Instituto Politécnico Nacional

Escuela Superior de Ingeniería Mecánica Y Eléctrica

UNIDAD "CULHUACAN"

Sistema Móvil 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.

TESINA

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACIÓN

PRESENTAN

WILLIAM LUIS CALDERON PALOMINO

DANIEL SÁNCHEZ CARMONA

ASESORES:

ING. MIGUEL ÁNGEL MIRANDA HERNÁNDEZ

M. en C. IXCHEL MARTÍNEZ PIRÓ

MÉXICO, D.F. DICIEMBRE 2013

Índice

Planteamiento del problema

En la actualidad la manera de llevar el control de tiempos de llegada a las paradas, es por medio de registros hechos a mano por el personal de las rutas en una libreta o en hojas, esto para determinar las frecuencias de salida y llegada en puntos estratégicos, lo que genera tener tiempos promedio para cada unidad, lamentablemente, esta información únicamente es accesible para el personal, más no para los usuarios; muchas veces estos tiempos no se cumplen en exactitud a causa de factores externos no predecibles como lluvia, tráfico, etc. Lo cual causa que muchos usuarios se pregunten: “¿Por dónde está el camión?”.

Los usuarios del medio de transporte público RTP de la ruta Xochimilco/ Bosque de Nativitas a Metro San Lázaro por Cafetales, de manera común deben esperar lapsos de minutos indeterminados bajo situaciones de conflicto vial, las cuales pueden tornarse en horas esperando a que llegue la unidad, para ello sería cómodo estar informado de la ubicación de las unidades a partir de algún punto en específico.

Hoy en día la ciudad de México carece de una herramienta dirigida a los usuarios del transporte público en la que se informen acerca de las ubicaciones de las unidades de la ruta anteriormente mencionada, que brinde apoyo para facilitar la movilidad de las personas.

Objetivo General

Desarrollar un sistema de información que dé seguimiento por medio del Sistema de Posicionamiento Global GPS, a las unidades de la ruta de la RTP Xochimilco/Embarcadero de Nativitas a Metro San Lázaro por Cafetales, el cual le notifique al usuario en el momento en que pase una unidad por un punto de ascenso y descenso (parada) especificado por el mismo, por medio de mensajería en la nube de Google. Teniendo como función principal orientar al usuario sobre las precauciones (tiempo, distancia, etc.) que deberá tomar para poder abordar en el

punto y momento deseados, a través de una aplicación móvil en la plataforma Android.

Resumen

En este trabajo se presentara 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 jugara 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.

Dado que este tipo de transporte tiene paradas establecidas de ascenso y descenso establecidos, se le mostrará al usuario una lista de estos puntos donde seleccionara en cuál de ellos desea ser notificado de la presencia de una unidad. Por medio de una aplicación móvil desarrollada para dispositivos con Sistema Operativo Android (debido a que es soportado por un mayor número de equipos y las herramientas ligadas a su fabricante 'Google').

El sistema estará constituido de los componentes listados a continuación:

- **La aplicación móvil** en donde al usuario se le mostrara una lista con las paradas de la ruta y seleccionará una de ellas, se le preguntara si desea darle seguimiento, si su respuesta es afirmativa podrá seleccionar otra u otras estaciones para ser notificado; posteriormente enviara su petición al servidor haciendo uso de la herramienta de mensajería en nube de Google (Google Cloud Messaging - GCM), también se hará uso de esta herramienta para notificar al usuario.
- **Un GPS en el vehículo** (GPS de un Dispositivo móvil) que a través de una aplicación Android, se estará solicitando la posición del mismo y será enviado al servidor por medio de GCM.
- **El servidor o central de control** que tendrá una base de datos de las ubicaciones (latitud-longitud) de las paradas, estableciendo un rango de

alcance. Se hará uso de los datos alojados en la base de datos para compararlos con los datos obtenidos del GPS ubicado en el vehículo, al encontrar alguna coincidencia entre el datos se hará la relación con la herramienta GCM para notificar al usuario.

Justificación

Es cotidiano que las personas tengan la incertidumbre, de a qué hora va a llegar a la parada la unidad de transporte que ellos desean tomar. Esta aplicación beneficiara a todas esas personas que en muchas ocasiones han estado esperando a alguna unidad durante bastante tiempo, y han perdido la oportunidad de tomar alguna otra opción de transporte, también apoyara a la reducción de retrasos de los usuarios del transporte público. (Llevando un control de los tiempos, logrando con esto una estadística más real de tiempos)

Hipótesis

La aplicación se utilizara para eliminar la incertidumbre de: si es más conveniente esperar o no alguna unidad de transporte de la ruta “N” a utilizar por el usuario.

Estado del Arte

Con respecto a este apartado, se pueden mencionar las siguientes investigaciones científicas, que bien son una base y se consideraran necesarias tener en cuenta para realizar esta investigación:

- TESIS: “Sistema de localización y seguimiento de móviles terrestres utilizando el Sistema de Posicionamiento Global”

  Autor: Antonio Romero Rojano

  Fecha: Julio 2003

  Escuela: ESIME Zacatenco

En este trabajo el autor presenta el diseño, construcción en implementación de un Sistema que determinara la posición geográfica de una unidad terrestre, asi 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.

- Tesis: “Sistema de rastreabilidad GPS con interface Web y SMS para dispositivos móviles”.

  Autor: José Albertino Martínez Osorio.

  Escuela: Universidad Nacional Autónoma de México. Facultad de Ingeniería.

En este trabajo 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. Envió de posición a Internet
    5. Envió de posición vía SMS
- WEB:
    1. Recepción de la Posición
    2. Monitoreo de la Posición
- Carriers de Telefonía

Capítulo 1:

Marco Teórico

1. Sistemas de información

1.1. Sistema

Es un conjunto de partes que están integradas con el propósito de lograr un objetivo.

En los cuales los objetivos pueden ser intrínsecos y asignados.

Los **objetivos intrínsecos** son los propósitos que el sistema adquiere con su propia conformación y que constituye parte de su naturaleza. Generalmente se trata de propósitos muy básicos, derivados de la forma de interacción de sus partes.

Los **objetivos asignados** son los que se imponen al sistema o una modificación al mismo, para que realice las funciones necesarias a fin de lograr un objetivo.

1.2. 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.

1.2.1. 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 unida 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.

1.3. 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.

1.4. Clasificación de los Sistemas de Información

La clasificación de los sistemas de información se realiza teniendo en cuenta sus características similares.

Entre las clasificaciones se encuentran:

- **Por estructura organizacional:** Se clasifican a lo largo de líneas estructuradas. Dentro de estos se encuentran los sistemas para divisiones, departamentos, unidades de operación e incluso para empleados individuales.
- **Por área funcional:** para todas las tareas rutinarias o repetitivas que se desarrollan en la empresa y que son esenciales para operación de la organización. Ejemplo: sistema de información contable, sistemas de comercialización, sistemas de información de recursos humanos.
- **De acuerdo con la ayuda brindada:** apoyan a los gerentes en la toma de decisiones o a empleados administrativos al momento de entregar un informe, como son gráficas, tablas, etc.
- **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.

- **Sistemas Estratégicos**

Sus principales características son:

Su función primordial no es apoyar la automatización de procesos operativos ni proporcionar información para apoyar la toma de decisiones. Sin embargo, este tipo de sistemas puede llevar a cabo dichas funciones.

Suelen desarrollarse in house, es decir, dentro de la organización, por lo tanto no pueden adaptarse fácilmente a paquetes disponibles en el mercado.

Típicamente su forma de desarrollo es con base a incrementos y a través de su evolución dentro de la organización. Se inicia con un proceso o función en particular y a partir de ahí se van agregando nuevas funciones o procesos.

Su función es lograr ventajas que los, competidores no posean, tales como ventajas en costos y servicios diferenciados con clientes y proveedores. En este contexto, los Sistemas Estratégicos son creadores de barreras de entrada al negocio. Por ejemplo, el uso de cajeros automáticos en los bancos es un Sistema Estratégico, ya que brinda ventaja sobre un banco que no posee tal servicio. Si un banco nuevo decide abrir sus puertas al público, tendrá que dar este servicio para tener un nivel similar al de sus competidores.

Apoyan el proceso de innovación de productos y procesos dentro de la empresa, debido a que buscan ventajas respecto a los competidores y una forma de hacerlo es innovando o creando productos y procesos.

2. 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).

2.1. 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.

2.1.1. 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 característica 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 expectacular crecimiento, alcanzando a finales de año una cuota de mercado del 75%.

2.1.2. ¿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 herda 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.

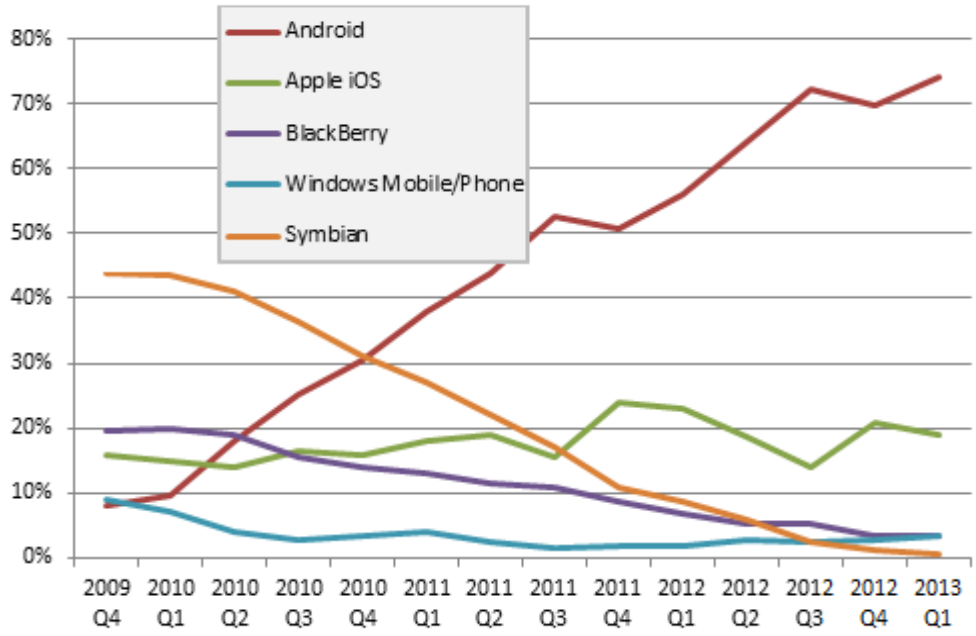
2.1.3. Comparativa con otras plataformas

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	Si	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 Comparativa de las principales plataformas móviles [11]

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 Gratner 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 despega; 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%. [11]



Gráfica 1 Porcentaje de teléfonos inteligentes vendidos según su sistema operativo [11]

2.1.4. 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.

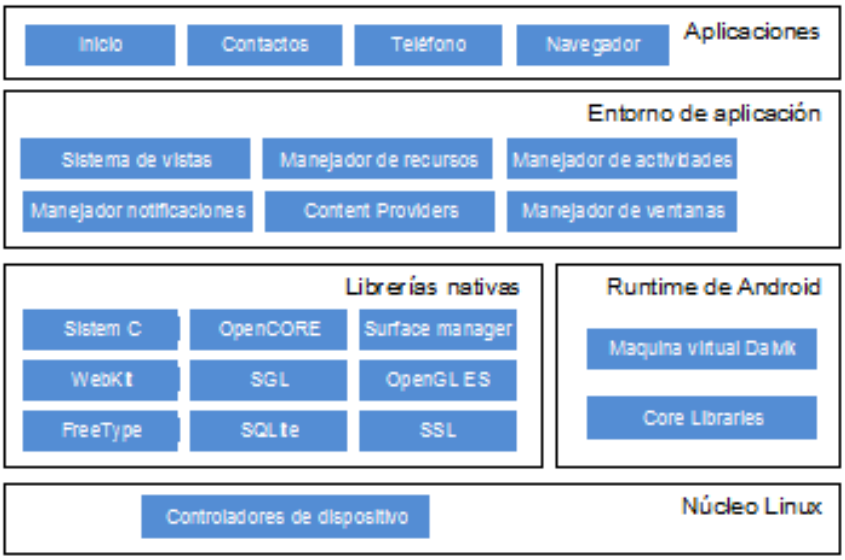


Figura 1 Arquitectura Android

2.1.4.1. 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.*

2.1.4.2. 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 *Runtine de Android* el "core libraries" con la mayoría de las librerías disponibles en el lenguaje Java.

2.1.4.3. 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 harware 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*.

2.1.4.4. 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).

2.1.4.4.1. 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.

3. Bases de Datos

Desde la aparición de los sistemas informáticos, una de las principales aplicaciones ha sido el almacenamiento y el tratamiento de grandes cantidades de datos para permitir su posterior consulta y utilización.

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.

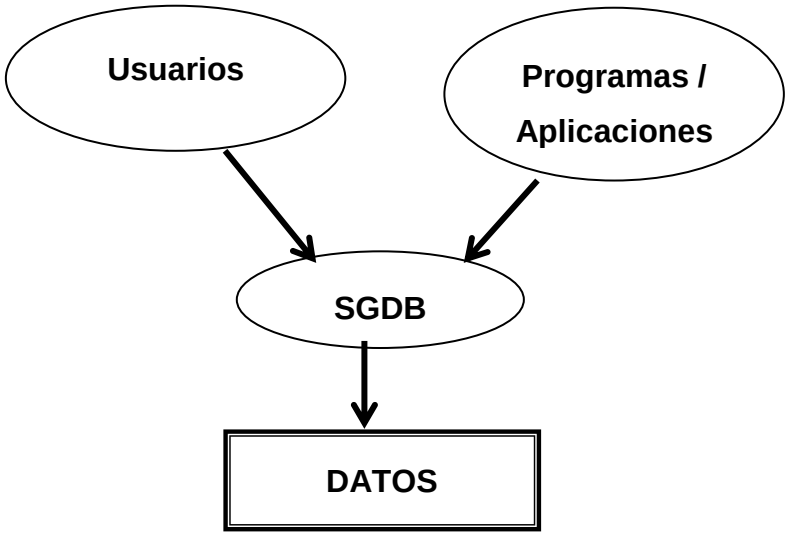


Figura 2 Sistema Gestor de Base de datos (SGDB). [3]

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”.

Una base de datos está formada por una secuencia de registros, entendiendo por “registro” el conjunto de información asociada a una entrada en la base de datos. Ahora bien, cada registro está constituido por varios datos que representan distintos aspectos o atributos de él y que se denominan campos.

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.

Otro concepto importante es el “tipo de datos”, que permite especificar el tipo de información que se va almacenar en un determinado campo de la base de datos: números, fechas, texto, cadenas, imágenes…

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.

Este concepto proporciona una mayor flexibilidad a la hora de ordenar y presentar los datos, a la vez que oculta los detalles reales de almacenamiento y manipulación: el

modelo no especifica exactamente como se representan físicamente las tablas, de manera que todo lo que ven los usuarios finales de la base de datos son filas y columnas.

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 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 univoca 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”.

3.1. Comparativa de los SGBD más utilizados

	CARACTERISTICAS	VENTAJAS	DESVENTAJAS
Oracle	- Entorno de cliente / Servidor - Gestión de grandes bases de datos - Usuarios concurrentes - Alto rendimiento - Gestión segura - Portabilidad - Compatibilidad	- Es muy usado a nivel Mundial - Puede ejecutarse en todas las plataformas - Permite el uso de particiones para mejorar su eficiencia - Un aceptable soporte	- Muy costoso - Mal configurado es extremadamente lento
SQL Server	- Nuevas herramientas integradas - Recuperación rápida - Mejoras en la recopilación - Aislamiento de imágenes	- Soporte de transacciones - Estabilidad, escalabilidad y seguridad. - Soporta procedimientos almacenados - Incluye un potente entorno grafico	- Utiliza mucha memoria RAM - No es gratuito - No es muy útil a la hora de las prácticas.
My SQL	- Interioridad y portabilidad - Escrito en C y C++ - Probado con un alto rango de compiladores diferentes - Funciona en diferentes plataformas - Proporciona un buen almacenamiento - Relativamente sencillo	- Es de código abierto - Bajo costo - Facilidad de configuración - Usa licencia GPL - Velocidad al realizar las operaciones.	- El soporte para disparadores es muy básico - El soporte para disparadores es muy básico - No soporta algunas conversiones de datos - Los privilegios de las tablas no se borran automáticamente

Tabla 2 Comparativa de los SGBD más utilizados

4. 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 maquina 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 Maquina), 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 maquina 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 maquina a medida que se va ejecutando la aplicación. A su vez, los “lenguajes compilados” son traducidos a código maquina 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 maquina 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.

4.1. 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.

4.1.1. 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.

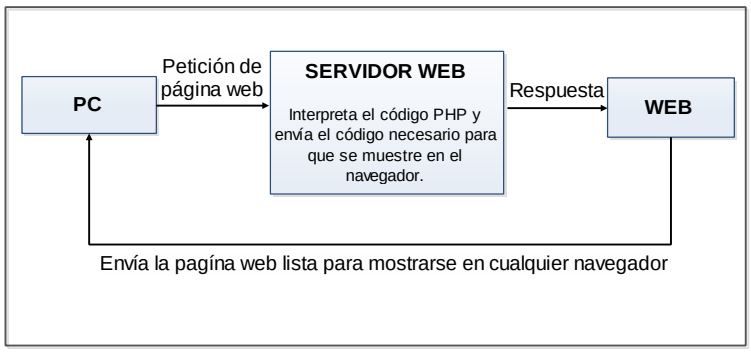


Figura 3 Proceso de visita a página PHP

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.

4.2. 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.[10]

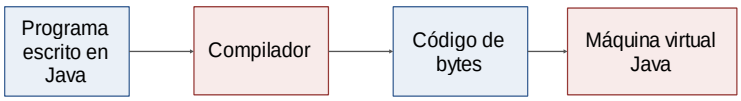


Figura 4 Proceso de ejecución de un programa en Java

5. Internet

5.1. ¿Qué es Internet?

Internet es un conjunto descentralizado de redes de comunicación interconectadas que utilizan la familia de protocolos TCP/IP, garantizando que las redes físicas heterogéneas que la componen funcionen como una red lógica única, de alcance mundial. Sus orígenes se remontan a 1969, cuando se estableció la primera conexión de computadoras, conocida como ARPANET, entre tres universidades en California y una en Utah, Estados Unidos.

Uno de los servicios que más éxito ha tenido en Internet ha sido la World Wide Web (WWW, o "la Web"), a tal grado que es habitual la confusión entre ambos términos. La WWW es un conjunto de protocolos que permite, de forma sencilla, la consulta remota de ficheros de hipertexto. Ésta fue desarrollo posterior (1990) y utiliza Internet como medio de transmisión.

Existen, por tanto, muchos otros servicios y protocolos en Internet, parte de la Web: el envío de correo electrónico (SMTP), la transmisión de ficheros (FTP y P2P), las conversaciones en línea (IRC), la mensajería instantánea y presencia, la transmisión de contenido y comunicación multimedia-telefonía (VoIP), la televisión (IPTV), los boletines electrónicos (NNTP), el acceso remoto a otros dispositivos (SSH y Telnet) o los juegos en línea.

5.2. Acceso a Internet

Internet incluye más de 100 protocolos distintos basados en TCP/IP, que se configuran como el protocolo de la red. Los servicios disponibles en la red mundial de PC, han avanzado mucho gracias a las nuevas tecnologías de transmisión de alta velocidad, como ADSL y Wireless, se ha logrado unir a las personas con videoconferencia, llamadas telefónicas gratuitas, ver imágenes por satélite, o

disfrutar de un juego multijugador en 3D, un buen libro PDF o eBook, o álbumes y películas para ver online o descargar.

El método de acceso a Internet vigente hace algunos años, la red telefónica básica (RTB), ha venido siendo sustituido gradualmente por conexiones más veloces y estables, entre ellas el ADSL, el RDSL, o conexión por cable de fibra óptica. También han aparecido formas de acceso a través de la red eléctrica, e incluso por satélite.

Y por otro lado 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.

5.3. El flujo de Información en Internet

5.3.1. 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.

5.3.2. Protocolo TCP/IP

Para intercambiar información entre computadoras es necesario desarrollar técnicas que regulen la transmisión de paquetes.

Dicho conjunto de normas se denomina protocolo. Hacia 1973 aparecieron los protocolos TCP e IP, utilizados ahora para controlar el flujo de datos en Internet.

El protocolo TCP, se encarga de fragmentar el mensaje emitido en paquetes. En el destino, se encarga de reorganizar los paquetes para formar de nuevo el mensaje, y entregarlo a la aplicación correspondiente.

El protocolo IP enruta los paquetes. Esto hace que los distintos paquetes que forman un mensaje pueden viajar por caminos diferentes hasta llegar al destino.

5.4. 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.

6. 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.[2]. 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 [2].

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 5). En un sistema distribuido cada máquina puede cumplir el rol de servidor para algunas tareas y el rol de cliente para otras.

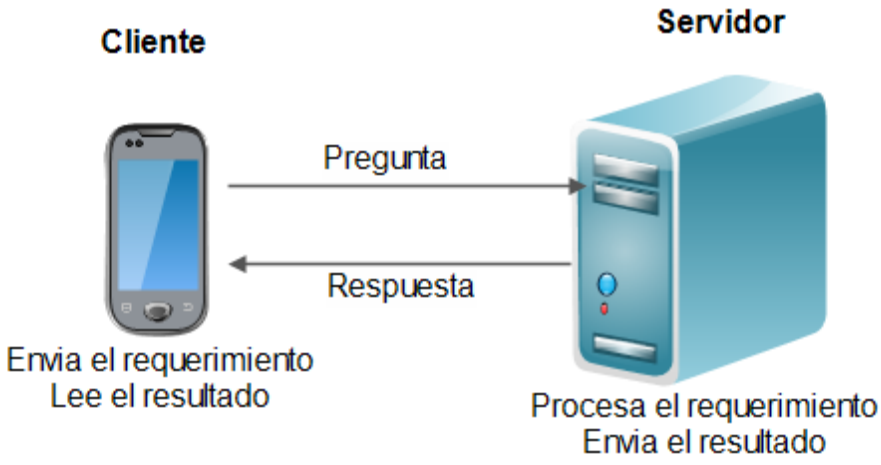


Figura 5 Modelo Cliente/Servidor

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

6.1. 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 [2].*

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

6.2. 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*[2].

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.

6.2.1. Tipos de servidores

Servidores de ficheros: Permiten el almacenamiento de diversas clases de ficheros para posteriormente enviárselas a otros clientes en la red.

Servidores de impresión: En un grupo de trabajo, llevan el control de una o varias impresoras, poniendo en cola de impresión aquello que solicitan los clientes de la red. Este servidor hace parecer que la impresora se encuentra conectada directamente a la computadora.

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.

DBMSs proporcionan acceso concurrente a los datos con varios niveles de granularidad de bloqueo y la integridad de los datos. DBMSs elimina la redundancia de datos, permitiendo la transparente distribución de datos de usuario, e incluso permite que las partes de sí mismo. Los clientes solicitan acceso a los datos deseados (contra el acceso de un servidor de ficheros para el fichero completo), y las manipulaciones necesarias sobre los datos requeridos se realizan en el servidor de

base de datos. Esto, permite que varios clientes puedan tener acceso a una base de datos al mismo tiempo.

Servidores de comunicación: En un entorno de grupo de trabajo que está conectado a un procesador host remoto, todo el software y hardware de comunicaciones pueden ser concentrados en un servidor de comunicación especial, al que los clientes pueden presentar sus peticiones de comunicación para el procesamiento.

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 fax: Realizan todas las actividades necesarias para que los faxes sean transmitidos, recibidos y distribuidos. Aquí se incluyen las tareas de envío, almacenamiento y recepción, entre otras.

Servidor del acceso remoto: Permiten la administración del acceso a internet en una determinada red. De esta forma, se puede negar el acceso a ciertos sitios web. Por otro lado, ofrece servicios de seguridad y controla las líneas de módem de los canales de comunicación de las redes para que las peticiones sean conectadas con las redes cuya posición es remota.

Servidor telnet: estos son los que admiten al usuario a entrar en una computadora huésped y hacer cualquier tipo de actividad como si estuviera trabajando directamente en esa computadora.

7. 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.

7.1. 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).

7.1.1. 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-urlcodificada	application/x-www-form-urlcodificada or multiparte/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 3 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.

7.2. Tipos de notificaciones

7.2.1. 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.

7.2.2. 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.

8. 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.

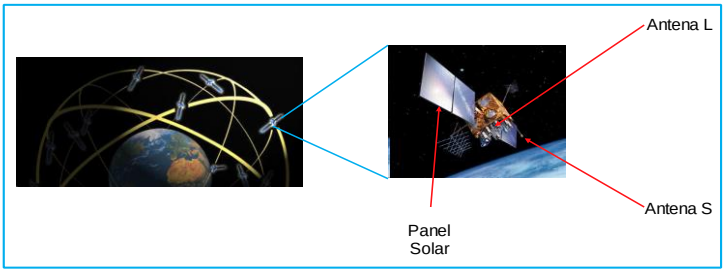


Figura 6 Constelaciones GPS

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).

8.1. Segmentos del GPS

El sistema GPS lo integran 3 segmentos: el segmento del espacio, el segmento del control y el segmento del usuario. Ver Figura 7, al segmento del espacio lo integra la constelación de 24 satélites.

El satélite GPS transmite señales de radio de microondas compuestas de 2 frecuencias portadoras (u ondas sinusoidales) moduladas por 2 códigos digitales y un mensaje de navegación. Las 2 frecuencias portadoras son generadas en 1,575.42 MHz (referidas como a la portadora L1) y 1,227.60 MHz (referidas como a la portadora L2). Las portadoras y los códigos son usados principalmente para determinar la distancia del usuario receptor a los satélites.

La disponibilidad de las frecuencias portadoras permite corregir un error GPS mayor, conocido como el retraso de ionosfera. Todos los satélites GPS transmiten en la misma frecuencia portadora L1 y L2. El código de modulación, sin embargo, es diferente para cada satélite, lo cual significa minimizar la señal de interferencia.

Los 2 códigos GPS son llamados (C/A- code) “Vasta Adquisición” (del Inglés Coarse Acquisition) y Precisión (o Código P). El código consiste en una ráfaga de datos binarios. Los códigos son comúnmente conocidos como códigos PRN porque parecen señales aleatorias (por ejemplo las señales de ruido), pero en realidad los códigos son generados usando un algoritmo matemático. Se tiene que el código C/A es modulado en la portadora L1 solamente, mientras que el código P es modulado en ambos portadoras L1 y L2.

El mensaje de navegación contiene además la información de las coordenadas aproximadas de los satélites como una función de tiempo. Las señales transmitidas son controladas con alta precisión por relojes atómicos situados en los satélites [7].

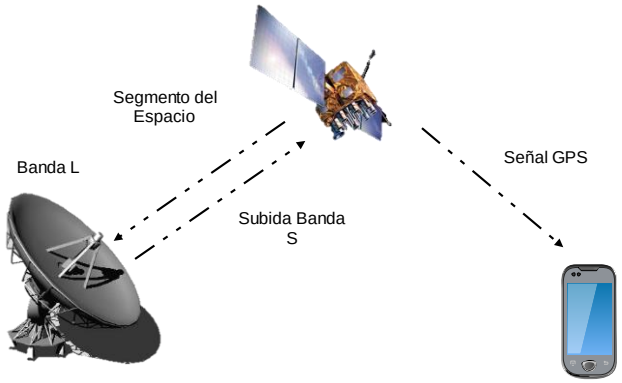


Figura 7 Segmento del GPS

El segmento de control del sistema GPS consiste en una red mundial de estaciones de rastreo, con una Estación Maestra de control (MCS de sus siglas en Ingles Master Control System) localizada en los Estados Unidos de América en Colorado Spring.

Su tarea principal de operación es el rastreo de satélites para determinar y predecir la localización de estos, la integración del sistema, y el comportamiento de los relojes atómicos de los satélites, datos atmosféricos, el almanaque del satélite y otras consideraciones, esta información es empaquetada y actualizada en los satélites a través de la banda S.

El uso del sistema GPS es actualmente permitido a todos los usuarios del mundo sin costo alguno.

8.2. Datos de Almanaque

Son los datos de la localización de los satélites y se tiene una unidad de almacenamiento en donde se les puede ubicar en cualquier momento. Algunas veces cuando la unidad de GPS esté fría “cold” (de la palabra en Ingles “cold”) éste podría apagarse por un largo período de tiempo, el almanaque puede estar fuera de actualización o “cold” (frío).

8.3. Sitios de Control

El segmento de control del GPS consiste en una Estación de Control Maestra, una red mundial de estaciones monitoras y estaciones de control terrestre. La Estación de control Maestra está localizada cerca de Colorado Spring, Colorado es el que realiza todo el proceso del segmento de control y es ocupado todo el tiempo.

Hay 5 estaciones monitoras, localizadas en Colorado Spring (con la estación de control maestra), Hawai, Kwajalein, Diego García y la Isla de Ascensión. Cada estación monitora es equipada con alta calidad de receptores GPS y un oscilador de Cesio para el continuo rastreo de todos los satélites GPS en vista.

8.4. La idea básica del GPS

La idea básica del GPS es simple, si la distancia de un punto de la Tierra (un recepto 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 8).

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.

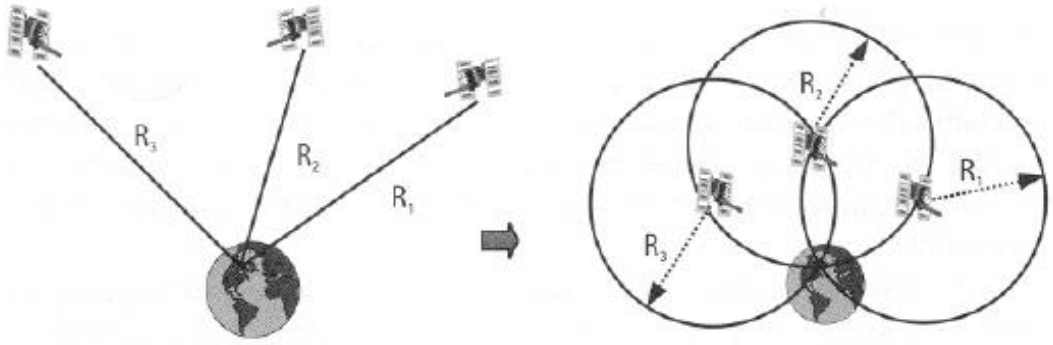


Figura 8 Posicionamiento Global idea básica

8.5. Medición del Pseudorango

El pseudorango es una medida del rango, o distancia entre el receptor y el satélite GPS, los rangos del receptor al satélite que son necesarios para el cálculo de la posición. Tanto el código P o el código C/A puede ser usado para medir el pseudorango. El procedimiento que determina la distancia del GPS o pseudorango puede ser descrito de la siguiente manera: suponer por un momento que ambos relojes del satélite y el receptor, los cuales generan la señal de control estén perfectamente sincronizados uno con otro, cuando el código PRN es transmitido del satélite, el receptor genera una réplica exacta de este código.

Después de un tiempo equivalente al tiempo de viaje de la señal en el espacio, el código transmitido será captado por el receptor. Comparando el código transmitido y su réplica, el receptor puede calcular el tiempo de viaje de la señal. Multiplicando el tiempo de viaje por la velocidad de la luz (299 729 458 m/seg.), se obtendrá el rango entre el satélite y el receptor.

Desafortunadamente, la suposición de que el receptor y el satélite tienen sincronizados los relojes no es verdadero. Realmente el rango de medida es contaminado con otros errores y predisposiciones como el error de sincronización entre el satélite y el reloj del GPS receptor. Por esta razón, la medición es referida al pseudorango. [7]

8.6. 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.**

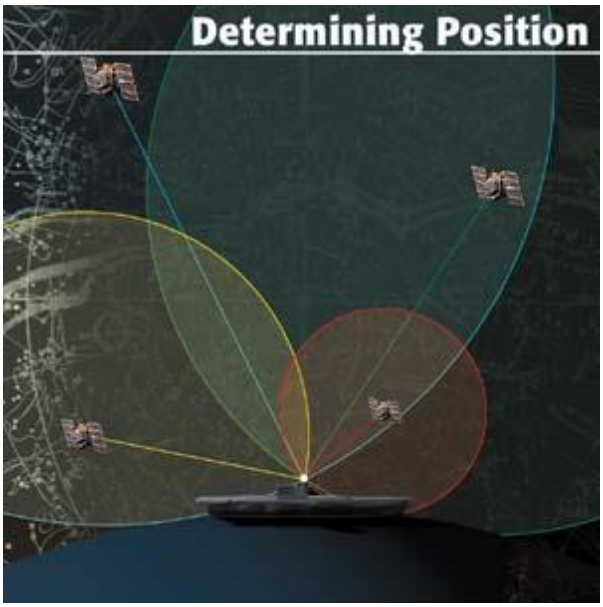


Figura 9 Localización con 4 satélites

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 1ns.

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 9.

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.

8.7. Cálculo de Posición y Tiempo

Una vez que el receptor comienza a localizar a los 4 satélites recibiendo las secuencias PRN de cada uno y generando los valores TDA, el procesador de datos se hace cargo. Este procesador de datos muestrea los valores TDA para cada uno de los 4 satélites y los multiplica por la velocidad de la luz para producir 4 mediciones pseudoaleatorias.

El procesador compensa las mediciones pseudoaleatorias para los errores determinísticos, incluyendo las diferencias en tiempo entre los relojes de satélites individuales y el GPS, la distorsión de las señales atmosféricas, efectos de relatividad y ruido del receptor internamente generado. El procesador de datos del receptor obtiene la información necesaria para realizar estas compensaciones del mensaje de datos de navegación.

Después, el procesador de datos resuelve las siguientes ecuaciones para calcular la posición/tiempo:

$$(X_1 - U_X)^2 + (Y_1 - U_Y)^2 + (Z_1 - U_Z)^2 = (PR_1 - CB * c)^2$$

$$(X_2 - U_X)^2 + (Y_2 - U_Y)^2 + (Z_2 - U_Z)^2 = (PR_2 - CB * c)^2$$

$$(X_3 - U_X)^2 + (Y_3 - U_Y)^2 + (Z_3 - U_Z)^2 = (PR_3 - CB * c)^2$$

$$(X_4 - U_X)^2 + (Y_4 - U_Y)^2 + (Z_4 - U_Z)^2 = (PR_4 - CB * c)^2$$

Ecuación 1 Calcular posición/tiempo

Dónde:

X_i, Y_i, Z_i = Posiciones del satélite (i=1,2,3,4). (Definida en el mensaje de navegación)

U_x,U_y, U_z = Posición del usuario

CB= Desviación del reloj receptor GPS.

Como se observa, estas ecuaciones son la representación de las cuatro esferas que rodean a los 4 satélites que se utilizan para determinar la posición exacta del usuario. La ecuación de la esfera es:

$$(X - X_0)^2 + (Y - Y_0)^2 + (Z - Z_0)^2 = r^2$$

Ecuación 2 De la Esfera

$P_0(X_0,Y_0,Z_0)$ es la posición del centro de la esfera. La distancia r se calcula en cada satélite con la siguiente expresión.

$$R = PR_i - CB * c$$

Donde:

PR_i = son las mediciones pseudoaleatorias de los 4 satélites (unidades de distancia).

CB = es de la desviación del reloj receptor GPS (Unidades de tiempo).

C = es la velocidad de la luz (distancia/tiempo).

9. 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 enviar datos desde sus servidores de sus aplicaciones Android en los dispositivos Android, y los mensajes serán transferidos del dispositivo del usuario a la nube.

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.

9.1. 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.

9.2. 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.

9.3. Arquitectura

Entes involucrados en GCM

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

Tabla 4 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	Información transmitida durante un intercambio de autenticación fuerte, que puede ser usada para autenticar su remitente. [9]
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 5 Credenciales GCM

Capitulo 2

Diseño del Sistema Móvil de notificación de proximidad para transporte.

1. Arquitectura propuesta para el sistema

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 paradas 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.

1.2. Hardware

1.2.1. El usuario

- Dispositivo móvil
- Conexión a Internet

1.2.2. El Vehículo

- Dispositivo Móvil
- GPS
- Conexión a Internet

1.3. Software

1.3.1. Sistema Operativo

1.3.1.1. Para desarrollo:

Linux Lubuntu 13.10,

Windows 7 Home Premium o Windows 8.1 OEM.

1.3.1.2. Para pruebas:

Apartir 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.

1.4. Metodología

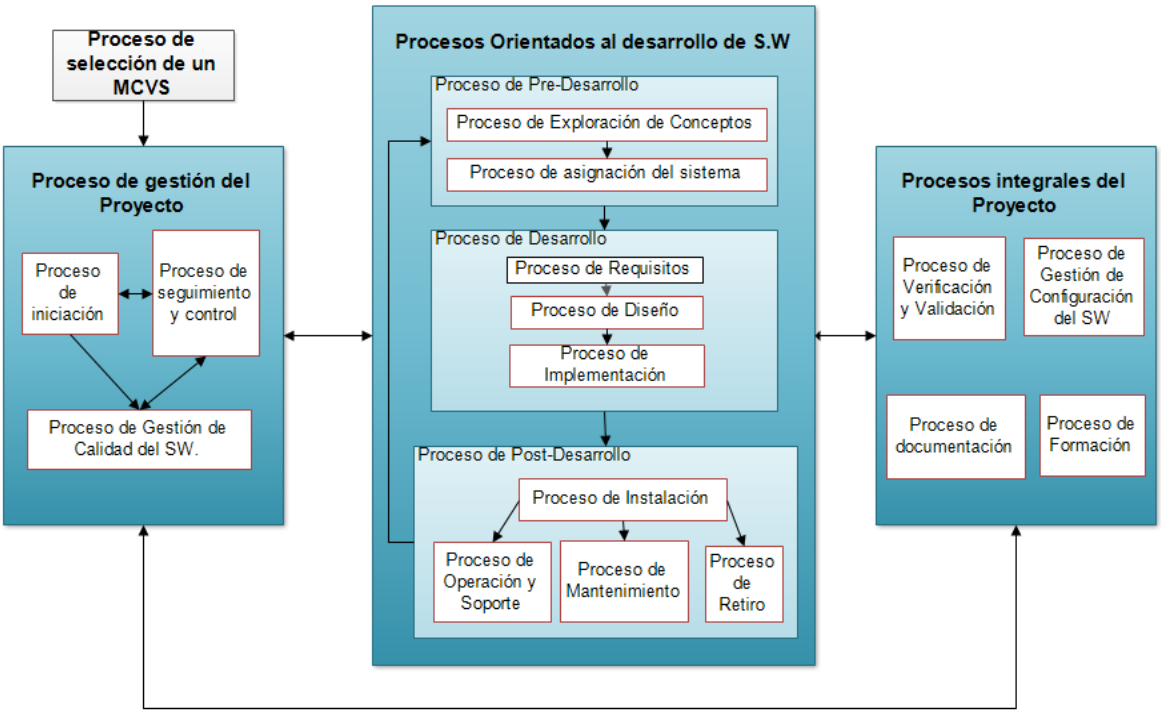


Figura 10 Modelo de procesos software IEEE

1.4.1. Ciclo de vida

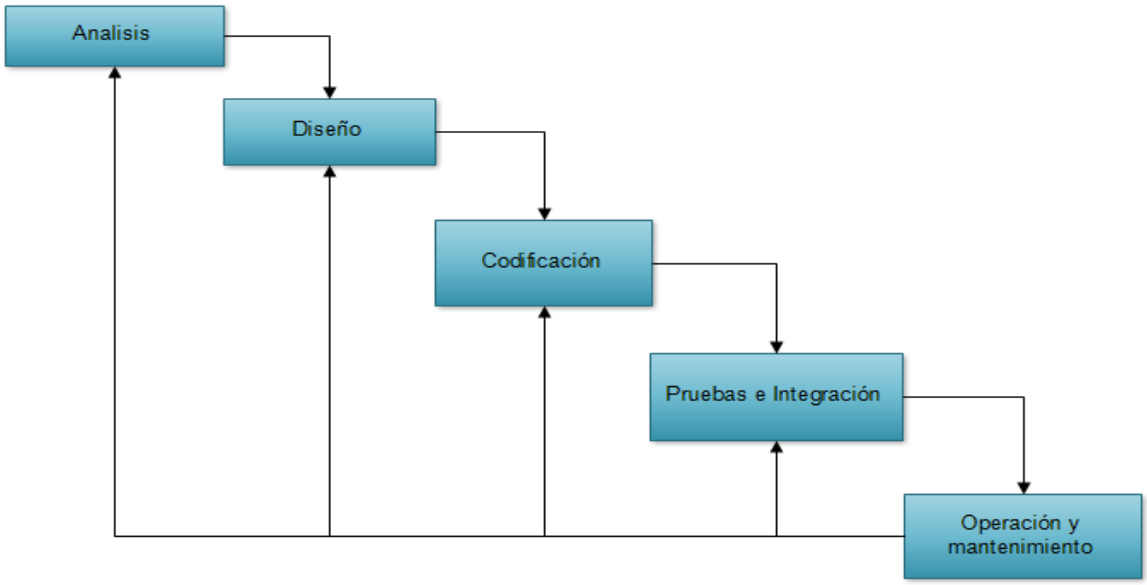


Figura 11 Ciclo de vida en cascada. Mejoramiento por pasos

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.

1.5. Estándares

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]

Referencias

[1]. 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

[2]. 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

[3]. Gómez Vieites,Álvaro. Suárez Rey, Carlos. Sistemas de información “Herramientas prácticas para la gestión”. Edit. Alfaomega 3ª Edición. Junio 2011.

[4]. Tomás Gironés, Jesus. “El gran libro de Android”. Edit. Alfaomega. 3ª Edición. Septiembre 2013.

[5]. Berson, Alex. “Client/Server Architecture”. Edit. McGraw-Hill. 1992

[6]. Tipos de servidores.

2013. Consultado octubre 2013

. http://www.tiposde.org/informatica/131-tipos-de-servidores/

[7]. Kaplan, E. Understanding GPS: Principles and Applications, Edit. Norwood. MA: Artech House. 1990

[8]. Project 2 GPS

Consultado octubre 2013.

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

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

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

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

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

Consultado diciembre 2013.

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

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 moviles 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 descriptas 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 es 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. Aun más, a nivel de subcaracterísticas se transformaron en prescriptitas 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).

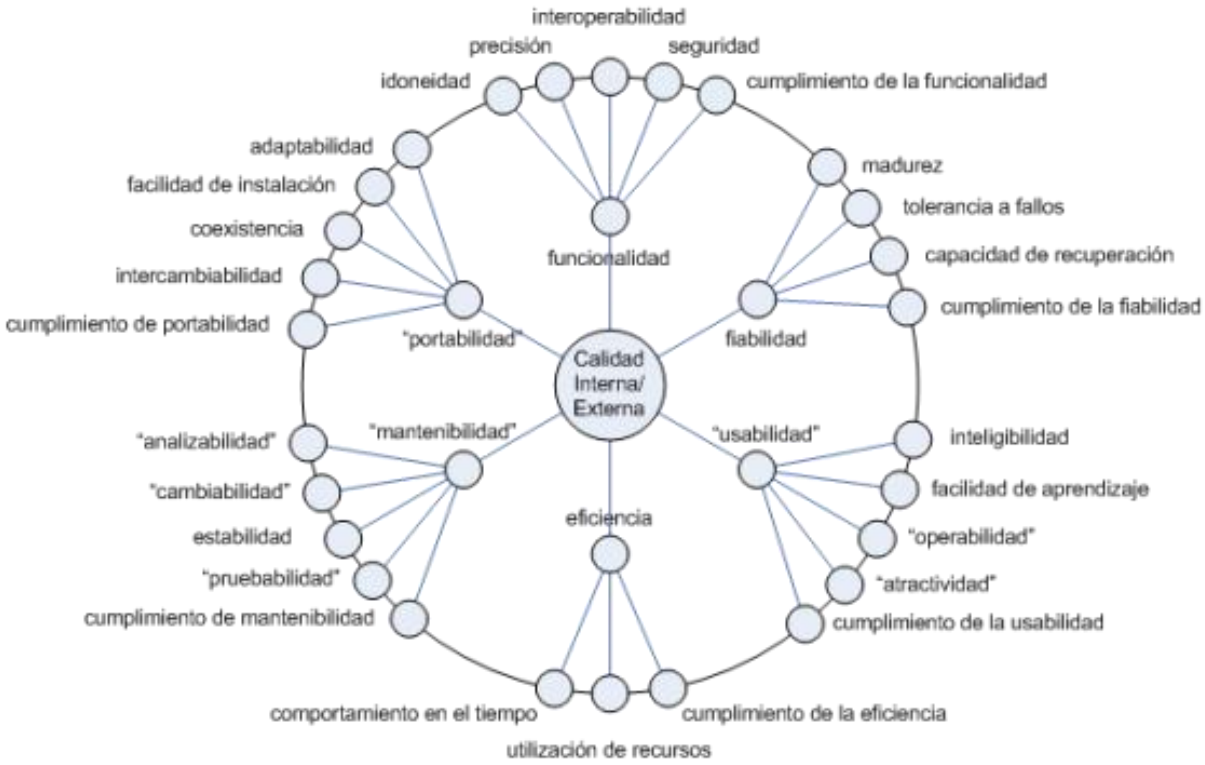


Figura 12 Características de la Calidad según la ISO/IEC 9126-1 (Fuente: Portal ISO 25000).

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	Tras la declaración del paquete se incluirán las sentencias de importación de los paquetes necesarios. Esta importación de paquetes obligatorios seguirá el siguiente orden: 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	**Comentario de documentación de la clase/interfaz /** ... */** - Permite describir la clase/interfaz desarrollada. Necesario para generar la documentación de la api mediante javadoc. **Comentario de implementación de la clase/interfaz, si es necesario /* ... */ -** Este comentario incluye cualquier información que no pueda incluirse en el comentario de documentación de la clase/interfaz. **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). **Variables de instancia -** Primero las públicas (public), después las protegidas (protected), luego las de nivel de paquete (sin modificador), y finalmente las privadas (private). **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.
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

<table>
<tr>
<td>División de líneas</td>
<td>Cuando una expresión ocupe más de una línea, esta se podrá romper o dividir en función de los siguientes criterios:
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.
Ejemplos:
unMetodo(expresionLarga1, expresionLarga 2, expresionLarga 3,
expresionLarga 4, expresionLarga 5);

if ((condicion1 && condicion2)
|| (condicion3 && condicion4)
||!(condicion5 && condicion6)) {
unMetodo();
}</td>
</tr>
<tr>
<td>Comentarios de implementación</td>
<td>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:
1. Comentarios de bloque: Permiten la descripción de ficheros, clases, bloques, estructuras de datos y algoritmos.
/*
* Esto es un comentario
* de bloque
*/
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.
/* Esto es un comentario de línea */
// Esto es otro comentario de línea
3. Comentario a final de línea: Comentario situado al final de una</td>
</tr>
</table>

	sentencia de código y en la misma línea.

Declaraciones	**Una declaración por línea:** Se recomienda el uso de una declaración por línea, promoviendo así el uso de comentarios.
	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.
	Localización: Las declaraciones deben situarse al principio de cada bloque principal en el que se utilicen, y nunca en el momento de su uso.
	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".
	for (int i=0; contador<10; i++) {
	...
	}
	Se debe evitar el uso de declaraciones que oculten a otras declaraciones de ámbito superior.
	int contador = 0; // Inicio del método
	public void unMetodo() {
	if (condicion) {
	int contador = 2; // ¡¡ EVITAR !!
	...
	}
	...
	}
	Declaración de clases / interfaces: Durante el desarrollo de clases / interfaces se deben seguir las siguientes reglas de formateo:
	No incluir ningún espacio entre el nombre del método y el paréntesis inicial del listado de parámetros.
	El carácter inicio de bloque ("{") debe aparecer al final de la línea que contiene la sentencia de declaración.
	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 "{".

<table>
<tr>
<td>Sentencias</td>
<td>
Cada línea debe contener como máximo una sentencia. Ejemplo,

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.

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.

La sentencia "try/catch" siempre debe tener el formato siguiente,

<pre>try {
   sentencias;
} catch (ClaseException e) {
   sentencias;
}</pre>

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.

Siempre se utilizará el bloque "finally" para liberar recursos y para imprimir trazas de monitorización de fin de ejecución.

<pre>try {
   sentencias;
} catch (ClaseException e) {
   sentencias;
} finally {
   sentencias;
}</pre>
</td>
</tr>
<tr>
<td>Espacios en blanco</td>
<td>
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:

1. Entre una palabra clave y un paréntesis. Esto permite que se
</td>
</tr>
</table>

	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 4. Para separar las expresiones incluidas en la sentencia "for". Por ejemplo:. for (expresion1; expresion2; expresion3) 5. Al realizar el moldeo o "casting" de clases. Ejemplo: Unidad unidad = (Unidad) objeto;
Paquetes	Se escribirán siempre en letras minúsculas para evitar que entren en conflicto con los nombres de clases o interfaces. El prefijo del paquete siempre corresponderá a un nombre de dominio de primer nivel, tal como: es, eu, org, com, net, etc. 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. . Ejemplos: es.provincia.organismo1.festivaldecine es.provincia.organismo2.vivienda java.util.ArrayList java.util.Date javax.servlet.http.HttpServletRequest javax.servlet.http.HttpServletResponse
Clases e interfaces	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. Los nombres serán simples y descriptivos. Debe evitarse el uso de acrónimos o abreviaturas.

	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"). Ejemplo: class Ciudadano
Métodos	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. Ejemplo: public void insertaUnidad(Unidad unidad);
Variables	Las variables se escribirán siempre en minúsculas. Las variables compuestas tendrán la primera letra de cada palabra componente en mayúsculas. 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. Ejemplo: Unidad unidad;

Constantes	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 "_".
	Ejemplos:
	int LONGITUD_MAXIMA;
	int LONGITUD_MINIMA;
	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.
	Ejemplos:
	// Uso incorrecto
	codigoErrorUsuarioEncontrado = 1;

	...
	switch (error) {
	  case codigoErrorUsuarioEncontrado:
	  ...
	}
	
	// Uso correcto
	public final int CODIGOERROR_USUARIOENCONTRADO = 1;
	...
	switch (error) {
	  case CODIGOERROR_USUARIOENCONTRADO:
	  ...
	}

Visibilidad de atributos de instancia y de clase	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.
	
	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.
	
	Ejemplo:
	
	public class Unidad {
	
	private int id;
	private String nombre;
	...
	
	public void actualizaUnidad(Unidad unidad) {
	this.setId(unidad.getId());
	this.setNombre(unidad.getNombre());
	}
	...
	}

Referencias a miembros de una clase	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 metodoUtilidad(); // Acceso desde la propia clase estática

	ClaseUtilidad.metodoUtilidad(); // Acceso común desde cualquier clase

Asignación sobre variables	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.
	
	int a = b = c = 2; // **Evitar**
	
	No utilizar el operador de asignación en aquellos lugares donde sea susceptible de confusión con el operador de igualdad. Por ejemplo:
	
	// INCORRECTO
	if ((c = d++) == 0) { }
	
	// CORRECTO
	c = d++;
	if (c == 0) { }
	
	No utilizar asignaciones embebidas o anidadas. Ejemplo:
	
	c = (c = 3) + 4 + d; **// Evitar**
	
	debería escribirse:
	c = 3;
	c = c + 4 + d;

Paréntesis	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.
	
	Ejemplo:
	if (w == x && y == z) // INCORRECTO
	if ((w == x) && (y == z)) // CORRECTO

Valores de retorno	Los valores de retorno tendrán que ser simples y comprensibles, de acuerdo al propósito y comportamiento del objeto en el que se utilicen.
	
	// 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;
	}

Expresiones en el operador condicional ternario	Toda expresión compuesta, por uno o más operadores binarios, situada en la parte condicional del operador ternario deberá ir entre paréntesis. Ejemplo: (x >= y) ? x : y;
Comentarios especiales (TODO, FIXME, XXX)	Utilizaremos XXX para comentar aquella porción de código que, aunque no tenga mal funcionamiento, requiera modificaciones. Usaremos FIXME para señalar un bloque de código erróneo que no funciona. Emplearemos TODO para comentar posibles mejoras de código, como pueden ser las optimizaciones, actualizaciones o refactorizaciones.

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 indendación (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 indendan 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 de 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.