

UNIVERSIDAD TÉCNICA DE MACHALA
UNIDAD ACADEMICA DE INGENIERÍA CIVIL
CARRERA DE INGENIERÍA DE SISTEMAS

TRABAJO DE TITULACIÓN

PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERA DE SISTEMAS

TEMA:

**“DESARROLLO E IMPLEMENTACIÓN DE UNA APLICACIÓN
MÓVIL PARA LA GESTIÓN Y SOPORTE DE LAS ACTIVIDADES DE
CAMPO REALIZADAS POR EL PERSONAL OPERATIVO DE
CONSTRUCCIONES E INSTALACIONES ELÉCTRICAS (CONIEL
CIA.LTDA.)”**

AUTORA:

LOAIZA GONZAGA ANDREA ANABELL

TUTOR:

ING. FAUSTO REDROVÁN CASTILLO, MG. SC.

CO-TUTOR

ING. JOFFRE CARTUCHE VARGAS

MACHALA - EL ORO - ECUADOR

2014

CERTIFICACIÓN DEL TUTOR

Ing. FAUSTO REDROVÁN CASTILLO, Mg. Sc, Profesor de la Unidad Académica de Ingeniería Civil de la Universidad Técnica de Machala, en calidad de Tutor de la Tesis de Grado titulada: “DESARROLLO E IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL PARA LA GESTIÓN Y SOPORTE DE LAS ACTIVIDADES DE CAMPO REALIZADAS POR EL PERSONAL OPERATIVO DE CONSTRUCCIONES E INSTALACIONES ELÉCTRICAS (CONIEL CIA.LTDA.)”, elaborado por la estudiante ANDREA ANABELL LOAIZA GONZAGA, egresada de la carrera de Ingeniería de Sistemas de la Escuela de Informática, certifico que la mencionada Tesis estuvo bajo mi dirección y supervisión ajustándose a los procedimientos académicos y metodológicos establecidos por la Facultad, razón por la que autorizo su presentación para el trámite legal correspondiente.

Ing. Fausto Redrován Castillo, Mg. Sc.

CERTIFICACIÓN DEL CO-TUTOR

Ing. JOFFRE CARTUCHE VARGAS, Profesor de la Unidad Académica de Ingeniería Civil de la Universidad Técnica de Machala, en calidad de Co-Tutor de la Tesis de Grado titulada: “DESARROLLO E IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL PARA LA GESTIÓN Y SOPORTE DE LAS ACTIVIDADES DE CAMPO REALIZADAS POR EL PERSONAL OPERATIVO DE CONSTRUCCIONES E INSTALACIONES ELÉCTRICAS (CONIEL CIA.LTDA.)”, elaborado por la estudiante ANDREA ANABELL LOAIZA GONZAGA, egresada de la carrera de Ingeniería de Sistemas de la Escuela de Informática, certifico que la mencionada Tesis estuvo bajo mi dirección y supervisión ajustándose a los procedimientos académicos y metodológicos establecidos por la Facultad, razón por la que autorizo su presentación para el trámite legal correspondiente.

Ing. Joffre Cartuche Vargas

DECLARACIÓN DE AUTORÍA

El desarrollo y levantamiento de información, expuesta en este trabajo de investigación denominado: **“DESARROLLO E IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL PARA LA GESTIÓN Y SOPORTE DE LAS ACTIVIDADES DE CAMPO REALIZADAS POR EL PERSONAL OPERATIVO DE CONSTRUCCIONES E INSTALACIONES ELÉCTRICAS (CONIEL CIA.LTDA.)”**, es de exclusiva autoría y responsabilidad de Andrea Anabell Loaiza Gonzaga con C.I.Nº 070569847-0.

Andrea Anabell Loaiza Gonzaga

AGRADECIMIENTO

A Dios por iluminar mi camino, dame fuerzas para vencer los obstáculos y culminar mi carrera.

A mis padres dignos de ejemplo, trabajo y constancia quienes me han brindado su amor incondicional, han estado en todos los momentos de mi vida apoyándome y alentándome a seguir adelante.

A una persona especial cuyo respaldo y ánimos han sido decisivos en momentos de angustia y desesperación.

A mi tutor de tesis Ingeniero Fausto Redrován, quien durante todo este tiempo me colaboró en el desarrollo de este trabajo.

A mis profesores que han sido participes en mi formación académica dentro de esta prestigiosa Institución.

Agradecer a todas aquellas personas que en mayor o menor medida han ayudado a que este trabajo se desarrolle.

Andrea Loaiza

DEDICATORIA

Quiero dedicar este trabajo en primer lugar a Dios, por ser quien guía mi vida y por darme la fortaleza para poder alcanzar esta meta.

A mis padres Daniel Loaiza y Herminia Gonzaga, quienes han sido el pilar fundamental de mi vida, por su apoyo, consejos, comprensión y por ayudarme con los recursos necesarios para estudiar.

A mis hermanos por estar siempre presentes, acompañándome para poderme realizar y por enseñarme a encarar las adversidades sin desfallecer en el intento.

Andrea Loaiza

“DESARROLLO E IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL PARA LA GESTIÓN Y SOPORTE DE LAS ACTIVIDADES DE CAMPO REALIZADAS POR EL PERSONAL OPERATIVO DE CONSTRUCCIONES E INSTALACIONES ELÉCTRICAS (CONIEL CIA.LTDA.)”

Andrea Anabell Loaiza Gonzaga

Resumen

La Compañía de construcciones e instalaciones eléctricas Coniel Cia.Ltda ha brindado un buen servicio a la comunidad en el área de ingeniería eléctrica, desempeñándose en estos últimos años bajo la adjudicación de contratos de control de pérdida de energía por la CNEL EP.

En la empresa el personal operativo ejecuta procesos manuales para el tratamiento y acceso a la información, por lo que se genera cierto retraso en las actividades que realizan diariamente.

Esto me ha motivado a poner en funcionamiento una propuesta innovadora desarrollada para la gestión de las actividades realizadas por el personal operativo con el fin de desarrollar e implementar una aplicación móvil que permita acceder a la información desde el sitio de trabajo y optimizar el tratamiento de los datos.

Esta aplicación interactiva, rápida y funcional ha sido desarrollada siguiendo los fundamentos de gestión de proyectos de la metodología ágil Mobile-D, junto con tecnologías como el Sistema Operativo Android, el lenguaje de programación Java y con acceso a un servidor con base de datos PostgreSQL.

Por lo que la aplicación se convirtió en gran apoyo para la compañía ya que aceleró la ejecución de sus procesos.

Palabras Clave: Aplicación móvil, acceso a la información, procesos manuales, retraso.

Abstract

INTRODUCCIÓN

La utilización inadecuada de las nuevas tecnologías de información, genera retraso en los procesos productivos dentro de las empresas u organizaciones, ya que el tratamiento de la información se la realiza mediante tareas manuales.

En la actualidad el avance tecnológico en el ámbito empresarial ha permitido que las organizaciones tengan una mayor rentabilidad, agilidad en la ejecución de sus procesos, rápido y fácil acceso a la información y la transparencia en los servicios o productos que ofertan.

La información es el activo más importante dentro de una organización por lo que debe ser confiable, integra y de fácil acceso.

Grandes empresas como Coca-Cola que cuentan con personal operativo refiriéndose a los trabajadores encargados de tomar pedidos desde los puntos de venta al cliente (tiendas, bares, etc.), cuentan con herramientas que facilitan la transferencia de información en tiempo real, ayudando de esta manera a agilizar los procesos y brindando fácil acceso a la información desde cualquier punto.

Las empresas que se dedican a brindar servicios específicamente de control de pérdida de energía a nivel nacional, no utilizan herramientas tecnológicas para realizar actividades operativas de campo y además el personal encargado no cuenta con la capacitación requerida para usar nuevas tecnologías, por lo que en la compañía de Construcciones e Instalaciones Eléctricas CONIEL CIA.LTDA aún utilizan métodos tradicionales como: el uso de formatos impresos para el control de las actividades operativas, la utilización de cámaras para la captura de fotografías concernientes al trabajo realizado, y la comunicación a través de radios o teléfonos celulares que en muchas veces no cuentan con la claridad necesaria para el intercambio de información eficiente.

Para mejorar el desarrollo de las actividades que realizan las compañías dedicadas a los trabajos de control de pérdida de energía, es necesario implementar nuevas técnicas o mecanismos para optimizar el uso de tecnologías como Aplicaciones móviles que permitan cubrir las falencias que se han venido dado en la ejecución de los trabajos.

Para el desarrollo e implementación de la aplicación móvil se utilizó la plataforma Android aplicando la metodología ágil Mobile-D y con una base de datos POSTGRESQL para el almacenamiento de la información.

JUSTIFICACIÓN

Los procesos automatizados son factores de vital importancia en cualquier empresa. El desarrollo de las tecnologías de información ha permitido la evolución de los dispositivos móviles capaces de manejar sistemas tecnológicos en distintas áreas.

Las actividades diarias desarrolladas en el campo por el personal operativo de CONIEL CIA.LTDA, requieren que la información sea procesada y almacenada de manera eficiente, hecho que agilizará los procesos de otras actividades.

Con el desarrollo de una aplicación que se ejecute desde un dispositivo móvil se resolverán las necesidades y requerimientos de la empresa, además se logrará un control integral de las actividades y se optimizará el procesamiento de los datos.

La portabilidad de los dispositivos móviles facilitan su traslado al lugar del trabajo, además esto incrementará la confiabilidad de la información, mediante la disminución de pérdida o alteración de los datos por el ejecución manual de este proceso.

Considerando lo señalado se propone denominar este proyecto como **“DESARROLLO E IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL PARA LA GESTIÓN Y SOPORTE DE LAS ACTIVIDADES DE CAMPO REALIZADAS POR EL PERSONAL OPERATIVO DE CONSTRUCCIONES E INSTALACIONES ELÉCTRICAS (CONIEL CIA.LTDA.)”**

Con esta aplicación se busca obtener una herramienta con opciones de tareas comunes realizadas por el personal operativo que necesitan soporte para un eficiente tratamiento de la información en lo posterior, haciendo uso de las nuevas tecnologías y del auge de las aplicaciones móviles para brindar un mejor desarrollo en las actividades de la compañía CONIEL CIA.LTDA.

1. MARCO REFERENCIAL

1.1 PLANTEAMIENTO DEL PROBLEMA

La compañía de Construcciones e Instalaciones Eléctricas CONIEL CIA.LTDA es una empresa dedicada a trabajos de Ingeniería Eléctrica, siendo su punto fuerte los contratos de pérdidas y control de energía adjudicados a través del Sercop (www.compraspublicas.gob.ec) por la CNEL EP Unidad de Negocios El Oro.

En CONIEL CIA.LTDA, las actividades operativas en el campo son realizadas por cuadrillas de trabajo conformadas por tres personas encargadas de trasladarse a lugares estratégicos o determinados anticipadamente por CONIEL para la realización de trabajos de electrificación correspondientes a control de pérdidas de energía y abastecimiento del servicio a zonas o abonados determinados, cada cuadrilla consta de un anotador, que es el encargado de capturar fotos a razón de justificar cada acción a tomar y llenar plantillas de datos (también llamadas fichas) con información relevante de las operaciones que se realicen, las mismas que pueden ser: servicios nuevos, cambios de medidor o mantenimientos del servicio dependiendo de las necesidades del cliente y lo notificado a realizar.

Dicha información al final de día es trasladada a la oficina central de la compañía para su posterior procesamiento, ingreso al Sistema SICO (Sistema Comercial de la CNEL EP) y tabulación.

1.1.1 PROBLEMA CENTRAL

La gestión de información actual desde el sitio de trabajo, genera retraso en los procesos productivos y disminuye el porcentaje de confiabilidad e integridad de la información registrada e ingresada en la empresa.

1.1.2 PROBLEMAS PARTICULARES

- En la empresa CONIEL CIA.LTDA las TIC's son desconocidas y no utilizadas por el personal.
- En ocasiones el personal operativo no cuenta con información necesaria para el registro de la actividad en la ficha (formato para el registro de la actividad realizada) desde el lugar de trabajo y requiere de ayuda del personal administrativo.
- La toma de Información en el sitio muchas veces es inconsistente e incompleta.
- Los procesos realizados por el personal operativo son netamente manuales.
- La empresa posee computadoras potentes que pueden ser utilizadas para la implementación de un servidor, pero únicamente son usadas para la tabulación, búsqueda y registro de la información.
- La empresa no cuenta con la información de ubicación de cada cuadrilla a lo largo del día.

1.2 PREGUNTAS CIENTIFICAS

1.2.1 PREGUNTA CENTRAL

¿La implementación de la aplicación móvil para la optimización de las actividades realizadas por el personal operativo permitirá gestionar la información desde el sitio de trabajo y brindará mejoras en los procesos productivos de la compañía CONIEL CIA.LTDA?

1.2.2 PREGUNTAS COMPLEMENTARIAS

- ¿Con la capacitación al personal de la empresa Coniel Cia. Ltda. se logrará un mejor uso de las TIC's?
- ¿Se logrará mejorar el acceso a la información desde el sitio de trabajo?
- ¿Se mejorará la integridad y confiabilidad de la información?
- ¿La aplicación móvil permitirá llevar una gestión automatizada de los procesos que realiza el personal operativo?
- ¿Se implementará un servidor haciendo uso de la potencia del hardware que posee la empresa?
- ¿A lo largo del día se contará con la información de ubicación donde se encuentra cada cuadrilla?

1.3 OBJETIVOS

1.3.1 OBJETIVO PRINCIPAL

- Implementar una aplicación móvil para la gestión y soporte de las actividades de campo realizadas por el personal operativo de CONIEL Cia.Ltda utilizando la metodología Mobile-D”.

1.3.2 OBJETIVOS ESPECIFICOS

- Capacitar al personal con el fin de lograr la actualización de conocimientos en el campo de nuevas tecnologías mediante cursos impartidos por profesionales en el tema
- Dar soporte necesario de información al personal operativo en los distintos puntos de trabajo mediante un módulo de búsqueda de datos para de esta manera abolir el sistema actual basado en fichas y obtener una mejor confiabilidad de la información registrada.
- Diseñar una aplicación móvil para automatizar los procesos manuales realizados por el personal operativo mediante la herramienta de desarrollo Android Studio.
- Implementar un servidor interno de la empresa con el fin de centralizar la información mediante el uso objetivo de nuevas tecnologías.
- Brindar geolocalización de los grupos de trabajo para llevar un mejor control de los mismos mediante la implementación de un módulo de monitorización que se añadirá al servidor web interno.

1.4 ALCANCE

Previo al desarrollo del proyecto de tesis se realizará una investigación en la compañía de Construcciones e Instalaciones Eléctricas Coniel Cia.Ltda., poniendo énfasis a los procesos que realiza el personal operativo durante sus actividades diarias e identificar y analizar los problemas que tienen con el manejo de la información, para de esta manera integrarlos en una aplicación que permitirá un eficiente tratamiento de los datos desde el sitio de trabajo y mejorar la confiabilidad e integridad de la información.

La aplicación será diseñada de tal manera que sea intuitiva y de fácil manejo, para que el personal operativo pueda acceder a las opciones de forma sencilla, pudiendo así obtener mejores resultados en la ejecución de las actividades en tiempo real.

Para la realización del proyecto de titulación se utilizará un lenguaje de programación Java, el IDE Android Studio, base de datos PostgreSQL y dispositivos móviles con Sistema Operativo Android, además para un mejor control y gestión del proyecto se usará la metodología ágil Mobile-D que contemplará las siguientes acciones:

MÓDULO	FUNCIÓN	SUBFUNCIÓN	DESCRIPCIÓN
Login de la Aplicación	Ingreso de usuario y contraseña	Validación y Verificación de los datos	Valida y verifica los datos ingresados, la disponibilidad de la sesión, las sesiones activas en el servidor y finalmente da apertura a una sesión. También comprueba si el usuario que se encuentra realizando la petición de ingreso a la aplicación tiene acceso al contrato que ha seleccionado. La persona encargada de recolectar la información de la actividad realizada ingresa a la aplicación mediante un usuario y una contraseña y accede al menú principal.
		Comprobación de la identidad del usuario ingresado	
		Ingreso a la aplicación	
		Comprobación de Sesiones del servidor	
		Comprobación de sesiones disponibles	
		Apertura de Sesión	
		Comprobación de acceso a contrato	

Ingreso de Actividades	Ingreso de información de las actividades realizadas	Búsqueda, Validación y Verificación de datos de abonado y de información de medidores	El personal operativo puede ingresar información necesaria de la actividad realizada, así mismo puede buscar datos del abonado y de los medidores asignados a él en el sistema comercial SICO, además puede capturar las imágenes para constatar la ejecución de su trabajo. Puede buscar las actividades realizadas para verificar si ya han sido ingresadas.
		Búsqueda de actividades realizadas	
		Ingreso de detalle de instalación del servicio	
		Captura de fotografías de la actividad realizada	
Búsqueda	Búsqueda de información de abonados por criterio en el sistema comercial SICO.	Búsqueda de datos por cuenta (código único de abonado)	Este módulo realiza búsquedas por criterios al sistema comercial SICO, y provee información necesaria para el personal operativo.
		Búsqueda de datos por medidor	
		Búsqueda de datos por nombre de abonado	
		Búsqueda de datos por geocódigo	
Fotos	Capturar fotografías de las actividades realizadas	Ingreso de cuenta para almacenar las fotografías	Este módulo permite capturar las fotografías de la actividad realizada, así mismo seleccionar la fecha para crear una nueva cuenta en la que almacenará las imágenes o seleccionar una cuenta de la lista a la que desee añadir más fotos o eliminar alguna(s) de la galería. También puede seleccionar una imagen de la galería de la cuenta seleccionada para visualizar en tamaño real.
		Capturar imágenes	
		Selección de fecha y de cuenta para captura de imágenes	
		Añadir más imágenes a una cuenta seleccionada	
		Eliminar imágenes seleccionadas	
		Visualización de imágenes	

Geolocalización	Detectar ubicación del sitio en el que se encuentran las cuadrillas	Detectar ubicación actual de las cuadrillas	Esta sección de la aplicación permite tener un mejor control de la ubicación actual de las cuadrillas, además de conocer su propia ubicación y obtener la ruta más corta hacia una cuadrilla específica.
		Detectar mi ubicación	
		Ruta más cercana a una cuadrilla específica	

Tabla 1. *Detalle de alcance del proyecto*

CÁPITULO II

2. MARCO TEÓRICO

2.1 ANTECEDENTES HISTÓRICOS DE LA EMPRESA

2.1.1 RESEÑA HISTÓRICA DE CONIEL CIA.LTDA

La compañía de Construcciones e Instalaciones Eléctricas CONIEL CIA.LTDA, tiene sus inicios debido a la gran demanda de servicios eléctricos en nuestra Provincia. Fue constituida el 02 de julio del 2002, resultado de la asociación entre el Tnlg. Julio Loaiza y el Sr. Luis Pérez, para de esta manera dar paso al crecimiento de la empresa Privada cuyo único objetivo era ganar experiencia en el ámbito profesional y brindar servicios de calidad a la ciudadanía. Con el pasar de los años, la compañía es fuertemente constituida y cuenta con un amplio número de contratos firmados con la CNEL EP, los mismos que conforman su experiencia profesional. La empresa ha cumplido a cabalidad con las actividades de la prestación de los servicios adjudicado con la CNEL EP. La Compañía ha ido creciendo y ganando terreno en la Provincia y haciéndose acreedora a muy buenas referencias de trabajo por parte de la CNEL EP.

Actualmente es una compañía fuertemente establecida que brinda servicios a la comunidad trabajando en conjunto con la CNEL EP.

2.1.2 UBICACIÓN

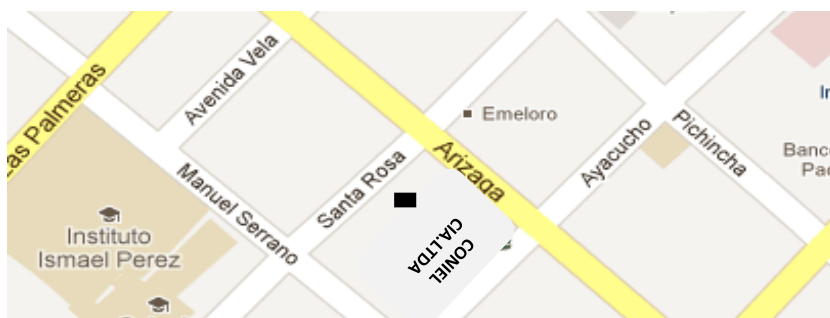


Figura 1. Croquis de ubicación de la Empresa Coniel Cía. Ltda.

Fuente: <https://www.google.com.ec/maps>

Dirección: Machala, Santa Rosa e/ Arizaga y Gral. Manuel Serrano.
Teléfono: 2938-581

2.1.3 ORGANIGRAMA ESTRUCTURAL

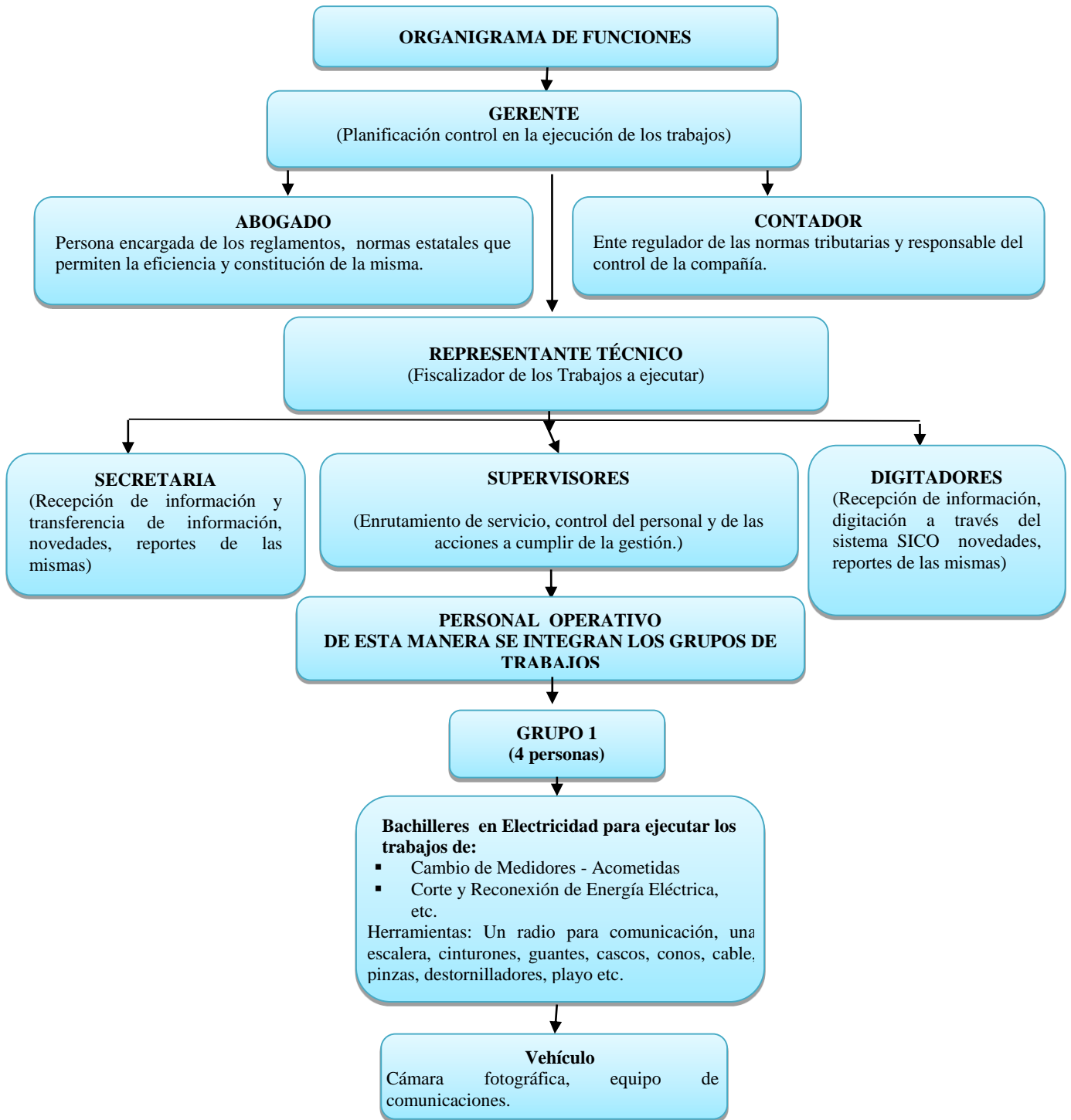


Figura 2. Estructura Organización de CONIEL Cia.Ltda.

Fuente: Imagen proporcionada por CONIEL CIA.LTDA

2.2 ANTECEDENTES CONCEPTUALES

2.2.1 TELEFONÍA MOVIL

2.2.1.1 Concepto

“La telefonía fija o móvil, es aquella que hace referencia a las líneas y equipos que se encargan de la comunicación entre terminales telefónicos, generalmente enlazados entre ellos con la central por medio de conductores metálicos.” (Procoop, 2010)

La telefonía móvil es un sistema que permite el acceso a un nuevo medio de comunicación con el objeto de proveer el contacto entre usuarios.

2.2.1.2 Historia

Según Martínez (2001):

Martin Cooper fue el pionero en esta tecnología, a él se le considera como "el padre de la telefonía celular" al introducir el primer radioteléfono en 1973 en los Estados Unidos mientras trabajaba para Motorola; pero no fue hasta 1979 en que aparece el primer sistema comercial en Tokio Japón por la compañía NTT (Nippon Telegraph & Telephone Corp.)

En 1981 en los países Nórdicos se introduce un sistema celular similar a AMPS (Advanced Mobile Phone System). Por otro lado, en los Estados Unidos gracias a que la entidad reguladora de ese país adopta reglas para la creación de un servicio comercial de telefonía celular, en octubre de 1983 se pone en operación el primer sistema comercial en la ciudad de Chicago. A partir de entonces en varios países se diseminó la telefonía celular como una alternativa a la telefonía convencional alámbrica. La tecnología inalámbrica tuvo gran aceptación, por lo que a los pocos años de implantarse se empezó a saturar el servicio, por lo que hubo la imperiosa necesidad de desarrollar e implementar otras formas de acceso múltiple al canal y transformar los sistemas analógicos a digitales para darle cabida a más usuarios. Para separar una etapa de la otra, a la telefonía celular se ha categorizado por generaciones. A continuación se describen cada una de ellas. (pág. 2)

2.2.1.3 Dispositivos Móviles

“Los dispositivos móviles son aparatos de pequeño tamaño, con algunas capacidades de procesamiento, con conexión permanente o intermitente a una red, con memoria limitada, diseñados específicamente para una función, pero que pueden llevar a cabo otras funciones más generales.” (Álvarez, 2008)

Son dispositivos que permiten acceder a los servicios que brinda la telefonía móvil o celular.

a) Características

Para Guevara (2010) son:

- Capacidades especiales de procesamiento

- Conexión permanente o intermitente a una red
- Memoria limitada
- Diseños específicos para una función principal y versatilidad para el desarrollo de otras funciones
- Movilidad, los dispositivos móviles son pequeños para poder portarse y ser fácilmente empleados durante su transporte.

b) Ventajas

- Movilidad.
- Conectividad.
- Funcionalidad. (Rabajoli, 2007)

2.2.1.4 Aplicaciones móviles

Es una aplicación de software que se instala en dispositivos móviles o Tablet, que ayudan a los usuarios con alguna labor en concreto, ya sea en el ámbito profesional o de entretenimiento. Son aplicaciones pensadas para facilitar la consecución de una tarea determinada o asistir en operaciones y gestiones del día a día.

2.2.2 SISTEMAS OPERATIVOS PARA DISPOSITIVOS MOVILES

Para Rivera (2012) :

El Sistema Operativo (SO) móvil de un teléfono o Tablet significa la interacción real con lo que podemos hacer a partir de las capacidades del hardware que conforman un equipo. A manera de traductor, esta plataforma interpreta lo que el usuario quiere que la terminal realice y cada vez, lo ejecuta con mayor inteligencia.

Una de las cualidades más atractivas de un sistema operativo móvil es la rapidez con la que en general se desempeña. No precisa apagar el equipo completamente, sino dejarlo en un estado de suspensión para ahorrar energía, las aplicaciones se lanzan en pocos segundos, la instalación es transparente para el usuario y muchos periféricos son actualmente compatibles con los dispositivos más comunes.

Tal pareciera que la única diferencia con una PC tradicional es que todavía no soportan aplicaciones robustas como podrían ser las enfocadas en diseño o edición de video profesional.

Un sistema operativo es un programa o conjunto de programas que gestiona recursos y brinda servicios a los programas.

2.2.2.1 Sistemas Operativos para móvil

Montoya (2012) explica los siguientes sistemas operativos para móvil:

Sistema Operativo	Descripción
Android	Es un sistema operativo para dispositivos móviles como teléfonos inteligentes y tabletas. Fue desarrollado inicialmente por Android Inc, una firma comprada por Google en 2005.
Symbian	Symbian es un sistema operativo que fue producto de la alianza de varias empresas de telefonía móvil. El objetivo de Symbian fue crear un sistema operativo para terminales móviles que pudiera competir con el de Palm o el Windows Phone de Microsoft y ahora Android de Google Inc. Y iOS de Apple Inc.
Ios	IOS es un sistema operativo móvil de Apple. La interfaz de usuario de iOS está basada en el concepto de manipulación directa, usando gestos multitáctiles. La interacción con el sistema operativo incluye gestos como deslices, toques, pellizcos, los cuales tienen definiciones diferentes dependiendo del contexto de la interfaz.
BlackBerry OS	El sistema permite multitarea y tiene soporte para diferentes métodos de entrada adoptados por RIM para su uso en computadoras de mano, particularmente la trackwheel, trackball, touchpad y pantallas táctiles.
Windows Phone	Anteriormente llamado Windows Mobile es un sistema operativo móvil compacto desarrollado por Microsoft, y diseñado para su uso en teléfonos inteligentes (Smartphone) y otros dispositivos móviles. Windows Phone hace parte de los sistemas operativos con interfaz natural de usuario.

Tabla 2. Sistemas Operativos para móvil

Fuente: <https://docs.google.com>

2.2.3 SISTEMA OPERATIVO ANDROID

2.2.3.1 Introducción

“Hace todavía poco tiempo, el sistema Android solo estaba presente en Smartphones. Más tarde, era posible encontrarlo también en televisores con conexión a Internet. Uno de los motivos de su éxito es, sin duda, la amplia oferta de aplicaciones disponibles para su descarga, que permiten a cualquiera personalizar su dispositivo Android.” (Sébastien, 2012)

2.2.3.2 Conceptos

Ribas (2013) afirma que:

Android es una plataforma de desarrollo libre, y de código abierto: El núcleo del sistema está basado en un Linux (versión 2.6 para versiones 3.0 del kernel para posteriores) al que se le han hecho ciertas modificaciones para que pueda ejecutarse en teléfonos y terminales móviles. Android es el nombre esencial para un sistema operativo enfocado al uso del mismo en dispositivos móviles, tomando en cuenta que al inicio se lo creó para ser usado solamente en teléfonos celulares. En la actualidad se puede encontrar a Android en todo tipo de dispositivos como tablets, Smartphone, netbooks, entre otros. (pág. 18)

Paredes Velasco, Santacruz Valencia y Domínguez Mateos (2012) consideran que:

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 (Software Development Kit, Kit de Desarrollo de Software) de Android proporciona varias herramientas y API (Applications Programming Interface, Interfaz de Programación de Aplicaciones) que son necesarias para desarrollar aplicaciones Android. (pág. 15)

Android es un sistema operativo para teléfonos móvil basado en el Kernel de Linux, de código libre, posee una amplia gama de aplicaciones disponibles. Permite el acceso a aplicaciones propias del sistema operativo como a (GPS, Cámara, Llamadas, Mensajes, etc), además de que da acceso a escritura y lectura a memoria tanto interna como externa.

2.2.3.3 Historia

Según Lara Cancela y Sara Ostos (2012):

Google adquirió Android Inc. en el 2005, en ese entonces la compañía se dedicaba a la creación de software para teléfonos móviles. Una vez dentro de Google, el equipo desarrolló una plataforma basada en el núcleo Linux para dispositivos móviles, que fue promocionado a fabricantes de dispositivos y operadoras con la promesa de proveer un sistema flexible y actualizable, Google adaptó su buscador y sus aplicaciones para el uso en móviles. Es el principal producto de la Open Handset Alliance, una alianza comercial de un conglomerado de compañías entre fabricantes y desarrolladores de hardware, software y operadores de servicio, dedicadas al desarrollo de estándares abiertos para dispositivos móviles, algunos de sus miembros son Google, HTC, Dell, Intel, Motorola, Qualcomm, Texas Instruments, Samsung, LG, T-Mobile, Nvidia y Wind River Systems. Google liberó la mayoría del código de Android bajo licencia Apache, una licencia libre y de código abierto.

2.2.3.4 Características

Para Ribas (2013) las características son las siguientes:

- Marco de aplicación que permite la reutilización y el reemplazo de los componentes.
- Dalvik optimizado para dispositivos móviles.
- Navegador integrado basado en la apertura del motor WebKit.
- Gráficos mejorados.

- SQLite para el almacenamiento de datos estructurados.
- Soporte para audio, vídeo, y formatos de imagen (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
- Telefonía GSM (dependiente del hardware).
- Bluetooth, EDGE, 3G, y Wi-Fi (dependiente del hardware).
- Cámara, GPS, brújula, y acelerómetro (dependiente del hardware). (pág. 19)

2.2.3.5 Arquitectura de Android

Ribas (2013) explica que:

La arquitectura del sistema operativo Android está formada por capas de software donde cada una puede utilizar los servicios de la capa inferior. Comenzando por la capa inferior se encuentra el conjunto de drivers basados en Linux, esta parte no es pública. Un nivel más arriba se encuentra un conjunto de librerías que no son accesibles directamente sino a través del nivel superior llamado Framework de aplicaciones y junto a la capa de aplicaciones son totalmente públicas, por lo tanto los usuarios pueden acceder libremente. Android es una plataforma para dispositivos móviles que contiene una pila de software donde se incluye un sistema operativo, middleware y aplicaciones básicas para el usuario. (pág. 42)

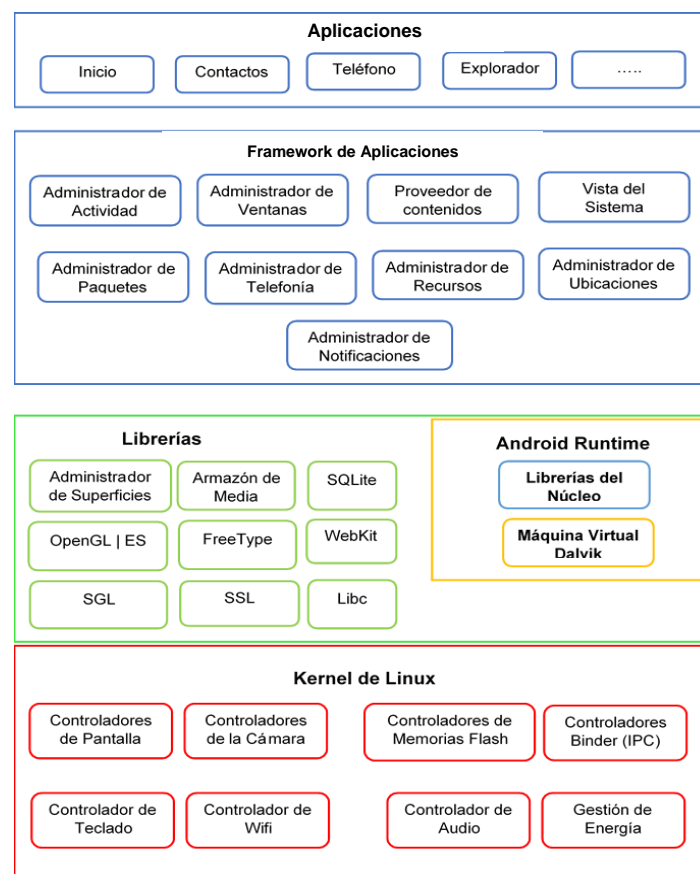


Figura 3. Arquitectura de Android

Fuente: Libro Desarrollo de Aplicaciones para Android 2013
Editado: Andrea Loaiza

Ribas (2013) explica cada una de las capas de la arquitectura de Android:

a) Aplicaciones

En la capa de aplicaciones es el lugar donde se incluyen todas las aplicaciones del dispositivo. Las aplicaciones básicas incluyen un cliente de email, programa de SMS, calendario, mapas, navegador, contactos, entre otros, las aplicaciones generalmente se encuentran escritas en lenguaje Java.

b) Framework de Aplicaciones

Esta capa se encuentra formada por las clases y servicios que utilizan las aplicaciones para realizar trabajo. La mayor parte de los componentes de esta capa son librerías Java que acceden a los recursos de las capas anteriores a través de la máquina virtual Dalvik.

Entre las API más importantes ubicadas aquí, se pueden encontrar las siguientes:

- **Activity Manager:** Conjunto de API que gestiona el ciclo de vida de las aplicaciones en Android.
- **Window Manager:** Gestiona las ventanas de las aplicaciones y utiliza la librería Surface Manager.
- **Telephone Manager:** Incluye todas las API vinculadas a las funcionalidades propias del teléfono (llamadas, mensajes, etc.).
- **Content Provider:** Permite a cualquier aplicación compartir sus datos con las demás aplicaciones de Android. Por ejemplo, gracias a esta API la información de contactos, agenda, mensajes, etc. será accesible para otras aplicaciones.
- **View System:** Proporciona un gran número de elementos para poder construir interfaces de usuario (GUI), como listas, mosaicos, botones, "check-boxes", tamaño de ventanas, control de las interfaces mediante teclado, etc. Incluye también algunas vistas estándar para las funcionalidades más frecuentes.
- **Location Manager:** Posibilita a las aplicaciones la obtención de información de localización y posicionamiento.
- **Notification Manager:** Mediante el cual las aplicaciones, usando un mismo formato, comunican al usuario eventos que ocurran durante su ejecución: una llamada entrante, un mensaje recibido, conexión Wi-Fi disponible, ubicación en un punto determinado, etc.
- **XMPP Service:** Colección de API para utilizar este protocolo de intercambio de mensajes basado en XML.

c) Las librerías nativas

Las librerías nativas son contenidas en la capa inmediata superior al kernel de Linux. Estas librerías compartidas están escritas en C o C++, fueron compiladas para la arquitectura de hardware utilizada por el dispositivo y preinstaladas en él por el proveedor del mismo.

Algunas de las más importantes son:

- **Surface Manager:** Es el manejador de ventanas compuesto para Android (similar a Metro, Vista o Compiz).

- **Gráficos en 2D y 3D:** Elementos de dos y tres dimensiones que pueden ser combinados en una sola interfaz de Android.
- **Media codecs:** Utilizada para grabación y reproducción de diversos formatos de medios (como AAC, AVC, H.564, MP3, MP4, etc.)
- **Base de datos SQL:** Android incluye el motor de base de datos **SQLite** que puede ser utilizado para almacenar datos en el dispositivo.
- **Motor de navegación:** Para mostrar el contenido HTML, Android utiliza la librería WebKit.

d) Runtime

Android incorpora un set de librerías que aportan la mayor parte de las funcionalidades disponibles en las librerías base del lenguaje de programación Java. Cada aplicación Android corre su propio proceso, con su propia instancia de la máquina virtual Dalvik. Dalvik ha sido escrito de forma que un dispositivo puede correr múltiples máquinas virtuales de forma eficiente. Dalvik ejecuta archivos en el formato Dalvik Executable (.dex), el cual está optimizado para memoria mínima. La Máquina Virtual está basada en registros, y corre clases compiladas por el compilador de Java que han sido transformadas al formato .dex por la herramienta incluida "dx".

e) El kernel Linux

Android está construido sobre el kernel de Linux. Linux provee la capa de abstracción de hardware para Android permitiendo que este sea transferido / portado a una gran variedad de plataformas en el futuro. Internamente, Android utiliza Linux para su manejo de memoria, control de procesos, trabajo con redes y otros servicios relativos al sistema operativo. El usuario de un teléfono Android jamás verá Linux en su dispositivo y los programas que desarrolles nunca podrán utilizar sus comandos directamente. Como desarrollador tú debes saber que ahí está. (págs. 42-43)

2.2.3.6 Ventajas y Desventajas

Ventajas





- Código libre
- Aplicaciones disponibles gratis
- Multitarea




Desventajas

- Duración de la batería. (Santa Maria, 2014)

2.2.3.7 Versiones

Martínez (2013) detalla las versiones del sistema operativo Android:

Nombre	Versión	Fecha de Lanzamiento	Descripción	Logo
Apple Pie	1.0	23 Septiembre del 2008	Fue lanzado el 23 de septiembre de 2008 y el primer smartphone en el que fue instalado es el HTC Dream, con teclado físico, su sistema operativo era muy básico. Este incluyó la primera versión de la Android Market, un Navegador Web, soporte para mensajes de texto SMS y MMS, discador para llamadas, y una aplicación para tomar fotos que no contaba con los ajustes de blancos y resolución.	
Banana Bread	1.1	9 Febrero del 2009	Llegó solo para el dispositivo HTC Dream, entre sus novedades se encontraban el soporte para marquesina en diseños de sistemas, la posibilidad de guardar los archivos adjuntos en los mensajes, y las reseñas al buscar negocios en los mapas.	
Cupcake	1.5	30 Abril del 2009	Contaba con un rediseño completo en su interfaz, además contaba con transiciones animadas, mejoras en la velocidad de la cámara, teclado en la pantalla y soporte de bluetooth stereo.	
Donut	1.6	15 Septiembre de 2009	Esta versión fue en realidad una pequeña actualización, pero vino empaquetada con un cuadro de búsqueda mejorado, cámara y aplicación de galería, y una renovada Android Market.	

Eclair	2.0–2.1	25 Octubre de 2009	<p>Se rediseño la interfaz del navegador, soporte nativo de flash para la cámara, zoom digital en las fotos.</p> <p>Integración social permitiendo sincronizar los contactos de Facebook, y más tarde, Twitter, que les permitió a sus usuarios tener todos sus contactos de todas las redes sociales en un solo lugar.</p>	
Froyo	2.2	Mayo de 2010	<p>Incorpora el motor de Java V8 y ofrece a los usuarios un aumento de velocidad gracias al compilador JIT que permite iniciar las solicitudes más rápido y mejorar el rendimiento general del sistema.</p> <p>Incluye la posibilidad de hacer tretheing, es decir, compartir la conexión 3G a través del wifi del teléfono con otros dispositivos, con la posibilidad de convertir tu móvil en un hotspot.</p>	
Gingerbread	2.3	6 Diciembre de 2010	<p>Incorporó una gran cantidad de novedades tanto a estético con una renovada interfaz de usuario con incrementos de velocidad y simpleza, y se preparó para la llegada de los smartphones de doble núcleo al cambiar al sistema de archivos EXT4 y de pantallas más grandes con el soporte para resoluciones WXGA y mayores</p>	
HoneyComb	3.0–3.1	22 Febrero de 2011	<p>Renovada interfaz de usuario con una nueva barra de sistema en la parte inferior de la pantalla que permitía el acceso rápido a notificaciones, estados y botones de navegación suavizados y el Action Bar.</p>	

Icecream Sandwich	4.0	19 Octubre 2011	Significó un importante paso en la evolución de Android que no solo vio renovada casi por completo su interfaz de usuario con el nuevo diseño Holo, sino que volvió a integrar el sistema operativo en sus versiones para Tablets y Smartphones.	 ice cream sandwich
Jelly Bean	4.1	Junio 2012	Mejoro la estabilidad, funcionalidad y rendimiento de la interfaz de usuario, para lo cual se implementó el núcleo de linux 3.0.31 y una serie de mejoras en lo que se llamó Project Butter que permitió aumentar hasta 60 FPS las transiciones en la interfaz de usuario, dando una experiencia realmente fluida.	 jelly bean
KitKat	4.4	Diciembre 2013	Es la versión vigente del sistema operativo para móviles, con una interfaz impecable Google presume esta fantástica versión de Android	 kitkat
Lollipop	5.0	Noviembre 2014	Este posee soporte para 64 bits y funciona en teléfonos, tabletas y tvs.	 lollipop

Tabla 3. Versiones del Sistema Operativo Android

Fuente: androidzone.org

Editado: Andrea Loaiza

2.2.3.8 Componentes de una Aplicación Android

Ribas Lequerica (2013) considera que:

Para diseñar una aplicación en Android, es necesario tener claros los elementos que la componen y la funcionalidad de cada uno de ellos. Uno de los aspectos más importantes a tener en cuenta es su funcionamiento. Android trabaja en Linux, y cada aplicación utiliza un proceso propio. Se distinguen por el ID, un identificador para que solo ella tenga acceso a sus archivos. Los componentes son los elementos básicos con los que se construyen el proyecto. Habrá tantas actividades como ventanas distintas tenga la aplicación. (pág. 41)



Figura 4. Componentes de una Aplicación Android

Fuente: Libro Desarrollo de Aplicaciones para Android 2013

a) Activity

Es el bloque más común de los cuatro, el que más se usa en las aplicaciones. Aunque es posible realizar una Activity sin representación gráfica, se puede decir que una Activity corresponde a una ventana o a un cuadro de dialogo en una aplicación de escritorio. Una Activity es una clase donde mostraremos Views (vistas) para generar la interfaz de usuario y seremos capaces de responder a eventos que se realicen sobre ella. Pese a que el conjunto de ellas forman la aplicación, son entidades independientes que son capaces de llamarse entre ellas, pasándose parámetros y recibiendo respuestas de modo que su funcionamiento sea el de un todo. A cada Activity se le asigna una ventana sobre la que se dibujara la interfaz de usuario. (pág. 45)

b) Broadcast Intent Receivers

Un Broadcast Intent Receiver (receptor de emisiones de intentos o más entendible: receptor de mensajes) es un componente que simplemente se encarga de recibir y reaccionar frente a ciertos mensajes emitidos por el sistema. Cada aplicación puede tener tantos Broadcast Intent Receiver como considere necesarios, de igual modo que puede emitir tantos mensajes como sean oportunos, eso sí, todos los receptores deben extender

de la clase BroadcastReceiver. Para que un BroadcastReceiver sea accesible al sistema, este debe registrarse mediante el fichero AndroidManifest.xml o mediante programación a través del método: Context.registerReceiver(). (pág. 46)

c) Service

Las actividades tienen un periodo de vida corto y pueden estar ejecutándose y al poco tiempo ser desechadas. Los Services (servicios) están diseñados para mantenerse ejecutándose (si fuera necesario) sin depender de ninguna Activity. Es un componente que corre de fondo para hacer operaciones de larga duración o trabajo en procesos remotos. Típicos Services son aquellos que periódicamente se conectan a algún servidor para ver si ha cambiado información. (pág. 47)

d) Content providers

Los Content providers (proveedores de contenido) proporcionan una capa de abstracción para acceder a datos almacenados por una aplicación de modo que puedan ser accesibles a otras aplicaciones. Mediante los Content providers, una aplicación puede hacer públicos sus datos a otras aplicaciones de manera sencilla y sin miedo a que se puedan corromper. (pág. 47)

e) Fragment

Aparecen a partir de la versión 3.0 de Android para solucionar el problema de las múltiples pantallas. Su cometido principal es la reutilización tanto de código de lógica de trabajo como de las interfaces de esos códigos. (pág. 48)

f) Intents

Son lanzados constantemente a lo largo del sistema para notificar diversos eventos, desde la inserción de una tarjeta SD o que el dispositivo se está quedando sin batería hasta eventos específicos de alguna aplicación (petición de ejecutar una nueva actividad o servicio). Los intentos son objetos de la clase Intent que contiene datos del mensaje a transmitir. Dentro de los datos transmitidos en el intent hay que diferenciar dos partes que son la acción a realizar y los datos sobre los que realizar la acción. La acción a realizar viene definida por cada aplicación. (pág. 49)

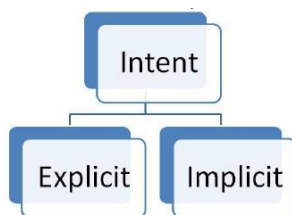


Figura 5. Tipos de Intent

Fuente: Libro Desarrollo de Aplicaciones para Android 2013

- ✓ **Intent Explícito:** Este tipo de Intent es aquel en el que se conoce exactamente qué componente lanzar, el cual especificaremos a través del nombre de componente. Se lo utiliza cuando se desea lanzar un servicio o actividad, típicamente de nuestra propia aplicación.
- ✓ **Intent Implícito:** En este caso, no se conoce el nombre de componente (estará vacío), pero si se sabe la acción que se desea ejecutar. De esta forma, se le pide al sistema operativo que busque por usted qué aplicación podría realizar dicha acción y, en el caso de que haya varias, nos dará a elegir cuál utilizar. (pág. 50)

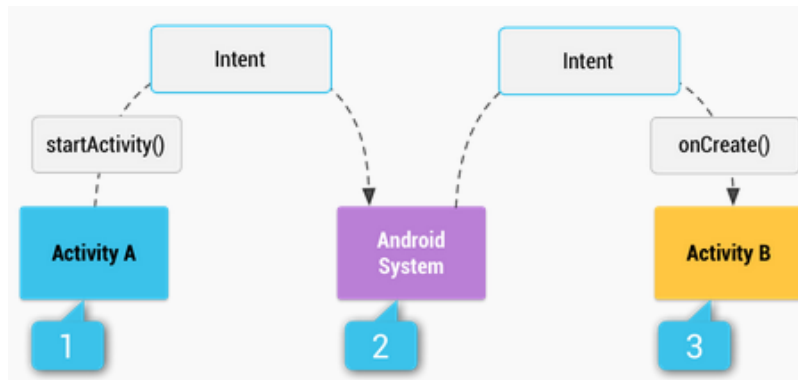


Figura 6. Ejemplo de ejecución de Intent

Fuente: Libro Desarrollo de Aplicaciones para Android 2013

2.2.3.9 Estado de los procesos

Ribas Lequerica (2013) menciona que:

Cada **aplicación** de Android corre en su **propio proceso**, el cual es creado por la aplicación cuando se ejecuta y permanece hasta que la aplicación deja de trabajar o el sistema necesita memoria para otras aplicaciones. Android sitúa cada proceso en una jerarquía de "importancia" basada en estados, como se puede ver a continuación:

- ✓ **Actividad de primer plano (Foreground Activity):** Se trata de la Activity con la que el usuario interactúa, la que se encuentra en la pantalla.
- ✓ **Actividad Visible:** Aquella actividad que es visible al usuario pero no es la Foreground Activity, es decir, tiene algo por encima de ella en pantalla, algo como un cuadro de diálogo que aunque no permite interactuar con ella sí permite verla.
- ✓ **Actividad de Fondo (Background Activity):** Son las actividades que el usuario no ve y que han sido pausadas.
- ✓ **Proceso vacío:** Son los procesos que no tienen asociados ninguna actividad ni dependen de ningún otro componente (Service o BroadcastReceiver). Son los primeros en ser sacrificados por el sistema si se necesitan recursos. (págs. 54 - 55)

2.2.3.10 Ciclo de vida de una actividad

Ribas Lequerica (2013) afirma que:

Los componentes de las aplicaciones tienen unos ciclos de vida que dependerán de la situación en la que se encuentre en cada momento la aplicación, desde que se crea y es capaz de responder a eventos hasta que se destruye y se liberan todos los recursos utilizados, la aplicación pasará por diferentes estados.

A lo largo de una ejecución normal de una aplicación, sus Activity las podemos encontrar en alguno de los siguientes cuatro estados:

- ✓ **Activada:** Es cuando el usuario ve la actividad y puede interactuar con ella desde la pantalla, o dicho de otro modo, cuando está la primera pila de ejecución.
- ✓ **Pausada:** Cuando la actividad ha pasado ya a segundo plano, pero aun esta visible, es cuando otra actividad se coloca sobre la actividad que pasa a pausa, pero la nueva actividad no tapa del todo a la actividad anterior (bien porque sea transparente o porque sea de menor tamaño su interfaz); la actividad pausada pierde el foco pero se ve parte de ella.
- ✓ **Parada:** Cuando la actividad pasa a segundo plano y además está totalmente tapada por la nueva actividad, es decir queda totalmente eclipsada por la nueva interfaz.
- ✓ **Destruída:** La actividad no está ya disponible, se han liberado todos sus recursos y en caso de ser llamada, necesitaría comenzar un nuevo ciclo de vida. (págs. 50 - 51)

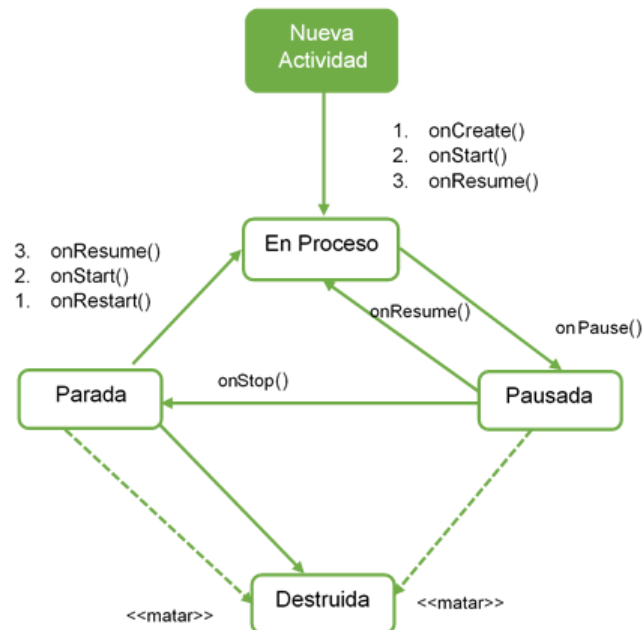


Figura 7. Estados de una Activity

Fuente: Libro Desarrollo de Aplicaciones para Android 2013, pág. 52

Editado: Andrea Loaiza

Según Ribas Lequerica (2013) los métodos para la gestión del ciclo de vida son:

- **onCreate():** Es llamado cuando la actividad arranca por primera vez. Puede ser utilizado para realizar tareas de inicialización como crear la interfaz de usuario.
- **onStart():** Se ejecuta cuando la aplicación va a ser visible al usuario.
- **onResume():** Se ejecuta antes de que el usuario pueda interactuar con la Activity. En este momento la Activity se encuentra en la parte superior de la pila de ejecución.
- **onPause():** Este método se ejecuta cuando la actividad está a punto de pasar a segundo plano, usualmente debido a que otra actividad ha sido lanzada frente a ella.
- **onStop():** Este método es llamado cuando la actividad ya no está visible para el usuario y no será requerida por un tiempo.
- **onRestart():** Si este método es llamado significa que una actividad que estaba detenida está volviendo a ser desplegada al usuario.
- **onDestroy():** Es llamado justo antes de que la actividad sea destruida. Durante la destrucción de la Activity se perderán todos los datos asociados a ella, de modo que si vuelve a ser llamada se ejecutara un nuevo ciclo de vida, por lo que es uno de los lugares para controlar la persistencia de datos.
- **onSaveInstanceState():** Android mandará llamar este método para permitir a la actividad el guardar un estatus por instancia (como la posición del cursor en un campo de texto). Usualmente no será necesario reescribir este método ya que la implementación por defecto guarda el estado de todos los controles de la interfaz de manera automática.
- **onRestoreInstanceState():** Este método es llamado cuando la actividad está siendo reiniciada desde un estatus previamente almacenado en onSaveInstanceState(). La implementación por defecto restaura el estado de la interfaz del usuario completamente. (pág. 53)

2.2.3.11 Acceso a Servicios web SOAP en Android

Gómez Oliver (2012) explica que:

Android no incluye ningún tipo de soporte para el acceso a servicios web de tipo SOAP. Es por esto por lo que se debe agregar una librería externa para hacer más fácil esta tarea. Entre la oferta actual, la opción más popular y más utilizada es la librería ksoap2-android. Esta librería es un fork, especialmente adaptado para Android, de la antigua librería kSOAP2. Este framework permitirá de forma relativamente fácil y cómoda utilizar servicios web que utilicen el estándar SOAP.

En la implementación del evento onClick del botón, que será el encargado de comunicarse con el servicio web y procesar el resultado, se definirán cuatro constantes que servirán en varias ocasiones durante el código:

- **NAMESPACE.** Espacio de nombres utilizado en nuestro servicio web.
- **URL.** Dirección URL para realizar la conexión con el servicio web.
- **METHOD_NAME.** Nombre del método web concreto que vamos a ejecutar.
- **SOAP_ACTION.** Equivalente al anterior, pero en la notación definida por SOAP.

Asignación de valores a las constantes:

```
1 String NAMESPACE = "http://sgoliver.net/";
2 String URL="http://10.0.2.2:1473/ServicioClientes.asmx";
3 String METHOD_NAME = "NuevoClienteSimple";
4 String SOAP_ACTION = "http://sgoliver.net/NuevoClienteSimple";
```

Figura 8. Ejemplo de asignación de valores a las constantes

Fuente: <http://www.sgoliver.net/blog/?p=2594>

En la URL se ha sustituido el nombre de máquina *localhost* por la dirección IP equivalente. Además se debe verificar que el puerto coincide con el que se está utilizando en la máquina. En este caso el servicio se ejecuta sobre el puerto **1473**, pero es posible que en otras ocasiones el número sea distinto. Los siguientes pasos del proceso serán crear la petición SOAP al servicio web, enviarla al servidor y recibir la respuesta.

En la siguiente figura se observa que justo debajo de la sección donde se solicitan los parámetros a pasar al método se incluye también un XML de muestra de cómo tendría que ser la petición al servidor. Se ha marcado las tres partes principales de una petición de tipo SOAP. Empezando por la “parte interna” del XML, en primer lugar se encuentran los datos de la petición en sí (Request) que contiene el nombre del método al que se desea llamar, y los nombres y valores de los parámetros en entrada. Rodeando a esta información se añaden otra serie de etiquetas y datos a modo de contenedor estándar que suele recibir el nombre de Envelope. La información indicada en este contenedor no es específica de la llamada al servicio, pero sí contiene información sobre formatos y esquemas de validación del estándar SOAP. Por último, durante el envío de esta petición SOAP al servidor mediante el protocolo HTTP se añaden determinados encabezados como los que se observan en la imagen. Todo esto junto hará que el servidor sea capaz de interpretar correctamente la petición SOAP, se llame al método web correcto, y devuelva el resultado en un formato.

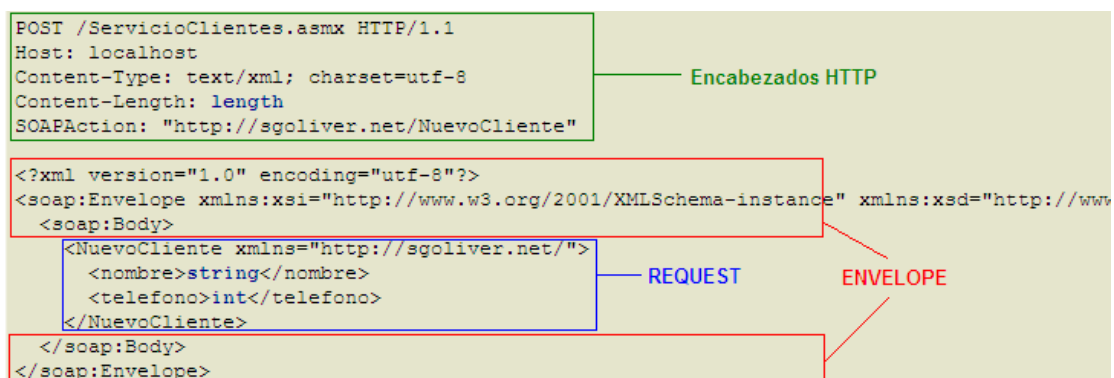


Figura 9. Ejemplo de request y response

Fuente: <http://www.sgoliver.net/blog/?p=2594>

Para crear la petición (request) al método, se debe crear un nuevo objeto `SoapObject` pasándole el namespace y el nombre del método web. A esta petición se debe asociar los parámetros de entrada mediante el método `addProperty()` al que se pasarán los nombres y valores de los parámetros.

```

1 SoapObject request = new SoapObject(NAMESPACE, METHOD_NAME);
2
3 request.addProperty("nombre", txtNombre.getText().toString());
4 request.addProperty("telefono", txtTelefono.getText().toString());

```

Figura 10. Ejemplo petición (request)

Fuente: <http://www.sgoliver.net/blog/?p=2594>

Se debe crear el contenedor SOAP (envelope) y asociarle la petición (request). Para ello se debe crear un nuevo objeto `SoapSerializationEnvelope` indicando la versión de SOAP que se va a usar (versión 1.1 en el caso de ejemplo). Además se debe indicar que se trata de un servicio web. Por último, se debe asociar la petición antes creada al contenedor llamando al método `setOutputSoapObject()`.

```

1 SoapSerializationEnvelope envelope =
2     new SoapSerializationEnvelope(SoapEnvelope.VER11);
3
4 envelope.dotNet = true;
5
6 envelope.setOutputSoapObject(request);

```

Figura 11. Ejemplo de creación del contenedor SOAP (envelope)

Fuente: <http://www.sgoliver.net/blog/?p=2594>

Ahora se crea el objeto que se encargará de realizar la comunicación HTTP con el servidor, de tipo `HttpTransportSE`, al que se pasará la URL de conexión al servicio web. Por último, se debe completar el proceso realizando la llamada al servicio web mediante el método `call()`.

```

1 HttpTransportSE transporte = new HttpTransportSE(URL);
2
3 try
4 {
5     transporte.call(SOAP_ACTION, envelope);
6
7     //Se procesa el resultado devuelto
8     //...
9 }
10 catch (Exception e)
11 {
12     txtResultado.setText("Error!");
13 }

```

Figura 12. Ejemplo de objeto para comunicación HTTP

Fuente: <http://www.sgoliver.net/blog/?p=2594>

Tras la llamada al servicio ya se puede obtener el resultado devuelto por el método `web` llamado. Esto se lo consigue mediante el método `getResponse()`. Dependiendo del tipo de resultado que se espere recibir se deberá convertir esta respuesta a un tipo u otro. En este caso, como el resultado que se espera es un valor simple (un número entero) se convertirá la respuesta a un objeto `SoapPrimitive`, que directamente se puede convertir a una cadena de caracteres llamado a `toString()`.


```

1 SoapPrimitive resultado_xml =(SoapPrimitive)envelope.getResponse();
2 String res = resultado_xml.toString();
3
4 if(res.equals("1"))
5     txtResultado.setText("Insertado OK");

```

Figura 13. Ejemplo de método getResponse

Fuente: <http://www.sgoliver.net/blog/?p=2594>

SOAP es un protocolo que permite especificar los objetos y métodos de un servicio web a los que el cliente tendrá acceso.

2.2.4 PROGRAMACIÓN ORIENTADA A OBJETOS

2.2.4.1 Introducción

Según Llobet Azpitarte, Alonso Jordá, Miedes De Elías, Ruiz Fuertes y Torres Goterris (2008):

La orientación a objetos promete mejoras de amplio alcance en la forma de diseño, desarrollo y mantenimiento del software ofreciendo una solución a largo plazo a los problemas y preocupaciones que han existido desde el comienzo en el desarrollo de software:

La falta de portabilidad del código y su escasa reusabilidad.

- Código que es difícil de modificar.
- Ciclos de desarrollo largos.
- Técnicas de codificación no intuitivas.

El concepto de programación orientada a objetos (POO) no es nuevo, lenguajes clásicos como SmallTalk se basan en ella. Dado que la POO se basa en la idea natural de la existencia de un mundo lleno de objetos y que la resolución del problema se realiza en términos de objetos, un lenguaje se dice que está basado en objetos si soporta objetos como una característica fundamental del mismo. No debemos confundir que esté basado en objetos con que sea orientado a objetos: para que sea orientado a objetos al margen que esté basado en objetos, necesita tener clases y relaciones de herencia entre ellas. (pág. 12)

La programación orientada a objetos intenta simular el mundo real mediante el uso de objetos, clases y herencia entre clases para el diseño de aplicaciones.

2.2.4.2 Propiedades de la programación orientada a objetos

a) Objetos

Llobet Azpitarte et al. (2008) definen que:

Un objeto es un conjunto complejo de datos y programas que poseen estructura y forman parte de una organización. En este caso las estructuras de datos y los

algoritmos usados para manipularlas están encapsulados en una idea común llamada objeto.

Esta definición especifica dos propiedades características de los objetos:

- En primer lugar, un objeto no es un dato simple, sino que contiene en su interior cierto número de componentes bien estructurados.
- En segundo lugar, cada objeto no es un ente aislado, sino que forma parte de una organización jerárquica o de otro tipo. (pág. 13)

b) Las clases

“Una clase es la descripción de una familia de objetos que tienen la misma estructura (atributos) y el mismo comportamiento (métodos).” (Llobet Azpitarte, Alonso Jordá, Miedes De Elías, Ruiz Fuertes, & Torres Goterris, 2008, pág. 15)

Terrero y Paredes (2011) consideran que:

Las clases son las plantillas para hacer objetos. Una clase sirve para definir una serie de objetos con propiedades (atributos), comportamientos (operaciones o métodos), y semántica comunes. Hay que pensar en una clase como un molde. A través de las clases se obtienen los objetos en sí. Es decir antes de poder utilizar un objeto se debe definir la clase a la que pertenece, esa definición incluye:

- **Sus atributos:** Es decir, los datos miembros de esa clase. Los datos pueden ser públicos (accesibles desde otra clase) o privados (sólo accesibles por código de su propia clase. También se las llama campos.
- **Sus métodos:** La función miembro de la clase. Son las acciones (u operaciones) que puede realizar la clase. (pág. 37)

c) Atributos

“Un atributo es una característica de un objeto. Mediante los atributos se define información oculta dentro de un objeto, la cual es manipulada solamente por los métodos definidos sobre dicho objeto.” (Carballo, 2010)

d) Métodos

“Los métodos se utilizan de la misma forma que los atributos, excepto porque los métodos poseen siempre paréntesis, dentro de los cuales pueden ir valores necesarios para la ejecución del método (parámetros).” (Terrero & Paredes, 2011, pág. 39)

e) Herencia

Llobet Azpitarte et al. (2008) explican que:

Herencia es la propiedad por la que los ejemplares de una clase hija (subclase) pueden tener acceso tanto a los datos como al comportamiento (métodos) asociados con una clase paterna (superclase). La herencia significa que el comportamiento y los datos asociados con las clases hijas son siempre una extensión de las propiedades asociadas con las clases paternas. Una subclase debe reunir todas las propiedades de la clase paterna y otras más. (pág. 19)

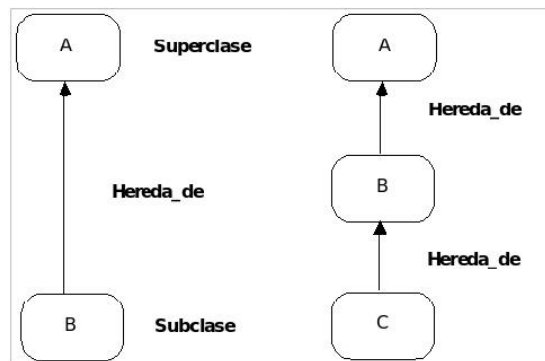


Figura 14. Esquema de Herencia

Fuente: Libro Introducción a la POO (Programación Orientada a Objetos)

f) Encapsulamiento

“Una clase se compone tanto de variables (propiedades) como de funciones y procedimientos (métodos). Hay una zona oculta al definir la clase (zona privada) que sólo es utilizada por esa clase y por alguna clase relacionada. Hay una zona pública (llamada también interfaz de la clase) que puede ser utilizada por cualquier parte del código.” (Terrero & Paredes, 2011, pág. 35)

g) Polimorfismo

“Cada método de una clase puede tener varias definiciones distintas. En el caso del parchís: partida.empezar (4) empieza una partida para cuatro jugadores, partida.empezar (rojo, azul) empieza una partida de dos jugadores para los colores rojo y azul; estas son dos formas distintas de emplear el método empezar, que es polimórfico.” (Terrero & Paredes, 2011, pág. 36)

2.2.5 ENTORNO DE DESARROLLO

2.2.5.1 IDE Android Studio

Google Inc. (2009) explica que:

Android Studio es un nuevo entorno de desarrollo de Android basado en IntelliJ IDEA. Ofrece nuevas características y mejoras con respecto a Eclipse ADT. En la parte superior de las capacidades que usted espera de IntelliJ, Android Studio ofrece:

- ✓ Sistema de construcción a base de Gradle flexible.
- ✓ Construir variantes y generación de múltiple APK.
- ✓ Extensas plantillas soportadas por los servicios de Google y varios tipos de dispositivos.
- ✓ Editor de diseño Rich con soporte para la edición de tema.
- ✓ Herramientas para capturar rendimiento, usabilidad, compatibilidad de versiones, y otros problemas
- ✓ ProGuard y aplicación de firma de capacidades.

El IDE Android Studio es el entorno de desarrollo Android creado por Google, posee características que facilitan el desarrollo, la refactorización es más potente, incorpora una herramienta para el análisis del código, la edición es más fluida, además de que permite la previsualización de los recursos.

2.2.5.2 Android SDK

“Es un kit de desarrollo de software para Android consta de un conjunto de herramientas para el desarrollo que permite a los desarrolladores y programadores confeccionar aplicaciones y juegos para el sistema dicho. Se trata pues de una interfaz de programación de aplicaciones que hace uso del lenguaje de programación Java.” (Ramírez Hernández, 2011).

Es un conjunto de herramientas de programación que permiten el desarrollo de aplicaciones de una manera sencilla, a través del lenguaje de programación java.

2.2.5.3 Xml

Sagástegui Lescano (2008) afirma que:

XML son las siglas en inglés de eXtensible Markup Language (lenguaje de marcas ampliable), este lenguaje es del tipo metalenguaje y es extensible mediante el uso de etiquetas, fue desarrollado por el World Wide Web Consortium. Nació como una forma de reducir SGML usando para ello la parte que permite definir la gramática de lenguajes específicos. XML (Extensible Markup Language) es un lenguaje de etiquetas, es decir, cada paquete de información está delimitado por dos etiquetas como se hace también en el lenguaje HTML, pero XML separa el contenido de la presentación. Sus características más importantes son que permite separar el código de la presentación del mismo y que es completamente extensible mediante el uso de nuevas etiquetas creadas por el desarrollador.

XML es una tecnología que sirve para estructurar, almacenar e intercambiar información.

2.2.5.4 Lenguaje de programación Java

Según Terrero y Paredes (2011):

La compañía Sun describe el lenguaje Java como: “simple, orientado a objetos, distribuido, interpretado, robusto, seguro, de arquitectura neutra, portable, de altas prestaciones, multitarea y dinámico. (pág. 3)

“Java es un lenguaje de programación de alto nivel con el que se pueden escribir tanto programas convencionales como para internet.” (Ceballos, 2011, pág. 22)

Java es un lenguaje de programación multiplataforma, orientado a objetos y basado en clases.

2.2.5.5 PostgreSQL

Gibert Ginestà y Pérez Mora (2012) consideran que:

En 1996, se hizo evidente que el nombre “Postgres95” no resistiría el paso del tiempo. Elegimos un nuevo nombre, PostgreSQL, para reflejar la relación entre el Postgres

original y las versiones más recientes con capacidades SQL. Al mismo tiempo, hicimos que los números de versión partieran de la 6.0, volviendo a la secuencia seguida originalmente por el proyecto Postgres. Durante el desarrollo de Postgres95 se hizo hincapié en identificar y entender los problemas en el código del motor de datos. Con PostgreSQL, el énfasis ha pasado a aumentar características y capacidades, aunque el trabajo continúa en todas las áreas.

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales. PostgreSQL utiliza un modelo cliente/servidor y usa *multiprocesos* en vez de *multihilos* para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

PostgreSQL es sistema de gestión de bases de datos relacional orientada a objetos, de código abierto.

a) Las principales mejoras en PostgreSQL incluyen:

- ✓ Los bloqueos de tabla han sido sustituidos por el control de concurrencia multi-versión, el cual permite a los accesos de sólo lectura continuar leyendo datos consistentes durante la actualización de registros, y permite copias de seguridad en caliente desde pg_dump mientras la base de datos permanece disponible para consultas.
- ✓ Se han implementado importantes características del motor de datos, incluyendo subconsultas, valores por defecto, restricciones a valores en los campos (constraints) y disparadores (triggers).
- ✓ Se han añadido funcionalidades en línea con el estándar SQL92, incluyendo claves primarias, identificadores entrecomillados, forzado de tipos cadena literal, conversión de tipos y entrada de enteros binarios y hexadecimales.
- ✓ Los tipos internos han sido mejorados, incluyendo nuevos tipos de fecha/hora de rango amplio y soporte para tipos geométricos adicionales.
- ✓ La velocidad del código del motor de datos ha sido incrementada aproximadamente en un 20-40%, y su tiempo de arranque ha bajado el 80% desde que la versión 6.0 fue lanzada.

b) Componentes de un sistema PostgreSQL

Martínez (2010) explica los componentes:

- ✓ **Aplicación cliente:** Esta es la aplicación cliente que utiliza PostgreSQL como administrador de bases de datos.
- ✓ **Demonio postmaster:** Este es el proceso principal de PostgreSQL. Es el encargado de escuchar por un puerto/socket por conexiones entrantes de clientes.
- ✓ **Ficheros de configuración:** Los 3 ficheros principales de configuración utilizados por PostgreSQL, postgresql.conf, pg_hba.conf y pg_ident.conf

- ✓ **Procesos hijos postgres:** Procesos hijos que se encargan de autenticar a los clientes, de gestionar las consultas y mandar los resultados a las aplicaciones clientes.
- ✓ **PostgreSQL share buffer cache:** Memoria compartida usada por PostgreSQL para almacenar datos en caché.
- ✓ **Write-Ahead Log (WAL):** Componente del sistema encargado de asegurar la integridad de los datos (recuperación de tipo REDO).
- ✓ **Kernel disk buffer cache:** Caché de disco del sistema operativo
- ✓ **Disco:** Disco físico donde se almacenan los datos y toda la información necesaria para que PostgreSQL funcione.

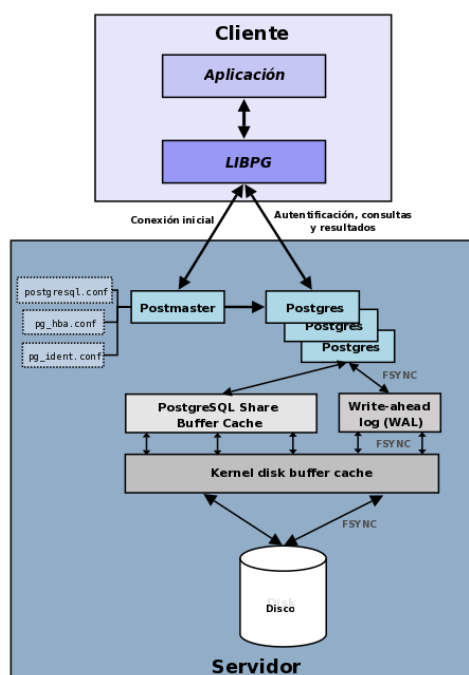


Figura 15. Componentes de un Sistema PostgreSQL

Fuente: http://www.postgresql.org.es/sobre_postgresql

c) Arquitectura de PostgreSQL

El siguiente gráfico muestra de forma esquemática las entidades involucradas en el funcionamiento normal del gestor de bases de datos:

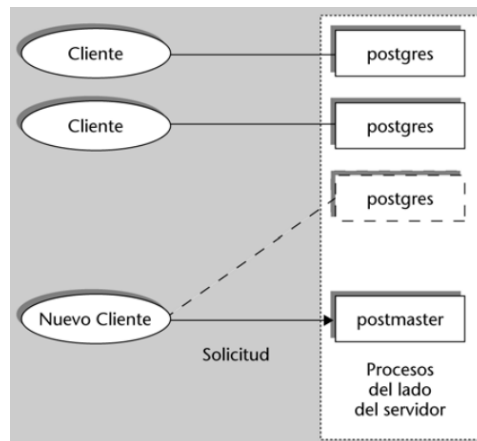


Figura 16. Arquitectura de PostgreSQL

Fuente: <http://ocw.uoc.edu/>

Según Gibert Ginestà y Pérez Mora (2012):

PostgreSQL está basado en una arquitectura cliente-servidor. El programa servidor se llama **Postgres** y entre los muchos programas cliente tenemos, por ejemplo, **pgaccess** (un cliente gráfico) y **psql** (un cliente en modo texto).

Un proceso servidor *postgres* puede atender exclusivamente a un solo cliente; es decir, hacen falta tantos procesos servidor *postgres* como clientes haya. El proceso **postmaster** es el encargado de ejecutar un nuevo servidor para cada cliente que solicite una conexión.

Se llama **sitio** al equipo anfitrión (*host*) que almacena un conjunto de bases de datos PostgreSQL. En un *sitio* se ejecuta solamente un proceso *postmaster* y múltiples procesos *postgres*. Los clientes pueden ejecutarse en el mismo sitio o en equipos remotos conectados por TCP/IP. Es posible restringir el acceso a usuarios o direcciones IP modificando las opciones del archivo `pg_hba.conf`, que se encuentra en `/etc/postgresql/pg_hba.conf`. Este archivo, junto con `/etc/postgresql/postgresql.conf` es particularmente importante, porque algunos de sus parámetros de configuración por defecto provocan multitud de problemas al conectar inicialmente y porque en ellos se especifican los mecanismos de autenticación que usará PostgreSQL para verificar las credenciales de los usuarios.

Para habilitar la conexión a PostgreSQL desde clientes remotos, debemos verificar el parámetro `tcpip_socket = true` en el fichero `/etc/postgresql/postgresql.conf`.

A continuación, para examinar los métodos de autenticación y las posibilidades de conexión de clientes externos, debemos mirar el fichero `/etc/postgresql/pg_hba.conf`, donde se explicita la acción que hay que emprender para cada conexión proveniente de cada *host* externo, o grupo de *hosts*. (pág. 65)

2.2.5.6 Python

González Duque (2010) afirma que:

Es un lenguaje interpretado, interactivo y orientado a objetos que ofrece una gran cantidad de estructuras de datos de alto nivel por medio de un tipado dinámico y fuerte, además de estas características es multiparadigma y multiplataforma. Python es un lenguaje de programación creado por Guido van Rossum a principios de los años 90 cuyo nombre está inspirado en el grupo de cómicos ingleses “Monty Python”. Es un lenguaje similar a Perl, pero con una sintaxis muy limpia y que favorece un código legible. Se trata de un lenguaje interpretado o de script, con tipado dinámico, fuertemente tipado, multiplataforma y orientado a objetos.

Python es un lenguaje de programación orientado a objetos, multiplataforma, dinámico que posee grandes estructuras de datos.

Ventajas de Python

- ✓ Python es un lenguaje muy expresivo, es decir, los programas Python son muy compactos: un programa Python suele ser bastante más corto que su equivalente en lenguajes como C. (Python llega a ser considerado por muchos un lenguaje de programación de muy alto nivel.)
- ✓ Python es muy legible. La sintaxis de Python es muy elegante y permite la escritura de programas cuya lectura resulta más fácil que si utilizáramos otros lenguajes de programación.
- ✓ Python ofrece un entorno interactivo que facilita la realización de pruebas y ayuda a despejar dudas acerca de ciertas características del lenguaje.
- ✓ El entorno de ejecución de Python detecta muchos de los errores de programación que escapan al control de los compiladores y proporciona información muy rica para detectarlos y corregirlos.
- ✓ Python puede usarse como lenguaje imperativo procedimental o como lenguaje orientado a objetos.
- ✓ Posee un rico juego de estructuras de datos que se pueden manipular de modo Sencillo. (págs. 7-8)

2.2.5.7 Servicios Web

“Son un conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la Web. Estas aplicaciones o tecnologías intercambian datos entre sí con el objetivo de ofrecer unos servicios.” (W3C, 2013)

Los servicios web es una tecnología que utiliza una serie de protocolos que permiten el intercambio de información entre distintas aplicaciones.

a) ¿Para qué sirven los Servicios Web?

“Estos servicios proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario. Para proporcionar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas.” (W3C, 2013)

2.2.5.8 Soap (Simple Object Access Protocol)

a) Introducción

Moscatelli (2008) considera que:

Simple Object Access Protocol (SOAP) surge a partir de la especificación realizada por Dave Winer, de la empresa UserLand Software a finales del año 1998, de un mecanismo basado en XML para realizar invocaciones RPC. A partir de dicha especificación y de la unión de las empresas DevelopMentor, UserLand Software y Microsoft surge públicamente la versión 0.9 de SOAP en setiembre de 1999. La versión 1.0 fue lanzada casi a continuación de la versión 0.9 (diciembre de 1999) y enviada a la IETF (Internet Engineering Task Force). Este organismo hizo de esta versión una recomendación oficial. Posteriormente se realizó la versión 1.1 de SOAP, donde, además de las empresas mencionadas, también participaron Lotus e IBM, entre muchas otras empresas de desarrollo de software. Esta nueva versión fue enviada al W3C (World Wide Web Consortium) en mayo del 2000, con el propósito de formar un grupo de trabajo en el área de los protocolos basados en XML. En julio de 2000 el grupo de trabajo en el área de XML del W3C edita el primer W3C Working Draft de la versión 1.2 de SOAP. La especificación del protocolo SOAP, básicamente, describe un formato de mensajes para comunicar aplicaciones. La filosofía de SOAP, para lograr dicha comunicación, es no inventar una nueva tecnología, sino combinar tecnologías existentes y de amplia aceptación en la industria de software. En particular, combina XML para la codificación de los mensajes y HTTP como protocolo de transporte (aunque no se excluye el uso de otros protocolos de transporte). (pág. 3)

b) ¿Qué es SOAP?

Moscatelli (2008) explica que:

SOAP define un mecanismo simple y liviano (en contraposición a sofisticado) para la comunicación, en un entorno distribuido o descentralizado, entre componentes de software o aplicaciones. La comunicación se realiza mediante mensajes codificados en XML y transportados por un protocolo de transporte (SOAP no mandata el uso de un protocolo de transporte en particular, aunque si define como es el transporte en caso de usar HTTP). En definitiva SOAP define un mecanismo para el intercambio de información, estructurada y tipeada, entre pares de aplicaciones en un entorno distribuido, teniendo como objetivos de diseño la simplicidad y la extensibilidad. SOAP no define por sí mismo la semántica de las aplicaciones, como ser un modelo de programación o algún tipo de semántica específica de una implementación, sino que provee un

mecanismo simple para expresar la semántica de las aplicaciones, mediante un modelo modular de empaquetado de mensajes y la definición de como codificar los datos de las aplicaciones en dichos módulos. (pág. 4)

c) Arquitectura básica de SOAP

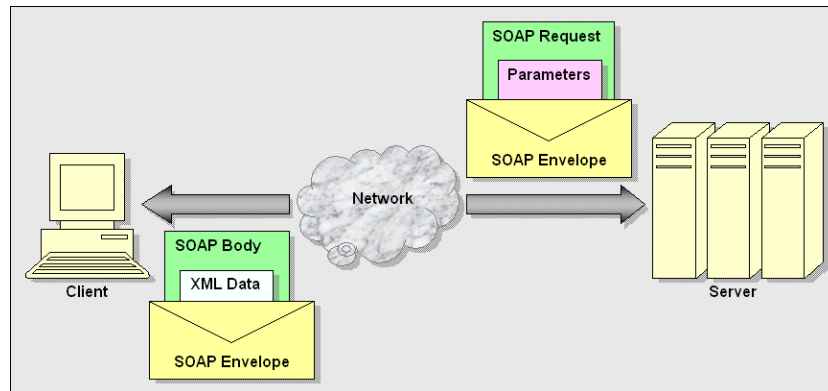


Figura 17. Arquitectura básica de SOAP

Fuente: <http://www.fing.edu.uy/>

Moscatelli (2008) explica que:

En la figura se observa la arquitectura básica de un sistema, análogo al descrito, construido sobre SOAP y los mensajes que definen la interacción entre la aplicación cliente y la aplicación servidor. Generalmente la aplicación cliente envía un mensaje (REQUEST vía HTTP), el cual al ser recibido por la aplicación servidor genera una respuesta (RESPONSE) que es enviada a la aplicación cliente vía HTTP. Se observa además que en el caso de usar SOAP para realizar RPC, la invocación RPC se mapea naturalmente al REQUEST de HTTP y la respuesta RPC se mapea al RESPONSE de HTTP. (pág. 5)

d) Protocolo SOAP

Según Moscatelli (2008) la especificación del protocolo SOAP indica que el mismo consiste de 3 partes:

- ✓ El constructor SOAP ENVELOPE que define un framework para expresar qué hay en un mensaje, a quién está dirigido el mensaje y cuando es opcional o mandatorio.
- ✓ Las reglas de codificación que definen un mecanismo de serialización para ser usado para intercambiar instancias de tipos de datos.
- ✓ La representación SOAP RPC que define una metodología que puede ser usada para representar invocaciones a procedimientos remotos y sus respuestas (pág. 6).

2.2.5.9 Wsdl

“WSDL es sinónimo de Web Services Description Language. WSDL es un lenguaje para describir servicios Web y cómo acceder a ellos, está escrito en XML.” (Refsnes Data Inc, 2012)

Es un formato xml que sirve para describir servicios web, que describe la forma de comunicación, protocolos y mensajes necesarios para la interacción.

a) Documento WSDL

Un documento WSDL es un documento XML simple. Contiene un conjunto de definiciones para describir un servicio web utilizando estos elementos principales:

Elemento	Descripción
<types>	Un contenedor para las definiciones de tipos de datos que utiliza el servicio web
<message>	Una definición con tipo de los datos que se comunica
<portType>	Un conjunto de operaciones con el apoyo de uno o más puntos finales
<binding>	Una especificación de protocolo y datos de formato para un tipo de puerto en particular

Tabla 4. Elementos para describir un Servicio Web

Fuente: <http://www.w3schools.com/>

La estructura principal de un documento WSDL es algo similar a esto:

```
<definitions>
  <types>
    data type definitions.....
  </types>
  <message>
    definition of the data being communicated...
  </message>
  <portType>
    set of operations.....
  </portType>
  <binding>
    protocol and data format specification...
  </binding>
</definitions>
```

Figura 18. Ejemplo de la estructura de un documento WSDL

Fuente: <http://www.w3schools.com/>

“Un documento WSDL puede contener también otros elementos, como elementos de extensión, y un elemento de servicio que permite agrupar las definiciones de varios servicios web en un documento único WSDL.” (Refsnes Data Inc, 2012)

2.2.5.10 ¿Qué es JSON?

Refsnes Data Inc (2012) afirma que:

JSON es sinónimo de JavaScript Object Notation, es un formato de intercambio de datos ligero independiente del lenguaje, es "auto-descripción" y fácil de entender.

Esta sintaxis JSON define un objeto empleados, con una serie de 3 registros de empleados (objetos):

```
JSON Example
{
  "employees": [
    { "firstName": "John", "lastName": "Doe" },
    { "firstName": "Anna", "lastName": "Smith" },
    { "firstName": "Peter", "lastName": "Jones" }
  ]
}
```

Figura 19. Ejemplo definición de objeto Json

Fuente: <http://www.w3schools.com/json>

a) Al igual que XML

- ✓ JSON es texto plano
- ✓ JSON es "auto-describe" (legible)
- ✓ JSON es jerárquica (valores dentro de los valores)
- ✓ JSON pueden recuperar con una HttpRequest

b) Diferencia de XML

- ✓ JSON no utiliza la etiqueta final
- ✓ JSON es más corto
- ✓ JSON es más rápido para leer y escribir
- ✓ JSON puede utilizar matrices

c) Reglas de sintaxis en JSON

- ✓ Los datos son en pares nombre / valor
- ✓ Los datos se separan por comas

- ✓ Las llaves tienen objetos
- ✓ Los corchetes tienen matrices

2.2.6 METODOLOGÍA MOBILE-D

VTT Technology for business (2007) afirma que:

Es una metodología para el desarrollo ágil de software, que no solamente está orientada al desarrollo de aplicaciones móviles, también se puede usar en aplicaciones de seguridad, financieras, de logísticas, y de simulación.

Mobile-D se basa en la programación extrema (XP) para la implementación, Crystal methodologies para la escalabilidad y en el Proceso Unificado de Desarrollo (RUP) para la cobertura del ciclo de vida.

- La fase de exploración, siendo ligeramente diferente del resto del proceso de producción, se dedica al establecimiento de un plan de proyecto y los conceptos básicos, por lo tanto, se puede separar del ciclo principal de desarrollo (aunque no debería obviarse). Los autores de la metodología ponen además especial atención a la participación de los clientes en esta fase.
- Durante la fase de inicialización, los desarrolladores preparan e identifican todos los recursos necesarios. Se preparan los planes para las siguientes fases y se establece el entorno técnico (incluyendo el entrenamiento del equipo de desarrollo). Los autores de Mobile-D afirman que su contribución al desarrollo ágil se centra fundamentalmente en esta fase, en la investigación de la línea arquitectónica. Esta acción se lleva a cabo durante el día de planificación.
- En la fase de "producción" se repite la programación de tres días (planificación, trabajo, liberación) se repite iterativamente hasta implementar todas las funcionalidades. Primero se planifica la iteración de trabajo en términos de requisitos y tareas a realizar. Se preparan las pruebas de la iteración de antemano (de ahí el nombre de esta técnica de TestDriven Development, TDD). Las tareas se llevarán a cabo durante el día de trabajo, desarrollando e integrando el código con los repositorios existentes. Durante el último día se lleva a cabo la integración del sistema (en caso de que estuvieran trabajando varios equipos de forma independiente) seguida de las pruebas de aceptación.
- En la fase de estabilización, se llevan a cabo las últimas acciones de integración para asegurar que el sistema completo funciona correctamente. Esta será la fase más importante en el proyecto multi-equipo con diferentes subsistemas desarrollados por equipos distintos. En esta fase, los desarrolladores realizarán tareas similares a las que debían desarrollar en la fase de "productización", aunque en este caso todo el esfuerzo se dirige a la integración del sistema. Adicionalmente se puede considerar en esta fase la producción de documentación.
- La última fase (prueba y reparación del sistema) tiene como meta la disponibilidad de una versión estable y plenamente funcional del sistema. El producto terminado e integrado se prueba con los requisitos de cliente y se eliminan todos los defectos encontrados.

BIBLIOGRAFÍA

- Álvarez, J. (10 de Abril de 2008). <http://es.slideshare.net/>. Obtenido de <http://es.slideshare.net/Jmaquino/dispositivos-moviles>
- Cancela, L., & Ostos, S. (2012). <https://sites.google.com/>. Obtenido de <https://sites.google.com/site/swcuc3m/home/android/>
- Carballo, P. Y. (21 de Febrero de 2010). *Universidad de los Andes*. Recuperado el 4 de Noviembre de 2013, de Web del profesor: http://webdelprofesor.ula.ve/ingenieria/hyelitza/materias/programacion2/oxo/ProfaYusneyi_Tema8_POOClasesyObjetos.pdf
- Ceballos, J. (2011). *Java 2*. Ra-Ma.
- Gibert Ginestà, M., & Pérez Mora, O. (2012). <http://ocw.uoc.edu/>. Obtenido de http://ocw.uoc.edu/computer-science-technology-and-multimedia/bases-de-datos/bases-de-datos/P06_M2109_02152.pdf
- Gómez Oliver, S. (27 de Febrero de 2012). *sgoliver.net blog*. Obtenido de <http://www.sgoliver.net/blog/?p=2594>
- González Duque, R. (2010). *Python para todos*. España: Autoedición.
- Google Inc. (2009). developer.android.com/. Obtenido de <http://developer.android.com/sdk/installing/studio.html>
- Guevara Soriano, A. (06 de Agosto de 2010). *Revista Seguridad*. Obtenido de <http://revista.seguridad.unam.mx/numero-07/dispositivos-m%C3%B3viles>
- Kovach, P. S. (19 de Agosto de 2013). *SoyEntrepreneur*. Recuperado el 6 de Noviembre de 2013, de SoyEntrepreneur: <http://www.soyentrepreneur.com/25686-17-momentos-clave-de-android.html>
- Llobet Azpitarte, R., Alonso Jordá, P., Miedes De Elías, E., Ruiz Fuertes, M., & Torres Goterris, F. (2008). *Introducción a la programación orientada a objetos con java*. Valencia: Universidad Politécnica de Valencia.
- Marco, C. (2010). Sistema Operativo Android Todo lo que querías saber y tenías miedo a preguntar. *Tecnología con estilo Gadgets*, 62.
- Martínez, E. (Mayo de 2001). <http://www.adecom.biz/>. Obtenido de http://www.adecom.biz/pdf/pdf_agosto2005/La%20evolucion%20de%20la%20telefon%C3%ADa%20movil.pdf
- Martínez, J. (21 de Mayo de 2013). *AndroidZone.org*. Obtenido de <http://androidzone.org/2013/05/historia-de-android-la-evolucion-a-lo-largo-de-sus-versiones/>
- Martínez, R. (2 de 10 de 2010). *Postgresql.org.es*. Obtenido de http://www.postgresql.org.es/sobre_postgresql

- Montoya, J. (2012). *Sistemas Operativos para M6viles*. Obtenido de https://docs.google.com/presentation/d/177BvKpFn3-B07mICmpHmsPsg9m35Sr-_K6qBOFwVju4/edit#slide=id.p
- Moscatelli, S. (21 de Agosto de 2008). *Fing.edu.uy*. Obtenido de www.fing.edu.uy/inco/grupos/lins/docsgen/soap/soap.doc
- Nicol6s, D. (30 de Junio de 2012). Jelly Bean: El nuevo "dulce" de Android. *Diario La tercera*, p6g. 22.
- Paredes Velasco, M., Santacruz Valencia, L., & Dom6nguez Mateos, F. (2012). *Programaci6n Multimedia y Dispositivos M6viles*. Cfgs. Ra-ma.
- Procoop. (23 de Marzo de 2010). *InfoProcoop*. Recuperado el 30 de Octubre de 2013, de http://www.infoprocoop.com.ar/index.php?option=com_content&view=article&id=86:introduccion-a-la-telefonía-fija&catid=37:telefonía&Itemid=62
- Rabajoli, G. (27 de Septiembre de 2007). *Pido Ayuda*. Recuperado el 30 de Octubre de 2013, de <http://pidoayuda.blogspot.com/2007/09/ventajas-y-desventajas-de-los.html>
- Ram6rez Hern6ndez, E. (14 de Marzo de 2011). *Universitat Polit6cnica de Val6ncia*. Recuperado el 5 de Noviembre de 2013, de Desarrollo de aplicaciones para dispositivos con sistema operativo Android:
<http://riunet.upv.es/bitstream/handle/10251/10299/Memoria.pdf?sequence=1&isAllowed=y>
- Refsnes Data Inc. (2012). <http://www.w3schools.com/>. Obtenido de <http://www.w3schools.com/>
- Ribas Lequerica, J. (2013). *Desarrollo de Aplicaciones para Android*. Espa6a: Anaya Multimedia.
- Rivera, A. (2012). *Sistemas Operativos M6viles: Comunicaci6n en tiempo real*. PCWorld.
- Sag6stegui Lescano, W. (2008). *Aprenderaprogramar*. Obtenido de http://aprenderaprogramar.com/index.php?option=com_content&view=article&id=102:ique-es-y-para-que-sirve-el-lenguaje-de-etiquetas-xml-extensible-markup-language&catid=46:lenguajes-y-entornos&Itemid=163
- Santa Maria, F. (26 de Febrero de 2014). *Staffcreativa*. Obtenido de <http://blog.staffcreativa.pe/android-ventajas-desventajas/>
- S6bastien, P. (2012). Android Guia de desarrollo de aplicaciones Smartphones y Tabletas. En P. S6bastien, *Android Guia de desarrollo de aplicaciones Smartphones y Tabletas* (p6g. 20). Barcelona: Ediciones ENI.
- S6bastien, P. (2012). Android Guia de desarrollo de aplicaciones para Smartphones y Tabletas. En P. S6bastien, *Android Guia de desarrollo de aplicaciones para Smartphones y Tabletas* (p6g. 11). Barcelona: Ediciones ENI.
- Terrero, H., & Paredes, J. (2011). *Desarrollo de Aplicaciones Java*. Fundaci6n C6digo Libre.
- Tom6s, G. (15 de Octubre de 2012). *Computer Hoy*. Recuperado el 5 de Noviembre de 2013, de <http://computerhoy.com/listas/moviles/5-mejores-versiones-android-os-1706>
- Tom6s, G. J. (2013). El gran libro de Android 3ra Edici6n. En G. J. Tom6s, *El gran libro de Android 3era Edici6n*. Barcelona: Ediciones Marcombo.

VTT Technology for business. (22 de Noviembre de 2007). *Agile.vtt.fi*. Obtenido de <http://agile.vtt.fi/mobiled.html>

W3C. (2013). <http://www.w3c.es/>. Obtenido de <http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb>