
**UNIVERSIDAD NACIONAL DE CHIMBORAZO**
**FACULTAD DE INGENIERÍA**

ESCUELA DE INGENIERÍA EN SISTEMÁS Y COMPUTACIÓN

**"Trabajo de grado previo a la obtención del Título de Ingeniero en Sistemas y Computación"**

**TRABAJO DE GRADUACIÓN**

**Título del proyecto**

# LA ARQUITECTURA DE LAS APLICACIONES MÓVILES BASADAS EN EL ENTORNO DE PROGRAMACIÓN JAVA Y SU APORTE CON LA SEGURIDAD Y EFICIENCIA EN LAS ACTIVIDADES DE LOS USUARIOS DE TELEFONÍA MÓVIL; CASO PRÁCTICO DESARROLLO DE UN SOFTWARE DE TRANSFORMACIÓN DE MENSAJES DE TEXTO SMS A MENSAJES DE VOZ Y VICEVERSA.

**AUTORES:**

Alfonso Guido Caguana Lliquin
Mercedes Elizabeth Guamán Ocaña

**Directora:** Ingeniera Lida Barba

Riobamba – Ecuador
2013

PÁGINA DE REVISIÓN

Los miembros del Tribunal de Graduación del proyecto de investigación de título "La arquitectura de las aplicaciones móviles basadas en el entorno de programación java y su aporte con la seguridad y eficiencia en las actividades de los usuarios de telefonía móvil; Caso Práctico Desarrollo de un software de transformación de mensajes de texto SMS a mensajes de voz y viceversa".

Presentado por:

Mercedes Elizabeth Guamán Ocaña y Guido Alfonso Caguana Lliquín.

Y dirigida por: Ing. Lida Barba

Una vez escuchada la defensa oral y revisado el informe final del proyecto de investigación con fines de graduación escrito en la cual se ha constatado el cumplimiento de las observaciones realizadas, remite la presente para uso y custodia en la biblioteca de la Facultad de Ingeniería de la UNACH.

Para constancia de lo expuesto firman:

Ing. Fernando Molina ……
Presidente del Tribunal Firma

Ing. Lida Barba ……
Miembro del Tribunal Firma

Ing. Diego Reina ……
Miembro del Tribunal Firma

## AUTORÍA DE LA INVESTIGACIÓN

"La responsabilidad del contenido de este Proyecto de Graduación, nos corresponde exclusivamente a: Mercedes Elizabeth Guamán Ocaña y Guido Alfonso Caguana Lliquín y de la Ing. Lida Barba quien es la Directora del Proyecto; y el patrimonio intelectual de la misma a la Universidad Nacional de Chimborazo.

Autores:

…… ……..……..

Mercedes Elizabeth Guamán Ocaña Guido Alfonso Caguana Llíquin

Directora del proyecto:

……

Ing. Lida Barba

**AGRADECIMIENTO**

Doy gracias a Dios, por estar conmigo en cada paso que doy, por fortalecer mi corazón e iluminar mi mente y por haber puesto en mi camino a aquellas personas que han sido mi soporte y compañía durante todo el periodo de estudio.

A mis maestros quienes me han enseñado a ser mejor en la vida y a realizarme profesionalmente.

A todas y cada una de las personas que han vivido conmigo la realización de esta tesis, mis padres, hermanos y mí esposo por haberme brindado gran apoyo y paciencia.

*Mercedes Guamán*

Agradezco a Dios por haberme dado la oportunidad de llegar a este mundo, en el cual han existido tristezas, llantos y risas, pero sobre todo apoyo incondicional de mis padres en especial a mi Madre Petrona Lliquín y nuestros profesores que siempre han estado como un pilar fundamental en nuestra labor estudiantil y llegar a cumplir con el objetivo propuesto, también de manera especial a la Ing. Lida Barba, por su valioso tiempo en el seguimiento al trabajo de graduación, y por su puesto a mi compañera de Tesis Mercedes Elizabeth Guamán Ocaña por su entrega total y el cariño que fue muy importante en el desarrollo y la culminación del trabajo de investigación.

*Guido Caguana*

**DEDICATORIA**

La presente tesis se la dedico a mi familia y mí esposo que gracias a su apoyo pude concluir mi carrera.

A mis padres y hermanos por su apoyo y confianza en todo lo necesario para cumplir mis objetivos como persona y estudiante.

A mi padre por brindarme los recursos necesarios y estar a mi lado apoyándome y aconsejándome siempre.

A mi madre por hacer de mí una mejor persona través de sus consejos enseñanzas y amor.

A mi esposo por el apoyo y paciencia durante la realización de la tesis.

A todo el resto de familia y amigos que de una u otra manera me han llenado de sabiduría para terminar la tesis.

*Mercedes Guamán*

El presente trabajo de investigación va dedicado con todo el cariño del mundo a mis padres Vicente Caguana y Petrona Lliquín, por su valioso incondicional apoyo, ya que gracias a su trabajo, esfuerzo y sacrificio que día a día llevan a cabo, he podido culminar con una más de las metas que he anhelado desde hace mucho tiempo, en especial a mi Madre PetronaLliquín, que ha sido la luz que siempre a guiado mi camino.

También al Ing. Jorge Delgado que siempre ha estado presente en los momentos más difíciles de mi vida aun cuando pensé en desistir de mis sueños siendo mi amigo y mi consejero y a todos quienes velaban por mis sueños.

*Guido Caguana*

## ÍNDICE DE TABLAS

**INDICE DE ILUSTRACIONES**

# ÍNDICE

## RESUMEN

El presente trabajo de investigación, consiste en la descripción y el análisis de la arquitectura de aplicaciones móviles basadas en el entorno de programación Java y su aporte en la eficiencia y seguridad para los usuarios de telefonía móvil.

La razón de la investigación es el analizar los aspectos que contemplan el funcionamiento de la estructura de una aplicación móvil que se ha convertido en una de las herramientas de desarrollo por parte de los usuarios de telefonía móvil, ya que estas aplicaciones son útiles en las labores diarias.

El problema, por otro lado, es la inquietud de investigar los factores que se involucran en el desarrollo de una aplicación móvil.

La presente investigación se elaboró en base a varios tipos de investigación en este caso se utilizó una investigación aplicada que está orientada a solucionar problemas reales, una investigación bibliográfica por el proceso de búsqueda de información, una investigación relacional por la valoración y medición de la Operacionalización de las variables según el estudio de la investigación. La metodología utilizada en la investigación son el método inductivo- deductivo por las etapas de observación y registro de hechos, método de análisis de las herramientas que se utilizaron en el desarrollo del software, método bibliográfico por las fuentes bibliográficas.

La hipótesis de investigación que es la arquitectura de las aplicaciones móviles basadas en el entorno de programación Java aporta con la eficiencia en el desarrollo y seguridad para el usuario móvil se acepta por medio de la prueba de hipótesis chi cuadrado y se rechaza la hipótesis nula que es la arquitectura de las aplicaciones móviles basadas en el entorno de programación Java no aporta con la eficiencia en el desarrollo y seguridad para el usuario móvil.

**SUMMARY**

The present research work consists in the description and analysis of mobile application architecture based on the Java programming environment and its contribution to the efficiency and security for mobile users.

The reason research is to analyze aspects that contemplate the operation of the structure of a mobile application that has become one of the development tools by mobile users, as these applications are useful in the work daily.

The problem, on the other hand, the concern is to investigate the factors involved in developing a mobile application.

This research was developed based on various types of research in this case, an applied research that is aimed at solving real problems, a literature research through the process of finding information, relational research by the valuation and measurement of operationalization of the variables in the research study. The methodology used in the research are the inductive-deductive stages of observation and recording of events, method of analysis tools used in software development, literature method for bibliographic sources.

The research hypothesis is the mobile application architecture based on the Java programming environment contributes to the development efficiency and security for the mobile user is accepted by the chi-square hypothesis test and reject the null hypothesis that is the mobile application architecture based on the Java programming environment does not contribute to efficiency in the development and security for the mobile user.

# INTRODUCCIÓN

Las leyes de tránsito en el Ecuador son rígidas, las penas para el uso de celular mientras se conduce un vehículo, hecho que no únicamente pasa por el ámbito legal sino que realmente puede traer graves consecuencias por la desconcentración ocasionada en el conductor.

Más aun la utilización del celular en llamadas y envió de mensajes de texto produce una desatención del conductor al momento de conducir su vehículo, siendo una de las causas principales para los accidentes de tránsito ocurrido en el país.

Él envió de mensajes de texto provoca que el conductor tenga que utilizar su mano o peor aún sus dos manos y su vista en el envío de dichos mensajes provocando que tenga que descuidar el volante y quitar su vista de la carretera, cuando la atención que debe prestarla es hacia la conducción del vehículo.

No existen celulares que no disponen de una KVM (Kilo Virtual Machine, la máquina virtual de Java para móviles) con una implementación de Java ME; y la mayoría de los juegos disponibles están desarrollados con esta plataforma.

En la actualidad un teléfono móvil, permite la comunicación de señales de voz digitalizadas, se sirve de Internet para conectarnos con cualquier red pública o privada, tiene capacidad para almacenar toda la información personal que puede necesitar, interacciona con el entorno, ayuda a conocer las coordenadas de la posición que se ocupa en un momento dado en el planeta, se puede emplear como radio y como máquina de fotos también permite jugar, trabajar, negociar, comprar, vender, informar y ser informados.

Los conductores de vehículos deben utilizar correctamente el celular, por este motivo la utilización de manos libres, bluetooth y una aplicación móvil para Android que ofrece seguridad y eficiencia al conductor y a sus acompañantes en el momento de conducir, ya que al recibir un mensaje de texto no tienen que utilizar las manos para leer o responder dicho mensaje este mensaje de texto llega como mensaje de voz y viceversa.

En el capítulo I se aborda toda la teoría referente a la investigación por ende al caso aplicativo como son conceptos de: Comunicación Móvil, La Arquitectura de la

Aplicaciones Móviles, el Patrón de desarrollo de la aplicación Habla SMS, el sistema de Reconocimiento de voz, La Seguridad y la Eficiencia de las Aplicaciones Móviles

En el capítulo II se describe los métodos que se toman en cuenta para la elaboración de la investigación como el tipo de estudio, la población, el procedimiento todo en cuanto tenga que ver con la investigación y el caso aplicativo.

En el capítulo III se describe los diferentes resultados obtenidos en la investigación para la comprobación de la hipótesis que es el aporte de la aplicación móvil HablaSMS en la seguridad y eficiencia del usuario de telefonía móvil e incluso dispositivos móviles que estén acorde a los requerimientos de hardware y software.

En el capítulo IV se describen las propuestas de la investigación, la ingeniería de software utilizada para el desarrollo de la aplicación HablaSMS, cuales son los requerimientos, los Smartphone e incluso el código fuente de dicha aplicación.

En el capítulo V se describe las conclusiones y recomendaciones que se obtiene en el transcurso de la realización de la investigación, recolección de datos para llegar al objetivo propuesto.

En el capítulo VI se describe las diferentes fuentes de investigación siguiendo el estándar de las Normas APA.

En el capítulo VII se anexa las fichas de hardware y software que se utilizó en la investigación y para el desarrollo de la aplicación HablaSMS, como se diseñaron las encuestas, glosario de términos, cd del proyecto de tesis.

# CAPITULO I
# FUNDAMENTACIÓN TEÓRICA

## 1.1 COMUNICACIÓN MÓVIL

La comunicación móvil es posible gracias a la interconexión entre centrales móviles y públicas. Según las bandas o frecuencias en las que operan el móvil, podrá funcionar en cualquier parte del mundo. Las comunicaciones móviles, se da cuando tanto emisor como receptor están en movimiento.

La movilidad de estos dos factores que se encuentran en los extremos de la comunicación hace que se excluya casi en su integridad la utilización de cables para realizar la comunicación en dichos extremos.

Se utiliza la comunicación vía radio, la ventaja es la movilidad de los extremos de la conexión.

Las comunicaciones móviles disponen de sistemas de telefonía móvil, Radio búsquedas (GPS), redes móviles privadas o Trunking[1], y sistemas de telefonía móvil avanzados, también la telefonía móvil digital, las agendas personales, laptops, notebooks y un sin fin de dispositivos dispuestos a conectarse vía radio con otros dispositivos o redes.

La tecnología móvil se forma básicamente por dos elementos: la red de comunicaciones y las terminales. En los años ochenta aparece la primera generación de telefonía móvil (1G), basadas en redes análogas con una limitada cobertura y capacidad de tráfico.

A principios de los años noventa, surge la segunda generación (2G), caracterizada por la digitalización de la información de voz y el aumento de la eficiencia espectral, lo que permitió un incremento de la calidad y la prestación de nuevos servicios como la mensajería instantánea (SMS). (Vásquez J. A., 2009)

En la segunda generación coexisten distintas tecnologías, en las que se destacan el acceso múltiple por división de tiempo (TDMA), el sistema global para las comunicaciones móvil (GSM), el celular digital personal (PDC), y el acceso múltiple por división de códigos (CDMA).

[1]Trunking: Sistemas de radiocomunicaciones móviles para aplicaciones privadas, formando grupos y subgrupos de usuarios.

La Unión Internacional de Telecomunicaciones estableció los requisitos técnicos y las normas para las futuras comunicaciones móviles avanzadas denominadas de tercera generación (3G). (Caldeón, 2007)

La telefonía móvil, también llamada telefonía celular, básicamente está formada por dos grandes partes: una red de comunicaciones y los terminales que permiten el acceso a dicha red.

El teléfono móvil es un dispositivo inalámbrico electrónico para acceder y utilizar los servicios de la red de telefonía celular o móvil.

Se denomina celular en la mayoría de países latinoamericanos debido a que el servicio funciona mediante una red de celdas, donde cada antena repetidora de señal es una célula, si bien también existen redes telefónicas móviles satelitales.

Su principal característica es su portabilidad, que permite comunicarse desde casi cualquier lugar.

La principal función es la comunicación de voz, como el teléfono convencional.

A partir del siglo XXI, los teléfonos móviles han adquirido funcionalidades que van mucho más allá de limitarse solo a llamar o enviar mensajes de texto, se podría decir que se han unificado con distintos dispositivos tales como PDA, cámara de fotos, agenda electrónica, reloj despertador, calculadora, micro proyector, GPS o reproductor multimedia, así como poder realizar una multitud de acciones en un dispositivo pequeño y portátil. (Marcombo, 1998).

## 1.2 APLICACIONES MÓVILES

Las aplicaciones móviles se dividen en tres grandes categorías en función de su arquitectura.

Comprender todo tipo de aplicaciones y su infraestructura de la información es necesario para llevar a cabo la actividad con éxito las pruebas de rendimiento y son las siguientes:

- Aplicación Nativas
- Aplicación Web
- Aplicación Híbrida

### 1.2.1 *APLICACIÓN NATIVA*

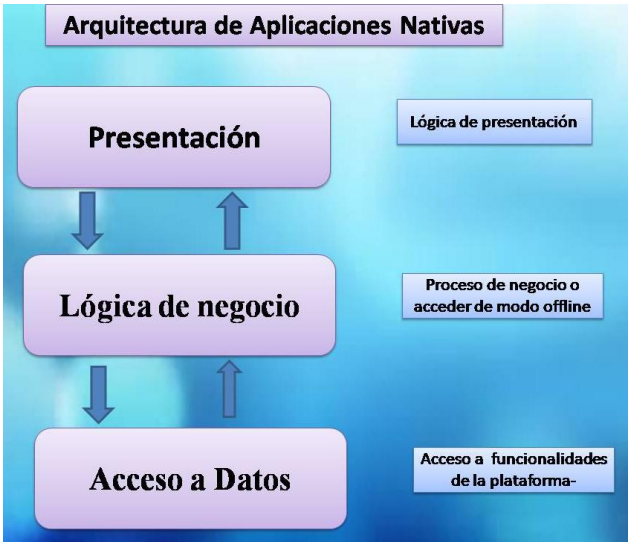



Ilustración 1: Arquitectura de Aplicaciones Nativas,
Autores: Mercedes Guamán, Alfonso Caguana.

La aplicación nativa se desarrolla utilizando un lenguaje de programación específico como Java para Android y Objetive-C para iOS y API dispositivo específico.

Los usuarios móviles tienen acceso a estas aplicaciones en todo momento sin necesidad de conectarse a Internet una vez que se instalen en sus dispositivos.

La aplicación nativa se basa en el dispositivo donde se implementa toda la interfaz, acceso a funcionalidades de la plataforma (cámara y demás) y lógica (al menos la que necesite ser accedida de modo offline). Si existe cierta información que no necesite ser accesible de modo constante y offline, sino que basta con recuperarla cada vez que haya una conexión disponible, se podría implementar una aplicación web o servicio web que implementase parte de la lógica y recuperación de información.

En este enfoque, la misma funcionalidad debe ser implementada desde cero para cada plataforma, ya que cada una de ellas usa su propio lenguaje de programación, por lo que no tenemos la opción de compartir implementaciones de funcionalidad común. (Pérez, 2011).

### 1.2.2 *APLICACIÓN WEB*


Ilustración 2: Arquitectura de Aplicaciones Web,

Autores: Mercedes Guamán, Alfonso Caguana

La aplicación web se ejecuta dentro del navegador web del dispositivo, desde el cual accede a la página que el servidor envía utilizando las mismas facilidades que un cliente de sobremesa.

La capa de presentación es responsable de generar la interfaz de usuario con la información proporcionada por la capa lógica de negocio.

La capa de lógica de negocio es responsable de implementar la lógica de la aplicación como respuesta a las peticiones de usuario, normalmente accediendo a la capa de datos.

La capa de datos es responsable de proporcionar acceso a los datos a la capa de lógica de negocio, normalmente accediendo a un sistema de gestión de base de datos.

Para utilizar una aplicación web el dispositivo requiere una conexión a un servidor para que la aplicación móvil pueda ejecutar (Pérez, 2011).

Estas aplicaciones son desarrolladas con tecnologías web como HTML, JQuery y JavaScript. Populares aplicaciones de medios sociales como Facebook y Gmail tienen sus distintas aplicaciones móviles basadas en la web que son muy famosos entre los usuarios móviles. (Pérez, 2011).

### 1.2.3 *APLICACIÓN HIBRIDAS*


Ilustración 3: Arquitectura de Aplicaciones Híbridas,

Autores: Mercedes Guamán, Alfonso Caguana

Básicamente la aplicación hibrida está diseñada de la siguiente manera:

- En la capa inferior se encuentra un Contenedor de Dispositivo (DC o Device Container) escrito con el SDK (aplicación nativa). Éste proporciona a las capas superiores acceso a las funciones del dispositivo (cámara, acelerómetro, etc.) y una interfaz común (DII o Device integration interfaces) para poder abstraer esta capa y hacer a las demás independientes del dispositivo
- Sobre el DC encontramos el Web Container, que en esencia se trata de un Navegador Web con capacidades para renderizar HTML y ejecutar JavaScript. Aquí el acceso a los servicios que dé la cámara, GPS, agenda, etc. los encontramos disponibles a través de funciones JS o lenguaje de marcas, dependiendo de las tecnologías ofrecidas por el framework.

Son la combinación de aplicaciones móviles web y aplicaciones móviles nativas. En tales aplicaciones, las aplicaciones web están integradas en aplicaciones nativas para móviles.

La interfaz de usuario es normalmente cubierto en parte nativa de la aplicación, mientras que el contenido se carga a través de la parte web de la aplicación. (Weiss, 2002).

### 1.2.4 *BENEFICIOS E INCONVENIENTES DE LAS APLICACIONES MÓVILES*

Tabla 1: Beneficios e inconvenientes en el desarrollo de aplicaciones para dispositivos móviles

Autores: Mercedes Guamán, Alfonso Caguana

| APLICACIONES | BENEFICIOS | INCONVENIENTES |
|---|---|---|
| **Nativa** | ○ Aplicaciones sofisticadas y control sobre el entorno local.<br>○ Capacidad multitarea sobre muchas plataformas. | ○ El más alto nivel de esfuerzo de desarrollo.<br>○ Diferente código base para diferentes dispositivos. |
| **Web** | ○ Desarrollo rápido.<br>○ No requiere mantenimiento del código cliente.<br>○ Métodos web 2.0 disponibles.<br>○ Trabaja con un amplio rango de dispositivos móviles. | ○ Menos sensible y capas que las aplicaciones nativas. |
| **Hibridas** | ○ El mismo código base puede soportar múltiples aplicaciones.<br>○ Aplicaciones más sofisticadas. | ○ Algunos límites de capacidad no multitarea.<br>○ Requiere pruebas y adaptación para las plataformas. |

### 1.2.5 *CARACTERÍSTICAS DE LAS APLICACIONES MÓVILES*

- ✓ Operación consistente tanto online como offline. En varias arquitecturas, los datos son almacenados en un sistema compartido accesible a través de la red, en forma de documentos, registros de datos o archivos binarios, donde se tiene un acceso coordinado a una copia de la información. Una aplicación móvil debe ser diseñada de forma que los usuarios puedan acceder a los datos sin importar si lo hace en forma online o en forma offline. Cuando se trabaja offline, el usuario percibe que la información compartida está disponible para lectura y escritura. Cuando la conectividad regresa, los cambios en la información local son integrados a la copia de red y viceversa.

- ✓ Conectividad Continua. Una aplicación diseñada para movilidad debe trabajar con un agente o servicio Proxy para permitir un manejo transparente de los cambios en la conectividad. La conectividad no tiene que ser un requerimiento para la funcionalidad y cortes intermitentes e inesperados en la conexión con la red deberían ser manejados satisfactoriamente. Así mismo este agente o servicio Proxy debe poder seleccionar la red óptima de las disponibles en ese momento, y manejar las tareas propias de la comunicación como autentificación segura o autorización y direccionamiento lógico.
- ✓ Clientes que soporten multi-plataformas. Una aplicación diseñada para movilidad debe al menos ajustar su interacción y comportamiento al dispositivo en el que corre, como por ejemplo tipo de entrada y salida, recursos disponibles y nivel de performance.
- ✓ Energía y performance optimizadas. Una aplicación diseñada para movilidad debe manejar de cerca el uso de la energía de un dispositivo portátil que comúnmente funciona a baterías. Por ejemplo una constante sincronización de los datos en memoria con los del disco rígido puede consumir las baterías rápidamente, al igual que con la actividad de radio al buscar constantemente el siguiente punto de comunicación.
- ✓ Administración de Recursos. Un recurso como la energía, el ancho de banda o el espacio de almacenamiento puede ser consumido y existe en una cantidad finita. La administración de recursos debe permitir el monitoreo de atributos como cantidad o tasa de uso, y soportar notificaciones basadas en disparadores predefinidos por el usuario.
- ✓ Administración del Contexto. Contexto es cualquier información que puede ser usada para caracterizar la situación de una entidad. Donde una entidad es una persona, lugar u objeto que es relevante para la interacción entre un usuario y una aplicación, incluyendo al usuario y la aplicación. La administración del contexto debe permitir el monitoreo de atributos como ubicación actual o tipo de dispositivo, y proveer notificación de cambios en el mismo.
- ✓ Codificación. La codificación involucra la modificación de los datos y protocolo en función de los requerimientos del actual contexto y recursos disponibles. Ejemplos de codificación son la encriptación, compresión y transcodificación.

- ✓ Vista consistente. Debe proveer una vista consistente y reconciliable de datos y el estado en que es compartida en medio de un sistema intermitentemente conectado, utilizando emulación, "pre-fetch" y reconciliación. La emulación permite a los usuario despreocuparse si están trabajando con una copia local o remota de la información. El "pre-fetch" maneja el caching de los datos. Y la reconciliación se asegura que aquellas modificaciones que hayan sido realizadas en estado offline, sean aplicadas a todas las copias de la información.
- ✓ Almacenamiento Duradero. La capacidad de manejar un almacenamiento duradero permite la persistencia de datos de configuración o información estática.
- ✓ Mensajería Confiable. La capacidad de una mensajería confiable provee la habilidad para definir y controlar la semántica del mensaje a entregar así como el tipo de entrega (sincrónica, asincrónica, etc.).
- ✓ Políticas. El uso de políticas permite tener un lugar común para colectar, correlacionar y reaccionar o adaptar la información y eventos provistos por las otras habilidades. Estas reciben los eventos y los confrontan a un conjunto de reglas previamente almacenadas. Si el resultado de esta confrontación es verdadero se ejecuta la acción previamente asociada al evento. A estos conjuntos de eventos, condiciones y acciones se los conoce comúnmente como ECA (Event-Condition-Action). Una implementación de políticas debe permitir definir, almacenar y administrar ECAs, y de ser capaz de interactuar con otras capacidades, recibiendo eventos e invocando métodos. Por ejemplo una condición como "bajo ancho de banda", puede tener una acción asociada como "usar compresión". Entonces al recibir el evento de bajo ancho de banda, en base a las políticas se evaluará el mismo, y de ser verdadero, se usará la capacidad de codificación, compresión donde sea posible.
- ✓ Seguridad. Para evitar las consecuencias de ataques maliciosos, aplicaciones con diseños pobres, y errores inadvertidos de usuarios, se deben tomar ciertas medidas de seguridad como son: Sistemas y usuarios, autentificación se sistemas, usuarios y acciones deben ser autorizados, y acciones e interacciones deben ser auditadas.

### 1.2.6 *SISTEMAS OPERATIVOS DE DISPOSITIVOS MÓVILES*


Ilustración 4: Participación de mercado mundial de sistemas operativos en teléfonos móviles 1Q 2012, Tomado de la fuente: http://www.evaluamos.com/2011/internal.php?load=detail&id=13283

Los sistemas operativos para dispositivos móviles que hay en la actualidad son:

#### 1.2.6.1 Android

Android es una plataforma formada por un conjunto de software en estructura de pila (software stack) que incluye un sistema operativo, software para conectar aplicaciones (middleware) y aplicaciones base. El SDK[2] de Android proporciona varias herramientas y API (Applications Programming Interface, Interfaz de Programación de Aplicaciones) que son necesarias para desarrollar aplicaciones Android.

Estas aplicaciones se desarrollan en lenguaje Java. Android está desarrollado por Open Handset Alliance (OHA), una agrupación de 78 compañías para desarrollar estándares abiertos para dispositivos móviles y que está liderada por Google. Inicialmente Android fue desarrollado por la compañía Android Inc., que fue comprada en el año 2005 por Google.

El sistema operativo se anunció el 5 de noviembre de 2007. Google libera la mayoría del código Android bajo una licencia Apache (licencia libre y de código abierto).

[2]SDK: Software Development Kit, Kit de Desarrollo de Software

Desde su creación ha ido pasando por diferentes versiones, desde la versión primera (1.0) hasta la actual (4.0, denominada también Ice Cream Sandwich).

Android se ha convertido de forma rápida en uno de los SO de dispositivos móviles con mayor presencia. Actualmente existen más de 200 millones de dispositivos móviles Android activados y cada día se activan más de 550.000 nuevos dispositivos en 137 países y regiones.

Las ventajas de Android que han hecho posible su gran éxito son:

- El ser código abierto con licencia Apache, lo cual permite que un desarrollador pueda, no solo ver el código, sino mejorarlo y ampliarlo.
- Dar libertad al usuario del dispositivo para instalar el software que crea oportuno sin imponer que sea software propietario.
- Los desarrolladores tienen libertad para desarrollar cualquier software y ofertarlo a los usuarios (dispone de una amplia comunidad de desarrolladores).
- No está limitado a determinados proveedores, operadoras o fabricantes. (Vásquez, 2011).

### *1.2.6.2* Arquitectura del sistema operativo Android

La arquitectura del sistema operativo Android está organizada en cuatro grandes bloques:

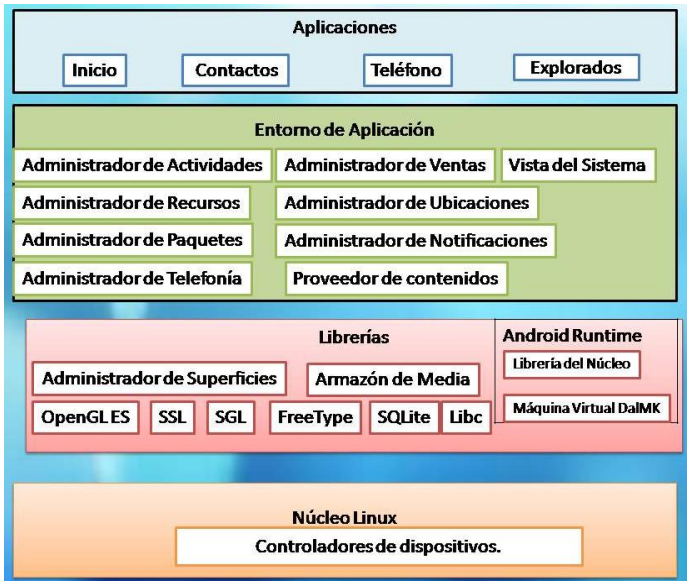



Ilustración 5: Arquitectura de SO Android,

Autores: Mercedes Guamán, Alfonso Caguana.

**El núcleo Linux de Android**.- Esta capa proporciona servicios como la seguridad, el manejo de la memoria, el multiproceso, la pila de protocolos y el soporte de drivers para dispositivos.

Esta capa del modelo actúa como capa de abstracción entre el hardware y el resto de la pila. Por lo tanto, es la única que es dependiente del hardware**.**

**Runtime de Android.-** Está basado en el concepto de máquina virtual utilizado en Java. Dado las limitaciones de los dispositivos donde se ejecuta Android (poca memoria y procesador limitado) no fue posible utilizar una máquina virtual Java estándar. Google tomó la decisión de crear una nueva, la máquina virtual Dalvik, que respondiera mejor a estas limitaciones.

Algunas características de la máquina virtual Dalvik que facilitan esta optimización de recursos son: que ejecuta ficheros Dalvik ejecutables (.dex) –formato optimizado para ahorrar memoria. Además, está basada en registros. Cada aplicación ejecuta en su propio proceso Linux con su propia instancia de la máquina virtual Dalvik. Delega al kernel de Linux algunas funciones como threading y el manejo de la memoria a bajo nivel.

También se incluye en el Runtine de Android el "corelibraries" con la mayoría de las librerías disponibles en el lenguaje Java.

**Librerías nativas.-** Incluye un conjunto de librerías en C/C++ usadas en varios componentes de Android. Están compiladas en código nativo del procesador. Muchas de las librerías utilizan proyectos de código abierto. Algunas de estas librerías son:

- **System C library**: Una derivación de la librería estándar (libc), adaptada para dispositivos embebidos basados en Linux.
- **Media Framework**: Librería basada en Packet Video's Open CORE; soporta codecs de reproducción y grabación de multitud de formatos de audio vídeo e imágenes MPEG4, H.264, MP3, AAC, AMR, JPG y PNG.
- **Surface Manager**: Maneja el acceso al subsistema de representación gráfica en 2D y 3D.
- **Web Kit**: Soporta un moderno navegador web utilizado en el navegador Android y en la vista webview. Se trata de la misma librería que utiliza Google Chrome y Safari de Apple.
- **SGL**: Motor de gráficos 2D.

- **Librerías 3D**: Implementación basada en OpenGL ES 1.0 API. Las librerías utilizan el acelerador hardware 3D si está disponible, o el software altamente optimizado de proyección 3D.
- **FreeType**: Fuentes en bitmap y renderizado vectorial.
- **SQLite**: Potente y ligero motor de bases de datos relacionales disponible para todas las aplicaciones.
- **SSL**: Proporciona servicios de encriptación Secure Socket Layer.

**Entorno de aplicación.-** Proporciona una plataforma de desarrollo libre para aplicaciones con gran riqueza e innovaciones (sensores, localización, servicios, barra de notificaciones).

Esta capa es diseñada para simplificar la reutilización de componentes. Las aplicaciones pueden publicar sus capacidades y otras pueden hacer uso de ellas (sujetas a las restricciones de seguridad). Este mismo mecanismo permite a los usuarios reemplazar componentes.

Una de las mayores fortalezas del entorno de aplicación de Android es que se aprovecha el lenguaje de programación Java. El SDK de Android no acaba de ofrecer todo lo disponible para su estándar del entorno de ejecución Java (JRE), pero es compatible con una fracción muy significativa de la misma. (Luis, 2008)

Los servicios más importantes que incluye son:

- **Views**: Extenso conjunto de vistas, (parte visual de los componentes).
- **Resource Manager**: Proporciona acceso a recursos que no son en código.
- **Activity Manager**: Maneja el ciclo de vida de las aplicaciones y proporciona un sistema de navegación entre ellas.
- **Notification Manager**: Permite a las aplicaciones mostrar alertas personalizadas en la barra de estado.
- **Content Providers**: Mecanismo sencillo para acceder a datos de otras aplicaciones (como los contactos).

**Aplicaciones.-** Este nivel está formado por el conjunto de aplicaciones instaladas en una máquina Android. Todas las aplicaciones se ejecutan en la máquina virtual Dalvik para garantizar la seguridad del sistema.

Normalmente las aplicaciones Android están escritas en Java. Para desarrollar aplicaciones en Java podemos utilizar el Android SDK.

#### 1.2.6.3 BlackBerry

BlackBerry está desarrollado por la compañía RIM (Research In Motion). Los móviles BlackBerry destacan principalmente por su capacidad de enviar y recibir correo electrónico por Internet a través de los operadores que ofrecen este servicio. Los dispositivos BlackBerry montan el sistema operativo BlackBerry OS, desarrollado por RIM. Este sistema operativo es propietario, con lo cual no hay información pública relevante sobre su diseño ni arquitectura. En la actualidad la última versión del sistema operativo es BlackBerry OS 7.

Las versiones del sistema operativo tienen un Kernel que se basa en Java, montando la mayoría de los dispositivos arquitecturas ARM. ARM no construye sus propios chips CPU pero da licencia a terceros para que los fabriquen.

El sistema operativo divide la memoria del dispositivo en tres secciones:

- Memoria de aplicación: espacio de memoria destinado para albergar las aplicaciones.
- Memoria de dispositivo: espacio para almacenar ficheros y otras fuentes.
- Memoria para tarjeta: espacio de almacenamiento adicional.

BlackBerry OS es un sistema operativo multitarea. Esto significa que puede ejecutar más de una aplicación a la vez. Por ejemplo, mientras que se está realizando una llamada, el usuario puede cambiar y consultar el calendario o los contactos sin cortar la llamada.

Estas aplicaciones quedan en modo background y continúa así su ejecución dando la ejecución e interacción del usuario a otras aplicaciones. Hay que tener en cuenta que muchas aplicaciones ejecutándose en background pueden dar la sensación que no hay mucha carga de ejecución para la máquina, pero lo cierto es que hay un alto consumo de memoria.

#### 1.2.6.4 Symbian

El sistema operativo Symbian viene como evolución del sistema operativo Epoc, este fue desarrollado por Psion en sus agendas electrónicas durante los 80.

Concretamente Symbian Inc. desarrolló el sistema operativo base y les vendió la licencia a los distintos fabricantes de teléfonos móviles. Estos a su vez construyen

interfaces de usuario sobre este sistema operativo base y personalizan el sistema para propósitos específicos.

Como hay muchas combinaciones de SO y UI, habrá también muchos kits de desarrollo (SDK), por ejemplo para la Serie60 hay SDK versión 6.1, 7.0, 8.0 y 9.0. Si nos centramos en la Serie60, tenemos que para cada edición de un SDK hay ampliaciones con mejoras llamadas paquetes de características o FP (Feature Pack), así tenemos:

- SDK for 1st Edition.
- SDK for 1st Edition, FP1.
- SDK for 2st Edition.
- SDK for 2st Edition, FP1.
- SDK for 2st Edition, FP2.

Además, para cada kit de desarrollo hay una variante en función del IDE en el que se instale por lo que habrá variantes diferentes.

Normalmente el entorno de desarrollo viene preparado para estar programado en C o C++ y aporta el emulador de la Serie para la que está desarrollado.

#### 1.2.6.5 Palms OS

Palm OS es el sistema operativo de los dispositivos móviles desarrollados por la empresa Palm Inc.

Esta empresa tuvo un gran auge con las agendas electrónicas o PDA. Comenzó su actividad en 1996, creando Palm OS, un sistema operativo fácil de utilizar con pantallas táctiles e interfaces de usuario gráficas.

Palm OS tiene una gran evolución de versiones desde la 1.0 hasta la 5.0, luego pasó a llamarse Palm OS Cobalt. Este último es un sistema operativo basado en Linux, que evolucionó para denominarse web OS en 2009, el primer dispositivo que lo incluía fue Palm Pre.

La compañía Palm Inc., fue adquirida en 2010 por HP y, actualmente, HP utiliza web OS en sus dispositivos móviles y tablets, como Pixi, Veer y HP TouchPad.

#### 1.2.6.6 Windows Phone

Este sistema operativo es lanzado a finales del año 2010 tras dos años de desarrollo. Entre las principales novedades se encuentran la denominada interfaz de usuario "Metro" basada en la utilización de mosaicos dinámicos que muestran información útil al usuario. Además se introduce el concepto de HUB, en donde se centralizan las acciones y las aplicaciones se agrupan por el tipo de actividad que representan. Por lo tanto, se encuentran diferentes HUB, por ejemplo, de Office, Xbox Live, Imágenes o Zune, desde los cuales tenemos acceso a tareas específicas. También incluye el motor de Internet Explorer 9, con soporte para HTML5, multitarea en aplicaciones de terceros e integración con Xbox 360 y Kinet.

Modelo de hardware: para ejecutar Windows Phone y asegurar la consistencia de todos los usuarios del sistema, es necesario que el teléfono cuente con unas características mínimas, o lo que se conoce como Chassis 1, que constituyen las especificaciones que debe seguir todo fabricante que quiera crear terminales con soporte para este SO:

- Procesador: ARMv7 Cortex / Scorpion a 1 GHz.
- Procesador gráfico: soporte hardware completo de DirectX9.
- Memoria: RAM 256 MB y ROM 8 GB.
- Sensores: A-GPS, acelerómetro, brújula, iluminación y proximidad.
- Cámara: 5 Mpx con flash y botón físico de disparo.
- Multimedia: aceleración de audio y vídeo por hardware.
- Pantalla: capacitiva, resolución 800 × 480.
- Botones físicos: Inicio, Buscar y Atrás.

Las especificaciones irán adaptando a los nuevos tipos de teléfonos móviles, manteniendo la experiencia del usuario independientemente del dispositivo que utilice.

Modelo software: Windows Phone se basa en el Windows CE 6.0 R3, el cual tiene soporte para Silverlight Mobile, Internet Explorer Embedded, entre otras tecnologías. También soporta Flash Lite de forma nativa en el sistema (no disponible en la versión Windows Phone 7.5). La Shell y la plataforma de aplicaciones residen en la memoria del usuario, mientras que el Kernel, los drivers y el sistema de archivos, networking, el sistema de rendering y gráficos y el sistema de actualizaciones,

residen en el espacio de Kernel. El sistema es de 32 bits, con lo cual maneja 4 GB de memoria, dos de los cuales son para procesos y los otros dos para el Kernel.

#### 1.2.6.7 IOS

A mediados de 2007 la tecnología Apple ofreció iOS, desarrollado originalmente para el IPhone y con él, una nueva definición del teléfono móvil. Más tarde fue introducido en el iPod Touch y actualmente en el IPad. Las actualizaciones de este S.O se enumeraron desde la 1.x hasta la 1.1.5. La versión 1.0 incorporaba aplicaciones como Mail, Fotos, iPod, Calculadora, entre otras, presentes en las versiones actuales y que no han sido modificadas prácticamente ni en las interfaces ni en las funcionalidades.

En el año 2008, se lanzó el iPhone OS 2.0, cuyas actualizaciones llegaron hasta la 2.2. Comenzó entonces la revolución de las aplicaciones móviles y uno de los modelos de negocio más productivos existentes en la actualidad. En el año 2009 se lanzó el iPhone OS 3.0 que evolucionó hasta la versión 3.1.3, la cual incluía el Spot light (para realizar búsquedas en el dispositivo), también ofrecía la posibilidad de incluir la API de Google Maps, las operaciones de copiar/cortar/pegar, interconexión por BlueTooth o P2P y librerías GPS. Esta versión estaba soportada por los iPhone e IPad de primera generación. En el 2010, empezó a llamarse iOS y se realizó el lanzamiento de la versión 4.0 del S.O.

La versión 4.1.2 fue la última en los iPhone 3G y los iPod Touch de segunda generación.

A finales de 2011 se lanzó la versión iOS 5, con una interfaz mejorada y funcionalidades como la presencia de un asistente personal Siri, facilidades para la sincronización sin cables, un centro de notificaciones mejorado, el servicio de iMessage, la navegación web con pestañas, entre otras.

Tabla 2: Sistemas Operativos,
Autora: Mercedes Guamán

| **S.O Características** | **Symbian** | **Windows Phone** | **IPhone** | **Palm** | **Android** | **BlackBerry** |
|---|---|---|---|---|---|---|
| Versión 2012 | Nokia Belle | Windows Phone 8 | IOS 5.0.1 | Garner OS 5.5 | 4.0.4 Ice Cream Sándwich | OS 7.0 |
| Dispositivos | Smartphone | Smartphone PDA | IPhone 3 y 4 | PDA | Smartphone | Smartphone |
| Interfaz | Apuntador Teclado | Apuntador Teclado | Touch | Apuntador Teclado | Touch Apuntador | Teclado |
| SDK | Gratuito | Gratuito/IDE Pagado | IPhone Developer Program | Gratuito | Gratuito | Gratuito |
| Aplicaciones | Nativas y JME | Nativas Compact Framework | Nativas y JME | Nativas y JME | Nativas y JME | JME |
| Firma | Obligatoria | Opcional | Obligatoria | Opcional | Opcional | Opcional |
| Proveedores | Nokia, Sony, Ericsson, Samsung, Siemens. | HP, HTC, Samsung, Dell. | IPhone, IPhone 3G | Familia Palm | HTC, LG, Samsung | Familia Blackberry. |

### 1.2.7 *LENGUAJE DE PROGRAMACIÓN JAVA J2ME PARA DISPOSITIVOS MÓVILES*

Para los dispositivos móviles la versión de Java en la que se programa es J2ME (Java Micro Edition), esta versión contiene la colección de tecnologías y de especificaciones para el desarrollo de aplicaciones con capacidades restringidas tanto en pantalla gráfica como de procesamiento y memoria. Contiene un perfil para dispositivos de información móvil conocido como MIDP (Mobile Information Device Profile), este perfil se apoya en CLDC (Connected Limited Device Configuration) y proporciona los paquetes y clases necesarios.

En el caso del lenguaje de java que es la herramienta de trabajo para la aplicación de la metodología orientada a objetos, existe un paquete que reúne a las clases que

permiten implementar lo que se conoce como GUI (Graphic User Interface) (Thefreakshow, oct. 2012).

La principal característica de utilizar las clases que se encuentran en este paquete es que el ambiente visual programado tendrá el diseño dependiendo de la plataforma en donde necesite, es decir si se programa en el sistema operativo Windows se adoptará éste ambiente visual para la interfaz del usuario, de la misma forma si se encuentra en el sistema operativo de Macintosh o cualquier otro sistema operativo, adoptará el diseño de los componentes de dicho sistema. (Díaz, 2003).

Existe otro paquete que desarrolló a partir de la segunda versión del lenguaje de java el cual se llama Swing que implementa varias partes de AWT, la ventaja es que provee mayor afinidad y se comporta de la misma manera en las diferentes plataformas, otra ventaja sobre AWT es que Swing contiene abundantes funcionalidades nuevas. Por lo que los desarrolladores prefieren utilizar Swing a pesar que en todas las versiones existentes de java sigue estando disponible AWT.

También es común encontrar que se desarrolla una aplicación para un dispositivo móvil, y al querer instalarla en otro dispositivo este no lo acepta por diferentes versiones de componentes estando lejos de lograr una compatibilidad aceptable.

Para esto Sun Micro Systems desarrollo un Kit estándar de Desarrollo llamado J2ME, cuyas siglas obedecen a Java 2 Micro Edition.

J2ME propone un estándar de desarrollo para dispositivos móviles, permitiendo a los diferentes fabricantes (Motorola, Nokia, Alcatel, etc.) implementar la plataforma de ejecución (Máquina Virtual Java o JVM), o a su vez utilizar la que gratuitamente brinda Sun Micro Systems para utilizar en sus dispositivos. (Soriano, 01/10/2011).

Las aplicaciones se escriben en Java y se compilan para ejecutarse contra una máquina virtual Java (JVM), diseñada específicamente para computadoras de mano y clientes móviles. Este enfoque proporciona dos beneficios principales: tiempo de desarrollo rápido y la posibilidad de utilizar el mismo código base en un gran número de dispositivos.

La cantidad de código necesario escrito en Java es por lo general menor que si se escribe en un lenguaje de bajo nivel. Esto se debe a que la máquina virtual se encarga de manejar automáticamente muchas de las operaciones tediosas, entre ellas el manejo de memoria. Esto significa que para realizar la misma operación se necesitan pocas líneas de código y menos errores. Además, la JVM está disponible

en varios dispositivos, por lo que la aplicación cliente trabajará en diferentes dispositivos sin necesidad de mantener múltiples versiones.

Sin embargo, debido a las diferencias en las implementaciones, es necesario probar la aplicación sobre cada plataforma. Los principales inconvenientes de este enfoque son el desempeño y la flexibilidad. Si la aplicación debe caber en un pequeño espacio de memoria o realizar operaciones que requieran mucha CPU (Central Processing Unit, Unidad Central de Procesamiento), entonces Java no es adecuado. Además, es posible que uno o más dispositivos no tengan la JVM. (Sergio Gálves Rojas, 2003)

#### *1.2.7.1* Dispositivos que utilizan J2ME

Existe una gran familia de productos que soportan este estándar de programación, tales como:

- Teléfonos Celulares Convencionales
- Smart Phones
- Palms
- Hanhelds
- Otros Dispositivos Portátiles

Una aplicación J2ME programada en su totalidad bajo los estándares propuestos puede ser portada en cualquier dispositivo J2ME, con lo cual no está sometido a un solo fabricante y es más, tampoco a un solo modelo en particular dentro de la misma marca.

De esta forma está en las condiciones de tener desarrollos ágiles y escalables de esta plataforma.

Los costos de los equipos en cuestión son los costos convencionales de los teléfonos o palms que conocemos.

Logic Nets brinda varios servicios sobre ésta plataforma, tales como desarrollo de aplicaciones, diseño de arquitecturas cliente - servidor, protocolos de comunicación, seguridad, alarmas etc.

La arquitectura J2ME brinda además conectividad a Internet a través de GPRs, con lo cual todos los dispositivos telefónicos celulares convencionales tienen enlace a Internet, evitando así disponibilidad de Wifi u otros servicios.

Los teléfonos se comunican a sus compañías por GPRs de ahí se conectan libremente a Internet, pudiendo acceder a servicios públicos, tanto de su empresa como de proveedores de servicios de terceros.

Entre otras ventajas de esta forma se puede extender la aplicación core a funcionalidades J2ME, aprovechando el negocio y alimentando de nueva información móvil al sistema de forma inmediata y bajo demanda.

#### *1.2.7.2* Arquitectura J2ME

Aunque los dispositivos mencionados tales como teléfonos móviles, agendas personales (PDAs) o televisores tienen muchos aspectos en común, también son muy diferentes en cuanto a forma y función. Estos contarán con diferentes configuraciones hardware, diferentes modos de uso (uso de teclados, voz, etc.), diferentes aplicaciones y características software, así como todo un amplio rango de futuras necesidades a cubrir. (Díaz, 2003)

Para considerar esta diversidad la arquitectura J2ME está diseñada de forma modular y extensible de tal forma que se dé cabida a esta extensa variedad de dispositivos así como a aquellos que sean desarrollados en un futuro.

La arquitectura J2ME tiene en cuenta todas aquellas consideraciones relativas a los dispositivos sobre los que tendrá que trabajar. De esta forma, son tres los conceptos básicos en los que se fundamenta:

**KVM (Kilo Virtual Machine):** Ya que el mercado de venta de los dispositivos a los que se refiere J2ME es tan variado y tan heterogéneo, existiendo un gran número de fabricantes con distintos equipos hardware y distintas filosofías de trabajo, J2ME proporciona una máquina virtual Java completamente optimizada que permitirá trabajar con diferentes tipos de procesadores y memorias comúnmente utilizados.

**Configuración y Perfil:** Como los dispositivos sobre los que trabaja J2ME son tan reducidos en lo que se refiere a potencia de cálculo y capacidad de memoria, J2ME proporciona una máquina virtual Java muy reducida que permite solo aquellas funciones esenciales y necesarias para el funcionamiento del equipo. Además, como los fabricantes diseñan distintas características en sus equipos y desarrollan continuos cambios y mejoras en sus aplicaciones estas configuraciones tan reducidas deben poder ser ampliadas con librerías adicionales. Para conseguir esto se han considerado dos conceptos extremadamente importantes que son las Configuraciones

y los Perfiles. El objetivo de realizar esta distinción entre Perfiles y Configuraciones es el de preservar una de las principales características de Java, que es la portabilidad.

### 1.2.8 *ENTORNOS INTEGRADOS DE TRABAJO Y COMPILACIÓN*

Herramientas de desarrollo y compilación para los lenguajes de programación de aplicaciones móviles.

#### 1.2.8.1 Entorno para BlackBerry

Para desarrollar aplicaciones Java para BlackBerry lo más recomendable es utilizar un entorno de desarrollo integrado. Se puede utilizar IDE generales como Eclipse y Netbeans, configurando el entorno mediante el plugin de BlackBerry, o bien utilizando un IDE específico: BlackBerry proporciona JDE (BlackBerry Java Development Enviroment). El JDE es un entorno de desarrollo integrado que permite desarrollar y simular aplicaciones Java especialmente pensadas para teléfonos BlackBerry. Gracias a BlackBerry JDE, los desarrolladores pueden crear aplicaciones con el lenguaje de programación Java ME y las API extendidas de Java para BlackBerry, El JDE está compuesto de los siguientes componentes.

- BlackBerry integrated Development Enviroment: el entorno integrado con capacidad de edición de código, gestión de proyectos, depuración y traza de programas.
- BlackBerry Smartphone simulator: Ofrece un entorno tipo Windows completo y permite simular interfaces de usuario y la interacción del usuario, conexiones de red, servicios de correo electrónico y sincronización inalámbrica de datos.
- Java ME y API de BlackBerry: Los paquetes de la especificación de Micro Edition de Java y el paquete que contiene la API específica de BlackBerry para sacarle todo el partido a los dispositivos BlackBerry.
- Aplicaciones de ejemplo: un conjunto de ejemplos de aplicaciones muy recomendables para iniciarse en el desarrollo de esta tecnología.

Además de lo anterior, el paquete JDE de BlackBerry incluye estas herramientas de soporte al desarrollo adicionales. Algunas de ellas permiten desarrollar sin necesidad de un IDE ya que son para su ejecución desde el símbolo del sistema y otras son para integrar en IDE de terceros:

- RAPC: es un compilador de línea de comando (a ejecutar desde el símbolo del sistema) para compilar archivos .java y .jar en archivos .cod. Los ficheros .cod se ejecutan en BlackBerry Smartphone Simulator o en un dispositivo BlackBerry.
- Java Loader: esta herramienta permite agregar o actualizar una aplicación en un dispositivo BlackBerry. De esta forma se pueden probar los archivos .cod de la aplicación.
- BlackBerry Signature Tool: esta utilidad permite enviar solicitudes de firma de código a BlackBerry Signing Authority Tool.
- Herramienta de verificación previa: permite comprobar parcialmente las clases que se han desarrollado antes de cargar la aplicación en un dispositivo BlackBerry.
- JDWP: facilita la depuración de aplicaciones utilizando entornos de desarrollo integrados de terceros, como Eclipse o Netbeans.

#### 1.2.8.2 Entorno Para Android

El desarrollo de aplicaciones Android se realiza con un grupo de herramientas que son suministradas en el SDK. La utilización de este grupo de herramientas puede ser de dos formas: utilizando un Entorno de Desarrollo Integrado (IDE) en combinación con un plugin llamado ADT[3] o bien desde la línea de comandos. Se puede utilizar cualquier IDE, si bien lo más común es usar Eclipse.

Si se decide prescindir de un IDE se necesita únicamente un editor de texto para escribir el código fuente e invocar las herramientas de compilación, depuración, etc., desde la línea de comandos o mediante scripts. Algunas de las herramientas más importantes de línea de comandos son:

[3]ADT: AndroidDevelopment Tools, Herramientas de Desarrollo para Android

- Android: crea y actualiza proyectos Android y crea y elimina AVD.
- Android Emulator: ejecuta aplicaciones en una plataforma Android emulada.
- Android Debug bridge: es una interfaz para conectar la aplicación con un emulador o con un móvil Android. Permite instalar aplicaciones, ejecutar comandos en línea, etc.
- Ant: permite compilar y construir proyectos generando el fichero .apk instalable de la aplicación.
- Keytool: permite generar una clave privada que se usa para firmar y autenticar el fichero .apk. Esta herramienta forma parte del JDK.
- Jarsigner: es una herramienta para firmar el fichero .apk con una clave privada generada con la herramienta Keytool. Esta herramienta también forma parte del JDK.

#### *1.2.8.2.1 Desarrollo de aplicaciones Android por etapas*

- Instalación: en esta etapa se instala el entorno de desarrollo completo incluyendo el EDI y el SDK de Android, y se crean AVD (Android Virtual Device, Dispositivos Virtuales Android).
- Desarrollo: en esta etapa se crea y desarrolla el proyecto Android, creando el código fuente de la aplicación y añadiendo todos los ficheros fuentes que pueda necesitar como imágenes o demás recursos.
- Depuración y pruebas: en esta etapa, se genera un paquete de depuración .apk que contiene el proyecto desarrollado en la etapa anterior. Este paquete se puede instalar y arrancar en cualquier emulador teléfono Android. Si se usa Eclipse, cada vez que se guarda el proyecto automáticamente se genera el .apk correspondiente. A continuación se depura la aplicación usando un depurador JDWP y las herramientas Debug del SDK Android. Eclipse proporciona su propio depurador. Por último, se comprueba el correcto funcionamiento de la aplicación usando varias herramientas del SDK como emuladores y herramientas de traza.
- Publicación: en esta última etapa se configura y se construye la aplicación para generarse una versión de entrega para distribuir entre los usuarios.

#### 1.2.8.3 Entorno para Palms OS (Web Os)

Para los antiguos entornos de desarrollo de Palm se requería el SDK y los compiladores cruzados dentro de la plataforma Linux emulado en Windows.

Para el desarrollo de sistemas en web OS se utiliza el entorno Ares, que es el primer entorno de desarrollo para móviles que funciona íntegramente dentro de un navegador. Para utilizarlo simplemente hay que registrarse en la web http://ares.palm.com y posteriormente introducir el usuario y clave con los cuales se ha registrado, no es necesario instalar nada, desde cualquier ordenador que disponga de navegador y conexión a Internet se puede hacer uso del entorno de desarrollo. La filosofía de este entorno al igual que el sistema web OS es que el futuro de la tecnología móvil, y del resto de tecnologías se va a construir dentro de la web. De esta manera, resulta muy sencillo desarrollar en la web, depurar en un dispositivo concreto conectado a la web y lanzar la aplicación desarrollada en el catálogo de aplicaciones de web OS en la web.

El sistema web OS también nos proporciona un SDK y un plugin PDk que permite desarrollar en C++ o en JavaScript aplicaciones para móviles. Puede instalarse en cualquier plataforma Windows, Linux o iOS y se integra dentro del IDE de Eclipse.

#### 1.2.8.4 Entorno para IPhone

El desarrollo de aplicaciones para iPhone se realizó a través de páginas o proyectos web, sin embargo, Apple cree la necesidad de controlar y unificar la programación de las aplicaciones para sus diferentes dispositivos. Es así como en el año 2008 decide crear el SDK (Software Development Kit, Kit de Desarrollo de Software) de Apple, un conjunto de herramientas y tecnologías de desarrollo para crear aplicaciones para iPhone e iPod Touch. Para obtener el SDK es necesario inscribirse en el iOS Dev Center (el sitio oficial de desarrolladores de Apple), desde donde se puede descargar gratuitamente. Una vez creada la aplicación decide comercializar, es necesario enviarla al App Store, lo cual requiere registrarse en el programa de desarrolladores de Apple y pagar una cuota de $99. El desarrollador

fija el precio de la aplicación y obtiene el 70% de las ventas realizadas a través del App Store. Un dispositivo Apple solo ejecuta aplicaciones firmadas por Apple, por lo que después de haber asignado el permiso de desarrollador, Apple expide un certificado que permite hacer pruebas en el iPhone, permite tres tipos de aplicaciones:

- Aplicación web: se basa en contenido web, se ejecutan en un navegador y no hacen uso de las funcionalidades del dispositivo.
- Aplicación nativa: este tipo de aplicación utiliza las funcionalidades del dispositivo e instala en el mismo, de modo que para acceder a ella no hace falta conexión/acceso a Internet.
- Aplicación híbrida: es posible desarrollar aplicaciones combinando los dos tipos anteriores, de modo que pueda instalarse en el dispositivo y además requiera de un navegador para utilizarla.

#### 1.2.8.5 Entorno para Windows Phone

Para desarrollar aplicaciones para teléfonos con S.O Windows Phone, necesita descargar el SDK que se encuentra disponible en la web oficial del App HUB: http://create.msdn.com/en-us/home/getting_started y que contiene las siguientes herramientas:

- Visual studio 2010, para windows phone. Es una de las herramientas que utiliza en el proceso de desarrollo de las aplicaciones. Esta versión es gratuita y completamente funcional. Desde la ventana principal puede definir dos tipos de proyectos diferentes, Silver light para Windows Phone 7.5 o XNA 4.0 para Windows Phone 7.5, en los dos casos utilizando el lenguaje C#, seleccionando new Project o desde el menú file > new > Project.
- Microsoft expression blend 4 para windows phone. Orientada al diseño de interfaces de usuario, esta herramienta trabaja en clave con Visual Studio 2010. Como desarrolladores se tiene control sobre el aspecto de la aplicación, pudiendo definir animaciones, transiciones o personalizando las plantillas de control, los estilos o plantillas de datos y también, puede enlazar los datos visualmente. Los proyectos que alberga en Visual Studio 2010 son accesibles

desde Expression Blend 4, por lo que es posible trabajar sobre un mismo proyecto en los dos entornos. Así por ejemplo, al abrir un archivo XAML, CS o VB puede seleccionar, a través del menú contextual, la opción edit en visual studio para editar el archivo seleccionado en Visual Studio y después de realizar los cambios guardar.

- Application Development Tool. Esta herramienta permite mostrar aplicaciones en formato XAP al emulador o a un dispositivo desbloqueado para desarrollo, con lo cual, se puede probar aplicaciones fuera de Visual Studio. Puede hacer lo mismo desde Visual Studio, realizando un clic sobre el proyecto y seleccionando la opción Deploy (Desplegar), con lo cual se instalará la aplicación en el dispositivo seleccionado en la barra de herramientas.
- Developer registration Tool. Con esta herramienta se registra un dispositivo físico Windows Phone con la cuenta de desarrollador de Market place. De esta manera, el dispositivo quedará desbloqueado para desarrollar, depurar y desplegar aplicaciones en él.
- Windows phone market place test tool. Esta herramienta permite realizar pruebas automáticas y manuales de la aplicación. Está integrada en Visual Studio 2010, basta con hacer clic con el botón derecho sobre el nombre del proyecto y escoger las opciones open Market test Kit.

#### 1.2.8.6 Entorno para Symbian

Este sistema puede programarse con variedad de entornos de desarrollo dependiendo de la tecnología de programación que necesite utilizar. Las principales tecnologías son Java Micro Edition y C++, para la primera es necesario disponer del SDK para la serie sobre la que quiera programar y luego utilizarla junto con un entorno de desarrollo como puede ser Eclipse o Netbeans, al ser J2ME una tecnología muy extendida es posible construir gran cantidad de aplicaciones sin los SDK, pero si desea utilizar el cien por cien de la funcionalidad de los dispositivos finalmente tiene que utilizarlo. Si bien instalando, por ejemplo, Netbeans con soporte para J2ME puede crear aplicaciones que se ejecutan directamente en todos los dispositivos Symbian.

A pesar de la facilidad de desarrollo de aplicaciones J2ME, para desarrollar aplicaciones realmente potentes en la plataforma Symbian hay que utilizar la tecnología basada en C++, para ello tiene que localizar el SDK para C++ de la serie para la que va a construir la aplicación. Para C++ existe un entorno basado en Eclipse y en el SDK Q llamado Carbidec ++, desarrollado expresamente por Nokia para el desarrollo de aplicaciones de todas las series.

### 1.2.9 *EMULADORES*

Los entornos de desarrollo utilizan emuladores para la simulación de las aplicaciones que desarrollan sin necesidad de utilizar un dispositivo móvil real. Estos emuladores suelen ir integrados en los entornos de desarrollo, se pueden instalar y utilizar en algunos casos independientemente.

#### 1.2.9.1 Emulador para Android

El SDK Android incluye un emulador de dispositivos móviles virtuales. El emulador puede invocar a otras aplicaciones, acceder a la red, reproducir audio y vídeo, almacenar y recuperar datos, etc., usando servicios de la plataforma Android. Además, también proporciona servicios de depuración y permite realizar prototipos de aplicaciones, desarrollar y testear aplicaciones Android sin necesidad de un dispositivo físico. El emulador proporciona una ventana en la que se visualiza en ejecución la aplicación que se está desarrollando junto con otras aplicaciones Android. El emulador utiliza configuraciones AVD (Android Virtual Device, Dispositivo Virtual para Android). Un AVD permite definir ciertas características hardware del teléfono a emular. Se pueden crear varias configuraciones para diferentes plataformas Android mediante los AVD.

Un AVD está formado por:

- Perfil hardware: define las características hardware de dispositivos virtuales. Por ejemplo, se puede definir si el dispositivo tiene cámara, si usa teclado físico QWERTY, disponibilidad de memoria, etc.
- Mapping del sistema: se puede definir la versión de la plataforma Android en el cual se ejecutará el emulador.

- Otras opciones: especifica otras características del emulador, como la dimensión de la pantalla, la apariencia, si quiere emular una tarjeta de almacenamiento SD, etc.
- Área de almacenamiento: es necesario un espacio de almacenamiento de la máquina de desarrollo para almacenar los datos del usuario del emulador (como aplicaciones que instala el usuario en el emulador) y la tarjeta SD emulada.

Eclipse proporciona con el menú Windows > AVD manager un gestor de dispositivos virtuales. Este gestor muestra una lista de dispositivos virtuales que están instalados en el entorno de desarrollo y diferentes opciones para gestionarlos (añadir nuevos, editar, borrar, etc.). Cuando se instala Eclipse y el SDK de Android, no se crean automáticamente los dispositivos virtuales. Esto tiene que hacer explícitamente, mediante el botón new, el cual muestra un cuadro de diálogo que permite configurar el emulador.

Para completar la instalación y configuración del entorno de desarrollo Eclipse es necesario crear un dispositivo virtual. Como se ha comentado se realiza mediante el botón new del menú Windows>AVD manager. En el nuevo cuadro de diálogo que aparece es necesario indicar un nombre y seleccionar el target (la API de la plataforma Android en la que se desarrolla).

Cuando se crea un dispositivo virtual o emulador el desarrollador también debe configurar algunos aspectos básicos de hardware (como LCD, tamaño de la máquina virtual en el emulador, tamaño de la RAM, etc.) y además puede añadir más características hardware al dispositivo virtual y configurarlas mediante el botón new como son pantalla sensible al tacto, teclado, GPS, batería, etc.

Es importante seleccionar el target apropiado ya que está desarrollando una aplicación con una cierta API, esta aplicación no podrá ejecutar en un dispositivo virtual que tenga un target con una versión de API inferior.


Ilustración 6: Emulador de Android,
Autores: Mercedes Guamán, Alfonso Caguana.

El emulador es proporcionado por el SDK de Android. Sin embargo, hay otros emuladores que pueden descargar independientemente y ejecutarlos, instalando aplicaciones en los mismos. En estos casos las capacidades de depuración del emulador puede disminuir o ser inexistente. Algunos de estos emuladores son Google Android Emulator. En general este tipo de emuladores no tiene demasiada presencia en el mundo de los desarrolladores.

#### 1.2.9.2 Emulador para BlackBerry

Hay varios simuladores BlackBerry disponibles para emular la funcionalidad de varios productos como dispositivos BlackBerry y BlackBerry Enterprise Server.
Los simuladores BlackBerry Enterprise Server simulan ciertos servicios (como MDS services y email) del servidor de BlackBerry (Enterprise Service), que puede ser usado en combinación con un simulador de dispositivos BlackBerry (simular contenidos web). Dentro de los simuladores para dispositivos BlackBerry se puede encontrar dos principales emuladores: BlackBerry Play Book Simulator, para desarrollos de aplicaciones para dispositivos Palm y Book, y BlackBerry Smartphone Simulator, para dispositivos móviles Smartphone. Estos simuladores permiten testear el software, pantallas, teclados, etc., con los que las aplicaciones que se desarrollan trabajarán en dispositivos reales.
Los simuladores también simulan el entorno de varias condiciones de redes wireless. El BlackBerry Smartphone Simulator puede ser descargado e instalado como un

programa .exe en una máquina Windows, suele estar integrado en el JDE de BlackBerry, con lo cual al descargar el entorno de desarrollo ya se está descargando también el simulador.

#### 1.2.9.3 Emulador para Symbian

Cada uno de los entornos de desarrollo suele llevar incorporado su propio emulador o utiliza los que proporcionan los SDK de las diversas series de Symbian. Aunque algunos lenguajes como pyS60, que es Python para la serie S60, no necesita emulador ya que este lenguaje puede ser ejecutado en cualquier plataforma, puede programarse incluso directamente en el propio dispositivo, sin necesidad de realizar compilaciones cruzadas.

Los emuladores permiten seleccionar las características del modelo a emular, de esta forma se puede observar cómo se adapta, sobre todo la interfaz gráfica a las diversas resoluciones y tamaños de pantalla de los diferentes dispositivos.

#### 1.2.9.4 Emulador para Palm Os

El antiguo emulador para Palm es monocromático, sin pantalla táctil y con muy baja resolución, los proporcionaba la empresa Palm como herramientas independientes y se disponía de una versión emulador y una versión simulador.

La primera emula la aplicación que desee depurar mientras que la segunda simula un dispositivo completo con todas las aplicaciones nativas que lleva instaladas.

Los emuladores actuales para la plataforma web OS van incluidos dentro del entorno de desarrollo SDK, permite emular diversas configuraciones y resoluciones de pantalla. Se puede ejecutar desde el propio entorno de desarrollo Ares pero es necesario tener instalado en local el SDK.

#### 1.2.9.5 Emulador para IPhone

El emulador iOS permite comprobar el correcto funcionamiento de la aplicación. Ofrece funcionalidades como: (i) rotación, cuando gira el dispositivo la aplicación rota con él; (ii) cambio de dispositivo, permite comprobar el

funcionamiento de la aplicación tanto en un iPhone como en un IPad; (iii) comprobar el funcionamiento de la aplicación para diferentes versiones del iOS; (iv) avisos de memoria, permite comprobar el funcionamiento de la aplicación dependiendo del consumo de recursos

#### 1.2.9.6 Emulador para Windows Phone

El kit de desarrollo de Windows Phone incluye un emulador que permite simular las aplicaciones en condiciones similares a las que se encuentra al ejecutarlas en el dispositivo.

El emulador se ejecuta desde inicio > programas >Windows Phone Developer tools> emulador de Windows Phone. También lo puede lanzar desde Visual Studio 2010.

El emulador permite también probar aplicaciones que utilice sensores (acelerómetro) o que dependa de la localización.

## 1.3 PATRÓN PARA EL DISEÑO DE APLICACIONES MÓVILES

Patrones de diseño o modelos de abstracción utilizados para definir y estructurar los componentes necesarios en el desarrollo de software. Los patrones para el desarrollo y creación de aplicaciones web en Java son:

### 1.3.1 *MODEL VIEW CONTROL (MVC)*

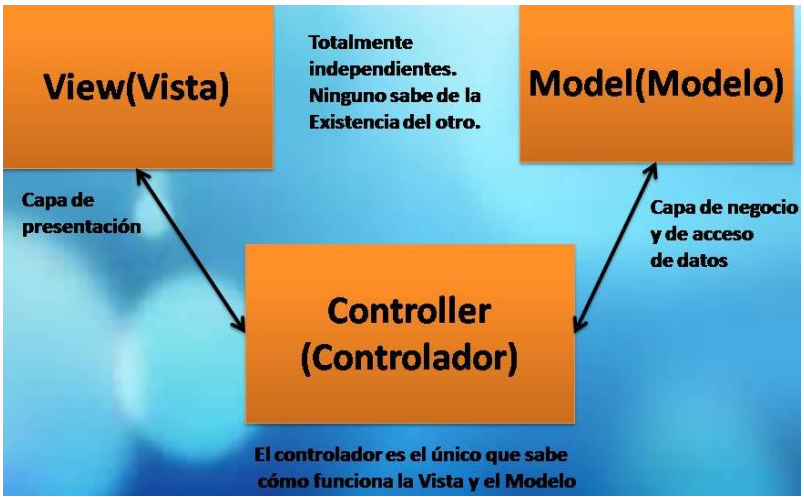



Ilustración 7: Patrón MVC,
Autores: Mercedes Guamán, Alfonso Caguana

Los componentes de MVC se definen como:

- El Modelo: Es la representación específica de la información con la cual el sistema opera, por lo tanto gestiona todos los accesos a dicha información, tanto consultas como actualizaciones, implementando también los privilegios de acceso que se haya descrito en las especificaciones de la aplicación (lógica de negocio). Envía a la 'vista' aquella parte de la información que en cada momento se solicita para que sea mostrada (típicamente a un usuario). La petición de acceso o manipulación de información llega al 'modelo' a través del 'controlador'.
- El Controlador: Responde a eventos (usualmente acciones del usuario) e invoca peticiones al 'modelo' cuando se hace alguna solicitud sobre la información (por ejemplo, editar un documento o un registro en una base de datos). También puede enviar comandos a la 'vista' asociada si solicita un cambio en la forma en que se presenta de 'modelo' (por ejemplo, desplazamiento o scroll por un documento o por los diferentes registros de una base de datos), por tanto se podría decir que el 'controlador' hace de intermediario entre la 'vista' y el 'modelo'.
- La Vista: Presenta el 'modelo' (información y lógica de negocio) en un formato adecuado para interactuar (usualmente la interfaz de usuario) por tanto requiere de dicho 'modelo' la información que debe representar como salida.

Modelo Vista Controlador, porque en este patrón de diseño se separan los datos de una aplicación, la interfaz de usuario, y la lógica de negocio en tres componentes distintos.

Cuando la lógica de negocio realiza un cambio, es necesario que ella sea la que actualiza la vista. .(Pérez F. F., 10-07-2012).

El controlador determina la vista de caso, que se llena con los modelos. El controlador entonces libera estos casos en el sistema subyacente que muestra el resultado al cliente. Se puede ver entonces que el controlador es el primero y el último en actuar. .(Pérez F. F., 10-07-2012), (Moreno, 2011).

### 1.3.2 *MODEL VIEW VIEW MODEL (MVVM)*

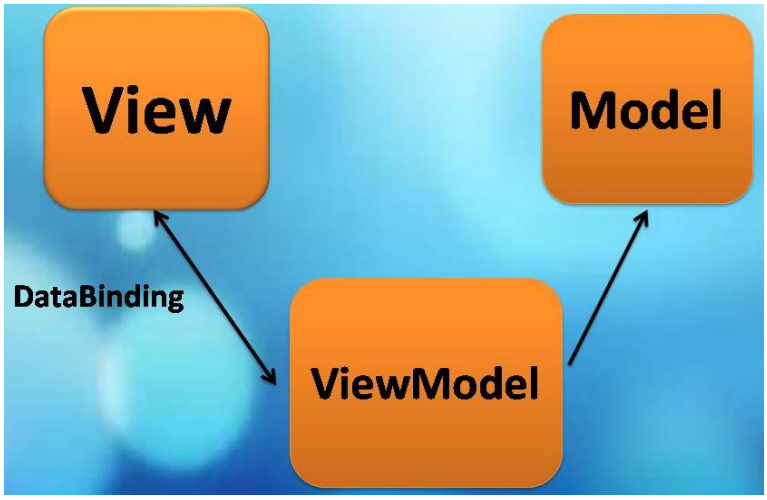



Ilustración 8: Componentes de MVVM,
Autores: Mercedes Guamán, Alfonso Caguana

Existen 3 componentes en el patrón: Model-**V**iew-**V**iewModel,

En el Modelo se adapta toda la lógica de negocio de la aplicación. El ViewModel es el modelo de la vista, o dicho de otra forma, es donde irá todo el trabajo que la vista necesita hacer, el ViewModel realiza todo el trabajo de la vista que tiene que ver con datos.

La Vista como cabe esperar será la que contendrá los datos ya procesados por el ViewModel. Explicado de otra forma que quizá quede más clara: Se tiene una aplicación para mostrar una serie de resultados que se extrae de una base de datos.

Para ello se tiene un ViewModel que extrae esos datos del modelo, y prepara esos datos para que una vista pueda usarlos. El ViewModel en sí no sabe nada acerca de las vistas, éste simplemente tiene un conjunto de propiedades para que una vista pueda hacer DataBinding a ellas. A este ViewModel se puede asignarle cualquier número de vistas que desee y dichas vistas representará esos datos como quiera.

Entonces las vistas representan los datos usando gráficas y esas graficas contendrán cosas como animaciones.

En este patrón de diseño se separan los datos de la aplicación, la interfaz de usuario pero en vez de controlar manualmente los cambios en la vista o en los datos, estos se actualizan directamente cuando sucede un cambio en ellos, por ejemplo si la vista

actualiza un dato que está presentando se actualiza el modelo automáticamente y viceversa. (Pérez F. F., 10-07-2012).

En MVVM, la interfaz de usuario (la vista), se enfrenta al usuario y toma la entrada del usuario directamente. En la vista, los comandos dentro de la ViewModel (que es el Data Context de la vista) son provocados por esta actividad. El Control de los flujos del ViewModel interpreta lo que la vista ha enviado y prepara a sus modelos. Después del control fluye de nuevo a la vista que se actualiza de acuerdo a los cambios en los modelos. Si una nueva vista requiere información, el ViewModel comunica con el Navigation Service (o cualquier otro método de navegación utiliza por la aplicación), que es de la competencia de la Vista o UI.(Pérez F. F., 10-07-2012).

## 1.4 RECONOCIMIENTO DE VOZ Y MENSAJERÍA INSTANTÁNEA

A partir de esta función básica, una línea de teléfonos inteligentes ha incluido el reconocimiento de voz para incrementar la productividad de estos dispositivos. Con esto, usted solo debe presionar un botón en el teléfono, decir el nombre de la persona que desea llamar, y el teléfono, hará el resto.

La marcación por voz, es solo un indicio de lo que el reconocimiento de voz puede hacer en los dispositivos móviles. Debido a que las computadoras portátiles encajan en la categoría de dispositivos móviles, y a medida que estas, se hacen cada vez más pequeñas, se puede predecir, la extensión de la tecnología de reconocimiento de voz a estos aparatos para mejorar su productividad y el estilo de vida.

Algunos dispositivos inteligentes de GPS, ya poseen la función de reconocimiento de voz, especialmente, los que se encuentran instalados en los vehículos, permitiéndole al conductor recibir instrucciones, pero al mismo tiempo, se puede concentrar en conducir. Solamente necesita decirle al dispositivo GPS a donde se dirige y este encontrará la ruta correcta, sin que usted toque nada.

El reconocimiento de voz consiste en convertir un flujo de palabras del lenguaje a texto.

De acuerdo con las características y funciones de los reconocedores, estos pueden clasificarse como:

- Reconocedores de propósito específico: Son aquellos cuyo vocabulario está restringido por un dominio, es decir, un conjunto o subconjunto determinado como letras, números, vocales, entre otros.
- Reconocedores de propósito general: Son aquellos cuyo dominio es general, como un idioma en particular cuyas palabras no caen en un conjunto determinado. (Peinado, 2006)

La arquitectura de un sistema de reconocimiento de voz**: C**onsta de los siguientes componentes:

- Extractor de características: Una vez que recibe la señal, el extractor de características divide la señal en una colección de segmentos, aplicándole alguna técnica de procesamiento de señales para obtener la representación de las características acústicas más distintivas de segmento. Con las características obtenidas, se construye un conjunto de vectores que constituyen la entrada al siguiente módulo.
- Clasificador probabilístico: Se crea un modelo probabilístico basado en redes neuronales, como sería el caso de los Modelos Ocultos de Markov (HMM). Una vez que se obtiene las probabilidades, realiza una búsqueda para obtener la secuencia de segmentos con mayor probabilidad de ser reconocidos.

### 1.4.1 *SISTEMAS DE SÍNTESIS DE VOZ*

La síntesis de voz es también llamada Text – To - Speech (TTS), Texto a Voz, convierte el texto en palabras audibles por medio de una voz artificial.

Es un modo de retro alimentación que puede ser usado en sustitución o como complemento de mensajes visuales, creando en el usuario mayor concentración en sus tareas al no ser distraído por las ventanas de mensajes. Por eso, la síntesis de voz ha sido empleada en sistemas diseñados para personas con capacidades diferentes, facilitándoles su uso.

La síntesis de voz es el proceso mediante el cual se busca que una aplicación móvil genere voz, o simule el habla humana, a partir de un texto. Es la producción artificial de expresiones vocales por parte de la máquina, sin que estas sean

pregrabadas. La síntesis de voz busca ser útil para situaciones en las cuales los ojos deben estar enfocados en otras cosas, como al conducir un automóvil, o en medio de cierto procedimiento científico en el cual se dificulta colocar la atención en el dispositivo móvil para leer un resultado. Así mismo se busca que sea útil a personas con discapacidades que les impidan leer correctamente (Llisterri).

**Sintetizadores por concatenación de forma de onda:** Tienen una base de datos con unidades, o fonemas, pregrabados, de modo que después del correcto análisis concatena las unidades adecuadas para generar nuevas frases. Su grado de complejidad es alto y se limita a las características de un solo hablante (aquel que realiza las grabaciones), pero mejora la calidad y se tienen muy buenos resultados, siendo el método más usado en la actualidad para diversas aplicaciones. Basado en bloques de voz pregrabada de forma tal, que dependiendo del sistema, se unan adecuadamente para llegar a formar las frases o palabras que se ingresaron por texto.

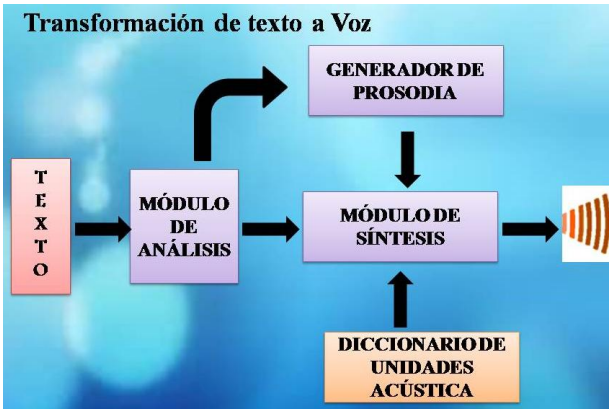



Ilustración 9: Sintetizador de Voz,
Autores: Mercedes Guamán, Alfonso Caguana

Para realizar este proceso, los sintetizadores de voz cuentan con diferentes módulos que los componen y son los siguientes:

**Diccionario de unidades acústicas:** Hace referencia a la base de datos con todos los bloques pregrabados que hagan parte del sistema de síntesis de voz. (Loeber, 2011)

**Módulo de análisis lingüístico**: Este módulo se encarga de procesar las abreviaturas, símbolos, números, etc. de modo que se llegue a un estándar en el texto. Además realiza la función de silabificación, descomponiendo cada palabra en sus respectivas silabas o bloques básicos, basado en reglas de posicionamiento relativo de vocales y consonantes. Otro proceso que lleva a cabo

en este módulo es el análisis de acentuación, representando la acentuación de cada palabra de manera textual, para saber que silaba es la acentuada y asignar el difonema correcto a cada parte de la palabra. Finalmente el último proceso se encarga de la transcripción fonética, es decir, pasar cada representación ortográfica del texto a la cadena de fonemas correspondientes, realizando un análisis letra por letra. Los análisis realizados en esta etapa alimentaran las siguientes a los otros dos módulos. Siempre se debe tener en cuenta que los procesos realizados en este módulo dependen de cada idioma, pues la forma de tratar tanto la prosodia como la estructura de frases, palabras, silabas y difonemas varía de idioma a idioma. Al módulo de síntesis se envía la cadena que se ha creado de fonemas a partir del texto, mientras que al generador de prosodia se le envía el tipo de frase que es para darle la correcta entonación. Este tipo se extrae del mismo texto, basado en la puntuación. Dependiendo del signo de puntuación que finalice cada frase se determina si es una frase enunciativa inacabada (",",";"), acabada ("."), exclamativa ("!") o interrogativa ("?"). (Loeber, 2011)

**Generador de Prosodia**: Se encarga de generar la plantilla de prosodia adecuada que permita al sintetizador generar voz con una buena entonación, de modo que se tenga una emisión natural de la voz generada. Además influye en la forma en que se entiende el mensaje, pues el significado se puede modificar, aunque se usen las mismas palabras. Dependiendo del tipo de información que posee cada frase se le asigna un patrón melódico, es decir, una onda especifica que permite el estudio de la entonación. (Loeber, 2011)

**Módulo de síntesis**: En el módulo de síntesis se toma toda la información de bloques, de prosodia y tomando de la base de datos los bloques adecuados, sintetiza la onda que se emite. (Loeber, 2011)

### 1.4.2 *MENSAJERÍA INSTANTÁNEA*

SMS consiste en el intercambio de mensajes únicamente de texto sin formato (es decir, no se pueden seleccionar diferentes fuentes, tamaños y estilos) entre teléfonos móviles garantizando al usuario el correcto envío de la información, aunque no posibilita el envío de mensajes en tiempo real. SMS permite típicamente

un máximo de 160 caracteres por mensaje (70 de utilizar otro alfabeto distinto del latino, como el chino o el árabe).

Existen varios terminales en el mercado que permiten enviar varios mensajes concatenados, pudiendo escribir textos más largos de hasta 480 caracteres. No obstante, esta funcionalidad únicamente es implementada en el terminal; es decir, la persona que envía un mensaje concatenado en realidad está enviando varios SMS y la red desconoce si los mensajes que procesa son concatenados o no.

Después, es el terminal destinatario el que tiene capacidad para identificarlos y re-ensamblarlos según su orden. También es posible enviar un SMS a un terminal móvil mediante un PC con e-mail y conexión a Internet, aunque el tamaño máximo de los mensajes suele poseer alrededor de 120 caracteres. Los mensajes SMS viajan a través del canal de señalización de GSM, lo cual permite a la operadora sacar partido a un recurso infrautilizado.

La mensajería instantánea inalámbrica o WIM[4] (Wireless Instant Messaging) permite enviar mensajes de texto en tiempo real; es decir, posibilita que el destinatario reciba la información al mismo tiempo que el emisor la produce. Esta plataforma, promovida principalmente por Open wave, presenta en la actualidad varios problemas de interoperabilidad entre distintos suministradores y aún no existe ninguna solución comercialmente disponible; pero la gran demanda de mensajería instantánea en las empresas, augura su pronta introducción en el ámbito inalámbrico. Una vez que la tecnología WIM esté totalmente desarrollada, permitirá también incluir la presencia, o la habilidad para que un usuario conozca si otros están en ese mismo momento conectados a la misma red; y posibilitará la creación, gestión y comunicación en tiempo real con grupos de usuarios.

Entre el SMS y el MMS, encontramos una tecnología de transición denominada servicio de mensajería mejorado o EMS (Enhaced Messaging Service), que es parte de las especificaciones del 3GPP (Third Generation Partnership Project). Los mensajes EMS pueden contener sonidos, imágenes, animaciones y texto con formato; soportando también mensajes concatenados, es decir, los mensajes ya no están restringidos a 160 caracteres como ocurre en SMS.

[4]WIM: mensajería instantánea inalámbrica

Los elementos del mensaje EMS, tales como las imágenes, son ubicados en las cabeceras del mensaje SMS; por ello, los teléfonos que sólo soportan SMS pueden mostrar la parte de texto del mensaje, incluso si no pueden visualizar la parte EMS.

## 1.5 SEGURIDAD EN LAS ACTIVIDADES DE LOS USUARIOS DE TELEFONÍA MÓVIL

La seguridad en las actividades de los usuarios de telefonía móvil es una de las mayores preocupaciones tanto para entornos móviles como no móviles.

Una solución de seguridad para aplicaciones móviles debe tratar todo el trayecto móvil, es decir desde los sistemas back-end hasta el dispositivo del cliente. Bajo este concepto puede decirse que la seguridad en una aplicación móvil tiene tres preocupaciones principales:

**Privacidad de la mensajería**. Asegurar que los datos sensibles no estén comprometidos durante su transmisión. Esto comúnmente es realizado con el uso de criptografía.

**Autentificación**. Asegurar que la identidad de todos los usuarios involucrados en la comunicación es correcta. Usualmente alcanzado con el uso de certificados digitales o técnicas de usuario / password

**Seguridad en el dispositivo**. Asegurar la protección de los datos almacenados en el dispositivo móvil, usualmente logrado a través del uso de password de protección y encriptación.

Según la arquitectura de la aplicación móvil, se dispone de distintos mecanismos de seguridad ya provistos por sistemas comerciales, a los cuales se puede adherir. (Garnacho, 2004)

Sistemas operativos como Palm OS ofrecen una serie de servicios que pueden ser incorporados a las aplicaciones y políticas de seguridad. A modo de ejemplo se listan algunos de ellos:

- **System-level authorization and authentication**. Permite especificar una serie de reglas para permitir el acceso de datos almacenados en el dispositivo. Estas incluyen el uso de password, PINs, y una variedad de opciones biométricas.

- **System-wide encryption**. Permite la utilización de algoritmos de encriptación actualmente considerados más seguros como RC4 y SHA-1.
- **Secure Sockets Layer (SSL) Encryption**. Incorpora SSL 3.0 para proveer protección "end-to-end" en las comunicaciones basadas en Internet.
- **Support for Signed Code**. Utiliza un esquema de firma digital para permitir el acceso a datos o recursos importantes.
- **Unique Device Identification**. Permite autenticar dispositivos a través de sus Flash ID, MAN (Mobile Access Number) y ESN (Electronic Serial Number).

## 1.6 EFICIENCIA QUE BRINDAN LAS APLICACIONES MÓVILES A LOS CLIENTES DE TELÉFONO MÓVIL

La eficiencia a los dispositivos móviles se asemeja más a los servidores que a computadoras de escritorio. A continuación se analizaran algunos de ellos.

### 1.6.1 *ALWAYS RUNNING*

Los dispositivos móviles y sus aplicaciones, comúnmente se encuentran activos las 24 horas del día, 7 días a la semana. Teléfonos celulares y PDAs son un ejemplo de esto, los mismos siempre se encuentran activos o en algún modo standby que les asegura estar disponibles casi instantáneamente cuando se los requiera. A pesar de que se ha incrementado el hecho de dejar las computadoras de escritorio encendidas por más tiempo, es normal que los usuarios reinicien las mismas, inicien y cierren sesiones, abran y cierren aplicaciones en reiteradas oportunidades en la misma sesión. En contraste las aplicaciones móviles con el objetivo de mantener un acceso instantáneo, se encuentran corriendo en back ground (Garnacho, 2004).

Por estas razones, los dispositivos móviles se asemejan a los servidores en el hecho que los mismos necesitan estar listo para proveer un servicio instantáneo a sus clientes.

### 1.6.2 *MANEJO DE FALLAS*

Las aplicaciones ejecutando en un dispositivo móvil, operan en un entorno altamente demandante, donde son comunes las interrupciones en la comunicación, la disminución del ancho de banda, la capacidad limitada de las batería, entre otros. El mismo sistema operativo puede detener una aplicación corriendo en background si

los recursos comienzan a disminuir. Y peor aún, el dispositivo puede ser extraviado, robado o sufrir cualquier tipo de daño.
Ante cualquiera de estas circunstancias, el usuario necesita tener la seguridad que la integridad de los datos residentes en el dispositivo móvil no se verá afectada.
Por estas razones, las aplicaciones móviles, como los servidores de misión crítica, deben asegurar que los datos importantes para el usuario y su administración puedan superar la ocurrencia de daños inesperados o fallas.

### 1.6.3 *USO DE LA MEMORIA*

Una computadora de escritorio tiene comúnmente asignada un gran espacio en disco que le permita bajar allí memoria que no está siendo usada en forma de archivo. Este archivo es comúnmente llamado Archivo de Intercambio (swap file). Si nueva memoria es requerida por una aplicación porque se está iniciando o porque tiene la necesidad de memoria adicional, y el sistema comienza a quedarse sin memoria física, el sistema operativo tomará secciones de memoria que no están siendo usadas y las enviará al archivo de intercambio. De esta manera una computadora de escritorio podrá funcionar como si tuviera mucha más cantidad de memoria RAM que realmente tiene. Esto permite a los usuarios correr varias aplicaciones simultáneamente.
En los servidores se tiende a no usar esta estrategia porque los mismos fueron diseñados para un máximo rendimiento. En un servidor se quiere mantener todo en la memoria física donde el acceso es mucho más veloz. En los dispositivos móviles comúnmente no se cuenta con grandes discos como para alojar un Archivo de Intercambio.
Disponer de esta capacidad en un dispositivo móvil puede ser prohibitivo por costos, tamaño, velocidad y consumo de energía. Nuevamente, como los servidores, los sistemas operativos y las aplicaciones de un dispositivo móvil, comúnmente no utilizan la técnica de archivos de intercambio.

### 1.6.4 *SERVICIOS CRÍTICOS*

Varios dispositivos móviles sirven a otro propósito crítico mientras están en ejecución aplicaciones prioritarias. Si un teléfono móvil deja de funcionar como teléfono móvil porque una aplicación colapsó, se vuelve drásticamente lento en su

respuesta, bloquea la interface del usuario, o presenta cualquier otro mal comportamiento, el usuario final no se encontrará muy conforme. La mayoría de los sistemas operativos móviles posee niveles de protección para salvaguardar las funciones críticas, pero de todas formas una aplicación móvil debe ser capaz de realizar que el dispositivo siga siendo útil bajo cualquier circunstancia.

Como los servidores, los dispositivos móviles necesitan manejar una serie de servicios críticos que deben estar disponibles para el usuario en todo momento.

# CAPITULO II

# METODOLOGÍA

## 2.1 DESCRIPCIÓN DE LA METODOLOGÍA

La metodología es el estudio de los procedimientos en la adquisición o exposición del conocimiento científico, conjuntos de procedimientos de investigación aplicables en alguna ciencia.

La metodología comprende de un conjunto de reglas que surgen del estudio de una disciplina científica.

En el presente trabajo de investigación se ha aplicado los siguientes métodos:

**Método Inductivo- Deductivo.-** Se lleva a cabo una etapa de observación y registro de los hechos. A continuación se procede al análisis de lo observado.

**Método Experimental.-** Análisis de las diversas herramientas de software utilizadas para el modelado de datos, diseño de datos, diseño de interfaz, programación móvil a utilizar en el proyecto**.**

**Método Bibliográfico.-** Se determina las fuentes más importantes que proporcionen información y documentación como: código fuente, libros, scripts, etc.

## 2.2 TIPO DE ESTUDIO

El tipo de estudio se elige de acuerdo a la función del objetivo que se pretende alcanzar, el caso es analizar la arquitectura de las aplicaciones móviles basadas en el entorno de programación Java y su aporte con la seguridad y eficiencia en las actividades diarias de los usuarios de telefonía móvil.

**Según el objeto de estudio:** Es una investigación aplicada porque está orientada a solucionar problemas reales en este caso como, la arquitectura de las aplicaciones móviles basadas en el entorno de programación Java aporta positivamente con la seguridad y eficiencia en las actividades diarias de los usuarios de telefonía móvil.

**Según la fuente de información:** Es una investigación bibliográfica porque es el proceso de búsqueda de información sobre la arquitectura de las aplicaciones

móviles basadas en el entorno de programación Java que aporta positivamente con la seguridad y eficiencia en las actividades diarias de los usuarios de telefonía móvil mediante libros, revistas, informes con relación al tema mencionado.

**Según las variables:** Es una investigación relacional por la valoración y medición de la Operacionalización de las variables según el estudio de la investigación.

## 2.3 POBLACIÓN Y MUESTRA

Para el presente estudio se toma en cuenta como población a los estudiantes de la escuela de conducción de la Universidad Nacional de Chimborazo, que se encuentran matriculados en el año 2013 que son 420 estudiantes.

**Tamaño de la muestra:** Para la realización del estudio se tomó en cuenta la opinión de los estudiantes de la escuela de conducción de la Universidad Nacional de Chimborazo.

La muestra estará constituida por 92 estudiantes, que están cursando el curso de conducción para choferes profesionales y contribuir positivamente en la sociedad.

El cálculo de la muestra se realiza aplicando la siguiente fórmula:

$$n = \frac{N\sigma^2 Z^2}{(N-1)e^2 + \sigma^2 Z^2}$$

Donde

n = tamaño de la muestra

N = tamaño de la población =420

o = desviación estándar valor constante de 0,5

Z = valor obtenido mediante el nivel de confianza 95%= 1,96

e = limite aceptable de errores muestra en este caso es 5% = 0,05

El tamaño de la población es 420 estudiantes de la escuela de conducción de la UNACH

Tamaño de muestra: 92 estudiantes es el tamaño de muestra.

## 2.4 HIPÓTESIS

### 2.4.1 *HIPÓTESIS DE INVESTIGACIÓN*

La arquitectura de las aplicaciones móviles basadas en el entorno de programación Java aporta con la eficiencia en el desarrollo y seguridad para el usuario móvil.

### 2.4.2 *HIPÓTESIS NULA*

La arquitectura de las aplicaciones móviles basadas en el entorno de programación Java no aporta con la eficiencia en el desarrollo y seguridad para el usuario móvil.

## 2.5 IDENTIFICACIÓN DE VARIABLES

La arquitectura de las aplicaciones móviles basadas en el entorno de programación Java aporta con la eficiencia en el desarrollo y seguridad para el usuario móvil.

### 2.5.1 *VARIABLE INDEPENDIENTE*

La arquitectura de las aplicaciones móviles basadas en el entorno de programación Java.

### 2.5.2 *VARIABLE DEPENDIENTE 1*

Eficiencia en el desarrollo de la aplicación móvil.

### 2.5.3 *VARIABLE DEPENDIENTE 2*

Seguridad en las actividades diarias de los usuarios de telefonía móvil.

## 2.6 OPERACIONALIZACIÓN DE VARIABLES

Tabla 3: Operacionalización de Variables, Autores: Mercedes Guamán, Alfonso Caguana.

| **Variable** | **Tipo** | **Definición Conceptual** | **Dimensión** | **Indicadores** |
|---|---|---|---|---|
| Arquitectura de las aplicaciones móviles basadas en el entorno de programación java. | Independiente | Esquema de los programas diseñados para funcionar en teléfonos inteligentes y unidades portátiles. Normalmente utilizan internet para funcionar e intercambiar datos. Algunas vienen preinstaladas en los dispositivos y otras se adquieren vía internet a través de tiendas. | Estándares básicos para el lenguaje de programación java. Componentes reusables y colaborativos. | Tiempo de Aprendizaje. Complejidad de Implementación. Reusabilidad del código. |
| Eficiencia en el desarrollo de la aplicación móvil, | Dependiente 1 | El KVM (Kilo Virtual Machine), proporciona el soporte necesario para la creación aplicaciones móviles permitiendo mejorar y facilitar el envío de SMS mediante comandos de voz. | Tiempo. Propiedades en MB. Número de líneas de código. | Velocidad de la Ejecución. Tamaño de memoria. Líneas de código. |
| Seguridad para el usuario móvil. | Dependiente 2 | El servicio de mensajes cortos o SMS (Short Message Service) es un servicio disponible en los teléfonos móviles que permite el envío de mensajes entre teléfonos. El SMS servía para teléfonos fijos y otros dispositivos de mano. SMS fue diseñado originariamente como parte del estándar de telefonía móvil digital GSM, pero en la actualidad está disponible en una amplia variedad de redes, incluyendo las redes 4G | Tiempo. Encuesta a 10 conductores. Número de procesos. | Envío y recepción de SMS. Traducción exacta de voz a texto y viceversa. Proceso de responde un mensaje. |

## 2.7 VALORIZACIÓN DE LAS VARIABLES DE ACUERDO A SUS INDICADORES.

### 2.7.1 *EFICIENCIA EN EL DESARROLLO DE LA APLICACIÓN MÓVIL*

El indicador1 es la velocidad de ejecución.

Tabla 4: Velocidad de ejecución,
Autores: Mercedes Guamán, Alfonso Caguana

| Tiempo en Segundos | Valores |
|---|---|
| 5 a 6 | 1 |
| 4 a 5 | 2 |
| 1 a 3 | 3 |

- Si la velocidad de ejecución demora de 5 a 6 segundos es lento y le damos un valor de 1 que es el valor mínimo.
- Si la velocidad de ejecución demora de 4 a 5 segundos es medio y le damos un valor de 2 que es el valor intermedio.
- Si la velocidad de ejecución demora de 1 a 3 segundos es lento y le damos un valor de 1 que es el valor mínimo

El indicador2 es el tamaño de memoria.

Tabla 5: Tamaño de memoria,
Autores: Mercedes Guamán, Alfonso Caguana

| MB | Volares |
|---|---|
| 8 a 10 | 1 |
| 6 a 8 | 2 |
| 4 a 6 | 3 |

- Si el tamaño de memoria es de 8 a 10 MB es demasiada memoria y espacio de disco y le damos un valor de 1 que es el valor mínimo.
- Si el tamaño de memoria es de 6 a 8 MB es una memoria y espacio de disco medio y le damos un valor de 2 que es el valor medio.
- Si el tamaño de memoria es de 4 a 6 MB es pequeña memoria y espacio de disco excelente y le damos un valor de 3 que es el valor máximo.

El indicador3 es las líneas de código

Tabla 6: Líneas de código,

Autores: Mercedes Guamán, Alfonso Caguana

| Número de líneas de código | Valores |
|---|---|
| 700 a 1000 | 1 |
| 400 a 600 | 2 |
| 200 a 300 | 3 |

Estos valores son de acuerdo al número de líneas de código que se generan en los elementos de View o Layout con respecto a las arquitecturas MVC y MVVM .

- Si el número de código de líneas es entre 200 a 300 le damos un valor de 3 que es el valor máximo.
- Si el número de código de líneas es entre 400 a 600 le damos un valor de 2 que es el valor medio.
- Si el número de código de líneas es entre 700 a 1000 le damos un valor de 1 que es el valor mínimo.

### 2.7.2 *SEGURIDAD PARA EL USUARIO DE TELEFONÍA MÓVIL*

El indicador1 es el envío y recepción de SMS

Tabla 7: Envío y recepción de SMS,

Autores: Mercedes Guamán, Alfonso Caguana

| Tiempo en segundos | Valores |
|---|---|
| 5 a 6 | 1 |
| 3 a 4 | 2 |
| 1 a 3 | 3 |

- Si al enviar y recibir mansajes de texto SMS se demora de 5 a 6 segundos le damos un valor de 1 que es el valor mínimo ya es que un tiempo muy demoroso.
- Si al enviar y recibir mansajes de texto SMS se demora de 3 a 4 segundos le damos un valor de 2 que es el valor medio.
- Si al enviar y recibir mansajes de texto SMS se demora de 1 a 3 segundos le damos un valor de 3 que es el valor máximo.

El indicador 2 es la traducción exacta de voz a texto y viceversa

Tabla 8: Traducción exacta de voz a texto y viceversa,
Autores: Mercedes Guamán, Alfonso Caguana

| Encuestas a 10 personas que su respuesta sea si | Valores |
|---|---|
| 2 a 4 | 1 |
| 5 a 7 | 2 |
| 8 a 10 | 3 |

- Si al realizar la encuesta a 10 conductores con respecto a la traducción exacta de voz a texto y viceversa en la app HablaSMS sus resultados de la cantidad de SI, al contestar de 2 a 4 conductores SI se da un valor de 1 ya que no es satisfactorio y es el valor mínimo.
- Si al realizar la encuesta a 10 conductores con respecto a la traducción exacta de voz a texto y viceversa en la app HablaSMS sus resultados de la cantidad de SI, al contestar de 5 a 7 conductores SI se da un valor de 2 ya que es un valor medio.
- Si al realizar la encuesta a 10 conductores con respecto a la traducción exacta de voz a texto y viceversa en la app HablaSMS sus resultados de la cantidad de SI, al contestar de 8 a104 conductores SI se da un valor de 3 ya que es satisfactorio y es el valor máximo.

El indicador 3 es el proceso de responder un SMS

Tabla 9: Proceso de responder un SMS,
Autores: Mercedes Guamán, Alfonso Caguana

| Número de procesos | Valores |
|---|---|
| 7 a 8 | 1 |
| 5 a 6 | 2 |
| 2 a 4 | 3 |

- Si la cantidad de procesos que debe realizar el conductor para responder un mensaje SMS es de 7 a 8 procesos se da un valor de 1 ya que es un valor mínimo.

- Si la cantidad de procesos que debe realizar el conductor para responder un mensaje SMS es de 5 a 6 procesos se da un valor de 2 ya que es un valor medio.
- Si la cantidad de procesos que debe realizar el conductor para responder un mensaje SMS es de 2 a 4 procesos se da un valor de 3 ya que es un valor máximo.
- Lectura, escritura, agregar destinatario, enviar

**Tabla 10: Valores máximos y mínimos**

| | |
|---|---|
| valor máximo | 3 |
| valor mínimo | 1 |
| margen de error | 0,05 |

## 2.8 PROCEDIMIENTO

Para implementar la arquitectura de las aplicaciones móviles basadas en el entorno de programación Java y su aporte con la seguridad y eficiencia en las actividades de los usuarios de telefonía móvil; caso práctico desarrollo de un software de transformación de mensajes de texto SMS a mensajes de voz y viceversa necesita las siguientes herramientas:

### 2.8.1 *HARDWARE*

- Procesador Intel(R) Core (TM) i3 CPU M 370
- Sistema operativo de 64 bits
- Memoria RAM 4GB
- Teléfono celular con sistema operativo Android.

### 2.8.2 *SOFTWARE*

- jdk-7u4-windows-x64
- jdk-7u4-windows-i586
- installer_r20.0.3-windows
- eclipse-jee-juno-SR1-win32-x86_64

### 2.8.3 *PROCESO Y CONFIGURACIÓN DE JAVA DEVELOPMENT KIT (JDK)*

**Paso 1:** Debe descargar el JDK de acuerdo a las características de hardware y software, en el caso se descargó para el sistema operativo Windows x64 Windows 64 bits.

Al instalarlo debe estar dentro del disco duro C:/Program Files/Java debe instalar normalmente como cualquier otro programa siguiente, siguiente, siguiente.

**Paso 2 configurar entorno:** Este paso la variable de entorno sirve únicamente para que desde la consola de Windows pueda compilar y ejecutar archivos java.

Para agregar una variable de entorno dirija a:

Windows Seven >Panel de control > Sistema > Configuración avanzada del sistema > Variables de entorno.

En este paso se crear una nueva variable de entorno presionando en el botón "Nueva", como se muestra en la imagen aparece una nueva cuadro de dialogo donde especificamos el nombre de la variable y el valor de la variable. El nombre recomendado es "JAVA_HOME". Al presionar el botón de aceptar se observa la nueva variable dentro de la lista de "Variables del Sistema" tal cuál figura en la imagen.

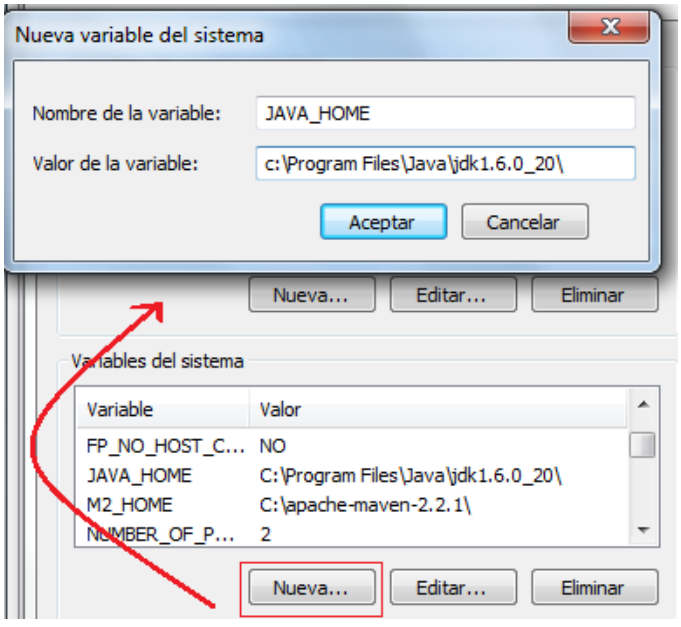



Ilustración 10: Nueva Variable del entorno,

Autores: Mercedes Guamán, Alfonso Caguana

Por último se modifica la variable del sistema ya existente "Path", para ello seleccionar la variable de la lista de variables y presionar el botón "Editar". Esto visualiza una ventana emergente donde puede ver el valor de la variable y editarlo.
Al final del toda la variable, agregar lo siguiente "**;%JAVA_HOME%bin;**", siendo "JAVA_HOME" el nombre de la variable que se agregó anteriormente. Presionar aceptar en ambas pantallas y finalizar.

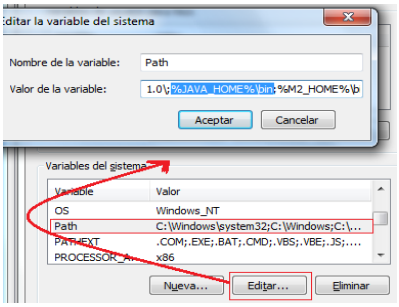



Ilustración 11: Editar la variable existente,
Autores: Mercedes Guamán, Alfonso Caguana

Para comprobar que la variable de entorno ha sido creada satisfactoriamente realizar lo siguiente tarea:
Abrir la consola DOS de Windows para verificar que se ha realizado los cambios correctos. Presionar **Tecla Windows + R** escribir "**cmd**" y presionar **enter**

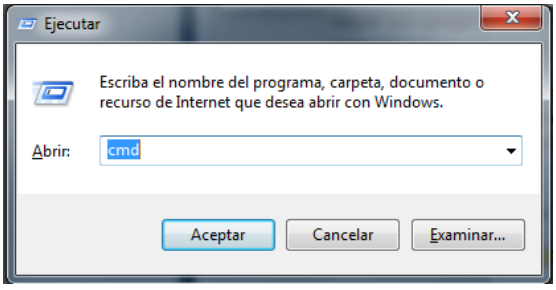



Ilustración 12: Pantalla para ingresar a CMD,
Autores: Mercedes Guamán, Alfonso Caguana

Digitar

javac>enter

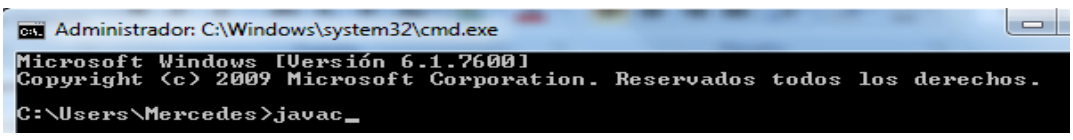


Ilustración 13: Pantalla cmd comprobación1,

Autores: Mercedes Guamán, Alfonso Caguana

Si esta correcto las modificaciones de las variables, aparece la siguiente información.

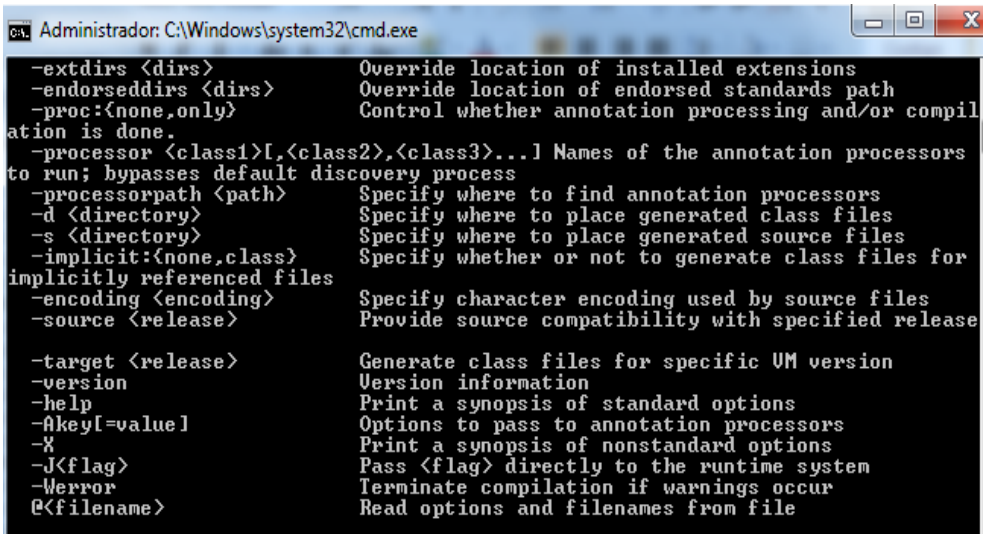



Ilustración 14: Pantalla comprobación2,

Autores: Mercedes Guamán, Alfonso Caguana

#### 2.8.3.1 Instalación y Configuración Eclipse para desarrollar aplicaciones Android

**Paso 1: Instalar Eclipse**

El entorno Eclipse se puede descargar desde el sitio web de descargas de Eclipse. La versión Eclipse IDE for Java Developers es una buena opción. La instalación consiste en descomprimir el fichero .zip descargado. Esto crea una carpeta llamada eclipse dentro de la cual hay, entre otros, el programa Eclipse, el cual inicia el entorno.

**Paso 2: Instalar el SDK Android.**

El paquete SDK se puede descargar en un fichero .zip o en una versión de instalador (solo para Windows). Si se descarga un fichero .zip, la instalación consiste únicamente en descomprimirlo (se descomprime automáticamente en el directorio android-sdk-<machine-platform>, donde <machine-platform> es el nombre de la

plataforma donde se instala). Si se descarga la versión de instalador para Windows únicamente se debe ejecutar el programa y seguir las indicaciones del mismo.

**Paso 3: Instalar el Plugin ADT para Eclipse**

Para instalar el plugin ADT se debe a realizar las siguientes acciones:

- Iniciar Eclipse y seleccionar del menú help la opción de instalación de nuevo software (menú help>install New software...).
- Hacer clic en el botón Add… y aparece un cuadro de diálogo en el que solicita introducir el nombre del repositorio donde está el plugin (introducir aDtplugin) y la URL de localización (introducir https://dl-ssl.google.com/android/eclipse/).
- El paso anterior origina que en el cuadro de diálogo aparezca una lista de software instalable del repositorio. En concreto solo aparece un paquete. Se selecciona el único que aparece haciendo clic en la correspondiente check box (Developer tools) y con el botón next avanza al siguiente cuadro de diálogo.
- En la nueva ventana se muestra una lista de herramientas que van a ser descargadas. Hacer clic en el botón next y a continuación aceptar los acuerdos de licencia. Por último hacer clic en el botón finish finalizando el proceso de instalación. Una vez que la instalación ha sido completada es necesario restaurar Eclipse para que la nueva instalación tenga efecto.

Si ya tiene instalado el plugin ADT conviene realizar actualizaciones cada cierto tiempo de las versiones más recientes. Para ello se utiliza el gestor de actualizaciones de Eclipse (Update Manager, accesible mediante el menú help>check for Update).

**Paso 4: Configurar el plugin ADT**

En este paso se configuran algunas preferencias del plugin. En concreto se debe asociar el plugin con el SDK de Android.

Para ello seleccionar el menú eclipse > preferences y en el panel de la izquierda seleccionar android. A continuación, en el panel principal se indica mediante el

botón browse la localización del SDK que fue descargado en su momento. Por último, hacer clic en apply y oK.

**Paso 5: Añadir Componentes**

El SDK de Android está organizado en varias partes: versiones de plataformas Android, herramientas, ejemplos y documentación, las cuales se pueden instalar separadamente. El paquete SDK instalado que ha sido descargado incluye solamente las herramientas. Para desarrollar una aplicación Android necesita descargar e instalar la plataforma Android.

Este proceso se realiza mediante el gestor AVD (Android Virtual Device, Dispositivo Virtual Android). El AVD se puede ejecutar de varias formas. Una de ellas es desde Eclipse con el menú Windows>Android SDK manager.

Este menú proporciona la ventana del gestor de AVD y muestra todos los componentes del SDK que puede descargar e instalar en el SDK local de la máquina en la que se trabaja. ¿Qué componentes son los que deben descargar? Los componentes básicos son SDK Tools, SDK Platform Tools y SDK platform.

## 2.9 PROCESAMIENTO Y ANÁLISIS

### 2.9.1 *TAMAÑO DE LA MUESTRA*

Una vez concluidas las etapas de colección y procesamiento de datos se inicia con el análisis de datos. Para obtener el tamaño de muestra se utiliza la siguiente fórmula:

$$n = \frac{N\sigma^2 Z^2}{(N-1)e^2 + \sigma^2 Z^2}$$

Dónde

n = tamaño de la muestra

N = tamaño de la población

o = desviación estándar valor constante de 0,5

Z = valor obtenido mediante el nivel de confianza 95%=1,96

e = limite aceptable de errores muestra en este caso es 9%=0,05

El tamaño de la población es 420 estudiantes de la escuela de conducción de la UNACH

n = 92 estudiantes

### 2.9.2 *VALORIZACIÓN DE LAS VARIABLES:*

#### 2.9.2.1 Seguridad en las actividades diarias de los usuarios de telefonía móvil con la utilización de la aplicación HablaSMS.

Tabla 11: Aplicaciones móviles para android,Autores: Mercedes Guamán, Alfonso Caguana

| | HablaSMS |
|---|---|
| Envío y recepción de SMS | 2 |
| Traducción exácta de voz a texto y viceversa | 3 |
| Proceso de responder un SMS | 3 |
| Total | 8 |
| Media | 2,66666667 |
| Porcentaje | 89% |

##### *2.9.2.1.1 Cálculo de la prueba de hipótesis chi-cuadrado con respecto a la seguridad que brinda la aplicación móvil a los clientes.*

Tabla 12: Frecuencias Obtenidas y Esperadas,
Autor: Mercedes Guamán

| | Frecuencias Obtenidas | Frecuencias Esperadas | Fórmula |
|---|---|---|---|
| Envío y recepción de SMS | 2 | 2,95 | 0,3059322 |
| Traducción exácta de voz a texto y viceversa | 3 | 2,95 | 0,00084746 |
| Proceso de responder un SMS | 3 | 2,95 | 0,00084746 |
| Total de Chi-Cuadrado | | | 0,30762712 |

#### 2.9.2.2 Eficiencia en el desarrollo de la aplicación móvil HablaSMS

Tabla 13: Eficiencia que aporta a los clientes de telefonía móvil, Autores: Mercedes Guamán, Alfonso Caguana

| | MVC | MVVM |
|---|---|---|
| Velocidad de ejecución | 1 | 2 |
| Tamaño de Memoria | 2 | 3 |
| Líneas de código | 2 | 3 |
| Total | 5 | 8 |
| Media | 1,66666667 | 2,66666667 |
| Porcentaje | 56% | 89% |

##### *2.9.2.2.1 Cálculo de la prueba de hipótesis chi-cuadrado con respecto a la eficiencia en el desarrollo de la aplicación HablaSMS*

Tabla 14: Frecuencias Obtenidas y Esperadas en VMMV,
Autor: Mercedes Guamán

| | Frecuencias Obtenidas | Frecuencias Esperadas | fórmula |
|---|---|---|---|
| Velocidad de ejecución | 2 | 2,95 | 0,3059322 |
| Tamaño de Memoria | 3 | 2,95 | 0,00084746 |
| Líneas de código | 3 | 2,95 | 0,00084746 |
| Total de Chi-Cuadrado | | | 0,30762712 |

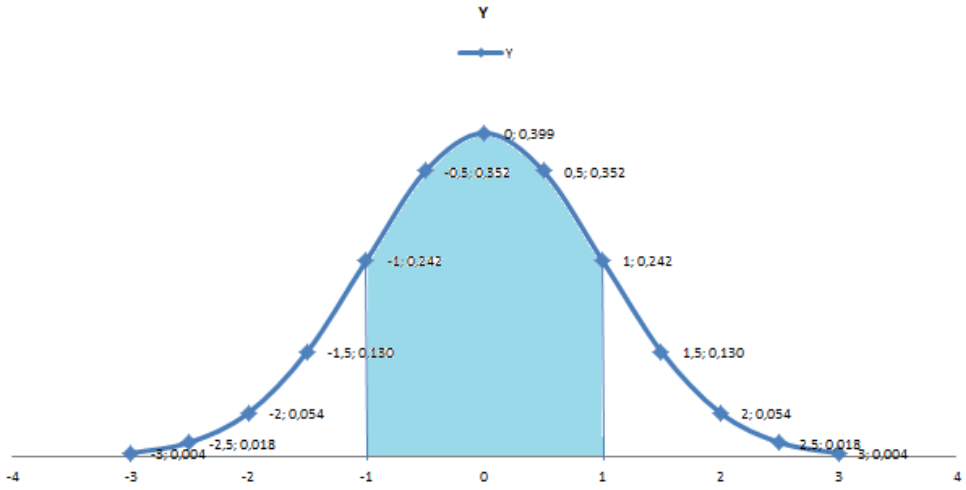
# CAPITULO III

# RESULTADOS

### 3.1.1 *CAMPANA DE GAUSS*

Los valores mayores que -1 o 0,242 y menoress que 1 o 0,2,42 es factible de acuerdo a nuestra investigación caso contrario no.

| X | Y |
|---|---|
| -3 | 0,004 |
| -2,5 | 0,018 |
| -2 | 0,054 |
| -1,5 | 0,130 |
| -1 | 0,242 |
| -0,5 | 0,352 |
| 0 | 0,399 |
| 0,5 | 0,352 |
| 1 | 0,242 |
| 1,5 | 0,130 |
| 2 | 0,054 |
| 2,5 | 0,018 |
| 3 | 0,004 |



### 3.1.2 *RESULTADOS DE ACUERDO A LA EFICIENCIA EN EL DESARROLLO DE LA APLICACIÓN HABLASMS*

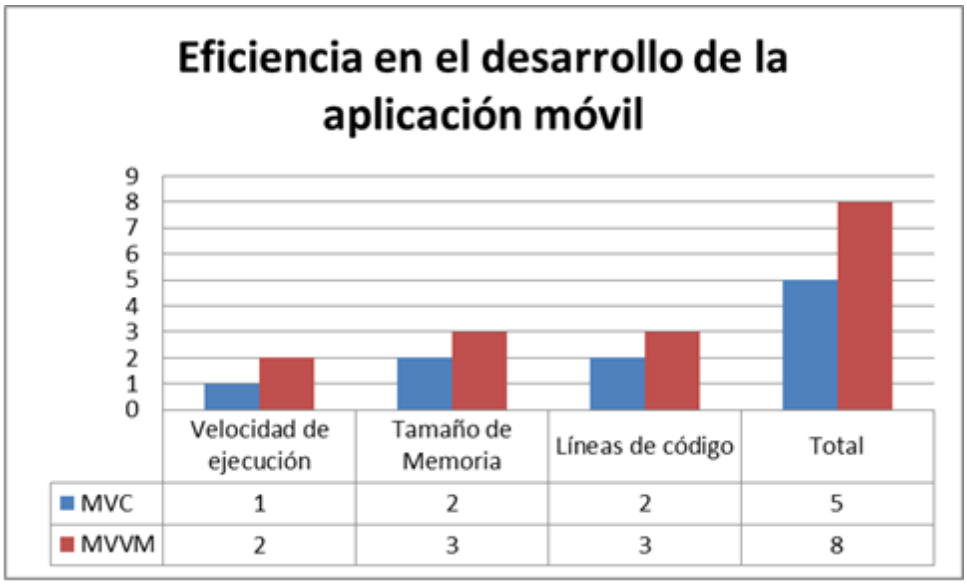



**Ilustración 15: Eficiencia en el desarrollo de la aplicación móvil.**

**Autores: Mercedes Guamán , Alfonso Caguana**

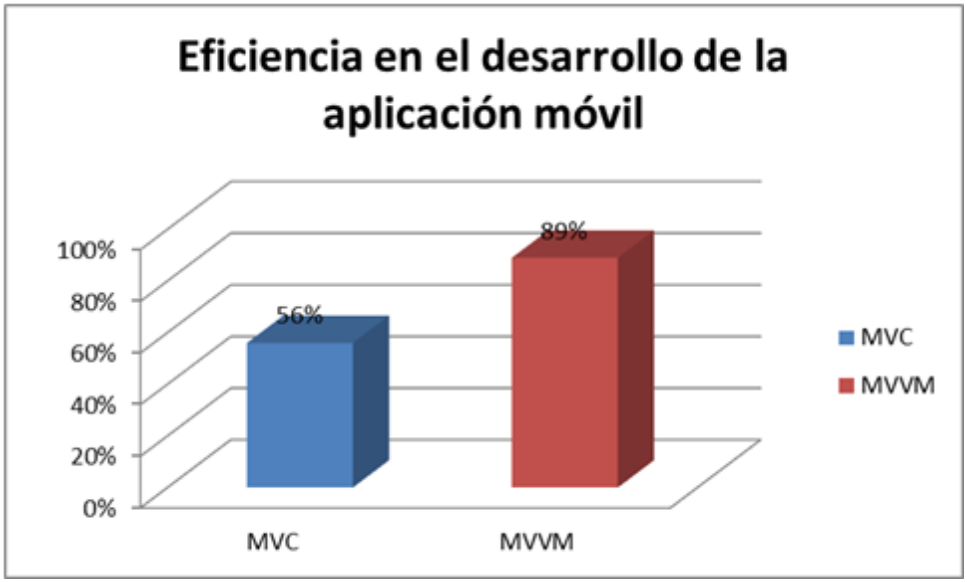



**Ilustración 16: Eficiencia en el desarrollo de la aplicación móvil.**

**Autores: Mercedes Guamán , Alfonso Caguana**

# CAPITULO IV

# PROPUESTAS

## 4.1 DESARROLLO DE LA APLICACIÓN HABLASMS

### 4.1.1 *PROCESO DE INGENIERÍA DE SOFTWARE EL MODELO OPEN UP*

#### 4.1.1.1 Antecedentes

En la actualidad existen muchas aplicaciones móviles para dispositivos móviles, cada una de ellas usa una determinada arquitectura según el tipo de aplicación se ha visto conveniente el inmiscuirse en el ámbito de desarrollo de una aplicación y de esa manera estar a la par de los desarrolladores que existen en el mundo y por ende en nuestro país.

Se ha visto muy tentativa la idea de desarrollar este tipo de aplicaciones.

#### 4.1.1.2 Objetivos

**General**

- Desarrollar una aplicación móvil que consiste en el envío de mensajes de texto que sean interpretados por el dispositivo móvil mediante comandos de voz para dispositivos móviles con sistema operativo Android.

**Específicos**

- Analizar el Proceso de Ingeniería de Software que sea acorde para desarrollo de la aplicación móvil.
- Contribuir y Promover el desarrollo de aplicaciones móviles.

#### 4.1.1.3 Alcance

El desarrollo de la aplicación móvil (Android) contempla las siguientes funcionalidades en base a las siguientes actividades:

- Se tendrá una reunión de toma de requerimientos para organizar la estructura del proyecto. De esta reunión se obtendrá un documento de alcance que defina la organización del proyecto que deberá ser aprobado previo al inicio del trabajo por los tutores y programadores (tesistas).
- Se realizará el desarrollo de la aplicación móvil para la plataforma Android.

#### 4.1.1.4 Fuera de alcance

- Labores de traducción de contenido y alertas.
- Soporte para plataformas no indicadas específicamente.

#### 4.1.1.5 Estructura de desglose de trabajo

**Tabla 15: Desglose de trabajo,**

**Autores: Mercedes Guamán, Alfonso Caguana**

| Id. | Descripción | Como | Deseo | Para | Criterios de Aceptación |
|---|---|---|---|---|---|
| **0** | Diseñar la arquitectura adecuada | Desarrollador | Diseñar la arquitectura adecuada en base a patrones y estándares actuales. | Desarrollar una aplicación móvil para la plataforma Android | Basada en estándares actuales. |
| **1** | Enviar SMS | Usuario | Enviar un SMS | Comunicar con distintos usuarios | *Enviar un SMS<br>*Recibir un SMS |
| **2** | Texto a Voz | Usuario | Transformar el texto que llega como SMS a Voz. | Evitar la lectura del mismo | *EnviarSMS.<br>*Recibir el SMS pero en comandos de voz. |
| **3** | Voz a Texto | Usuario | Dictar el SMS a enviar a un usuario | Evitar usar el teclado en caso de ser inapropiado | *Dictar el SMS.<br>*Recibir el SMS |
| **4** | Responder a comandos de voz | Usuario | Dar ciertas ordenes al teléfono | Que responda a las funcionalidades programadas en el desarrollo de la aplicación | *Enviar SMS<br>*Leer SMS<br>*Responder SMS |

#### 4.1.1.6 Equipo de trabajo

**Tabla 16: Equipo de trabajo,**

**Autores: Mercedes Guamán, Alfonso Caguana**

| Talento Humano | Tiempo relativo de trabajo |
|---|---|
| Programadores (Alfonso Caguana, Mercedes Guamán) | 100% |

#### 4.1.1.7 Roles tareas y funciones

##### *4.1.1.7.1 Programadores*

- Asegurar la ejecución de las actividades programadas, realizando el seguimiento de las fechas planificadas para la entrega de los diferentes tipos de procesos, entregables, documentación, e iteraciones subsiguientes en caso de encontrarse correcciones a realizar.
- Levantamiento de información.
- Elaboración de informes.
- Implementación de la solución.
- Acompañamiento.

##### *4.1.1.7.2 Entregables del proyecto*

Durante la etapa de desarrollo del proyecto se entregará partes funcionales de la aplicación de manera periódica, mismas que serán revisadas y aprobadas por los estudiantes de la escuela de conducción, tutores y desarrolladores de la tesis.

**Tabla 17: Etapas de desarrollo, Autores:**

**Mercedes Guamán, Alfonso Caguana**

| Hito | Descripción | Entregable |
|---|---|---|
| **Hito I** | Fin de etapas de inicio y planificación. | Planificación del inicio del proyecto. (tutores) |
| **Hito II** | Propuesta gráfica | Propuesta gráfica final.<br>Aprobación de la propuesta final. (programadores, tutores, usuarios) |
| **Hito III** | Programación de la Aplicación | Código |
| **Hito IV** | Implementación | Aplicación beta |
| **Hito V** | Corrección de Pruebas y Fallas | Aplicación beta |
| **Hito VI** | Aprobación final de la aplicación | Capacitación. |
| **Hito VII** | Cierre del proyecto | Entrega del proyecto |

#### *4.1.1.7.3 Plataformas*

Las plataformas sobre las cuales se desarrollará la aplicación son:

- Java Eclipse Juno
- Android

#### *4.1.1.7.4 Condiciones y supuestos*

- Los estudiantes de la escuela de conducción de la Universidad Nacional de Chimborazo y los Tutores y Tesistas son responsables de las postergaciones de las reuniones para las revisiones, correcciones y el desarrollo en sí de la aplicación.
- Los Estudiantes de la escuela de conducción, tutores y tesistas serán parte del suministro de información necesaria para que el desarrollo se pueda realizar con éxito.

#### *4.1.1.7.5 Riesgos*

**Tabla 18: Riesgos, Autores: Mercedes Guamán, Alfonso Caguana**

| N° | Área de riesgo | Probabilidad | Descripción |
|---|---|---|---|
| **1** | Negocio | Alta | Los usuarios y tutores asignados a revisiones no están disponibles para las fechas planeadas. |
| **2** | Negocio | Alta | No se entrega información necesaria al momento de requerirse. |
| **3** | Sistemas | Media | No se establece una persona que sirva como canal técnico para solucionar problemas, disponible durante el tiempo del proyecto en caso de presentarse. |
| **4** | Negocio | Media | Solicitudes de funcionalidad que no se encuentren establecidas en el documento de alcance. |
| **5** | Negocio | Alta | Solicitud de modificaciones a las propuestas gráficas originales. |

#### *4.1.1.7.6 Capacitación*

- Proceso de instalación en dispositivos de los usuarios finales y el uso de la aplicación móvil HablaSMS.

#### *4.1.1.7.7 Costos*

El costo de desarrollo del proyecto es de:

**Tabla 19: Costos del proyecto,** **Autores:**

**Mercedes Guamán, Alfonso Caguana**

| **Detalle** | **Valor** |
|---|---|
| Computador Portátil HP Core I3, HD 500GB, RAM 4GB. | $870 |
| Teléfono Celular Samsung Galaxy Ace | $280 |
| Cable de Datos | $5 |
| Horas de Internet. | $60 |
| Impresiones. | $120 |
| Viáticos | $150 |
| Imprevistos | $50 |
| Tienda Google Play | $100/año |
| **Costo Total** | **$1635.00** |

#### *4.1.1.7.8 Cuadro de aportes*

**Tabla 20: Financiamiento,** **Autores:**

**Mercedes Guamán, Alfonso Caguana**

| **Colaboradores** | **Valor $** | **Porcentaje %** |
|---|---|---|
| Financiamiento | $0 | 0% |
| Autofinanciamiento (Alfonso Caguana). | $817,5 | 50% |
| Autofinanciamiento (Mercedes Guamán). | $817,5 | 50% |
| **Total** | **$1635.00** | **100%** |

### 4.1.2 *METODOLOGÍA OPEN UP*

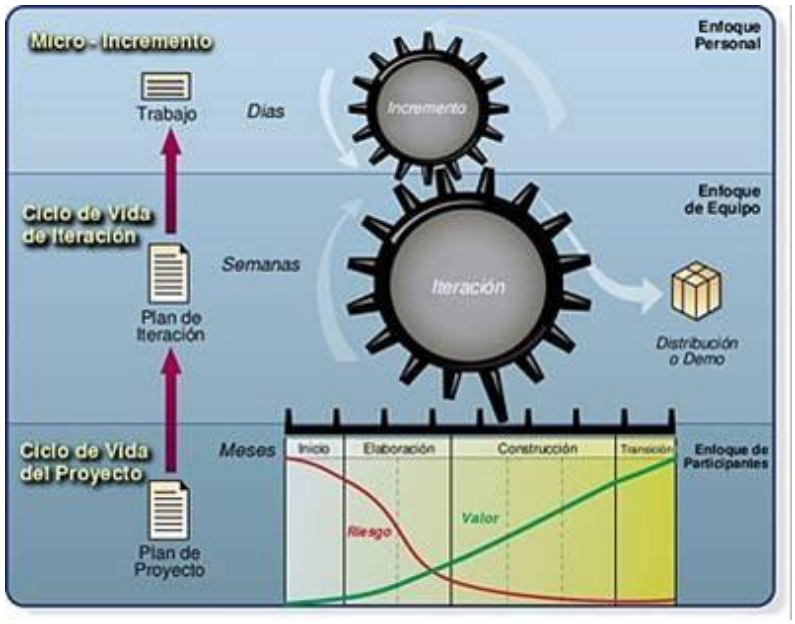



Ilustración 15: Modelo Open Up

Autores: Mercedes Guamán, Alfonso Caguana

Open UP es un marco del proceso del desarrollo del software Open Source que en un cierto plazo, se espera que cubra un amplio sistema de necesidades para los proyectos de desarrollo.

Open UP es un proceso iterativo para el desarrollo de software que es:

- Mínimo: Solo incluye el contenido del proceso fundamental
- Completo: Puede ser manifestado como proceso entero para construir un sistema.
- Extensible: Puede ser utilizado como base para agregar o para adaptar más procesos.

#### 4.1.2.1 El Open Up/Basic

Open UP/Basic es un subconjunto de Open UP que lleva un acercamiento ágil para el desarrollo del software, con solo un contenido fundamental provee un conjunto simplificado de artefactos, roles, tareas y guías de trabajo.

Open UP/Basic es un proceso iterativo del desarrollo del software que es mínimo, completo, y extensible. Es un proceso para equipos de desarrollo pequeños y que le dan valor a la colaboración y a las necesidades de los stake holder.

Open UP/Basic es extensible, porque puede ser utilizada como base para agregar o adaptar según las necesidades. Es un proceso ejemplar y extensible para una gama de los procesos

del desarrollo y de la gerencia del software que apoya el desarrollo iterativo, ágil, e incremental y es aplicable a un amplio sistema de plataformas y de usos del desarrollo.

#### 4.1.2.2 Beneficios en el uso del Open UP

- Ya que es apropiado para proyectos pequeños y de bajos recursos permite disminuir las probabilidades de fracaso en los proyectos pequeños e incrementar las probabilidades de éxito.
- Permite detectar errores tempranos a través de un ciclo iterativo.
- Evita la elaboración de documentación, diagramas e iteraciones innecesarios requeridos en la metodología RUP.
- Por ser una metodología ágil tiene un enfoque centrado al cliente y con iteraciones cortas.

#### 4.1.2.3 Principios del Open UP

- Colaborar para alinear intereses y para compartir conocimiento.
- Balancear las prioridades para maximizar las necesidades de los stake holder.
- Centrado en la Arquitectura.
- Desarrollo Iterativo.

#### 4.1.2.4 Organización de los componentes del Open UP

El Open UP está organizado en dos dimensiones diferentes pero interrelacionadas: el método y el proceso. El contenido del método es donde los elementos del método (roles, tareas, artefactos y lineamientos) son definidos, sin tener en cuenta como son utilizados en el ciclo de vida del proyecto. El proceso es donde los elementos del método son aplicados de forma ordenada en el tiempo. Muchos ciclos de vida para diferentes proyectos pueden ser creados a partir del mismo conjunto de elementos del método.

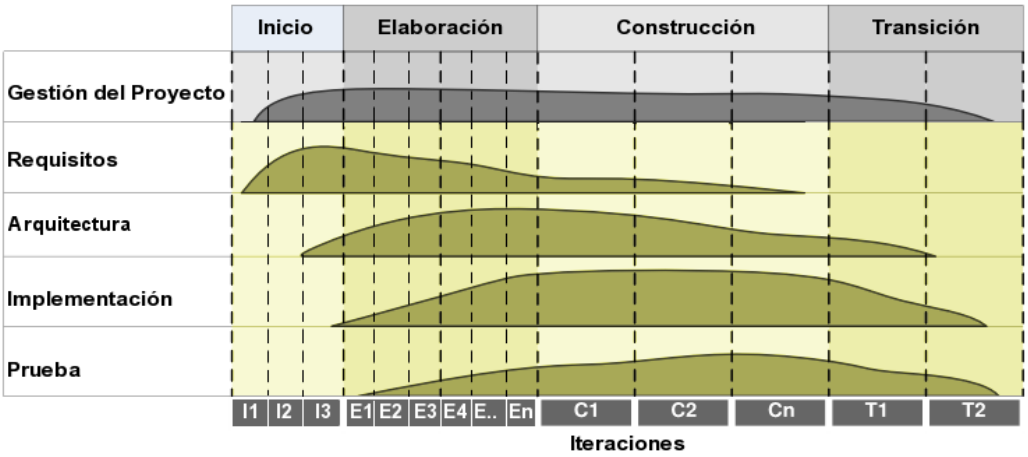



Ilustración 17: Ciclo de vida de un proyecto según Open UP,

Autores: Mercedes Guamán, Alfonso Caguana

#### 4.1.2.5 Áreas de interés

Los elementos del Open UP dirigen la organización del trabajo en los niveles personal, de equipo y de interesados.

A nivel personal, los integrantes de un proyecto contribuyen con su trabajo con pequeños incrementos en funcionalidad, denominados micro incrementos, los cuales representan los resultados obtenidos en pocas horas o pocos días de trabajo. La solución evoluciona basada en dichos micro incrementos de tal forma que el progreso puede ser visualizado efectivamente cada día. Los integrantes del equipo de desarrollo de forma abierta comparten su progreso diario el cual incrementa la visibilidad en el trabajo, la confianza y el trabajo en equipo.

El proyecto en general se divide en iteraciones, las cuales son planificadas en un intervalo definido de tiempo que no superan las pocas semanas. El Open UP tiene elementos que ayudan a los equipos de trabajo a enfocar los esfuerzos a través del ciclo de vida de cada iteración de tal forma que se puedan distribuir funcionalidades incrementales de una manera predecible, una versión totalmente probada y funcional al final de cada iteración.

El Open UP estructura el ciclo de vida de un proyecto en cuatro fases: concepción, elaboración, construcción y transición. El ciclo de vida del proyecto provee a los interesados un mecanismo de supervisión y dirección para controlar los fundamentos del proyecto, su ámbito, la exposición a los riesgos, el aumento de valor y otros aspectos.

#### 4.1.2.6 Fases del Open UP

##### *4.1.2.6.1 Concepción*

Primera de las 4 fases en el proyecto del ciclo de vida, acerca del entendimiento del propósito y objetivos y obteniendo suficiente información para confirmar que el proyecto debe hacer. El objetivo de ésta fase es capturar las necesidades de los stake holder en los objetivos del ciclo de vida para el proyecto.

En esta fase se define las especificaciones de la aplicación móvil que se va a realizar para el desarrollo del HablaSMS.

##### *4.1.2.6.2 Elaboración*

Es el segundo ciclo de vida del Open UP donde se trata los riesgos significativos para la arquitectura. El propósito de esta fase es establecer la base la elaboración de la arquitectura del sistema.

Pues de éste modo se define la arquitectura a utilizar que la Model -View –View Model (MVVM), que es la arquitectura que se adapta a tipo de aplicación a desarrollar

##### *4.1.2.6.3 Construcción*

Esta fase está enfocada al diseño, implementación y prueba de las funcionalidades para desarrollar un sistema completo. El propósito de esta fase es completar el desarrollo del sistema basado en la Arquitectura definida como la MVVM.

Para el desarrollo de la aplicación HablaSMS se utiliza la arquitectura hibrida de aplicaciones móviles basadas en Java siguiendo el patrón de diseño MVVM (Model, View, ViewModel), que consta de 3 capas o componentes fundamentales que son:

- Model (Capa de lógica de negocio de la aplicación).
- View (Capa de presentación de los datos)
- ViewModel (Modelo de Vista donde va todo el trabajo que la vista necesita hacer).

El ViewModel extrae los datos del modelo, y prepara los datos para que una vista pueda usarlos. El ViewModel en sí no sabe nada acerca de las vistas, éste simplemente tiene un conjunto de propiedades para que una vista pueda hacer DataBinding a ellas. A este

ViewModel podemos asignarle cualquier número de vistas que queramos y dichas vistas representará esos datos como quiera.

Este patrón de diseño se separan los datos de la aplicación, la interfaz de usuario pero en vez de controlar manualmente los cambios en la vista o en los datos, estos se actualizan directamente cuando sucede un cambio en ellos, por ejemplo si la vista actualiza un dato que está presentando se actualiza el modelo automáticamente y viceversa.

El patrón MVVM de la aplicación HablaSMS estáestructurada de la siguiente manera.

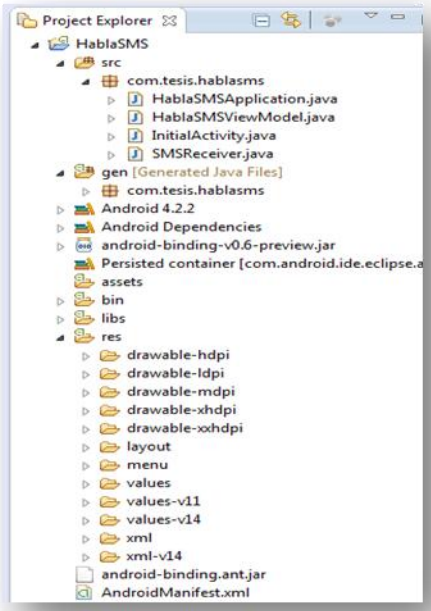



Ilustración 18: Patrón MVVM en la aplicación HablaSMS, Autores: Mercedes Guamán, Alfonso Caguana

La carpeta HablaSMS es el proyecto o la aplicación que contiene sub carpetas, las cuales se describen a continuación:

- **src:** Es la carpeta que contiene el código fuente de la aplicación. Como puede observar los ficheros Java se almacenan en un espacio de nombres como se describe:

  HablaSMSAplication.java, HablaSMSViewModel.java, InictialActivity.java, SMSReceiver.java.
- **gen:** Carpeta que contiene el código generado de forma automática por el SDK. No se recomienda modificar de forma manual estos ficheros. Como son:

    - **BuildConfig.java:** Define la constante DEBUG para que desde Java pueda saber si la aplicación está en fase de desarrollo.
    - **R.java:** Define una clase que asocia los recursos de la aplicación con identificadores. De esta forma los recursos podrán ser accedidos desde Java.
- **Android 4.2.2:** Código JAR, el API de Android según la versión seleccionada.
- **Android Dependencies:** Son las librerías asociadas al proyecto.
- **assets:** Carpeta que puede contener una serie arbitraria de ficheros o carpetas que podrán ser utilizados por la aplicación son los ficheros de datos, fuentes, etc. A l igual que la carpeta res, no se modifica el contenido de los ficheros de esta carpeta ni se asocia un identificador.
- **bin:** En esta carpeta se compila el código y se genera el .apk, fichero comprimido que contiene la aplicación final lista para instalar en el caso del desarrollo de la aplicación HablaSMS.
- **libs:** Código JAR con librerías que se usa en la aplicación HablaSMS. Se ha añadido una librería android-support-v4.jar, android-binding-v0.6-preview.ja, androiid-support-v7-gridlayout.jar cuyo objetivo es añadir nuevas funcionalidades que no aparecían en el nivel de API 4 y que aparecen en versiones más recientes del SDK.
- **res:** Carpeta que contiene los recursos usados por la aplicación. Como se ve existen subcarpetas que pueden tener un sufijo si desea que el recurso solo se cargue al cumplirse una condición.
    - **drawable:** En esta carpeta se almacenan los ficheros de imágenes (JPG o PNG) y descriptores de imágenes en XML.
    - **layout:** Contiene ficheros XML con vistas de la aplicación. Las vistas permitirán configurar las diferentes pantallas que son parte de la interfaz de usuario de la aplicación. Se utiliza un formato similar al HTML usado para diseñar páginas Web.
    - **menu:** Ficheros XML con los menús de cada actividad.

- **values:** También utiliza ficheros XML para indicar valores del tipo string, color o estilo. De esta manera se puede cambiar los valores sin necesidad de ir al código fuente.
- **xml:** Otros ficheros XML requeridos por la aplicación.
- **raw:** Ficheros adicionales que no se encuentran en formato XML.

- **AndroidManifest.xml:** Este fichero describe la aplicación Android. En él se indican las actividades, intenciones, servicios y proveedores de contenido de la aplicación. También se declaran los permisos que requerirá la aplicación. Se indica la versión mínima de Android para ejecutarla, el paquete Java, la versión de la aplicación.
- **ic_launcher-web.png:** Icono de la aplicación de gran tamaño en este caso la aplicación HablaSMS para ser usado en páginas Web. El nombre puede variar si se indica uno diferente en el proceso de creación del proyecto.
- **proguard-project.txt:** Fichero de configuración de la herramienta ProGuard, que te permite optimizar y ofuscar el código generado. Es decir, se obtiene un .apk más pequeño y donde resulta más difícil hacer ingeniería inversa.
- **default.properties:** Fichero generado automáticamente por el SDK. Nunca hay que modificarlo. Se utiliza para comprobar la versión del API y otras características cuando se instala la aplicación en el terminal.

### 4.1.3 *SMARTPHONE*

La aplicación del HablaSMS está diseñada para funcionar en todos los dispositivos Smartphone, con la única recomendación que deben poseer el sistema operativo Android.

### 4.1.4 *CÓDIGO DE LA APLICACIÓN*

La aplicación HablaSMS está compuesta por el siguiente código de acuerdo a las diferentes funciones que realiza la aplicación HablaSMS para el gran aporte de la seguridad al usuario de telefonía móvil también a la eficiencia del desarrollo de la aplicación HablaSMS.

```
//declaraciónde variables
        protectedstaticfinalintRESULT_SPEECH = 1;
        protectedstaticfinalintCONTACT_SELECTED = 2;
        protectedstaticfinalintLISTEN_MESSAGE = 3;
        private ArrayList<String> numbers = new ArrayList<String>();
        private ArrayList<String> names = new ArrayList<String>();
        publicstatic TextToSpeech tts;
        //texto binding alcuerpodelsms
        publicfinal StringObservable SMSText =
                new StringObservable("");

        //texto binding alcampodedestinatario
        publicfinal StringObservable DisplayNumber =
                new StringObservable("");

        //comando binding para el botónIniciarMenú
        publicfinal Command Listen =
                        new Command(){
                                @Override
publicvoid Invoke(View arg0, Object... arg1) {
                                        listenMenu();
            }
        };

        //comando binding para el botón leer SMS
        publicfinal Command Read =
                        new Command(){
                                @Override
publicvoid Invoke(View arg0, Object... arg1) {
                                        readSMS();
            }
```

```
        };

        //comando binding para el botónenviar SMS
        publicfinal Command Send =
                        new Command(){
                                @Override
publicvoid Invoke(View arg0, Object... arg1) {
                                        sendSMS();
            }
        };

        //comando binding para el botónañadirdestinatario
        publicfinal Command AddContacts =
                        new Command(){
                                @Override
publicvoid Invoke(View arg0, Object... arg1) {
                                        selectContacts();
            }
        };

        //funciónparaseleccionarloscontactos
        privatevoid selectContacts() {
                //creaunnuevo intent paraabrirloscontactosen el celular
Intent intent = newIntent(Intent.ACTION_GET_CONTENT);
      intent.setType
(ContactsContract.CommonDataKinds.Phone.CONTENT_ITEM_TYPE);
startActivityForResult(intent, CONTACT_SELECTED);
        }

        //funciónparamostrartodosloscontactos ->alterna
        @SuppressWarnings("unused")
```

```
        privatevoid displayContacts() {


            ContentResolver cr = getContentResolver();
              Cursor cur = cr.query(ContactsContract.Contacts.CONTENT_URI,
        null, null, null, null);
        if (cur.getCount() > 0) {
        while (cur.moveToNext()) {
                    String id =
cur.getString(cur.getColumnIndex(ContactsContract.Contacts._ID));
                    String name =
cur.getString(cur.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME));
        if (Integer.parseInt(cur.getString(
        cur.getColumnIndex(ContactsContract.Contacts.HAS_PHONE_NUMBER))) > 0) {
                     Cursor pCur = cr.query(
                            ContactsContract.CommonDataKinds.Phone.CONTENT_URI,
        null,
                            ContactsContract.CommonDataKinds.Phone.CONTACT_ID +"
= ?",
        new String[]{id}, null);
        while (pCur.moveToNext()) {
                          String phoneNo =
pCur.getString(pCur.getColumnIndex(ContactsContract.CommonDataKinds.Phone.NUMB
ER));
        Toast.makeText(HablaSMSViewModel.this, "Name: " + name + ", Phone No: " +
phoneNo, Toast.LENGTH_SHORT).show();
        }
        pCur.close();
                }
              }
            }
          }
```

```
        //comandopara el botón leer SMS
        privatevoid readSMS() {
                if (SMSText.get().isEmpty()) {
                        tts.speak(getResources().getString(R.string.body_empty),
TextToSpeech.QUEUE_ADD, null);
                } else {
                        tts.speak(SMSText.get(), TextToSpeech.QUEUE_ADD, null);
                }
        }

        //comandopara el botónenviar SMS
        publicvoid sendSMS() {
                //controlarestadodelmensaje
                String SENT = "SMS_SENT";
      String DELIVERED = "SMS_DELIVERED";

      PendingIntent deliveredPI = PendingIntent.getBroadcast(this, 0,
new Intent(DELIVERED), 0);

      //cuando el mensaje ha sidoenviado
registerReceiver(new BroadcastReceiver(){
        @Override
publicvoid onReceive(Context arg0, Intent arg1) {
switch (getResultCode())
            {
case HablaSMSViewModel.RESULT_OK:
        tts.speak(getResources().getString(R.string.send_confirmation),
TextToSpeech.QUEUE_ADD, null);
Toast.makeText(getBaseContext(), R.string.send_confirmation,
Toast.LENGTH_SHORT).show();
```

```
break;
case SmsManager.RESULT_ERROR_GENERIC_FAILURE:
        Toast.makeText(getBaseContext(), R.string.generic_failure,
Toast.LENGTH_SHORT).show();
break;
case SmsManager.RESULT_ERROR_NO_SERVICE:
        tts.speak(getResources().getString(R.string.no_service),
TextToSpeech.QUEUE_ADD, null);
Toast.makeText(getBaseContext(), R.string.no_service,Toast.LENGTH_SHORT).show();
break;
case SmsManager.RESULT_ERROR_NULL_PDU:
        tts.speak(getResources().getString(R.string.null_pdu),
TextToSpeech.QUEUE_ADD, null);
Toast.makeText(getBaseContext(), R.string.null_pdu, Toast.LENGTH_SHORT).show();
break;
case SmsManager.RESULT_ERROR_RADIO_OFF:

}
        }
      }, new IntentFilter(SENT));

      //cuando el mensajese ha entregado
registerReceiver(new BroadcastReceiver(){
        @Override
publicvoid onReceive(Context arg0, Intent arg1) {
switch (getResultCode())
            {
case HablaSMSViewModel.RESULT_OK:
        tts.speak(getResources().getString(R.string.sms_delivered),
TextToSpeech.QUEUE_ADD, null);
```

```
Toast.makeText(getBaseContext(), R.string.sms_delivered,
Toast.LENGTH_SHORT).show();
break;
case HablaSMSViewModel.RESULT_CANCELED:
        tts.speak(getResources().getString(R.string.sms_not_delivered),
TextToSpeech.QUEUE_ADD, null);
Toast.makeText(getBaseContext(), R.string.sms_not_delivered,
Toast.LENGTH_SHORT).show();
break;
            }
         }
      }, newIntentFilter(DELIVERED));

               //envía el sms
      SmsManager sender = SmsManager.getDefault();
               if (!numbers.isEmpty()) {
                     if (!SMSText.get().equals("")) {
                           for (int i = 0; i < numbers.size(); i++) {
                                 sender.sendTextMessage(numbers.get(i), null,
SMSText.get(), sentPI, deliveredPI);
                                 clearData();
                           }

                     } else
{
                     tts.speak(getResources().getString(R.string.number_empty),
TextToSpeech.QUEUE_ADD, null);
                     Toast.makeText(this, R.string.number_empty,
Toast.LENGTH_LONG).show();
                     tts.speak(getResources().getString(R.string.blank_fields),
TextToSpeech.QUEUE_ADD, null);
```

```
                    Toast.makeText(this, R.string.blank_fields,
Toast.LENGTH_LONG).show();
                }
        }

        //abre el menú
        privatevoid listenMenu() {
                Intent speechIntent =
newIntent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
                startActivityForResult(speechIntent, RESULT_SPEECH);
                //SMSText.set("");
        }

        //sealistaparaescuchar el sms
        privatevoid listenSMS() {
                tts.speak(getResources().getString(R.string.speak_now),
TextToSpeech.QUEUE_ADD, null);
                Handler handler = newHandler();

           }, 2000);
        }

        //borratodoslosdatos
        privatevoid clearData() {
                numbers.clear();
                names.clear();
                SMSText.set("");
                DisplayNumber.set("");
        }

        //insertalosnúmeros
```

```
        privatevoid insertNumber(String number) {
                String check = DisplayNumber.get();
                if ((check.length() ==
10)&&(check.charAt(0)=='0')&&(check.charAt(1)=='9')) {
                        numbers.add(number);
                } else
        }

        @SuppressWarnings("unused")
        privatevoid trimContacts() {
                String[] splits = DisplayNumber.get().split(";");
                for (int i = 0; i < splits.length; i++) {
                        insertNumber(splits[i]);
                }
        }

        //selecciónde opciones
        privatevoid selectOptions(String option) {
                int caso =0;
                if ((option.equals("agregar contacto"))
                                ||(option.equals("agregar destinatario"))
                                ||(option.equals("agregar número"))) {
                        caso = 1;
                }

                if ((option.equals("dictar mensaje"))
                                ||(option.equals("agregar mensaje"))) {
                        caso = 2;
                }

                if (option.equals("enviar mensaje")) {
```

```
            caso = 3;
        }

        if (option.equals("salir")) {
            caso = 4;
        }

        if ((option.equals("muestrame los destinatarios"))
                ||(option.equals("muestrame los contactos agregados"))
                ||(option.equals("muestrame la lista de contactos"))
                ||(option.equals("listar destinatarios"))) {
            caso = 5;
        }

        if ((option.equals("borrar texto"))
                ||(option.equals("eliminar texto"))
                ||(option.equals("borrar mensaje"))) {
            caso = 6;
        }

        if ((option.equals("borrar destinatarios"))
                ||(option.equals("eliminar destinatarios"))
                ||(option.equals("eliminar contactos"))) {
            caso = 7;
        }

        switch (caso) {
        case 0:
            tts.speak(getResources().getString(R.string.invalid_string),
TextToSpeech.QUEUE_ADD, null);
            break;
```

```
case 1:
        selectContacts();
        break;
case 2:
        listenSMS();
        break;

case 3:
        sendSMS();
        break;

case 4:
        closeApplication();
        break;

case 5:
        sayAddedContacts();
        break;

case 6:
        clearSMSText();
        break;

case 7:
        clearAddedContacts();
        break;

default:
        break;
}
```

```
        }

        //cierralaaplicación
                publicvoid closeApplication()
           {
        tts.speak(getResources().getString(R.string.good_bye),
TextToSpeech.QUEUE_ADD, null);
              Handler handler = newHandler();
        handler.postDelayed(new Runnable() {

                                @Override
                                publicvoid run() {
                                        finish();
                                }
                        }, 1000);

           }

        //borra el textodelsms
        privatevoid clearSMSText() {
                if (SMSText.get().equals("")) {
                        tts.speak(getResources().getString(R.string.body_empty),
TextToSpeech.QUEUE_ADD, null);
                tts.speak(getResources().getString(R.string.blank_fields),
TextToSpeech.QUEUE_ADD, null);
                } else {
                        SMSText.set("");
                        tts.speak("Mensaje Borrado", TextToSpeech.QUEUE_ADD, null);

                }
        }
```

```
        //borratodosloscontactosañadidos
privatevoid clearAddedContacts() {
        if (names.isEmpty()) {
                tts.speak(getResources().getString(R.string.number_empty),
TextToSpeech.QUEUE_ADD, null);
                tts.speak(getResources().getString(R.string.blank_fields),
TextToSpeech.QUEUE_ADD, null);
                } else {
                        names.clear();
                        numbers.clear();
                        DisplayNumber.set("");
                        tts.speak("Contactos Eliminados", TextToSpeech.QUEUE_ADD,
null);
                }
        }

   //lista todos los contactos añadidos
        privatevoid sayAddedContacts() {
                if (names.isEmpty()) {
                tts.speak(getResources().getString(R.string.number_empty),
TextToSpeech.QUEUE_ADD, null);
                tts.speak(getResources().getString(R.string.blank_fields),
TextToSpeech.QUEUE_ADD, null);
                } else {
                        for (int i = 0; i < names.size(); i++) {
                                tts.speak(names.get(i).toString(),
TextToSpeech.QUEUE_ADD, null);
                        }
                }
        }
```

```
        @Override
publicvoid onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
this.inflateAndBind(R.xml.hablasmsactivity_metadata, this);

int currentapiVersion = android.os.Build.VERSION.SDK_INT;

try {
        if (currentapiVersion >= android.os.Build.VERSION_CODES.JELLY_BEAN){
        tts= new TextToSpeech(this,this);
        tts.speak(getResources().getString(R.string.hello), TextToSpeech.QUEUE_ADD,
null);
        Toast.makeText(this, R.string.init_tts_success, Toast.LENGTH_LONG).show();
        } else{
        Intent checkIntent = newIntent();
checkIntent.setAction(TextToSpeech.Engine.ACTION_CHECK_TTS_DATA);
startActivityForResult(checkIntent, RESULT_SPEECH);
        }

                } catch (Exception e) {
                        Toast.makeText(this, R.string.init_tts_fail,
Toast.LENGTH_LONG).show();
                }
   }

   @Override
protectedvoid onActivityResult(int requestCode, int resultCode, Intent data){
        super.onActivityResult(requestCode, resultCode, data);

        switch (requestCode) {
```

```
                case RESULT_SPEECH:{
                        if (resultCode == RESULT_OK&&null != data){
                                ArrayList<String> text =
data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
                                selectOptions(text.get(0));
                                //SMSText.set(text.get(0));
                        }
                        if (resultCode ==
TextToSpeech.Engine.CHECK_VOICE_DATA_PASS){
                                tts= new TextToSpeech(this,this);
                        }
                }
                        break;
                case CONTACT_SELECTED:{
                        if (resultCode == RESULT_OK&&null != data) {
                        Uri contactData = data.getData();
                           Cursor c = null;
                        try {
                                 c = getContentResolver().query(contactData, new String[]{
                            ContactsContract.CommonDataKinds.Phone.NUMBER,
                            ContactsContract.CommonDataKinds.Phone.DISPLAY_NAME },
null, null, null);
                                if (c != null&& c.moveToFirst()) {
                             String number = c.getString(0);
                             String name = c.getString(1);
        addSelectedNumber(name, number);
                            }

                        }
                        finally {
        if (c != null) {
```

```
        c.close();
                    }
                        }
                    }
                }
                break;
                caseLISTEN_MESSAGE:{
                    if (resultCode == RESULT_OK&&null != data){
                        ArrayList<String> text =
data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
                        SMSText.set(text.get(0));
                    }
                    if (resultCode ==
TextToSpeech.Engine.CHECK_VOICE_DATA_PASS){
                        tts= new TextToSpeech(this,this);
                    }
                }
                    break;
        }
    }

        //añadir los números seleccionados
    //se verifica si están repetidos, si es así, no se agregan
privatevoid addSelectedNumber(String name, String number) {
        Boolean repeated = false;
        if (names.isEmpty() && numbers.isEmpty()) {
                    DisplayNumber.set(name+";");
                    names.add(name);
                    numbers.add(number);
                    //Toast.makeText(this, "primera vez",
Toast.LENGTH_LONG).show();
```

```
                } else {
                        for (int i = 0; i < names.size(); i++) {
                                if
(names.get(i).equals(name)&&numbers.get(i).equals(number)) {
                                        repeated=true;
                                }
                        }

                        if (repeated==false){
                                names.add(name);
                                numbers.add(number);
                                DisplayNumber.set(DisplayNumber.get()+";"+name);
                        }
                        //Toast.makeText(this, String.valueOf(numbers.size()),
Toast.LENGTH_LONG).show();
                }
        }

  //message box de confirmación de salida
publicvoid msBoxCloseConfirm()
   {
      AlertDialog.Builder dlgAlert  =new AlertDialog.Builder(this);
dlgAlert.setTitle(getResources().getString(R.string.title_close));
dlgAlert.setMessage(getResources().getString(R.string.title_confirm_close));
dlgAlert.setPositiveButton(getResources().getString(R.string.button_yes),new
DialogInterface.OnClickListener() {
publicvoid onClick(DialogInterface dialog, int whichButton) {
        tts.speak(getResources().getString(R.string.good_bye),
TextToSpeech.QUEUE_ADD, null);
        Handler handler = newHandler();
        handler.postDelayed(new Runnable() {
```

```
                                        @Override
                                        publicvoid run() {
                                                finish();
                                        }
                                }, 1000);

        }
    });
dlgAlert.setNegativeButton(getResources().getString(R.string.button_no),null);
dlgAlert.setCancelable(true);
dlgAlert.create().show();
}

 //override al método de salida
   @Override
        publicboolean onKeyDown(int keyCode, KeyEvent event) {
        switch(keyCode){
        case KeyEvent.KEYCODE_BACK:
                msBoxCloseConfirm();
        returntrue;
           }
        returnsuper.onKeyDown(keyCode, event);
        }

        //@Override
        publicvoid onInit(int status) {
                if (status == TextToSpeech.SUCCESS) {
                        tts.speak(getResources().getString(R.string.hello),
TextToSpeech.QUEUE_ADD, null);
```

```
                Toast.makeText(HablaSMSViewModel.this,
R.string.init_tts_success, Toast.LENGTH_LONG).show();
            } elseif (status == TextToSpeech.ERROR) {
                Toast.makeText(HablaSMSViewModel.this,R.string.init_tts_fail,
Toast.LENGTH_LONG).show();
            }

        }
```

## 4.2 EJECUCIÓN EN EL EMULADOR Y EN DISPOSITIVO REAL

### 4.2.1 *EJECUCIÓN EN EL EMULADOR*

1. Selecciona Run > Run. (Ctrl-F11).
2. Seleccionar Android Application. Eclipse creará automáticamente una nueva configuración de ejecución para el proyecto y lanzará el emulador. (La inicialización del emulador puede ser algo lenta, por esta razón es mejor no cerrar el emulador) Una vez que el emulador esté cargado, debe ver algo así:


Ilustración 19: Ejecución en Emulador Android,
Autores: Mercedes Guamán, Alfonso Caguana

### 4.2.2 *EJECUCIÓN EN UN TERMINAL REAL*

Ejecutar y depurar los programas en un terminal real resulta posible desde el SDK. No tiene más que usar un cable USB para conectarlo al PC. Resulta imprescindible haber instalado un driver especial en el PC. Puede encontrar un driver genérico que se encuentra en la carpeta android-sdk-Windows\usb_driver del SDK Android. Aunque lo más probable es que tenga que utilizar el driver del fabricante.

Para la ejecución en un terminal real siga los siguientes pasos:

1. Abra Android SDK and AVD Manager y asegúrese que está instalado el paquete USB Driver. En caso contrario instálelo.


Ilustración 20: Búsqueda de driver,
Autores: Mercedes Guamán, Alfonso Caguana.

Posiblemente este driver genérico no sea adecuado para el terminal y tenga que utilizar del fabricante. Si no dispone de él, puede buscarlo en: http://developer.android.com/sdk/oem-usb.html

2. En el terminal accede al menú Ajustes > Opciones de desarrollador y asegúrese que la opción Depuración de USB está activada.


Ilustración 21: Depuración usb activada,

Autores: Mercedes Guamán, Alfonso Caguana

3. Conecta el cable USB.
4. Se indicará que hay un nuevo hardware y le pedirá que le indique el controlador.

Por último se ejecuta directamente el apk en el teléfono móvil que lo esté probando.


Ilustración 19: Inicio del Sistema

Autores: Mercedes Guamán, Alfonso Caguana

## 4.3 TRANSICIÓN

Es la última fase, cuyo propósito es asegurar que el sistema es entregado a los usuarios, y evalúa la funcionalidad y performance del último entregable de la fase de construcción y para aquello se presenta el siguiente manual de usuario y su respectivo manual técnico en el caso que alguien quiera basarse en la aplicación para realizar alguna mejora, pues lo puede realizar con toda la confianza ya que es una de las ventajas de utilizar código abierto.

## 4.4 MANUAL DE USUARIO

Este manual de usuario contiene la información y los pasos necesarios para la utilización correcta del software, aplicación móvil HablaSMS.

El objetivo del manual de usuario es dar a conocer al usuario final los pasos necesarios para utilizar y manipular el software, aplicación móvil HablaSMS mediante comandos de voz.

La aplicación móvil HablaSMS está compuesta por comandos de voz que son:

- Agregar contacto, agregar destinatario, agregar número
- Dictar mensaje, agregar mensaje
- Enviar mensaje
- salir

### 4.4.1 *REQUISITOS PARA LA CONFIGURACIÓN DEL TELÉFONO MÓVIL PARA LA UTILIZACIÓN CORRECTA DE LA APLICACIÓN HABLASMS*

Para configurar el teléfono móvil debe seguir los siguientes pasos:

- Verificar la versión del sistema operativo android, cabe mencionar que la aplicación es compatible con la mayoría de las versiones del sistema operativo android con la única excepción de la versión 2.1.2 no es compatible con esta versión.
- Debe tener una conexión a internet o a su vez contratar el paquete de datos de cualquier operadora que sea del agrado.
- Debe tener instalado el motor de síntesis para texto hablado (pico tts) o cualquier motor de texto hablado en caso de no tener instalado el motor de síntesis para texto hablado debe crear una cuenta en gmail para descargar dicho motor de síntesis de voz.

### 4.4.2 *INGRESO A LA APLICACIÓN HABLASMS*

Para la utilización de la aplicación HablaSMS debe seguir los siguientes pasos:

- Una vez instalada la aplicación desplegamos el menú del dispositivo móvil y observa que se agregó el ícono de HablaSMS, Ingresa a la aplicación dando clic en el icono HablaSMS

Ilustración 22: icono HablaSMS,
Autores: Mercedes Guamán, Alfonso Caguana

- Aparece la siguiente interfaz que tiene los siguientes iconos: un micrófono para los comandos de voz, un recuadro para ingresar los contactos, un recuadro para dictar el mensaje, ícono de carta para enviar los mensajes, una opción para ayuda donde puede leer los comandos a utilizar.

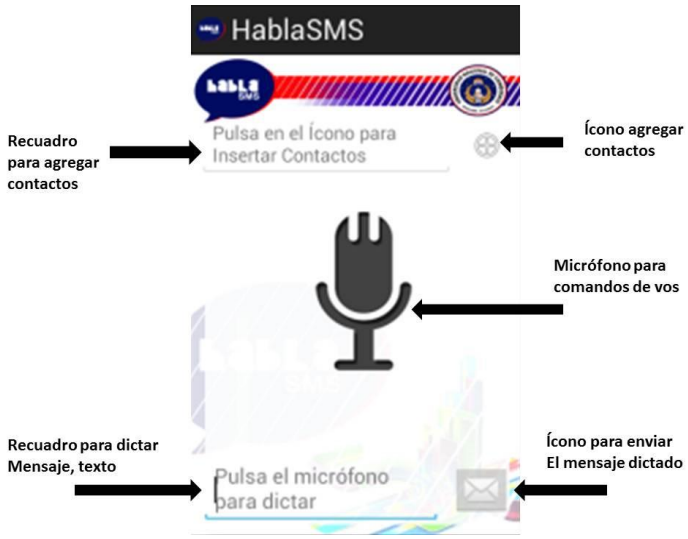



Ilustración 21: interfaz de la aplicación HablaSMS,
Autores: Mercedes Guamán, Alfonso Caguana.

### 4.4.3 *AGREGAR CONTACTO*

- Presionar el icono del micrófono


Ilustración 22: ícono micrófono,

Autores: Mercedes Guamán, Alfonso Caguana

- Hablar y decir los comandos de voz que son los siguientes: agregar contacto, agregar destinatario, agregar número.


Ilustración 23: Contactos o destinatarios,

Autores: Mercedes Guamán, Alfonso Caguana

### 4.4.4 *DICTAR MENSAJE*

- Presionar el icono del micrófono


Ilustración24: ícono micrófono,

Autores: Mercedes Guamán, Alfonso Caguana

- Hablar y decir los comandos de voz que son los siguientes: dictar mensaje, agregar mensaje.

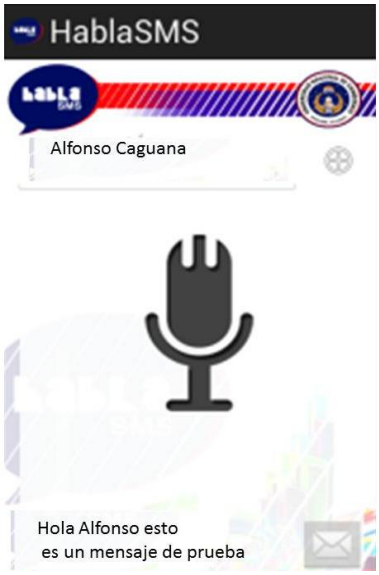



Ilustración 25: Insertar mensaje,

Autores: Mercedes Guamán, Alfonso Caguana.

### 4.4.5 *ENVIAR MENSAJE*

- Presionar el icono del micrófono


Ilustración 26: ícono del micrófono,

Autores: Mercedes Guamán, Alfonso Caguana

- Hablar el comando de voz: enviar mensaje


Ilustración 27: Enviar  mensaje,

Autores: Mercedes Guamán, Alfonso Caguana

### 4.4.6 *BORRAR TEXTO*

- Presionar el icono del micrófono


Ilustración 238: ícono de micrófono,

Autores: Mercedes Guamán, Alfonso Caguana.

- Hablar y decir el comando de voz: borrar texto, eliminar texto, borrar mensaje.


Ilustración 29: Borrar texto,

Autores: Mercedes Guamán, Alfonso Caguana.

### 4.4.7 *BORRAR DESTINATARIO*

- Presionar el icono del micrófono


Ilustración 24: ícono micrófono,

Autores: Mercedes Guamán, Alfonso Caguana

- Hablar y decir el comando de voz: borrar destinatarios, eliminar destinatarios, eliminar contactos.


Ilustración 31: Eliminar destinatario,

Autores: Mercedes Guamán, Alfonso Caguana

### 4.4.8 *SALIR*

- Presionar el icono del micrófono


Ilustración 32: ícono micrófono,
Autores: Mercedes Guamán, Alfonso Caguana

- Una vez que llegue el mensaje si desea responder sólo dicte el comando si caso contrario.
- Hablar y decir el comando de voz: salir
- La aplicación se despide.

## 4.5 MANUAL TÉCNICO

Este manual técnico contiene la información y los requerimientos para la instalación y utilización de las correctas herramientas del software, aplicación móvil HablaSMS. El objetivo de esta guía técnica es dar a conocer las herramientas correspondientes al desarrollo y utilización del software o aplicación móvil HablaSMS y que éste sea a su vez una base para nuevos desarrollos ya que esa es la ventaja de Android que se debe compartir el código.

### 4.5.1 *REGLAS DEL SOFTWARE, APLICACIÓN MÓVIL HABLASMS*

- Para el desarrollo de la aplicación móvil HablaSMS se debe instalar: jdk-7u4-windows-x64, jdk-7u4-windows-i586, installer_r20.0.3-windows, eclipse-jee-juno-SR1-win32-x86_64.

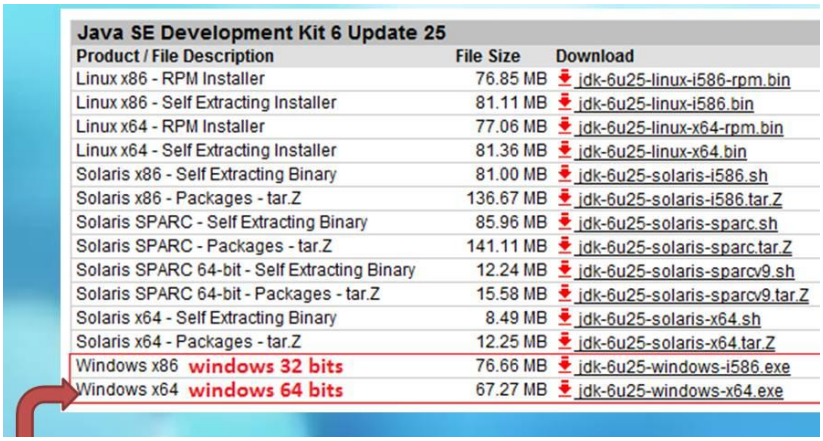
- Para la utilización de la aplicación móvil HablaSMS el teléfono celular debe tener instalado el motor de reconocimiento de voz para su correcto funcionamiento.
- Los datos que deben llenarse es obligatorio.

### 4.5.2 *DESCRIPCIÓN DEL SOFTWARE, APLICACIÓN MÓVIL HABLASMS*

El software, aplicación móvil HablaSMS el funcionamiento es mediante comandos de voz con respecto al envío y recepción de mensajes de texto, mensajes de voz.

### 4.5.3 *INSTALACIÓN DE Y CONFIGURACIÓN DE JDK-7U4-WINDOWS-X64*

**Paso 1:** Lo primero que se hace es descargar JDK de acuerdo a las características de hardware y software en este caso se descargó Windows x64 Windows 64 bits.



**Java SE Development Kit 6 Update 25**

| Product / File Description | File Size | Download |
|---|---|---|
| Linux x86 - RPM Installer | 76.85 MB | jdk-6u25-linux-i586-rpm.bin |
| Linux x86 - Self Extracting Installer | 81.11 MB | jdk-6u25-linux-i586.bin |
| Linux x64 - RPM Installer | 77.06 MB | jdk-6u25-linux-x64-rpm.bin |
| Linux x64 - Self Extracting Installer | 81.36 MB | jdk-6u25-linux-x64.bin |
| Solaris x86 - Self Extracting Binary | 81.00 MB | jdk-6u25-solaris-i586.sh |
| Solaris x86 - Packages - tar.Z | 136.67 MB | jdk-6u25-solaris-i586.tar.Z |
| Solaris SPARC - Self Extracting Binary | 85.96 MB | jdk-6u25-solaris-sparc.sh |
| Solaris SPARC - Packages - tar.Z | 141.11 MB | jdk-6u25-solaris-sparc.tar.Z |
| Solaris SPARC 64-bit - Self Extracting Binary | 12.24 MB | jdk-6u25-solaris-sparcv9.sh |
| Solaris SPARC 64-bit - Packages - tar.Z | 15.58 MB | jdk-6u25-solaris-sparcv9.tar.Z |
| Solaris x64 - Self Extracting Binary | 8.49 MB | jdk-6u25-solaris-x64.sh |
| Solaris x64 - Packages - tar.Z | 12.25 MB | jdk-6u25-solaris-x64.tar.Z |
| Windows x86 windows 32 bits | 76.66 MB | jdk-6u25-windows-i586.exe |
| Windows x64 windows 64 bits | 67.27 MB | jdk-6u25-windows-x64.exe |

Ilustración33: Descargar JDK,

Autores: Mercedes Guamán, Alfonso Caguana

Para seleccionar la versión debe aceptar el reglamento y cliclear para descargarlo.

Al instalar debe estar dentro del disco duro C:/Program Files/Java se debe instalar normalmente como cualquier otro programa siguiente, siguiente, siguiente.

**Paso 2 configurar entorno:** Este paso la variable de entorno sirve únicamente para que desde la consola de Windows pueda compilar y ejecutar archivos java.

Para agregar una variable de entorno deben ir a:

Windows Seven >Panel de control > Sistema > Configuración avanzada del sistema > Variables de entorno.

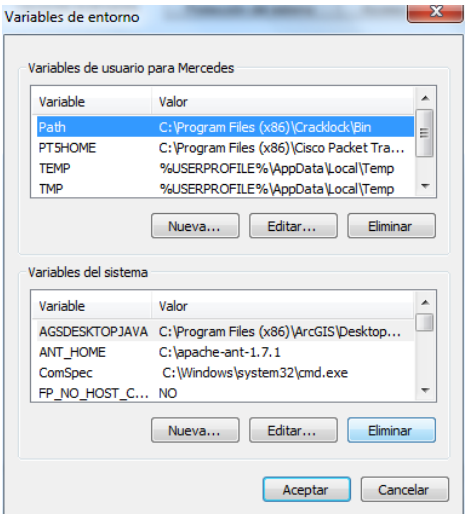



Ilustración 34: Pantalla de Variables de entorno,

Autores: Mercedes Guamán, Alfonso Caguana

En este paso se crear una nueva variable de entorno presionando en el botón "Nueva", como muestra la imagen donde especifica el nombre de la variable y el valor de la variable. El nombre recomendado es "JAVA_HOME". Al presionar aceptar verán la nueva variable dentro de la lista de "Variables del Sistema" tal cuál figura en la imagen.

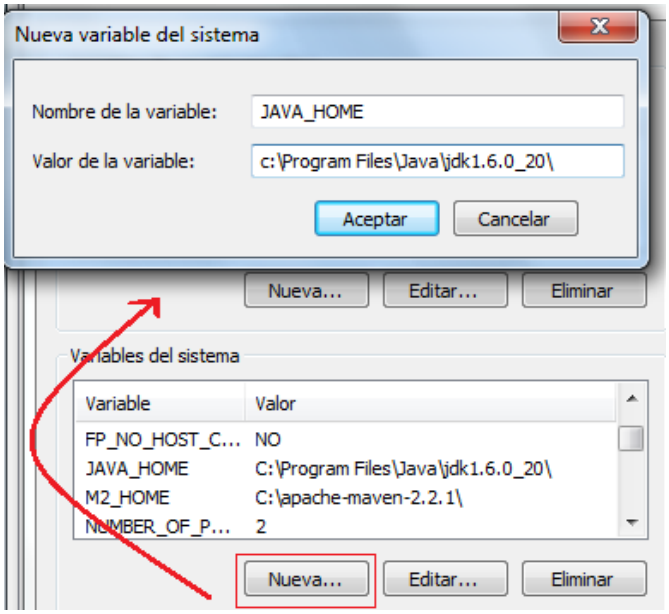



Ilustración 35: Nueva Variable del entorno,

Autores: Mercedes Guamán, Alfonso Caguana

Por último se modificala variable de sistema ya existente "Path", para ello seleccionar la variable de la lista de variables y presionamos el botón "Editar". Esto abrirá una ventana emergente donde puede ver el valor de la variable y editarlo.

Dirigir al final de la variable y agregamos lo siguiente "**; %JAVA_HOME%bin;**", siendo "JAVA_HOME" el nombre de la variable que se agregó anteriormente. Presionar aceptar en ambas pantallas y finalizar.

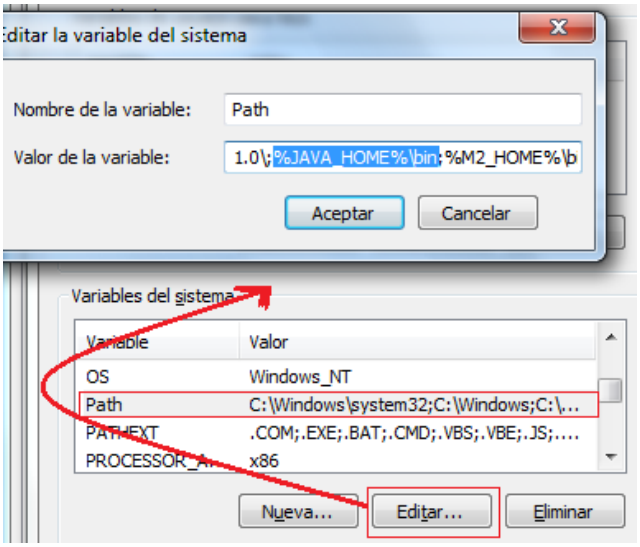



Ilustración 36: Editar la variable existente,

Autores: Mercedes Guamán, Alfonso Caguana

Para comprobar que la variable de entorno haya sido creada satisfactoriamente realizar lo siguiente:

Abrir la consola DOS de Windows para verificar que se realizó los cambios correctos.

Presionamos **Tecla Windows + R** escribimos "**cmd**" y presionar **enter**

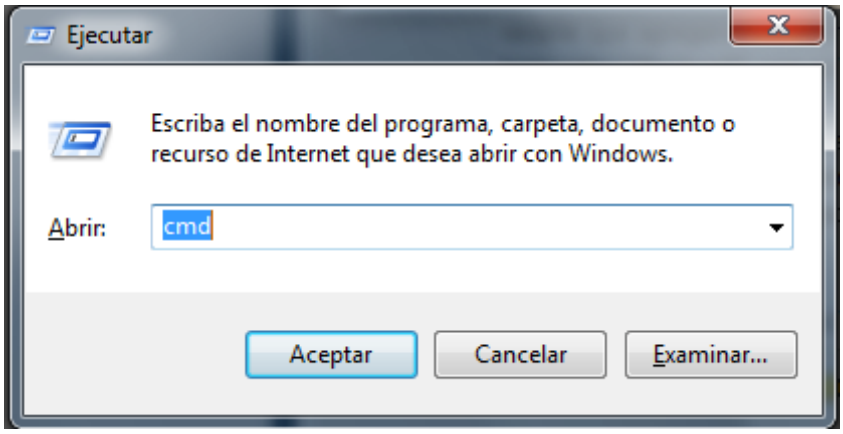



Ilustración 37: Pantalla para ingresar a CMD,

Autores: Mercedes Guamán, Alfonso Caguana

Digitar javac y enter

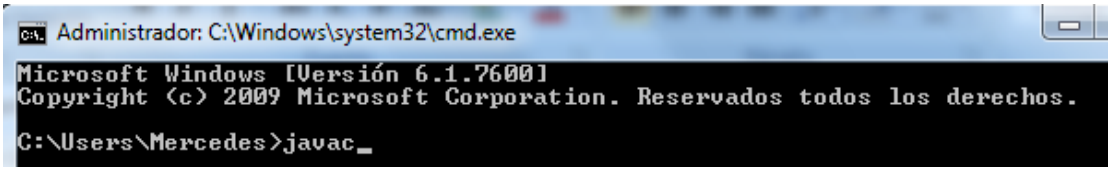



Ilustración 38: Pantalla cmd comprobación1,

Autores: Mercedes Guamán, Alfonso Caguana

Si esta correcto las modificaciones de las variables aparece la siguiente información.

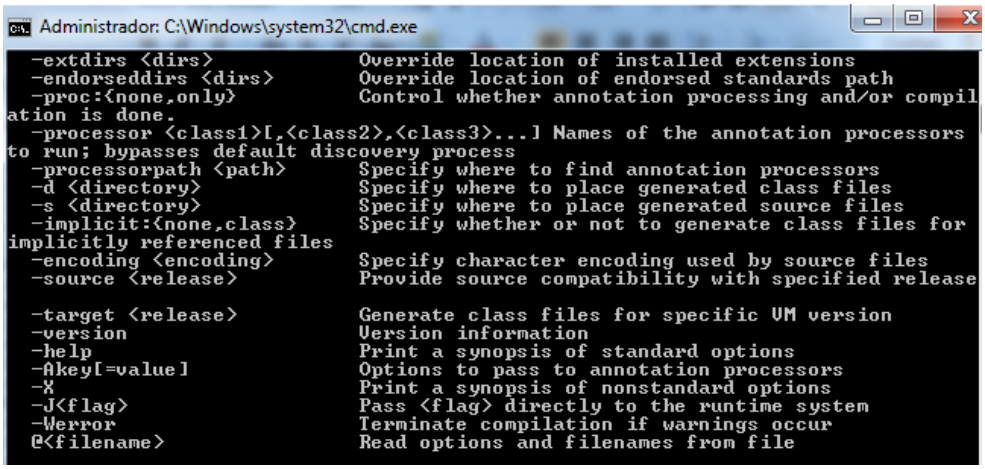



Ilustración 39: Pantalla comprobación2,

Autores: Mercedes Guamán, Alfonso Caguana

### 4.5.4 *INSTALACIÓN Y CONFIGURACIÓN ECLIPSE PARA DESARROLLO APLICACIONES ANDROID.*

**Paso 1: Instalar Eclipse**

El entorno Eclipse se puede descargar desde el sitio web de descargas de Eclipse. La versión Eclipse IDE for Java Developers es una buena opción. La instalación consiste en descomprimir el fichero .zip descargado. Esto crea una carpeta llamada eclipse dentro de la cual hay, entre otros, el programa Eclipse, el cual inicia el entorno.

**Paso 2: Instalar el SDK Android.**

El paquete SDK se puede descargar en un fichero .zip o en una versión de instalador (solo para Windows). Si se descarga un fichero .zip, la instalación consiste únicamente en descomprimirlo (se descomprime automáticamente en el directorio android-sdk-<machine-platform>, donde <machine-platform> es el nombre de la plataforma donde se instala). Si se descarga la versión de instalador para Windows únicamente se debe ejecutar el programa y seguir las indicaciones del mismo.

**Paso 3: Instalar el Plugin ADT para Eclipse**

Para instalar el plugin ADT se deben a realizar las siguientes acciones:

- Iniciar Eclipse y seleccionar del menú help la opción de instalación de nuevo software (menú help>install New software...).
- Hacer clic en el botón Add… y aparecerá un cuadro de diálogo en el que solicita introducir el nombre del repositorio donde está el plugin (introducir aDtplugin) y la URL de localización (introducir https://dl-ssl.google.com/android/eclipse/).
- El paso anterior origina que en el cuadro de diálogo aparezca una lista de software instalable de ese repositorio. En concreto solo aparece un paquete. Se selecciona el único que aparece haciendo clic en la correspondiente check box (Developer tools) y con el botón next avanza al siguiente cuadro de diálogo.
- En la nueva ventana se muestra una lista de herramientas que van a ser descargadas. Hacer clic en el botón next y a continuación aceptar los acuerdos de licencia. Por último hacer clic en el botón finish finalizando el proceso de instalación. Una vez que la instalación ha sido completada es necesario restaurar Eclipse para que la nueva instalación tenga efecto.

Si ya se tiene instalado el plugin ADT conviene hacer actualizaciones cada cierto tiempo de las versiones más recientes. Para ello se utiliza el gestor de actualizaciones de Eclipse (Update Manager, accesible mediante el menú help>check for Update).

**Paso 4: Configurar el plugin ADT**

En este paso se configuran algunas preferencias del plugin. En concreto se debe asociar el plugin con el SDK de Android. Para ello seleccionar el menú eclipse >preferences y en el panel de la izquierda seleccionar android. A continuación, en el panel principal se podrá indicar mediante el botón browse la localización del SDK que fue descargado en su momento. Por último, hacer clic en apply y oK.

**Paso 5: Añadir Componentes**

El SDK de Android está organizado en varias partes: versiones de plataformas Android, herramientas, ejemplos y documentación, las cuales se pueden instalar separadamente. El paquete SDK instalado que ha sido descargado incluye solamente las herramientas. Para desarrollar una aplicación Android necesita descargar e instalar la plataforma Android. Este proceso se realiza mediante el gestor AVD (Android Virtual Device, Dispositivo Virtual Android). El AVD se puede ejecutar de varias formas. Una de ellas es desde Eclipse con el menú Windows>Android SDK manager. Este menú proporciona la ventana del gestor de AVD y muestra todos los componentes del SDK que se puede descargar e instalarse en el SDK local de la máquina en la que se trabaja. Los componentes básicos son SDK Tools, SDK Platform Tools y SDK platform.

# CAPÍTULO V

# CONCLUSIONES Y RECOMENDACIONES

## 5.1 CONCLUSIONES

- Con la investigación de la arquitectura de aplicaciones móviles basados en el entorno de programación Java se concluye que es factible el desarrollo de una aplicación móvil siguiendo la arquitectura MVVM en este caso la aplicación Habla SMS que consiste en la transformación de texto a voz y viceversa que será para el uso de conductores de vehículos de forma segura en especial en la escuela de conducción de la Universidad Nacional de Chimborazo aportando en la seguridad y eficiencia del conductor
- El análisis de la arquitectura de las aplicaciones móviles basadas en el entorno de programación Java se concluye que, para lograr el propósito del desarrollo una aplicación para dispositivos móviles con sistema operativo Android se trabaje sobre el entorno de Eclipse junto al SDK que es una de las herramientas más de Java.
- Existen soluciones en el mercado que nos ayudan en la creación de aplicaciones móviles. Cada uno de ellos utiliza diferentes técnicas y conducen a resultados diferentes, pero la característica común de todos ellos es que utilizan tecnologías web para crear las aplicaciones.
- En el caso de desarrollo de la aplicación HablaSMS, se sigue un modelo bastante novedoso en el proceso de Ingeniería de software como es el Modelo Open UP que es un modelo para desarrollo de aplicaciones ágiles, que contiene las 4 fases Concepción, Elaboración, Construcción y Ejecución ya que es un modelo que se asemeja al modelo espiral clásico para desarrollos de software, con la diferencia que este modelo evita diagramas, documentación e iteraciones innecesarios en el proceso de desarrollo, y que conforme avanza el desarrollo vamos realizando las pruebas correspondientes no necesariamente tenemos que finalizar el desarrollo para realizar las correcciones del caso.

- El proceso de aprendizaje de la herramienta Eclipse es lento en el comienzo. Uno puede encontrar muchos problemas, mientras se empieza a utilizar, pero una vez que sabemos cómo usarlo, el tiempo de desarrollo de una aplicación se disminuye.
- Realmente es muy confortable que se pueda analizar, investigar y ser parte del desarrollo de una aplicación móvil, desde su fase de análisis, diseño, codificación e implementación, y desarrollar una propia aplicación y demostrar que realmente se puede desarrollar aplicaciones móviles para clientes de telefonía móvil.

## 5.2 **RECOMENDACIONES**

- En base a la investigación de la arquitectura de las aplicaciones móviles debe tomar en cuenta el factor principal la definición de los alcances de que desea que tenga una aplicación y de esa manera establecer la arquitectura adecuada para el desarrollo de la aplicación móvil.
- En cuanto a la aplicación como las aplicaciones móviles diseñadas para Android están relacionadas con Google se deben establecer parámetros de conexión a internet o contratar el servicio de paquete de datos para interactuar con la aplicación móvil HablaSMS
- Además cabe mencionar que la aplicación sólo funciona de manera segura y eficiente con la conexión a internet, porque se está utilizando el motor de texto hablado de google que es una herramienta el cual nos permite interactuar con el dispositivo.
- Se recomienda que a las futuras generaciones promover e incentivar que se involucren en el desarrollo de aplicaciones móviles ya que es un campo bastante provechoso y por ende que el entorno de vida está basado en la independencia de sitio (movilidad continua).

# CAPITULO VI

## BIBLIOGRAFÍA


- ✓ CatalunyaRa-Ma. (2011). Análisis de tecnologías para aplicaciones en dispotivos móviles
- ✓ GranadaMarcombo. (01/10/2011). Android: Programación de dispositivos móviles
- ✓ BarcelonaISBN. (2012). Android: Programacion de dispositivos móviles a través de ejemplos.
- ✓ EspañaDías de Santos. (2004). Avances en Criptografía y Seguridad de la Información.
- ✓ Washinton. (2004). Colisión y atropellamiento po vehículo.
- ✓ De Aristóteles a Ptolomeo. (29/01/2008). Universidad de Cienfuegos Francisco A. Ruiz Martínez
- ✓ Handheid Usabillity. (2002). Dispositivos Móviles y Multimedia.
- ✓ CaribeCEPAL. (2007). Evolución de la tecnología móvil.
- ✓ MálagaISBN. (2003). J2ME Java 2 Micro Edition.
- ✓ United States of AmericaPrentice Hall. (2012). Java Application Architecture: Modularity Patterns with Examples Using OSGi.
- ✓ Marcombo. (1998). Telecomunicaciones MóvilesBarcelona ISBN
- ✓ 2011Modelo Vista Controlador
- ✓ Chile Jurídica de Chile. (1994). Regimen del transito leyes vigentes.
- ✓ Tecnologi@ y Desarrollo. (2006). Sistema de Adquisición y Reconocimento de la Señal de Voz Aplicaciones Java.

- ✓ EspañaARÁN. (2009). Técnico en emergencias, Teleemergencias.
- ✓ EspañaISBN. (2008). Tecnología Móvil.
- ✓ Barcelona ISBN. (2008). Tecnologías del texto y del habla
- ✓ Thefreakshowoct. (2012).Lenguaje de Programación y Frameworks Dispositivos Móviles 3 páginas.

# CAPITULO VII
# ANEXOS

## 7.1 GLOSARIO DE TÉRMINOS

3GPP = (Third Generation Partnership Project).

ADT= (Android Development Tools) Herramienta de desarrollo para Android

ADT= (Android Development Tools) Herramientas de Desarrollo de Android

API= (Applications Programming Interface) Interfaz de Programación de Aplicaciones

ARM= Arquitectura de Procesos

AVD= (Android Virtual Device) Dispositivo Virtual de Android.

AWT= (Abstract Windows Toolkit) Herramientas Abstractas de Windows.

CDMA= (Codes Divided Multiple Acces) Acceso Múltiple por División de Códigos

CLDC= (Connected Limited Device Configuration)

Dalvik = Máquina Virtual de Android.

DC= (Device Container) Contenedor de Dispositivo

DII= (Device Integration Interfaces) Interfaz de Integración de Dispositivos.

ECA= (Event-Condition-Action) Acción Condición Evento

EMS= (Enhaced Messaging Service) Servicio de mensajería mejorado

ESN= (Electronic Serial Number) Serie de numerous electrónicos

FP= (Feature Pack)

GPRs= (General Packets Radio Services) Servicio General de Paquetes via Radio

GPS= (Glogal Position Systema) Sistema de Posicionamiento Global.

GSM= (Global System Mobil) Sistema Global para las comunicaciones móviles.

GUI = (Graphic User Interface) Interfaz Gáfica de Usuario.

HMM= Hide Markov Models Modelos Ocultos de Markov.

IDE= Entorno de Desarrollo Integrado

IDE= Entorno de Desarrollo Integrado.

JDE= (BlackBerry Java Development Enviroment) Entorno de programación de Java para BlackBerrry.

JRE= Java Run Time Enviroment

JS= Java Script

JVM= (Java Virual Machine) Máquina Virtual de Java.

KVM= (Kilo Virtual Machine)

MAN= (Mobile Access Number) Número de acceso móvil.

MIDP= (Mobile Information Device Profile)

OHA = Open Handset Alliance

PDA= Agendas Personales

PDC= (Personal Digital Celular) Celular Digital Personal.

RAPC= Compilador de línea de comando.

RIM= (Research In Motion)

SDK= (Software Development Kit) Herramientas de Desarrollo de Software.

SMS= (Short Message System).

SSL: Proporciona servicios de encriptación *Secure Socket Layer*

TDMA= (Time Divide Multiple Acces) Acceso Múltiple por división de Tiempo.

UI= (User Interface) Interfaz de Usuario.

WIM= (Wireless Instant Messaging)

## 7.2 FICHA TÉCNICA DE HARDWARE

**Tabla 21: Ficha técnica de hardware,** **Autores:**

**Mercedes Guamán, Alfonso Caguana**

| Hardware | Descripción | Características |
|---|---|---|
| **Laptop** | • Laptop es un computador personal móvil, realiza tareas de las computadoras de escritorio con la gran ventaja de su movilidad. | • Procesador Intel(R) Core(TM) i3 CPU M 370<br>• Sistema operativo de 64 bits<br>• Memoria RAM 4GB |
| **Teléfono celular**<br>**Smartphone** | • Android es un sistema operativo creado por Google para los dispositivos móviles, que convierte a un dispositivo móvil en un ordenador de bolsillo, se puede navegar por Internet, instalar más de 80.000 aplicaciones del Android Market (como Gmail, Pandora o Facebook), jugar a videojuegos, escuchar música, ver vídeos, enviar mensajes de texto y realizar llamadas**.** | • Sistema operativo Android mínimo 2.2 y un máximo de 4 que son las versiones de android. |

## 7.3 FICHA TÉCNICA DE SOFTWARE

**Tabla 22: Ficha técnica de software,** **Autores:**

**Mercedes Guamán, Alfonso Caguana**

| Hardware | Descripción | Características |
|---|---|---|
| jdk-7u4-windows-x64 | El software Java te permite ejecutar aplicaciones denominadas "applets" que están escritas en el lenguaje de programación Java. La tecnología Java Plug-in, incluida como parte del Runtime Enviroment de Java 2, Standard Edition (JRE), establece una conexión entre los navegadores más populares y la plataforma Java. | Tamaño de fichero es de 31.59MB (33,119, 648 bytes).<br>Licencia freeware<br>Windows 8<br>Windows 7<br>Vista<br>Windows XP<br>Windows Server 2008 |
| jdk-7u4-windows-i586 | El software Java te permite ejecutar aplicaciones denominadas "applets" que están escritas en el lenguaje de programación Java | Tamaño de fichero es de 30.20 MB (31,666, 592 bytes).<br>Licencia freeware<br>Windows 8<br>Windows 7<br>Vista<br>Windows XP<br>Windows Server 2008. |
| eclipse-jee-juno-SR1-win32-x86_64 | Herramientas para desarrolladores de Java EE y Java que crean aplicaciones Web, incluyendo un IDE Java, herramientas para Java EE, JPA, JSF, Mylyn, EGit y otros. | |

## 7.4 DISEÑO DE LAS ENCUESTAS

**ENCUESTA**

1. **Conoce la ley de transito con respeto a la utilización del teléfono móvil mientras conduce?**

   **Si** ☐

   **No** ☐

2. **Sabe usted que el mayor índice de accidentes de tránsito se debe a la utilización del teléfono móvil mientras conduce?**

   **Si** ☐

   **No** ☐

3. **Utiliza el teléfono móvil mientras conduce?**

   **Si** ☐

   **No** ☐

4. **Al recibir y enviar un mensaje sms a su teléfono móvil utiliza las manos para responder o leer?**

   **Si** ☐

   **No** ☐

5. **Utiliza dispositivos de manos libres en su teléfono móvil?**

   **Si** ☐

   **No** ☐

6. **Utiliza aplicaciones móviles en su teléfono celular?**

   **Si** ☐

   **No** ☐

7. **Le gustaría utilizar una aplicación móvil que transforme su mensaje sms a mensaje de voz y viceversa para no utilizar las manos en su totalidad al conducir el vehículo?**

   **Si** ☐

   **No** ☐