UML Interacción Aplicación Móvil (MisContactos) – Aplicación Web (MiscontactosWeb). [25]

6.4. Análisis de propuesta

6.4.1. Factibilidad

6.4.1.1. Análisis de mercado

Según datos oficiales de La Agencia de Regulación y Control de las Telecomunicaciones (ARCOTEL), publicadas en una infografía con datos sobre los abonados del Servicio Móvil Avanzado – SMA y de acceso a internet, correspondiente a junio del presente año:

- 46,4% de usuarios del Servicio Móvil Avanzado poseen un Smartphone.
- En Ecuador existen 8,1 millones de cuentas de internet móvil.
- La tenencia de internet móvil ha crecido desde 2010 en 46 puntos: de 2,4% a 48,7%.
- 11 de cada 100 niños, de entre 5 y 15 años, posee un teléfono móvil activo, y de ellos 6 usan un Smartphone.

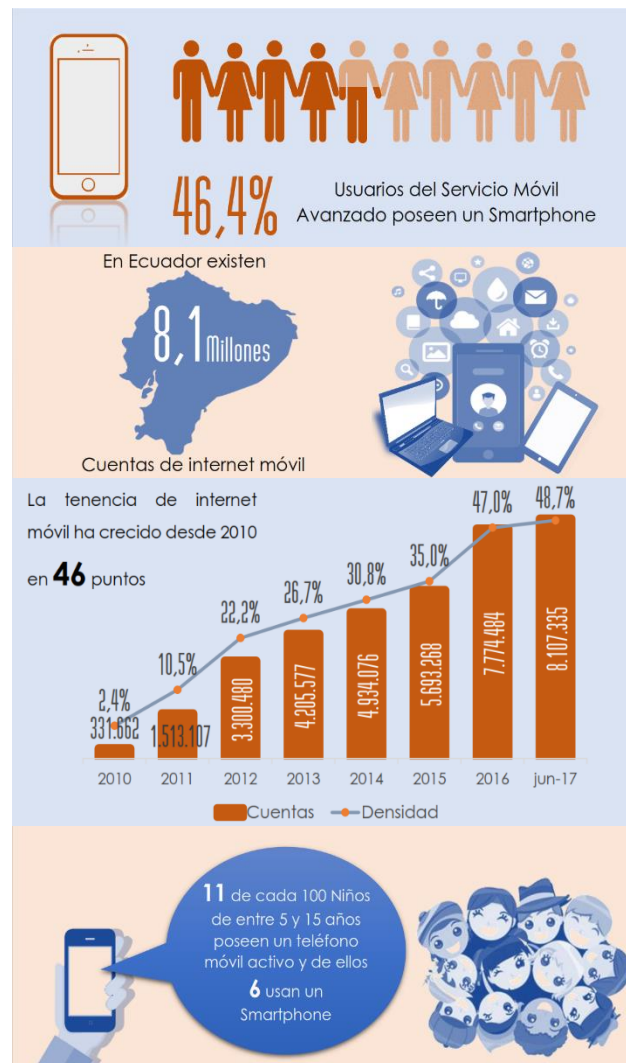


Figura 15. Infografía servicio SMA Ecuador - junio 2017. [18]

A la fecha el número de líneas activas sobrepasa los 15.061.858 en el país, de los cuales la modalidad prepago abarca más de 2/3 del total, equivalente a 10.656.096 usuarios aproximadamente, razón por la cual se podría afirmar que ID Móvil podrá brindar sus servicios a alrededor de 7 millones de usuarios a nivel del país.

6.4.1.2. Análisis técnico

Cumpliendo con el esquema propuesto para la plataforma IDMóvil, se considera 4 aspectos:

- Plataforma de almacenamiento web.
- Desarrollo del sistema IDMóvil.
- Hardware para administración del sistema IDMóvil.
- Presupuesto.

Plataforma de almacenamiento Web

La plataforma deberá tener la capacidad de almacenar una base de datos con al menos 28 millones de filas con un mínimo de dos atributos por fila, que incluyen el número telefónico y la operadora a la que corresponde.

En la web existen múltiples plataformas que ofrecen servicios tanto de hosting, dominio y capacidad de almacenamiento a precios considerablemente bajos y alta fiabilidad.

Para el presente caso de estudio se considera la plataforma GoDaddy y su plan Ultimate que ofrece las siguientes prestaciones:

- Sitios web ilimitados.
- Almacenamiento ilimitado.
- Ancho de banda sin medición.
- Correo comercial de Microsoft Office 365 gratis: primer año.
- Bases de datos MySQL ilimitadas (1 GB cada una).
- Bases de datos MSSQL ilimitadas (200 MB cada una).
- Usuarios de FTP ilimitados.
- Dominio gratis con un plan anual.
- DNS Premium.
- Certificado SSL de un año para proteger los datos del cliente y aumentar el posicionamiento de búsqueda. (Se aplican restricciones).

- Hosting PHP5.
- Más de 50 aplicaciones gratis.
- Asistencia técnica de expertos en hosting las 24 horas, los 7 días de la semana: (480) 463-8300.
- Tareas programadas.
- Copia de seguridad y restauración de bases de datos.

También incluye:

- Windows® Server 2012 R2.
- Instalación con 1 clic de más de 50 aplicaciones (WordPress, Joomla, Drupal, etc.).
- 200 MB de almacenamiento en la base de datos MSSQL.
- Protección contra DDoS y monitoreo de seguridad las 24 horas del día, 7 días a la semana.
- Configuración con 1 clic para los dominios registrados con GoDaddy.
- Panel de control flexible y fácil de usar.

Desarrollo del sistema IDMóvil

Al tener una composición modular, el desarrollo del sistema IDMóvil, se puede llevar a cabo dentro de 160 horas de trabajo, considerando para ello un solo desarrollador, trabajando 8 horas diarias, por 4 semanas.

Hardware para administración del sistema IDMóvil

Este ítem es opcional, puesto que la administración del sistema se recomendaría hacerlo en las madrugadas u horas donde los usuarios no demanden en lo posible de los servicios de IDMóvil, pudiéndose administrar desde cualquier portátil o computadora disponible en ese momento.

Sin embargo, se recomienda en caso de adquisición considerar que el equipo tenga las siguientes características:

- Windows 10 Pro.
- 8GB de RAM / DDR4 SDRAM.
- Intel / I7-7700 / Core i7.
- 1TB de Almacenamiento.

Presupuesto

Unidad	Descripción	Cantidad	Valor Unitario	Valor Total
Unidad	Computador de Escritorio última generación – Gasto Único	1	1200,00	1200,00
Unidad	Plan Anual – Plataforma de almacenamiento web	1	96,00	96,00
Unidad	Equipo de desarrollo – 1 Persona	1	400,00	400,00
Unidad	Soporte Anual – Sistema de Identificación Móvil	1	200,00	200,00
Total			1896,00	1896,00

6.4.1.3. Viabilidad

IDMóvil pretende llegar a cada usuario de manera gratuita, por lo que no lucraría por el servicio prestado, ante esto se considera las siguientes variantes:

- Financiamiento total por parte de superintendencia de control y poder de mercado
- Agregar opciones de personalización bajo pago en IDMóvil – Opciones Premium.

Si considera que de 1000.000 usuarios que adquieran IDMóvil, al menos un 10% optaran por opciones de personalización bajo pago, con un costo no superior a los 0.40 de dólares, se obtendrían ganancias cercanas a los 40.000 dólares, cantidad suficiente para mantener la plataforma por al menos 2 años sin necesidad de pedir renovación en la suscripción de usuarios Premium.

Por otra parte, en caso de recibir financiamiento total de la superintendencia de control y poder de mercado, la inversión que la entidad será mínima en función del tiempo.

Independientemente del método de financiamiento por el que se opte, el desarrollo de IDMóvil es factible y sustentable en el tiempo.

6.5. Desarrollo aplicación móvil

6.5.1. Aplicación escritorio

Se considera la incorporación de 3 opciones básicas, que permitirá interactuar de manera directa con la aplicación web y por medio de esta con la aplicación móvil.



Figura 16. Interfaz Aplicación Escritorio. [25]

Cargar Teléfonos

Permite cargar una base de datos actualizada en formato **CSV** a la sección escritorio del gestor My SQL, donde se cambian y aumentan dinámicamente cada uno de los números modificados o ingresados por primera vez.

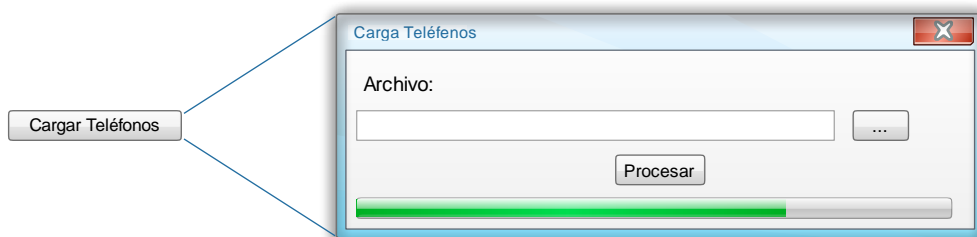


Figura 17. Interfaz Opción Cargar Teléfonos. [25]

Sincronizar

Realiza el envío de los números modificados e ingresados a la sección web del gestor My SQL, es en esta sección donde se realizan las peticiones por parte de la aplicación web.

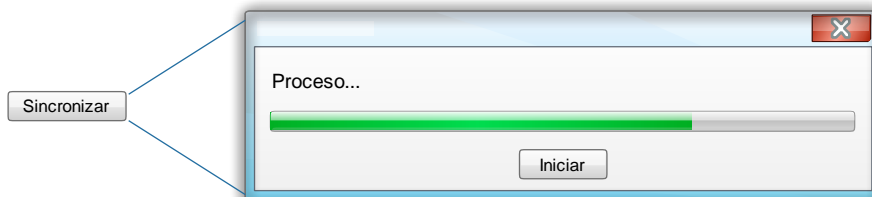


Figura 18. Interfaz Opción Sincronizar. [25]

Configurar Operadoras

Permite crear y eliminar un operador, así como modificar su logo y asignarle un nuevo código.

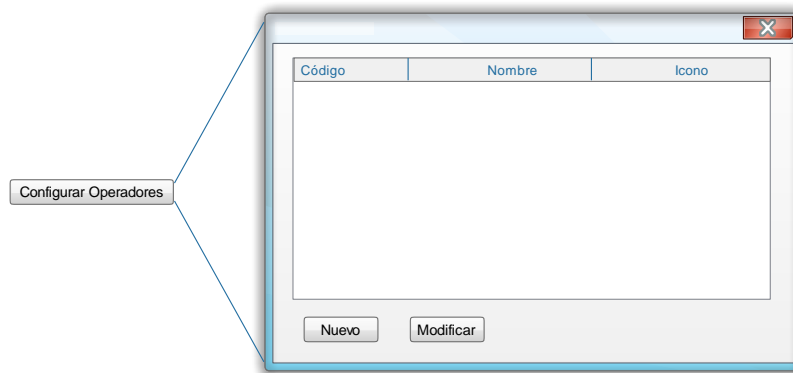


Figura 19. Interfaz Opción Configurar Operadoras. [25]

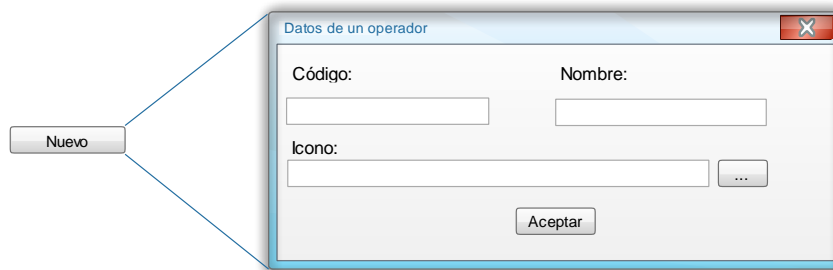


Figura 20. Interfaz Opción Configurar Operadoras - Nuevo. [25]

Estructura Lógica

- Librerías Utilizadas:



Figura 21. Estructura lógica librerías – Aplicación de Escritorio. [25]

Librería Makval que incluye los siguientes módulos:

- **makval.gimak.bo:**
 - I. **BusinessException.java:** Representa una excepción con lógica de negocio.
- **makval.gimak.security:**
 - I. **SessionManager:** Administra sesiones de administración.
- **makval.gimak.security.vo:**
 - I. **UserVO:** Contiene los datos de un usuario loggeado.
- **makval.gimak.utils:**
 - I. **AttributeInfo:** contiene información del atributo de una clase.
 - II. **FileUtils:** Servicios sobre archivos.
 - III. **ObjectUtils:** Brinda servicios sobre objetos.
 - IV. **SecurityUtils:** Servicios para seguridad y relacionados.
 - V. **StringUtils:** Clases utilitarias para manipulación de cadenas de texto.
 - VI. **ThreadUtils:** Hilos Utilitarios.
- **makval.gimak.web:**
 - I. **HttpUtils:** Utilitario HTTP.
- **makval.gimak.web.soap:**
 - I. **SimpleSoapClient:** Clase protocolo de WebServices.
 - II. **SoapClientGenerator:** Clase que genera clientes de WebServices.
 - III. **SoapClientProcessor:** Clase que se encarga de procesar los requests hechos a una interfaz y retransmitirlos vía el webservice.
 - IV. **SoapFunctionAttributes:** Indica si es una llamada anónima.
 - V. **SoapRequestEncoder:** Codifica un REQUEST Soap.
 - VI. **SoapSimpleRequestInfo:** Representa la información de un REQUEST hecho para un web service de GIMAK.
- **makval.gimak.web.soap.annotation:**
 - I. **SoapFunction:** Indica los nombres de los parámetros de una función SOAP.
 - II. **SoapService:** Define un WEB Service.
- **makval.gimak.web.xml:**
 - I. **XmlConversionException:** Representa una excepción generada al momento de convertir un XML en un objeto.
 - II. **XmlDeserializer:** Deserializa un objeto desde XML.

- III. **XmlEncoder:** Brinda servicios de codificación y decodificación de caracteres especiales de XML.
- IV. **XmlNodeInfo:** Contiene información sobre un nodo.
- V. **XmlParameter:** Representa el nombre de un parámetro XML.
- VI. **XmlReader:** Lee un archivo XML y genera un evento cada vez que se detecta la presencia de un nuevo nodo y la terminación de un nodo.
- VII. **XmlReaderEvtProcessor:** Representa el procesador de eventos encontrados por el lector de XML.
- VIII. **XmlReaderException:** Excepciones de lectura.
- IX. **XmlSerializer:** Serializa un objeto a XML.

- **Módulos de la aplicación:**

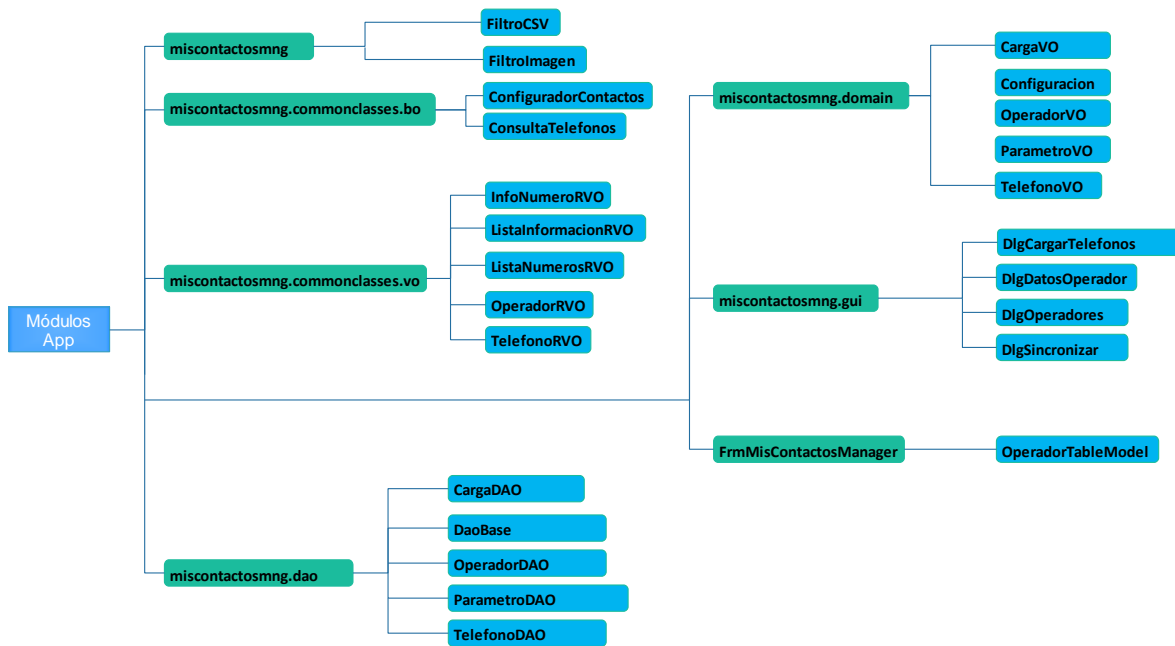
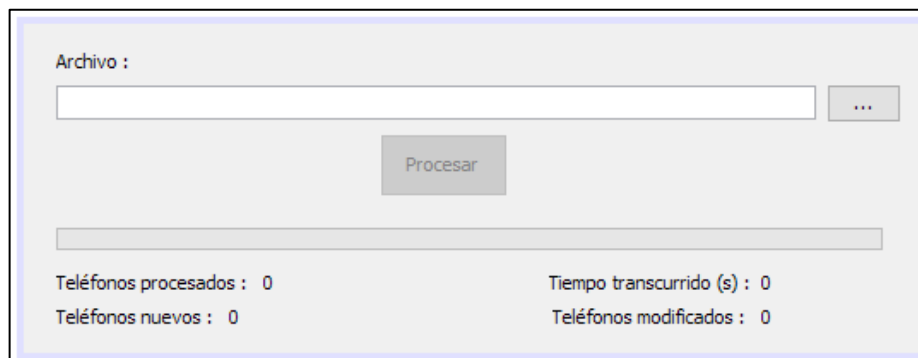


Figura 22. Estructura lógica módulos miscontactosmng – Aplicación de Escritorio. [25]

▪ **miscontactosmng:**

- I. **FiltroCSV.java:** Filtra Archivos CSV para la carga de una nueva base de datos.
- II. **FiltroImagen.java:** Filtra Archivos JPG, PNG, JPEG, para la carga de un nuevo Icono.

- **miscontactosmng.commonclasses.bo**
 - I. **ConfiguradorContactos:** Interface con los servicios para configurar los contactos.
 - II. **ConsultaTelefonos:** Brinda servicios sobre teléfonos.
- **miscontactosmng.commonclasses.vo**
 - I. **InfoNumeroRVO:** Información de un número.
 - II. **ListalInformacionRVO:** Lista con la información recibida.
 - III. **ListaNumerosRVO:** Lista de números cuya información se solicita.
 - IV. **OperadorRVO:** Información de un operador.
 - V. **TelefonoRVO:** Datos de un teléfono.
- **miscontactosmng.dao**
 - I. **CargaDAO:** Clase que brinda servicios de datos sobre cargas de teléfonos.
 - II. **DaoBase:** Conexión de base de datos con la que trabaja.
 - III. **OperadorDAO:** Brinda servicios de datos para operadores.
 - IV. **ParametroDAO:** Tabla que brinda servicios de datos para parámetros.
 - V. **TelefonoDAO:** Brinda servicios de datos para teléfonos.
- **miscontactosmng.domain**
 - I. **CargaVO:** Representa una carga de todos los teléfonos, se emplea como una especie de bitácora.
 - II. **Configuracion:** Configuración de la aplicación.
 - III. **OperadorVO:** Datos de un operador.
 - IV. **ParametroVO:** Almacena un parámetro de la base de datos.
 - V. **TelefonoVO:** Representa un registro en la tabla de teléfonos.
- **miscontactosmng.gui**
 - I. **DlgCargarTelefonos:**



II. DlgDatosOperador

Código : Nombre :

Icono : ...

Aceptar

III. DlgOperadores

Codigo	Nombre	Icono

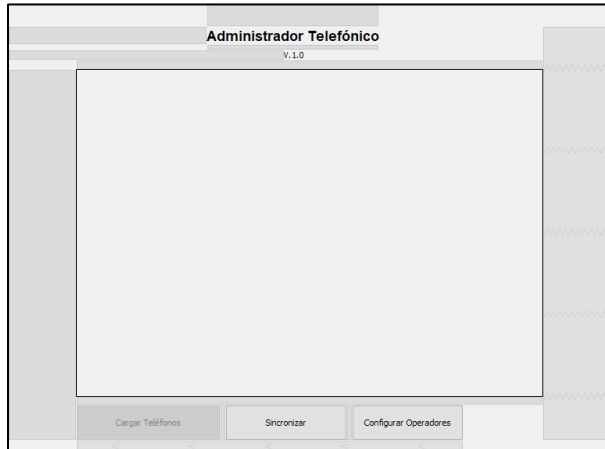
Nuevo Modificar

IV. DlgSincronizar

Telefonos transferidos : 0 de 0

Iniciar

VI. FrmMisContactosManager



VII. **OperadorTableModel:** Modelo de datos para visualizar operadores en la tabla de operadores.

- **miscontactosmng.resource.images**

I. Fondo:



6.5.2. Aplicación web

Interfaz

Contiene las configuraciones de la base de datos alojada en el servidor, además hace las veces de interfaz entre la APP móvil y la aplicación de escritorio, trabaja en segundo plano, transparente tanto para el usuario administrador como para el usuario final por lo que no cuenta con interfaz gráfica.

- cargardatos.php

```
<?php
include_once '../proyConfig.php';
/*
 * Endpoint del servicio de carga de datos
 */
// Crea el objeto implementador del servicio
$serviceImplementation=new \miscontactos\service\soap\ConfiguradorContactosSOAP();
// Crea el objeto que valida la seguridad
// $securityValidator=new \gsad\commonclasses\bo\GsadRemoteSecurityValidator();
// $securityValidator->setAnonymousRequests(true);
// Crea y configura el servicio
$server=new \gimak\web\soap\SoapServerGenerator();
// $server->setSecurityValidator($securityValidator);
// El nombre del servicio se llama MisContactos
// El namespace se llama miscontactos
// El objeto implementador es $serviceImplementation
// El URL del WSDL se calcula concatenando el URL base de la aplicacion mas "service/usuario.php"
// se espera que exista una carpeta llamada service
$server->setService("MisContactos", "miscontactos", $serviceImplementation, "service/cargardatos.php");
// GimakLogger::debug("Procesando mensaje SOAP");
// Atiende los requests
try
{
    $server->processRequest();
}
catch ( \Exception $e )
{
    \GimakLogger::error("Error ",$e);
}
// GimakLogger::debug("Fin de proceso de mensaje SOAP");
// Indica que es la ultima llamada de un script
// \gimak\security\SessionManager::exitScript();
?>
```

- consultar.php

```
<?php
include_once '../proyConfig.php';
\GimakLogger::debug("Inicio de consultar.php");
/*
 * Endpoint del servicio de usuarios
 */
// Crea el objeto implementador del servicio
$serviceImplementation=new \miscontactos\service\soap\ConsultaTelefonosSOAP();
// Crea el objeto que valida la seguridad
// $securityValidator=new \gsad\commonclasses\bo\GsadRemoteSecurityValidator();
// $securityValidator->setAnonymousRequests(true);
// Crea y configura el servicio
$server=new \gimak\web\soap\SoapServerGenerator();
// $server->setSecurityValidator($securityValidator);
// El nombre del servicio es MisContactos
// El namespace se llama miscontactos
// El objeto implementador es $serviceImplementation
// El URL del WSDL se calcula concatenando el URL base de la aplicacion mas "service/consultar.php"
// se espera que exista una carpeta llamada service
$server->setService("MisContactos", "miscontactos", $serviceImplementation, "service/consultar.php");
// GimakLogger::debug("Procesando mensaje SOAP");
// Atiende los requests
try
{
    $server->processRequest();
}
catch ( \Exception $e )
{
    \GimakLogger::error("Error ",$e);
}
// GimakLogger::debug("Fin de proceso de mensaje SOAP");
// Indica que es la ultima llamada de un script
// \gimak\security\SessionManager::exitScript();
?>
```

Estructura lógica

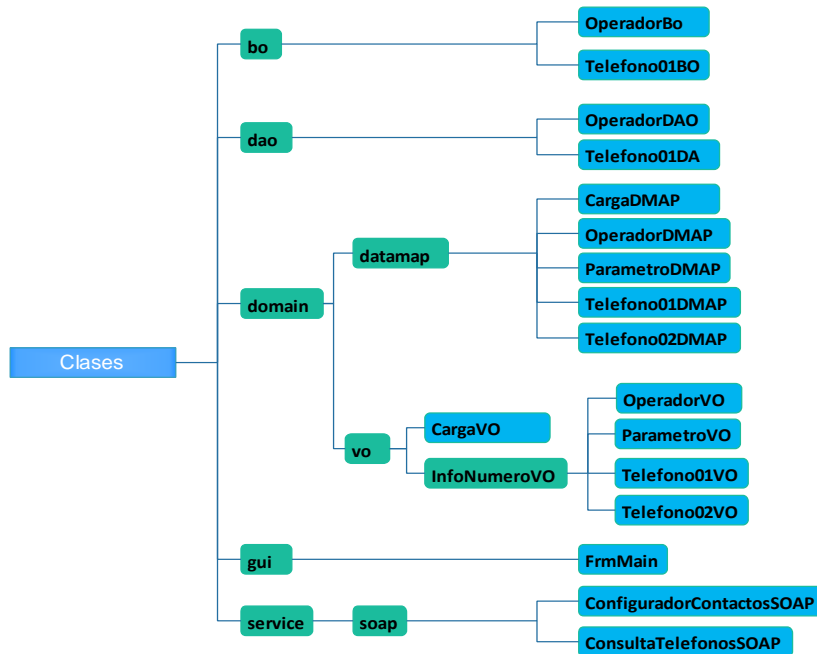


Figura 23. Estructura lógica clases - Aplicación Web. [25]

- Classes

▪ bo

- I. **OperadorBo.php**: Clase que brinda servicios de acceso a datos
- II. **Telefono01BO.php**: Clase que brinda servicios de acceso a datos

▪ dao

- I. **OperadorDAO.php**: Clase que brinda servicios de acceso a datos
- II. **Telefono01DAO.php**: Clase que brinda servicios de acceso a datos

▪ domain

- datamap

- I. **CargaDMAP.php**: Registro de las cargas hechas
- II. **OperadorDMAP.php**: Tabla con los datos de los operadores
- III. **ParametroDMAP.php**: Contiene parámetros de la aplicación
- IV. **Telefono01DMAP.php**: Tabla con la base de datos de los teléfonos
- V. **Telefono02DMAP.php**: Tabla con la base de datos de los teléfonos alterna

- vo

- I. **CargaVO.php**: Registro de las cargas hechas
- InfoNumeroVO.php**: Información de un número

- I. **OperadorVO.php**: Tabla con los datos de los operadores
 - II. **ParametroVO.php**: Contiene parámetros de la aplicación
 - III. **Telefono01VO.php**: Tabla con la base datos de los teléfonos
 - IV. **Telefono02VO.php**: Tabla con la base de datos de los teléfonos alterna
- **gui**
 - I. **FrmMain.php**: Panel principal
 - **Service**
 - **soap**
 - I. **ConfiguradorContactosSOAP.php**: Implementación del servicio WEB que actualiza la información de teléfonos y operadoras
 - II. **ConsultaTelefonosSOAP.php**: Implementación del servicio que retorna información sobre un numero
 - **Commonclasses**

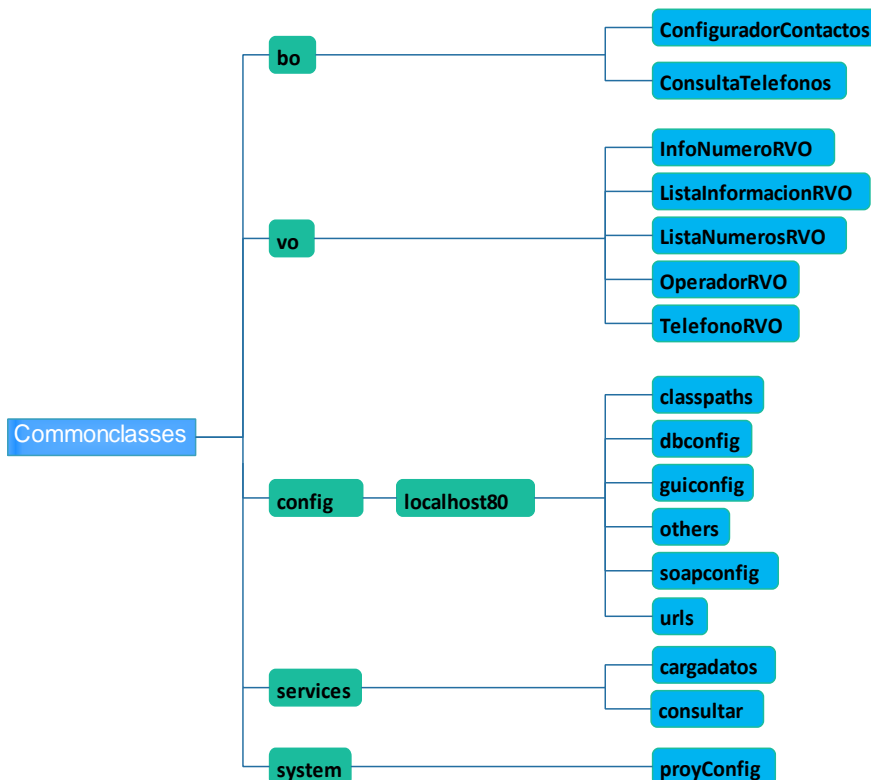


Figura 24. Estructura lógica commonclasses - Aplicación Web. [25]

- **bo**
 - I. **ConfiguradorContactos.php**: Descripción de Configurador Contactos
 - II. **ConsultaTelefonos.php**: Brinda servicios sobre teléfonos
- **vo**
 - I. **InfoNumeroRVO.php**: Información de un numero
 - II. **ListainformacionRVO.php**: Descripción de Lista InformacionRVO
 - III. **ListaNumerosRVO.php**: Lista de números que se consulta
 - IV. **OperadorRVO.php**: Tabla con los datos de los operadores
 - V. **TelefonoRVO.php**: Datos de un teléfono
- **Config**
 - **localhost80**
 - I. **classpaths.php**: Configura las rutas de clases de la aplicación
Configura las rutas para acceder a las clases de gsad
Configura las rutas para acceder a las clases de PHPMailer
 - II. **dbconfig.php**: Configura datasources
Configura la visualización de las consultas SQL
Configura el nombre del datasource por omisión aplicación
Configura el datasource por omisión que trabaja con MySql
 - III. **guiconfig.php**: Se configuran parámetros relacionados con la interface visual GUI
 - IV. **others.php**: Establece la autenticación de la aplicación
 - V. **soapconfig.php**: Configuración de servicios soap
 - VI. **urls.php**: Configuración de URLs
- **services**
 - I. **cargadatos.php**: Punto final del servicio de carga de datos
 - II. **consultar.php**: Punto del servicio de usuarios
- **system**
 - I. **proyConfig.php**: Configura las rutas de la aplicación

6.5.3. Base de datos

Está bajo el sistema de gestión de bases de datos relacional My Sql, aquí se gestionan los de datos suministrada por la aplicación de escritorio, la misma que a su vez es comparada con la base de datos web para encontrar posibles actualizaciones de cambio de operador móvil.

Base de datos Web

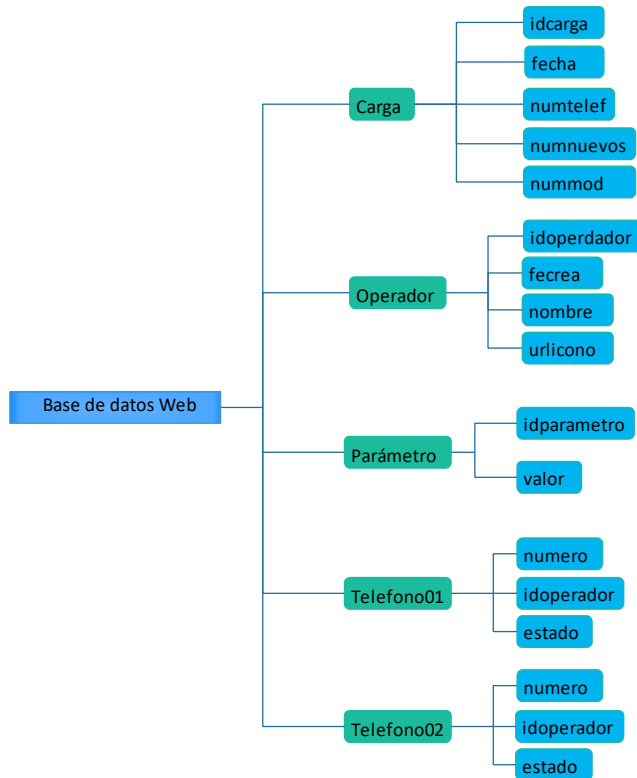


Figura 25. Estructura lógica base de datos Web. [25]

Conformada por 5 tablas que alojan información referente al operador móvil que corresponde un número telefónico en específico, actualizaciones, registro de carga de nuevos datos.

▪ Estructura General

Está compuesta por la tabla carga, operador, parámetro, telefono01, telefono02 (véase fig.25)

- I. **Carga:** Lleva el historial de cargas realizadas a la base de datos web.
- II. **Operador:** Lleva el registro, modificación y eliminación de un operador móvil.

- III. **Parámetro:** Lleva el registro de la base de datos actual (Telefono01 o Telefono02), que debe enviarse a la base de datos web.
- IV. **Telefono01:** Lleva el registro de los números de teléfonos registrados, alterna con la tabla Telefono02 para él envió a la base de datos web.
- V. **Telefono02:** Lleva el registro de los números de teléfonos registrados, alterna con la tabla Telefono01 para él envió a la base de datos web.

Tabla	Acción	Filas	Tipo	Cotejamiento	Tamaño
carga	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	utf8_unicode_ci	16 KB
operador	Examinar Estructura Buscar Insertar Vaciar Eliminar	4	InnoDB	utf8_unicode_ci	16 KB
parametro	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	utf8_unicode_ci	16 KB
telefono01	Examinar Estructura Buscar Insertar Vaciar Eliminar	11	InnoDB	utf8_unicode_ci	48 KB
telefono02	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	utf8_unicode_ci	48 KB
5 tablas		Número de filas		15	InnoDB latin1_swedish_ci 144 KB

Figura 26. Estructura General Base de Datos Web. [25]

▪ **Estructura secundaria carga**

Está compuesta por la tabla idcarga, fecha, numtelefon, numnuevos, nummod (véase fig.26)

- I. **idcarga:** Lleva el registro del código de la carga efectuado, comenzando en 1.
- II. **fecha:** Lleva el registro de la fecha, en que se efectuó la carga.
- III. **numtelefon:** Lleva el registro de los números de teléfonos procesados.
- IV. **numnuevos:** Lleva el registro de la cantidad de teléfonos nuevos.
- V. **nummod:** Lleva el registro de la cantidad de teléfonos modificados.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios
1	idcarga	int(11)			No	Ninguna	Codigo de la carga
2	fecha	datetime			No	Ninguna	Fecha de la carga
3	numtelefon	int(11)			Si	NULL	Telefonos procesados
4	numnuevos	int(11)			Si	NULL	Cantidad de teléfonos nuevos
5	nummod	varchar(45)	utf8_unicode_ci		Si	NULL	Cantidad de teléfonos modificados

Figura 27. Estructura Secundaria Carga Base de Datos Web. [25]

▪ **Estructura secundaria operador**

Está compuesta por la tabla idoperador, feccrea, nombre, urlicono (véase fig.27)

- I. **idoperador:** Lleva el registro del código de la carga del operador efectuado, comenzando en 1.
- II. **feccrea:** Lleva el registro de la fecha, en que se creó el operador.

- III. **nombre:** Lleva el registro del nombre del operador.
- IV. **urlicono:** Lleva el registro del nombre del icono del operador.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios
1	<u>idoperador</u>	varchar(40)	utf8_unicode_ci		No	Ninguna	Codigo del operador
2	<u>feccrea</u>	datetime			No	Ninguna	Fecha en la que se creo el operador
3	<u>nombre</u>	varchar(45)	utf8_unicode_ci		No	Ninguna	Nombre del operador
4	<u>urlicono</u>	varchar(45)	utf8_unicode_ci		Si	NULL	Nombre del icono del operador

Figura 28. Estructura Secundaria Operador Base de Datos Web. [25]

- **Estructura secundaria parámetro**

Está compuesta por la tabla idparametro, valor (véase fig.28)

- I. **idparametro:** Lleva el registro del parámetro.
- II. **valor:** determina la tabla que tiene la información actual de toda la base de datos.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra
1	<u>idparametro</u>	varchar(40)	utf8_unicode_ci		No	Ninguna	Codigo del parametro	
2	<u>valor</u>	varchar(400)	utf8_unicode_ci		Si	NULL	Valor del parametro	

Figura 29. Estructura Secundaria Parámetro Base de Datos Web. [25]

- **Estructura secundaria telefono01**

Está compuesta por la tabla número, idoperador, estado (véase fig.29)

- I. **numero:** Lleva el registro del número de teléfono.
- II. **idoperador:** contiene la data del operador al que pertenece el número (ítem anterior).
- III. **estado:** define el estado de un numero ingresado, el cual puede variar entre R(registrado), N (Nuevo número), M (número modificado) y S (sin cambios).

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios
1	<u>numero</u>	varchar(16)	utf8_unicode_ci		No	Ninguna	Numero del telefono
2	<u>idoperador</u>	varchar(40)	utf8_unicode_ci		No	Ninguna	Codigo del operador al que pertenece
3	<u>estado</u>	char(1)	utf8_unicode_ci		No	Ninguna	Estado del registro: R : registrado, esta pendiente de validar N : Nuevo telefono M : Modificado S : Sin cambios

Figura 30. Estructura Secundaria Telefono01 Base de Datos Web. [25]

- **Estructura secundaria telefono02**

Está compuesta por la tabla número, idoperador, estado (véase fig.30)

- I. **numero:** Lleva el registro del número de teléfono.

- II. **idoperador**: contiene la data del operador al que pertenece el número (ítem anterior).
- III. **estado**: define el estado de un numero ingresado, el cual puede variar entre R(registrado), N (Nuevo número), M (número modificado) y S (sin cambios).

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios
1	<u>numero</u>	varchar(16)	utf8_unicode_ci		No	Ninguna	Numero del telefono
2	<u>idoperador</u>	varchar(40)	utf8_unicode_ci		No	Ninguna	Código del operador al que pertenece
3	<u>estado</u>	char(1)	utf8_unicode_ci		No	Ninguna	Estado del registro: R : registrado, esta pendiente de validar N : Nuevo teléfono M : Modificado S : Sin cambios

Figura 31. Estructura Secundaria Telefono02 Base de Datos Web. [25]

Base de datos Escritorio

Está compuesta por todos los datos que serán migrados o transferidos a la base de datos web por lo que cuenta es una réplica de la estructura general de esta, formada por 5 tablas que alojan información referente al operador móvil que corresponde un número telefónico en específico, actualizaciones, registro de carga de nuevos datos.

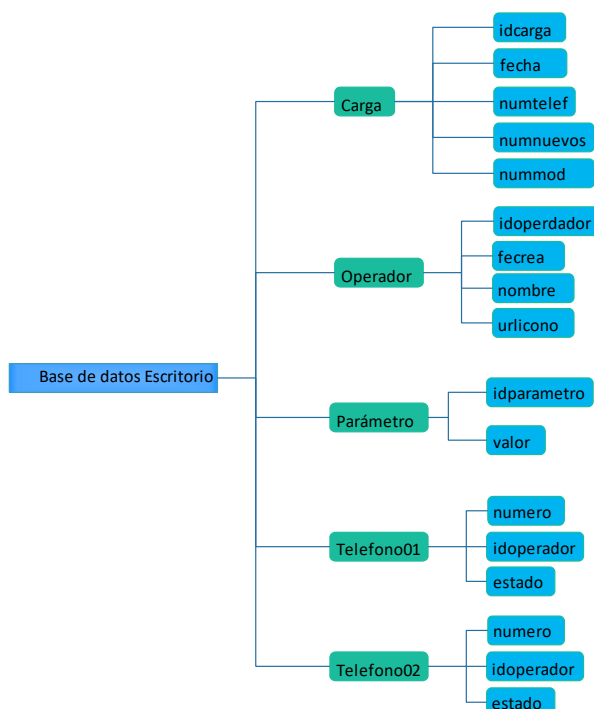


Figura 32. Estructura lógica base de datos escritorio. [25]

- **Estructura General**

Está compuesta por la tabla carga, operador, parámetro, telefono01, telefono02 (véase fig.32)

- I. **Carga:** Lleva el historial de cargas realizadas a la base de datos de escritorio.
- II. **Operador:** Lleva el registro, modificación y eliminación de un operador móvil.
- III. **Parámetro:** Lleva el registro de la base de datos actual (Telefono01 o Telefono02), que debe enviarse a la base de datos web.
- IV. **Telefono01:** Lleva el registro de los números de teléfonos registrados, alterna con la tabla Telefono02 para él envío a la base de datos web.
- V. **Telefono02:** Lleva el registro de los números de teléfonos registrados, alterna con la tabla Telefono01 para él envío a la base de datos web.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios
1	<u>idcarga</u>	int(11)			No	Ninguna	Codigo de la carga
2	<u>fecha</u>	datetime			No	Ninguna	Fecha de la carga
3	<u>numtelefon</u>	int(11)			Si	NULL	Telefonos procesados
4	<u>numnuevos</u>	int(11)			Si	NULL	Cantidad de teléfonos nuevos
5	<u>nummod</u>	varchar(45)	utf8_unicode_ci		Si	NULL	Cantidad de teléfonos modificados

Figura 33. Estructura General Base de Datos Escritorio. [25]

- **Estructura secundaria carga**

Está compuesta por la tabla idcarga, fecha, numtelefon, numnuevos, nummod (véase fig.33)

- I. **idcarga:** Lleva el registro del código de la carga efectuado, comenzando en 1.
- II. **fecha:** Lleva el registro de la fecha, en que se efectuó la carga.
- III. **numtelefon:** Lleva el registro de los números de teléfonos procesados.
- IV. **numnuevos:** Lleva el registro de la cantidad de teléfonos nuevos.
- V. **nummod:** Lleva el registro de la cantidad de teléfonos modificados.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra
1	<u>idcarga</u>	int(11)			No	Ninguna	Codigo de la carga	AUTO_INCREMENT
2	<u>fecha</u>	datetime			No	Ninguna	Fecha de la carga	
3	<u>numtelefon</u>	int(11)			Si	NULL	Telefonos procesados	
4	<u>numnuevos</u>	int(11)			Si	NULL	Cantidad de teléfonos nuevos	
5	<u>nummod</u>	varchar(45)	utf8_unicode_ci		Si	NULL	Cantidad de teléfonos modificados	

Figura 34. Estructura Secundaria Carga Base de Datos Escritorio. [25]

- **Estructura secundaria operador**

Está compuesta por la tabla idoperador, feccrea, nombre, urlicono (véase fig.34)

- I. **idoperador:** Lleva el registro del código de la carga del operador efectuado, comenzando en 1.
- II. **feccrea:** Lleva el registro de la fecha, en que se creó el operador.
- III. **nombre:** Lleva el registro del nombre del operador.
- IV. **urlicono:** Lleva el registro del nombre del icono del operador.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	<u>idoperador</u>	varchar(40)	utf8_unicode_ci		No	Ninguna	Codigo del operador		Cambiar
2	<u>feccrea</u>	datetime			No	Ninguna	Fecha en la que se creo el operador		Cambiar
3	<u>nombre</u>	varchar(45)	utf8_unicode_ci		No	Ninguna	Nombre del operador		Cambiar
4	<u>urlicono</u>	varchar(45)	utf8_unicode_ci		Si	NULL	Nombre del icono del operador		Cambiar

Figura 35. Estructura Secundaria Operador Base de Datos Escritorio. [25]

- **Estructura secundaria parámetro**

Está compuesta por la tabla idparametro, valor (véase fig.35)

- I. **idparametro:** Lleva el registro del parámetro.
- II. **valor:** determina la tabla que tiene la información actual de toda la base de datos.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	<u>idparametro</u>	varchar(40)	utf8_unicode_ci		No	Ninguna	Codigo del parametro		Cambiar
2	<u>valor</u>	varchar(400)	utf8_unicode_ci		Si	NULL	Valor del parametro		Cambiar

Figura 36. Estructura Parámetro Operador Base de Datos Escritorio. [25]

- **Estructura secundaria telefono01**

Está compuesta por la tabla número, idoperador, estado (véase fig.36)

- I. **numero:** Lleva el registro del número de teléfono.
- II. **idoperador:** contiene la data del operador al que pertenece el número (ítem anterior).
- III. **estado:** define el estado de un numero ingresado, el cual puede variar entre R(registrado), N (Nuevo número), M (número modificado) y S (sin cambios).

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios
1	<u>numero</u>	varchar(16)	utf8_unicode_ci		No	Ninguna	Numero del telefono
2	<u>idoperador</u>	varchar(40)	utf8_unicode_ci		No	Ninguna	Codigo del operador al que pertenece
3	<u>estado</u>	char(1)	utf8_unicode_ci		No	Ninguna	Estado del registro: R : registrado, esta pendiente de validar N : Nuevo telefono M : Modificado S : Sin cambios O : Modificado sincronizado C : Nuevo sincronizado

Figura 37. Estructura Teléfono01 Operador Base de Datos Escritorio. [25]

▪ Estructura secundaria telefono02

Está compuesta por la tabla número, idoperador, estado (véase fig.37)

- I. **numero:** Lleva el registro del número de teléfono.
- II. **idoperador:** contiene la data del operador al que pertenece el número (ítem anterior).
- III. **estado:** define el estado de un numero ingresado, el cual puede variar entre R(registrado), N (Nuevo número), M (número modificado) y S (sin cambios).

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios
1	<u>numero</u>	varchar(16)	utf8_unicode_ci		No	Ninguna	Numero del telefono
2	<u>idoperador</u>	varchar(40)	utf8_unicode_ci		No	Ninguna	Codigo del operador al que pertenece
3	<u>estado</u>	char(1)	utf8_unicode_ci		No	Ninguna	Estado del registro: R : registrado, esta pendiente de validar N : Nuevo telefono M : Modificado S : Sin cambios

Figura 38. Estructura Teléfono02 Operador Base de Datos Escritorio. [25]

6.5.4. Aplicación móvil

Interfaz

Formada por interfaz interactiva de 4 cuatro módulos, tal como se lo mostro en la propuesta, comienza por la superior izquierda con la opción llamar, seguido por registro de llamadas, agenda de contactos, favoritos y sincronizar.



- Código Fuente

```
@Override
protected void onCreate(@Nullable Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    ListView listView;
    EditText editText;
    setContentView(R.layout.activity_lst_contactos);
    opcionActual="agenda";
    adaptadorContacto=new AdaptadorContacto(this);
    listView=(ListView) findViewById(R.id.lstContactos);
    listView.setAdapter(adaptadorContacto);
    listView.setOnItemClickListener(new AdapterView.OnItemClickListener()
    {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int position,
long id)
        {
            visualizaContacto(adaptadorContacto.getItem(position));
        }
    });
    editText=(EditText) findViewById(R.id.txtBuscar);
    editText.addTextChangedListener(new TextWatcher()
    {
        @Override
        public void beforeTextChanged(CharSequence s, int start, int count, int
after){
        }
        @Override
        public void onTextChanged(CharSequence s, int start, int before, int count)
        {
        }
        @Override
        public void afterTextChanged(Editable s)
        { adaptadorContacto.filtroContactos();}
    });
    imgAgenda=(ImageView) findViewById(R.id.imgAgenda);
    imgAgenda.setOnClickListener(new View.OnClickListener()
    {
        @Override
        public void onClick(View v)
        {procesaAgenda();}
    });
    imgTelefono=(ImageView) findViewById(R.id.imgTelefono);
    imgTelefono.setOnClickListener(new View.OnClickListener()
    {
    @Override
    public void onClick(View v)
    {procesaTelefono();});
    imgHistorial=(ImageView) findViewById(R.id.imgHistorial);
    imgHistorial.setOnClickListener(new View.OnClickListener()
    {
        @Override
        public void onClick(View v)
        {procesaHistorial();}
    });
    imgFavoritos=(ImageView) findViewById(R.id.imgFavoritos);
    imgFavoritos.setOnClickListener(new View.OnClickListener()
    {
        @Override
        public void onClick(View v)
        {procesaFavoritos();}
    });
    imgSincronizar=(ImageView) findViewById(R.id.imgSincronizar);
    imgSincronizar.setOnClickListener(new View.OnClickListener()
    {
        @Override
        public void onClick(View v)
        {
            procesaSincronizar();
        }
    });
    });
}
```

Llamar

Redirección a la opción llamar de la aplicación nativa **“Teléfono”** (aplicación por defecto destinada a crear, modificar o contactos, además de realizar llamadas).



- Código Fuente

```
Llama a un contacto*/
private void llamarContacto( int position )
{
    Intent intent;
    ContactoVO contactoVO;
    contactoVO=getItem(position);
    intent = new Intent(Intent.ACTION_DIAL);
    intent.setData(Uri.parse("tel:" + contactoVO.getTelefonoPrincipal()));
    frmLstContactos.startActivity(intent);
}
/** Llama a un contacto*/
private void llamarHistorial( int position )
{
    Intent intent; HistorialVO historialVO;
    historialVO=(HistorialVO) getItem(position);
    intent = new Intent(Intent.ACTION_DIAL); intent.setData(Uri.parse("tel:" +
historialVO.getNumero()));
    frmLstContactos.startActivity(intent);
}
}
```

Historial

Carga el historial de llamadas registrado por la aplicación **“Teléfono”** (aplicación por defecto destinada a crear, modificar o contactos, además de realizar llamadas).



- Código Fuente

```
public class HistorialBO
{ /**
 * Carga la lista de contactos * @param resolver objeto con el que se accede a
 los logs * @param lstContactos lista de contactos actual* @return*/
    public List<HistorialVO> cargaHistorial( ContentResolver resolver,
List<ContactoVO> lstContactos)
    {
        ArrayList<HistorialVO> lst;
        Cursor managedCursor;HistorialVO historialVO;
        lst = new ArrayList<HistorialVO>();
        { // Carga los logs
            managedCursor = resolver.query(CallLog.Calls.CONTENT_URI, null, null,
null, null);
            while (managedCursor.moveToNext ())
            { historialVO=new HistorialVO();
historialVO.setNumero (managedCursor.getString (managedCursor.getColumnIndex (CallLog.
Calls.NUMBER));
historialVO.setTipo (managedCursor.getInt (managedCursor.getColumnIndex (CallLog.Calls
.TYPE));
            historialVO.setFecha (new
Date (managedCursor.getLong (managedCursor.getColumnIndex (CallLog.Calls.DATE))););
historialVO.setDuracion (managedCursor.getInt (managedCursor.getColumnIndex (CallLog.C
alls.DURATION)););
            lst.add (historialVO);}
        managedCursor.close();} return lst;}}
```

Agenda

Muestra en una pantalla diferente la lista de todos los contactos disponibles en la base de datos del Smart Phone.



- Código Fuente

```
private void visualizaCeldaContacto( View fila, final int posicion )
{
    TextView textView;
    ImageView imageView;
    ContactoVO contactoVO;
    String rutaIconoOperador;
    contactoVO=getItem(posicion);
    if ( contactoVO.isDetallesCalculados() == false )
    {
        ContentResolver resolver;
        ContactoBO contactoBO;
        resolver=frmLstContactos.getContentResolver();
        contactoBO=new ContactoBO();
        contactoBO.calculaEmail(resolver,contactoVO);
        contactoBO.calculaTelefonos(resolver,contactoVO);
        contactoBO.cargaIconoOperador(frmLstContactos,contactoVO);
        contactoVO.setDetallesCalculados(true);
    }
    textView=(TextView) fila.findViewById(R.id.lblNombreContacto);
    textView.setText(contactoVO.getNombre());

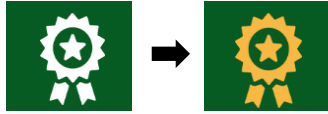
    imageView=(ImageView) fila.findViewById(R.id.imgEmail);
    imageView.setOnClickListener(new View.OnClickListener()
    {
        @Override
        public void onClick(View v)
        {
            procesaEnviaEmail(posicion);
        }
    });
    imageView=(ImageView) fila.findViewById(R.id.imgSms);
    imageView.setOnClickListener(new View.OnClickListener()
    {
        @Override
        public void onClick(View v)
        {
            procesaEnviaSms(posicion);
        }
    });
    if ( contactoVO.getPrimerTelefonoMovil() == null )
    {
        imageView=(ImageView) fila.findViewById(R.id.imgSms);
        imageView.setVisibility(View.GONE);
    }
    if ( contactoVO.isTieneEmail() == false )
    {
        imageView=(ImageView) fila.findViewById(R.id.imgEmail);
        imageView.setVisibility(View.GONE);
    }
    calculaFoto(contactoVO, fila, posicion);
    rutaIconoOperador= descargaIconoOperador(contactoVO);

    imageView=(ImageView) fila.findViewById(R.id.imgOperador);
    if ( rutaIconoOperador != null )
    {
        imageView.setImageURI(Uri.fromFile(new File(rutaIconoOperador)));
    }
    else
    {
        imageView.setVisibility(View.INVISIBLE);
    }
    textView=(TextView) fila.findViewById(R.id.lblFechaHistorial);
    textView.setVisibility(View.GONE);

    textView=(TextView) fila.findViewById(R.id.lblTipoHistorial);
    textView.setVisibility(View.GONE);
}
```


Favoritos

Permite al usuario realizar una selección de los contactos con los que tenga mayor concurrencia de llamadas.



- Código Fuente

```
/**
 * Agrega un favorito
 * @param contactoVO
 */
public void agregaContacto( ContactoVO contactoVO )
{
    lstOriginal.add(contactoVO);
}
```

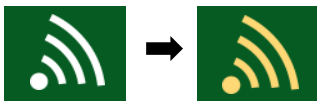
```
private void procesaFavoritos()
{
    imgFavoritos.setImageDrawable(getResources().getDrawable(R.drawable.favoritos_amari
llo));
    desmarcaOpciones(imgFavoritos);

    opcionActual="favoritos";

    adaptadorContacto.verFavoritos();
}
```

Sincronizar

Permite al usuario actualizar el icono del operador al que pertenece uno varios contactos que realizaron portabilidad numérica.



- Código Fuente

```
Procesa la opcion de sincronizacion
*/
private void procesaSincronizar()
{
    progressDialog = new ProgressDialog(this);
    progressDialog.setCancelable(false);
    progressDialog.setMessage((String)getText(R.string.msgSincronizando));
    progressDialog.show();
    new Thread(new Runnable()
    {
        @Override
        public void run()
        {
            ejecutaSincronizacion();
        }
    }).start();
}
```

```

/**
 * Ejecuta la sincronizacion
 */
private void ejecutaSincronizacion()
{
    List<ContactoVO> lst;
    MiContactosBd bd;
    OrmSession session;
    ContactoDAO contactoDAO;

    // Obtiene la lista de todos los contactos
    lst=adaptadorContacto.getLstOriginal();

    bd=MiContactosBd.getInstance(this);
    session=bd.createOrmSession();
    contactoDAO=new ContactoDAO(session);

    calculaIconoOperador(lst);

    // Bucle que actualiza el icono de los contactos
    for ( ContactoVO contacto : lst )
    {
        try
        {
            if (contactoDAO.getByKey(contacto) != null )
            {
                // El contacto ya existe solo se actualiza
                contactoDAO.save(contacto,false);
            }
            else
            {
                // El contacto es nuevo, se crea
                contactoDAO.save(contacto,true);
            }
        }
        catch ( OrmException e )
        {
            Log.e("AdaptadorToolBar","Error al guardar icono operado",e);
        }
    }

    adaptadorContacto.descargaIconosOperadores();

    session.close();
    bd.close();

    runOnUiThread(new Runnable()
    {
        @Override
        public void run()
        {
            progressDialog.hide();
            adaptadorContacto.verTodos();
        }
    });
}

```

```

/**
 * Inicializa la sesion para usar Webservicices
 */
private void iniciaSesion()
{
    UserVO usuario;

    usuario=new UserVO();

    SessionManager.initSession();
    SessionManager.applicationId="miscontactos";
    SessionManager.setAppSecurityCredentials("miscontactos", "123456");
    SessionManager.setCurrentUser(usuario);
}

```

```

/**
 * Metodo que calcula el icono del operador
 * @param lst lista de contactos a los que se les debe calcular el operador
 */
private void calculaIconoOperador( List<ContactoVO> lst )
{
    StringBuffer numeros=new StringBuffer("");
    String listaNumeros,num;
    ArrayList<Operadora>lstOp;
    ListaNumerosRVO lista;
    ArrayList<InfoNumeroRVO>lstInfo;
    ContentResolver resolver;
    ContactoBO contactoBO;
    resolver=getContentResolver();
    contactoBO=new ContactoBO();
    lista=new ListaNumerosRVO();
    // Calcula la lista de numeros separados por coma
    for( ContactoVO contacto: lst )
    {
        if ( contacto.isDetallesCalculados() == false )
        {
            contactoBO.calculaEmail(resolver,contacto);
            contactoBO.calculaTelefonos(resolver,contacto);
            contactoBO.cargaIconoOperador(this,contacto);

            contacto.setDetallesCalculados(true);
        }
        num=contacto.getTelefonoPrincipal();
        if ( num != null )
        {
            lista.getLstNumeros().add(contactoBO.limpiaNumero(num));
        }
    }

    // Llamar al web service
    try
    {
        lstInfo = getInfo(lista);
    }
    catch ( IOException e )
    {
        final String msgError=e.getMessage();
        runOnUiThread(new Runnable()
        {
            @Override
            public void run()
            {
                showToast("Error :"+msgError);
            }
        });
        return;
    }
    if ( lstInfo == null ) return;
    // Bucle ficticio que calcula el URL del operador de los contactos
    for( ContactoVO contacto: lst )
    {
        num=contacto.getTelefonoPrincipal();
        if ( num != null )
        {
            for (InfoNumeroRVO info : lstInfo)
            {
                if ( num.equals(info.getNumero()) )
                {
                    contacto.setUrlOperador(info.getUrlicono());
                }
            }
        }
    }
}

```

```

/**
 * Llama al web service para obtener informacion de numeros
 * @param lst lista de numeros
 * @return lista de informacion
 */
private ArrayList<InfoNumeroRVO>getInfo( ListaNumerosRVO lst ) throws IOException
{
    ConsultaTelefonos consultaTelefonos;
    String msg;

    iniciaSesion();

    msg="Error al crear el cliente web service";
    try
    {
        consultaTelefonos=(ConsultaTelefonos)
SoapClientGenerator.getInstance(ConsultaTelefonos.class,
        "http://192.168.1.4:8080/miscontactos/service/consultar.php?wsdl",
        "http://192.168.1.4:8080/miscontactos/service/consultar.php");

        msg="Error al obtener la informacion de los telefonos";

        return consultaTelefonos.getOperador(lst).getLstInfo();
    }
    catch( Exception e )
    {
        Log.e("FrmLstContactos",msg,e);
        throw new IOException(msg,e);
    }
}

```

```

private void procesaTelefono()
{
    Intent intent;
    ContactoVO contactoVO;

    intent = new Intent(Intent.ACTION_DIAL);
    intent.setData(Uri.parse("tel:"));

    startActivity(intent);

    //
    imgTelefono.setImageDrawable(frmLstContactos.getResources().getDrawable(R.drawable.
telefono_amarillo));
    desmarcaOpciones(imgTelefono);
}

```

Estructura lógica

- Librerías

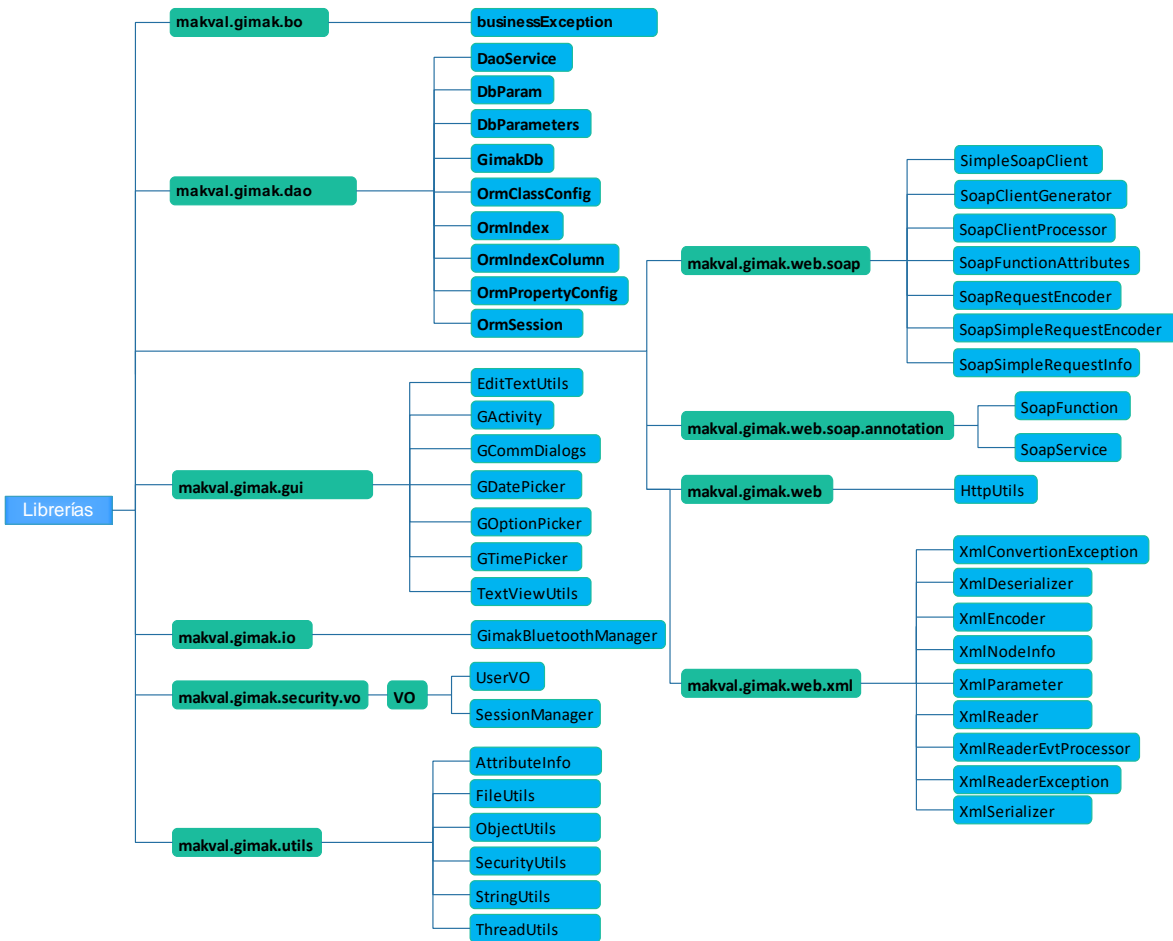


Figura 39. Estructura lógica librerías – Aplicación Móvil. [25]

- **makval.gimak.bo**

- I. **BusinessException:** Representa una excepción con lógica de negocio.

- **makval.gimak.dao**

- I. **DaoService:** Clase base para la creación de clases que brindan servicios de datos.

- II. **DbParam:** Clase que representa el parámetro que se envía a un query.

- III. **DbParameters:** Clase que representa una lista de parámetros.

- IV. **GimakDb:** Base de datos con la que trabaja.

- V. **OrmClassConfig:** Clase que representa la configuración.

- VI. **OrmIndex:** Nombre del índice.

- VII. **OrmIndexColumn:** Representa la columna de un índice.

- VIII. **OrmPropertyConfig:** Configuración de una propiedad de una clase.
- IX. **OrmSession:** Clase que representa una sesión en la Base de Datos.
- **makval.gimak.gui**
 - I. **EditTextUtils:** Utilitarios para trabajar con cajas de texto.
 - II. **GActivity:** Representa la clase base de las actividades de Gimak.
 - III. **GCommDialogs:** Clase que brinda servicios para la generación de pantallas comunes.
 - IV. **GDatePicker:** Representa un dialogo para seleccionar fechas desde un TextView.
 - V. **GOptionPicker:** Representa una pantalla de dialogo para seleccionar opciones.
 - VI. **GTimePicker:** Representa un dialogo para seleccionar una hora desde un TextView.
 - VII. **TextViewUtils:** Utilitarios para trabajar con cajas de texto.
- **makval.gimak.io**
 - I. **GimakBluetoothManager:** Servicios para el proceso de bluetooth
- **makval.gimak.security.vo**
- **VO**
 - I. **UserVO:** Contiene los datos de un usuario loggeado.
- **SessionManager:** Nombre del atributo de sesión que almacena al usuario actual.
- **makval.gimak.utils**
 - I. **AttributeInfo:** Contiene información del atributo de una clase.
 - II. **FileUtils:** Servicios sobre archivos.
 - III. **ObjectUtils:** Brinda servicios sobre objetos.
 - IV. **SecurityUtils:** Servicios para seguridad y relacionados.
 - V. **StringUtils:** Clases utilitarias para manipulación de cadenas de texto.
 - VI. **ThreadUtils:** Ejecuta el método sleep dentro de un try catch, silencia el try catch.
- **makval.gimak.web.soap**
 - I. **SimpleSoapClient:** URL del webservice.

- II. **SoapClientGenerator:** Clase que genera clientes de WebServices.
- III. **SoapClientProcessor:** Clase que se encarga de procesar los requests hechos a una interfaz y retransmitirlos vía el webservice.
- IV. **SoapFunctionAttributes:** Indica si es una llamada anónima.
- V. **SoapRequestEncoder:** Codifica un REQUEST Soap.
- VI. **SoapSimpleRequestEncoder:** Implementación básica del codificador de invocaciones SOAP.
- VII. **SoapSimpleRequestInfo:** Representa la información de un REQUEST hecho para un web service de GIMAK
- **makval.gimak.web.soap.annotation**
 - I. **SoapFunction:** Indica los nombres de los parámetros de una función SOAP.
 - II. **SoapService:** Define un WEB Service.
- **makval.gimak.web**
 - I. **HttpUtils:** Utilitario HTTP.
- **makval.gimak.web.xml**
 - I. **XmlConversionException:** Representa una excepción generada al momento de convertir un XML en un objeto.
 - II. **XmlDeserializer:** Deserializa un objeto desde XML.
 - III. **XmlEncoder:** Brinda servicios de codificación y decodificación de caracteres especiales de XML.
 - IV. **XmlNodeInfo:** Contiene información sobre un nodo.
 - V. **XmlParameter:** Representa el nombre de un parámetro XML.
 - VI. **XmlReader:** es un archivo XML y genera un evento cada vez que se detecta la presencia de un nuevo nodo y la terminación de un nodo.
 - VII. **XmlReaderEvtProcessor:** Representa el procesador de eventos encontrados por el lector de XML.
 - VIII. **XmlSerializer:** Indica si al generar el XML se debe generar el atributo "javaObjectClassName".

- **Módulos aplicación Móvil**

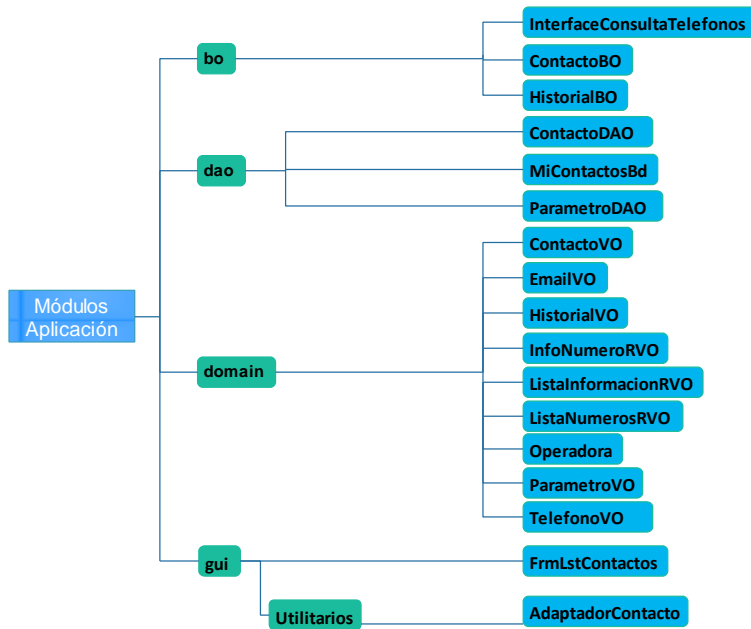


Figura 40. Estructura lógica Módulos – Aplicación Móvil. [25]

▪ **bo**

- I. **InterfaceConsultaTelefonos:** Retorna los datos del operador de un número telefónico.
- II. **ContactoBO:** Clase que brinda servicios de lógica de negocios de contactos.
- III. **HistorialBO:** Carga la lista de contactos.

▪ **dao**

- I. **ContactoDAO:** Servicios de datos para contactos.
- II. **MiContactosBd:** Representa la base de datos en la que se guarda información extra.
- III. **ParametroDAO:** Brinda servicio de datos para acceder a parámetros.

▪ **domain**

- I. **ContactoVO:** Contiene los datos de un contacto.
- II. **EmailVO:** Datos de un correo electrónico.
- III. **HistorialVO:** Datos del historial.

- IV. **InfoNumeroRVO:** información número de teléfono.
- V. **ListalInformacionRVO:** Lista con la información recibida.
- VI. **ListaNumerosRVO:** Lista de números.
- VII. **Operadora:** retorna información acerca del Operador
- VIII. **ParametroVO:** Parámetro simple que se almacena en la base de datos.
- IX. **TelefonoVO:** Representa un número telefónico.
- **gui**
 - I. **FrmLstContactos:** Pantalla que lista los contactos.
- Utilitarios
 - I. **AdaptadorContacto:** Genera la representación visual de un contacto.

7. RESULTADOS

7.1. Pruebas

7.1.1. Opción Llamar

Al seleccionar la opción llamar la aplicación responde de manera correcta conforme a lo establecido en el código fuente y propuesta realizada (ver figura 41).

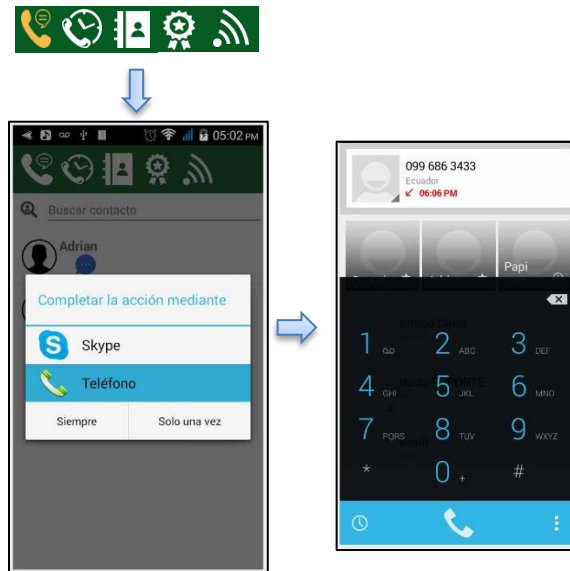


Figura 41. Funcionalidad opción Llamar aplicación móvil. [25]

7.1.2. Opción Historial

Al seleccionar la opción Historial la aplicación responde de manera correcta conforme a lo establecido en el código fuente y propuesta realizada (ver figura 42).



Figura 42. Funcionalidad opción historial aplicación móvil. [25]

7.1.3. Opción Agenda

Al seleccionar la opción Agenda la aplicación responde de manera correcta conforme a lo establecido en el código fuente y propuesta realizada (ver figura 43).

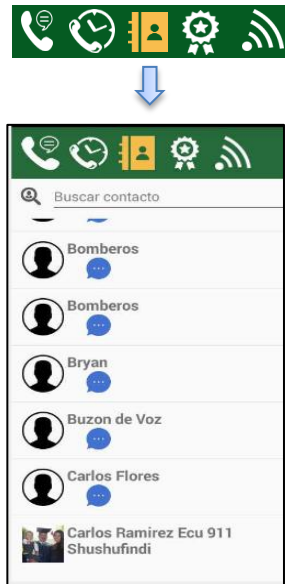


Figura 43. Funcionalidad opción agenda aplicación móvil. [25]

7.1.4. Opción Favoritos

Al seleccionar la opción Favoritos la aplicación responde de manera correcta conforme a lo establecido en el código fuente y propuesta realizada (ver figura 44).



Figura 44. Funcionalidad opción favoritos aplicación móvil. [25]

7.1.5. Opción Sincronizar

Al seleccionar la opción Sincronizar la aplicación responde de manera correcta conforme a lo establecido en el código fuente y propuesta realizada (ver figura 45).

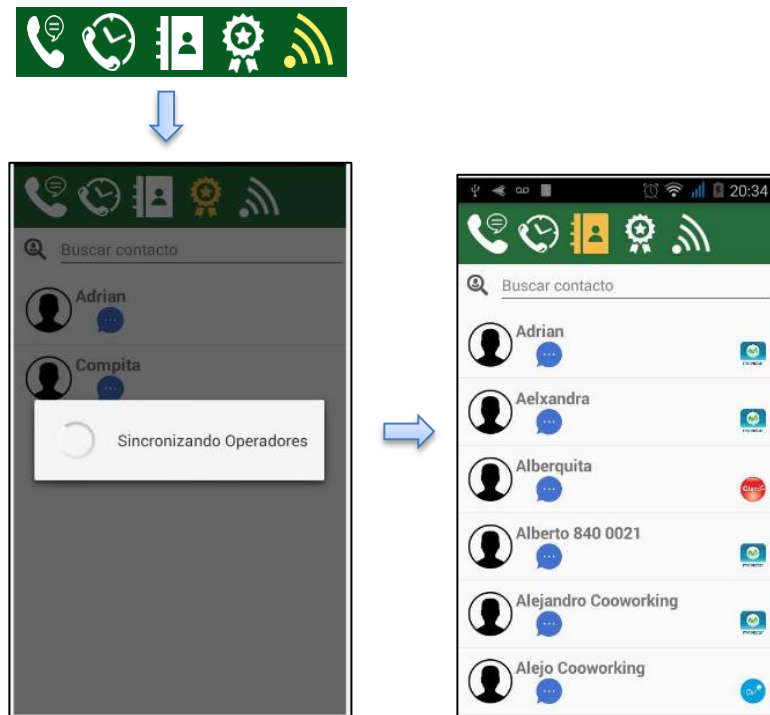


Figura 45. Funcionalidad opción sincronizar aplicación móvil. [25]

8. DISCUSIÓN

Para el desarrollo de IDmovil, aplicación móvil para dispositivos con OS Android orientada a la identificación de Operadoras Móviles, se propuso una arquitectura del tipo modular que consta de 3 componentes; 1. Aplicación Escritorio, 2. Aplicación Web, 3. Aplicación Móvil, capaz de optimizar recursos a nivel de procesamiento en el Smartphone, teniendo como referencia que el número de líneas activas al momento asciende a 15.061.858, líneas que tienen que ser comparadas una a una con una base de datos que contiene la información actual de los números telefónicos registrados con una operadora móvil en específico.

Planteándose para ello como objetivo general “Desarrollar una aplicación móvil para dispositivos con OS Android orientada a la identificación de Operadoras Móviles”, tomando como referencia 4 premisas, iniciando por “Desarrollar un algoritmo basado en SDK capaz de gestionar, procesar y comparar una base de datos web, con la agenda telefónica almacenada en el dispositivo con OS Android”, “Modificar por medio de la aplicación móvil la presentación de agenda contactos del dispositivo, identificando con una etiqueta de color rojo a la operadora móvil claro, verde a la operadora móvil movistar y azul a la operadora móvil cnt”, “Brindar al usuario una aplicación que le permita el ahorro de recursos económicos por concepto de conexión de voz inter e intra operadoras” y la última “establecer vínculos de colaboración y cooperación entre la Superintendencia de Control del Poder de Mercado y la Universidad Nacional de Loja”.

En base a los resultados obtenidos, la arquitectura propuesta tiene un gran desempeño en cuanto a tiempo de procesamiento y recursos destinados para esta acción, consiguiendo una apreciación positiva por parte del usuario, dando cumplimiento con esto a tres de las 4 premisas planteadas, puesto que se consiguió Desarrollar un algoritmo basado en SDK capaz de gestionar, procesar y comparar una base de datos web, con la agenda telefónica almacenada en el dispositivo con OS Android.

Aunque no fue posible modificar por medio de la aplicación móvil la presentación de agenda contactos del dispositivo, identificando con una etiqueta de color rojo a la operadora móvil claro, verde a la operadora móvil movistar y azul a la operadora móvil cnt, se optó por la creación de una agenda paralela co-dependiente de la nativa del dispositivo, a la que se le agregó una personalización con iconos que permiten identificar la operadora a la que pertenece un contacto en particular, entre las principales limitantes que se encontró para

cumplir con esta premisa, estuvo la arquitectura nativa del sistema Android y la seguridad implementada, que tiene como objetivo asegurarnos que no perderemos el control del dispositivo gracias a que no habrá aplicaciones que puedan acceder a la raíz del sistema, para ser más específicos era necesario que IDMóvil tenga permisos root, opción que por el momento no está disponible en los dispositivos que no han sido modificados por algún software externo.

También fue posible brindar al usuario una aplicación que le permita el ahorro de recursos económicos por concepto de conexión de voz inter e intra operadoras, la interfaz de IDMóvil muestra en tiempo real, por medio de iconos laterales, a que operadora pertenece un número en particular, incentivando de esta manera al ahorro consciente de tiempo aire, al momento IDMóvil solo está disponible para pruebas, su lanzamiento oficial a la plataforma de Google play dependería del presupuesto para adquirir un dominio y hosting de alojamiento web.

Sin embargo, no fue posible establecer vínculos de colaboración y cooperación entre la superintendencia de control del poder de mercado y la Universidad Nacional de Loja, donde la Universidad Nacional de Loja cedería los derechos de autoría por IDMóvil a la Superintendencia de Control del Poder de Mercado, y esta se comprometería a hacerse cargo del soporte y de facilitadora de las bases de datos nacional para la comparación de las líneas móviles, el conocimiento de las razones se manejó internamente dentro de las instituciones por lo que no son adjuntadas en la descripción.

Actualmente, existen páginas web que permiten realizar consultas del operador móvil al que pertenece un número telefónico en específico, los cuales presentan muchas limitantes a la hora de funcionalidad y aceptación, a diferencia de IDMóvil que ha más de ser portable está orientando específicamente para dispositivos móviles.

9. CONCLUSIONES

- Se implementó una aplicación externa a la agenda contactos por defecto del Smartphone, basada en el SDK nativo de Android, con características similares a lo propuesto en el planteamiento del proyecto, que tiene la capacidad de crear una copia dinámica de los contactos registrados y asignarle un icono identificativo por operadora a cada uno.
- La aplicación permite a los usuarios identificar a que operador móvil pertenece un contacto en específico, proporcionándole una opción a tomar en cuenta antes de realizar la llamada telefónica, promulgando así una cultura menos consumista que tienda al ahorro de recursos económicos de cada hogar.
- No fue posible establecer vínculos de colaboración y cooperación entre la Superintendencia de Control del Poder de Mercado y la Universidad Nacional de Loja, por cuestiones burocráticas ajenas al sector académico, determinando con ello el aplazamiento de la publicación oficial de IDMóvil en la plataforma Google Play.
- No es posible modificar la presentación de la aplicación agenda de contactos que viene por defecto en cada Smartphone, entre las limitantes encontradas más sobresalientes se tienen; las personalizaciones del sistema operativo Android característico de cada fabricante y asignación de permisos usuario root para la aplicación.

10. RECOMENDACIONES

- Para posibles mejoras, sería conveniente incrementar módulos de seguridad en el aplicativo móvil, esto para evitar ataques al servidor principal y perder el servicio de manera temporal o permanente.
- Los módulos de IDMóvil permiten también agregar cualquier tipo de operadora que incluya un número telefónico, por lo que la telefonía fija podría ser considerada en una versión posterior.
- Considerar la personalización y permisos del sistema operativo característico de cada fabricante antes de proponer el desarrollo de una aplicación móvil.
- Escoger entornos de desarrollo integrales de acuerdo a la solución que se desee brindar.
- Considerar utilizar arquitecturas de software preestablecidas y que hayan sido probadas por desarrolladores con experiencias.
- Evitar incluir dependencias de terceros al momento de proponer un proyecto de titulación.

11. BIBLIOGRAFÍA

- [1] R. R. J. M. M. P. Josep Prieto, Tecnología y desarrollo en dispositivos móviles, Barcelona : Eureka Media, SL, 2011.
- [2] Microsoft , «windowphone,» [En línea]. Available: <http://www.windowsphone.com/es-ec/how-to/wp8/basics/windows-phone-8-update-history>. [Último acceso: 26 04 2016].
- [3] Apple, «apple,» [En línea]. Available: <http://www.apple.com/es/ios/what-is/>. [Último acceso: 26 04 2016].
- [4] BlackBerry, «global.blackberry,» [En línea]. Available: <http://global.blackberry.com/es/software/smartphones/blackberry-10-os.html>. [Último acceso: 26 04 2016].
- [5] Android: Guía de desarrollo de aplicaciones para Smartphones y Tabletas, Barcelona: eni ediciones , 2014.
- [6] Universidad Carlos III de Madrid, «sites.google,» [En línea]. Available: <https://sites.google.com/site/swcuc3m/>. [Último acceso: 29 04 2016].
- [7] U. P. d. Valencia, «androidcurso,» [En línea]. Available: <http://www.androidcurso.com/index.php/99>. [Último acceso: 29 04 2016].
- [8] Android , «android,» [En línea]. Available: <https://www.android.com/history/#/donut>. [Último acceso: 29 04 2016].
- [9] Google, «developer.android,» [En línea]. Available: <https://developer.android.com/topic/libraries/support-library/features.html#v7-appcompat>. [Último acceso: 29 04 2016].
- [10] Google, «developer.android,» [En línea]. Available: <http://developer.android.com/intl/es/tools/support-library/index.html>. [Último acceso: 29 04 2016].
- [11] Google, «developer.android.,» [En línea]. Available: <https://developer.android.com/studio/releases/index.html?hl=es>. [Último acceso: 29 04 2016].
- [12] Google, «developer.android.com,» [En línea]. Available: https://developer.android.com/studio/intro/index.html#herramientas_de_depuracion_y_perfil. [Último acceso: 27 04 2017].

- [13] Geekytheory, «geekytheory,» [En línea]. Available: <https://geekytheory.com/que-es-el-android-ndk-parte-1/>. [Último acceso: 29 04 2016].
- [14] F. Minera, PHP + MySQL desde cero, Ciudad Autónoma de Buenos Aires: Fox Andina, 2014.
- [15] E. M. Talón, Apache, España: Aula Mentor, 2012.
- [16] C. Zapata, «MANTENIMIENTO DE UNA COMPUTADORA,» [En línea]. Available: <http://mantenimientosdeunapc.blogspot.com/2011/11/que-es-xampp-y-para-que-sirve.html>. [Último acceso: 02 05 2017].
- [17] C. Peñarrieta, Java y NetBeans, México: Anón , 2011.
- [18] A. D. R. Y. C. D. L. TELECOMUNICACIONES, «<http://www.arcotel.gob.ec>,» Junio 2017. [En línea]. Available: http://www.arcotel.gob.ec/wp-content/uploads/2017/05/Infografia4_2017-Internet-movil_v2.pdf. [Último acceso: 07 Noviembre 2017].
- [19] Android, «source.android,» [En línea]. Available: <https://source.android.com/devices/index.html>. [Último acceso: 29 04 2016].
- [20] Cedric Simon , Curso de introducción a las bases de datos y SQL: Manual del alumno, Nicaragua : ©Copyright Cédric Simon, 2008.
- [21] A. R. F. M. MARIA JESUS RAMOS, SISTEMAS GESTORES DE BASES DE DATOS, Madrid: McGraw-Hill, 2006.
- [22] M. d. T. y. d. I. S. d. I. Información, «Servicio Móvil Avanzado,» *Boletín Estadístico del Sector de Telecomunicaciones* , vol. 1, p. 23, 2014.
- [23] J. Arias, *Ilustración Interacción Cliente Servidor*, Loja: Anón, 2017.
- [24] J. Arias, *Ilustración Metodología de Investigación*, Loja: Anón, 2017.
- [25] J. Arias, *Ilustración IDMóvil*, Loja: Anón, 2017.

12. ANEXOS

ANEXO I: CAPTURAS DE APLICACIÓN DE ESCRITORIO



Figura 46. Interfaz de Usuario Aplicación de Escritorio. [25]

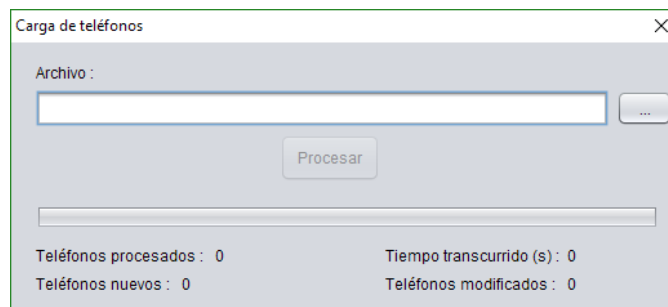


Figura 47. Interfaz de Usuario – Opción Cargar Teléfonos. [25]

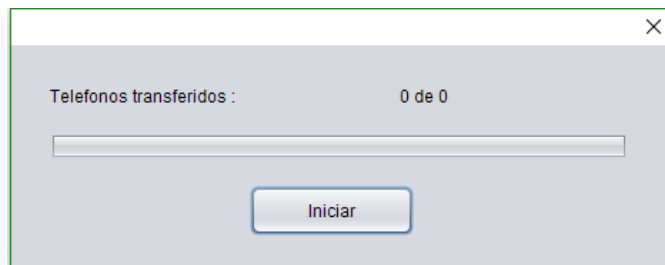


Figura 48. Interfaz de Usuario – Opción Sincronizar. [25]



Figura 49. Interfaz de Usuario – Configurar Operadoras. [25]

ANEXO 2: ENTORNOS DE DESARROLLO UTILIZADOS

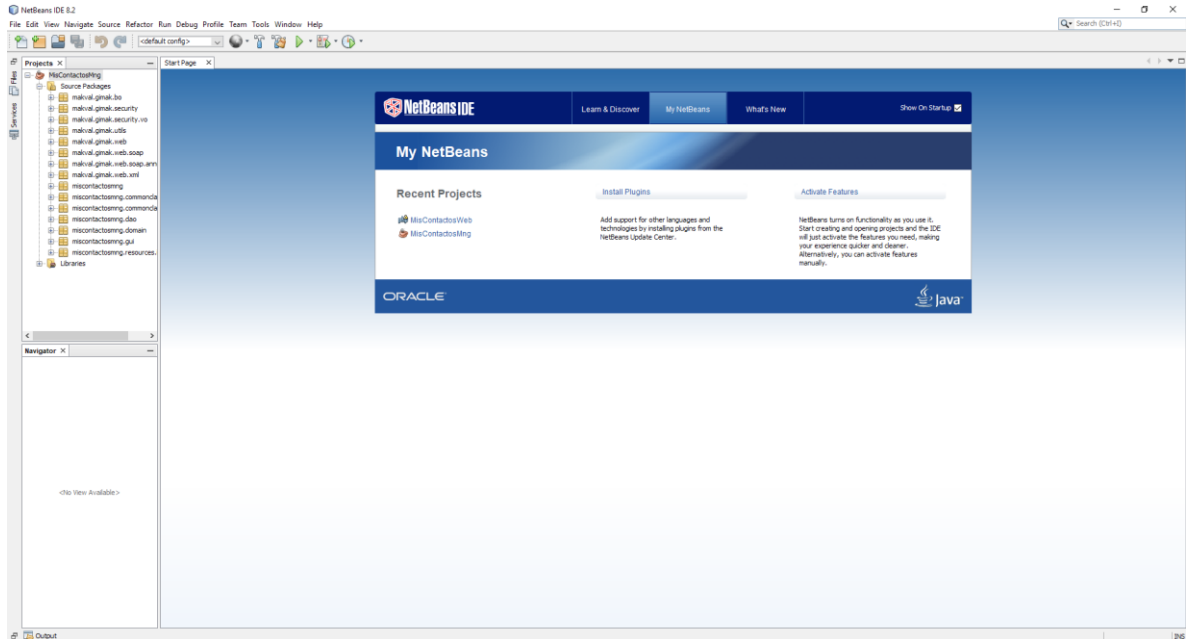


Figura 51. Entorno de desarrollo Netbeans IDE 8.2. [25]

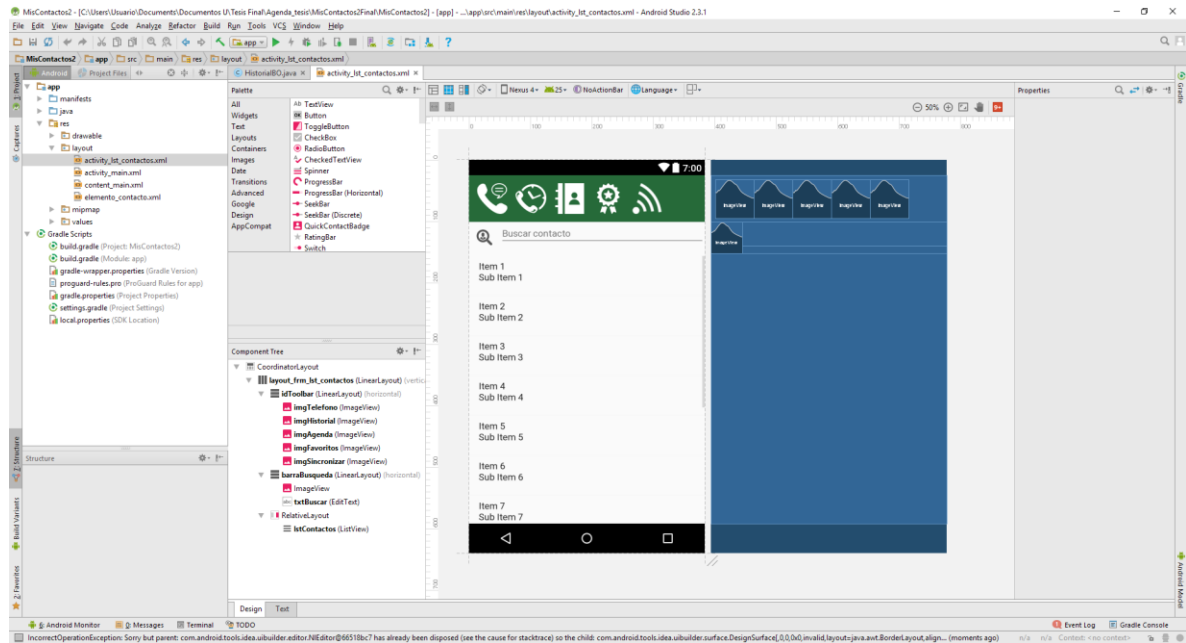


Figura 50. Entorno de desarrollo Android Studio 2.3.1. [25]

ANEXO 3: CÓDIGO FUENTE

Adjunto en CD N° 2