UNIVERSIDAD POLITECNICA SALESIANA

SEDE GUAYAQUIL

CARRERA: INGENIERÍA DE SISTEMAS

Tesis previa a la obtención del título de: INGENIERO DE SISTEMAS

TEMA:

SISTEMA DE GEOUBICACION TURISTICA BASADO EN SISTEMAS ANDROID, TOMANDO COMO REFERENCIA EL SECTOR NORESTE DE LA CUIDAD DE GUAYAQUIL Y SU SISTEMA DE TRANSPORTE URBANO.

AUTORES:

JUAN JAVIER MONTENEGRO BAJAÑA

LUIS LEONARDO SÁNCHEZ SÁNCHEZ

DIRECTOR:

ING. MÁXIMO TANDAZO

Guayaquil, Marzo de 2015

DECLARACIÓN DE RESPONSABILIDAD

Nosotros Montenegro Bajaña Juan Javier y Sánchez Sánchez Luis Leonardo, autorizamos a la Universidad Politécnica Salesiana la publicación total o parcial de este trabajo de grado y su reproducción sin fines de lucro.

Además declaramos que los conceptos y análisis desarrollados y las conclusiones del presente trabajo son de exclusiva responsabilidad de los autores.

_____ _____

Juan Javier Montenegro Bajaña
0918218314

Luis Leonardo Sánchez Sánchez

DEDICATORIA

Dedico este trabajo de tesis a mis padres: La señora Flora Bajaña y el señor Alberto Montenegro que gracias a su ayuda y consejos pude culminar mi carrera universitaria y ser un profesional más en la familia, a mis hermanos que siempre me brindaron su ayuda y fueron ejemplo a seguir para mí, a mi esposa y a mis hijas que son el motivo principal de seguir creciendo personal y profesionalmente y a todo el personal que forma parte de la Universidad Politécnica Salesiana que día a día forma a estudiantes inculcando conocimientos, valores y ética que les servirá a lo largo de nuestra vida profesional.

Juan Javier Montenegro Bajaña

DEDICATORIA

Dedico esta tesis a mis padres porque ellos han estado cuando yo más los he necesitado siempre han estado guiándome y siempre dándome consejos muy valiosos para mi vida personal por eso se los dedico de todo corazón esta tesis y también le dedico a mi esposa y a mi hijo y también a mis hermanos porque siempre me han apoyado y me han dado consejos positivos.

Luis Leonardo Sánchez Sánchez

AGRADECIMIENTO

Agradezco a Dios infinitamente por haberme dado la fortaleza para poder culminar mi carrera universitaria, a mis padres que fueron un pilar fundamental para mí, que con su apoyo y consejos siempre estuvieron ahí cuando yo más necesité de ellos, a mi esposa que siempre me dio fuerzas y su ayuda incondicional, a los profesores que día a día compartieron sus conocimientos y enseñanzas y a nuestro tutor el Ing. Tandazo por habernos brindado su tiempo orientándonos con consejos y mejoras para el desarrollo de nuestra tesis.

Juan Javier Montenegro Bajaña

AGRADECIMIENTO

Agradezco a Dios porque él me ha dado la fortaleza y el conocimiento para poder culminar con éxito el objetivo que me propuse realizar de obtener mi Título Profesional y agradezco de todo corazón a mis padres porque ellos han sido el pilar fundamental para lograr la meta que me propuse ellos siempre han estado en los momentos que yo más lo necesitaba y por eso y por muchas cosas más yo les quedo inmensamente agradecido y agradezco al ing. Máximo Tandazo por tenernos la paciencia en cada revisión y por los consejos que aportaba para nuestro tema de tesis.

Luis Leonardo Sánchez Sánchez

INDICE

INDICE DE FIGURAS

INDICE DE ANEXOS

RESUMEN

En los últimos años, se ha observado un crecimiento considerable en la disponibilidad y el uso de las redes inalámbricas para una conexión a internet, el acceso a servicios remotos por medio de dispositivos móviles y diversidad de aplicaciones; en este caso nos centraremos en diseñar una aplicación que pueda ser utilizada en cualquier dispositivo que utilice sistema operativo Android.

Esta aplicación facilitará a los habitantes del sector Centro-Noreste de la ciudad de Guayaquil, poder llegar a su lugar de destino haciendo uso del transporte urbano y cómo hacerlo a través de una amigable aplicación móvil, que mostrará el punto exacto donde se encuentra a través del Sistema de Posicionamiento Global (GPS), además esta aplicación mostrará que líneas de buses transitan cerca del lugar a donde desea llegar, adicionalmente le permitirá buscar sitios de referencia como hoteles, centros comerciales, hospitales, etc, permitiendo a los Guayaquileños y turistas conocer un poco más acerca del transporte urbano de la ciudad.

La meta de este proyecto es aprovechar la tecnología para ayudar a los ciudadanos de determinado sector de la ciudad facilitándoles información del transporte urbano e incentivándoles el uso del mismo de manera adecuada.

ABSTRACT

In the recent years, it has been observed a considerable increase in the availability and the use of wireless networks for internet connection, the access to remote services through mobile devices and the different applications, in this case will focus this project on designing on application that can be used in any type of device with on Android Operating System.

This Application will provide the residents of the center-northwest area of Guayaquil, to reach their destination making use of the urban transportation and noting how to do it, through a user-friendly application, it will shows the exact point where the person is by using the GPS, well this application will shows that bus lines pass near the place where you want to reach additionally allows you to search reference sites such as hotels, shopping malls, hospitals, etc, allowing a the tourists learn more about urban transport city.

The idea of this project is to leverage technology to help the residents of certain sector of the Guayaquil city will provide urban transport information and encouraging them use it properly.

INTRODUCCIÓN

Actualmente la comunicación juega un papel muy importante en la sociedad, ya que permite la constante interrelación y comunicación entre personas, sociedades, empresas, etc. Con el paso del tiempo, la necesidad de estar cada vez más comunicados se hizo mayor.

Los avances en la tecnología han ido aumentando en la historia de estos equipos. Los celulares han ido evolucionando mucho en cuanto a su forma, rendimiento, características, desarrollo, incluso hasta en su sistema operativo.

Uno de los sistemas operativos de los dispositivos móviles inteligentes es Android, es un sistema operativo Open Source, una plataforma de desarrollo bajo Linux y de código abierto, las aplicaciones finales son desarrolladas bajo Java asegurando que podrán ser ejecutadas en una gran variedad de dispositivos.

Con la tecnología Android se desarrollará una aplicación que facilitará el recorrido de las líneas urbanas del sector noroeste de Guayaquil, facilitándole ver el punto exacto donde se encuentra a través del GPS, además esta aplicación mostrará que líneas transitan cerca del lugar a donde desea llegar, permitiendo a los guayaquileños tener diferentes opciones de transportarse, ahorrando tiempo y dinero.

CAPÍTULO I
PLANTEAMIENTO DEL PROBLEMA

1.1. Problema de la Investigación

1.1.1 Planteamiento del problema de investigación

Con el pasar del tiempo la ciudad ha ido creciendo con urbanizaciones e invasiones a los alrededores de Guayaquil, por lo que también ha aumentado en números de líneas urbanas y los vehículos de alquiler llamados taxis e incluyendo la población que va incrementando.

La mayoría de los habitantes de la ciudad de Guayaquil, a pesar de ser ciudadanos y de radicar aquí, desconocen las líneas de nuestro sistema de transporte urbano por lo que se pierden al buscar un lugar, por lo que llegan a una única opción de hacer uso de un transporte particular llamado Taxi, por el cual pagarán un monto más elevado en comparación con el transporte urbano.

Al encontrar este problema existe la necesidad de diseñar una aplicación que ayude a los habitantes de determinado sector llegar al lugar deseado, facilitándole información de líneas urbanas que pasen por un punto de ubicación y llegar al lugar destino ingresándolo previamente como un parámetro de entrada y permita elegir la línea más adecuada que llevará al destino ahorrando dinero.

- Desconocimiento del recorrido de los buses urbanos.
- Gasto innecesario de recursos (dinero).
- Pérdida de tiempo por una mala planificación y desconocimiento de la información.
- Optar por servicios particulares de transporte por desconocimiento e inseguridad.

1.1.2 Formulación del Problema de investigación

¿De qué manera se podría ayudar a los habitantes del sector noroeste de Guayaquil ahorrar dinero, eligiendo un sistema de transporte más adecuado para dirigirse a otro punto determinado de la ciudad?

A. ¿Qué beneficio existe al proporcionar información de los recorridos de las líneas urbanas a los habitantes del sector noroeste de Guayaquil?
B. ¿Qué tecnología podría usarse para efectuar este proyecto ahorrando dinero en el pago de licencias?
C. ¿Qué aspectos están incidiendo para que los habitantes del sector noroeste de Guayaquil elijan un servicio particular de transporte para dirigirse a otro punto de la ciudad?

1.1.3 Sistematización del problema

¿Qué beneficios dará la aplicación móvil a la sociedad, mostrando el recorrido de los buses urbanos del sector noroeste de Guayaquil?
¿Qué influencia positiva tendrá el avance tecnológico móvil ante la sociedad ecuatoriana?
¿Qué aportes dará el uso de esta aplicación móvil en la ciudad?

1.2 Objetivos

1.2.1 Objetivo General

Desarrollar e implementar una aplicación móvil para dispositivos con sistema Android que muestre el recorrido de líneas urbanas que circulen por el sector Centro-Noreste de la ciudad y ayude a escoger la línea adecuada para dirigirse a su lugar destino.

1.2.2 Objetivos específicos

A. Identificar las líneas de buses urbanos que circulen por nuestra área o zona de referencia e investigar el recorrido de inicio a fin para poder hacer uso de esa información en la aplicación.

B. Identificar los diferentes puntos de intersección entre las líneas que circulan por la zona de referencia.

1.3 Justificación

Gracias al gran auge que han tenido los dispositivos móviles dentro de las preferencias de conectividad de las personas y de las grandes prestaciones que se puede obtener de este tipo de dispositivos se pueden aprovechar para diseñar diferentes aplicaciones para ayudar a la comunidad.

La meta de este proyecto es aprovechar la tecnología para ayudar a los ciudadanos de determinado sector de Guayaquil a través de una aplicación en Android que muestre a través de un mapa nuestra ubicación, indicando las líneas urbanas que recorren la ciudad y llegar al lugar de destino. Así también incentivará a los demás alumnos a que puedan desarrollar nuevas aplicaciones para equipos inteligentes.

Los Beneficiarios serán los habitantes del sector centro-noreste de la ciudad de Guayaquil, teniendo en cuenta que con la ayuda de esta aplicación se fomentaría el turismo en la ciudad.

CAPÍTULO 2
MARCO TEÓRICO

2.1 Marco Teórico

2.1.1 Android

Figura # 1. Representación de sistema operativo Android
Nota: Representación gráfica del sistema operativo Android. (Wikipedia, Septiembre 2007)

Android es un sistema operativo basado en el núcleo Linux. Fue diseñado principalmente para dispositivos móviles con pantalla táctil, como teléfonos inteligentes o tablets; y también para relojes inteligentes, televisores y automóviles. Inicialmente fue desarrollado por Android Inc., empresa que Google respaldó económicamente y más tarde, en 2005, compró. Android fue presentado en 2007 junto la fundación del Open Handset Alliance (un consorcio de compañías de hardware, software y telecomunicaciones) para avanzar en los estándares abiertos de los dispositivos móviles.

El éxito del sistema operativo se ha convertido en objeto de litigios sobre patentes en el marco de las llamadas «Guerras por patentes de teléfonos inteligentes» (en inglés, Smartphone patent wars) entre las empresas de tecnología. Según documentos secretos filtrados en 2013 y 2014, el sistema operativo es uno de los objetivos de las agencias de inteligencia internacionales.

La versión básica de Android es conocida como Android Open Source Project (AOSP).

El 25 de junio de 2014 en la Conferencia de Desarrolladores Google I/O, Google mostró una evolución de la marca Android, con el fin de unificar tanto el hardware como el software y ampliar mercados. Para ello mostraron nuevos productos como Android TV, Android Auto, Android Wear o una serie de "Smartphone" de baja gama bajo el nombre de Android One. Esto sirvió para estabilizar la imagen de la marca de cara a los mercados y al público

¿Qué hace que Android sea especial?

Como se ha comentado, existen muchas plataformas para móviles (iPhone, Symbian, Windows Phone, Blackberry, Palm, Java Mobile Edition, Linux Móviles (Limo), etc.); sin embargo Android presenta una serie de características que lo hacen diferente. Es el primero que combina, en una misma solución, las siguientes cualidades son: (Girones, 2012)

Plataforma realmente abierta. Es una plataforma de desarrollo libre basada en Linux y código abierto. Una de sus grandes ventajas es que se puede usar y "customizar" el sistema sin pagar royalties.

Portabilidad Asegurada. Las aplicaciones finales son desarrolladas en java, lo que asegura que va a ser ejecutadas en una gran variedad de dispositivos, tanto presentes como futuros. Esto se consigue gracias al concepto de máquina virtual.

Arquitectura basada en componentes inspirados en internet. Por ejemplo el diseño de la interfaz de usuario se hace XML, lo que permite que una misma aplicación se ejecute en un móvil de pantalla reducida o en una netbook.

Filosofía y dispositivos siempre conectados a internet.

Gran variedad de servicios incorporados. Por ejemplo, localización basada tanto en GPS como en redes, base de datos con SQL, reconocimiento y síntesis de voz, navegador, multimedia, etc.

Aceptable a nivel de seguridad. Los programas se encuentran aislados unos de otros gracias al concepto de ejecución dentro de una caja que hereda de Linux. Además, cada aplicación dispone de una serie de permisos que limitan su rango de actuación (servicios de localización, acceso a Internet, etc.).

Optimizado para baja potencia y poca memoria. Por ejemplo, Android utiliza la máquina virtual Dalvik. Se trata de una implementación de Google de la máquina virtual de Java optimizada para dispositivos móviles.

Alta calidad de gráficos y sonido. Gráficos vectoriales suavizados, animaciones inspiradas en flash, gráficos en 3 dimensiones basadas en OpenGL. Incorpora los códecs estándar más comunes audio y video, incluyendo H.264 (AVC), MP3, ACC, etc.

Comparativa con otras plataformas. Se describe las principales características de las plataformas móviles disponibles en la actualidad. Dado la gran cantidad de datos que se indican, se ha utilizado una tabla para representar la información. (Soriano, 2012)

Compañía	Apple	Open Handset	Windows	RIM	Symbian
Núcleo del S.O	Mac OS X	Linux	Windows CE	Movile OS	Movile
Lenguaje de programación	Objetive-C,C++	Java, C++	C#, muchos	Java	C++
Licencia de Software	Propietaria	Software libre y abierto	Propietaria	Propietaria	Software libre
Año de lanzamiento	2007	2008	2010	2003	1997
Motor del	Webkit	Webkit	Pocket	Webkit	Webkit

navegador web			Internet		
HTML5	Si	Si	No	Si	Si
Tienda de Aplicaciones	App Store	Google Play	Windows Marketplace	Blackberry App World	OviStore
Numero de Aplicaciones	400.000	300.000	50.000	30.000	50.000
Coste Publicar	$99 al año	$25 una sola vez	$99 al año	Sin coste	$1 una sola vez

FIGURA #2. Comparativa de plataformas móviles.

Nota: Tabla comparativa de las principales plataformas móviles. (Android, Mayo 2012)

2.1.2 Arquitectura de Android

El siguiente grafico muestra la arquitectura de Android. Como se puede ver, está formada por cuatro capas. Una de las características más importante es que las capas están basadas en Software libre.

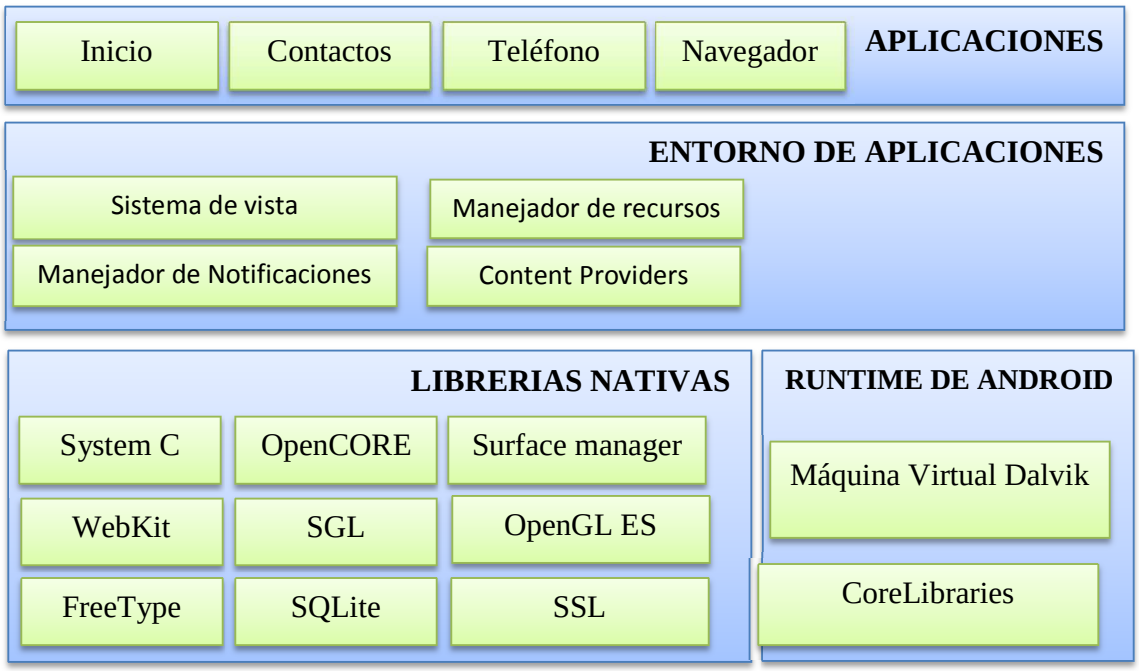


FIGURA # 3. Arquitectura de Android

Nota: Esquema que muestra la arquitectura de android. (Android, Mayo 2012)

El núcleo Linux

El núcleo de Android está formado por el Sistema Operativo Linux, versión 2.6. 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. (Ableson, 2011)

Esta capa 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 maquia virtual utilizado en Java. Dado las limitaciones de los dispositivos donde ha de ejecutarse Android (poca memoria y procesador limitado) no fue posible utilizar una máquina virtual Java estándar. Google tomo la decisión de crear una nueva, la máquina virtual Dalvik, que respondiera mejor estas limitaciones.

Algunas características de la máquina virtual utilizada en 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 corre en su propio proceso Linux algunas funciones como threading y el manejo de la memoria a bajo nivel.

2.1.3 Librerías Nativas

Incluye un conjunto de librerías en C/C++ usadas en varios componentes de Android. Están compiladas en el código abierto nativo del procesador. Muchas de las librerías utilizan proyectos de código abierto. En, (Jesús Tomás, 2011), se dice que algunas de estas librerías son:

- **System C library:** una derivación de la librería BSD de C estándar (libc), adaptada para dispositivos embebidos basado en Linux.

- **Media Framework:** Librería basada en PacktVideo'sOpenCORE; soporta codecs de reproducción ya grabación de multitud de formatos de audio, video e imágenes MPEG4, H.264, ACC, AMR, JPG y PNG.

- **Surface Manager:** maneja el acceso al subsistema de representación gráfica en 2D y 3D.

- **WebKit:** 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.

- **Librería 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 base de datos relacionales disponible para todas las aplicaciones.

- **SSL:** proporciona servicios de encriptación Secure Socket Layer.

Entorno de aplicación

Esta capa ha sido 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 remplazar 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 en una fracción muy significativa de la misma.

En (Jesús Tomás, 2011), se dice que los servicios más importantes que incluye son:

- **Views:** Extenso conjunto de vistas (parte de los componentes).

- **Resource Manager:** proporciona acceso a recursos que no son en el código.

- **Activity Manager:** maneja el ciclo de vida de las aplicaciones y proporciona un sistema de navegación entre ellas.

- **Notificación Manager:** permite a las aplicaciones mostrara 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 maquina Android. Todas las aplicaciones han de ser ejecutadas en la máquina virtual Dalvik para garantizar la seguridad del sistema.

Normalmente, las aplicaciones Android están escritas en Java. Para desarrollar aplicaciones en Java se puede utilizar el Android SDK. Existe otra opción consistente en desarrollar las aplicaciones utilizando C/C++. Para esta opción se puede utilizar el Android NDK (NativeDevelopment Kit).

2.1.4 Android en el mercado

La compañía de investigación de mercado Canalys estima que en el segundo trimestre de 2009, Android tendría 2,8% del mercado de teléfonos inteligentes en el ámbito mundial.

En febrero de 2010, ComScore dijo que la plataforma Android tenía el 9% del mercado de teléfonos inteligentes en los Estados Unidos, como estaba tasado por los operadores. Esta cifra fue superior al estimado anterior de noviembre de 2009, el cual fue del 9%.Para finales del tercer trimestre de 2010, el mercado de Android en los Estados Unidos había crecido en un 21,4%.

En mayo de 2010, Android superó en ventas a iPhone, su principal competidor. De acuerdo a un informe del grupo NPD, Android obtuvo un 28% de ventas en el mercado de los Estados Unidos, un 8% más que en el trimestre anterior. En el segundo trimestre de 2010, los dispositivos iOS incrementaron su participación en un 1%, indicando que Android está tomando mercado principalmente de RIM. Adicionalmente, los analistas apuntaron que las ventajas de que Android fuera un sistema multi-canal, multi-operador, le permitiría duplicar el rápido éxito que obtuvo el sistema Windows Mobile de Microsoft.

A principios de octubre de 2010, Google agregó 20 países a su lista de lugares geográficos donde los desarrolladores pueden enviar aplicaciones. Para mediados de octubre, la compra de aplicaciones estaba disponible en un total de 32 países.

En noviembre de 2013 Andy Rubin dijo que se activaban 1.500.000 dispositivos diariamente y también. En diciembre de 2011 Andy Rubin dijo que se activaban 700.000 dispositivos diariamente, anteriormente en julio de 2011 se declaró que se activan unos 550.000 dispositivos Android cada día, en comparación con diciembre de 2010 que se activaban 300.000 dispositivos móviles con Android, y los 100.000 que se activaban en mayo de 2010.

En abril de 2013 se hizo público que Android alcanzó el 92% en ventas de nuevos Smartphone para el trimestre comprendido entre diciembre 2012 y febrero 2013 en España, seguido de iOS con un 4.4% (Cosmos, 2015)

Versiones de Android

Version	Codename	API	Distribution
2.2	Froyo	8	0.4%
2.3.3 - 2.3.7	Gingerbread	10	6.9%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	5.9%
4.1.x	Jelly Bean	16	17.3%
4.2.x		17	19.4%
4.3		18	5.9%
4.4	KitKat	19	40.9%
5.0	Lollipop	21	3.3%

FIGURA # 4. Versiones de Android en el mercado

Nota: Tabla que muestra porcentaje de versiones de android en el mercado. (Xatakandroid, Marzo 2015).

Las nuevas versiones And**roid 4.4 Kitkat y Android 5.0 Lollipop** siguen expandiéndose en el mercado, el 40,9% de los dispositivos Android están actualizados a Kitkat y **Lollipop el 3,3%**, siendo la versión más reciente de Android y la que mayor crecimiento ha tenido.

El resto de versiones siguen en su ya habitual decadencia. Android 4.3 está en el 5,9% (antes 6,3%), Android 4.2 está en el 19,4% (antes 19,8%), Android 4.1 está en el 17,3% (antes 18,4%), Android 4.0 está en el 5,9% (antes 6,4%), Android 2.3 está en el 6,9% (antes 7,4%) y Android 2.2 se mantiene con el 0,4%.

Durante los próximos meses por fin Android 5.0 Lollipop comienza a expandirse con una mayor velocidad, gracias a las actualizaciones de las gamas de 2014 y a la nueva generación de dispositivos que ya están saliendo al mercado con la nueva versión de Android.

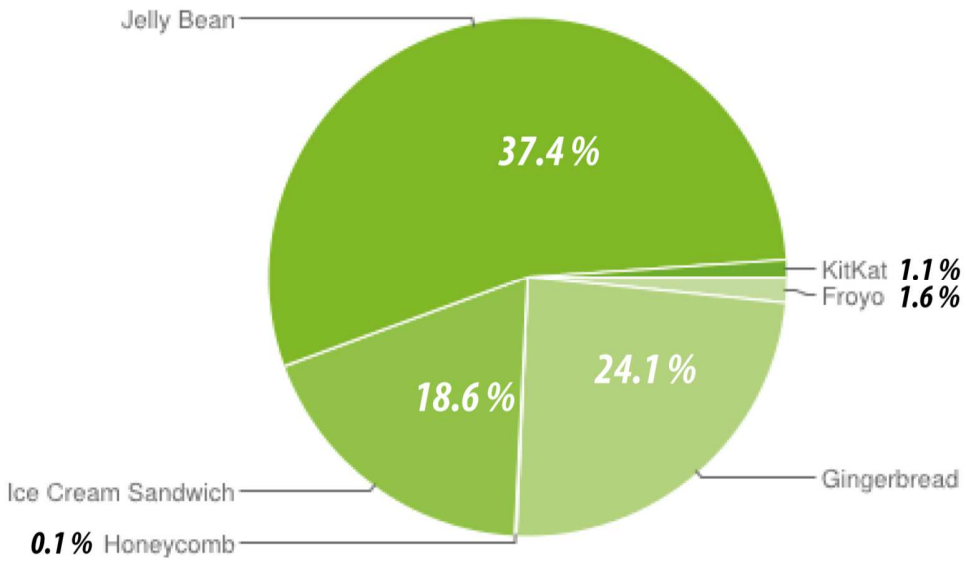


FIGURA # 5. Porcentaje de distribución de Versiones de Android

Nota: Tabla que muestra porcentaje de versiones de android. (Vierco, 2010).

Desde que se creó el sistema operativo Android hasta la actualidad ha lanzado algunas versiones, con el objetivo de mejorar el rendimiento y la interfaz para el usuario, entre las cuales se puede mencionar las más importantes:

Eclair.- Se aumentó la resolución en la pantalla en cuanto a la versión anterior Donut, debido a los nuevos dispositivos móviles. Incluyo ser multitouch y el uso de gestos para interactuar con el usuario.
Froyo.- Se mejoró el rendimiento, soporte a Adobe Flash. Permitió a los desarrolladores el uso de la memoria externa SD para instalar aplicaciones.

Gingerbread.- Se mejoró la velocidad en la interfaz con el usuario. Se presentó el NFC (*Nearfield Communication*) en los dispositivos móviles Android, para permitir el intercambio de datos a través de la comunicación inalámbrica.

Honeycomb.- Pensada y diseñada para tablets. Donde cabe destacar la conectividad USB y el soporte de teclados.

Ice Cream Sandwich.- Trajo novedades en la interfaz y nuevas carpetas para accesos directos. Reunificación entre las versiones para Smartphone y tablets.

Jelly Bean.- generar fotos de 360 grados y la inclusión de Google Now. Además permite poseer varias cuentas en un dispositivo pero solo en tablets. En el teclado se añade escritura por gestos. (Condesa, Androideity, 2012)

Kit Kat.- Da prioridad a contactos en función de las personas que más te llaman. Al tiempo que permite realizar búsquedas de lugares cercanos, negocios y contactos.

Cabe indicar que todas las versiones de Android oficiales al ser código abierto puedes ser modificadas por terceros agregando así múltiples funciones características que no poseen. (Cosmos, 2015)

2.1.5 Lenguaje de Programación PHP

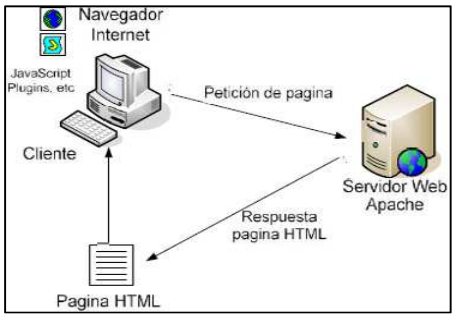


FIGURA # 6. Arquitectura de PHP

Nota: Grafico que muestra la arquitectura de php. (PHP, 2005).

Es un lenguaje de programación, relativamente nuevo (su antecesor, PHP/FI, data de finales de 1994), concebido principalmente como herramienta para el desarrollo de aplicaciones web.

PHP permite diseñar páginas dinámicas de servidor, es decir, genera páginas bajo petición capaces de responder de manera inteligente a las demandas del cliente y que permita la automatización de gran cantidad de tareas. Si se tuviera definir PHP en una sola línea, se podría decir que es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor. (Rodríguez, 2005)

Aunque existe una multitud de lenguajes y entornos de desarrollo concebidos para internet, PHP se ha convertido en uno de los lenguajes, del lado del servidor, más ampliamente utilizados para el desarrollo de páginas dinámicas junto con ASP, JSP, Coldfusion y Perl.

En los últimos años, el número de servidores que utilizan PHP se ha disparado. De hecho, según datos de Netcraft a fecha de Agosto de 2004 son casi 17 millones de dominios los que usan PHP.

En PHP se combinan muchas características que contribuyen notablemente a su masiva utilización: entre otras cosas, está el hecho de ser software de libre distribución y

multiplataforma (existen versiones de PHP para U*ix, win32, mac OSX, etc.) que sigue la filosofía Open Source. También ha contribuido con éxito el hecho de haberse convertido en el complemento ideal para el popular tándem Linux Apache en el desarrollo de sitios web. Pero lo más destacable del lenguaje y una de las características que más ha influido en su popularización es la sencillez de uso.

¿Qué se puede hacer con PHP?

Aunque principalmente se utiliza para programar scripts que van a ser ejecutados en servidor Web, no hay que olvidar que se puede utilizar en cualquier otro lenguaje (Perl, C, Python, Shell, etc.) para escribir programas que se ejecuten desde las líneas de comandos, es decir sin la necesidad que se ejecute conjuntamente con el servidor Web.

De todas formas, es el entorno Web donde ha conseguido su mayor aceptación, y es PHP no solo permite realizar todas las acciones propias de un scripts CGI tradicional (procesamiento de formularios, manipulación de cookies, generación de páginas con contenido dinámicos…), sino que también proporciona las siguientes posibilidades:

- Soporte para múltiples sistemas operativos: Unix (entre otras, Linux, HP-UX, Solaris, y OpenBSD), Microsoft Windows, Mac, OS X, RIC OS. Actualmente está en preparación para las plataformas IBM OS/390 y AS/400.
- Soporte para múltiples servidores Web: Apache, Microsoft Internet Information Server, Personal Web Server, Netscape e iPlanet, OreillyWebsite Pro Server, Caudium, Xitami, OmniHTTPd y muchos otros.
- Soporte para más de 25 gestores de base de datos: Adabas D, Ingres, Oracle, dBase, InterBase, Ovrimos, Empress, FrontBase, PostgreSQL, Msql, Solid, Hyperwave, Direct MY-SQL, Sybase, IBM DB2, Informix, Unix dbm y MySQL, entre otras.
- Soporte para ODBC, y extensiones DBX.
- Soporte para comunicarme con otros servicios usando protocolos tales como LDPA, IMAP, SNMP, NNTP, POP3, HTTP, COM (en Windows) y muchos otros.
- Puede usar objetos Java de forma transparente, como objetos PHP.
- La extensión CORBA puede ser utilizada para acceder a objetos remotos.

- PHP soporta WDDX para intercambio de datos entre lenguajes de programación Web.
- Generación de resultados en múltiples formatos como XHTML, XML, ficheros de imágenes, ficheros PDF y películas flash…
- Parser de documentos XML, soporte de los estándares SAX y DOM. Manejo de XSLT para transformar documentos XML.
- Manejo de expresiones regulares POSIX Extended o Perl.
- Funciones de comercio electrónico, como cybercash, cyberMUT, VeriSingPayflow Pro y CCVS para las pasarelas de pago.
- Otras extensiones muy interesantes son las funciones del motor de búsqueda mnoGoSEACH, funciones para pasarelas de IRC, utilidades de compresión (gzip, bz2), conversión de calendarios.

Comparativa entre ASP y PHP

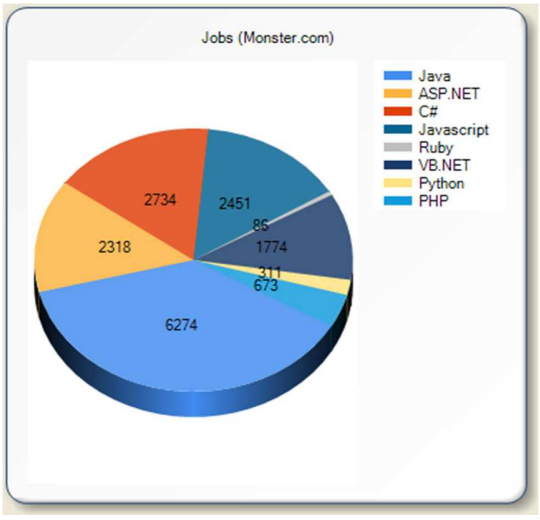


FIGURA # 7.

Comparativa de Asp y Php

Nota: Grafico que muestra comparativa de php y asp. (Vaswani, 2009)

El otro lenguaje utilizado para el diseño de páginas dinámicas de servidor y ampliamente extendido ASP (Active Server Page). Es un lenguaje derivado de Visual Basic (aunque se puede programar con otros lenguajes como VBScript y JScript) desarrollado por Microsoft. Evidentemente, se emplea principalmente sobre plataformas que funcionan

bajo sistemas operativos Windows (aunque desde hace poco tiempo existe un software de SUN, Sun ONE Active Server Pages, anteriormente conocido como chili Soft ASP, que permite trabajar con ASP en plataformas Unix/Linux). (Holzner, 2009)

PHP es en las plataformas Unix lo que ASP en las plataformas Windows. De hecho, ambos lenguajes se insertan en documentos HTML haciendo uso de emulación de etiquetas (otros lenguajes como Perl deben generar toda la página HTML de respuesta).

La principal diferencia es que ASP es una tecnología propietaria de Microsoft, mientras que PHP sigue la filosofía Open Source. Esto hace que ASP esté pensando para funcionar sobre plataforma Microsoft, a pesar de que existan soluciones con un coste económico elevado como SUN ONE ASP que permite su utilización sobre Unix/Linux. Sin embargo, PHP permite que sin ningún problema se pueda migrar las aplicaciones Web entre todos los sistemas operativos y servidores en los que funciona.

PHP por Internet

ES muy probable que su proveedor de servicios de Internet (PSI) ya cuente con soporte de PHP – puede preguntar a su personal de soporte o intentar la carga y ejecución de un archivo PHP. (Vaswani, 2009)

PHP DATA OBJECT

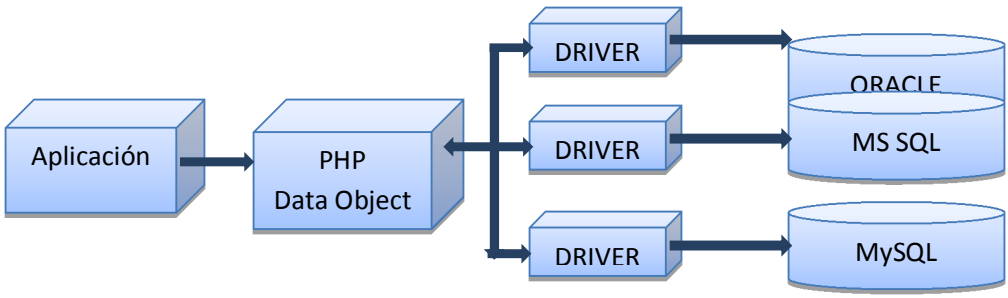


FIGURA # 8. Esquema de trabajo con PHP PDO

Nota: Grafico que muestra esquema de trabajo de php. (Vaswani, 2009)

El mayor problema cuando se desarrolla un aplicación PHP es cuando se cambia de motor de base de datos, salvo que se aplique el patrón de diseño DAO con abstract Factory y cada implementación del DAO seria para un motor diferente, pero aun si se trata de librerías diferentes con sus propios nombres de funciones y parámetros, y tener que lidiar con el SQL malicioso (SQL Injection). (Coronel, 2010)

Por ejemplo cuando se utiliza las funciones de las librerías de MySQL se tiene que validar cada una de las variables que contienen la sentencia SQL que pase para su ejecución.

Primero se tendría que verificar que **magicquotes** se encuentre desactivado, luego utilizando muchas funciones como por ejemplo addslashes() y mysql_real_escape_string(), verificar que no quede ningún agujero en tu código.

2.1.6 Motor de Base de Datos MySQL

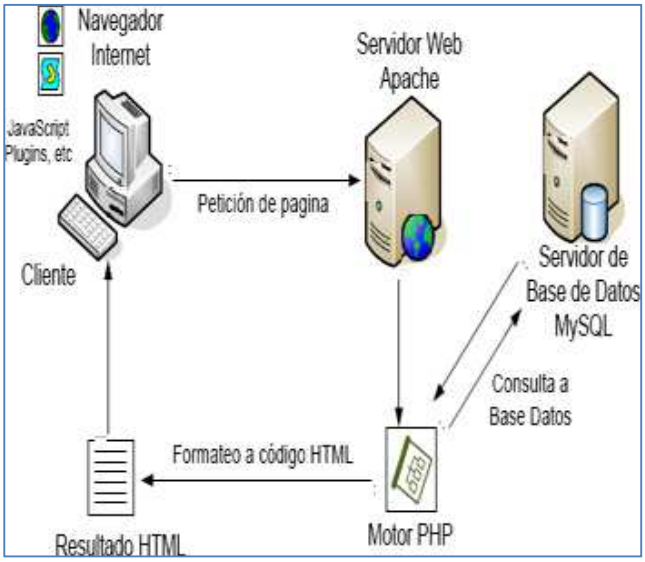


FIGURA # 9. Base de Datos Msql

Nota: Grafico que muestra gestor de base de datos Msql. (Coronel, 2010)

Es un sistema gestor de base de datos relacionales, que además ofrece compatibilidad con PHP, Perl, C y HTML, y funciones avanzadas de administración y optimización de base de datos para facilitar las tareas habituales. Implementa funcionalidades Web,

permitiendo un acceso seguro y sencillo a los datos a través de Internet. Este sistema gestor de base de datos incluye capacidades de análisis integradas, servicios de transformación y duplicación de datos y funciones de programación mejoradas.

Se puede decir que MySQL es un sistema cliente servidor de administración de base de datos relacionales diseñado para el trabajo tanto en los sistemas operativos Windows como en los sistemas Unix/Linux. Además determinadas sentencias de MySQL pueden ser embebidas por código PHP y HTML para diseñar aplicaciones Web. Así mismo, es compatible con el software más potente de diseño Web, como dreamweaver MX.
MySQL es un sistema de gestor de base de datos relacional cliente-servidor de coste mínimo que incluye un servidor SQL, programas clientes para acceder al servidor, herramientas administrativas y una interfaz de programación para escribir programas. MySQL es portable y se ejecuta en sistemas operativos comerciales como Linux y Windows. (López, 2008)

Estructura de la Base de Datos

Los sistemas de gestión de base de datos (abreviado mediante SGBD o DBMS) organizan y estructuran los datos de tal forma que puedan ser recuperados y manipulados por usuarios y programas de aplicación. Las estructuras de los datos y las técnicas de acceso proporcional por un DBMS particular se denominan modelo de datos. El modelo de datos determina la personalidad de un DBMS y las aplicaciones para las cuales está particularmente bien formado.

Existe un tipo de lenguaje estándar normalizado para trabajar con base de datos denominado SQL (StructuredQueryLanguaje). SQL es un lenguaje de base de datos para base de datos relacionales y utiliza el modelo de datos relacionales.

Sqlite

FIGURA # 10. Representación de SQLITE

Nota: Representación de Sqlite. (López, 2008)

Definición

SQLite® es una biblioteca de bases de datos relacionales de código abierto. Se ha diseñado para efectuar un uso eficaz de los recursos de la memoria e incluye pocas funciones. Como consecuencia, puede resultar una buena elección para aplicaciones incrustadas e inalámbricas. Los dispositivos BlackBerry® que ejecutan BlackBerry® Device Software versión 5.0 o posterior tienen la biblioteca de SQLite integrada en el sistema operativo y en la máquina virtual. La API de SQLite permite desarrollar aplicaciones que usan la base de datos SQLite integrada. Se presentó en BlackBerry® Java® DevelopmentEnvironment versión 5.0.

SQLite es una librería escrita en lenguaje C que implementa un manejador de base de datos SQL embebido.

Los programas que se enlacen con la librería SQLite pueden tener acceso a una base de datos SQL, sin tener que ejecutar un programa de RDBMS separado.

SQLite es Software Libre por lo tanto el código fuente es del dominio público y licencia GPL.

Además se puede entender como bases de datos empotradas aquellas que no inician un servicio en nuestra máquina independiente de la aplicación, pudiéndose enlazar directamente a nuestro código fuente o bien utilizarse en forma de librería (include)". (The php group, 2013)

Clases de Almacenamiento y tipos de datos

Para maximizar la compatibilidad entre SQLite y otros motores de BBDD, SQLite soporta el concepto de "Afinidad Tipo" en las columnas. La afinidad tipo de una columna es el tipo recomendado para el valor almacenado en esta columna.

La idea central es que este tipo es el recomendado pero no obligatorio. Cualquier columna puede almacenar cualquier tipo de dato. Es solo que algunas columnas tienen que escoger usar una clase de almacenamiento sobre otra. La clase de almacenamiento para una columna es llamada afinidad.

Cada valor almacenado en una BBDD SQlite tiene uno de las siguientes clases de almacenamiento:

- **NULL**, un valor nulo.

- **INTEGER**, un entero con signo que se almacena en 1, 2, 3, 4, 5, 6 o 8 bytes de acuerdo a la magnitud del valor.

- **REAL**, un número de coma flotante (real), almacenado en 8 bytes.

- **TEXT**, una cadena de texto almacenada con las codificaciones UTF-8, UTF-16BE o UTF-16-LE.

- **BLOB**, datos en formato binario, se almacenan exactamente como se introdujeron.

Características de Sqlite

SQLite es un sistema completo de bases de datos que soporta múltiples tablas, índices, triggers y vistas. No necesita un proceso separado funcionando como servidor ya que lee y escribe directamente sobre archivos que se encuentran en el disco duro. El formato de la base de datos es multiplataforma e indistintamente se puede utilizar el mismo archivo en sistemas de 32 y 64 bits.

La base de datos se almacena en un único fichero a diferencia de otros DBMS que hacen uso de varios archivos. SQLite emplea registros de tamaño variable de forma tal que se utiliza el espacio en disco que es realmente necesario en cada momento.

El código fuente está pensado para que sea entendido y accesible por programadores promedio. Todas las funciones y estructuras están bien documentadas.

Existe un programa independiente de nombre sqlite que puede ser utilizado para consultar y gestionar los ficheros de base de datos SQLite. También sirve como ejemplo para la escritura de aplicaciones utilizando la biblioteca SQLite.

Ventajas de Sqlite

- **Tamaño:** SQLite tiene una pequeña memoria y una única biblioteca es necesaria para acceder a bases de datos, lo que lo hace ideal para aplicaciones de bases de datos incorporadas.

- **Rendimiento de base de datos:** SQLite realiza operaciones de manera eficiente y es más rápido que MySQL y PostgreSQL.

- **Portabilidad:** se ejecuta en muchas plataformas y sus bases de datos pueden ser fácilmente portadas sin ninguna configuración o administración.

- **Estabilidad:** SQLite es compatible con ACID, reunión de los cuatro criterios de Atomicidad, Consistencia, Aislamiento y Durabilidad.

- **SQL:** implementa un gran subconjunto de la ANSI – 92 SQL estándar, incluyendo sub-consultas, generación de usuarios, vistas y triggers.

- **Interfaces:** cuenta con diferentes interfaces del API, las cuales permiten trabajar con C++, PHP, Perl, Python, Ruby, Tcl, Groovy, Qt ofrece el plugin sqlite, etc.

- **Costo:** SQLite es de dominio público, y por tanto, es libre de utilizar para cualquier propósito sin costo y se puede redistribuir libremente.

Desventaja de Sqlite

- **Limitaciones en where:** Esta limitación está dada por el soporte para clausuras anidadas.

- **Falta de clave Foránea:** Cuando se realice la creación de la tabla desde el modo consola, está permitido el uso de la clausura, aunque no realizara el chequeo de la misma. (Ecured, 2014)

2.1.7 Kit de Desarrollo de Software de Android (SDK)

El SDK de Android es un conjunto de herramientas para el desarrollo de una aplicación Android, y comprende un depurador de código, biblioteca, un simulador de teléfono

basado en QEMU, documentación, ejemplos de código y tutoriales para facilitar crear aplicaciones.

Eclipse es la herramienta por excelencia para desarrollar aplicaciones Android, el problema es que la instalación de Android es lenta y complicada. Pero eso era antes de que Google lanzara su propia versión de Eclipse que ya viene con el SDK y todas las herramientas de desarrollo instaladas.

El paquete de desarrollo incluye:

- Eclipse Juno, la más reciente versión de este IDE.
- El SDK de Android todas las herramientas de desarrollo y emuladores.
- La versión de ADT (Android Developer tools) preinstalada en Eclipse
- Los paquetes y la máquina virtual de Android 4.2 (Jelly Bean).

Android SDK es el kit de desarrollo necesario para programar e implementar todo tipo de aplicaciones para Android, el sistema operativo para teléfonos móviles de Google.

Este paquete o kit de desarrollo incluye las APIs y herramientas necesarias para desarrollar las aplicaciones utilizando JAVA como lenguaje de programación y testear el código, respectivamente.

Las librerías adjuntas en Android SDK son compatibles con los siguientes entornos de desarrollo: Eclipse, JDK5 o JDK6, Android Development Tool Plugin y Apache Ant.

Cambios recientes

- Soporte para CDMA
- Soporte para resoluciones adicionales
- Soporte para QVGA y WVGA
- Reconocimiento de gestos y de voz
- Posibilidad de integrar Quick Search Box

2.1.8 API (Interfaz de Programación de Aplicaciones)

Se define un API como el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece ciertas bibliotecas para ser utilizado por otro software como una capa de abstracción. Son generalmente usadas en las bibliotecas por que incluye un conjunto de funciones de uso general.

Google Maps Android API V2

En (Google, 2013), se menciona que Google Maps proporciona un servicio de cartografía online que se puede utilizar en las aplicaciones Android. Las claves necesarias para utilizarlo. Se estudiará la versión 2 del API que incorpora interesantes ventajas respecto a la versión anterior. Entre estas ventajas destaca el menor tráfico intercambiado con el servidor, la utilización de fragments y los gráficos en 3D. Como inconveniente resaltar que la nueva versión solo funciona en el dispositivo con Google Play instalado.

Conviene destacar que a diferencia de Android, Google Maps no es un software libre, por lo que está limitado a una serie de condiciones de servicio. Se puede usar de forma gratuita siempre que nuestra aplicación no solicite más de 15.000 codificaciones geográficas al día. Se puede incluir propaganda en los mapas o incluso se puede usarlo en aplicaciones móviles de pago (para sitios Web de pago es diferente).

En, Morales (2012), se dice que entre las características se puede mencionar lo siguiente:

- Integración con los Servicios de Google Play (*Google Play Services*) y la Consola de APIs.
- Utilización a través de un nuevo tipo específico de fragment *(*MapFragment*)*, una mejora muy esperada por muchos.
- Utilización de mapas vectoriales, lo que repercute en una mayor velocidad de carga y una mayor eficiencia en cuanto a uso de ancho de banda.
- Mejoras en el sistema de caché, lo que reducirá en gran medida las famosas áreas en blanco que tardan en cargar.

- Los mapas son ahora 3D, es decir, se podrá mover nuestro punto de vista de forma que lo se verá en perspectiva.

2.1.9 Fórmula Haversine

Es una ecuación que permite obtener la distancia entre dos puntos geográficos donde interviene la latitud y longitud. La función se expresa en términos de funciones Haversine o múltiplos de esta y se define así:

$$haversine\left(\frac{D}{R}\right) = haversine(\phi_1 - \phi_2) + cos(\phi_1).cos(\phi_2).haversine\ (\Delta\lambda)$$

Nota: Formula para el cálculo de puntos geográficos de una circunferencia.

Dónde:

$D = distancia\ entre\ los\ 2\ puntos$
$R = radio\ de\ la\ esfera$
$\phi = latitud$
$\lambda = longitud$
$\Delta\phi = diferencia\ de\ latitud$
$\Delta\lambda = diferencia\ de\ longitud$

En Wikipedia (2014), se menciona que al despejar la distancia de la fórmula de haversine se obtiene una nueva ecuación, que presenta algunos problemas para ciertos hemisferios, para solucionar estos se propone la fórmula directa de la distancia de haversine:

$$\Delta\phi = \phi_1 - \phi_2$$
$$\Delta\lambda = \lambda_2 - \lambda_1$$
$$A = sin^2\left(\frac{\Delta\phi}{2}\right).cos(\phi_1).cos(\phi_2).sin^2\left(\frac{\Delta\lambda}{2}\right)$$
$$C = 2.atan2(\sqrt{A}, \sqrt{1-A})$$
$$D = R.C$$

2.1.10 GPS (Sistema de Posicionamiento Global)

(Global Positioning System - Sistema de Posicionamiento Global). Se trata de un sistema global de navegación por satélite (GNSS) que permite localizar con precisión un dispositivo GPS en cualquier lugar del mundo.

El sistema es desarrollado y actualmente operado por el Departamento de Defensa de EE.UU. Está compuesto por veinticuatro satélites (21 operativos y 3 de respaldo) que están en órbita a unos 20.200 km de la Tierra con trayectorias sincronizadas para cubrir toda la superficie terrestre.

Para ubicar un punto se utilizan como mínimo cuatro satélites. El dispositivo GPS recibe las señales y las horas de cada uno de ellos. Con estos datos y por triangulación calcula la posición en el mundo donde se encuentra.

Los aparatos receptores, a su vez, tienen la función de enviar una señal para los satélites. Así, haciendo algunos cálculos, los cuales puedes visualizar más abajo, el receptor GPS consigue determinar cuál es tu posición y, con ayuda de algunos mapas de ciudades, señalar varios recorridos posibles para llegar al destino deseado. Para que un GPS funcione adecuadamente es necesario el uso de tres componentes: espacial, control y utilizador. El espacial es compuesto de 27 satélites dispuestos en órbita. 24 de ellos están activos y 3 son de reserva, que entran en operación en caso de ocurrir un fallo en uno de los satélites principales.

La colocación de esos satélites en órbita garantiza que siempre haya por lo menos cuatro de ellos disponibles en cualquier lugar del planeta. Así, siempre que una persona que viva en Japón utilice un GPS, con seguridad obtendrá la información sin ningún problema. El segundo componente, de control, se encarga de la configuración de los satélites. Son cinco estaciones distribuidas alrededor del globo terrestre. Su función principal es actualizar la posición actual de los satélites y sincronizar el reloj atómico presente en cada uno de los satélites.

El último componente, pero no menos importante, es el receptor GPS, y este es el único que se debe adquirir a fin de utilizar esta imprescindible tecnología. Un receptor GPS es

un aparato que refleja su posición, hora y otros recursos variables según el aparato y su marca comercial. El funcionamiento del sistema GPS implica algunos cálculos complejos, pero apenas uno de ellos resulta esencial para este artículo. Se trata del cálculo realizado por el receptor a fin de calcular la localización geográfica.

Funcionamiento de un GPS

FIGURA # 11. Funcionamiento de GPS

Nota: Grafico que muestra el funcionamiento de un gps. (Escuelapedia, Agosto 2013)

Los satélites, al igual que los receptores GPS, tienen un reloj interno, el cual marca la hora con una precisión de nanosegundos. Cuando la señal es emitida, también es enviado el horario de salida del satélite. Esa señal es una transmisión de radio que viaja a la velocidad de la luz (300.000 kilómetros por segundo, en el vacío).

2.1.11 Arquitectura Cliente Servidor en 3 niveles

La programación por capas es una arquitectura cliente-servidor en el que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario.

La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido sin

tener que revisar entre código mezclado. Un buen ejemplo de este método de programación sería el modelo de interconexión de sistemas abiertos.

Además, permite distribuir el trabajo de creación de una aplicación por niveles; de este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles, de forma que basta con conocer la API que existe entre niveles.

En el diseño de sistemas informáticos actual se suelen usar las arquitecturas multinivel o Programación por capas. En dichas arquitecturas a cada nivel se le confía una misión simple, lo que permite el diseño de arquitecturas escalables (que pueden ampliarse con facilidad en caso de que las necesidades aumenten).

En Oasis,Cisc (2014), se dice que el diseño más utilizado actualmente es el diseño en tres niveles (o en tres capas).

1.- Capa de Presentación

2.- Capa de Negocio

3.- Capa de Datos

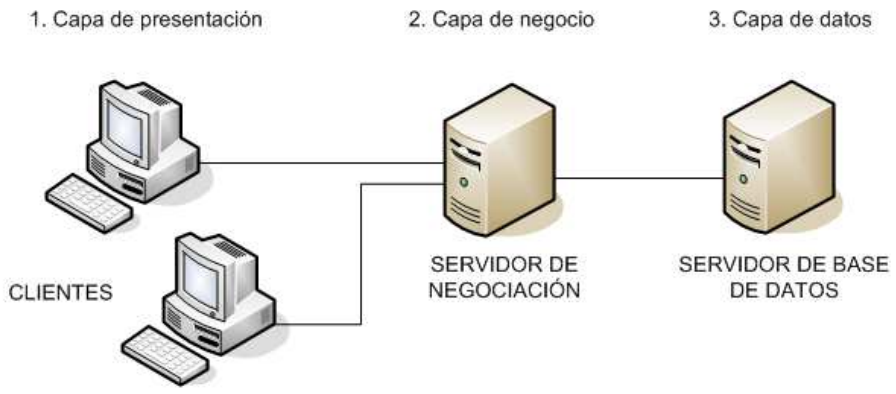


FIGURA # 12. Arquitectura Cliente Servidor en 3 Capas

Nota: Grafico que muestra las tres capas de la arquitectura cliente-servidor. (Oasis,Cisc, 2014)

Capas y niveles

Capa de presentación: es la que ve el usuario (también se la denomina "capa de usuario"), presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). También es conocida como interfaz gráfica y debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario. Esta capa se comunica únicamente con la capa de negocio.

Capa de negocio: es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él. También se consideran aquí los programas de aplicación.

Capa de datos: es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

Todas estas capas pueden residir en un único computador, si bien lo más usual es que haya una multitud de computadoras en donde reside la capa de presentación (son los clientes de la arquitectura cliente/servidor). Las capas de negocio y de datos pueden residir en el mismo computador, y si el crecimiento de las necesidades lo aconseja se pueden separar en dos o más computadoras. Así, si el tamaño o complejidad de la base de datos aumenta, se puede separar en varias computadoras los cuales recibirán las peticiones del computador en que resida la capa de negocio. Si, por el contrario, fuese la complejidad en la capa de negocio lo que obligase a la separación, esta capa de negocio podría residir en uno o más computadores que realizarían solicitudes a una única base de datos. En sistemas muy complejos se llega a tener una serie de computadores sobre los

cuales corre la capa de negocio, y otra serie de computadores sobre los cuales corre la base de datos.

Oasis,Cisc (2014)

Diferencia entre capas y niveles

En una arquitectura de tres niveles, los términos "capas" y "niveles" no significan lo mismo ni son similares.

El término "capa" hace referencia a la forma como una solución es segmentada desde el punto de vista lógico

Por ejemplo:

1. Presentación.
2. Lógica de Negocio.
3. Datos.

Por ejemplo:

Una solución de tres capas (presentación, lógica del negocio, datos) que residen en una solo computadora (Presentación + lógica + datos). Se dice que la arquitectura de la solución es de tres capas y un nivel.

Una solución de tres capas (presentación, lógica del negocio, datos) que residen en dos computadores (presentación + lógica por un lado; lógica + datos por el otro lado). Se dice que la arquitectura de la solución es de tres capas y dos niveles.

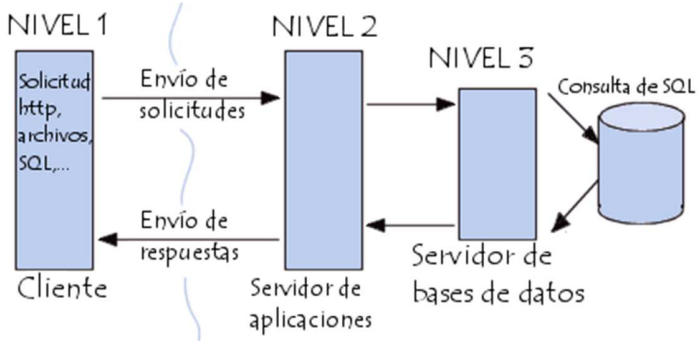


FIGURA # 13. Arquitectura en 3 Niveles

Nota: Esquema que muestra las diferencias de los niveles en la arquitectura de tres niveles. (Iutll, 2012)

2.1.12 Framework “CodeIgniter”

CodeIgniter de Ellislab es uno de esos frameworks minimalistas que ofrece sólo un andamiaje a partir del cual construir tu aplicación MVC. Su núcleo provee un controlador de enrutado básico, carga de modelos e interfaz con la base de datos y una carpeta conveniente de "vistas" para guardar todas las vistas de la aplicación (en plantillas de PHP).

Cuando descomprimes su código de distribución por primera vez, se encuentra la siguiente estructura de árbol para nuestra aplicación (dentro de la carpeta application).

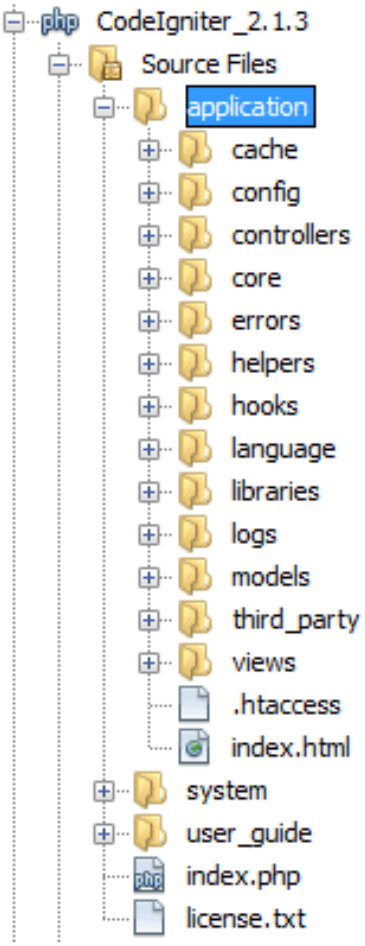


FIGURA # 14. Estructura de árbol de Framework CodeIgniter

Nota: Figura que muestra las carpetas incluidas en el FrameWork CodeIgniter. (Wonderbits 2010).

Sus tres carpetas principales son:

Carpeta de Controladores, Modelos y Vistas.

Se Puede ver que además de las tres carpetas principales hay varias carpetas más, como cache(CodeIgniter tiene cacheo de páginas completo e integración con APC), config(configuración personalizada de la aplicación), core(para sobreescribir funciones del nucleo, auxiliares, crear un controlador padre personalizado para la apliación), errors(mesajes de error personalizados), hooks(enganches del nucleo), language(traducciones locales de textos), librerías, logs y módulos o librerías de terceros(third_party).

El Controlador

En la carpeta de controladores, CodeIgniter viene con un controlador de ejemplo llamado welcome.php que sigue el principio de que un ejemplo es la mejor forma de aprender al dejarte jugar directamente y ver la lógica inherente. Para las rutas, CodeIgniter procesa el primer directorio en el URL, el segundo sería la "acción" que se enlaza con la función de igual nombre en el controlador llamado y finalmente sus parámetros.

Así que una ruta como http://localhost/welcome/helloworld/dotheharlemshake llamaría al controlador "welcome", luego a la función helloworld() dentro con el primer parámetro teniendo el texto "dotheharlemshake". Es una forma bastante directa de operar que recuerda a la de Ruby on Rails o cualquier otro framework MVC no saturado con carga de multiples módulos como la mayoría de CMSs.

Pero cuando se construye una aplicación en CodeIgniter y se pone su URL directamente, se da cuenta que el controlador de bienvenida es llamado y que una función por defecto es invocada. Esto es porque se puede definir en application/config/routes.php el controlador por defecto cuando una URL no da uno y por defecto CodeIgniter llama a la función index() en él.

Se va a analizar el código del controlador welcome.php:

```php
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

class Welcome extends CI_Controller {

	/**
	 * Index Page for this controller.
	 *
	 * Maps to the following URL
	 *		http://example.com/index.php/welcome
	 *	- or -
	 *		http://example.com/index.php/welcome/index
	 *	- or -
	 * Since this controller is set as the default controller in
	 * config/routes.php, it's displayed at http://example.com/
	 *
	 * So any other public methods not prefixed with an underscore wil
	 * map to /index.php/welcome/<method_name>
	 * @see http://codeigniter.com/user_guide/general/urls.html
	 */
	public function index()
	{
		$this->load->view('welcome_message');
	}
}

/* End of file welcome.php */
/* Location: ./application/controllers/welcome.php */
```

FIGURA # 15. Código de Controlador welcome.php de CodeIgniter

Nota: Código que muestra la estructura de la clase welcome.php del framework codeIgniter. (Wonderbits, 2010).

Como la documentación en-linea comenta, cuando se llama a un controlador pero no se especifica la acción, la función index() es llamada, así que se tiene al menos dos formas de acceder a ella, implícitamente */welcome* o explícitamente añadiendo */index* por lo que la ruta será */welcome/index*. Se puede dar cuenta también de que CodeIgniter permite la definición de funciones públicas no mapeadas para poder ser llamadas desde una función de auxiliar (helper) por ejemplo pero no accesible por el usuario.

Se ha llegado desde la ruta al controlador, y ahora se va directo a la vista con la llamada a la función $this->load->view('welcome_message');. Esa llamada carga /application/views/welcome_message.php comose puede adivinar. Pero porqué $this, que representa la instancia del controlador, tiene la clase de carga (loader)? Se puede dar cuenta que la clase del controlador Welcome está extendiendo/heredando de CI_Controller, la clase base Controller de CodeIgniter que provee al controlador con toda la funcionalidad del nucleo como las clases Input, Loader, Router y varias más. Se verá un poco más en la vista después, pero primero se va a ver qué opciones se tiene para pasarles parámetros a las funciones de los controladores.

Cuando se pasa parámetros a una función de un controlador en CondeIgniter, ya se ha visto que se puede usar la ruta, pasando cada parámetros separado por una barra "/", pero qué pasa con los formularios o las llamadas ajax con muchos parámetros que no dependan del orden? Se puede usar $_GET o $_POST ya que CodeIgniter deja libertad de usar estas variables superglobales de PHP pero recomienda usar las funciones filtradas $this->input->get('nombredevariable'); and $this->input->post('nombredevariable'); que provee, previniendo los ataques desde otros sitios web(en inglés Cross Site Scripting o XSS abreviado) y la falsificación de datos.

Varias opciones de configuración pueden habilitarse en */application/config/config.php*, como la relacionada con XSS $config['global_xss_filtering'] y la falsificación de peticiones a través de sitios web o en inglés Cross Site Request Forgery o CSRF $config['csrf_protection'] que previene que los formularios sean enviados desde otros sitios u orígenes diferentes a los generados por CodeIgniter.

El Modelo y ActiveRecord

CodeIgniter tiene lo que llama drivers de base de datos y cache en su nomenclatura, que básicamente abstrae las acciones comunes para varias formas internas de funcionamiento o back-ends. Eso permite usar sin problemas MySQL, ODBC, SQLite, SQLServer, PDO, Postgre or Cubrid directamente a través de su patrón de acceso ActiveRecord modificado.

El patrón ActiveRecord también se usa en Ruby on Rails y es básicamente un constructor de consultas a bases de datos relacionales que permite acceder, insertar y actualizar información con mínimo código según se expone en la documentación de CodeIgniter.

Explicar el patrón de ActiveRecord va más allá del ámbito de este artículo, se recomienda leer la documentación de CodeIgniter sobre ello.

Se tiene en cuenta de todas formas que se ha de configurar qué driver de base de datos ha de ser cargado y que cuando se defina un modelo heredando de CI_Model ese driver será

cargado y las funciones públicas del controlador serán accesibles en la instancia de clase $this.

Se verá un ejemplo sencillo de un modelo así:

FIGURA # 16. Código de Modelo de CodeIgniter

```php
<?php
if ( ! defined('BASEPATH')) exit('No direct script access allowed');

class Themodel_model extends CI_Model {

  function __construct() {
    parent::__construct();
  }

  public function getLirics(){
    $query = $this->db->get_where('songs', array('name' => 'The model'));
    return $query->row();
  }

  public function increaseViewCount(){
    $this->db->where('name', 'The model');
    $this->db->where('artist', 'Kraftwerk');
    $this->db->set('viewcount', 'viewcount+1', FALSE); //el útlimo parámetro evita que 'viewcount+1' sea transformado
    $query = $this->db->update('songs');
    return $query ? TRUE : FALSE;
  }
}
```

Nota: Código que muestra la estructura de la clase modelo del framework codeIgniter. (Wonderbits, 2010).

Una vez definido un modelo como este, sólo se tiene que llamar al cargador en la instancia del controlador `$this->load->model('themodel);` y después acceder a sus funciones así `$this->Themodel->getLirics();`.

Se puede configurar qué driver cargar en */application/config/database.php*, su configuración de acceso, la cache de consultas de CI, codificación de carácteres, conexiones persistentes, auto-inicialización e incluso si cargar o no ActiveRecord para hacer más escuálido el uso de memoria y proceso de CodeIgniter.

La Vista

Las vistas en CodeIgniter son sencillos archivos de HTML/XML/texto conteniendo PHP incrustado pero con un ámbito especial donde las variables pasadas como segundo parámetro a load->view() serán locales.

El controlador welcome.php está cargado y se va a cargar ahí la vista welcome_message.php. Es una página HTML en sí misa. En pos de la concisión se va a

acortar el welcome_message.php original de CodeIgniter quitándole los estilos incrustados y se asume que en lugar de cargarlo con $this->load->view('welcome_message'); se está llamándolo pasándole algunas variables como lo siguiente $this->load->view('welcome_message', array('user_name' => 'Major Tom')); y entonces el welcome_message.php, sería así:

FIGURA # 17. Código de Vistas de CodeIgniter

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Bienvenido <?= $user_name ?> a CodeIgniter</title>
  </head>
  <body>

    <div id="container">
      <h1>Bienvenido <?= $user_name ?> a CodeIgniter!</h1>

      <div id="body">
        <p>The page you are looking at is being generated dynamically by CodeIgniter.</p>

        <p>If you would like to edit this page you'll find it located at:</p>
        <code>application/views/welcome_message.php</code>

        <p>The corresponding controller for this page is found at:</p>
        <code>application/controllers/welcome.php</code>

        <p>If you are exploring CodeIgniter for the very first time, you should start by reading the <a href="user_guide/">User Guide</a>.</p>
      </div>

      <p class="footer">Page rendered in <strong>{elapsed_time}</strong> seconds</p>
    </div>

  </body>
</html>
```

Nota: Código que muestra la estructura de la clase vistas del framework codeIgniter. (Wonderbits,2010) .

En el ejemplo se está referenciado la clave del array que se le pasa como segundo parámetro a view() en la forma de una variable local. `<h1>Bienvenido <?= $user_name ?> a CodeIgniter!</h1>` mostrará `<h1>Bienvenido Major Tom a CodeIgniter!</h1>`. Por supuesto también se puede iterar en la forma compacta de PHP para plantillas usando `foreach($array as $clave => $valor) :` y `endforeach;`, condicionales `if () : elseif() : else : endif;` etc.

Enrutado

El enrutado puede ser una tarea muy directa como ya se ha comentado de la forma */controller/function/parameter1/parameter2/..* o se puede personalizarla en la forma que mejor se adapte a los planes. Se puede definir alias, reglas o expresiones regulares, lo que mejor sirva a los propósitos en el archivo de configuración */application/config/routes.php*.

Como también se ha comentado, en dicho archivo se puede definir el controlador por defecto, que será usado en caso de que CodeIgniter no encuentre un controlador con el nombre del primer fragmento de la ruta. En caso de que el sistema de enrutado de CodeIgniter por defecto no encaje con las necesidades de ninguna forma, se puede definir un controlador o función de enrutado asignándolo en index.php.

CodeIgniter provee de un par de palabras clave para usar en las definiciones de enrutado, :num y :any que pueden ser usado para coincidir con un valor numérico o cualquier valor respectivamente. Así se dice que se quiere mapear /search/freebies+para+geeks al controlador Search_engine y pasarle el texto después de "sear" a la función search del controlador como primer parámetro. La ruta sería como la siguiente:

```
1. $route['search/(:any)'] = 'search_engine/search/$1';
```

Se ha rodeado a :any con paréntesis, eso la hace ser un grupo de expresión regular y se puede referenciarla en la segunda parte de la regla, siendo el primer grupo será $1. También en la segunda parte de la regla, se usa el sistema de nomenclatura de enrutado original de CodeIgniter especificando primero el controlador, luego la función en él a la que se llama y finalmente el parámetro de texto referido como $1.

CodeIgniter provee de lo básico para cualquier proyecto web y hay clases y helpers auxiliares que no se ha cubierto como el helper de formularios, la clase de cache con muchos controladores para distintas caches, están también los auxiliares de ayuda de URL, de descarga. captcha, email, y varios más. Tampoco se ha cubierto la extensión del núcleo y los auxiliares con los personalizados.

2.2 Marco conceptual

ACTIVITY: Estas acciones se pueden comunicar con otras a fin de hacer determinadas tareas u obtener diferentes resultados. Una aplicación de Android es realmente un conjunto de pequeñas activity enlazadas.

VIEWS: Son las clases básicas del interfaz de usuario que controlan el layout de la pantalla y la funcionalidad otorgada al usuario, por ejemplo botones, cajas de texto, etc.

SERVICES: Son componentes que da la posibilidad de realizar acciones duraderas en un segundo plano y que no ofrecen en sí mismas una interfaz de usuario.

STARTED: Son iniciados por algún componente y su ejecución es independiente del componente que lo ha iniciado, pudiendo continuar su ejecución aunque el componente que lo ha iniciado ya no exista.

BOUND: Son iniciados cuando algún componente se "ata" a este servicio, y proporcionará una suerte de cliente-servidor que permite que los componentes se comuniquen con el service, etc.

CONTENT PROVIDER: Proporciona una interfaz de acceso a datos. Conecta datos con código y además Android ofrece una base de datos SQLite que permite obtener persistencia con la estructura de datos del contentprovider. Android ofrece unos contentprovider por defecto que permitirá usar lista de contactos, videos, imágenes, etc…

BROADCAST RECEIVER: Receptor de mensajes del sistema e intents.

WIDGET: Son pequeños componentes que normalmente se muestran al usuario en pantalla y muestran cierta información, permitiendo al usuario realizar ciertas acciones.

GPS: Global PositioningSystem o Sistema de Posicionamiento Global, es el nombre de un sistema de posicionamiento que actúa en todo el mundo y permite saber en qué lugar se está en cada momento. En el ámbito móvil es muy útil para utilizarlo en conjunto con diferentes aplicaciones.

IPL: InitialProgramLoader o Cargador del Programa Inicial, es el encargado de cargar al SPL en RAM e iniciarlo.

JIT: en Android los programas son cargados con una máquina virtual llamada Dalvik, y jit es la versión mejorada de ésta que apareció a partir de la versión 2.2 de este SO.

MMS: Multimedia MessagingSystem o Sistema de Mensajería Multimedia, es un estándar de mensajería que permite enviar y recibir todo tipo de contenidos multimedia en forma de mensajes.

NFC: Near-FieldComunication o Comunicación de Campo Cercano, es un estándar creado para la comunicación sin cables de corto alcance para la realización de pagos a través de nuestros dispositivos móviles mayoritariamente.

OTA: OverThe Air o Sobre El Aire, es la acción de descargar datos a través del aire.

QR: es un código, como el de barras, pero creado en base a píxeles que puede ser leído fácilmente por nuestros dispositivos móviles (a través de las cámaras) y que sirven para abrir enlaces con diferentes funciones.

ROM: ReadOnlyMemory o Memoria de Sólo Lectura, en realidad es una memoria física, pero en el mundo Android se le conoce como el software que se instala en su interior, que es el sistema operativo. Ésta puede ser oficial, que son lanzadas por Google o por las compañías fabricantes de nuestros teléfonos o cocinadas (o custom), que son

aquellas que son realizadas por programadores independientes normalmente sin ánimo de lucro, las ROMs cocinadas suelen dar características y opciones mejoradas con respecto a las oficiales.

SDK: Software Development Kit o Kit de Desarrollo de Software, es un conjunto de herramientas de desarrollo que permite al programador crear aplicaciones para un sistema, en este caso Android.

SPL: SecondProgramLoader o Programa Cargador Secundario, es el encargado de cargar el sistema operativo.

GPRS: General Packet Radio Service o Servicio General de Paquetes por Radio, es una de las formas de transmitir datos que está disponible en nuestros dispositivos, normalmente se encuentra mostrada como G. Sus velocidades están comprendidas entre los 56Kbps y los 114Kbps.

GSM: Groupe Special Mobile o Sistema Global para las Comunicaciones Móviles, es un sistema que está perfectamente definido para la comunicación entre dispositivos móviles de manera digital. Es comúnmente llamado 2G.

3G o UMTS/WCDMA: (Universal Mobile Telecommunications System), es la tecnología que hace que se pueda usar redes de datos junto con nuestra línea de voz, se denomina así porque representa la tercera generación de comunicaciones móviles.

3.5G o HSDPA: (High Speedy Down link Packet Access), es la evolución inmediata del estándar UMTS, mejora la velocidad de descarga hasta los 14Mbps. Normalmente todos los dispositivos de gama alta son compatibles con ella, no así las infraestructuras de que se disponen.

4G: Exactamente igual que la anterior, pero representando la cuarta generación de esta tecnología. En este caso las velocidades de trabajo van de los 100Mbps a 1Gbps.

ADB y ADB: Android Depuration Bridge o Puente de Depuración de Android. Es la herramienta utilizada para enviar comandos desde un ordenador a un dispositivo con este sistema.

SHELL: El Shell es un programa terminal que hace de traductor de nuestras órdenes, actúa sobre una línea de comandos.

AOSP: Android Open Source Project o Proyecto de Código Abierto Android. Las propias siglas ya lo indican, Android es un sistema de código abierto, lo que quiere decir que cualquiera puede descargar su código fuente y modificarlo a su gusto. Este detalle precisamente es el que hace que Android sea un grande entre los grandes, consiguiendo diferenciarse así de su competidor más directo, idos de Apple, que se trata de un sistema cerrado.

APK: Android Package Archive o Paquete de Archivo de Android, es la extensión de los paquetes de los programas que operan en este sistema. Pueden ser abiertos y modificados fácilmente.

APN: Access Point Ñame o Nombre del Punto de Acceso, son los datos para poder acceder a Internet desde el móvil que tiene cada operadora de forma única. En la mayoría de los casos estos datos vienen ya configurados.

APP: Abreviatura de la palabra inglesa aplicación, que quiere decir aplicación.

ARM: Avance RISC Machines o Máquinas RISC Avanzadas, es una familia de microprocesadores RISC diseñados por la empresa Acom Computers. Son microprocesadores muy usados en los dispositivos móviles gracias a su gran característica, su bajo consumo.

FC: Force Close o Cierre Forzoso, es el error más común en las aplicaciones Android e indica que debe cerrarse debido a un mal funcionamiento de la misma.

CREATE: Utilizado para crear nuevas tablas, procedimientos almacenamientos e índices.

DROP: Empleado para eliminar tablas, procedimientos almacenados e índices.

ALTER: Utilizado para modificar las tablas agregando campos o cambiando la definición de los campos.

SELECT: Utilizado para consultar registros de la base de datos que satisfagan un criterio determinado.

INSERT: Utilizado para cargar lotes de datos en la base de datos, una única operación.

CAPÍTULO 3
ANÁLISIS DEL SISTEMA

La mayoría de los habitantes de la ciudad de Guayaquil desconocen las líneas de nuestro sistema de transporte urbano por lo que se pierden al buscar un lugar, por lo que llegan a una única opción de hacer uso de un transporte particular llamado Taxi, por el cual pagarán un monto más elevado en comparación con el transporte urbano.

Al encontrar este problema existe la necesidad de diseñar una aplicación amigable diseñada en java e instalada en un equipo móvil con sistema android que muestre las rutas de distintas líneas urbanas que les ayude a los habitantes de determinado sector llegar al lugar deseado, que pasen por un punto de ubicación y llegar al lugar destino ingresándolo previamente como un parámetro de entrada y permita elegir la línea más adecuada que llevará al destino ahorrando dinero.

Para ello se necesita recolectar información de líneas, recorridos y requerimientos necesarios para un análisis previo al desarrollo de la aplicación.

3.1 Recolección de Datos

3.1.1 Consulta a Director de Cooperativas

El proceso de investigación se recopilará información acerca de las cooperativas de transportes urbanos que circulan por la zona de referencia sur-noreste, consultando a directores de dichas cooperativas información sobre recorridos, turnos y demás información útil para el proceso de información.

3.1.2 Consulta a Choferes de líneas

Se realizará preguntas puntuales a los choferes de las líneas 63 Orquídeas, línea 12 OCP (Florida) y línea 83 (Bastión Popular) sobre el recorrido normal que cumple cada unidad.

3.1.3 Consulta a Usuario Final

Se realizará preguntas a usuarios de los transportes públicos que hicieron uso de las líneas indicadas para movilizarse a distintos lugares de nuestra zona de referencia, los cuales manifestaron que siempre utilizan una de estas tres líneas para dirigirse a su trabajo, unidad educativa, etc, ya que estas líneas tienen un recorrido muy similar y tienen muchos puntos de intersección a lo largo del recorrido.

3.2 Tratamiento de la Información.

Los datos van hacer ingresados de manera manual a la base de datos en Mysql por medio de un pequeño sistema web desarrollado en PHP que servirá para alimentar la base de datos que usará la aplicación.

La información ingresada pertenecerá a recorridos de las líneas urbanas, cooperativas, lugares de referencia, etc.

La información será presentada por medio del aplicativo web de manera ordenada y categorizada.

3.3 Ciclo de Vida del Proyecto

Análisis.- En esta fase se analizará toda la información recolectada para poder identificar los requerimientos necesarios para el desarrollo de esta aplicación. También se debe realizar un análisis de factibilidad del desarrollo de la aplicación.

Diseño.- En esta fase de desarrollará las diferentes interfaces que tendrá la aplicación en Android así como también las interfaces que tendrá el sitio web.

Construcción.- Esta fase comprende el desarrollo de la base de datos en mysql, desarrollo de código en Android y PHP.

Instalación, pruebas y corrección de errores.- Esta es última fase del ciclo, en la cual se instalará la aplicación en un dispositivo con sistema Android para proceder a evaluar la aplicación y si se detecta errores o modificaciones poder aplicarlas.

3.4 Cronograma del Proyecto

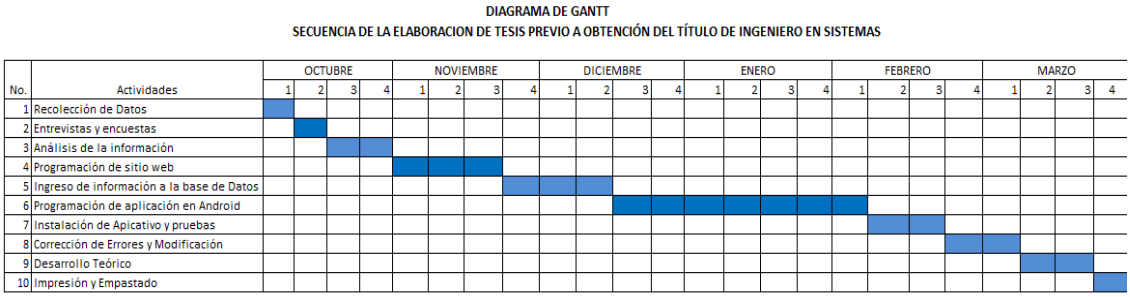


FIGURA # 18. Diagrama de Gantt – Cronograma

Nota: Diagrama que muestra la secuencia del cronograma del proyecto. Elaborado por los Autores. (Marzo 2015)

3.5 Análisis de Factibilidad

3.5.1 Factibilidad Técnica

Para el desarrollo de esta aplicación se necesitó cumplir requerimientos mínimos en hardware y software que se utiliza a lo largo de la ejecución del proyecto y que se detalla a continuación:

Hardware:

1 Computadora Core i5 – 4 GB ram – 500 gb HD

1 Tablet Dual Core -1 GB ram

Software:

Base de Datos:

Mysql server

Programación:

Lenguaje PHP

Kit de Desarrollo de Software de Android (SDK)

Google maps Android API V2

Servicio de Internet:

Internet de 2MB para hogar

Plan de Datos (Internet) 300 MB

Requerimientos de Hardware y Software en Equipo Móvil

Los requisitos necesarios para la aplicación son los siguientes:

- Tener Sistema Operativo Android desde la versión 3.1 en adelante
- Soporte a GPS
- Poseer paquetes de datos o acceso a Internet.

La aplicación funciona en dispositivos que posean un mínimo de 512Mb de memoria ram.

3.5.2 Factibilidad Operacional

La aplicación instalada en un equipo con sistema operativo Android funcionará correctamente siempre y cuando cumpla con los requerimientos de hardware y software especificados en este documento, además de servicios como Internet.

3.5.3 Factibilidad Económica

Para el desarrollo de esta aplicación se realizó un análisis de costo del proyecto, por lo que se optó por trabajar con software libre para no generar gastos en pago de licencias. Los demás requerimientos de materiales y servicios necesarios para la elaboración de este proyecto de tesis se detallan a continuación:

Análisis de Costo del Proyecto

RUBRO	VALOR
• TELÉFONO CON SISTEMA ANDROID	$220.00
• PAPELERIA	$ 20.00
• REPRODUCCIÓN DEL MATERIAL (FOTOCOPIAS)	$ 40.00
• IMPRESIÓN.	$ 170.00
• INTERNET	$ 30.00
• SERVICIOS BASICOS	$ 90.00
• EMPASTADOS	$90.00
• TRANSPORTE	$30.00
• GASTOS VARIOS	$100.00
• IMPREVISTOS	$ 80.00
MATERIAL	
• LIBROS Y REVISTAS	$ 60.00
TOTAL	$ 840.00

3.5.4 Alcance

La Aplicación mostrará la ubicación del usuario y el recorrido de la línea urbana graficado en un mapa de Google Maps por medio de un trazo de línea donde el usuario podrá visualizar su recorrido de inicio a fin y así poder escoger la línea adecuada para dirigirse a determinado lugar de destino, además tendrá la opción de búsquedas de lugares como hoteles, centro comerciales, hospitales, etc, que servirán de referencia para la localización de calles e intersecciones.

3.5.5 Limitaciones

La aplicación mostrará las líneas y sus recorridos de nuestra área o zona de referencia que será el sector Centro-Noreste de la Ciudad de Guayaquil, de sur a norte; del parque Victoria hasta los sectores Orquídeas y Florida y de Este a Oeste; de la Av. Daule hasta la Autopista Narcisa de Jesús (Orillas del Rio Guayas).

3.6 Resultado Esperado

El resultado que se desea obtener a través de este investigación es la aplicación lista para ser instalada en un dispositivo con sistema Android, que pueda ser descargada gratuitamente de Google Play Store y los usuarios puedan hacer uso de los beneficios para lo cual fue desarrollada.

CAPÍTULO 4
DISEÑO DEL SISTEMA

4.1 Descripción de Arquitectura Usada

4.1.1 Arquitectura del Sistema

La aplicación Móvil es un sistema que usa la Arquitectura Cliente-Servidor y se basa en el modelado de la Arquitectura en tres capas.

1.- Capa de Presentación
2.- Capa de Negocio
3.- Capa de Datos

La Capa de presentación es cada una de las interfaces que la aplicación muestra al usuario final en el dispositivo móvil la cual dará información acerca de rutas, líneas, lugares y demás información que será receptada de la capa de negocio.

La Capa de Negocio es la capa intermediaria, es el vínculo entre la capa de Presentación y la capa de Datos, esta capa consta de métodos, funciones, scripts y demás servicios web, se denomina la capa lógica porque es aquí donde se definen reglas y procedimientos que emplearan para enviar peticiones y recibir resultados.

La Capa de Datos es donde residen los datos y es la encargada de acceder a los mismos. Nuestra base de datos fue creada en MySql y consta de tres tablas que se encargarán de almacenar toda la información utilizada en esta aplicación móvil.

4.1.2 Diagrama de Bloques

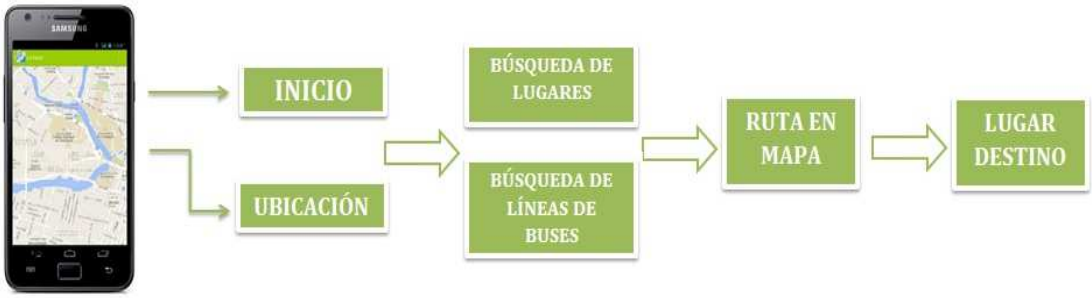


FIGURA # 19. Diagrama de Bloques

Nota: Diagrama que muestra la secuencia de procesos de la aplicación móvil. Elaborado por los Autores. (Marzo 2015)

Este tipo de diagrama de bloques da una mejor visión del funcionamiento interno que posee la aplicación móvil, desde que se ingresa a la aplicación “Mi Ruta”, muestra las 2 opciones que podrá escoger el usuario y el resultado final que es la selección de la línea de bus que lo llevará al lugar destino deseado.

Como se puede apreciar en la figura, el proceso inicia desde que el usuario accede a la aplicación, el cual podrá realizar ciertas funcionalidades donde podrán realizar búsquedas por lugares o búsqueda de líneas de buses, que mostrarán el recorrido del bus mediante el trazo de líneas en una mapa de google en donde el usuario podrá visualizar el lugar destino a donde se dirige.

4.1.3 Fórmula Haversine

Para encontrar las líneas de buses que circulen por el lugar donde se encuentre el usuario a cierta distancia de un radio de Latitud/longitud dada se usa una consulta SELECT a la base de datos usando la Fórmula de Haversine explicada anteriormente.

En esta aplicación la formula funciona solamente con una coordenada y calcula los puntos cercarnos dentro de un radio de 25 km

Consulta en base de datos:

```
SELECT id, ( 3959 * acos( cos( radians(37) ) * cos( radians( lat ) )  * cos( radians( lng ) ) - radians(-122) ) + sin( radians(37) ) * sin(radians(lat)) ) ) AS distance  FROM markers HAVING distance < 25  ORDER BY distance  LIMIT 0 , 20;
```

FIGURA # 20.Fórmula de Haversine en MySql

Nota: Esquema muestra la consulta sql basada en la formula de Haversine. Elaborado por los Autores. (Marzo 2015)

4.1.4 Trazos de rutas en Google Maps

Para poder dibujar los trazos de las rutas de las líneas de buses de nuestra aplicación en el mapa de Google es necesario llevar a cabo un proceso el cual comprende de los siguientes pasos:

1. Armar la dirección de la consulta que se utilizará para obtener los datos necesarios para la aplicación, para esto se usa el formato json para mostrar los datos, los parámetros que se utilizan son los siguientes: a) origen (latitud/longitud), b) destino (latitud/longitud).

2. Los datos que recibimos de la consulta esta en forma de nodo debido al formato utilizado, este formato ayuda a poder guardar en un listado los datos que contiene la consulta para posteriormente utilizarlos en nuestro mapa.

A continuación las líneas de código para la obtención de ruta de un bus:

```
jObject = jParser.getJSONFromUrl("http://landaer.com/app_buses/auth/json_ruta/"+id);
        try {
            Jlist=jObject.getJSONArray("ruta");
            for(int i = 0;i < Jlist.length();i++){
                jObjPoint= Jlist.getJSONObject(i);
                HashMap<String, String> map = new HashMap<String, String>();
                map.put("cord_lat",jObjPoint.getString("cord_lat"));
                map.put("cord_lon",jObjPoint.getString("cord_lon"));
                List.add(map);
            }
        }catch (JSONException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            Log.i("Error parsing: ", e.getMessage());
        }
```

FIGURA # 21.Código para obtener la ruta de un bus

Nota: Líneas de código desarrolladas para la obtención de la ruta de un bus. Elaborado por los Autores. (Marzo 2015)

Luego de que los datos se los tiene en un listado se procede a dibujar en el mapa, la ruta de la línea de bus:

```
ArrayList<LatLng> points = new ArrayList<LatLng>();
    PolylineOptions lineOptions = null;

    for(int i=0;i<result.size();i++){
        lineOptions = new PolylineOptions();

        double lat = Double.parseDouble(result.get(i).get("cord_lat"));
        double lon = Double.parseDouble(result.get(i).get("cord_lon"));

        LatLng position = new LatLng(lat,lon);

        points.add(position);

        lineOptions.addAll(points);
        lineOptions.width(4);
        lineOptions.color(Color.BLUE);
    }
mMap.addPolyline(lineOptions);
```

FIGURA # 22.Código para dibujar la ruta en un mapa

Nota: Líneas de código desarrolladas para la gráfica de la ruta en el mapa de google. Elaborado por los Autores. (Marzo 2015)

4.1.5 Diagrama de Casos de Uso

Aplicación Móvil

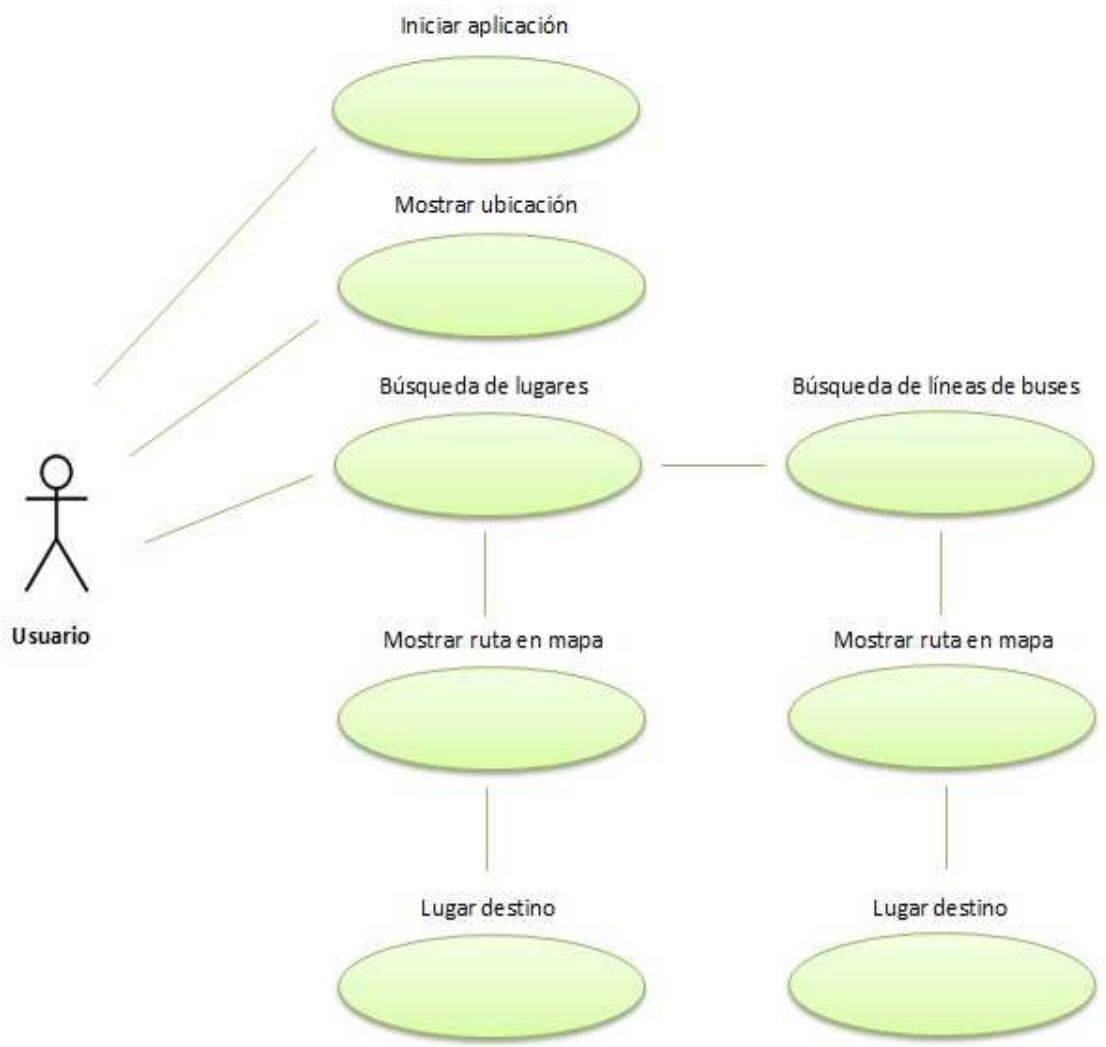


FIGURA # 23. Casos de Uso de Aplicación Móvil

Nota: Diagrama que muestra los diferentes casos de uso de la aplicación móvil. Elaborado por los Autores. (Marzo 2015)

Se ha definido un solo actor:

El Usuario: Quien podrá hacer búsquedas de líneas, lugares, para poder escoger la mejor opción de entre las que le da la aplicación.

Se han definido los siguientes casos de uso de nuestra aplicación:

Iniciar Aplicación: Luego de haber descargado e instalado la aplicación en su dispositivo móvil.

Mostrar Ubicación: Luego de haber iniciado la aplicación la primera pantalla que mostrará la aplicación es un mapa de google con un ícono de referencia marcando el lugar donde se encuentre.

Búsqueda de Lugares: Se podrá buscar el lugar que quiera y lo mostrará en un mapa trazando la ruta de la línea que pase por dicho lugar, estos lugares son referencia si no se sabe el nombre de las calles donde se quiere transportar, y estarán almacenados en la base de datos.

Búsqueda de Líneas de Bus: Se podrá buscar Líneas de buses, para ello mostrará una lista de todas las líneas ingresadas a la aplicación.

Mostrar Ruta en Mapa: La aplicación mostrará todo el recorrido de cada una de las líneas que se escoja y la graficará con un trazo de línea sobre el mapa de google.

Lugar Destino: El usuario podrá transportarse hacia su lugar destino después de haber seguido el recorrido graficado en un mapa de google a través de la aplicación.

4.1.6 Diagrama de Casos de Uso

Sistema Web

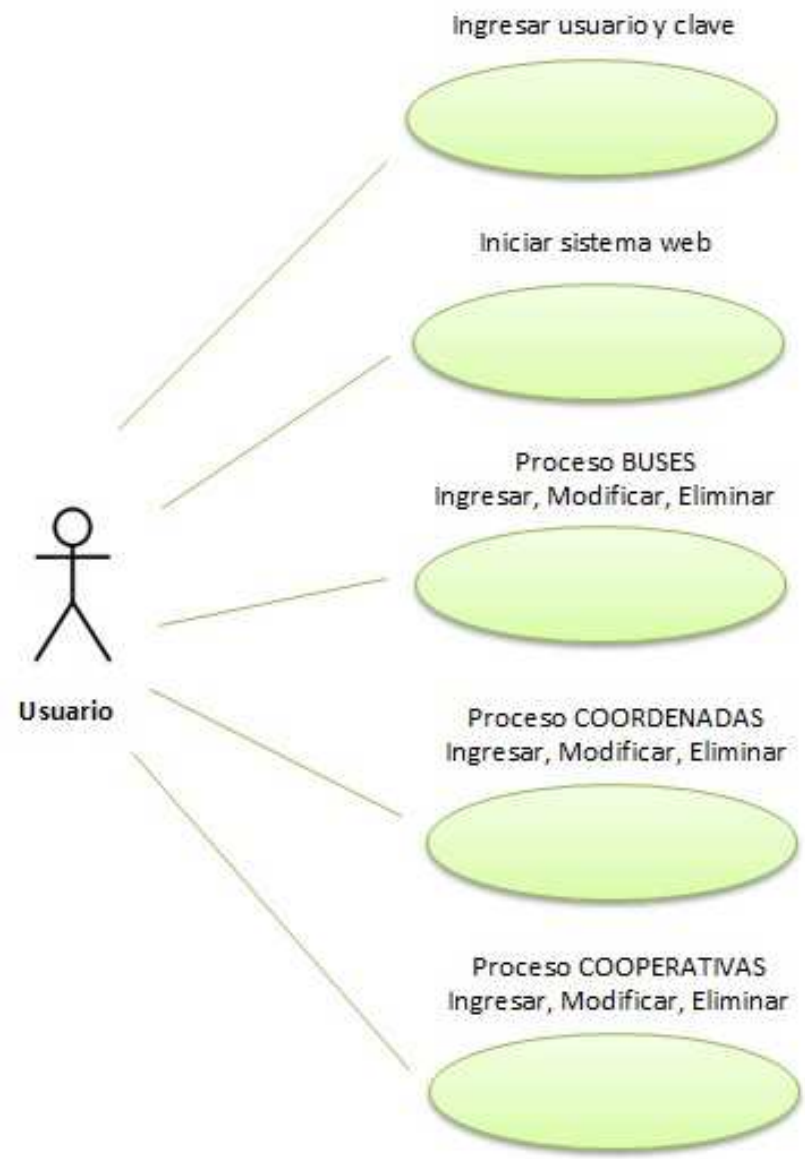


FIGURA # 24. Casos de Uso de Sistema Web

Nota: Diagrama que muestra los diferentes casos de uso del sistema web. Elaborado por los Autores. (Marzo 2015)

Se ha definido un solo actor:

El Usuario (Administrador): Quien podrá ingresar al Sistema y efectuar las operaciones de ingreso, edición y eliminación de la información que alimentará la aplicación Móvil.

Se han definido los siguientes casos de uso de nuestra aplicación:

Acceder al Sistema Web: Luego de haber ingresado su usuario y contraseña podrá acceder al sistema web, en el cual podrá hacer uso de los privilegios de administrador.

Ingresar, Editar y Eliminar Buses: Podrá ver las líneas de buses ingresadas al sistema web, la cual muestra toda la ruta de cada línea y tiene la opción de editar, eliminar e ingresar una nueva línea de bus.

Ingresar, Editar y Eliminar Coordenadas: Podrá ver cada una de las coordenadas ingresadas al sistema web, y tendrá la opción de editar, eliminar e ingresar una nueva coordenada.

Ingresar, Editar y Eliminar Cooperativas: Podrá ver cada una de las cooperativas ingresadas al sistema web, y tendrá la opción de editar, eliminar e ingresar una nueva cooperativa.

4.2 Diseño de Base de Datos

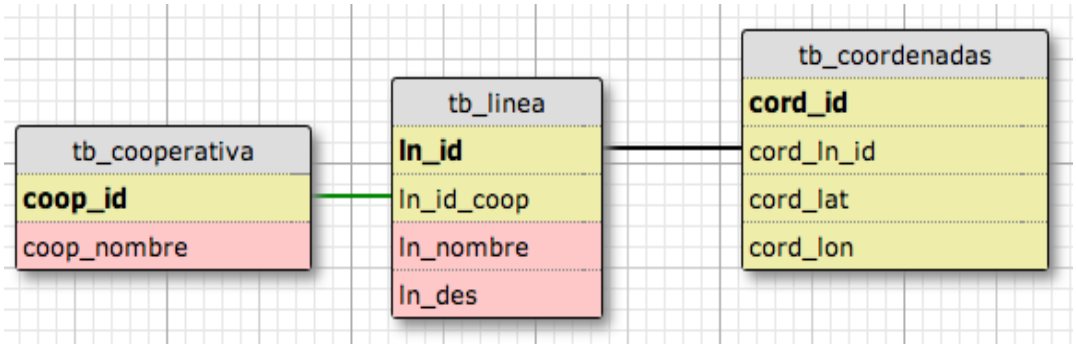


FIGURA # 25. Modelo Entidad-Relación de la Base de Datos

Nota: Esquema que muestra el Modelo Entidad-Relación de la Base de Datos. Elaborado por los Autores. (Marzo 2015)

Se desarrolló una base de Datos en MySql que consta de 3 tablas necesarias para la ejecución de esta aplicación.

Tb_cooperativa.- Tabla que guardará información sobre las cooperativas ingresadas al sistema web y que alimentará a la aplicación.

Tb_linea.- Esta tabla guardará información sobre las diferentes líneas de buses y su recorrido de inicio a fin, esta información será ingresada al sistema web y alimentará a la aplicación.

Tb_coordenadas.- Esta tabla guardará información sobre cada una de las coordenadas de Latitud y Longitud a lo largo del recorrido de cada línea de bus y servirá para trazar la línea del recorrido graficada en un mapa de google.

4.3 Diseño de Aplicación móvil

Pantallas y Menús de Aplicación Móvil – La Ruta

FIGURA # 26. Pantalla Inicio de Aplicación “La Ruta”

Nota: Gráfico que muestra la interfaz de inicio de la aplicación “La ruta”. Elaborado por los Autores. (Marzo 2015)

Cuando abrimos la aplicación “La Ruta” mostrará el mapa de Google Maps, mostrando la zona de referencia para nuestro estudio “Centro-Noreste”, mostrará también nuestra ubicación por medio del GPS que deberá estar incorporado y activado en el equipo móvil l.

En la parte superior derecha se encuentra el ícono del menú Inicio, donde se podrá encontrar las dos opciones disponibles en la aplicación, como son:

- Búsquedas de Lugares
- Búsqueda de Líneas de Buses

Búsqueda de Lugares

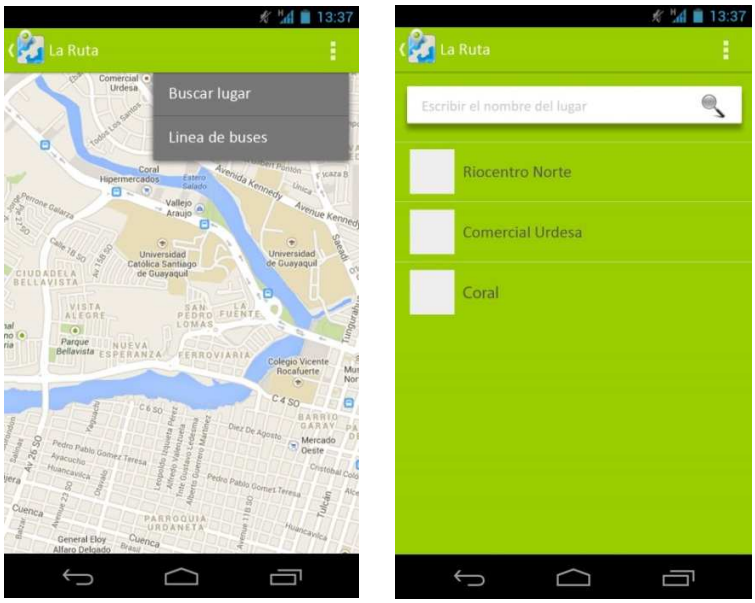


FIGURA # 27. Pantalla Búsqueda de Lugares de Aplicación “La Ruta”

Nota: Gráfico que muestra el proceso búsqueda de lugares de la aplicación “La ruta”. Elaborado por los Autores. (Marzo 2015)

En la opción búsqueda de Lugares se puede escribir en la caja de texto el lugar que se quiera buscar y presionar el ícono de la lupa para poder encontrar lugares conocidos que servirá de referencia para buscar el lugar a donde se quiere transportar, estos lugares están almacenados en la base de datos y son lugares referencias que también sirven como lugar destino en la aplicación.

Búsqueda de Líneas de Buses

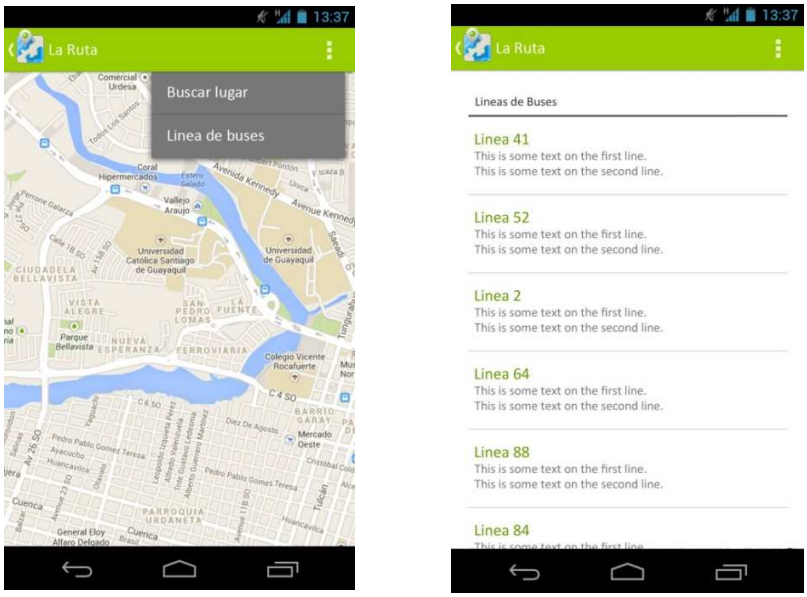


FIGURA # 28. Pantalla Búsqueda de Líneas de Aplicación “La Ruta”

Nota: Gráfico que muestra el proceso de búsqueda de líneas de la aplicación “La ruta”. Elaborado por los Autores. (Marzo 2015)

En la búsqueda de Líneas de buses se desplegará una lista de todas las líneas ingresadas a la aplicación, la cual al dar clic en una línea de bus mostrará todo el recorrido de dicha línea graficado con un trazo sobre el mapa de google.

FIGURA # 29. Recorrido de una Línea - Aplicación “La Ruta”

Nota: Gráfico que muestra el recorrido de una línea trazado en el mapa. Elaborado por los Autores. (Marzo 2015)

4.4 Diseño del Sistema Web

Sistema Web creado para la administración de contenidos y alimentación de información a la aplicación móvil.

Menú Inicio

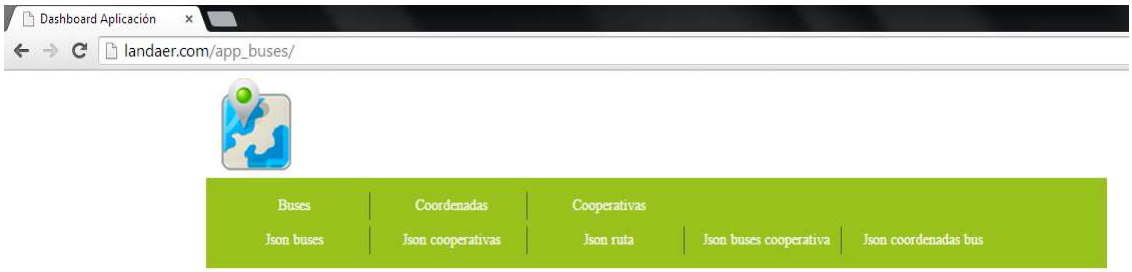


FIGURA # 30. Menú Inicio de Sistema Web

Nota: Interfaz que muestra el menú y las diferentes opciones de la aplicación "La ruta". Elaborado por los Autores. (Marzo 2015)

Este es el sitio web diseñado en Php con MySql, fue creado para facilitar el ingreso, modificación y eliminación de Buses, Coordenadas Geográficas y Cooperativas, las mismas que alimentarán la Aplicación Móvil que se instalará en un equipo con Android.

Buses

FIGURA # 31. Líneas de Buses ingresadas al Sistema Web

Nota: Interfaz que muestra la información de las líneas de la aplicación "La ruta". Elaborado por los Autores. (Marzo 2015)

Muestra las líneas de buses ingresadas al sistema web, la cual muestra toda la ruta de cada línea y tiene la opción de editar, eliminar e ingresar una nueva línea de bus.

Coordenadas

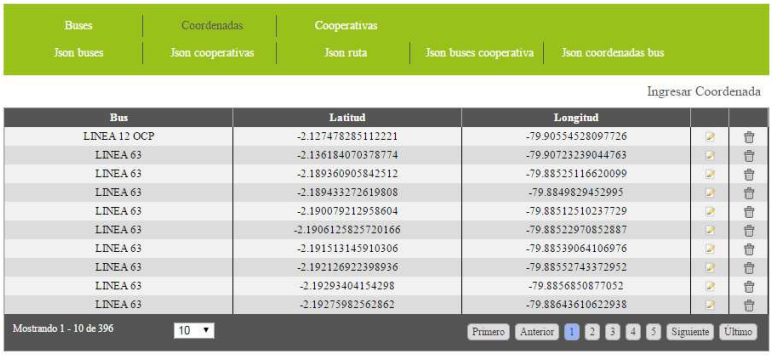


FIGURA # 32. Coordenadas ingresadas al Sistema Web

Nota: Interfaz que muestra la información de las coordenadas de las líneas de la aplicación "La ruta". Elaborado por los Autores. (Marzo 2015)

Muestra cada una de las coordenadas de Latitud y Longitud ingresadas en cada línea de bus, estas coordenadas son puntos de referencia que servirán para el trazo del recorrido de cada línea en el mapa de google. Así mismo tiene las opciones de editar, eliminar e ingresar nuevas coordenadas de Latitud y longitud.

Cooperativas

FIGURA # 33. Cooperativas ingresadas al Sistema Web

Nota: Interfaz que muestra la información de las cooperativas de las líneas de la aplicación "La ruta". Elaborado por los Autores. (Marzo 2015)

Muestra cada una de las cooperativas ingresadas al sistema, tiene las opciones de editar, eliminar e ingresar nueva cooperativa.

CAPÍTULO 5
IMPLEMENTACIÓN Y PRUEBAS

5.1 instalación e Implementación de la Aplicación en el dispositivo móvil.

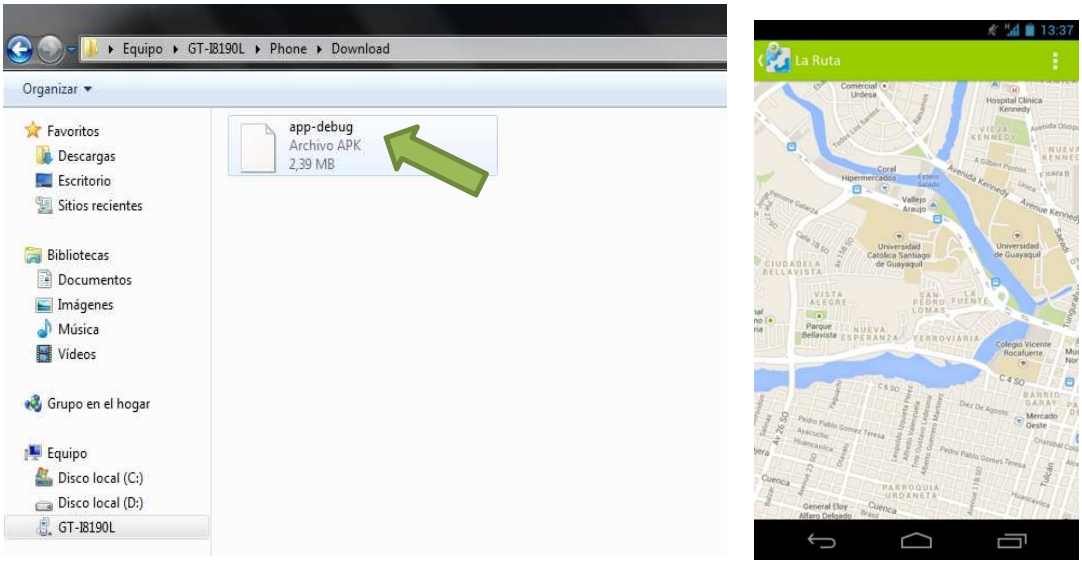


FIGURA # 34. Instalador de Aplicación “La Ruta”

Nota: Muestra el archivo instalador de la aplicación móvil y la pantalla inicio de la misma. Elaborado por los autores. (Enero 2015)

La instalación de la aplicación móvil es sumamente fácil para ello solo es necesario subirla al Play Store de google para que los usuarios puedan descargarla el app.

Para proceder con la implementación se pasó el archivo que se genera al compilar la aplicación (**app-debug.apk)** al dispositivo móvil y luego se lo instaló. Al momento de instalar la aplicación, ésta pedirá los permisos necesarios.

5.2 Instalación e Implementación de Sistema web

La aplicación móvil necesita de información que alimente la capa de base de datos, para ello se diseñó un sistema web bajo lenguaje php que contenga toda la información necesaria y útil para la aplicación.

El sistema web se lo instaló en un servidor web para proceder con la implementación y prueba de la sincronización de información entre el sistema web y la aplicación móvil.

5.3 Capas del Sistema y Comunicación entre capas

La aplicación móvil recibe información del sistema web y actualiza la misma de manera automática cada vez que se accesa a la aplicación.

El sistema web contiene las opciones de edición, modificación y eliminación de datos ingresados previamente de manera manual y se encarga de sincronizar y actualizar la información de la aplicación móvil mediante el formato Json implementado en el sistema web. De esta manera existe comunicación y sincronización del contenido entre la capa del sistema web y la capa de la aplicación móvil.

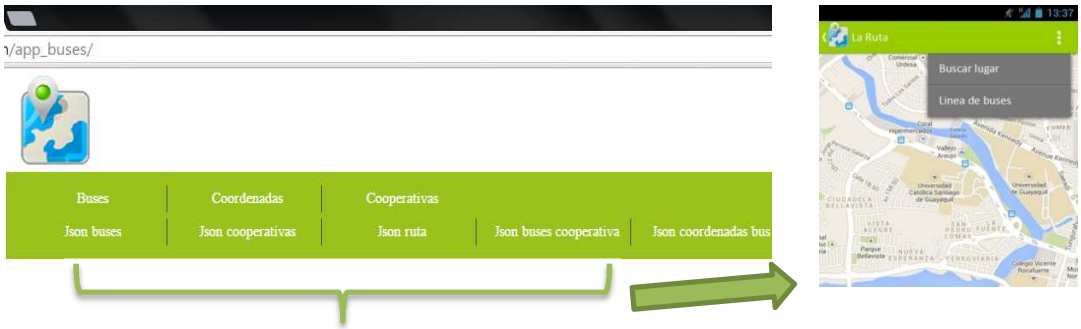


FIGURA # 35. Sincronización de Información

Nota: Muestra la sincronización y actualización de información de la capa web a la capa de la aplicación móvil. Elaborado por los autores. (Enero 2015)

La sincronización de la información entre estas dos capas del sistema es automática, bastará solamente con ejecutar una opción de la aplicación móvil y ésta verificará si existe información nueva y la actualizará.

5.4 Plan de pruebas

Después de haber realizado los procesos de instalación del sistema web y aplicativo móvil, se procedió a realizar las pruebas, para ello fue necesario dirigirse a la zona de referencia del proyecto que es centro noreste de la ciudad, y así poder correr la aplicación y verificar tiempo de respuesta de actualización de información del sistema web hacia la aplicación móvil y si los procesos se efectuaban de manera adecuada.

En el siguiente diagrama se detalla las actividades realizadas de acuerdo al cronograma realizado al inicio del proyecto.

No.	Actividades	FEBRERO			
		1	2	3	4
7	Instalación de Aplicativo y Pruebas				
	*Instalación de Sistema web en servidor				
	*Instalación de aplicación en equipo móvil				
	*Prueba de la aplicación móvil				
	*Corrección de errores				

FIGURA # 36. Diagrama de Gantt del proceso Pruebas

Nota: Diagrama que muestra el cronograma de actividades de las pruebas de la aplicación móvil. Elaborado por los autores. (Enero 2015)

Se instaló el sistema web en un servidor de la compañía donde labora un compañero, la dirección url es https: \\www.landaer.com.

Como lugar de referencia se tomó el parque centenario y se prueba la aplicación en una Tablet con un paquete activo de datos, en donde se realizó pruebas de las diferentes opciones de la aplicación móvil, la cual funcionó correctamente, teniendo un tiempo promedio de respuesta del servidor de 15 segundos, el cual es aceptable y depende también de la banda de navegación del equipo móvil.

CAPÍTULO 6
CONCLUSIONES Y RECOMENDACIONES

6.1 Conclusiones

Utilizar entornos de desarrollo gratuito y aprovechar la tecnología existente fue el objetivo de este proyecto teniendo como meta el desarrollo e implementación de esta aplicación.

La solución propuesta facilitará a las personas una herramienta de apoyo que le proporcionará información útil que le ayudará a escoger una línea de bus que la llevará al lugar indicado.

El desarrollar aplicaciones móviles gratuitas que puedan ser accedidas por cualquier persona en un dispositivo inteligente con sistema operativo Android, ayuda a incentivar a que las personas utilicen la aplicación de manera correcta y para los fines que fue creada.

Estas aplicaciones gratuitas diseñadas para el uso de la comunidad son un aporte para la misma y ayuda a ser más cooperativos y con voluntad de ayudar a la sociedad en el diario vivir.

6.2 Recomendaciones

Se recomienda a los usuarios el buen uso de la aplicación móvil, ya que fue desarrollada para fines comunitarios y sociales.

El equipo móvil deberá poseer Android como sistema operativo y la versión instalada deberá ser la 3.1 en adelante para garantizar el buen funcionamiento de la aplicación, además deberá contar con una conexión a internet.

Bibliografía

Ableson, F. (2011). Android in Action 2da. Edition. Stamford CT.

Coronel, E. G. (2010). PHP Profesional 1 era. Edición. Lima - Perú: Editorial Macro E.I.R.L.

Girones, J. (2012). El gran Libro de Android. 2da. Edición. Barcelona - España.

Holzner, S. (2009). PHP Manual de Referencia 1era. Edición. Interamericana Editores S.A.

López, C. P. (2008). MySQL para Windows y Linux 2da. Edición. Madrid - España: Editorial RA-MA.

Rodríguez, A. G. (2005). PHP 5 a traves de ejemplos 1era Edición. Madrid - España: RA-MA.

Soriano, A. (2012). Programación de Dispositivos Móviles. Barcelona - España.

Vaswani, V. (2009). PHP Soluciones de Programación 1era. Edición. Interamericana Editores.

Jesús Tomás (2011-2015) Arquitectura de Android. Diploma en especialización en desarrollo de aplicaciones para android. Recuperado de http://www.androidcurso.com/index.php/tutoriales-android/31-unidad-1-vision-general-y-entorno-de-desarrollo/99-arquitectura-de-android

Cosmos (2015) Distribución de versiones Android. Recuperado de http://www.xatakandroid.com/tag/distribucion-de-versiones-android

The php group (2013) Extensiones de bases de datos Sqlite. Recuperado de http://php.net/manual/es/book.sqlite.php

Ecured (2014) Sqlite. Recuperado de http://www.ecured.cu/index.php/SQLite

Morales (2012). Difusión Tecnológica. Recuperado de http://mappinggis.com/2012/06/glosario-terminos-gis/

Wikipedia (Abril, 2014). Formula de Haversine. Recuperado de http://es.wikipedia.org/wiki/F%C3%B3rmula_del_Haversine

Lara, A. L. (12 de Marzo de 2013). Saduce Blog. Retrieved 2 de Septiembre de 2013 from Saduce Blog: http://blog.saducelabs.com/android/google-maps-en-android/

Marco Besteiro, M. R. (n.d.). Universidad de Pais Basco . Retrieved 2 de Septiembre de 2013 from http://www.ehu.es/mrodriguez/archivos/csharppdf/ServiciosWeb/WebServices.pdf

Vierco. (7 de Agosto de 2010). El androide libre. Retrieved 2 de Septiembre de 2013 from El androide libre: http://www.elandroidelibre.com/2010/08/aprendiendo-androidv-inicializacion-a-la-api-del-gps.html

Google. (29 de Enero de 2013). Google Developer. Retrieved 2 de Septiembre de 2013 from Distance Matrix: https://developers.google.com/maps/documentation/distancematrix/?hl=es

Wonderbits (Marzo 2010): https://www.wonderbits.net/es/posts/15-tutorial-de-codeigniter.

ANEXOS

ANEXO 1

Definición de Tablas de la Aplicación

TABLA: Línea de Buses

columna	tipo de dato	detalle	descripción
ln_id	integer	Primary Key Auto-increment	id línea de bus
ln_id_coop	integer	Foreign Key	id cooperativa
ln_nombre	varchar	not null	nombre de la línea de bus
ln_des	varchar	not null	lugares por donde pasa la línea de bus

TABLA: Coordenadas

columna	tipo de dato	detalle	descripción
cord_id	integer	Primary Key Auto-increment	Id de coordenada
cord_ln_id	integer	Foreign Key not null	id linea de bus
cord_lat	double	no null	latitud
cord_lon	double	not null	longitud

TABLA: Cooperativas

columna	tipo de dato	detalle	descripción
coop_id	integer	Primary Key	id cooperativa
coop_nombre	varchar	not null	nombre de la cooperativa

Anexo 2. Función para el Cálculo de Ubicación

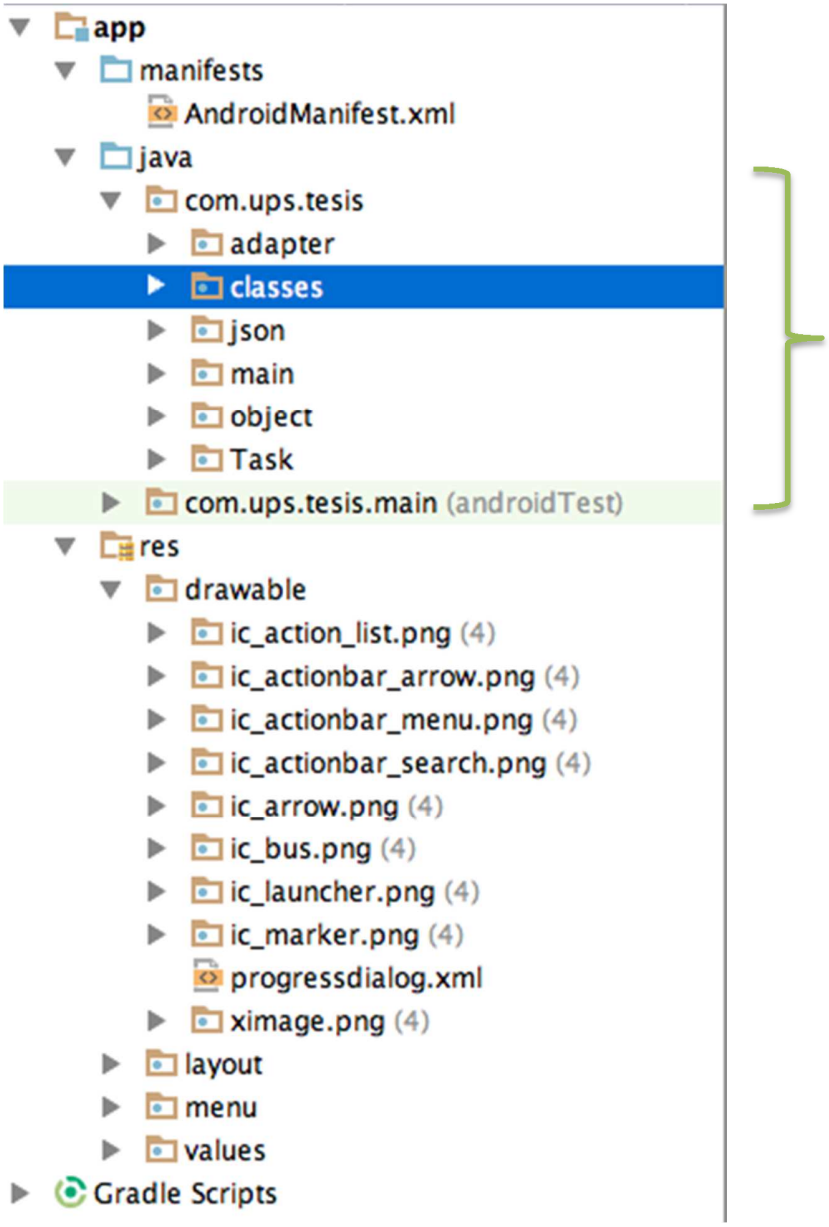
```
public Location getLocation() {

      try{
         locationManager=(LocationManager)
mContext.getSystemService(mContext.LOCATION_SERVICE);

         isGPSEnabled=
locationManager.isProviderEnabled(LocationManager.GPS_PROVIDER);
         isNetworkEnabled=
locationManager.isProviderEnabled(LocationManager.NETWORK_PROVIDER);
         isPassiveEnabled=
locationManager.isProviderEnabled(LocationManager.PASSIVE_PROVIDER);

         Log.i("gps",String.valueOf(isGPSEnabled));
         Log.i("network",String.valueOf(isNetworkEnabled));
         Log.i("passive",String.valueOf(isPassiveEnabled));

         if(!isGPSEnabled && !isNetworkEnabled && !isPassiveEnabled){

            System.exit(0);
            android.os.Process.killProcess(android.os.Process.myPid());

         }else{
            if(isGPSEnabled){
               this.canGetLocation=true;
            }
            locationManager.requestLocationUpdates(
LocationManager.NETWORK_PROVIDER,MIN_TIME_BW_UPDATES,
MIN_DISTANCE_CHANGE_FOR_UPDATES,this);
            locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER,
MIN_TIME_BW_UPDATES,MIN_DISTANCE_CHANGE_FOR_UPDATES,this);
            locationManager.requestLocationUpdates(
LocationManager.PASSIVE_PROVIDER,MIN_TIME_BW_UPDATES,
MIN_DISTANCE_CHANGE_FOR_UPDATES,this);

            locnetwork=
locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
            locgps=
locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER);
            locpassive=
locationManager.getLastKnownLocation(LocationManager.PASSIVE_PROVIDER);
```

```
        Log.i("network",String.valueOf(isNetworkEnabled));

        if(isNetworkAvailable()){
          if(isBetterLocation(locnetwork,locgps)){
            latitude = locnetwork.getLatitude();
            longitude = locnetwork.getLongitude();
            bestlocation = locnetwork;
            Log.i("locnetwork", bestlocation.toString());
          }else{
            if(isBetterLocation(locpassive,locgps)){
              latitude = locpassive.getLatitude();
              longitude = locpassive.getLongitude();
              bestlocation = locpassive;
              Log.i("locpassive", bestlocation.toString());

            }else{
              latitude = locgps.getLatitude();
              longitude = locgps.getLongitude();
              bestlocation = locgps;
              Log.i("locgps", bestlocation.toString());
            }
          }
        }else{
          if(isBetterLocation(locpassive,locgps)){
            latitude = locpassive.getLatitude();
            longitude = locpassive.getLongitude();
            bestlocation = locpassive;
            Log.i("locpassive", bestlocation.toString());

          }else{
            latitude = locgps.getLatitude();
            longitude = locgps.getLongitude();
            bestlocation = locgps;
            Log.i("locgps", bestlocation.toString());
          }
        }
      }
    }catch (Exception e) {
      //Log.e("Location Manager", e.getMessage());
      e.printStackTrace();
    }

    return bestlocation;

  }
```

Anexo 3. Función para leer url de sitio web y cargarla en la aplicación (Formato Json)

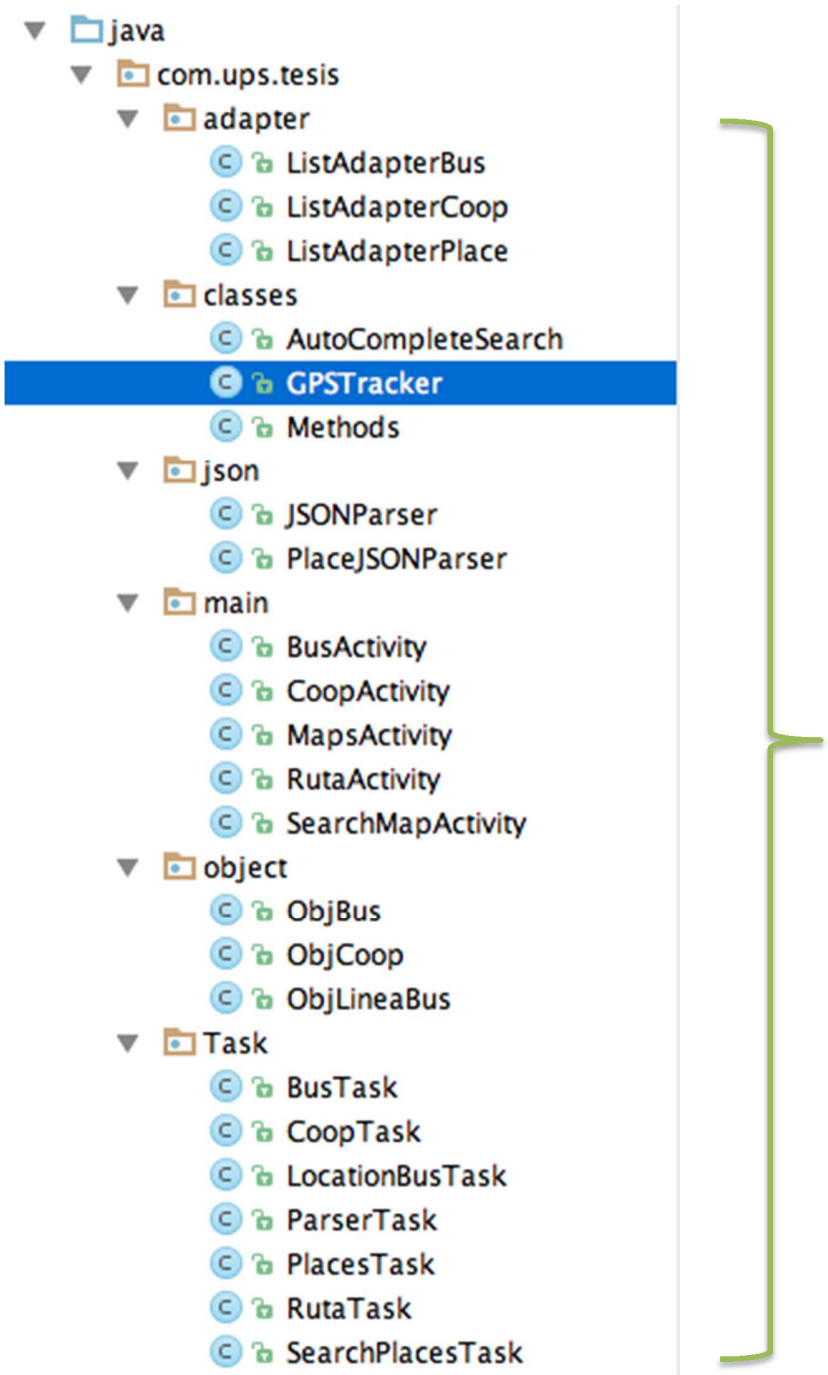
```
public JSONObject getJSONFromUrl(String url) {

        // Making HTTP request
        try {
            // defaultHttpClient
            DefaultHttpClient httpClient = new DefaultHttpClient();
            HttpPost httpPost = new HttpPost(url);

            HttpResponse httpResponse = httpClient.execute(httpPost);
            HttpEntity httpEntity = httpResponse.getEntity();
            is = httpEntity.getContent();

        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        } catch (ClientProtocolException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

        try {
            BufferedReader reader = new BufferedReader(new
InputStreamReader(
                    is, "iso-8859-1"), 8);
            StringBuilder sb = new StringBuilder();
            String line = null;
            while ((line = reader.readLine()) != null) {
                sb.append(line + "\n");
            }
            is.close();
            json = sb.toString();
        } catch (Exception e) {
            Log.e("Buffer Error", "Error converting result " + e.toString());
        }

        // try parse the string to a JSON object
        try {
            jObj = new JSONObject(json);
        } catch (JSONException e) {
            Log.e("JSON Parser", "Error parsing data " + e.toString());
        }

        // return JSON String
        return jObj;

    }
```

Anexo 4. Carpetas creadas para el desarrollo de la Aplicación



- Carpeta Adapter
- Carpeta Classes
- Carpeta Json
- Carpeta Main
- Carpeta Object
- Carpeta Task

Anexo 5. Descripción de Carpetas creadas para el desarrollo de la Aplicación



Carpeta Adapter

Esta carpeta contiene las clases creadas para mostrar el listado de información.

Carpeta Classes

Esta carpeta contiene las clases de métodos para la localización de ubicación de la persona, empleado para búsqueda de lugares y otros métodos de conversión de datos.

Carpeta JSON

Esta carpeta contiene las clases para leer y llamar las url con la información tipo Json que se encuentra en el servidor donde se guardan los datos de los buses, lugares y coordenadas.

Carpeta Main

En esta carpeta se encuentras las clases de las ventanas de la aplicación, mapa, ruta, buses, cooperativas y búsqueda en el mapa.

Carpeta Object

En esta carpeta se encuentran las clases para poder obtener los datos de las url de tipo json y mostrarlas en la aplicación

Carpeta Task

En esta carpeta se encuentra las clases en las cuales se va ejecutando los procesos de espera mientras se recopila la información en segundo plano.

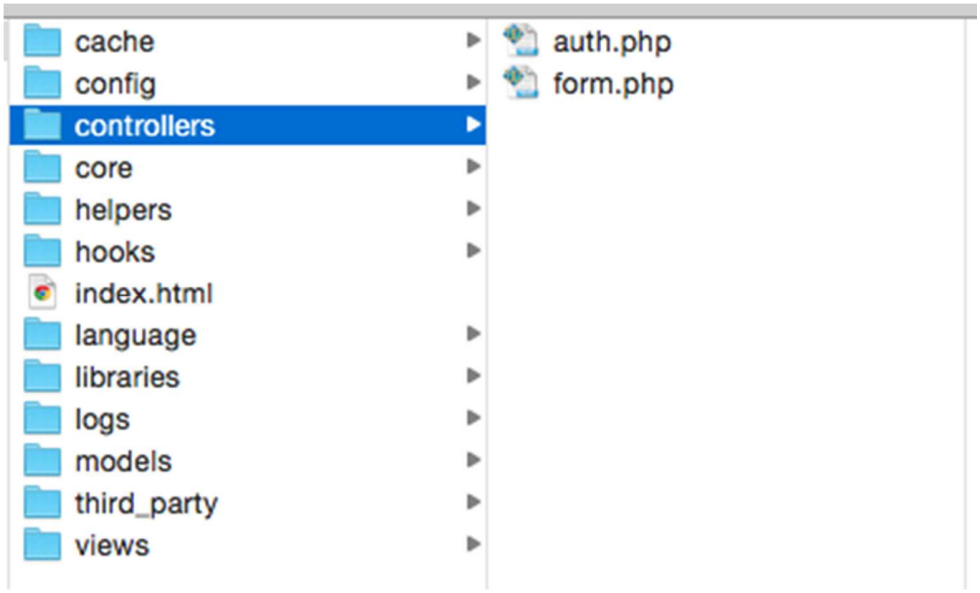
Anexo 6. Descripción de Carpetas creadas para el desarrollo del Sitio Web

Para el desarrollo del Sitio Web se ha utilizado el framework CodeIgniter de Ellislab.

CodeIgniter es un framework minimalista que ofrece sólo un andamiaje a partir del cual construir tu aplicación MVC. Su nucleo provee un controlador de enrutado básico, carga de modelos e interfaz con la base de datos y una carpeta conveniente de "vistas" para guardar todas las vistas de la aplicación (en plantillas de PHP).

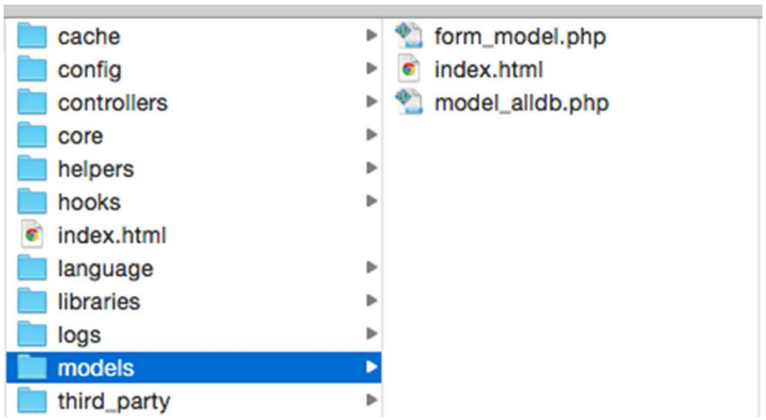
Al descargar el framework y descomprimir su código de distribución por primera vez, se encuentra la siguiente estructura de árbol.

Carpeta Controladores



En la carpeta de controladores, para las rutas, CodeIgniter procesa el primer directorio en el URL, el segundo sería la "acción" que se enlaza con la función de igual nombre en el controlador llamado y finalmente sus parámetros.

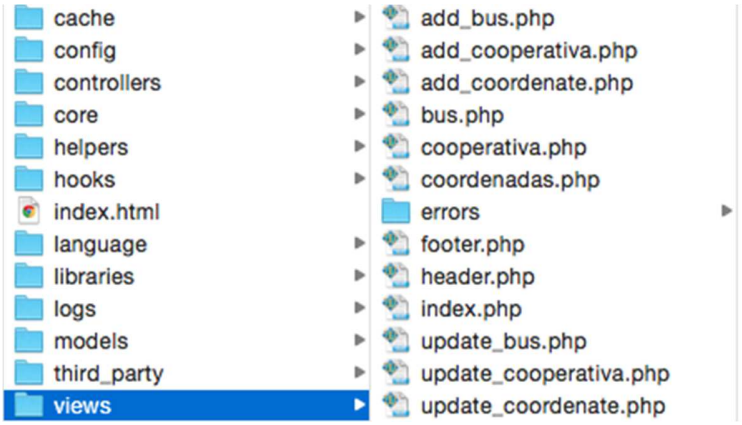
Carpeta Modelos y Active Record



CodeIgniter tiene lo que se llama drivers de base de datos y cache en su nomenclatura, que básicamente abstrae las acciones comunes para varias formas internas de funcionamiento o back-ends. Eso permite usar sin problemas MySQL, ODBC, SQLite, SQLServer, PDO, Postgre or Cubrid directamente a través de su patrón de acceso ActiveRecord modificado.

Una vez que se haya definido un modelo a utilizar sólo se tiene que llamar al cargador en la instancia del controlador $this->load->model('themodel); y después acceder a sus funciones así $this->Themodel->getLirics();

Carpeta Vistas



Las vistas en CodeIgniter son sencillos archivos de HTML/XML/texto conteniendo PHP incrustado pero con un ámbito especial donde las variables pasadas como segundo parámetro a load->view() serán locales. Para nuestro proyecto se crearon los archivos php que se muestran en la imagen.