



Instituto Politécnico Nacional

Escuela Superior de Ingeniería Mecánica Y Eléctrica

UNIDAD "CULHUACAN"

**Sistema de Notificación de Proximidad para
Transporte Público en ruta:
Xochimilco/ Bosque de Nativitas - Metro San Lázaro,
utilizando el Sistema de Posicionamiento Global
y Google Cloud Messaging (GCM).**

TESINA

**QUE PARA OBTENER EL TITULO DE
INGENIERO EN COMPUTACION**

PRESENTAN

WILLIAM LUIS CALDERON PALOMINO

DANIEL SANCHEZ CARMONA

ASESORES:

M. en C. IXCHEL MARTINEZ PIRO

ING. MIGUEL ANGEL MIRANDA HERNANDEZ

MEXICO, D.F. MAYO 2014



Índice

Resumen	I
Antecedentes	II
Planteamiento del problema	II
Objetivo General.....	III
Objetivos Particulares.....	III
Justificación	III
Hipótesis	IV
Estado del Arte	IV
Capítulo 1 Marco Teórico	1
1.1. Sistemas de Información	1
1.1.1. Componentes del Sistema de Información	1
1.1.2. Objetivos del Sistema de Información.....	2
1.1.3. Clasificación de los Sistemas de Información	2
1.2. Sistema de Coordenadas Geográficas	4
1.3. Sistemas en Tiempo Real	5
1.3.1. Clases de Sistemas en Tiempo Real	5
1.4. Middleware	6
1.5. Servlet	6
1.6. Sistemas Operativos	7
1.6.1. Sistema Operativo Android	8
1.7. Bases de Datos	16
1.7.1. Comparativa de los SGBD más utilizados	18
1.8. Lenguajes y Entornos de Programación	19

1.8.1.	PHP	21
1.8.2.	Java	23
1.9.	Acceso a Internet para conexiones móviles	24
1.9.1.	Los paquetes de información	25
1.9.2.	Servicio de Nombres.....	25
1.10.	La arquitectura Cliente/Servidor	25
1.10.1.	Cliente	26
1.10.2.	Servidor	27
1.11.	Arquitectura de una Aplicación Web	28
1.12.	Servicios WEB	29
1.12.1.	HTTP.....	29
1.12.2.	Tipos de notificaciones	32
1.13.	Sistema de Posicionamiento Global	33
1.13.1.	La idea básica del GPS	34
1.13.2.	Concepto de Localización por medio del GPS	35
1.13.3.	Cálculos con coordenadas de GPS	37
1.14.	Google Cloud Messaging para Android.....	39
1.14.1.	Funcionamiento	39
1.14.2.	Características	40
1.14.3.	Arquitectura	40
Capítulo 2 Diseño		43
2.	Diseño del Sistema Móvil de notificación de proximidad para transporte.....	43
2.1.	Arquitectura propuesta para el sistema	43
2.1.1.	Sistema de Información.....	43

2.1.2.	Hardware	43
2.1.3.	Software	44
2.1.4.	Metodología.....	46
2.1.5.	Estándares	47
2.2.	Propuesta de solución	48
2.2.1.	Diagrama General del Sistema	49
2.2.2.	Diseño de la Base de Datos.....	51
Capítulo 3	Desarrollo	53
3.	Desarrollo del Sistema Móvil de notificación de proximidad para transporte.....	53
3.1.	Central de Control.....	53
3.1.1.	Funciones en PHP	54
3.2.	Desarrollo de la Aplicación de Abordo	55
3.2.1.	Diagrama a Bloques.....	56
3.2.2.	Módulo de envío de datos.....	57
3.2.3.	Módulo de Registro	60
3.2.4.	Módulo de Obtención de Datos del GPS	61
3.3.	Aplicación del cliente	62
3.3.1.	Extensiones necesarias	62
3.3.2.	Permisos.....	62
3.3.3.	Diagrama lógico	64
3.3.4.	Registro GCM.....	65
3.3.5.	Recepción de cadenas JSON.....	68
3.3.6.	Notificaciones	69
Capítulo 4	Resultados.....	71

4. Resultados del Sistema Móvil de notificación de proximidad para transporte.	71
4.1. Pruebas con GCM.	71
4.2. Pruebas con la Aplicación del Cliente	72
4.3. Pruebas con la Aplicación de Abordo	74
4.4. Pruebas Reales recorriendo parte de la Ruta 39-A Xochimilco/Bosque de Nativitas – Metro San Lazaro	77
4.5. Pagina web de Monitoreo.....	78
Referencias.....	82
Glosario	85
Modelos ISO	86
TABLA DE ESTANDARES DE CODIFICACION EN JAVA	89
PEAR: Estándares de desarrollo para PHP	97
Código del proyecto	100
Matriz de Pruebas	100

Índice de figuras

Figura 1.1 Arquitectura Middleware.....	6
Figura 1.2 Arquitectura Android	13
Figura 1.3 Sistema Gestor de Base de datos (SGDB). [5].....	16
Figura 1.4 Proceso de visita a página PHP	21
Figura 1.5 Proceso de ejecución de un programa en Java.....	24
Figura 1.6 Modelo Cliente/Servidor	26
Figura 1.7 Arquitectura a tres capas (Aplicación Web).....	29
Figura 1.8 Constelaciones GPS.....	34
Figura 1.9 Posicionamiento Global idea básica.....	35

Figura 1.10 Localización con 4 satélites	36
Figura 1.11 Triangulo imaginario.....	38
Figura 2.1 Modelo de procesos software IEEE	46
Figura 2.2 Ciclo de vida en cascada. Mejoramiento por pasos	47
Figura 2.3 Diagrama General.....	49
Figura 2.4 Diagrama de Componentes [Ver anexo]	50
Figura 2.5 Diagrama de Arquitectura.....	51
Figura 2.6 Diagrama entidad - relación de la base de datos del sistema	52
Figura 3.1 Diagrama Central de Control	53
Figura 3.2 Diagrama a bloques de la aplicación de abordó	56
Figura 3.3 Diagrama de clases para conexión asíncrona	59
Figura 3.4 Diagrama de Registro del vehículo	60
Figura 3.5 Diagrama módulo de obtención de datos del GPS	61
Figura 3.6 Diagrama a bloques de la aplicación del cliente	64
Figura 3.7 Diagrama de Validación Registro	65
Figura 3.8 Diagrama de Registro en nuestro servidor	67
Figura 3.9 Diagrama para almacenar las rutas seleccionadas	68
Figura 3.10 Diagrama de solicitud de ruta.....	69
Figura 4.1 Portal web de pruebas con GCM	71
Figura 4.2 Notificación del mensaje enviado desde el portal web	72
Figura 4.3 Muestra del mensaje	72
Figura 4.4 Pantalla de bienvenida.....	73
Figura 4.5 Registrando al usuario.....	73
Figura 4.6 Mensajes en el proceso de registro.....	73

Figura 4.7 Lista de Rutas	74
Figura 4.8 Lista de estaciones descargadas de la Ruta	74
Figura 4.9 Pantalla de bienvenida de la aplicación de abordó	75
Figura 4.10 Pantalla de registro del vehículo y conductor	75
Figura 4.11 Lista de Rutas para el vehículo	76
Figura 4.12 Registrando a una persona y vehículo	76
Figura 4.13 Pantalla de en el momento que ya se esta obteniendo la posición	76
Figura 4.14 Usuarios registrados	79
Figura 4.15 Unidades registradas	80
Figura 4.16 Relación petición - usuario	81

Resumen

En este trabajo se presentará el desarrollo de un Sistema de Información móvil de seguimiento de transporte público; se propone a forma de prueba, el uso de un vehículo propio que jugará el rol de uno de los autobuses de la RTP (ruta Xochimilco/Embarcadero Nativitas a Metro San Lázaro que va por Cafetales) por medio del GPS integrado en un dispositivo móvil a bordo del vehículo.

El sistema está constituido de 3 partes: la central de control, la aplicación del cliente y la aplicación de simulación de autobús(a bordo de un automóvil), las cuales serán desarrolladas en el transcurso de este documento.

El objetivo de este proyecto es desarrollar un sistema de notificación basado en GPS para proporcionar información de la proximidad de unidades, orientado a la red de transporte de pasajeros del Distrito Federal, utilizando los recursos brindados por los dispositivos móviles. Este sistema es una posible herramienta para apoyar a los usuarios de la red de transporte de pasajeros del Distrito Federal.

Antecedentes

En la Ciudad de México, a partir de que se comenzaron a diseñar las diferentes rutas de transporte, y se empiezan a utilizar los autobuses como sistema de transporte colectivo, se ha planteado ofrecer a los usuarios periodos promedio que tarde la unidad para cada estación a la que tiene que arribar, esto, regularmente carece de exactitud dado que los factores externos no se pueden medir, causando en el usuario la dificultad de decidir si esperar o no una unidad de esa ruta.

En la actualidad se puede observar que se ha incrementado considerablemente el uso de dispositivos móviles, con la aparición de estas tecnologías móviles se le han facilitado múltiples servicios a los usuarios, como conexiones a internet por redes de datos, mensajerías instantáneas, servicios de ubicación, servicios de notificaciones, etc. Elementos que en conjunto se pueden aprovechar para contribuir en la experiencia de los usuarios del transporte público y brindarles un mejor servicio.

Planteamiento del problema

El problema principal que al mismo tiempo representa una gran oportunidad para los objetivos de este trabajo es que los usuarios de la ruta Bosque de Nativitas – Metro San Lázaro de la red de transporte de pasajeros del Distrito Federal carecen de una herramienta que les permita decidir acertadamente que tan conveniente es esperar una unidad de la ruta u optar por una alternativa de transporte, debido a que por razones externas, llámese accidentes viales, condiciones climáticas, movimientos de protesta, las unidades de transporte se ven afectadas para cumplir con el itinerario establecido.

Se optó en direccionar los beneficios de este proyecto a la red de transporte público dada la preferencia que tiene por los usuarios del transporte público en general por sus ventajas de rapidez y de seguridad del mismo, siendo la forma de transporte público más económica del Distrito Federal.

Objetivo General

Desarrollar un sistema de notificación basado en GPS para proporcionar información sobre la proximidad de unidades de la red de transporte público ruta Xochimilco / Bosque Nativitas- Metro San Lázaro a usuarios de la ruta con dispositivos compatibles con el sistema GCM.

Objetivos Particulares

- Implementar estación central de control para almacenar bases de datos, funciones para la comunicación cliente - servidor, Almacenamiento, Actualizaciones y cálculos necesarios.
- Desarrollar aplicación para extraer los datos del GPS de un Dispositivo Móvil, informando así a la central de control de la ubicación de las unidades de transporte.
- Desarrollar aplicación para el usuario de la RTP que le notifique cuando una unidad de transporte esté dentro de la periferia de los puntos de ascenso y descenso autorizados seleccionados por el mismo.

Justificación

Es cotidiano que las personas tengan la incertidumbre, de a qué hora va a arribar a la estación la unidad de transporte que ellos desean abordar.

El desarrollo de este proyecto en el ámbito social se justifica dado que beneficiará a un amplio número de personas dada la preferencia que tiene este tipo de transporte en la población del Distrito Federal, que en muchas ocasiones han esperado alguna unidad durante un periodo considerable de tiempo, y han perdido la oportunidad de tomar alguna otra opción de transporte; contribuyendo a la reducción de retrasos de los usuarios del transporte público apoyándoles en su toma de decisiones.

En el ámbito tecnológico se justifica al usar dispositivos móviles, con sistema operativo Android, dada la facilidad de adquirir los equipos, elevando el índice de usuarios que portan dispositivos que trabajen en esa plataforma.

Hipótesis

El uso de tecnologías móviles en Sistemas de Información puede dar mayor tranquilidad a los usuarios que estén en espera de una unidad de la red de transporte de pasajeros.

Estado del Arte

El uso de sistemas GPS es un tema ampliamente estudiado en el área de transporte en especial para empresas que cuenten con móviles terrestres y tener un control de sus unidades.

En [1], Romero Rojano describe en su trabajo de tesis titulado “Sistema de localización y seguimiento de móviles terrestres utilizando el Sistema de Posicionamiento Global” donde se plantea el problema de la necesidad de observar un móvil terrestre y saber de dónde viene y hacia a donde se dirige.

En este trabajo el autor presenta la solución como el diseño, construcción e implementación de un Sistema que determinará la posición geográfica de una unidad terrestre, así como transmitir esta información por medio de un enlace de radio a una estación de control y monitorear en tiempo real la ubicación geo-referenciada de dicha unidad terrestre.

El sistema se constituye de tres partes:

1. El equipo de Abordo (Dispositivos en los Vehículos Terrestres).
2. Una Red de Radiocomunicaciones.
3. Una estación de control o estación base.

Se presenta este sistema como una alternativa en la seguridad sobre vehículos y/o sistemas orientados a la administración de flotillas.

En [2], Martínez Osorio describe en su trabajo de tesis titulado “Sistema de rastreabilidad GPS con interface Web y SMS para dispositivos móviles”. Donde se plantea el problema de la necesidad de explotar el modulo GPS de un dispositivo móvil, ya sea que se necesite en caso de emergencia y poder compartir la ubicación del usuario, se aprovecha también para hacer frente como una alternativa a las aplicaciones de alto costo de diversas compañías que ofrecen localización global.

En este trabajo el autor presenta la idea de localización de un dispositivo, haciendo uso del Sistema de Posicionamiento Global (GPS), enviando los datos por medio del Servicio de Mensajes Cortos (SMS), mostrando la información en un sitio WEB.

Haciendo uso de un dispositivo con Sistema Operativo Windows Mobile 6. El sistema se constituye de:

- Control de GPS:

1. Apertura y Lectura del Puerto.
2. Filtrado de Sentencias NMEA.
3. Decodificación de Sentencias.
4. Cálculo de la información.

- Interface del Usuario:

1. Parámetros de Configuración de Puerto.
2. Parámetros de Configuración de Envío.
3. Visualización de la información.
4. Envío de posición a Internet.
5. Envío de posición vía SMS.

- WEB:

1. Recepción de la Posición.
2. Monitoreo de la Posición.

Capítulo 1 Marco Teórico

1.1. Sistemas de Información

Es un conjunto de elementos que interactúan entre sí con el fin de apoyar las actividades de una empresa o negocio.

Es el conjunto total de procedimientos, operaciones, funciones y difusión de datos o información en una organización.

1.1.1. Componentes del Sistema de Información

Un Sistema de Información realiza cuatro actividades básicas: entrada de información, almacenamiento, procesamiento y salida de información.

➤ Entrada de información:

Es el proceso mediante el cual el Sistema de Información toma los datos que requiere para procesar la información. Las entradas pueden ser manuales o automáticas. Las **manuales** son aquellas que se proporcionan en forma directa por el usuario, mientras que las **automáticas** son datos o información que provienen o son tomados de otros sistemas o módulos. Esto último se denomina interfaces automáticas.

Las unidades típicas de entrada de datos a las computadoras son las terminales, las cintas magnéticas, las unidades de disquete, los códigos de barras, los escáner, la voz, los monitores sensibles al tacto, el teclado y el ratón, entre otras cosas.

➤ Almacenamiento de Información:

Es una de las actividades o capacidades más importantes que tiene una computadora, ya que a través de esta propiedad el sistema puede recordar la información guardada en la sesión o proceso anterior. Esta información suele ser almacenada en estructuras de información denominadas ficheros. La unidad típica de almacenamiento son los discos magnéticos o duros, de estado sólido, los discos compactos (CD-ROM).

➤ **Procesamiento de Información:**

Es la capacidad del Sistema de Información para efectuar cálculos de acuerdo con una secuencia de operaciones preestablecidas. Estos cálculos pueden efectuarse con datos introducidos recientemente en el sistema o bien con datos que están almacenados. Esta característica de los sistemas permite la transformación de datos fuente en información que puede ser utilizada para la toma de decisiones.

➤ **Salida de Información:**

La salida es la capacidad de un Sistema de Información para sacar la información procesada o bien datos de entrada al exterior. Las unidades típicas de salida son las impresoras, terminales, cintas magnéticas, la voz, los graficadores y los plotters, entre otros. Es importante aclarar que la salida de un Sistema de Información puede constituir la entrada a otro Sistema de Información o módulo. En este caso también existe una interface automática de salida.

1.1.2. Objetivos del Sistema de Información

Algunos de los principales objetivos de los Sistemas de Información, son:

- Proporcionar datos oportunos y exactos que permitan tomas de decisiones acertadas y mejorar la relación entre los recursos de la empresa.
- Garantizar información exacta y confiable, así como su almacenamiento de tal forma que esté disponible cuando se necesite.
- Servir como herramienta para que los gerentes realicen planeación, control y toma de decisiones en sus empresas.

1.1.3. Clasificación de los Sistemas de Información

- **Sistemas transaccionales:**

A través de éstos suelen lograrse ahorros significativos de mano de obra, debido a que automatizan tareas operativas de la organización.

Con frecuencia son el primer tipo de Sistemas de Información que se implanta en las organizaciones. Se empieza apoyando las tareas a nivel operativo de la organización para continuar con los mandos intermedios y posteriormente con la alta administración conforme evolucionan.

Son intensivos en entrada y salida de información; sus cálculos y procesos suelen ser simples y poco sofisticados. Estos sistemas requieren mucho manejo de datos para poder realizar sus operaciones y como resultado generan también grandes volúmenes de información.

Tienen la propiedad de ser recolectores de información, es decir, a través de estos sistemas se cargan las grandes bases de información para su explotación posterior. Estos sistemas son los encargados de integrar gran cantidad de la información que se maneja en la organización, la cual será utilizada posteriormente para apoyar a los mandos intermedios y altos.

Son fáciles de justificar ante la dirección general, ya que sus beneficios son visibles y palpables. El proceso de justificación puede realizarse enfrentando ingresos y costos. Esto se debe a que en el corto plazo se pueden evaluar los resultados y las ventajas que se derivan del uso de este tipo de sistemas. Entre las ventajas que pueden medirse se encuentra el ahorro de trabajo manual.

Son fácilmente adaptables a paquetes de aplicación que se encuentran en el mercado, ya que automatizan los procesos básicos que por lo general son similares o iguales en otras organizaciones. Ejemplos de este tipo de sistemas son la facturación, nóminas, cuentas por cobrar, cuentas por pagar, contabilidad general, conciliaciones bancarias, inventarios, etcétera.

- **Sistemas de Apoyo a las Decisiones**

Las principales características de estos sistemas son las siguientes:

Suelen introducirse después de haber implantado los Sistemas Transaccionales más relevantes de la empresa, ya que estos últimos constituyen su plataforma de información.

La información que generan sirve de apoyo a los mandos intermedios y a la alta administración en el proceso de toma de decisiones.

Suelen ser intensivos en cálculos y escasos en entradas y salidas de información. Así, por ejemplo, un modelo de planeación financiera requiere poca información de entrada, genera poca información como resultado, pero puede realizar muchos cálculos durante su proceso.

No suelen ahorrar mano de obra. Debido a ello, la justificación económica para el desarrollo de estos sistemas es difícil, ya que no se conocen los ingresos del proyecto de inversión.

Suelen ser Sistemas de Información interactivos y amigables, con altos estándares de diseño gráfico y visual, ya que están dirigidos al usuario final.

Apoyan la toma de decisiones que, por su misma naturaleza son repetitivas y de decisiones no estructuradas que no suelen repetirse. Por ejemplo, un Sistema de Compra de Materiales que indique cuándo debe hacerse un pedido al proveedor o un Sistema de Simulación de Negocios que apoye la decisión de introducir un nuevo producto al mercado.

Estos sistemas pueden ser desarrollados directamente por el usuario final sin la participación operativa de los analistas y programadores del área de Informática.

Este tipo de sistemas puede incluir la programación de la producción, compra de materiales, flujo de fondos, proyecciones financieras, modelos de simulación de negocios, modelos de inventarios, etcétera.

1.2. Sistema de Coordenadas Geográficas

Es un sistema de referencia que utiliza las dos coordenadas angulares, latitud (norte o sur) y longitud (este u oeste) y sirve para determinar los ángulos laterales de la superficie terrestre (o en general de un círculo o un esferoide).

Latitud y Longitud

El sistema de coordenadas geográficas, basado en paralelos y meridianos, se utiliza para determinar la posición de cualquier punto en el planeta. Para ello se miden dos distancias: entre el punto deseado y el Ecuador, y entre ese punto y el meridiano cero. Estas distancias reciben el nombre de latitud y longitud, respectivamente, y se miden en grados (°) debido a la forma esférica del globo terráqueo.

- La Latitud se mide desde el Ecuador hasta los polos. Las líneas de latitud son los paralelos, y sus valores van desde el 0° (Ecuador) al 90° (polos). El Ecuador es la latitud más baja, cero grados. La latitud puede ser norte (Hemisferio Norte) o sur (Hemisferio Sur).
- La Longitud se mide según los meridianos. Las líneas de longitud son los meridianos, y sus valores van desde el 0° (meridiano de Greenwich) hasta su complementario, el 180°, el meridiano de cambio de fecha. El meridiano de Greenwich es la longitud más baja, cero grados. La longitud puede ser oeste o este.

Un punto cualquiera de la superficie terrestre puede ser situado exactamente por la intersección de un paralelo y un meridiano, es decir por dos números o coordenadas que representan a la latitud y la longitud. Para señalar en el mar una posición exacta se indica la latitud y la longitud. Para dar las coordenadas geográficas, primero se escribe la latitud y luego la longitud.

1.3. Sistemas en Tiempo Real

Un sistema de Tiempo Real es un sistema informático que:

- Interacciona con su entorno físico.
- Responde a los estímulos del entorno dentro de un plazo de tiempo determinado.

No basta con que las acciones del sistema sean correctas, sino que, además, tienen que ejecutarse dentro de un intervalo de tiempo determinado.

Los sistemas tiempo real suelen estar integrados en un sistema de ingeniería más general, en el que realizan funciones de control y/o monitorización llamados sistemas empotrados.

1.3.1. Clases de Sistemas en Tiempo Real

Críticos (hard real-time systems):

- Los plazos de respuesta deben respetarse siempre estrictamente.
- Una sola respuesta tardía a un suceso externo puede tener consecuencias fatales.

Acríticos (soft real-time systems):

- Se pueden tolerar retrasos ocasionales en la respuesta a un suceso.

1.4. Middleware

Es la capa intermedia entre el sistema operativo y la aplicación, asistiéndola para interactuar o comunicarse con otras aplicaciones, eliminando los problemas de heterogeneidad.

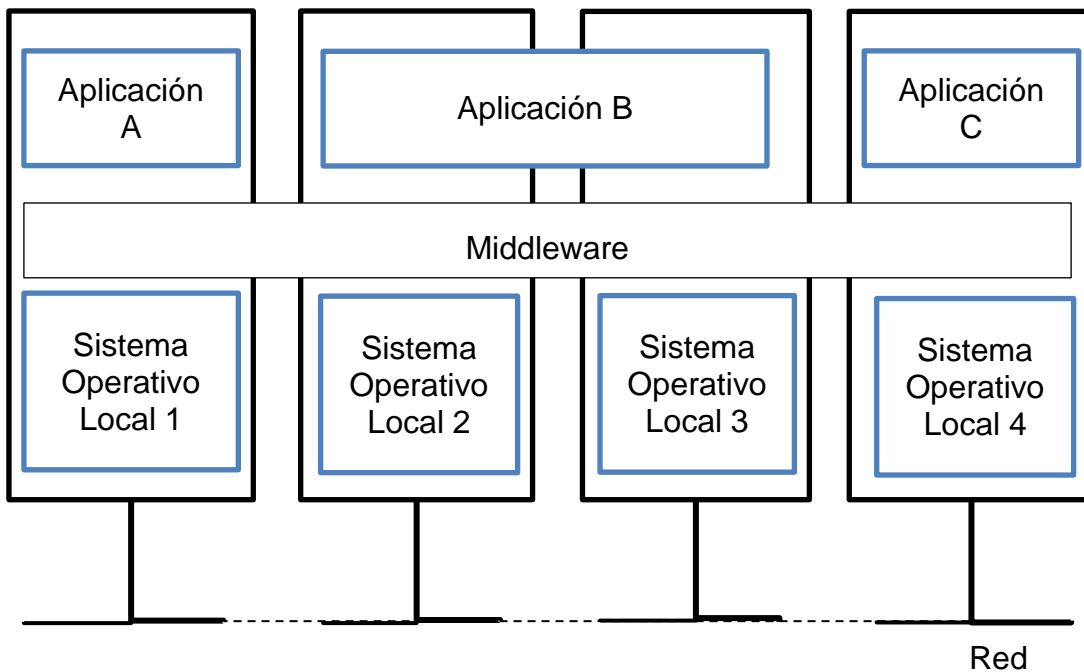


Figura 1.1 Arquitectura Middleware

1.5. Servlet

Es la tecnología de Sun Java, en donde se desarrollan programas que realizan tareas en el servidor, teniendo la función de una capa intermedia entre una solicitud proveniente de un navegador Web u otro cliente HTTP, y las aplicaciones del servidor.

Teniendo como función principal otorgar páginas web dinámicas y personalizadas, utilizando el acceso a bases de datos, flujos de trabajo y otros recursos.

1.6. Sistemas Operativos

El sistema operativo es el programa que actúa de interfaz entre usuarios y hardware de un sistema informático, ofreciendo un entorno para que se puedan ejecutar otros programas.

Su principal objetivo es facilitar el uso del sistema informático y garantizar que sus recursos se utilizan en forma eficiente. Se puede considerar que todo sistema informático está constituido por cuatro componentes o elementos.

- El Hardware (CPU, memoria, discos duros, periféricos, etc.)
- El sistema operativo.
- Los programas de aplicación (gestores de bases de datos, aplicaciones de gestión, hojas de cálculo, procesadores de textos).
- Los Usuarios (personas u otros ordenadores).

Por lo tanto, el sistema operativo actúa como un asignado de los recursos disponibles (tiempo de CPU, memoria, dispositivos de entrada/salida de datos, espacio de almacenamiento en disco...) para facilitar la ejecución de otros programas y ofrecer una serie de servicios a los usuarios mediante el intérprete de comandos y un entorno gráfico.

Los sistemas operativos modernos presentan una arquitectura por niveles, que simplifica su diseño e implementación. Esta es la estructura típica de un sistema operativo, desde las funciones más básicas a las de mayor nivel.

- Planificación de la CPU (ejecución de procesos).
- Gestión de la memoria principal.
- Control de dispositivos entrada/salida.
- Gestión del sistema de ficheros.
- Interfaz de Usuario (interprete de comandos y entorno gráfico).

Las aplicaciones y programas de usuarios acceden a los recursos de la maquina mediante llamadas a funciones del sistema operativo, definidas en una API (Aplicación Programa Interface, “Interfaz de Programación de Aplicaciones”).

Los primitivos sistemas operativos eran “monousuario” y “monotarea” (no podían ejecutar varias aplicaciones de forma simultanea).

1.6.1. Sistema Operativo Android

La telefonía móvil está cambiando la sociedad actual de una forma tan significativa como lo ha hecho Internet. Esta revolución no ha hecho más que empezar, los nuevos terminales ofrecen unas capacidades similares a un ordenador personal, lo que permite que puedan ser utilizados para leer nuestro correo o navegar por Internet. Pero a diferencia de un ordenador, un teléfono móvil siempre está en el bolsillo del usuario. Esto permite un nuevo abanico de aplicaciones mucho más cercanas al usuario. De hecho, muchos autores coinciden en que el nuevo ordenador personal del siglo veintiuno será un terminal móvil.

El lanzamiento de Android como nueva plataforma para el desarrollo de aplicaciones móviles ha causado una gran expectación y está teniendo una importante aceptación tanto por los usuarios como por la industria. En la actualidad se está convirtiendo en una seria alternativa frente a otras plataformas como Symbian, iPhone o Windows Phone.

1.6.1.1. Orígenes

Google adquiere Android Inc. en el año 2005. Se trataba de una pequeña compañía, que acababa de ser creada, orientada a la producción de aplicaciones para terminales móviles. Ese mismo año empiezan a trabajar en la creación de una máquina virtual Java optimizada para móviles (Dalvik VM).

En el año 2007 se crea el consorcio Handset Alliance con el objetivo de desarrollar estándares abiertos para móviles. Está formado por Google, Intel, Texas Instruments, Motorola, T-Mobile, Samsung, Ericson, Toshiba, Vodafone, NTT DoCoMo, Sprint Nextel y otros. Una pieza clave de los objetivos de esta alianza es promover el diseño y difusión de

la plataforma Android. Sus miembros se han comprometido a publicar una parte importante de su propiedad intelectual como código abierto bajo licencia Apache v2.0.

En noviembre del 2007 se lanza una primera versión del Android SDK. Al año siguiente aparece el primer móvil con Android (T-Mobile G1). En octubre Google libera el código fuente de Android principalmente bajo licencia de código abierto Apache (licencia GPL v2 para el núcleo). Ese mismo mes se abre Android Market, para la descarga de aplicaciones. En abril del 2009 Google lanza la versión 1.5 del SDK que incorpora nuevas características como el teclado en pantalla. A finales del 2009 se lanza la versión 2.0 y durante el 2010 las versiones 2.1, 2.2 y 2.3.

Durante el año 2010 Android se consolida como uno de los sistemas operativos para móviles más utilizados, con resultados cercanos al iPhone e incluso superando al sistema de Apple en EE.UU.

En el 2011 se lanzan la versión 3.0, 3.1 y 3.2 específica para tabletas y la 4.0 tanto para móviles como para tabletas. Durante este año Android se consolida como la plataforma para móviles más importante alcanzando una cuota de mercado superior al 50%.

En 2012 Google cambia su estrategia en su tienda de descargas online, reemplazando Android Market por Google Play Store. Donde en un solo portal unifica tanto la descarga de aplicaciones como de contenidos. En este año aparecen las versiones 4.1 y 4.2 del SDK. Android mantiene su espectacular crecimiento, alcanzando a finales de año una cuota de mercado del 75%.

1.6.1.2. **¿Que hace a Android especial?**

Existen muchas plataformas para móviles (iPhone, Symbian, Windows Phone, BlackBerry, Palm, Java Mobile Edition, Linux Mobile (LiMo),...); sin embargo Android presenta una serie de características que lo hacen diferente. Es el primero que combina en una misma solución las siguientes cualidades:

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

- ❖ **Adaptable a cualquier tipo de hardware.** Android no ha sido diseñado exclusivamente para su uso en teléfonos y tabletas. Hoy en día podemos encontrar relojes, cámaras, electrodomésticos y gran variedad de sistemas empotrados que se basan en este sistema operativo. Este hecho tiene sus evidentes ventajas, pero también va a suponer un esfuerzo adicional al programador. La aplicación ha de funcionar correctamente en dispositivos con gran variedad de tipos de entrada, pantalla, memoria, etc. Esta característica contrasta con la estrategia de Apple. En iOS tenemos que desarrollar una aplicación para iPhone y otra diferente para iPad.
- ❖ **Portabilidad asegurada.** Las aplicaciones finales son desarrolladas en Java lo que nos asegura que podrán ser ejecutadas en cualquier tipo de CPU, tanto presente como futuro. 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 en xml, lo que permite que una misma aplicación se ejecute en un móvil de pantalla reducida o en un TV.

Filosofía de dispositivo siempre conectado a Internet.

- **Gran cantidad de servicios incorporados.** por ejemplo, localización basada tanto en GPS como en redes, bases de datos con SQL, reconocimiento y síntesis de voz, navegador, multimedia.
- **Aceptable 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 basados en OpenGL. Incorpora

codecs estándar más comunes de audio y vídeo, incluyendo H.264 (AVC), MP3, AAC, etc.

1.6.1.3. Comparativa con otras plataformas

Las plataformas comparadas y la versión que se ha utilizado como referencia se muestran a continuación:

	Apple iOS 7	Android 4.3	Windows Phone 8	BlackBerry OS 7	Symbian 9.5
Compañía	Apple	Open Handset Alliance	Microsoft	RIM	Symbian Foundation
Núcleo del SO	Mac OS X	Linux	Windows NT	Mobile OS	Mobile OS
Licencia de software	Propietaria	Software libre y abierto	Propietaria	Propietaria	Software libre
Año de lanzamiento	2007	2008	2010	2003	1997
Fabricante único	Sí	No	No	Sí	No
Variedad de dispositivos	modelo único	muy alta	media	baja	muy alta
Soporte memoria externa	No	Sí	Sí	Sí	Sí
Motor del navegador web	WebKit	WebKit	Pocket Internet Explorer	WebKit	WebKit
Soporte Flash	No	Sí	No	Sí	Sí
HTML5	Sí	Sí	Sí	Sí	No
Tienda de aplicaciones	App Store	Google Play	Windows Marketplace	BlackBerry App World	Ovi Store
Número de aplicaciones	825.000	850.000	160.000	100.000	70.000
Coste publicar	\$99 / año	\$25 una vez	\$99 / año	sin coste	\$1 una vez
Actualizaciones automáticas del S.O.	Sí	depende del fabricante	depende del fabricante	Sí	Sí
Familia CPU soportada	ARM	ARM, MIPS, Power, x86	ARM	ARM	ARM
Máquina virtual	No	Dalvik	.net	Java	No
Aplicaciones nativas	Siempre	Sí	Sí	No	Siempre
Lenguaje de programación	Objective-C, C++	Java, C++	C#, muchos	Java	C++
Plataforma de desarrollo	Mac	Windows, Mac, Linux	Windows	Windows, Mac	Windows, Mac, Linux

Tabla 1.1 Comparativa de las principales plataformas móviles [13]

Otro aspecto fundamental a la hora de comparar las plataformas móviles es su cuota de mercado. En la siguiente gráfica podemos ver un estudio realizado por la empresa Grartner Group, donde se muestra la evolución del mercado de los sistemas operativos para móviles según el número de terminales vendidos. Podemos destacar: el importante descenso de ventas de la plataforma Symbian de Nokia; el declive continuo de BlackBerry; como la plataforma de Windows que parece que no despegue; como Apple tiene afianzada una cuota de mercado en torno al 15%. Finalmente destacamos el espectacular ascenso

de la plataforma Android, que le ha permitido alcanzar en dos años una cuota de mercado superior al 75%. [13]

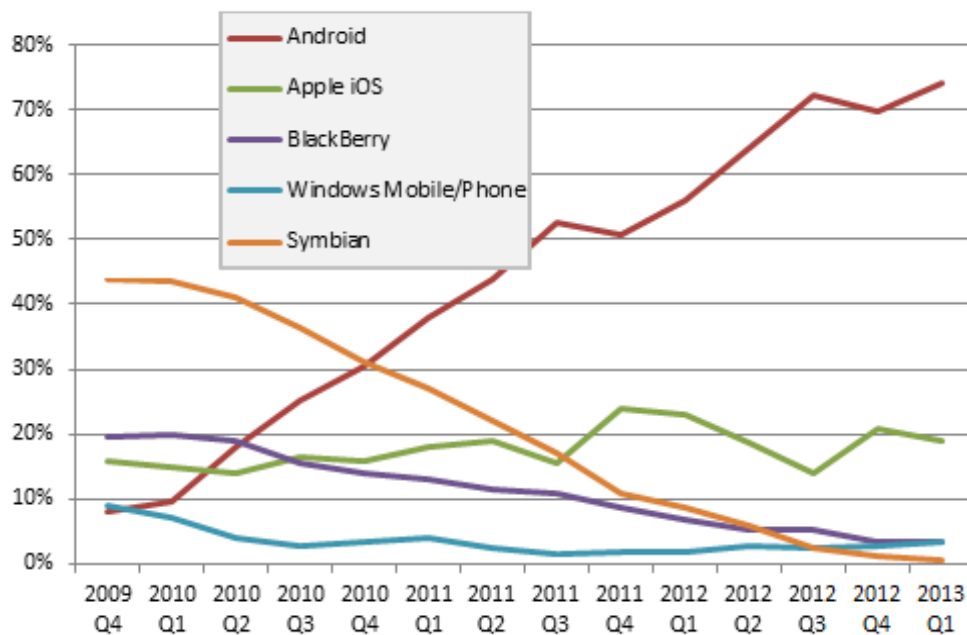


Tabla 1.2 Porcentaje de teléfonos inteligentes vendidos según su sistema operativo [13].

1.6.1.4. Arquitectura de Android

El siguiente gráfico muestra la arquitectura de Android. Como se puede ver está formada por cuatro capas. Una de las características más importantes es que todas las capas están basadas en software libre.

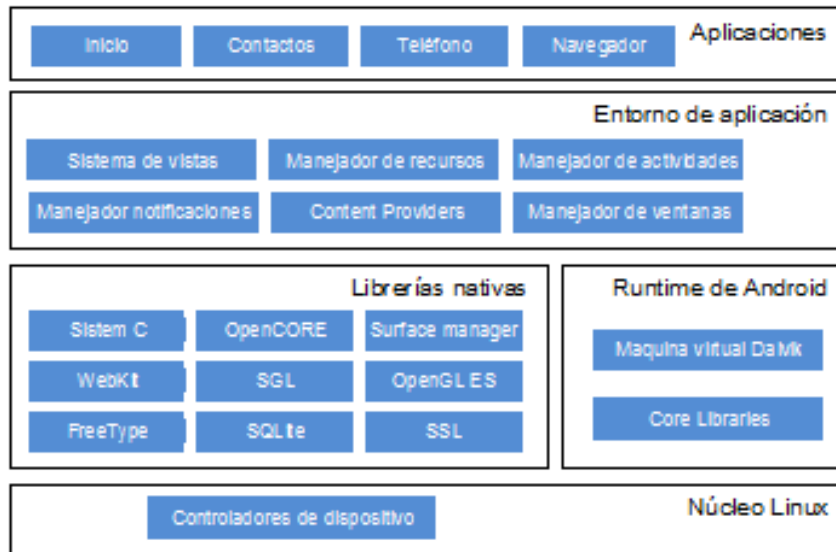


Figura 1.2 Arquitectura Android

1.6.1.4.1. El núcleo de 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.

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

1.6.1.4.2. El runtime de Android

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

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

También se incluye en el *Runtime de Android* el “core libraries” con la mayoría de las librerías disponibles en el lenguaje Java.

1.6.1.4.3. Librerías Nativas

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

- **System C library:** una derivación de la librería BSD de C estándar (libc), adaptada para dispositivos embebidos basados en Linux.
- **Media Framework:** librería basada en PacketVideo's OpenCORE; soporta codecs de reproducción y grabación de multitud de formatos de audio vídeo e imágenes MPEG4, H.264, MP3, AAC, AMR, JPG y PNG.
- **Surface Manager:** maneja el acceso al subsistema de representación gráfica en 2D y 3D.
- **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.
- **Librerías 3D:** implementación basada en OpenGL ES 1.0 API. Las librerías utilizan el acelerador hardware 3D si está disponible, o el software altamente optimizado de proyección 3D.
- **FreeType:** fuentes en bitmap y renderizado vectorial.
- **SQLite:** potente y ligero motor de bases de datos relacionales disponible para todas las aplicaciones.
- **SSL:** proporciona servicios de encriptación *Secure Socket Layer*.

1.6.1.4.4. Entorno de Aplicación

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

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 reemplazar componentes.

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

Los servicios más importantes que incluye son:

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

1.1.1.1.1.1. Aplicaciones

Este nivel está formado por el conjunto de aplicaciones instaladas en una máquina Android. Todas las aplicaciones han de correr en la máquina virtual Dalvik para garantizar la seguridad del sistema.

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

1.7. Bases de Datos

La manipulación directa de los datos entraña una complejidad importante, además de que, de esta forma, se corre el riesgo de realizar operaciones incorrectas o no deseadas con dichos datos. Por este motivo, se han desarrollado una serie de programas informáticos que tratan de aislar al usuario final de los ficheros de datos: son los llamados Sistemas Gestores de Bases de Datos (SGDB – DBMS, *Data Base Management Systems* -), programas que se encargaran de gestionar y controlar el Acceso a los datos ofreciendo una representación más sencilla de ellos.

Los Sistemas Gestores de Bases de Datos más conocidos y extendidos hoy en día son Access de Microsoft, base, FileMaker, y Paradox en el segmento de aplicaciones para particulares y pequeñas empresas y SQL Server de Microsoft, Oracle, DB2, de IBM, Informix y Sybase, en el segmento de las medianas y grandes bases de Datos.

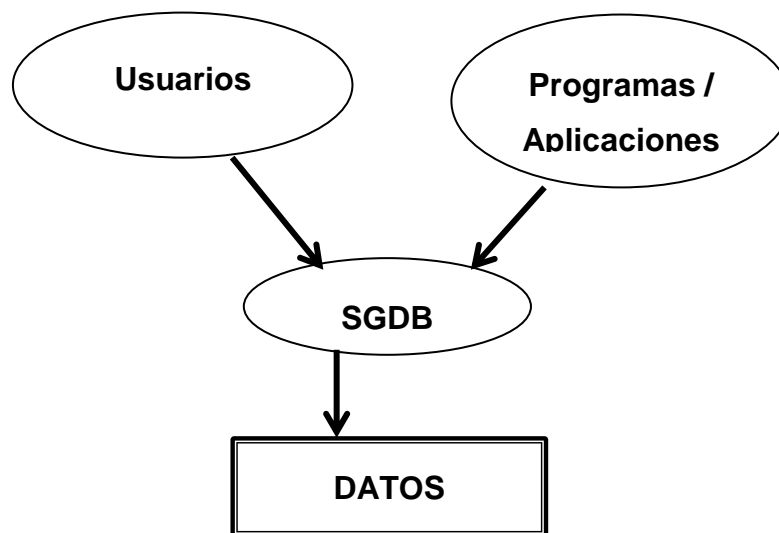


Figura 1.3 Sistema Gestor de Base de datos (SGDB). [5]

Podemos definir una base de datos como “un conjunto estructurado de datos que se guardan en un sistema informático y sobre los cuales es posible efectuar una serie de operaciones básicas de consulta, modificación, inserción o eliminación”.

Con objeto de acelerar la búsqueda de información, se han ideado distintos métodos de acceso rápido a los ficheros, siendo el más extendido el basado en la utilización de “índices”. Básicamente, un índice se puede construir mediante un fichero auxiliar que permite localizar donde se encuentra cada uno de los registros del fichero auxiliar de datos.

De esta forma, se dota de un contenido semántico a la estructura de la base de datos. Además, hay que tener en cuenta que el SGDB reserva para cada tipo de datos un determinado espacio físico en el dispositivo de almacenamiento.

Una característica adicional de los tipos de datos es que permiten al SGDB controlar la inserción o modificación de datos, de manera que este se va encargar de detectar e impedir que se introduzcan valores inapropiados en un determinado campo de un registro.

El “modelo relacional”, propuesto por E. F. Codd, es el más utilizado por las modernas bases de datos, ya que cuenta con una sólida base conceptual lógico-matemática y resulta muy sencillo y potente a la vez. En este modelo los ficheros de datos se representan de manera abstracta mediante tablas de valores.

El modelo relacional no se aplicó a la práctica hasta que, a finales de los años setenta, las minicomputadoras y las grandes computadoras empezaron a disponer de suficiente capacidad de procesamiento para el desarrollo de grandes bases de datos. IBM fue pionero en estas experiencias y desarrollo un lenguaje de consulta denominado SQL (del inglés Structure Query Language) que, a la postre, ha llegado a convertirse en el estándar de las bases de datos relacionales.

En un sistema relacional los datos se relacionan por sus valores y no mediante punteros, en el modelo relacional, por lo tanto, las filas de una determinada tabla constituyen los registros guardados en el fichero de datos y, a su vez, las columnas representan los campos o atributos de dichos registros.

La “Clave Candidata” es el atributo o conjunto de atributos que permite identificar de manera unívoca a un registro de la tabla (en la tabla no pueden existir dos o más registros en los cuales ese atributo o conjunto de atributos tenga el mismo valor). Por su parte, la

“Clave Primaria” es la clave que se utiliza para identificar de manera univoca a un registro (es la escogida entre las posibles claves candidatas).

Con los datos de las tablas es posible realizar múltiples tipos de operaciones, conocidas por el nombre genérico de consultas. La característica que confiere una mayor potencia y flexibilidad al modelo relacional es la posibilidad de establecer relaciones entre dos o más tablas por medio de determinados campos clave.

De este modo, es posible realizar operaciones de cruce de datos entre tablas, así como llevar a cabo “uniones de tablas” (joins), obteniendo nuevas tablas cuyos registros se forman al combinar los datos de las nuevas tablas de partida. La relación de los datos se establece mediante “Claves Foráneas”.

1.7.1. Comparativa de los SGBD más utilizados

	CARACTERISTICAS	VENTAJAS	DESVENTAJAS
Oracle	<ul style="list-style-type: none"> - Entorno de cliente / Servidor - Gestión de grandes bases de datos - Usuarios concurrentes - Alto rendimiento - Gestión segura - Portabilidad - Compatibilidad 	<ul style="list-style-type: none"> - Es muy usado a nivel Mundial - Puede ejecutarse en todas las plataformas - Permite el uso de particiones para mejorar su eficiencia - Un aceptable soporte 	<ul style="list-style-type: none"> - Muy costoso - Mal configurado es extremadamente lento
SQL Server	<ul style="list-style-type: none"> - Nuevas herramientas integradas - Recuperación rápida - Mejoras en la recopilación - Aislamiento de imágenes 	<ul style="list-style-type: none"> - Soporte de transacciones - Estabilidad, escalabilidad y seguridad. - Soporta procedimientos almacenados - Incluye un potente entorno grafico 	<ul style="list-style-type: none"> - Utiliza mucha memoria RAM - No es gratuito - No es muy útil a la hora de las prácticas.
My SQL	<ul style="list-style-type: none"> - Interioridad y portabilidad - Escrito en C y C++ 	<ul style="list-style-type: none"> - Es de código abierto 	<ul style="list-style-type: none"> - El soporte para disparadores es

- Probado con un alto rango de compiladores diferentes	- Bajo costo	muy básico
- Funciona en diferentes plataformas	- Facilidad de configuración	- El soporte para disparadores es muy básico
- Proporciona un buen almacenamiento	- Usa licencia GPL	- No soporta algunas conversiones de datos
- Relativamente sencillo	- Velocidad al realizar las operaciones.	- Los privilegios de las tablas no se borran automáticamente

Tabla 1.3 Comparativa de los SGBD más utilizados

1.8. Lenguajes y Entornos de Programación

Un lenguaje de programación está constituido por una serie de instrucciones, de operadores y de reglas que permiten controlar el hardware de un equipo informático para que realice determinado proceso o función. Se trata de la herramienta básica para el desarrollo del software, entendiendo como tal el conjunto de aplicaciones y programas que se van ejecutar en un sistema informático.

A lo largo de la historia, se pueden distinguir varias generaciones de lenguajes de programación, en función a su nivel de abstracción y de su dependencia de la arquitectura hardware de la máquina sobre la cual se van a ejecutar las aplicaciones desarrolladas.

El primer tipo de lenguaje de programación es el código máquina, totalmente dependiente del conjunto de instrucciones del ordenador en que se va a ejecutar y en el que la programación se debe realizar en el sistema binario, codificando mediante ceros y unos las instrucciones y los datos a procesar. Se trata de la primera generación de lenguajes, muy difícil y engorrosa de utilizar y totalmente dependiente de la arquitectura del equipo (hardware).

El segundo tipo de lenguaje de programación es el conocido como Lenguaje Ensamblador, que represento en su día un importante avance al utilizar instrucciones y etiquetas que eran posteriormente traducidas por las correspondientes secuencias de ceros y unos (Código Máquina), que es lo que, en definitiva, entiende el ordenador.

Con la aparición de los lenguajes de Alto nivel, surge una nueva generación, más orientada a la resolución general de operaciones, con independencia de la máquina en la que se realice tal operación, y que incorporan un nuevo nivel de abstracción. Con ello, se facilita enormemente la programación, al utilizar un conjunto de expresiones y operaciones más amigables y próximos al lenguaje natural, que equivalen a varias decenas de instrucciones básicas en código máquina.

De este modo, se mejora la legibilidad del código fuente de los programas y se reduce su tamaño, con la ventaja añadida que supone su portabilidad, es decir, que con un mismo programa de alto nivel pueda ser traducido al código máquina de distintos ordenadores. En estos lenguajes procedimentales de tercera generación, como Basic, C, Fortran, Cobol, Pascal, Etc., el programador debe indicar el algoritmo con la secuencia de comandos que debe ejecutar el ordenador para cumplir con las distintas funcionalidades de la aplicación.

Los entornos de programación cuentan con herramientas que posibilitan y facilitan la labor de programación, entre las cuales podemos citar:

- **Editor:** es la herramienta que permite escribir y revisar el código fuente, de acuerdo con el léxico (conjunto de instrucciones) y las reglas sintácticas del lenguaje de programación utilizado.
- **Depurador:** se trata de una herramienta que analiza el código generado para detectar errores lógicos o de sintaxis, así como posibles fallos en la aplicación, para eliminar errores en la codificación del programa.
- **Compilador:** los “lenguajes interpretados” van traduciendo las instrucciones a código máquina a medida que se va ejecutando la aplicación. A su vez, los “lenguajes compilados” son traducidos a código máquina una sola vez (al finalizar la creación del programa), obteniendo así un programa ejecutable directamente en la máquina. Esta labor es realizada por el compilador y proporciona una aplicación más rápida y eficiente, por cuanto no tiene que ser traducida cada vez que se va a ejecutar en el ordenador.
- **Enlazador:** es el encargado de enlazar el código máquina obtenido del compilador con las distintas librerías del sistema operativo. Estas librerías facilitan una serie de

funciones básicas sobre las que se puede apoyar el programador para simplificar la creación de sus aplicaciones.

1.8.1. PHP

PHP es un lenguaje de alto que se ejecuta en el servidor. PHP (Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.

Un lenguaje de servidor es aquel que se ejecuta en el servidor donde están alojadas las páginas, al contrario que otros lenguajes que son ejecutados en el propio navegador.

1.8.1.1. Ventajas

La principal ventaja es que, al ejecutarse el código en el servidor, todas las páginas van a poder ser vistas en cualquier ordenador, independientemente del navegador que tenga. En cambio, el gran problema de que se ejecute el código en el navegador es que muchos navegadores no son capaces de entender, lo que presentaría errores al mostrar el resultado de las páginas.

Otras ventajas que presenta el lenguaje PHP, es que se trata de un lenguaje de programación gratuito y, por tanto, todo el mundo puede utilizarlo sin ningún coste, frente a otros lenguajes cuyo software es necesario comprar para su utilización.

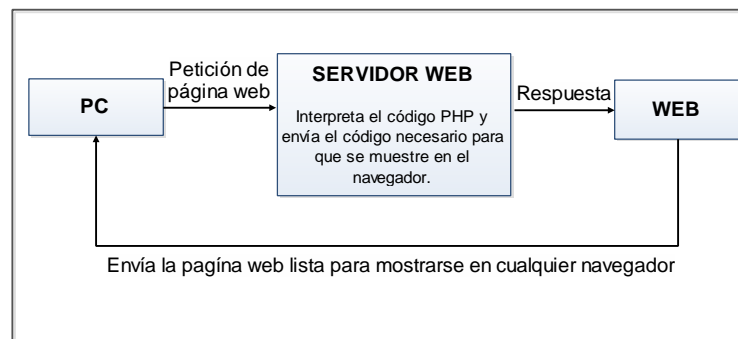


Figura 1.4 Proceso de visita a página PHP

Existen principalmente tres campos principales donde se usan scripts de PHP.

- **Scripts del lado del servidor:**

Este es el campo más tradicional y el foco principal. Se necesitan tres cosas para que esto funcione. El analizador de PHP (módulo CGI o servidor), un servidor web y un navegador web. Es necesario ejecutar el servidor, con una instalación de PHP conectada. Se puede acceder al resultado del programa PHP con un navegador, viendo la página de PHP a través del servidor. Todo esto se puede ejecutar en su máquina si está experimentado con la programación de PHP.

- **Scripts desde la línea de comando:**

Se puede crear un script de PHP y ejecutarlo sin necesidad de un servidor o navegador. Solamente es necesario el analizador de PHP para utilizarlo de esta manera. Este tipo de uso es ideal para scripts ejecutados regularmente usando cron (en Unix o Linux) o el Planificador de tareas (en Windows). Estos scripts también pueden usarse para tareas simples de procesamiento de texto.

Características

Probablemente PHP no sea el lenguaje más apropiado para crear aplicaciones de escritorio con una interfaz gráfica de usuario, pero si se conoce bien PHP, y se quisiera utilizar algunas características avanzadas de PHP en aplicaciones del lado del cliente, se puede utilizar PHP-GTK para escribir dichos programas. También es posible de esta manera escribir aplicaciones independientes de una plataforma. PHP-GTK es una extensión de PHP, no disponible en la distribución principal. Si está interesado en PHP-GTK, puede visitar su » propio sitio web.

PHP puede usarse en todos los principales sistemas operativos, incluyendo Linux, muchas variantes de Unix (incluyendo HP-UX, Solaris y OpenBSD), Microsoft Windows, Mac OS X, RISC OS y probablemente otros más. PHP admite la mayoría de servidores web de hoy en día, incluyendo Apache, IIS, y muchos otros. Esto incluye cualquier servidor web que pueda utilizar el binario de PHP FastCGI, como lighttpd y nginx. PHP funciona tanto como módulo como procesador de CGI.

De modo que con PHP se tiene la libertad de elegir el sistema operativo y el servidor web. Además, se tiene la posibilidad de utilizar programación por procedimientos o programación orientada a objetos (POO), o una mezcla de ambas.

Con PHP no se está limitado a generar HTML. Entre las capacidades de PHP se incluyen la creación de imágenes, ficheros PDF e incluso películas Flash (usando libswf y Ming) generadas sobre la marcha. También se puede generar fácilmente cualquier tipo de texto, como XHTML y cualquier otro tipo de fichero XML. PHP puede autogenerar estos ficheros y guardarlos en el sistema de ficheros en vez de imprimirlos en pantalla, creando una caché en el lado del servidor para contenido dinámico.

Una de las características más potentes y destacables de PHP es su soporte para un amplio abanico de bases de datos. Escribir una página web con acceso a una base de datos es increíblemente simple utilizando una de las extensiones específicas de bases de datos (por ejemplo para mysql), o utilizar una capa de abstracción como PDO, o conectarse a cualquier base de datos que admita el estándar de Conexión Abierta a Bases de Datos por medio de la extensión ODBC. Otras bases de datos podrían utilizar cURL o sockets, como lo hace CouchDB.

PHP también cuenta con soporte para comunicarse con otros servicios usando protocolos tales como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (en Windows) y muchos otros. También se pueden crear sockets de red puros e interactuar usando cualquier otro protocolo.

PHP tiene soporte para el intercambio de datos complejos de WDDX entre virtualmente todos los lenguajes de programación web. Y hablando de interconexión, PHP posee soporte para la instalación de objetos Java y usarlos de forma transparente como objetos de PHP.

1.8.2. Java

Lenguaje de programación de alto nivel con el que se pueden escribir tanto programas convencionales para PC y dispositivos móviles como para Internet.

Una de las ventajas significativas de Java sobre otros lenguajes de programación es que es independiente de la plataforma, tanto en código fuente como en binario. Esto quiere decir que el código producido por el compilador Java puede transportarse a cualquier plataforma que tenga instalada una máquina virtual Java y ejecutarse.

Java incluye un compilador y un intérprete. El compilador produce un código de bytes que se almacena en un fichero para ser ejecutado por el intérprete Java denominado máquina virtual de Java.[12]



Figura 1.5 Proceso de ejecución de un programa en Java

1.9. Acceso a Internet para conexiones móviles

Se encuentran las conexiones para teléfonos móviles en las que el móvil actúa como módem y se puede elegir la manera de conexión, como podría ser usando el sistema GSM (Global System Mobile) que emplea ondas de radio como medio de transmisión y cuando establece una conexión reserva la línea, ocupando parte del ancho de banda del operador de 9.6 Kbps. El sistema GPRS (General Packet Radio Service) que es la evolución del GSM, el cual ocupa una conexión por paquetes similar al usado en Internet. Este sistema está más orientado y mejor adaptado al tráfico de datos que el sistema GSM y es facturado por cantidad de datos enviados y recibidos y no por el tiempo de conexión a comparación de GSM.

Los sistemas GSM y GPRS se consideran de segunda generación (2G).

El UMTS (Universal Mobile Telecommunications System) inaugura la tercera generación de tecnología para móviles (3G). Permite velocidades de transferencia mucho mayores que GSM y GPRS, llegando hasta los 2 Mbps, permitiendo así el uso de aplicaciones que hasta ahora parecían imposibles en un móvil.

Una mejora del UMTS es el HSDPA (High Speed Downlink Packet Access), que llega a alcanzar los 14 Mbps de velocidad de transferencia. Existe ya una mejora comercializada

de este sistema, HSDPA+, que permite (teóricamente) llegar a los 80 Mbps de transferencia.

1.9.1. Los paquetes de información

En Internet la información se transmite en pequeños trozos llamados “paquetes”. Lo importante es la reconstrucción en el destino del mensaje emitido, no el camino seguido por los paquetes que lo componen.

Si se destruye un nodo de la red, los paquetes encontrarán caminos alternativos. Este procedimiento no es el más eficiente, pero resiste las averías de una parte de la red.

1.9.2. Servicio de Nombres

Existe un servicio que se encarga de proporcionar la correspondencia entre una dirección IP y su nombre de dominio, y viceversa. Este servicio es el DNS (Domain Name System, Sistema de Nombres de Dominio).

Cada vez que se inicia una comunicación con un nombre de dominio, la computadora realiza una petición a su servidor DNS para que le proporcione la IP asociada a ese nombre.

El sistema DNS es jerárquico. Cada subdominio de Internet suele tener su propio servidor DNS, responsable de los nombres bajo su dominio. A su vez, hay un servidor encargado de cada dominio (por ejemplo un nivel nacional (.mx)), y hay una serie de servidores raíz, que conocen toda la estructura DNS superior.

1.10. La arquitectura Cliente/Servidor

En el ámbito de TCP/IP las comunicaciones entre computadoras se rigen básicamente por lo que se llama modelo Cliente-Servidor, éste es un modelo que intenta proveer usabilidad, flexibilidad, interoperabilidad y escalabilidad en las comunicaciones. El término Cliente/Servidor fue usado por primera vez en 1980 para referirse a PC's en red.

Este modelo Cliente/Servidor empezó a ser aceptado a finales de los 80's.[4]. Su funcionamiento es sencillo: se tiene una máquina cliente, que requiere un servicio de una máquina servidor, y éste realiza la función para la que está programado.

Desde el punto de vista funcional, se puede definir el modelo Cliente/Servidor como una arquitectura distribuida que permite a los usuarios finales obtener acceso a la información en forma transparente [4].

En el modelo cliente servidor, el cliente envía un mensaje solicitando un determinado servicio a un servidor (hace una petición), y este envía uno o varios mensajes con la respuesta (provee el servicio) (Ver Figura 1.6). En un sistema distribuido cada máquina puede cumplir el rol de servidor para algunas tareas y el rol de cliente para otras.

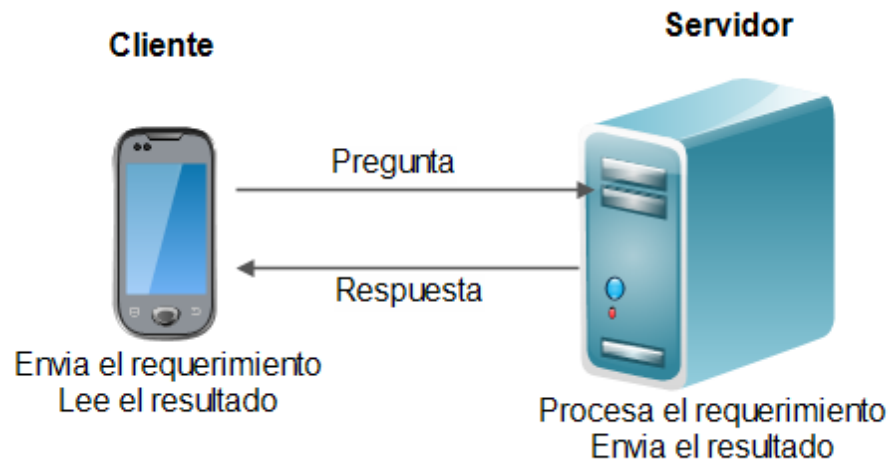


Figura 1.6 Modelo Cliente/Servidor

De esta manera lo que plantea es que una computadora pueda realizar múltiples tareas por si sola.

1.10.1. Cliente

El cliente es el proceso que permite al usuario formular las solicitudes y pasarlos al servidor, se le conoce con el término *front-end* [4].

El cliente normalmente maneja todas las funciones relacionadas con la manipulación y despliegue de datos, por lo que están desarrollados sobre plataformas que permiten construir interfaces gráficas de usuario (GUI), además de acceder a los servicios distribuidos en cualquier parte de una red.

Las funciones que lleva a cabo el proceso cliente se resumen en los siguientes puntos:

- Administrar la interfaz de usuario
- Interactuar con el usuario
- Procesar la lógica de la aplicación y hacer validaciones locales
- Generar requerimientos de bases de datos
- Recibir resultados del servidor
- Formatear resultados

1.10.2. Servidor

Es el proceso encargado de atender múltiples clientes que hacen peticiones de algún recurso administrado por él. Al proceso servidor se le conoce con el término *back-end*[4].

El servidor normalmente maneja todas las funciones relacionadas con la mayoría de las reglas del negocio y los recursos de datos.

Las funciones que lleva a cabo el proceso servidor se resumen en los siguientes puntos:

- Aceptar los requerimientos de bases de datos que hacen los clientes.
- Procesar requerimientos de bases de datos.
- Formatear datos para transmitirlos a los clientes.
- Procesar la lógica de la aplicación y realizar validaciones a nivel de bases de datos.

1.10.2.1. Tipos de servidores

Servidor de acceso a base datos: En un ambiente cliente/servidor, el proceso de aplicación se divide entre los sistemas cliente y servidor. Los servidores pueden ejecutar

una parte de la lógica de negocio (Business logic) y la lógica de base de datos. Al igual que en los servidores de ficheros, los servidores de bases de datos ofrecen a los clientes el acceso a los datos que residen en el servidor. Sin embargo, los sistemas de gestión de bases de datos son más sofisticados que los métodos de acceso de E / S del fichero de básico.

Servidor web: Provee de contenidos estáticos a los navegadores. Este le envía los ficheros que carga por medio de la red al navegador del usuario. Los ficheros pueden ser imágenes, escrituras, documentos HTML y cualquier otro material web.

Servidor de aplicaciones: Proporciona servicios que soportan la ejecución y disponibilidad de las aplicaciones desplegadas. Es el corazón de un gran sistema distribuido.

1.11. **Arquitectura de una Aplicación Web**

La arquitectura de las aplicaciones web suele presentar un esquema de tres niveles:

- El primer nivel consiste en la capa de presentación que incluye no sólo el navegador, sino también el servidor web que es el responsable de presentar los datos un formato adecuado.
- El segundo nivel está referido habitualmente a algún tipo de programa o script.
- Finalmente, el tercer nivel proporciona al segundo los datos necesarios para su ejecución. Una aplicación Web típica recogerá datos del usuario (primer nivel), los enviará al servidor, que ejecutará un programa (segundo y tercer nivel) y cuyo resultado será formateado y presentado al usuario en el navegador (primer nivel otra vez).



Figura 1.7 Arquitectura a tres capas (Aplicación Web)

1.12. Servicios WEB

Un servicio web es un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de computadoras como la Internet.

La interoperabilidad se consigue mediante adaptación de estándares abiertos. Las organizaciones OASIS y W3C son los comités responsables de la arquitectura y reglamentación de los servicios Web. Para mejorar la interoperabilidad entre distintas implementaciones de servicios Web se ha creado el organismo WS-I, encargado de desarrollar diversos perfiles para definir de manera más exhaustiva estos estándares.

1.12.1. HTTP

Desde 1990, el protocolo HTTP (Protocolo de transferencia de hipertexto) es el protocolo más utilizado en Internet. Permite la transferencia de mensajes con encabezados que describen el contenido de los mensajes mediante la codificación MIME -extensiones multipropósito de correo en Internet.

Se utiliza en las transferencias de información de páginas en Internet, de tal forma que puedan ser visualizadas en un navegador o explorador; habitualmente comprenderá, entre

otros elementos, textos en lenguaje HTML, imágenes, Applets de Java, datos, documentos de diversos tipos, animaciones y elementos multimedia.

HTTP es el protocolo de transferencia de información que forma la base de la colección de información distribuida denominada World Wide Web. El protocolo HTTP no fija exactamente lo que se envía o cómo está programado, sólo se refiere al mecanismo empleado para hacer llegar y recibir dicha información entre los servidores y el usuario final. Por tanto, controlará el mecanismo de comunicación entre los servidores. Debido a la gran evolución de Internet, se están estudiando alternativas más ágiles al HTTP original, como pueden ser el HTTP-NG (HyperText Transport Protocol-Next Generation).

1.12.1.1. Métodos de Petición HTTP

Tanto GET como POST, justamente por ser métodos ambos de HTTP, ejecutan un request y response.

- a) **El Método GET:** es obtener información del servidor. Traer datos que están en el servidor, ya sea en un fichero o base de datos, al cliente. Independientemente de que para eso tengamos que enviar (request) algún dato que será procesado para luego devolver la respuesta (response) que esperamos, como por ejemplo un identificador para obtener una noticia de la base de datos.

Algunas otras notas sobre las peticiones GET:

Pueden almacenar en caché.

Permanecen en el historial del navegador.

Se pueden marcar.

Nunca deben ser utilizados cuando se trata de datos sensibles.

Recibimos solicitudes tienen restricciones de longitud.

Solo deben utilizarse para recuperar datos.

- b) **El Método POST:** es enviar información desde el cliente para que sea procesada y actualice o agregue información en el servidor, como sería la carga o actualización

en sí de una noticia. Cuando enviamos (request) datos a través de un formulario, estos son procesados y luego a través de una redirección por ejemplo devolvemos (response) alguna página con información.

Algunas otras notas sobre las peticiones POST:

- Nunca están en caché.
- Permanecen en el historial del navegador.
- No se pueden marcar.
- No tienen restricciones en la longitud de datos.

	GET	POST
En cache	Guardada en Caché	No guardada en Caché
Tipo de codificación	application/x-www-form-urlencoded	application/x-www-form-urlencoded or multipart/form-data Utiliza la codificación multiparte para datos binarios
Historial del Navegador	Los parámetros se guardan en el historial del navegador	Los parámetros no son guardados en el historial del navegador

Restricciones a la longitud de los datos	Sí, cuando el envío de datos, el método GET añade los datos a la dirección URL, y la longitud de una URL es limitada (longitud máxima de la URL es de 2048 caracteres)	Sin restricciones
Restricciones al tipo de Datos	Solo son permitidos caracteres ASCII	Sin restricciones. Datos binarios son también permitidos
Seguridad	Consigo es menos seguro en comparación a POST ya que los datos enviados son parte de la URL. Nunca utilice GET al enviar contraseñas u otra información sensible	POST es un poco más seguro de conseguir porque los parámetros no se almacenan en el historial del navegador o en los registros del servidor web
Visibilidad de los datos	Los datos son vistos desde la URL	Los datos no son mostrados en la URL

Tabla 1.4 Comparativa GET vs POST

Ambos métodos solicitan una respuesta del servidor y ahí es donde parece que los conceptos son iguales ya que con ambos se podría lograr los mismos objetivos.

1.12.2. Tipos de notificaciones

1.12.2.1. Push

Una notificación push es, básicamente, un mensaje enviado por un servidor a un cliente que está “suscrito” a sus notificaciones.

La tecnología push, a nivel de infraestructura, puede implementarse en prácticamente todos los servidores. ¿Cómo funcionan este tipo de mensajes? Hay varias opciones.

Por una parte, el servidor mantiene la conexión abierta, de manera que tan pronto el servidor reciba un determinado evento, puede enviar la información correspondiente al cliente. De haberse cerrado la conexión, el servidor necesitaría mantener una cola para enviar la información tan pronto el cliente vuelva a conectarse.

Debemos darnos cuenta de que estamos hablando siempre a un nivel muy alto: realmente, en este contexto, nos da igual qué tipo de paquetes se envíen, qué tipo de protocolo se use para realizar la comunicación y qué tipo de red haya por debajo. Únicamente importa que se envíe el mensaje, y que el mensaje se reciba correctamente.

1.12.2.2. **Pull**

El servidor envía información bajo demanda. Es el cliente el que inicia la acción.

Se utiliza la tecnología pull (tirar) cuando se navega por el World Wide Web para buscar y descargar información en el ordenador. Esto contrasta con la tecnología push (empujar), cuando los datos son entregados directamente al ordenador del usuario.

1.13. **Sistema de Posicionamiento Global**

El Sistema de Posicionamiento Global (de sus siglas en Inglés GPS Global Positioning System), es un sistema de navegación basado en satélites de comunicación que fue desarrollado por el Departamento de Defensa Americano (DoD) a principios de los 70's. Inicialmente fue desarrollado para cumplir las necesidades de la Milicia de los Estados Unidos.

Sin embargo, más tarde fue permitido su uso a personas civiles, y es ahora un sistema de uso dual que puede ser utilizado tanto por militares como civiles. El GPS continuamente envía información de posición y tiempo en cualquier parte del mundo bajo cualquier condición climática.

El GPS es un sistema basado en satélites artificiales activos, formando una constelación con un mínimo de 24 satélites operacionales. La primera constelación inició operaciones en 1990.

Para llevar a cabo la cobertura continua mundial, los satélites del sistema GPS son agrupados en órbitas de 4 satélites, así forman un total de 6 órbitas.

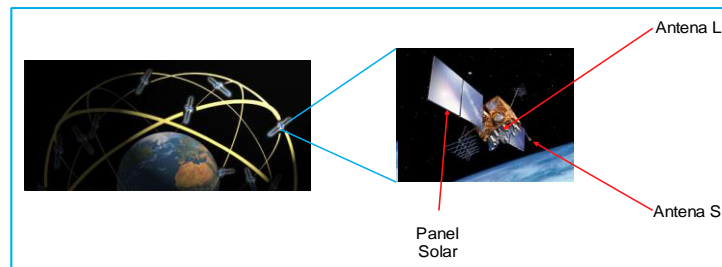


Figura 1.8 Constelaciones GPS

Con esta configuración, de 4 a 10 satélites serán visibles en cualquier parte del mundo, pero sólo 4 satélites son necesarios para determinar la información de la localización. Las órbitas de los satélites del GPS se asemejan a una forma circular (una forma elíptica con un máximo de excentricidad de 0.01) con una inclinación de 55° del ecuador. El semieje mayor de una órbita GPS es alrededor de 26,560 Km (por ejemplo la altura del satélite es de 23,000 Km. sobre la superficie terrestre).

El período correspondiente de la órbita GPS es de 12 horas, siderales aproximadamente (11 horas 58 minutos y 2 segundos).

1.13.1. La idea básica del GPS

La idea básica del GPS es simple, si la distancia de un punto de la Tierra (un receptor GPS) a 3 satélites son conocidas, y se conoce la localización de los satélites, entonces un punto (receptor GPS) puede ser determinado simplemente aplicando conceptos conocidos de intersección de esferas.

Cada satélite GPS continuamente transmite una señal de radio de microonda compuesta de 2 portadoras, 2 códigos y un mensaje de navegación. Cuando un receptor GPS es encendido, este recogerá la señal del satélite GPS a través de una antena. Una vez que el

receptor GPS adquiere la señal, éste la procesará usando el software incluido. El resultado parcial de este procesamiento de la señal consiste en la distancia del satélite a través del mensaje de navegación.

Teóricamente, solamente 3 distancias para 3 satélites rastreados son necesarias, en este caso el receptor sería localizado en la intersección de las 3 esferas; cada uno tiene un radio de distancia del receptor GPS al satélite y en el centro un satélite en particular (Figura 1.9).

Desde un punto de vista práctico, sin embargo, un cuarto satélite es necesario a considerar para compensar errores del reloj del receptor.

Para mejorar la precisión del posicionamiento GPS, se utiliza un método que es llamado el método diferencial, el cual emplea dos receptores simultáneamente rastreando los mismos satélites GPS- En este caso el nivel de precisión es del orden de centímetros a pocos metros.

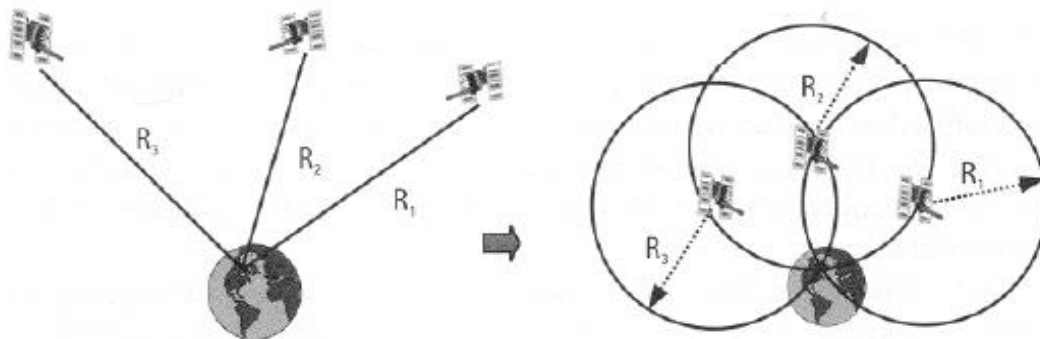


Figura 1.9 Posicionamiento Global idea básica

1.13.2. Concepto de Localización por medio del GPS

Cuando se conoce el tiempo que una señal es transmitida y se mide el tiempo que tarda la señal en ser recibida, un tiempo después, este intervalo es conocido como el tiempo de Arribo TDA.

Las posiciones de los GPS son controladas por el Segmento de Control (como se mencionó anteriormente), y cada satélite envía su posición en sus mensajes. El Segmento

de Control mantiene síncronos los tiempos de los satélites. Suponiendo que el receptor con antena tiene un reloj atómico síncrono también, y todos los satélites junto con el receptor están sincronizados. Si se sabe la velocidad de la luz (30 cm/ns); entonces se puede conocer el tiempo que le toma a la señal (línea recta) para llegar al receptor GPS. Si se conoce el tiempo que las señales (líneas rectas) de cada uno de los satélites ocupan para arribar al receptor GPS, entonces se puede calcular la longitud de cada línea recta, la cual es TDA veces la velocidad de la luz. Si las posiciones de los satélites son conocidas a un metro y los 4 relojes están sincronizados mejor que 3 ns, la posición del receptor GPS puede ser calculada con una precisión menor a un metro.

Sin embargo, el uso de un reloj atómico en el receptor GPS es muy costoso y espacioso. Como solución a este problema se utiliza el 4° satélite para calcular el tiempo, la altitud, latitud y longitud del receptor. Cada línea recta (vector) produce 4 ecuaciones y existen 4 incógnitas: x , y , z y t .



Figura 1.10 Localización con 4 satélites

Con ello el receptor sólo necesita un reloj de cuarzo estable, el cual puede ser fácilmente usado para medir las diferencias en tiempo de los TDA de cada una de las señales recibidas con una precisión de 1 ns.

El error de reloj para este receptor no caro puede ser calculado usando los datos de los 4 satélites. En otras palabras, existirán esferas rodeando a los satélites hacia el receptor GPS Ver Figura 1.10.

El radio de cada esfera será el de la distancia recorrida calculada. Para que esto suceda, el tiempo usado en el receptor debe ser síncrono al tiempo GPS, y de esta forma las 4 esferas se interceptarán y lograrán indicar la posición exacta.

1.13.3. Cálculos con coordenadas de GPS

Para resolver el problema de decidir si un punto en el plano esta sobre el triángulo, hacemos uso de algunas ideas del Algebra Lineal como es la Combinación Lineal de espacios vectoriales.

Teorema de la combinación Lineal [16]

Sean $\mathbf{V}_1, \mathbf{V}_2, \dots, \mathbf{V}_n$, vectores en un espacio vectorial V .

Entonces cualquier vector de la forma:

$$V = a_1V_1 + a_2V_2 + \dots + a_nV_n$$

Ecuación 1.1 General de la suma de vectores

Se llama combinación lineal de los Vectores $\mathbf{V}_1, \mathbf{V}_2, \dots, \mathbf{V}_n$.

Los escalares $\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n$. Se llaman coeficientes de **combinación lineal**.

Dependiendo de los valores que tomen los coeficientes de la combinación lineal, determinará que el vector resultante esté dentro de la combinación lineal. La forma de solucionar para cada valor de los coeficientes se hace con una resolución de un sistema de ecuaciones formado con los componentes de los vectores conocidos, o con una serie de sumas y restas de productos cruz entre ellos $\mathbf{V}, \mathbf{V}_1, \mathbf{V}_2, \dots, \mathbf{V}_n$. si los coeficientes son mayores que cero, se dice que \mathbf{V} es una combinación lineal de los vectores.

Para esto consideremos un ángulo con un vértice V_0 , como se muestra en la figura que sigue.

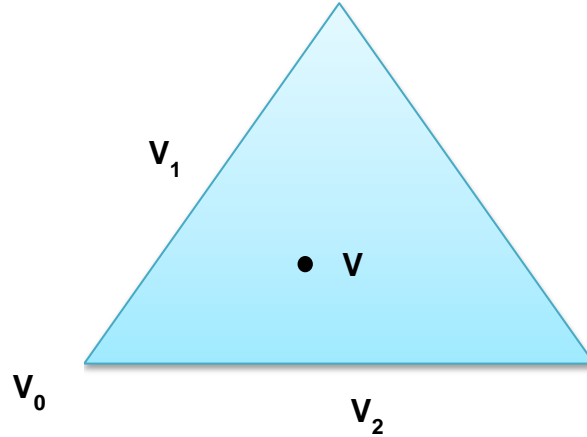


Figura 1.11 Triangulo imaginario

En esta figura, los vectores V_1 y V_2 son los rayos del ángulo con vértice en V_0 . Ahora tomamos un punto v y lo expresamos como:

$$V = V_0 + a_1 V_1 + a_2 V_2$$

Ecuación 1.2 Tomando en cuenta el vértice V_0

Resolviendo para a_1 y a_2 obtenemos:

$$a_1 = \frac{V \otimes V_2 - V_0 \otimes V_2}{V_1 \otimes V_2}$$

Ecuación 1.3 Para obtener el valor de a_1

$$a_2 = - \frac{V \otimes V_1 - V_0 \otimes V_1}{V_1 \otimes V_2}$$

Ecuación 1.4 Obtener valor de a_2

Dónde:

Sabemos que el producto cruz entre dos vectores de dos dimensiones es:

$$U \otimes V = U_x V_y - U_y V_x$$

Ecuación 1.5 Producto cruz de dos dimensiones

La fórmula definida para vectores en dos dimensiones $U \times V$, es una derivación del cálculo formal.

Para este caso si \mathbf{a}_1 y \mathbf{a}_2 son mayores que cero tenemos una combinación lineal de los Vectores V_1 y V_2 , esto quiere decir que el punto V está dentro del ángulo formado por los dos vectores.

1.14. Google Cloud Messaging para Android

Google Cloud Messaging para Android (GCM) es un servicio proporcionado por Google de manera gratuita para ayudar a los desarrolladores en el envío de datos desde sus servidores de aplicaciones Android a los dispositivos Android, y los mensajes serán transferidos de la nube al dispositivo del usuario.

A través de GCM se pueden enviar todo tipo de texto y datos, siempre y cuando no exceda los 4Kb por mensaje, sin tener un límite de mensajes.

1.14.1. Funcionamiento

Etapa 1- Identificación:

Identificar el dispositivo móvil cuyo primer paso es enviar una señal al servidor GCM que está en “la nube”, esta señal permite al móvil registrarse en ese servidor; si ese proceso es exitoso el servidor GCM envía una identificación para que tanto el servidor GCM como el servidor del desarrollador pueda determinar “quién es el dispositivo móvil”, como si fuera el CURP en México; por último el dispositivo contacta al servidor del desarrollador para ser incluido en la base de datos del desarrollador usando esa identificación.

Etapa 2 – Mensajería:

El servidor del desarrollador recibe de la base de datos la orden de generar un mensaje que se debe enviar a ciertos dispositivos, de esta forma un texto del mensaje es enviado al servidor GCM junto con la lista de dispositivos a los que debe enviarse, entonces es el servidor GCM el que se encarga de la distribución de los mensajes.

1.14.2. Características

- No es necesario que la aplicación se esté ejecutando para recibir mensajes. Ya que los servicios de Google son los encargados de recibir el mensaje y en ese momento el Sistema despierta la aplicación Android, siempre y cuando la aplicación sea configurada con el receptor de difusión adecuado y permisos.
- No incluye una interfaz de usuario. GCM únicamente pasa los datos del mensaje a la aplicación.
- El dispositivo debe contar con Android 2.2 o superior, ya que tiene instalado la suite de aplicaciones Google Store.
- Para los dispositivos con Sistema Android menor a 4.0.4 deben tener una cuenta de Google en sus dispositivos móviles, para posteriores no es necesario.

1.14.3. Arquitectura

Entes involucrados en GCM

Mobile Device	Dispositivo en el cual se está ejecutando el Sistema Operativo.
3ª Parte Servidor de Aplicación	Servidor implementado por los desarrolladores para el funcionamiento de la aplicación Android.
Servidores GCM	Servidor de Google que es intermediario entre el servidor de los desarrolladores y la aplicación Android.

Tabla 1.5 Entes involucrados en el proceso con GCM

Credenciales

Identificador de remitente	Se utiliza para identificar una aplicación de Android a la cual se le concedió el permiso para enviar y recibir mensajes.
Identificador de aplicación	La aplicación Android se identifica por el nombre del paquete. Asegurando que los mensajes están dirigidos a la aplicación Android correcta-
Identificador de Registro	Es un ID proporcionado por los servidores GCM a la aplicación Android para Android que le permite recibir mensajes
Cuenta de usuario Google	Es un requisito indispensable que el dispositivo cuente por lo menos con una cuenta de Google si el dispositivo está ejecutando una versión inferior a Android 4.0.4
Remitente autenticación Token	Es un requisito indispensable que el dispositivo cuente por lo menos con una cuenta de Google si el dispositivo está ejecutando una versión inferior a Android 4.0.4.
Clave Notificación	Es la señal que GCM utiliza para avivar las notificaciones a todos los dispositivos cuyo registro ID se asocian con la clave.
Notificación Nombre de Clave	Es un nombre o identificador (puede ser un nombre de usuario para una aplicación de 3ª parte) que es único para un usuario determinado. Se utiliza para agrupar los ID de registro para un solo usuario.

Tabla 1.6 Credenciales GCM

Capítulo 2 Diseño

2. Diseño del Sistema Móvil de notificación de proximidad para transporte.

2.1. Arquitectura propuesta para el sistema

2.1.1. Sistema de Información

Este Sistema de Información se puede clasificar dentro de los Sistemas de Información como Sistema de Apoyo a las Decisiones, debido a que nuestro sistema será:

- De apoyo a los usuarios en su proceso de toma de decisiones, dentro del contexto.
- Será intensivo en cálculos, realizando una gran cantidad de comparaciones con respecto a los datos almacenados en la base de datos de ubicación de las estaciones y los datos de localización del vehículo; y escaso en entradas y salidas, tomando como entradas la solicitud del usuario y la posición de la unidad de transporte y como salidas las notificaciones.
- Será un Sistema de Información interactivo y amigable con el usuario final, enfocándose al ser utilizado por personas de con conocimiento en el uso del Sistema Operativo Android.
- Apoyará a la toma de decisiones que, por su misma naturaleza son repetitivas y de decisiones no estructuradas que no suelen repetirse.

2.1.2. Hardware

2.1.2.1. El usuario

- Dispositivo móvil
- Conexión a Internet

2.1.2.2. El Vehículo

- Dispositivo Móvil
- GPS
- Conexión a Internet

2.1.3. Software

2.1.3.1. Sistema Operativo

2.1.3.1.1. Para desarrollo:

Linux Ubuntu 13.10,

Windows 7 Home Premium o Windows 8.1 OEM.

2.1.3.1.2. Para pruebas:

A partir de Android 2.3.3 Ginger Bread

- Google API: Google Cloud Messaging
- Android SDK Manager Revision 22.2.1 (Android Software Development Kit Manager).

Encapsula un conjunto de herramientas de desarrollo. Como es un depurador de código, biblioteca, un simulador de teléfono basado en QEMU("Quick EMUlator"), documentación, ejemplos de código y tutoriales.

- Android Developer Tools v22.2.1-833290

Plugin para Eclipse, para desarrollar aplicaciones de Android. Complementando con funciones avanzadas para ayudar a crear, probar, depurar y empaquetar aplicaciones Android.



- Eclipse Java Development Tools v 3.8.2

Entorno de Desarrollo para lenguaje Java.

- Eclipse Platform v 4.2.1.

Conjunto de marcos y servicios comunes, que en conjunto conforman la infraestructura necesaria para apoyar el uso de Eclipse.

- Eclipse RCP 4.2.2 (Plataforma de cliente enriquecido).
- MySQL 5.5.24
- PHP 5.3.13
- Apache 2.2.22
- GitHub

Software para alojar proyectos, haciendo uso del sistema de control de versiones Git. Utiliza el framework Ruby on Rails por GitHub.Inc.

El código se almacena de forma pública, o de manera privada, creando una cuenta de pago.

2.1.4. Metodología

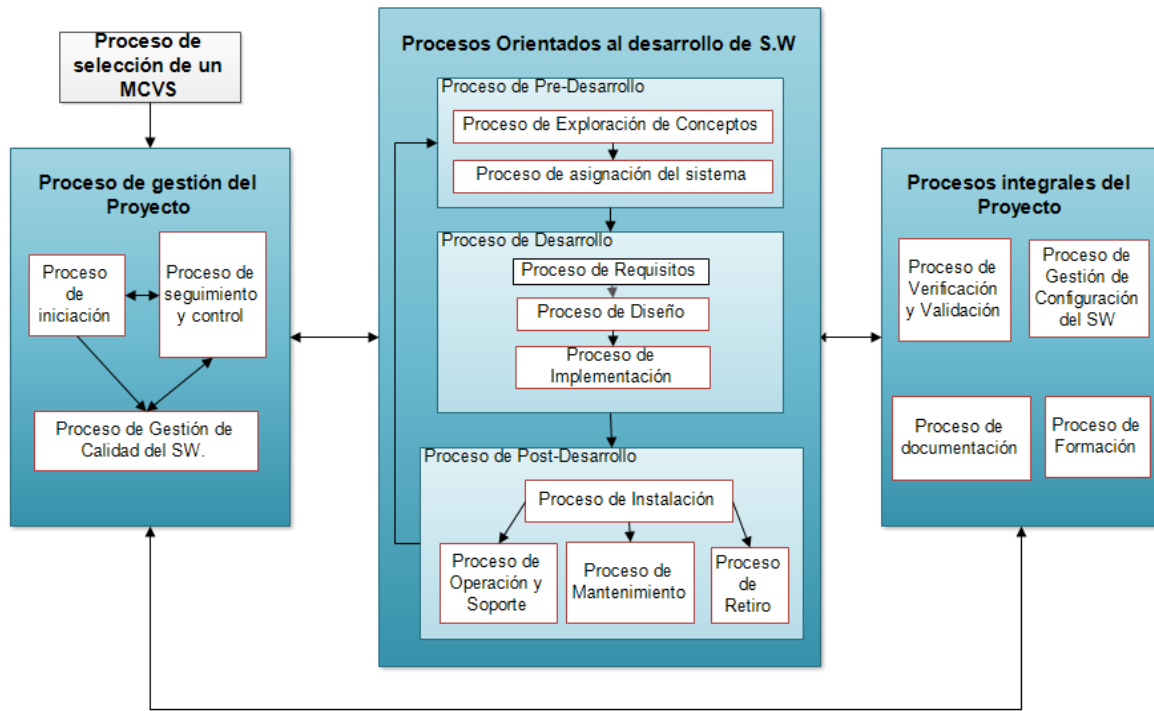


Figura 2.1 Modelo de procesos software IEEE

2.1.4.1. Ciclo de vida

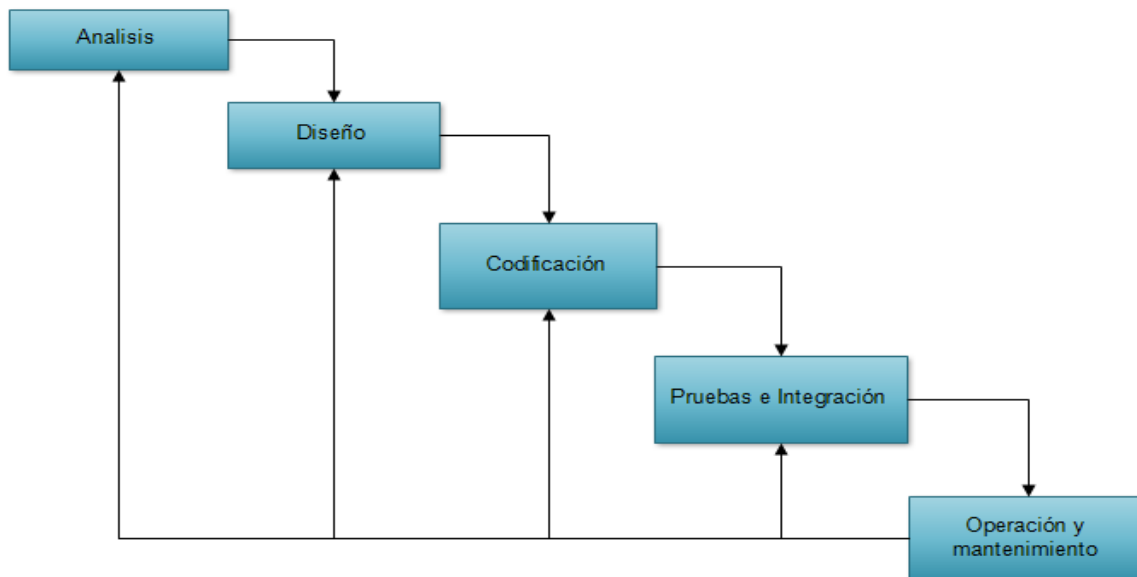


Figura 2.2 Ciclo de vida en cascada. Mejoramiento por pasos

Debido a que este es utilizado para proyectos pequeños y es muy versátil, ya que nos podemos mover entre etapas durante el desarrollo del mismo para poder hacer alguna mejora.

2.1.5. Estándares

2.1.5.1. ISO

Organización Internacional de Normalización. Es una federación mundial de organismos nacionales de normalización (organismos miembros de ISO). El trabajo de preparación de las normas internacionales normalmente se realiza a través de los comités técnicos de ISO. Cada organismo miembro interesado en una materia para la cual se haya establecido un comité técnico, tiene el derecho de estar representado en dicho comité. Las organizaciones internacionales, públicas y privadas, en coordinación con ISO, también

participan en el trabajo. ISO colabora estrechamente con la Comisión Electrotécnica Internacional (IEC) en todas las materias de normalización electrotécnica. La tarea principal de los comités técnicos es preparar Normas Internacionales. [Revisar Anexo]

2.2. Propuesta de solución

Desarrollar un sistema de notificación, implementado bajo el estándar ISO9126-1(2001), estándares de desarrollo para PHP y las convenciones de código para el lenguaje de programación Java (Code Conventions for the Java Programming Language)[20].

Con ayuda de herramientas para el diagramado y planificación como son Edraw Max 7 free versión, Microsoft Office Project 2010 y para el funcionamiento MySQL Server 5.0, Apache 5, HeidiSQL 8.2, Filezilla 3.7.4, Eclipse Kepler y Eclipse Juno con el SDK de Android y la una extensión de la biblioteca de soporte diseñado, ActionBarSherlock[21].

Utilizando GPS para proporcionar información sobre la proximidad de unidades de la red de transporte de pasajeros (anteriormente conocida como red de transporte público RTP) ruta Bosque Nativitas - Metro San Lázaro a los usuarios con dispositivos compatibles con el sistema GCM [22].

Siendo una herramienta en el día a día que brinde la posibilidad del análisis de otras alternativas de transporte, llámese taxi, metro, microbús, etc.

2.2.1. Diagrama General del Sistema

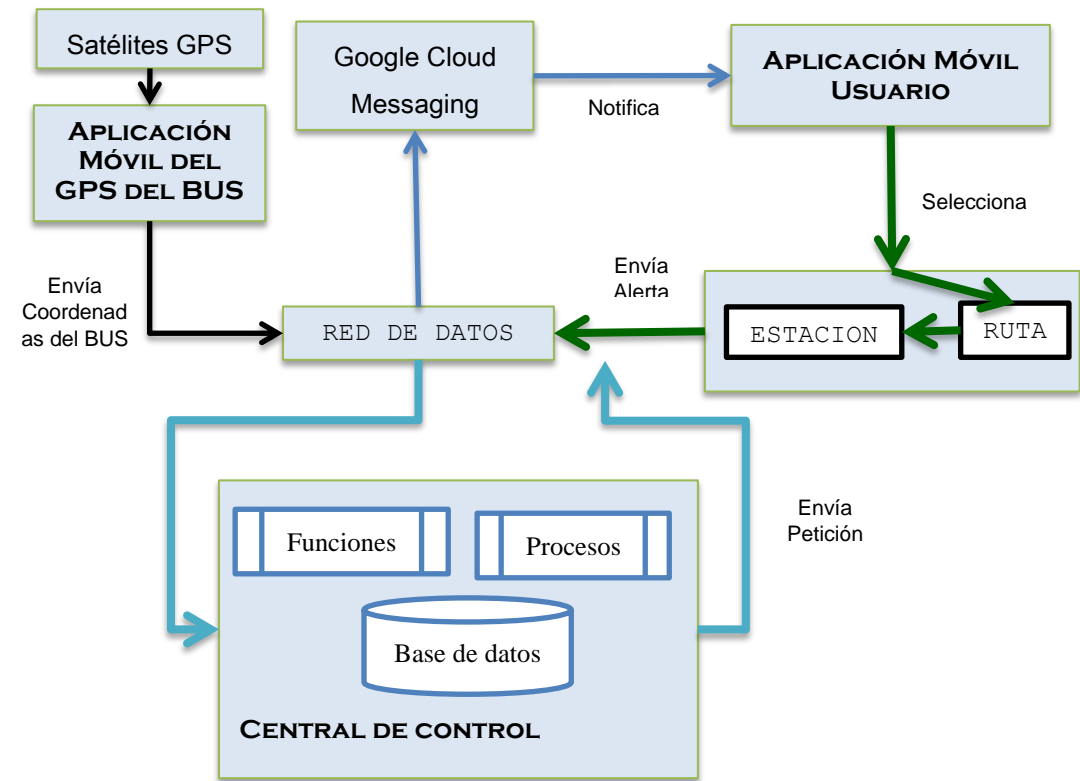


Figura 2.3 Diagrama General

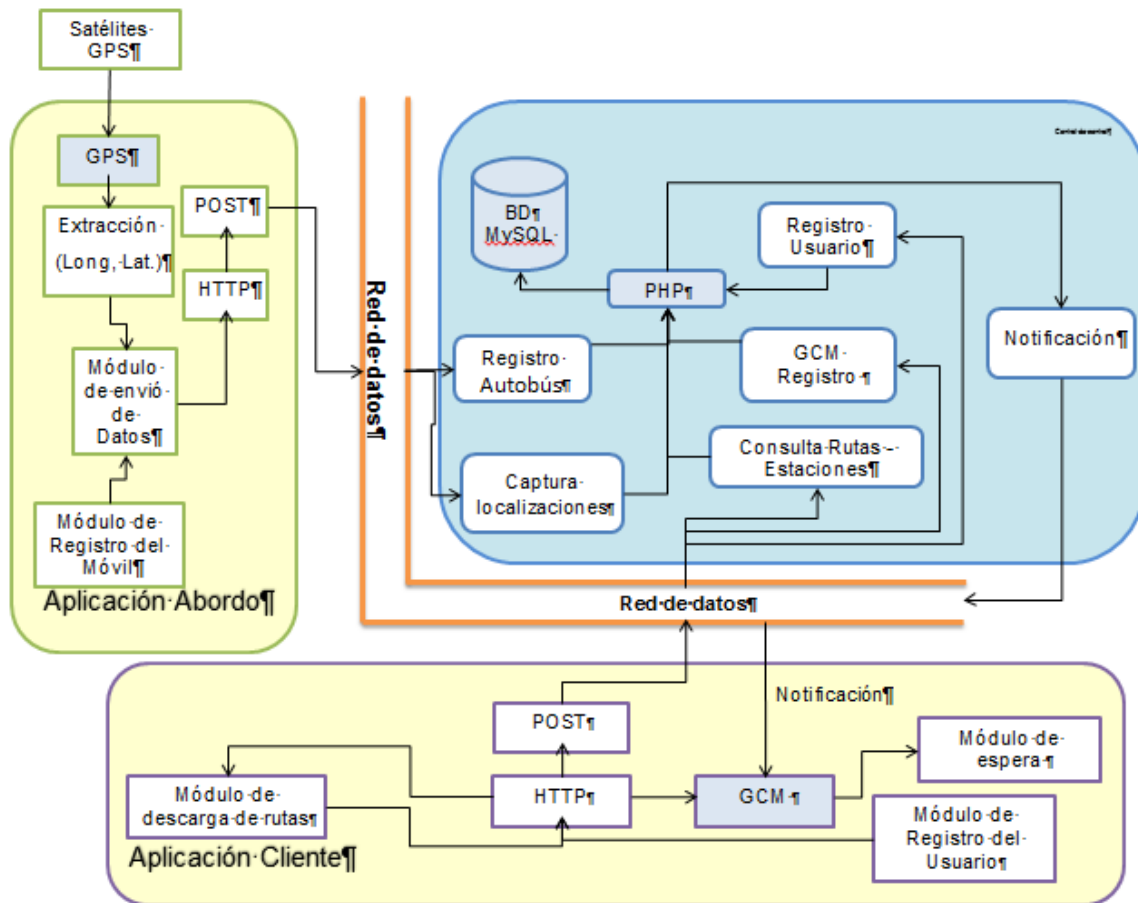


Figura 2.4 Diagrama de Componentes [Ver anexo]

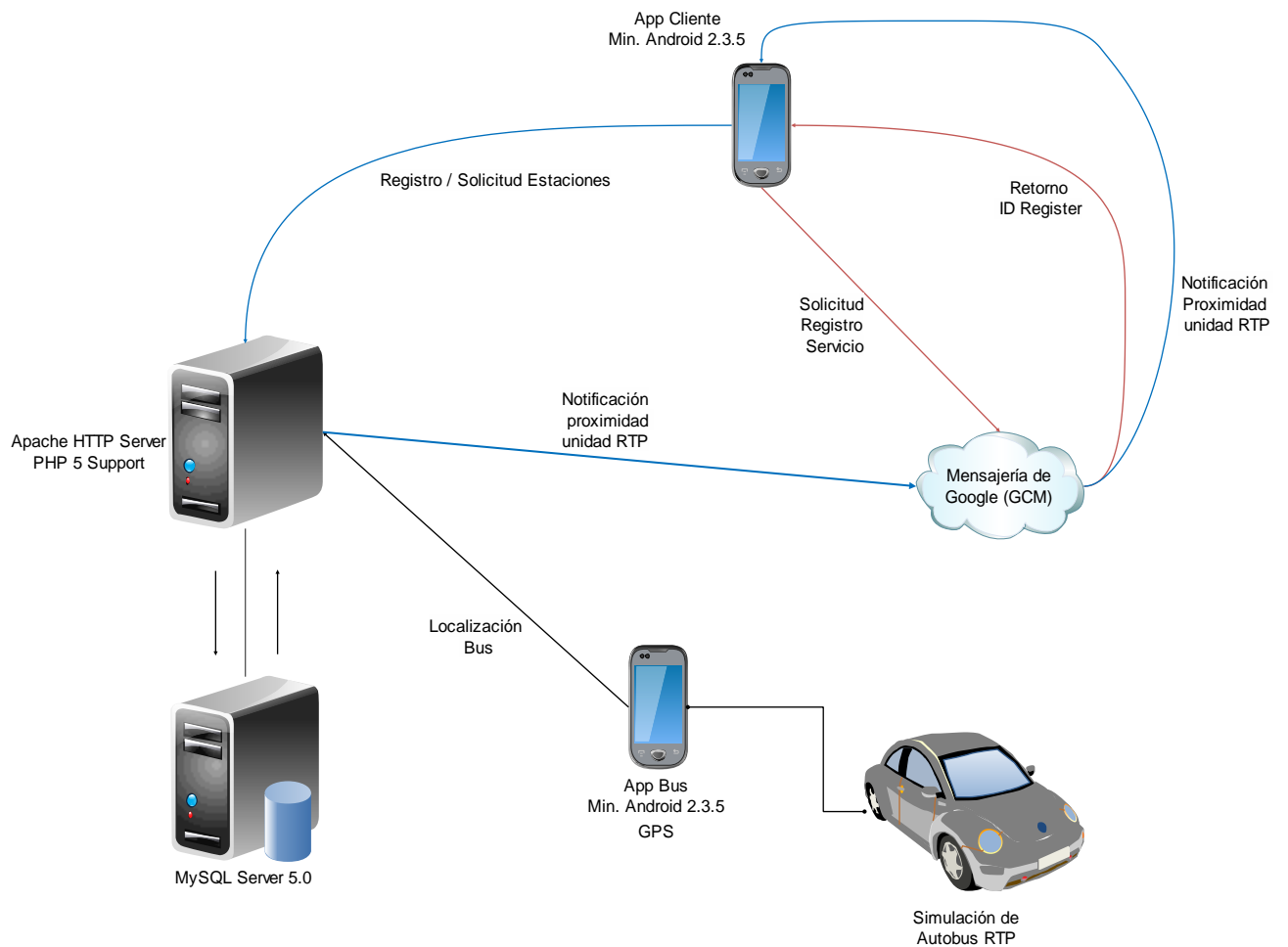


Figura 2.5 Diagrama de Arquitectura

2.2.2. Diseño de la Base de Datos

Se optó por la idea más actual en los modelos de bases de datos, el modelo Entidad-Relación. El diagrama de la base de datos se muestra a continuación.

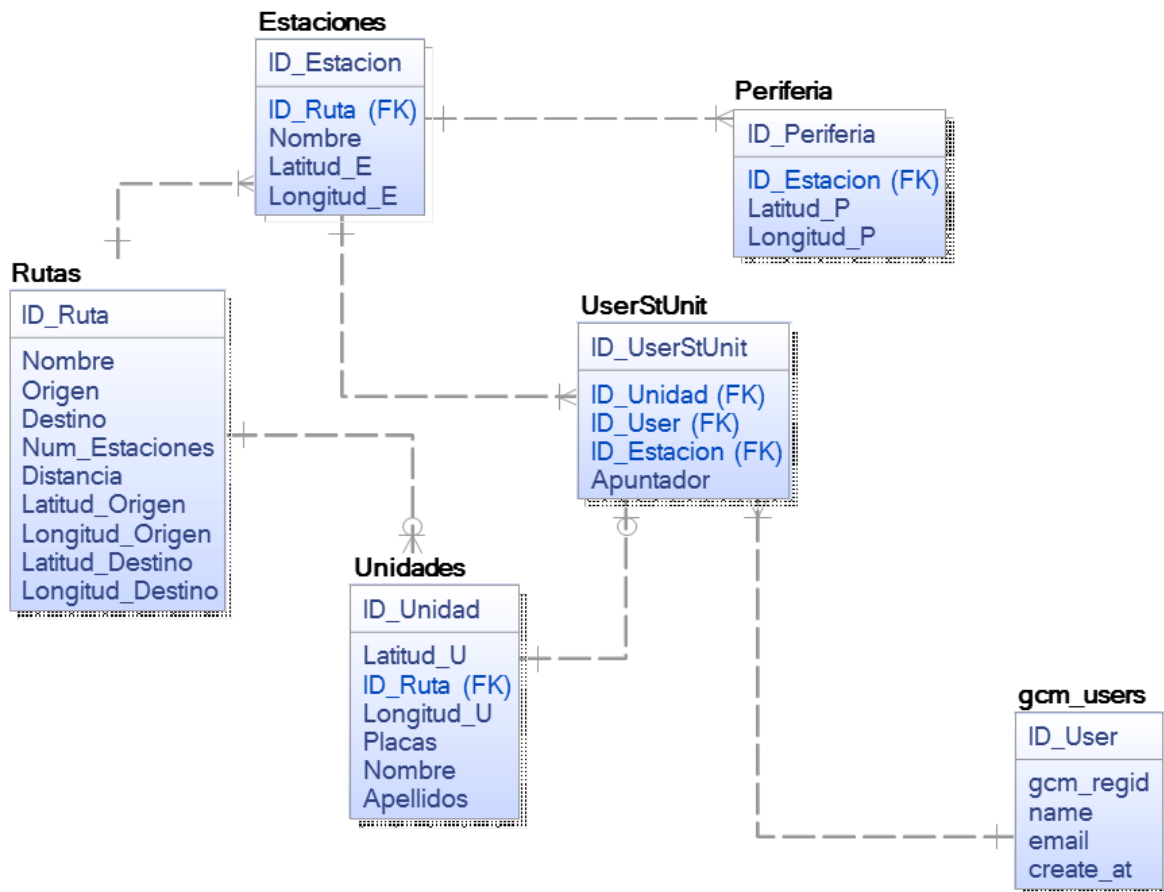


Figura 2.6 Diagrama entidad - relación de la base de datos del sistema

Capítulo 3 Desarrollo

3. Desarrollo del Sistema Móvil de notificación de proximidad para transporte.

3.1. Central de Control

La definición de una central de control es la parte más importante del sistema ya que es la base de todo el sistema propuesto.

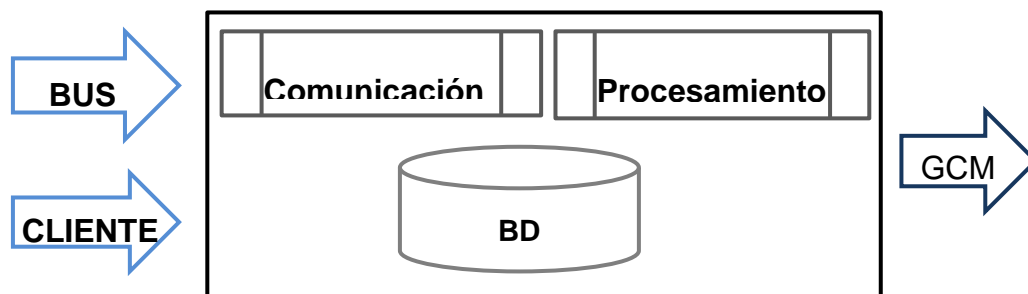


Figura 3.1 Diagrama Central de Control

El equipo con el que se cuenta y juega este papel es un servidor Mac OSx Server 10.7 con microprocesador Intel Xeon 3.4 GHz, 12 GB de RAM y 1 TB de almacenamiento; el cual tiene instalado Apache 5.0 HTTP Server, PHP 5 y MySQL Server 5.0.

Las partes principales de la central de control son las funciones de procesamiento y comunicación de datos, las cuales fueron desarrolladas en PHP y la Base de Datos que se creó con MySQL donde se tiene almacenada información fija como lo es el conjunto de rutas, periféricas, estaciones, etc. Así como información que es dinámica como las peticiones de los clientes, el envío de información de la posición de los vehículos, etc.

Se montó también sobre ese servidor una página web de Monitoreo o de consulta que nos permite hacer consultas básicas para visualizar la información actual en la base de datos.

3.1.1. Funciones en PHP

Se utilizó como lenguaje de servidor el lenguaje PHP, con él se definen diversas funciones que apoyan con las peticiones post que realizan los usuarios para la recepción de estos y su manejo tanto para enviarlos a la base de datos y como también extraer datos de ella mediante cadenas de consultas dentro de estas funciones, funcionando así como una interfaz entre las aplicaciones y la base de datos.

Los archivos de Script en PHP que hacen posible la funcionalidad del sistema son:

- **localización.php:** En este archivo está programada la forma de cómo validar cuando un vehículo está dentro de la periferia establecida ya en la base de datos, en este script viene programada la función matemática y se le envían como parámetros los puntos que conforman una periferia y así mandar y marcar las contestaciones al usuario de acuerdo a su petición, así también como el registro de los vehículos.
- **Lista_descarga.php:** Esta función nos apoya para poder ordenar las estaciones de una ruta determinada para darle un formato especial para poder ser descargada al usuario.
- **db_functions.php:** Este Script es uno de los más importantes de todos los demás programados, en él se almacenan todas las funciones necesarias para poder ser usadas por las demás y así poder hacer consultas, actualizaciones y devolver arreglos y estructuras de datos, a partir de estas y poder ser manipulados.
- **GCM.php:** Este Script siendo otro de los más importantes nos proporciona las funciones necesarias para utilizar el servicio de mensajería en la nube de Google, y manejar las variables especiales que GCM nos devuelve.
- **register.php:** Hace uso de las funciones de GCM.php y db_functions.php para poder hacer un registro del usuario en la base de datos.
- **send_message.php:** Aplica un formato específico al mensaje que se envía como notificación a los usuarios,
- **db_conect.php:** Permite hacer la conexión a la base de datos.

- **Config.php:** Define los parámetros necesarios para la conexión a la base de datos, Host, usuario, password, nombre de la base de datos, así como el APIKEY del proyecto.
- **DatosBD.php:** Arma una página Web de Monitoreo o consulta para el acceso y visualización de las tablas más importantes de la base de datos y realizar consultas más rápido.
- **Index.php:** Arma una página web donde se puede visualizar a los usuarios que ya están registrados en GCM así como también poder enviar algún mensaje de prueba y probar el servicio.
- **Test.php:** Genera una página web para registrar manualmente un dispositivo, pasándole como parámetros el nombre, correo y regid.

3.2. Desarrollo de la Aplicación de Abordo

Como se mencionó se tomará en cuenta la posición de las unidades de transporte, dado que se necesitaba ubicar los vehículos terrestres, en este caso las unidades de transporte, como se vio en el diseño, se pretende aprovechar los recursos de hardware que los dispositivos móviles cuentan como lo es el módulo GPS.

Actualmente, es bastante común encontrar un dispositivo GPS integrado en los terminales móviles de nueva generación. Esta característica nos proporcionará nuestra información GPS, como las coordenadas de nuestra posición, su precisión, la altitud sobre el nivel del mar, los satélites que estamos escuchando, etc.

En cuanto a los datos que proporciona la aplicación se encuentran las coordenadas geográficas de nuestra posición (latitud, longitud) y la precisión.

El sistema GPS es, por tanto, ideal para establecer la posición de un dispositivo, ya que prácticamente en cualquier espacio abierto se tiene cobertura, pero se debe tener en cuenta que la precisión de las coordenadas geográficas en condiciones óptimas tiene un error, de aproximadamente 15 metros.

Sin embargo, la aplicación adolece de un problema importante en núcleos urbanos y lugares montañosos, y es que en situaciones en las que permanecemos rodeados de edificios u obstáculos altos en general, no disponemos de línea de visión directa con el número suficiente de satélites necesarios para fijar nuestra posición. Por tanto, el sistema no es suficiente para ello. Esto puede solucionarse con el sistema A-GPS, que también puede funcionar de esta forma.

Es, por tanto, uno de los tres elementos para proporcionar servicios basados en datos de localización en el sistema propuesto.

3.2.1. Diagrama a Bloques

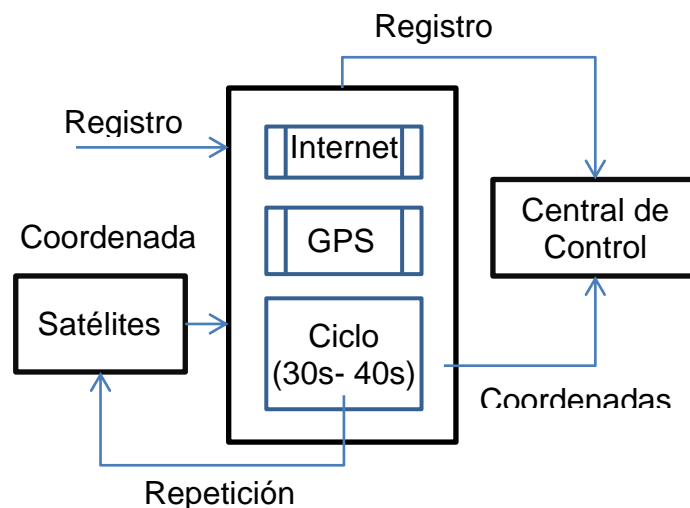


Figura 3.2 Diagrama a bloques de la aplicación de abordó

El orden con el que la aplicación funciona es la siguiente:

1. El usuario en este caso el conductor, se registra enviando los siguientes datos: nombre y el número de placas del vehículo para poder ser ubicado en la base de datos.
2. Tras haber registrado los datos del registro, pasaremos a la pantalla donde se podrán observar los datos importantes para el sistema que son la longitud, latitud que el

dispositivo va registrando, mismos datos que se enviaran a la base de datos para su procesamiento y validación.

3. Este proceso se repetirá en un ciclo de 30000 a 40000 milisegundos dependiendo del dispositivo y la rapidez con que el módulo GPS pueda resolver una ubicación.

3.2.2. Módulo de envío de datos

El lenguaje que se maneja para desarrollar aplicaciones en Android es Java para la parte lógica y operacional, y XML para la parte de interfaz de usuario y también para poder administrar los permisos necesarios para la comunicación con la red y la obtención de los datos GPS, que son los permisos que tendrá la aplicación en los dispositivos donde se instale.

Los permisos necesarios que la aplicación utiliza se declaran en el archivo Android Manifest.XML de ña aplicación Android, para esta sección del proyecto se emplearon los siguientes:

Permiso	Descripción
ACCESS_COARSE_LOCATION	Permite que una aplicación acceda a la ubicación aproximada derivado de las fuentes de ubicación de red, tales como torres de telefonía móvil y Wi-Fi.
ACCESS_FINE_LOCATION	Permite que una aplicación acceda a la ubicación precisa de fuentes de localización como GPS, antenas de telefonía móvil y Wi-Fi.
INTERNET	Permite a las aplicaciones abrir sockets de red.

ACCESS_NETWORK_STATE	Permite que las aplicaciones accedan a información sobre redes de datos
----------------------	---

Tabla 3.1 Permisos para la aplicación de abordó

Para poder enviar los datos se tuvieron que importar clases tanto de Android como de JAVA para la comunicación con la red de datos:

Clase	Descripción
android.net.ConnectivityManager	Clase que responde a preguntas sobre el estado de la conectividad de red.
android.net.NetworkInfo	Un Enumerado con El estado fino de una conexión de red y estado de red
java.net.HttpURLConnection	Define una conexión a un URL para la lectura o la escritura.
java.net.MalformedURLException	Esta excepción se produce cuando un programa intenta crear una URL de una especificación incorrecta.
java.net.URL	Un localizador uniforme de recursos que identifica la ubicación de un recurso de Internet tal como se especifica en la RFC 1738.

Tabla 3.2 Clases que se deben importar para la aplicación de Abordó

El módulo de envío de datos es más una clase especial que tiene las características necesarias para poder conectarse a la dirección del servidor.

1. La clase hereda de una clase especial de Android “AsyncTask”, como lo muestra el diagrama en UML.

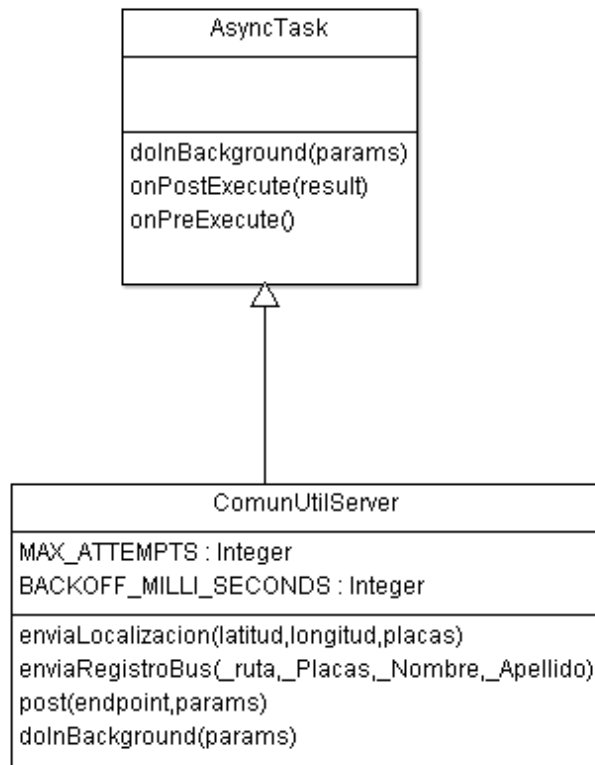


Figura 3.3 Diagrama de clases para conexión asíncrona

2. Se sobrescribe uno de sus métodos “doInBackground” de la Clase AsyncTask para poder mandar en segundo plano los datos que se le envíen.
3. Cuando declaramos una instancia de esta clase referenciamos a un método especial y mandamos los datos en un Arreglo como parámetro con el método “execute” que manda a llamar el método doInBackground.

Por Ejemplo.

```

ComunUtilServer llamada = new ComunUtilServer();
llamada.execute(Arreglo_de_datos);
  
```

1. Esto permite que los datos se manden en segundo plano en la aplicación.
2. Los métodos “envíaLocalizacion” y “envíaRegistroBus” Arman un HashMap que es una colección especial de JAVA donde a cada elemento le asigna un identificador a su contenido,

3. Una vez ya armada la colección para cada método se manda como parámetro al método post el cual enviara los datos a la dirección especificada por el servidor.
4. Estos métodos se mandan a llamar dentro del método heredado doInBackground.

3.2.3. Módulo de Registro

El proceso de cómo se realiza el registro del Vehículo se detalla en el siguiente diagrama de flujo.

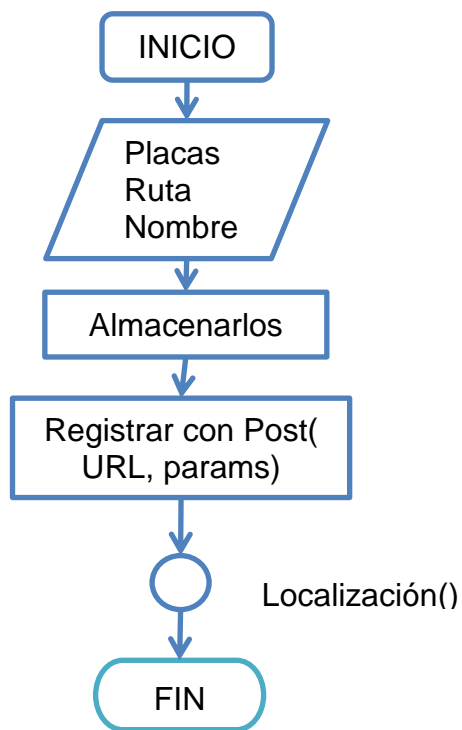


Figura 3.4 Diagrama de Registro del vehículo

1. El Conductor manda los datos de entrada
2. Estos se almacenaran en un Arreglo de Datos
3. Posteriormente mediante un objeto de una clase especial para realizar la comunicación con el servidor se hace un post para mandar el Arreglo al Servidor de base de datos.

4. Después de esto entramos a la fase de localización.

3.2.4. Módulo de Obtención de Datos del GPS

El proceso de cómo se realiza la obtención de los datos del módulo GPS lo describe el siguiente diagrama de flujo.

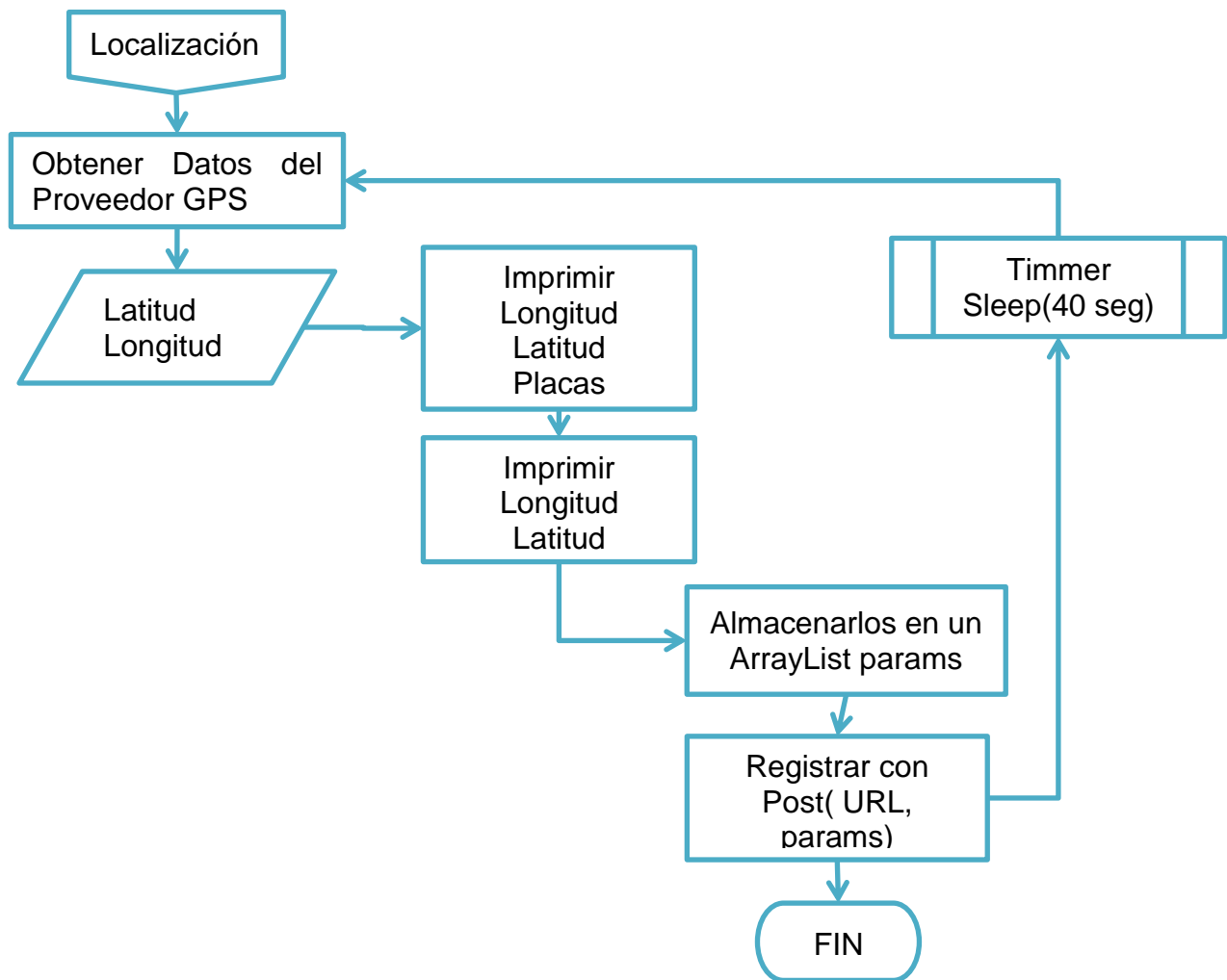


Figura 3.5 Diagrama módulo de obtención de datos del GPS

1. Tras ser registrada la unidad de transporte el usuario activara manualmente la función para comenzar el envío de los datos de posición.
2. Estará en espera en un intervalo de 30 a 40 segundos para poder obtener la primera lectura.

3. Una vez obtenida una lectura del GPS, la latitud y la longitud actuales se imprimirán en pantalla, y se agruparan en un Arreglo de Datos.
4. Este arreglo de Datos que contiene la longitud y la latitud se actualiza y se vuelve a enviar en un periodo de 30 a 40 segundos actualizando en la base de datos la posición del autobús.

3.3. Aplicación del cliente

La aplicación del cliente es dependiente de una serie de clases para su efectivo funcionamiento y se debe tomar en cuenta que toda conexión es de manera asíncrona, ya que no se requiere tener una conexión permanente. En los próximos apartados se explica cómo se desarrolló cada módulo de la aplicación.

3.3.1. Extensiones necesarias

Este proyecto requirió de extensiones para poder ingresar a los servicios de Google y para hacer una interfaz gráfica más llamativa y amigable.

3.3.1.1. Para servicios de GCM

Se debe tomar en cuenta que para desarrollar un proyecto que haga uso del servicio de Mensajería en la nube de Google, tenemos que contar con el paquete que contiene las funciones necesarias para el intercambio de datos.

El paquete es el archivo nombrado como “GCM.jar” [22], que es un recurso que se obtiene de la página web oficial de los desarrolladores de Android (Android Developers); teniendo el mencionado archivo se debe importar al proyecto.

3.3.1.2. ActionBarSherlock

ActionBarSherlock es un complemento que te permite hacer uso de pestañas (Tabs) a partir de dispositivos con Android 2.3, cabe señalar que las pestañas sólo se pueden implementar nativamente a partir de Android 3.0.

3.3.2. Permisos

En la tabla se listarán algunos permisos necesarios para el funcionamiento de la aplicación, pero cabe señalar que no son los únicos que se requieren, se requieren algunos otros que se mencionaron en la tabla 3.1, como es el de internet y el de estado de la red.

Permiso	Descripción
ipn.esimecu.gcmpushnotif.permission.C2D_MESSAGE	Permiso para que únicamente esta aplicación reciba los mensajes
com.google.android.c2dm.permission.RECEIVE	Permiso para recibir mensajes de GCM
android.permission.GET_ACCOUNTS	Permiso para acceder a las cuentas de correo.
android.permission.WRITE_EXTERNAL_STORAGE	Permiso para poder escribir en la SD.
android:name="android.permission.READ_EXTERNAL_STORAGE"	Permiso para poder leer la SD.
android.permission.WAKE_LOCK	Permiso para mantener los servicios de una aplicación trabajando.
android:name="android.permission.VIBRATE"	Permiso para que la aplicación haga uso de la vibración.

Tabla 3.3 Permisos para la aplicación del cliente

3.3.3. Diagrama lógico

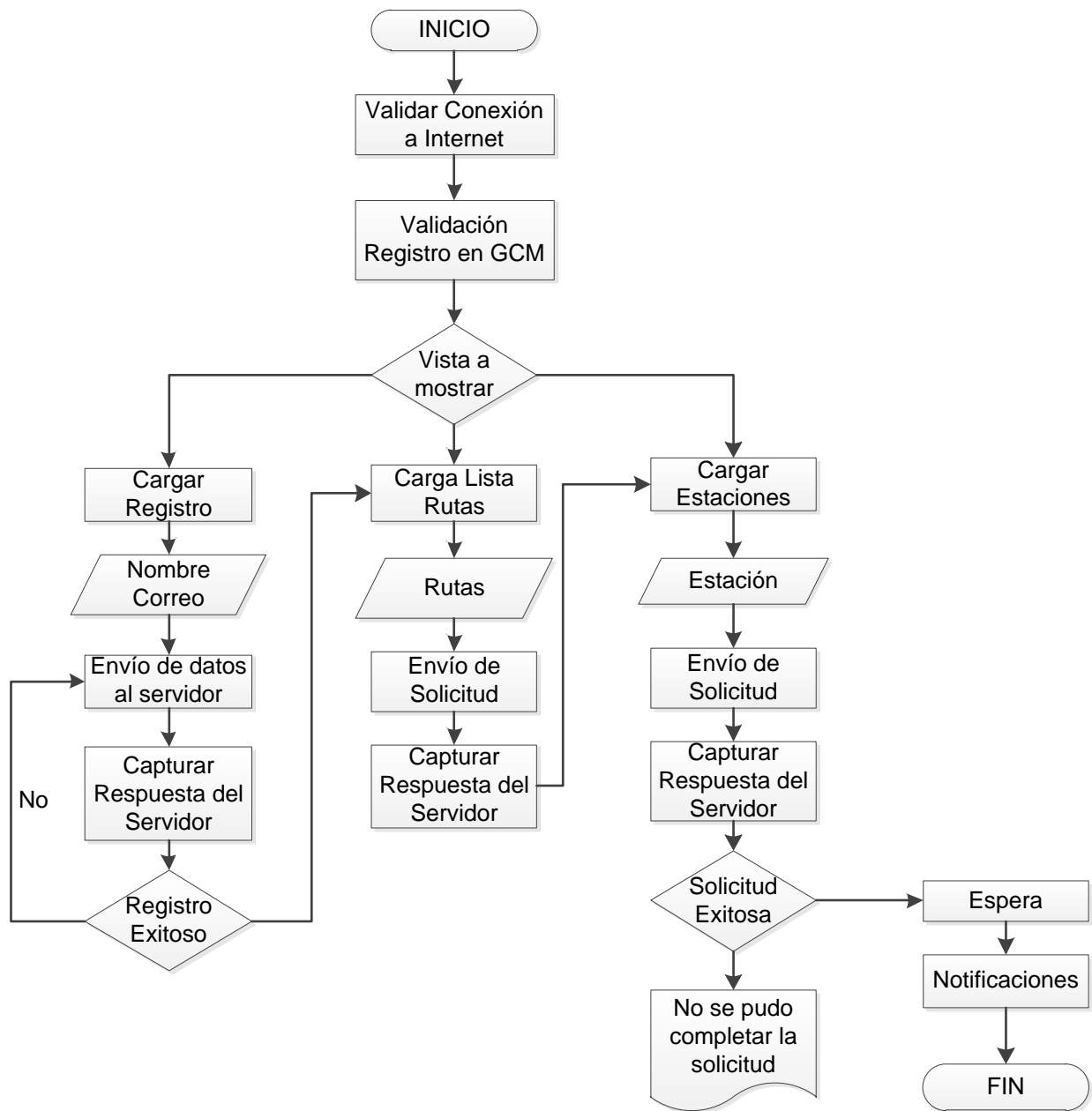


Figura 3.6 Diagrama a bloques de la aplicación del cliente

3.3.4. Registro GCM

3.3.4.1. Validación Registro en GCM

El proceso de validación primeramente manda a llamar una función de GCM, que se encarga de obtener el ID de registro, lo almacenamos en una variable de tipo String, si fuera el caso que la variable quedará vacía significa que el dispositivo no está registrado para hacer uso del servicio de GCM. En este punto nos encontramos dos casos:

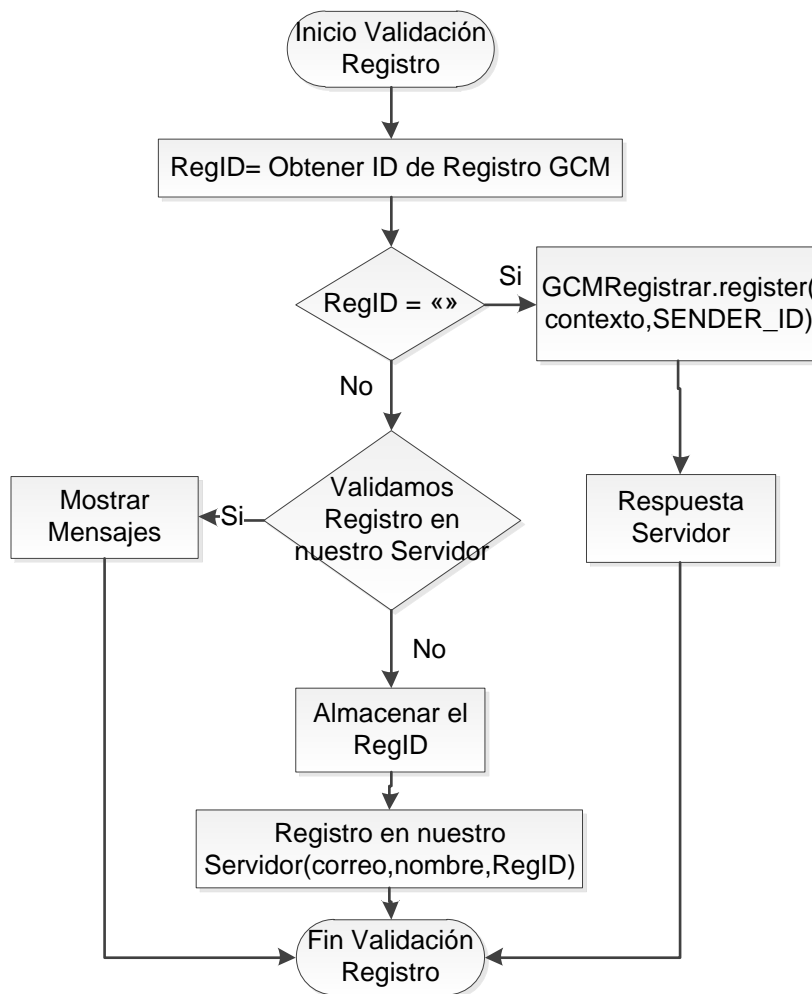


Figura 3.7 Diagrama de Validación Registro

El SENDER ID es el parámetro que identifica a la aplicación, este parámetro se obtiene en la plataforma de Google APIs Console [23].

3.3.4.2. Vista a mostrar

Dependiendo de la respuesta de la Validación del Registro en GCM se hará el despliegue de Registro, lista de rutas o la lista de estaciones.

Para moverse entre cada vista se utiliza un objeto Intent del paquete “android.content” y se le debe pasar al constructor el parámetro del contexto de la aplicación y el nombre de la clase que se va a ejecutar, se debe tomar en cuenta que la clase a ejecutar, debe extender de la clase Activity del paquete “android.app”, posteriormente se llama a la función startActivity, pasándole el objeto Intent como parámetro; hay posibilidad de finalizar el ciclo de vida de la actividad desde donde se está llamando el inicio de la nueva vista, lo que causará que cuando se presione el botón regresar, no se muestre la vista anterior.

Ejemplo:

```
Intent i = new Intent(getApplicationContext(), Inicio.class);  
startActivity(i);
```

3.3.4.3. Cargar Registro

El módulo cargar registro es el encargado de desplegar la vista para registrar al usuario. En donde el usuario deberá ingresar su nombre y correo. Posteriormente debe presionar el botón “Registrar”. Los datos se almacenaran en una colección de tipo Map.

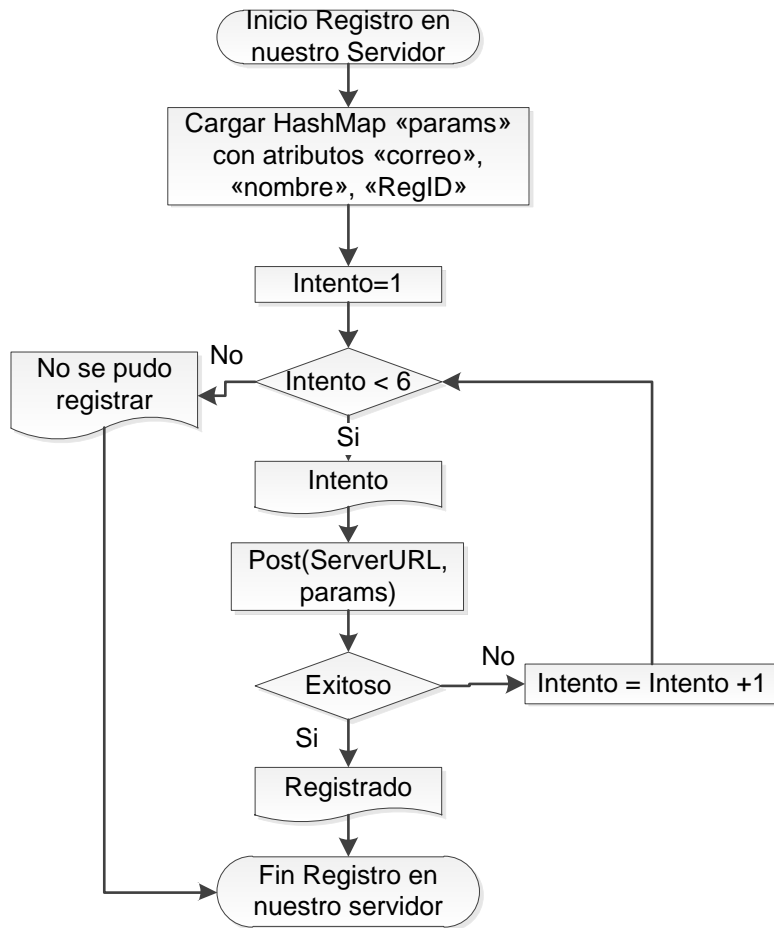


Figura 3.8 Diagrama de Registro en nuestro servidor

La URL proporcionada es la del servidor donde se almacenará la información, el de nosotros.

3.3.4.4. Cargar Lista Rutas

Se carga un ListView con todas las rutas de la red de transporte de pasajeros del D.F., esta lista se precarga en la aplicación haciendo uso de un ArrayList.

Al ArrayList se le da como parámetros un objeto ModelParada, que tiene como atributos el “código” y “nombre” de la ruta, y “seleccionado” que permite verificar si esa parada fue seleccionada, por default es falso.

El usuario selecciona una o múltiples rutas y debe presionar el botón “Descargar Seleccionados”.

En seguida se entra en un ciclo para validar las rutas seleccionadas, y se almacenarán los códigos de las rutas dentro de un ArrayList.

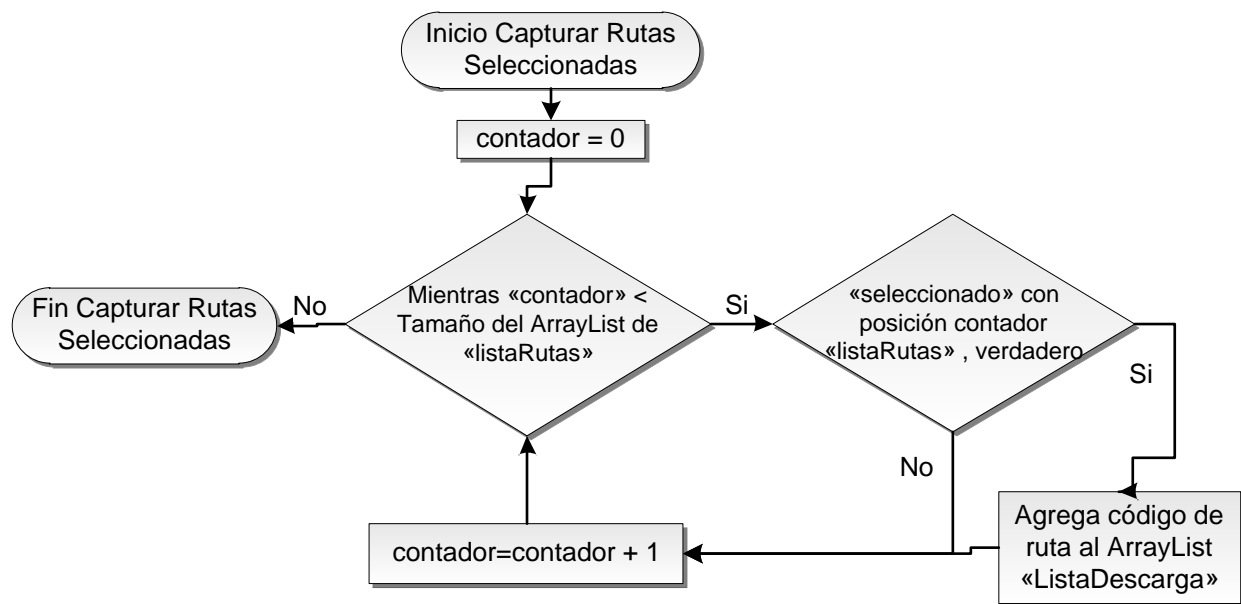


Figura 3.9 Diagrama para almacenar las rutas seleccionadas

Teniendo los códigos de las rutas, se procede a construir el objeto JSON, que nos permite enviar un conjunto de parámetros en un solo objeto, que incluirá el ID de registro para enviarlo al servidor a través de un POST.

3.3.5. Recepción de cadenas JSON

Se espera la respuesta del servidor que nos retornará, en caso de ser exitosa la transacción, una cadena JSON con las paradas pertenecientes a las rutas seleccionadas.

En el momento en el que se tiene el objeto JSON, se procede a almacenarlo en la tarjeta SD del dispositivo.

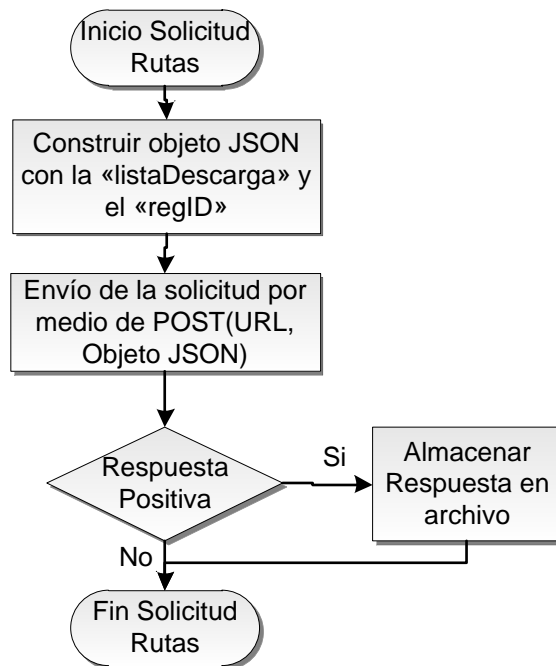


Figura 3.10 Diagrama de solicitud de ruta

Hasta el momento en el que se reciban las estaciones y se almacenen, se podrá mostrar la vista de las estaciones, en el cual se carga una pestaña por cada ruta y dependiendo de la pestaña en la que este posicionado mostrará las estaciones correspondientes, la relación se logró al usar un ArrayList y el número de pestaña.

Para solicitar la notificación de proximidad en una estación, a diferencia de la solicitud de ruta, sólo se puede solicitar una estación a la vez, en cuanto al método de envío y recepción es el mismo, a diferencia que la respuesta es un mensaje de afirmación.

En esta vista es donde se hace uso de ActionBarSherlock.

3.3.6. Notificaciones

Una vez dada de alta la petición en el servidor, queda únicamente esperar la notificación. Cuando el servidor de Google envía el mensaje, el servicio de escucha despierta la aplicación y genera una alerta, levantando el servicio de sonido y vibración.

Capítulo 4 Resultados

4. Resultados del Sistema Móvil de notificación de proximidad para transporte.

Después de dar a conocer por parte todos los módulos que integran el proyecto ahora se unirán para verificar su funcionamiento en modulo funcional conjunto así como dar a conocer sus alcances, se implementó todo lo visto en desarrollo teniendo éxito en el diseño del sistema.

4.1. Pruebas con GCM.

Se hicieron las pruebas para poder mandar los mensajes a través de GCM se montó una página web de prueba donde se pueden visualizar los usuarios

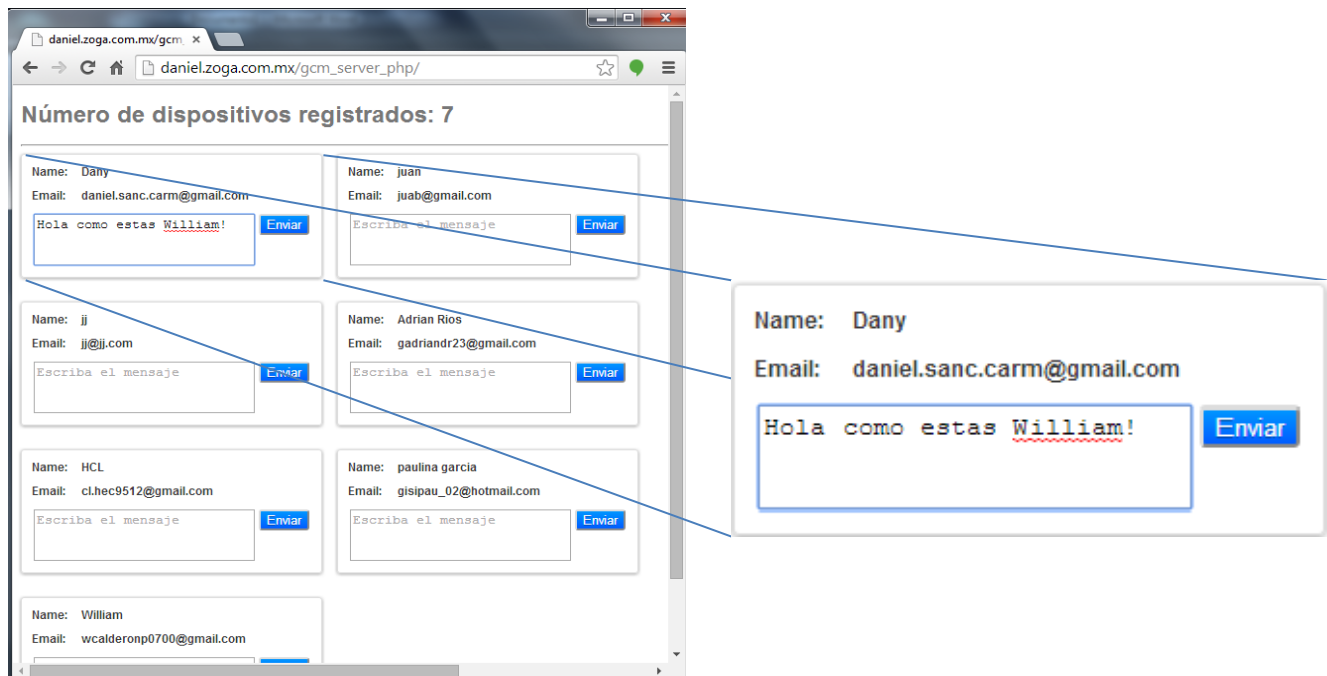


Figura 4.1 Portal web de pruebas con GCM

El fin de esta página es visualizar a los usuarios registrados en GCM, todo usuario que desde su aplicación se registre en GCM se verá reflejado, así también en un principio fue para probar el envío de información por medio de GCM.

Las siguientes pantallas muestran cómo es que se muestra la información cuando llega al dispositivo que este registrado en GCM.

Así es como se probó el envío de información por medio de GCM y sea exitoso para cualquier dispositivo que se registre.

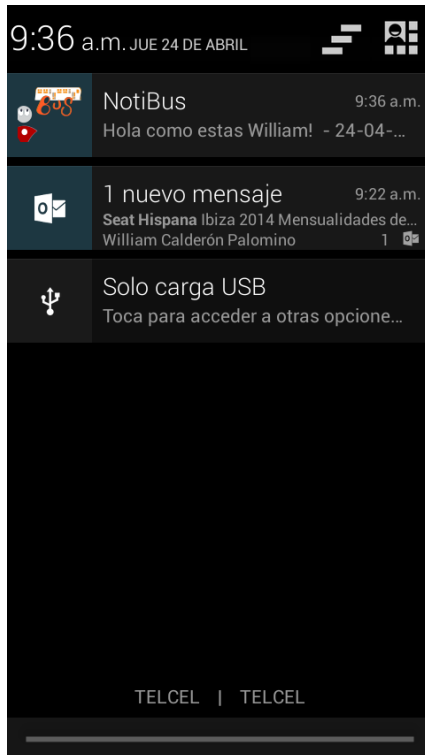


Figura 4.2 Notificación del mensaje enviado desde el portal web

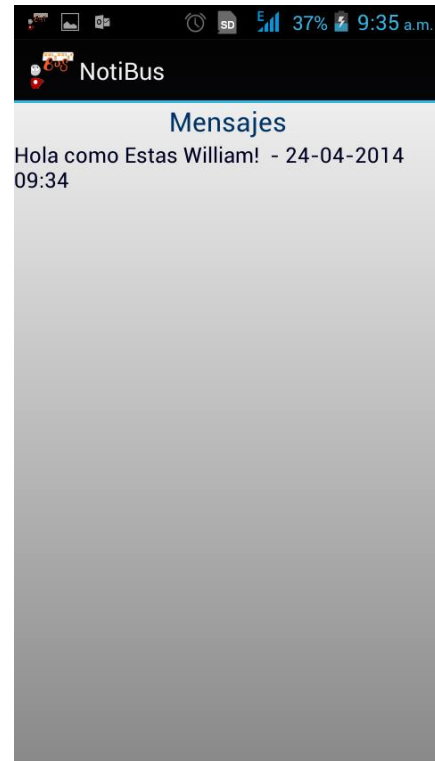


Figura 4.3 Muestra del mensaje

4.2. Pruebas con la Aplicación del Cliente

El para el registro del usuario de “NotiBus” se le piden dos datos que son explícitos, el correo electrónico y su nombre, e internamente el dispositivo queda registrado en GCM y en la Base de datos devolviendo una confirmación.

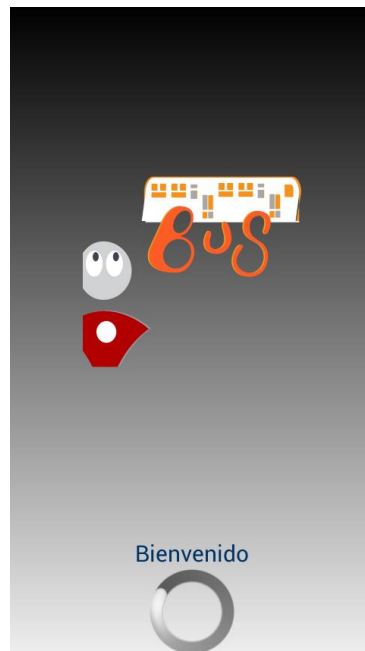


Figura 4.4 Pantalla de bienvenida

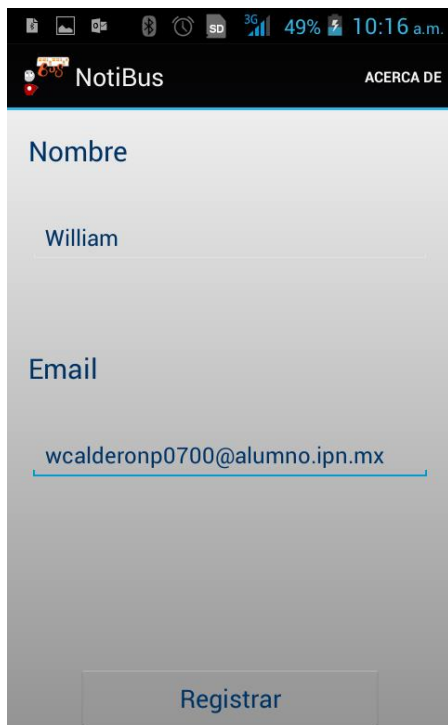


Figura 4.5 Registrando al usuario



Figura 4.6 Mensajes en el proceso de registro

Una vez hecho esto se verifico la selecció de rutas una lista que nos muestra las rutas para poder descargar sus estaciones por ruta.



Figura 4.7 Lista de Rutas

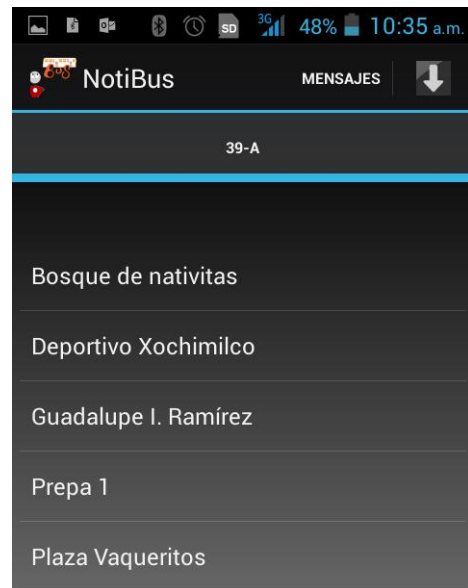


Figura 4.8 Lista de estaciones descargadas de la Ruta

Posteriormente se mandan las peticiones al servidor seleccionando la estación que el usuario requiere ser notificado que este su autobús, ahora solo resta esperar por la notificación.

Se observó que los registros fueron exitosos, la aplicación fue probada en dispositivos con versión de Android 2.3.5 hasta dispositivos con versión 4.2.1

Se concluye que es funcional para cualquier dispositivo que este dentro del rango de versiones de Android.

4.3. Pruebas con la Aplicación de Abordo

Como se describió en el diseño la idea es tener corriendo la aplicación en el dispositivo móvil de los conductores de las unidades y de esta forma, tener monitoreadas las unidades en cuanto sus datos de localización, y la ruta a la que pertenecen.

Como se explicó, la aplicación extraerá los datos de ubicación del dispositivo y las enviará al servidor para que estos puedan ser procesados y validados, y así determinar si la ubicación del móvil se encuentra dentro de la periferia de alguna estación elegida por el usuario.

Se probó el registro de los datos del móvil como son la ruta, el nombre del conductor y las placas del vehículo, la ruta y las placas del vehículo son datos que nos ayudarán a identificar una unidad de transporte en la base de datos.



Figura 4.9 Pantalla de bienvenida de la aplicación de abordó

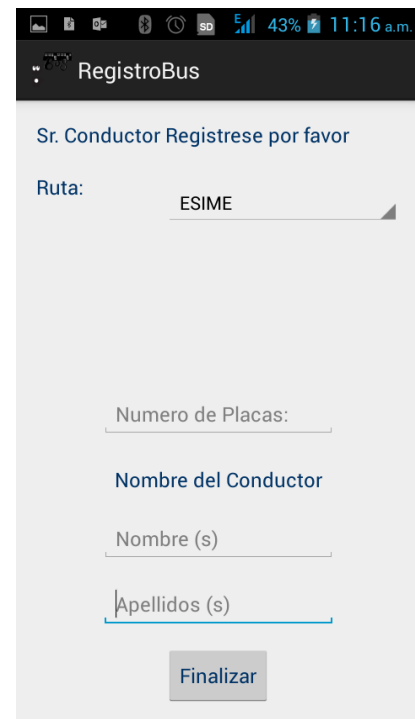


Figura 4.10 Pantalla de registro del vehículo y conductor

Se seleccionó la ruta que pasa por la Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Culhuacan.



Figura 4.11 Lista de Rutas para el vehículo

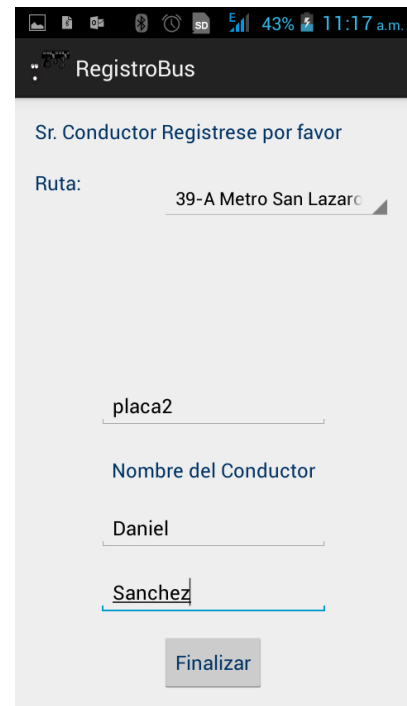


Figura 4.12 Registrando a una persona y vehículo

Posteriormente pasamos a la Actividad en donde se estarán mandando los datos de posición al servidor, en un periodo de entre 30 a 40 Segundos se estará actualizando los datos de latitud y longitud, y se estará actualizando en la base de datos.



Figura 4.13 Pantalla de en el momento que ya se esta obteniendo la posición

Los resultados fueron positivos dado que el registro es exitoso y la actualización de la información de localización del vehículo se actualiza exitosamente.

4.4. Pruebas Reales recorriendo parte de la Ruta 39-A Xochimilco/Bosque de Nativitas – Metro San Lazaro

Se recorrió parte de la ruta desde el Bosque de Nativitas hasta la parada de Coyuya y se obtuvieron los siguientes resultados mostrados en la tabla, para hacer esta prueba se activó la Aplicación de Abordo sobre un Automóvil que simuló ser el autobús, mandando su ubicación.

Al mismo tiempo se dieron de alta todas las Estaciones de la ruta con el fin de probar la exactitud con el que la aplicación de abordo resuelve y se procesa en el servidor la información la siguiente tabla muestra los resultados.

Estación	Resultado
Bosque de Nativitas	Notificó Satisfactoriamente
Deportivo Xochimilco	Notificó Satisfactoriamente
Guadalupe I. Ramírez	No mando mensaje de Notificación
Prepa 1	Notificó Satisfactoriamente
Plaza Vaqueritos	Notificó Satisfactoriamente
Wall-Mart	Notificó Satisfactoriamente
Hda. Mazatepec	Notificó Satisfactoriamente
Clazada Hueso	Notificó Satisfactoriamente
Calzada las Bombas	Notificó Satisfactoriamente

Av. Santa Ana	Notificó Satisfactoriamente
Tasqueña	No mando mensaje de Notificación
Ermita	Notificó Satisfactoriamente
Escuadron 201	Notificó Satisfactoriamente
Aculco	Notificó Satisfactoriamente
Apatlaco	No mando mensaje de Notificación
Iztacalco	Notificó Satisfactoriamente
Cuyuya	Notificó Satisfactoriamente

Tabla 4.1 Relación estación - notificación

Los resultados anteriores marcan que se debe poner un poco de atención en la parte de como se estan identificando las estaciones o tomar en cuenta unos puntos diferentes para las periferias de las estaciones ya que en algunas no se recibe mensaje de notificacion.

También es probable que factores como el estado de la red de datos celular, el ancho de banda, el nivel de señal en el area sean factores que alenten las transacciones y por ello no se ejecuten las transacciones de envio y recepcion de informacion.

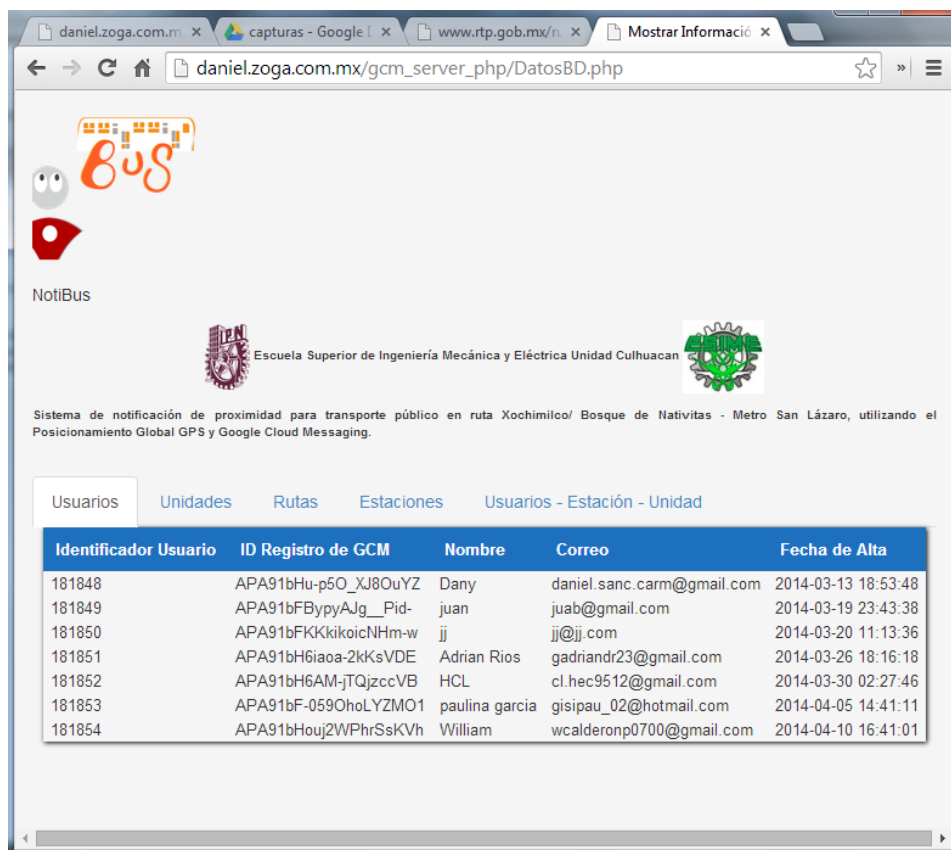
4.5. Pagina web de Monitoreo

Mientras se hacian pruebas de registro y evitar tener que accesar a la base de datos desde el manejador directamente surgio la propuesta de realizar una pagina web que nos de informes sobre lo que se esta actualizando y/o registrando en la base de datos.

Servirá como una herramienta de apoyo para consulta de los desarrolladores y administradores del Sistema .

La página web nos devuelve los siguientes informes:

- Informe de los usuarios registrados



NotiBus

Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Culhuacán

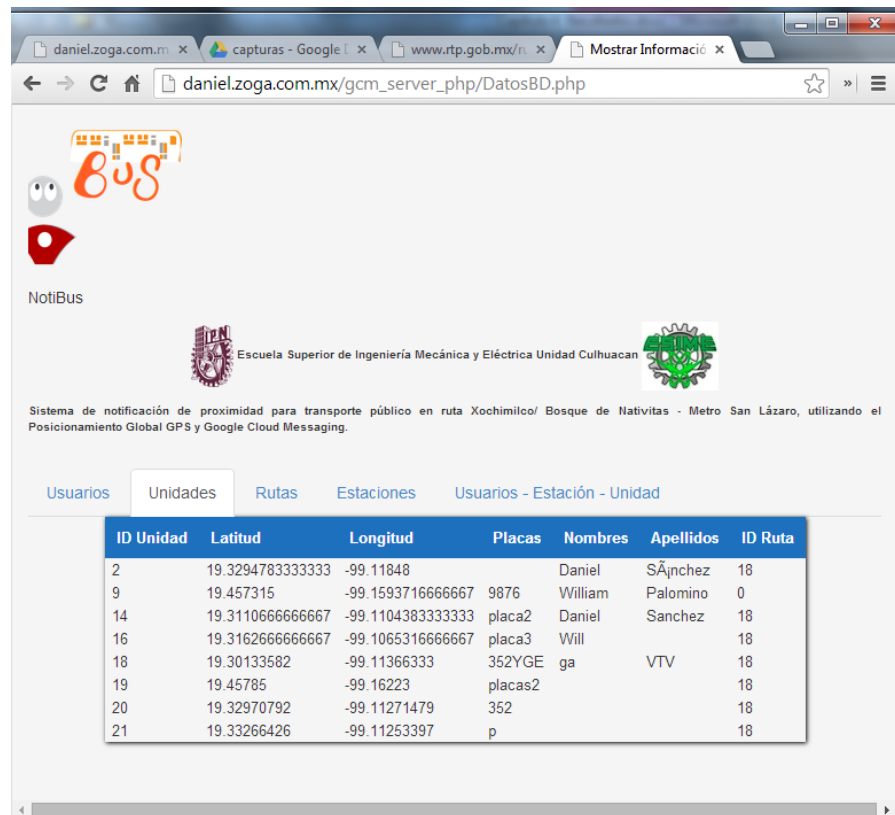
Sistema de notificación de proximidad para transporte público en ruta Xochimilco/ Bosque de Nativitas - Metro San Lázaro, utilizando el Posicionamiento Global GPS y Google Cloud Messaging.

Usuarios Unidades Rutas Estaciones Usuarios - Estación - Unidad

Identificador Usuario	ID Registro de GCM	Nombre	Correo	Fecha de Alta
181848	APA91bHu-p5O_XJ8OuYZ	Dany	daniel.sanc.carm@gmail.com	2014-03-13 18:53:48
181849	APA91bFBypAJg_Pid-	Juan	juab@gmail.com	2014-03-19 23:43:38
181850	APA91bFKKkikoicNHm-w	jj	jj@jj.com	2014-03-20 11:13:36
181851	APA91bH6iaoa-2kKsVDE	Adrian Rios	gadriandr23@gmail.com	2014-03-26 18:16:18
181852	APA91bH6AM-jTQjzccVB	HCL	cl.hec9512@gmail.com	2014-03-30 02:27:46
181853	APA91bF-059OhoLYZMO1	Paulina Garcia	gisipau_02@hotmail.com	2014-04-05 14:41:11
181854	APA91bHouj2WPhrSsKVh	William	wcalderonp0700@gmail.com	2014-04-10 16:41:01

Figura 4.14 Usuarios registrados

- Informe de la unidades de transporte



NotiBus

Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Culhuacán

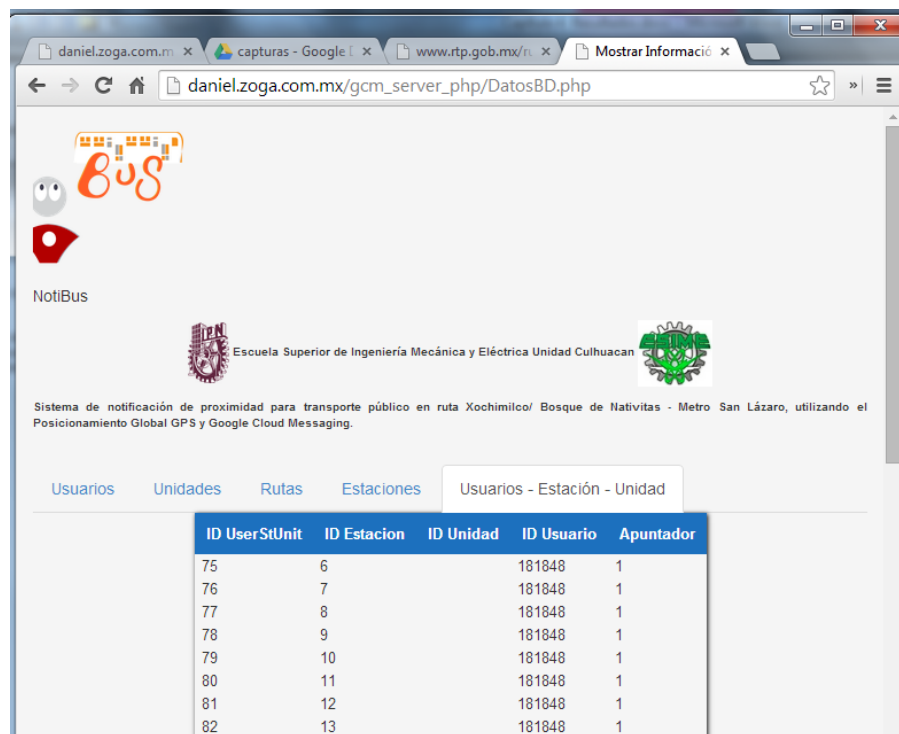
Sistema de notificación de proximidad para transporte público en ruta Xochimilco/ Bosque de Nativitas - Metro San Lázaro, utilizando el Posicionamiento Global GPS y Google Cloud Messaging.

Usuarios Unidades Rutas Estaciones Usuarios - Estación - Unidad

ID Unidad	Latitud	Longitud	Placas	Nombres	Apellidos	ID Ruta
2	19.3294783333333	-99.11848		Daniel	Sánchez	18
9	19.457315	-99.1593716666667	9876	William	Palomino	0
14	19.3110666666667	-99.1104383333333	placa2	Daniel	Sanchez	18
16	19.3162666666667	-99.1065316666667	placa3	Will		18
18	19.30133582	-99.11366333	352YGE	ga	VTV	18
19	19.45785	-99.16223	placas2			18
20	19.32970792	-99.11271479	352			18
21	19.33266426	-99.11253397	p			18

Figura 4.15 Unidades registradas

- Informe de las peticiones de los Usuarios y Con las estaciones que pidieron donde el Status “Apuntador” son los usuarios que esperan una notificación.



The screenshot shows a web browser window with the URL `daniel.zoga.com.mx/gcm_server_php/DatosBD.php`. The page features a header with a 'Bus' logo and a 'NotiBus' title. Below the header, there are logos for 'Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Culhuacán' and a description of the system as a proximity notification system for public transport. A navigation bar includes tabs for 'Usuarios', 'Unidades', 'Rutas', 'Estaciones', and a dropdown menu currently showing 'Usuarios - Estación - Unidad'. The main content area displays a table with the following data:

ID UserStUnit	ID Estacion	ID Unidad	ID Usuario	Apuntador
75	6		181848	1
76	7		181848	1
77	8		181848	1
78	9		181848	1
79	10		181848	1
80	11		181848	1
81	12		181848	1
82	13		181848	1

Figura 4.16 Relación petición - usuario

Referencias

- [1]. Romero Rojano, Antonio [2003]. *Sistema de localización y seguimiento de móviles utilizando el Sistema de Posicionamiento Global*. Tesis de maestría publicada. Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Zacatenco, Distrito Federal, México.
- [2]. Martínez Osorio, José A. *Sistema de rastreabilidad GPS con interface Web y SMS para dispositivos móviles*. Tesis licenciatura. Universidad Nacional Autónoma de México, Facultad de Ingeniería, Distrito Federal, México.
- [3]. Conceptos básicos de los Sistemas de Información.
- Viernes, 23 de julio de 2004 10:00:24 a. m.
<http://fccea.unicauca.edu.co/old/siconceptosbasicos.htm>
- [4]. Márquez Avendaño, Bertha M., Zulaica Rugarcía, José M. "Implementación de un reconocedor de voz gratuito a el sistema de ayuda a invidentes Dos-Vox en español". Univ. De las Américas Puebla. Cholula, Puebla, México, 2004. Consultada. Octubre 2013.
- http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/marquez_a_bm/portada.html
- [5]. Gómez Vieites, Álvaro. Suárez Rey, Carlos. *Sistemas de información "Herramientas prácticas para la gestión"*. Edit. Alfaomega 3ª Edición. Junio 2011.
- [6]. Tomás Gironés, Jesus. "El gran libro de Android". Edit. Alfaomega. 3ª Edición. Septiembre 2013.
- [7]. Berson, Alex. "Client/Server Architecture". Edit. McGraw-Hill. 1992
- [8]. Tipos de servidores.
2013. Consultado octubre 2013
- . <http://www.tiposde.org/informatica/131-tipos-de-servidores/>
- [9]. Kaplan, E. *Understanding GPS: Principles and Applications*, Edit. Norwood. MA: Artech House. 1990

[10]. Project 2 GPS

Consultado octubre 2013.

<http://mason.gmu.edu/~hsyed1/math447/p2/index.html>

[11]. A. Ribagorda, Glosario de Términos de Seguridad de las T.I., Ediciones CODA, 1997.

[12]. Ceballos, Fco. Javier. Java 2 “Curso de programación”. 4ª Edición. Edit. Alfaomega - Ra-Ma. Febrero 2011.

[13]. Tomas Girones, Jesús. “El Gran Libro de Android”. 4ª Edición. Edit. Marcombo Enero 2013.

[14]. Los Estándares internacionales y su importancia para la industria del software.

Consultado diciembre 2013.

<http://www.cyta.com.ar/ta1202/v12n2a3.htm>

[15]. Red de Transporte de Pasajeros del Distrito Federal.-RTP. (Última modificación: viernes, 8 de febrero de 2013 12:46:24 p. m.) Consultado febrero 2014.

http://www.rtp.gob.mx/red_rutas.html

[16]. Combinación lineal. Departamento de Matemáticas, CCIR/ITESM. 10 de enero de 2011. Consulta diciembre 2013.

<http://cb.mty.itesm.mx/ma1010/materiales/ma1010-05.pdf>

[17]. Logistec - MIDDLEWARE RFID: LA IMPORTANCIA DE FACILITAR PROCESOS. (n.d.). Logistec. Retrieved March 6, 2014, from

<http://revistalogistec.com/index.php/equipamiento-y-tecnologia/383-packaging-y-picking/950-middleware-rfid-la-importancia-de-facilitar-procesos>

[18]. Lafuente, Alberto, Larrea, Mikel. Sistemas Distribuidos. Introducción. Dpto. ATC, UPV/EHU. Consultado 8 Marzo, 2014. <http://www.sc.ehu.es/acwlaalm/sdi/introduccion-slides.pdf>

- [19]. Villaroel Salcedo, José Luis. Sistema en tiempo Real. Centro politécnico Superior. Consultado 8 de marzo,2014. <http://webdiis.unizar.es/~joseluis/STR.pdf>
- [20]. 18 Configuring for Multiple Tenants. (n.d.). Configuring for Multiple Tenants. Modificado 6 Marzo, 2014. <http://www.oracle.com/technetwork/java/codeconv-138413.html>
- [21]. Usage. (n.d.). ActionBarSherlock. Modificado 10 Marzo, 2014. Consultado 10 de Febrero, 2014. <http://actionbarsherlock.com/index.html>
- [22]. Google Cloud Messaging for Android. (n.d.). Android Developers. Modificado 10 de marzo, 2014. Consultado 21 de noviembre, 2013 <http://developer.android.com/google/gcm/>
- [23]. Google APIs Console – Google Developers. Modificado viernes, 18 de octubre de 2013. Consultado octubre 2013. <https://developers.google.com/console/help/?csw=1>

Glosario

Descentralizado: Que carecen de dependencia de una unidad central.

Hipertexto: Cualquier texto que contenga enlaces con otros documentos o ficheros.

Modem: es un acrónimo formado por dos términos: modulación y demodulación. Se trata de un aparato utilizado para convertir las señales digitales en analógicas y viceversa, de modo tal que éstas puedan ser transmitidas de forma inteligible.

En las computadoras o dispositivos móviles es un periférico de entrada/salida que puede ser tanto interno como externo. Permitiendo conectar una línea telefónica al equipo y acceder a distintas redes, como Internet.

Concurrente: propiedad de los sistemas que permiten que múltiples procesos sean ejecutados al mismo tiempo, y que potencialmente puedan interactuar entre sí.

Granularidad: Describe el nivel de detalle de la base de datos en un DataWarehouse (almacén de datos).

Host: es un ordenador que funciona como el punto de inicio y final de las transferencias de datos. Más comúnmente descrito como el lugar donde reside un sitio web. Un host de Internet tiene una dirección de Internet única (dirección IP) y un nombre de dominio único o nombre de host.

Dual: Hace referencia a que un elemento se aplica para dos casos.

Excentricidad: Distancia que media entre el centro de la elipse y uno de sus focos.

Sideral (Tiempo): Es el tiempo determinado en base a la rotación aparente de las estrellas.

Así, el día sideral es el periodo de tiempo entre dos pasos sucesivos por el meridiano (o culminación) de una misma estrella; tiene una duración de 23h 56m 04s, inferior en 3m 56s con respecto al día solar.

El año sideral es el tiempo empleado por la Tierra en realizar una vuelta en su órbita con referencia a las estrellas fijas; tiene una duración de 365d 6h 9m 10s.

Anexo

Modelos ISO

El estándar ISO/IEC 9126

El estándar ISO9126 (2001), presenta un marco conceptual para el modelo de calidad y define un conjunto de características, refinadas en subcaracterísticas, las cuales debe cumplir todo producto software para ser considerado de calidad.

En relación al modelo de calidad del producto software, el estándar ISO/IEC 9126 (2001), está dividido en cuatro partes:

ISO/IEC 9126-1 (2001): Presenta un modelo de calidad del software, estructurado en características y subcaracterísticas.

- **ISO/IEC TR 9126-2 (2003):** Proporciona métricas externas para medir los atributos de seis características de calidad externa definidas en la ISO/IEC 9126-1 (2001) y una explicación de cómo aplicar las métricas de calidad de software.
- **ISO/IEC TR 9126-3 (2003):** Proporciona métricas internas para medir atributos de seis características de calidad interna definidas en la ISO/IEC 9126-1 (2001).
- **ISO/IEC TR 9126-4:** Define métricas de calidad en uso para medir los atributos definidos en la ISO/IEC 9126-1 (2001).

Sólo la primera parte de la norma ISO 9126-1 (2001) es un estándar aprobado y publicado, siendo los restantes informes que componen la parte identificada como Reportes Técnicos (Technical Report TR).

El estándar **ISO9126-1(2001)**, presenta dos modelos de calidad. La primera referida a la calidad interna y externa (Figura 12) y la segunda a la calidad en uso (Figura 13). A continuación se definen las características descritas en la ISO/IEC 9126 (2001) y citadas en Abrahão et al. (2001):

- **Usabilidad:** Capacidad del producto software de ser entendido, aprendido y usado por los usuarios bajo condiciones específicas.

- **Funcionalidad:** Capacidad del producto software de proporcionar funciones que ejecuten las necesidades explícitas e implícitas de los usuarios cuando el software es usado bajo condiciones específicas.
- **Confiabilidad:** Capacidad del producto software de mantener un nivel especificado de rendimiento cuando es usado bajo condiciones específicas.
- **Eficiencia:** Representa la relación entre el grado de rendimiento del sitio y la cantidad de recursos (tiempo, espacio, entre otros) usados bajo ciertas condiciones.
- **Mantenimiento:** Capacidad del producto software de ser modificado y probado.
- **Portabilidad:** Capacidad del producto software de ser transferido de un ambiente a otro.

3.1.1. Calidad Interna y externa

En Olsina et al. (2005) citado en Covella (2005), se sintetizan los enfoques de calidad interna y externa del producto software, en el estándar ISO9126-1(2001).

- **Calidad Interna:** Especificada por un modelo de calidad similar al modelo 9126. Puede ser medida y evaluada por medio de atributos estáticos de documentos tales como: i) Especificación de requerimientos, ii) Arquitectura o diseño, iii) Piezas de código fuente, entre otros. En etapas tempranas del ciclo de vida del software es posible medir, evaluar y controlar la calidad interna de estos productos. Sin embargo, asegurar la calidad interna no es generalmente suficiente para asegurar la calidad externa.
- **Calidad Externa:** Especificada también por un modelo de calidad similar al modelo 9126. Puede ser medida y evaluada por medio de propiedades dinámicas del código ejecutable en un sistema de computación, esto es, cuando un módulo o la aplicación completa es ejecutado en una computadora o en una red simulando lo más cercanamente posible un ambiente real. En fases tardías del ciclo de vida del software (principalmente en distintas etapas de testing o ya en estado operativo de un producto de software o aplicación Web), es posible medir, evaluar y controlar la calidad externa de estos productos ejecutables.

La calidad interna expuesta en ISO9126-1(2001), se define como “*la totalidad de atributos de un producto que determina su capacidad de satisfacer necesidades explícitas e implícitas cuando son usadas bajo condiciones específicas*”. Se define como calidad externa “el grado en la que un producto satisface necesidades explícitas e implícitas cuando se utiliza bajo condiciones especificadas” (Covella, 2005).

Para los modelos de calidad interna y externa, se mantuvieron en la revisión las seis características principales de calidad. Aún más, a nivel de subcaracterísticas se transformaron en prescriptivas en vez de informativas. Además, se añadieron nuevas subcaracterísticas y otras redefinidas en términos de “capacidad del software” para facilitar la interpretación de las mismas desde una perspectiva de calidad interna o de calidad externa (Covella, 2005).

Estándar ISO/IEC 25000:2005

Los aspectos más importantes en el desarrollo de software son la **calidad del producto y del proceso**. **ISO/IEC 25000, proporciona** una guía para el uso de las nuevas series de estándares internacionales, llamados Requisitos y Evaluación de Calidad de Productos de Software (SQuaRE). Constituyen una serie de **normas basadas en la ISO 9126 y en la ISO 14598**, y su objetivo principal es guiar el desarrollo de los productos de software con la especificación y evaluación de requisitos de calidad (Portal ISO 25000).

La familia ISO 25000 está **orientada al producto software**, permitiendo definir el modelo de calidad y el proceso a seguir para **evaluar dicho producto**.

La familia de normas SQuaRE está compuesta por 5 divisiones:

- i. ISO 2500n: Gestión de la calidad,
- ii. ISO 2501n: Modelo de calidad
- iii. ISO 2502n: Medida de la calidad
- iv. ISO 2503n: Requisitos de calidad
- v. ISO 2504n: Evaluación de la calidad.

El estándar **ISO/IEC 25000** (2005), contiene una explicación sobre el proceso de transición entre el estándar ISO/IEC 9126, las series 14598 y SQuaRE. También presenta información sobre cómo utilizar la norma ISO/IEC 9126 y la serie 14598 en su forma anterior. Ofrece términos y definiciones, modelos referencia, guía general, guías de división individual y los estándares para fines de especificación, planificación y gestión, medición y evaluación. [12]

TABLA DE ESTANDARES DE CODIFICACION EN JAVA

Sentencias de importación	<p>Tras la declaración del paquete se incluirán las sentencias de importación de los paquetes necesarios. Esta importación de paquetes obligatorios seguirá el siguiente orden:</p> <ol style="list-style-type: none"> 1. Paquetes del JDK de java. 2. Paquetes de utilidades no pertenecientes al JDK de Java, de frameworks de desarrollo o de proyectos opensource tales como apache, hibernate, springframework, etc. 3. Paquetes de la aplicación.
Declaraciones de clases e interfaces	<p>Comentario de documentación de la clase/interfaz <code>/** ... */</code> - Permite describir la clase/interfaz desarrollada. Necesario para generar la documentación de la api mediante javadoc.</p> <p>Comentario de implementación de la clase/interfaz, si es necesario <code>/* ... */</code> -Este comentario incluye cualquier información que no pueda incluirse en el comentario de documentación de la clase/interfaz.</p> <p>Variables de clase (estáticas) - En primer lugar las variables de clase públicas (public), después las protegidas (protected), posteriormente las de nivel de paquete (sin modificador), y por último las privadas (private).</p> <p>Variables de instancia - Primero las públicas (public), después las protegidas (protected), luego las de nivel de paquete (sin modificador), y finalmente las privadas (private).</p> <p>Métodos - Deben agruparse por funcionalidad en lugar de agruparse por ámbito o accesibilidad. Por ejemplo, un método privado puede estar situado entre dos métodos públicos. El objetivo es desarrollar código fácil de leer y comprender.</p>

Sangría	Como norma general se establecen 4 caracteres como unidad de sangría. Los entornos de desarrollo integrado (IDE) más populares, tales como Eclipse o NetBeans, incluyen facilidades para formatear código Java.
Longitud de línea	La longitud de línea no debe superar los 80 caracteres por motivos de visualización e impresión
División de líneas	<p>Cuando una expresión ocupe más de una línea, esta se podrá romper o dividir en función de los siguientes criterios:</p> <ol style="list-style-type: none"> 1. Tras una coma. 2. Antes de un operador. 3. Se recomienda las rupturas de nivel superior a las de nivel inferior. 4. Alinear la nueva línea con el inicio de la expresión al mismo nivel que la línea anterior. 5. Si las reglas anteriores generan código poco comprensible, entonces estableceremos tabulaciones de 8 espacios. <p>Ejemplos:</p> <pre>unMetodo(expresionLarga1, expresionLarga 2, expresionLarga 3, expresionLarga 4, expresionLarga 5); if ((condicion1 && condicion2) (condicion3 && condicion4) !(condicion5 && condicion6)) { unMetodo(); }</pre>
Comentarios de implementación	<p>Estos comentarios se utilizan para describir el código ("el cómo"), y en ellos se incluye información relacionada con la implementación, tales como descripción de la función de variables locales, fases lógicas de ejecución de un método, captura de excepciones, etc. Distinguimos tres tipos de comentarios de implementación:</p> <ol style="list-style-type: none"> 1. Comentarios de bloque: Permiten la descripción de ficheros, clases, bloques, estructuras de datos y algoritmos. <pre>/* * Esto es un comentario * de bloque */</pre> 2. Comentarios de línea: Son comentarios cortos localizados en una sola línea y tabulados al mismo nivel que el código que describen. Si ocupa más de una línea se utilizará un comentario de bloque. Deben estar precedidos por una línea en blanco. <pre>/* Esto es un comentario de línea */</pre>

	<p>// Esto es otro comentario de línea</p> <p>3. Comentario a final de línea: Comentario situado al final de una sentencia de código y en la misma línea.</p>
Declaraciones	<p>Una declaración por línea: Se recomienda el uso de una declaración por línea, promoviendo así el uso de comentarios.</p> <p>Inicialización: Toda variable local tendrá que ser inicializada en el momento de su declaración, salvo que su valor inicial dependa de algún valor que tenga que ser calculado previamente.</p> <p>Localización: Las declaraciones deben situarse al principio de cada bloque principal en el que se utilicen, y nunca en el momento de su uso.</p> <p>La única excepción a esta regla son los índices de los bucles "for", ya que, en Java, pueden incluirse dentro de la propia sentencia "for".</p> <pre>for (int i=0; contador<10; i++) { ... }</pre> <p>Se debe evitar el uso de declaraciones que oculten a otras declaraciones de ámbito superior.</p> <pre>int contador = 0; // Inicio del método public void unMetodo() { if (condicion) { int contador = 2; // ¡¡ EVITAR !! ... } }</pre> <p>Declaración de clases / interfaces: Durante el desarrollo de clases / interfaces se deben seguir las siguientes reglas de formato:</p> <p>No incluir ningún espacio entre el nombre del método y el paréntesis inicial del listado de parámetros.</p> <p>El carácter inicio de bloque ("{") debe aparecer al final de la línea que contiene la sentencia de declaración.</p> <p>1. El carácter fin de bloque ("}") se sitúa en una nueva línea tabulada al mismo nivel que su correspondiente sentencia de inicio de bloque, excepto cuando la sentencia sea nula, en tal caso se situará detrás de "{".</p>

Sentencias	<p>Cada línea debe contener como máximo una sentencia. Ejemplo,</p> <p>Las sentencias pertenecientes a un bloque de código estarán tabuladas un nivel más a la derecha con respecto a la sentencia que las contiene. El carácter inicio de bloque "{" debe situarse al final de la línea que inicia el bloque. El carácter final de bloque "}" debe situarse en una nueva línea tras la última línea del bloque y alineada con respecto al primer carácter de dicho bloque.</p> <p>Todas la sentencias de un bloque deben encerrarse entre llaves "{ ... }", aunque el bloque conste de una única sentencia. Esta práctica permite añadir código sin cometer errores accidentalmente al olvidar añadir las llaves.</p> <p>La sentencia "try/catch" siempre debe tener el formato siguiente,</p> <pre>try { sentencias; } catch (ClaseExcepción e) { sentencias; }</pre> <p>En el bloque "catch" siempre se imprimirá una traza de error indicando el tipo de excepción generada y posteriormente se elevará dicha excepción al código invocante, salvo que la lógica de ejecución de la aplicación no lo requiera.</p> <p>Siempre se utilizará el bloque "finally" para liberar recursos y para imprimir trazas de monitorización de fin de ejecución.</p> <pre>try { sentencias; } catch (ClaseExcepción e) { sentencias; } finally { sentencias; }</pre>
Espacios en blanco	<p>Las líneas y espacios en blanco mejoran la legibilidad del código permitiendo identificar las secciones de código relacionadas lógicamente. Se utilizarán espacios en blanco en los siguientes casos:</p> <ol style="list-style-type: none"> 1. Entre una palabra clave y un paréntesis. Esto permite que se distingan las llamadas a métodos de las palabras clave. 2. Tras cada coma en un listado de argumentos. Por ejemplo: objeto.unMetodo(a, b, c); 3. Para separar un operador binario de sus operandos, excepto en el caso del operador ("."). Nunca se utilizarán espacios entre los operadores unarios ("++" o "--") y sus operandos

	<p>4. Para separar las expresiones incluidas en la sentencia "for". Por ejemplo: for (expresion1; expresion2; expresion3)</p> <p>5. Al realizar el moldeado o "casting" de clases. Ejemplo: Unidad unidad = (Unidad) objeto;</p>
Paquetes	<p>Se escribirán siempre en letras minúsculas para evitar que entren en conflicto con los nombres de clases o interfaces.</p> <p>El prefijo del paquete siempre corresponderá a un nombre de dominio de primer nivel, tal como: es, eu, org, com, net, etc.</p> <p>El resto de componentes del paquete se nombrarán de acuerdo a las normas internas de organización de la empresa: departamento, proyecto, máquina, sección, organismo, área, etc.</p> <p>Ejemplos:</p> <p>es.provincia.organismo1.festivaldecine es.provincia.organismo2.vivienda</p> <p>java.util.ArrayList java.util.Date</p> <p>javax.servlet.http.HttpServletRequest javax.servlet.http.HttpServletResponse</p>
Clases e interfaces	<p>Los nombres de clases deben ser sustantivos y deben tener la primera letra en mayúsculas. Si el nombre es compuesto, cada palabra componente deberá comenzar con mayúsculas.</p> <p>Los nombres serán simples y descriptivos.</p> <p>Debe evitarse el uso de acrónimos o abreviaturas.</p> <p>Las interfaces se nombrarán siguiendo los mismos criterios que los indicados para las clases. Como norma general toda interfaz se nombrará con el prefijo "I" para diferenciarla de la clase que la implementa (que tendrá el mismo nombre sin el prefijo "I").</p> <p>Ejemplo: class Ciudadano</p>
Métodos	<p>Los métodos deben ser verbos escritos en minúsculas. Cuando el método esté compuesto por varias palabras cada una de ellas tendrá la primera letra en mayúsculas.</p> <p>Ejemplo: public void insertaUnidad(Unidad unidad);</p>

Variables	<p>Las variables se escribirán siempre en minúsculas. Las variables compuestas tendrán la primera letra de cada palabra componente en mayúsculas.</p> <p>Las variables nunca podrán comenzar con el carácter "_" o "\$". Los nombres de variables deben ser cortos y sus significados tienen que expresar con suficiente claridad la función que desempeñan en el código. Debe evitarse el uso de nombres de variables con un sólo carácter, excepto para variables temporales.</p> <p>Ejemplo: Unidad unidad;</p>
Constantes	<p>Todos los nombres de constantes tendrán que escribirse en mayúsculas. Cuando los nombres de constantes sean compuestos las palabras se separarán entre sí mediante el carácter de subrayado "_".</p> <p>Ejemplos:</p> <pre>int LONGITUD_MAXIMA; int LONGITUD_MINIMA;</pre> <p>Los valores constantes (literales) nunca aparecerán directamente en el código. Para designar dichos valores se utilizarán constantes escritas en mayúsculas y se declararán, según su ámbito de uso, o bien en una Clase de constantes creada para tal efecto, o bien en la clase donde sean utilizadas.</p> <p>Ejemplos:</p> <pre>// Uso incorrecto codigoErrorUsuarioEncontrado = 1; ... switch (error) { case codigoErrorUsuarioEncontrado: ... }</pre> <pre>// Uso correcto public final int CODIGOERROR_USUARIOENCONTRADO = 1; ... switch (error) { case CODIGOERROR_USUARIOENCONTRADO: ... }</pre>

<p>Visibilidad de atributos de instancia y de clase</p>	<p>Los atributos de instancia y de clase serán siempre privados, excepto cuando tengan que ser visibles en subclases herederas, en tales casos serán declarados como protegidos.</p> <p>El acceso a los atributos de una clase se realizará por medio de los métodos "get" y "set" correspondientes, incluso cuando el acceso a dichos atributos se realice en los métodos miembros de la clase.</p> <p>Ejemplo:</p> <pre>public class Unidad { private int id; private String nombre; public void actualizaUnidad(Unidad unidad) { this.setId(unidad.getId()); this.setNombre(unidad.getNombre()); } ... }</pre>
<p>Referencias a miembros de una clase</p>	<p>Evitar el uso de objetos para acceder a los miembros de una clase (atributos y métodos estáticos). Utilizaremos en su lugar el nombre de la clase. Por ejemplo</p> <pre>metodoUtilidad(); // Acceso desde la propia clase estática ClaseUtilidad.metodoUtilidad(); // Acceso común desde cualquier clase</pre>
<p>Asignación sobre variables</p>	<p>Se deben evitar las asignaciones de un mismo valor sobre múltiples variables en una misma sentencia, ya que dichas sentencias suelen ser difíciles de leer.</p> <pre>int a = b = c = 2; // Evitar</pre> <p>No utilizar el operador de asignación en aquellos lugares donde sea susceptible de confusión con el operador de igualdad. Por ejemplo:</p> <pre>// INCORRECTO if ((c = d++) == 0) { }</pre> <pre>// CORRECTO c = d++; if (c == 0) { }</pre> <p>No utilizar asignaciones embebidas o anidadas. Ejemplo:</p> <pre>c = (c = 3) + 4 + d; // Evitar</pre> <p>debería escribirse:</p> <pre>c = 3; c = c + 4 + d;</pre>

Paréntesis	<p>Es una buena práctica el uso de paréntesis en expresiones que incluyan distintos tipos de operadores para evitar problemas de precedencia de operadores. Aunque la precedencia de operadores nos pueda parecer clara, debemos asumir que otros programadores no tengan un conocimiento exhaustivo sobre las reglas de precedencia.</p> <p>Ejemplo:</p> <pre>if (w == x && y == z) // INCORRECTO if ((w == x) && (y == z)) // CORRECTO</pre>
Valores de retorno	<p>Los valores de retorno tendrán que ser simples y comprensibles, de acuerdo al propósito y comportamiento del objeto en el que se utilicen.</p> <pre>// INCORRECTO public boolean esProgramador(Empleado emp) { if (emp.getRol().equals(ROL_PROGRAMADOR)) { return true; } else { return false; } } // CORRECTO public boolean esProgramador(Empleado emp) { boolean esUnProgramador = false; if (emp.getRol().equals(ROL_PROGRAMADOR)) { esUnProgramador = true; } return esUnProgramador; }</pre>
Expresiones en el operador condicional ternario	<p>Toda expresión compuesta, por uno o más operadores binarios, situada en la parte condicional del operador ternario deberá ir entre paréntesis.</p> <p>Ejemplo:</p> <pre>(x >= y) ? x : y;</pre>
Comentarios especiales (TODO, FIXME, XXX)	<p>Utilizaremos XXX para comentar aquella porción de código que, aunque no tenga mal funcionamiento, requiera modificaciones.</p> <p>Usaremos FIXME para señalar un bloque de código erróneo que no funciona.</p> <p>Emplearemos TODO para comentar posibles mejoras de código, como pueden ser las optimizaciones, actualizaciones o refactorizaciones.</p>

PEAR: Estándares de desarrollo para PHP

PEAR es el acrónimo de PHP Extension Application Repository. Es parte del proyecto PHP y consiste en una biblioteca de extensión programada en lenguaje PHP que permite desarrollar aplicaciones, módulos y extensiones de alto nivel y calidad.

Estándar 1: La indentación (margen a la izquierda) debe ser a cuatro espacios sin caracteres de tabulación. Esto es debido a que ciertos IDE's de desarrollo introducen caracteres de tabulación cuando indentan un texto automáticamente. Se recomienda el uso de herramientas o editores generales como EMACS u otros.

Estándar 2: Las estructuras de control deben tener un espacio entre el **keyword** de la estructura y el signo de apertura de paréntesis para distinguir entre las llamadas de las funciones y el signo de llaves debe estar sobre la línea de la estructura.

Estándar 3: Las funciones deben ser llamadas sin espacios entre el nombre de la función, el signo de paréntesis y el primer parámetro; espacios entre cada coma por parámetro y sin espacios entre el ultimo paréntesis, el signo de paréntesis cerrado y el signo de punto y coma (;).

Estándar 4: El estilo de los comentarios debe ser como el estilo de comentarios para C (`/*` `*/` o `//`), no debe utilizarse el estilo de comentarios de Perl (`#`).

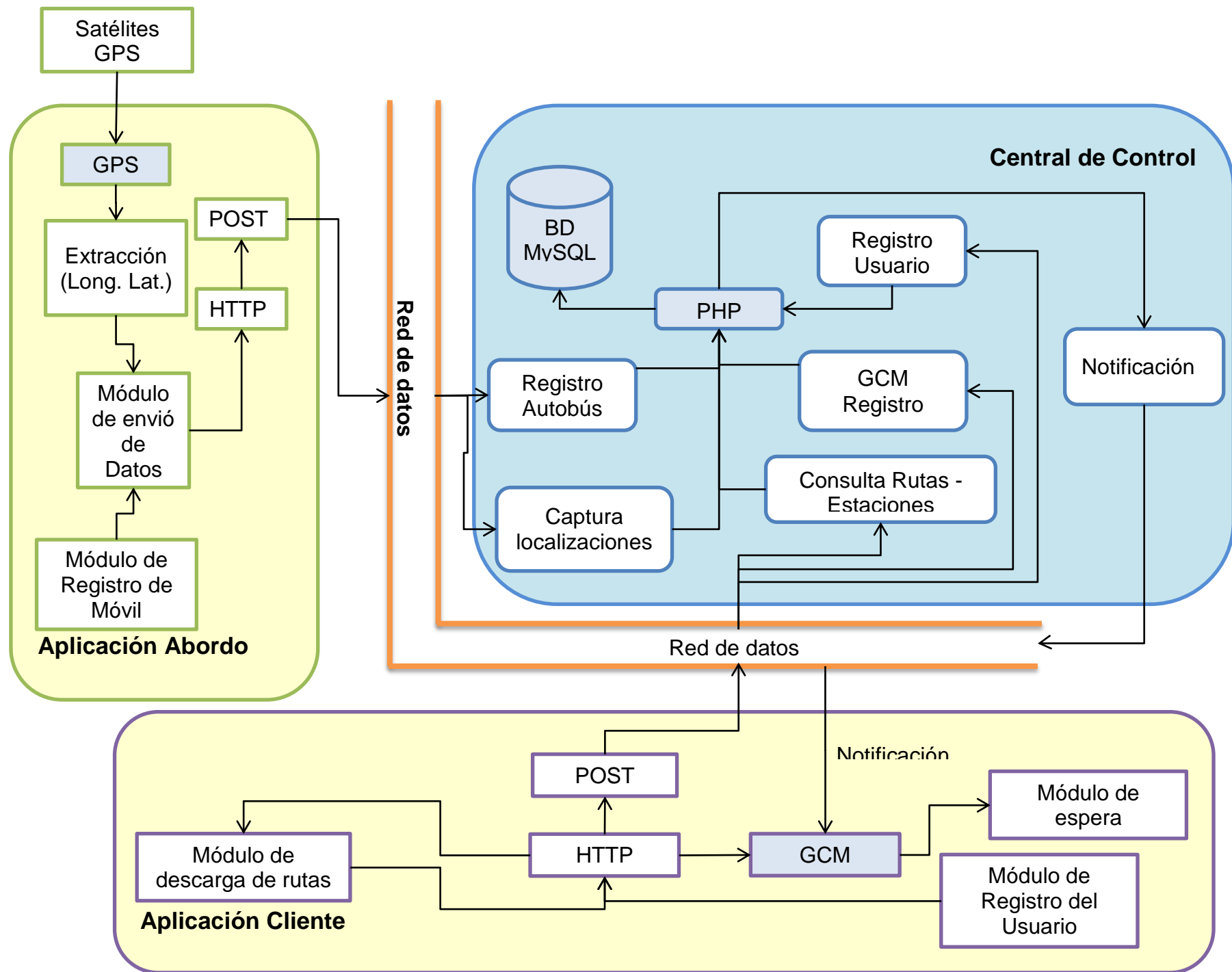
Estándar 5: Cuando se incluya un archivo de dependencia incondicionalmente utilice **require_once** y cuando sea condicionalmente, utilice **include_once**.

Estándar 6: siempre utilice las etiquetas `<?php ?>` para abrir un bloque de código. No utilice el método de etiquetas cortas, porque esto depende de las directivas de configuración en el archivo PHP.INI y hace que el script no sea tan portable.

Estándar 7: Los nombres de las clases deben de iniciar con letra mayúscula. Los nombres de las variables y de las funciones pueden iniciar con letra minúscula, pero si estas tienen más de una palabra, cada nueva palabra debe iniciar con letra mayúscula (el nombre puede escribirse separado por signos de guion mayor). Si una función, en una clase, es privada; deberá comenzar con el signo de guion mayor para una fácil identificación. Las

constantes deben de escribirse siempre en mayúsculas y tanto estas como las variables globales deben de tener como prefijo el nombre de la clase a la que pertenecen.

Estándar 8: Los archivos con código PHP, deben de ser guardados en formato ASCII utilizando la codificación **ISO-8859-1. (Actualizado)**. El formato ASCII con codificación ISO-8859-1 es el formato en que se guardan los archivos de texto plano (.txt). La razón de este estándar es que determinados editores HTML (en especial Dreamweaver), agregan códigos de carácter extraño de salto de línea (como si se tratara de un archivo binario) y esto puede ocasionar que el intérprete de PHP, encuentre problemas a la hora de leer el script.



Código del proyecto

La codificación del proyecto se puede encontrar en el versionador GitHub:

Servidor

<https://github.com/danielsanccarm/Server.git>

Aplicación del Cliente

https://github.com/danielsanccarm/NotiBus_ActionBarSherlock.git

Aplicación del Abordo

https://github.com/WilliamCalderon/Bus_4to.git

Matriz de Pruebas

Sistema de notificación de proximidad para transporte público en ruta Xochimilco/ Bosque de Nativitas - Metro San Lázaro, utilizando el Posicionamiento Global GPS y Google Cloud Messaging.

Generales

Nombre del caso de prueba	El objetivo	El escenario de prueba	Condición de la prueba
El sistema tiene redacción y ortografía correcta en la presentación	Evitar errores de ortografía y que la información sea clara y precisa para el usuario final.	Verificar en: App del autobús App del usuario "DatosBD.php"	(Si - Aprobado)

El sistema tiene los nombres de todas las rutas cargadas	Verificar la coherencia de la información de las rutas cargada en el sistema con la existente en la página de la red de transporte de pasajeros del D.F.	Verificar en: App del usuario Base de datos del sistema.	Si - Aprobado)
El sistema tiene todas las estaciones de la Ruta 39-A	Corroborar la existencia de todas las estaciones pertenecientes a la Ruta 39-A	Verificar en Base de datos del sistema.	Si - Aprobado)
Logotipo del sistema	Verificar si es la idea a transferir	Ver logo	(Si -Aprobado)
Capacidad del Servidor	Verificar cuantos usuarios pueden estar conectados en el servidor	Servidor	(Si - Aprobado) (No - Aprobado) Acción: ____

Aplicación Autobús

Nombre del caso de prueba	El objetivo	El escenario de prueba	Condición de la prueba
Prueba de funcionamiento en múltiples dispositivos.	Verificar el correcto funcionamiento de la aplicación en por lo menos 7 dispositivos con plataforma Android, diferentes versiones.	Manipular aplicación	(Si - Aprobado)
Reacción sin conexión a Internet	Evitar la pérdida del mensaje a enviar y entregar en el momento que se conecte a Internet	Ver reacción de la aplicación sin Internet y verificar que se pueda entregar el mensaje al servidor, en el momento que se conecte a Internet y no se pierda el mensaje en caso de falta de Internet.	(No - Aprobado) Acción: <u>Programar ciclo</u>

Tiempo de respuesta de obtención de posición	Tener la certeza de que se obtendrá la localización cada 30 segundos.	App del autobús.	(Si - Aprobado)
Diferentes tamaños de pantalla	Probar en diversos dispositivos la correcta vista de la aplicación	App del autobús	(Si - Aprobado)
Ejecución en background	Verificar que se envía la información al servidor, ejecutándose en background.	App del autobús.	(Si - Aprobado)
Duración de la batería	Verificar el funcionamiento con la batería con carga completa, carga media y carga con menos del 20%	App Autobus	(Si - Aprobado) (No - Aprobado) Acción: ____

Aplicación Cliente

Nombre del caso de prueba	El objetivo	El escenario de prueba	Condición de la prueba
Prueba de funcionamiento en múltiples dispositivos.	Verificar el correcto funcionamiento de la aplicación en por lo menos 7 dispositivos con plataforma Android, diferentes versiones.	Manipular aplicación	(Si - Aprobado)
Reacción sin conexión a Internet	Evitar la pérdida del mensaje a enviar y entregar en el momento que se conecte a Internet	Ver reacción de la aplicación sin Internet y verificar que se pueda entregar el mensaje al servidor, en el momento que se conecte a Internet y no se pierda el mensaje en caso de falta de Internet.	(Si - Aprobado)

Diferentes tamaños de pantalla	Probar en diversos dispositivos la correcta vista de la aplicación	App Usuario	(Si - Aprobado)
Ejecución en background	Verificar que se envía la información al servidor, ejecutándose en background.	App Usuario	(Si - Aprobado)
Notificación de inicio de descarga de las estaciones.	Informarle al usuario del envío de su solicitud de descarga.	App Usuario	(Si - Aprobado)
Espera de conexión servidor	Poner un límite de espera para la conexión con el servidor.	App Usuario	(Si - Aprobado)
Verificación memoria externa	Mostrar un mensaje en caso de no estar conectada la SD	App Usuario	(No - Aprobado) Acción: <u>Generar mensaje</u>
Presentación de la GUI cambiada por la versión de	Verificar una vista agradable en diferentes	App Usuario	Si - Aprobado)

Android	versiones de Android		
Funcionamiento de la Aplicación en Tablets	Verificar el correcto funcionamiento de la aplicación para tablets con servicio de datos y wi-fi	App usuario	(Si - Aprobado) (No - Aprobado) Acción: ____
Duración de la batería	Verificar el funcionamiento con la batería con carga completa, carga media y carga con menos del 20%	App Usuario	(Si - Aprobado) (No - Aprobado) Acción: ____