



**ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO**  
**FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**  
**ESCUELA DE INGENIERÍA EN SISTEMAS**

**TEMA:**

**“ESTUDIO ESTADÍSTICO DE TIPOS DE SERVIDORES PARA LA  
COMUNICACIÓN CON PLATAFORMAS DE RASTREO MÓVIL EN LA  
IMPLEMENTACIÓN DE UNA PUERTA DE ENLACE PARA EQUIPOS  
ENFORA”**

**TESIS DE GRADO**

**Previa obtención del título de:**  
**INGENIERO EN SISTEMAS INFORMÁTICOS**

**PRESENTADO POR:**

**ANGEL MARCELO MONTESDEOCA CHACHA**

**RIOBAMBA – ECUADOR**

**2011**

En este trabajo se recopila los conocimientos adquiridos en el transcurso de la carrera, por lo que quiero agradecer a la ESPOCH institución que me ha dado la oportunidad de formarme como profesional a través de sus maestros que han sabido depositar sus conocimientos, de igual manera al Ing. Jorge Menéndez Director quien me ha sabido guiar para culminar con éxito este proyecto.

***Angel Marcelo Montesdeoca Chacha***

Este trabajo va dedicado a mis padres por su apoyo incondicional en el transcurso de mi vida y de mi carrera, a mi hermana y mis tíos por estar siempre a mi lado, a mi hijo Cristofer por ser la fuente de inspiración para culminar mi carrera, así como a mis amigos con quienes comparti buenos y malos momentos dentro de las aulas.

***Angel Marcelo Montesdeoca Chacha.***

**FIRMAS RESPONSABLES Y NOTA**

<b>NOMBRE</b>	<b>FIRMA</b>	<b>FECHA</b>
Ing. Iván Menes <b>DECANO FACULTAD INFORMÁTICA Y ELECTRÓNICA</b>	.....	.....
Ing. Raúl Rosero <b>DIRECTOR ESCUELA INGENIERÍA EN SISTEMAS</b>	.....	.....
Ing. Jorge Menéndez <b>DIRECTOR DE TESIS</b>	.....	.....
Ing. Wladimir Castro <b>MIEMBRO DEL TRIBUNAL</b>	.....	.....
Tlgo. Carlos Rodríguez <b>DIRECTOR CENTRO DE DOCUMENTACIÓN</b>	.....	.....
<b>NOTA DE LA TESIS:</b>	.....	

## **RESPONSABILIDAD DEL AUTOR**

“Yo Angel Marcelo Montesdeoca Chacha, soy el responsable de las ideas, doctrinas y resultados expuestos en esta Tesis de Grado, y el patrimonio intelectual de la misma pertenece a la Escuela Superior Politécnica de Chimborazo”

**FIRMA:**

-----

Angel Marcelo Montesdeoca Chacha

## ÍNDICE DE ABREVIATURAS

<b>ASCII</b>	American Standard Code for Information Interchange Código Estadounidense Estándar para el Intercambio de Información
<b>AT</b>	Attention Command
<b>AVL</b>	Automatic Vehicle Location Localización Automática de Vehículos
<b>CPU</b>	Central Processing Unit Unidad Central de Procesamiento
<b>CRC</b>	Class, Responsibilities and Collaboration
<b>FTP</b>	File Transfer Protocol Protocolo de Transferencia de Archivos
<b>GPS</b>	Global Positioning System Sistema de Posicionamiento Global
<b>GUI</b>	Graphic User Interface Interfaz Gráfica de Usuario
<b>RDS</b>	Rastreo Directo Satelital
<b>SIG o GIS</b>	Sistemas de Información Geográfica Geographic Information System
<b>UDP</b>	User Datagram Protocol
<b>XP</b>	eXtreme Programming Programación Extrema

## ÍNDICE GENERAL

RESPONSABILIDAD DEL AUTOR	
ÍNDICE DE ABREVIATURAS	
ÍNDICE GENERAL	
ÍNDICE DE FIGURAS	
ÍNDICE DE TABLAS	
INTRODUCCIÓN	
CAPÍTULO I	
MARCO REFERENCIAL.....	14
1.1 ANTECEDENTES .....	14
1.2 JUSTIFICACIÓN DEL PROYECTO .....	16
1.3 OBJETIVOS .....	17
1.3.1 OBJETIVO GENERAL.....	18
1.3.2 OBJETIVOS ESPECÍFICOS .....	18
1.4 HIPÓTESIS .....	18
CAPÍTULO II	
MARCO TEÓRICO .....	19
2.1 Introducción.....	19
2.2 Servidores .....	19
2.2.1 Clasificación de servidores .....	21
2.3 Modelo cliente- servidor.....	21
2.3.1 Cliente.....	21
2.3.2 Características de una arquitectura cliente – servidor .....	22
2.4 Multiplataforma .....	23
2.4.1 Sistemas Operativos.....	24
2.4.2 Lenguajes de programación .....	30
2.5 Creación de scripts en Linux .....	39
2.5.1 ¿Qué es el SHELL SCRIPT?.....	39
2.5.2 ¿Cómo ejecutar un SCRIPT?.....	40
2.5.3 Código de un SCRIPT .....	40
2.5.4 Depuración.....	42
2.5.5 Estructuras condicionales .....	43

2.5.6 Comando test .....	48
2.6 Demonios o Servicios .....	51
2.6.1 Historia de los demonios.....	51
2.6.2 Características de los demonios.....	52
2.6.3 Iniciando demonios manualmente, directorio init.d .....	53
2.6.4 Comando service.....	55
2.6.5 Iniciando servicios desde el arranque del sistema .....	56
2.6.6 Comando chkconfig.....	58
2.7 Hebras .....	60
2.7.1 Hebras en java.....	61
2.7.2 Estado de una hebra .....	62
2.8 Concurrencia .....	64
2.9 Servicios no Orientados a la Conexión.....	65
2.10 Multihebras.....	67
2.11 Multiprocesos.....	68
2.12 Georeferenciación.....	70
2.13 Sistemas de Información Geográfica (SIG).....	70
2.13.1 ¿Que es un SIG? .....	71
2.13.2 Importancia de los SIG .....	73
2.13.3 Componentes de los SIG.....	73
2.13.4 Funcionamiento de los SIG.....	77
2.14 Sistemas de Gestión de Bases de Datos.....	79
2.14.1 Objetivos de los Sistemas de Gestión de Bases de Datos.....	79
2.14.2 Ventajas .....	81
2.14.3 Inconvenientes .....	81
2.14.4 Productos SGBD.....	82
2.14.5 MySQL .....	84
2.14.5.1 Lenguajes de Programación compatibles con MySQL.....	85
2.14.5.2 Plataformas .....	85
<b>CAPÍTULO III</b>	
<b>ANÁLISIS COMPARATIVO ENTRE LOS TIPOS DE SERVIDORES.....</b>	<b>87</b>
3.1 Introducción .....	87
3.2 Determinar los parámetros a comparar .....	88



3.3 Desarrollo de los prototipos de pruebas.....	88
3.4 Análisis Comparativo .....	89
3.5 Puntajes Alcanzados .....	90
3.6 Interpretación .....	97
3.7 Resultado del análisis.....	99
3.8 Comprobación de la Hipótesis.....	99
CAPÍTULO IV	
IMPLEMENTACIÓN DEL SERVIDOR .....	102
4.1 Introducción .....	102
4.2 Planificación .....	102
4.2.1. Descripción del Sistema.....	103
4.2.2 Requerimientos .....	103
4.2.3 Historias de Usuario.....	105
4.2.4 Iteraciones .....	105
4.2.5 Velocidad del proyecto .....	107
4.3 Diseño .....	107
4.3.1 Diseños simples .....	107
4.3.2 Riesgos.....	108
4.3.3 Funcionalidad extra.....	108
4.3.4 Tarjetas C.R.C.....	109
4.4 Codificación.....	109
4.5 Pruebas.....	110
4.5.1 Pruebas de aceptación.....	110
CONCLUSIONES	
RECOMENDACIONES	
RESUMEN	
SUMMARY	
GLOSARIO DE TÉRMINOS	
BIBLIOGRAFÍA	
ANEXOS	

## ÍNDICE DE FIGURAS

Figura II. 1 Logo de Linux.....	28
Figura II. 2 Logo de Microsoft Windows .....	29
Figura II. 3 Logo de Mac OS .....	30
Figura II. 4 Inicia el servicio Samba desde una ruta determinada.....	54
Figura II. 5 Apaga el servicio Samba desde una ruta determinada.....	54
Figura II. 6 Detiene el servicio Samba con el comando service.....	55
Figura II. 7 Inicia un servicio con el comando service .....	56
Figura II. 8 Estados de un thread .....	63
Figura II. 9 Funcionamiento de los hilo.....	67
Figura II. 10 Elementos que intervienen en un SIG. ....	74
Figura II. 11 Esquema componentes básicos de un Sistema de Información Geográfica. ...	77
Figura II. 12 Modelo Raster.....	78
Figura III. 13 Comparación consumo de CPU .....	98
Figura III. 14 Comparación consumo de Memoria RAM .....	99

## ÍNDICE DE TABLAS

Tabla II. I Características Servicios no Orientados a la Conexión .....	66
Tabla III. II Datos Consumo CPU enviando 10000 paquetes/hora .....	91
Tabla III. III Datos Consumo CPU enviando 20000 paquetes/hora .....	91
Tabla III. IV Datos Consumo CPU enviando 40000 paquetes/hora.....	91
Tabla III. V Datos Consumo CPU enviando 60000 paquetes/hora .....	92
Tabla III. VI Datos Consumo CPU enviando 80000 paquetes/hora.....	92
Tabla III. VII Datos Consumo Memoria RAM enviando 10000 paquetes/hora .....	93
Tabla III. VIII Datos Consumo Memoria RAM enviando 20000 paquetes/hora .....	93
Tabla III. IX Datos Consumo Memoria RAM enviando 40000 paquetes/hora.....	93
Tabla III. X Datos Consumo Memoria RAM enviando 60000 paquetes/hora .....	94
Tabla III. XI Datos Consumo Memoria RAM enviando 80000 paquetes/hora.....	94
Tabla III. XII Datos Multihebras .....	96
Tabla III. XIII Datos Multiproceso.....	97
Tabla III. XIV Operacionalización conceptual.....	100
Tabla III. XV Operacionalización metodológica.....	100
Tabla III. XVI Datos Finales.....	101

# INTRODUCCIÓN

En la actualidad es de gran utilidad tener un conocimiento del lugar exacto (ubicación, en cualquier instante de tiempo) en el que se encuentra un bien que nos pertenece como por ejemplo un vehículo, por lo que la empresa de Rastreo Directo Satelital (RDS) que radica en la ciudad de Ambato, se dedica a ofrecer el servicio Localización Satelital por medio de los dispositivos de rastreo móvil que presentan el inconveniente de tener que pagar una licencia por cada N clientes que registran su información, por protocolo UDP, en la puerta de enlace (gateway) que la fábrica proporciona, lo que implica mayores gastos operacionales, lo cual se podría solucionar desarrollando una puerta de enlace local. Por lo que se realizará “El estudio estadístico de los tipos de servidores no orientados a la conexión, multihebras y multiprocesos, permitirá seleccionar el más eficiente para el desarrollo de una puerta de enlace con los equipos de rastreo móvil de Enfora.”

Para la implementación de la puerta de enlace se desarrollará el estudio de los siguientes puntos:

- Estudiar las características de las formas de implementar servidores no orientados a la conexión por multihebras y multiprocesos.
- Definir los parámetros de comparación estadística de los tipos de servidores (multihebras y multiprocesos).
- Realizar el estudio estadístico entre los dos tipos de servidores no orientados a la conexión por multihebras y multiprocesos.
- Implementar la puerta de enlace de los dispositivos de rastreo móvil de Enfora.

La metodología con la que se desarrollará la investigación es la metodología Ágil XP la misma que fue utilizada por su rapidez y relativa facilidad para ser implementada.

Los resultados que se obtengan con la investigación entre las dos formas de implementar la Puerta de Enlace (multihebras, multiprocesos) y los parámetros establecidos para el estudio estadístico (consumo de CPU y memoria RAM) indicarán que forma de implementación consume una menor cantidad de recursos hardware con la que se culminará la investigación y su implementación.

# **CAPÍTULO I**

## **MARCO REFERENCIAL**

### **1.1 ANTECEDENTES**

En un entorno móvil como el actual, aprovechar el valor de la ubicación geográfica se ha convertido en una herramienta clave para obtener beneficio de la información que puede ser de vital importancia para las compañías y es una tecnología que se está utilizando ya en muchas partes del planeta. La tecnología de geolocalización se basa en los Sistemas de Información Geográfica (GIS por sus siglas en inglés) para la gestión, análisis y visualización de conocimiento geográfico [1].

La geolocalización es beneficiosa para el sector automotriz, ya que al instalar un Sistema de Posicionamiento Global (GPS por sus siglas en inglés) en sus vehículos, éste puede emitir información por el protocolo UDP cada cierto tiempo (la cual va a contener datos como ubicación, velocidad, etc.) hacia una puerta de enlace, permitiendo disponer de la ubicación geoespacial de los mismos en cualquier instante de tiempo.

---

<sup>1</sup> <http://www.idg.es/computerworld/articulo.asp?id=167772>

Una puerta de enlace es un servicio que cumple funciones como el de intermediario para transmitir paquetes desde una aplicación cliente hacia una aplicación servidor para su procesamiento y/o almacenamiento, concepto que se utilizará para la investigación dado que se maneja éste mismo término para especificar al equipo que traduce un protocolo de red a otro [<sup>2</sup>].

La empresa de Rastreo Directo Satelital (RDS) que radica en la ciudad de Ambato, se dedica a ofrecer el servicio Localización Satelital por medio de los dispositivos de rastreo móvil (AVL) cuatribanda Enfora MT que presentan el inconveniente de tener que pagar una licencia por cada N clientes que registran su información, por protocolo UDP, en la puerta de enlace que la fábrica proporciona, lo que implica mayores gastos operacionales, lo cual se podría solucionar desarrollando una puerta de enlace local.

De la investigación preliminar realizada y bajo un análisis minucioso finalmente se logró seleccionar dos formas de implementar la Puerta de Enlace, a continuación se da una pequeña instrucción de las mismas:

- **Multihebra.-** Considerando el entorno multithread (multihilo), cada thread (hilo, flujo de control del programa) representa un proceso individual ejecutándose en un sistema. A veces se les llama procesos ligeros o contextos de ejecución.

---

<sup>2</sup> [http://es.wikipedia.org/wiki/Puerta\\_de\\_enlace](http://es.wikipedia.org/wiki/Puerta_de_enlace)

- **Multiproceso.-** El multiproceso se refiere a dos programas que se ejecutan a la vez, bajo el control del Sistema Operativo. Los programas no necesitan tener relación unos con otros, simplemente el hecho de que el usuario desee que se ejecuten a la vez [<sup>3</sup>].

Con el objetivo de desarrollar una puerta de enlace de código abierto se propone el siguiente tema de tesis titulado “Estudio estadístico de tipos de servidores para la comunicación con plataformas de rastreo móvil en la implementación de una Puerta de Enlace para equipos Enfora”.

## 1.2 JUSTIFICACIÓN DEL PROYECTO

Con el “Estudio estadístico de tipos de servidores para la comunicación con plataformas de rastreo móvil en la implementación de una Puerta de Enlace para equipos Enfora” se analizará cual es el tipo de servidor más eficiente, desde el punto de vista hardware (consumo de cpu y memoria), para la implementación de la puerta de enlace. A través de pruebas estadísticas que permitan comparar el uso de memoria y cpu para desarrollar una puerta de enlace que no tendría límites lógicos en el número de clientes que envíen su información.

Se implementará una aplicación (puerta de enlace) la misma que cumplirá la función de receptor la información que emitan los GPS (equipos Enfora) para su procesamiento y/o almacenamiento.

---

<sup>3</sup> <http://www.itapizaco.edu.mx/paginas/JavaTut/froufe/parte10/cap10-1.html>



Para la administración de la puerta de enlace se desarrollará una aplicación la cual asumirá funcionalidades como la de enviar peticiones al GPS para que realice ciertas operaciones en el vehículo (bloqueo, desbloqueo, apagado), mostrar información emitida por un GPS en particular.

La puerta de enlace será desarrollada utilizando el lenguaje de programación Java, con lo cual se logrará un servidor multiplataforma (podrá instalar en varios sistemas operativos como Windows, Linux, etc.). De la misma forma para el almacenamiento de la información se utilizará el motor de base de datos MySQL, además a esto, tanto Java como MySQL no tendrán un costo adicional a la empresa por la utilización de licencias por ser software libre.

La metodología, para la planificación y ejecución del sistema de administración, será Programación eXtrema (XP por sus siglas en inglés) ya que permite comunicación, rapidez, simplicidad en el desarrollo de la aplicación con una excelente calidad en el producto final.

Con el desarrollo de este proyecto la empresa podrá atender las peticiones de un mayor número de usuarios sin ningún tipo de restricción lógica ni económica.

### **1.3 OBJETIVOS**

A continuación se detalla el objetivo general y los objetivos específicos de la tesis.

### **1.3.1 OBJETIVO GENERAL**

Estudiar estadísticamente los servidores no orientados a la conexión, multihebras y multiprocesos, para determinar el más eficiente y desarrollar una puerta de enlace para equipos Enfora.

### **1.3.2 OBJETIVOS ESPECÍFICOS**

- Estudiar las características de las formas de implementar servidores no orientados a la conexión por multihebras y multiprocesos.
- Definir los parámetros de comparación estadística de los tipos de servidores (multihebras y multihilos).
- Realizar el estudio estadístico entre los dos tipos de servidores no orientados a la conexión por multihebras y multiprocesos.
- Implementar la puerta de enlace de los dispositivos de rastreo móvil de Enfora.

### **1.4 HIPÓTESIS**

El estudio estadístico de los tipos de servidores no orientados a la conexión, multihebras y multiprocesos, permitirá seleccionar el más eficiente para el desarrollo de una puerta de enlace con los equipos de rastreo móvil de Enfora.

# **CAPÍTULO II**

## **MARCO TEÓRICO**

### **2.1 Introducción**

Este capítulo abarca información que es necesaria para la comprensión de la investigación propuesta, la misma que permitirá sentar bases sólidas en el desarrollo, implementación y culminación de este proyecto.

Se introducen conceptos importantes para el tema a investigar como servidores, multiplataforma, creación de scripts en Linux, demonios o servicios, multihebras, georeferenciación, sistemas de información geográfica entre otros, los cuales son esenciales en el proyecto y se detallan a continuación.

### **2.2 Servidores**

En informática, un servidor es un tipo de software que realiza ciertas tareas en nombre de los usuarios. El término servidor ahora también se utiliza para referirse al ordenador físico

en el cual funciona ese software, una máquina cuyo propósito es proveer datos de modo que otras máquinas puedan utilizar esos datos [<sup>4</sup>].

También se suele denominar con la palabra servidor a: Una aplicación informática o programa que realiza algunas tareas en beneficio de otras aplicaciones llamadas clientes. Algunos servicios habituales son los servicios de archivos, que permiten a los usuarios almacenar y acceder a los archivos de una computadora y los servicios de aplicaciones, que realizan tareas en beneficio directo del usuario final. Este es el significado original del término. Es posible que un ordenador cumpla simultáneamente las funciones de cliente y de servidor.

Ejemplo de un Servidor. Un servidor no es necesariamente una máquina de última generación de grandes proporciones, no es necesariamente un superordenador; un servidor puede ser desde una computadora vieja, hasta una máquina sumamente potente (ej.: servidores web, bases de datos grandes, etc.). Todo esto depende del uso que se le dé al servidor. Si usted lo desea, puede convertir al equipo desde el cual usted está leyendo esto en un servidor instalando un programa que trabaje por la red y a la que los usuarios de su red ingresen a través de un programa de servidor web como Apache.

Por lo cual se logra llegar a la conclusión de que un servidor también puede ser un proceso que entrega información o sirve a otro proceso [<sup>5</sup>].

---

<sup>4</sup> <http://www.masadelante.com/faqs/servidor>.

<sup>5</sup> <http://es.wikipedia.org/wiki/Servidor>

### **2.2.1 Clasificación de servidores**

Para el estudio de los servidores se los ha clasificado por la forma de diseño de los mismos:

- Servidor Multihebra.- Considerando el entorno multithread (multihilo), cada thread (hilo, flujo de control del programa) representa un proceso individual ejecutándose en un sistema. A veces se les llama procesos ligeros o contextos de ejecución.
- Servidor Multiproceso.- El multiproceso se refiere a dos programas que se ejecutan a la vez, bajo el control del Sistema Operativo. Los programas no necesitan tener relación unos con otros, simplemente el hecho de que el usuario desee que se ejecuten a la vez.

### **2.3 Modelo cliente- servidor**

Para que la comunicación entre dos aplicaciones se lleve a cabo uno de los programas de aplicación debe estar esperando por requerimientos por parte del programa llamador.

Este modelo, un programa espera pasivamente y el otro inicia la comunicación se conoce como el paradigma de interacción cliente servidor.

La aplicación que espera pasivamente es llamada SERVIDOR y la que inicia el contacto es llamada CLIENTE.

#### **2.3.1 Cliente**

El cliente es una aplicación informática que se utiliza para acceder a los servicios que ofrece un servidor, normalmente a través de una red de telecomunicaciones.

El término se usó inicialmente para los llamados terminales tontos, dispositivos que no eran capaces de ejecutar programas por sí mismos, pero podían conectarse a un ordenador

central y dejar que éste realizase todas las operaciones requeridas, mostrando luego los resultados al usuario. Se utilizaban sobre todo porque su coste en esos momentos era mucho menor que el de un ordenador [6].

### **2.3.2 Características de una arquitectura cliente – servidor**

Las características básicas de una arquitectura Cliente/Servidor son:

- Combinación de un cliente que interactúa con el usuario, y un servidor que interactúa con los recursos compartidos. El proceso del cliente proporciona la interfaz entre el usuario y el resto del sistema. El proceso del servidor actúa como un motor de software que maneja recursos compartidos tales como bases de datos, impresoras, módems, etc.
- Las tareas del cliente y del servidor tienen diferentes requerimientos en cuanto a recursos de cómputo como velocidad del procesador, memoria, velocidad, capacidades del disco y dispositivos de entrada-salida.
- Se establece una relación entre procesos distintos, los cuales pueden ser ejecutados en la misma máquina o en máquinas diferentes distribuidas a lo largo de la red.
- Existe una clara distinción de funciones basada en el concepto de "servicio", que se establece entre clientes y servidores.
- La relación establecida puede ser de muchos a uno, en la que un servidor puede dar servicio a muchos clientes, regulando su acceso a recursos compartidos.
- Los clientes corresponden a procesos activos en cuanto a que son éstos los que hacen peticiones de servicios a los servidores. Estos últimos tienen un carácter pasivo ya que esperan las peticiones de los clientes.

---

<sup>6</sup> [http://es.wikipedia.org/wiki/Cliente\\_\(inform%C3%A1tica\)](http://es.wikipedia.org/wiki/Cliente_(inform%C3%A1tica))

- No existe otra relación entre clientes y servidores que no sea la que se establece a través del intercambio de mensajes entre ambos. El mensaje es el mecanismo para la petición y entrega de solicitudes de servicio.
- El ambiente es heterogéneo. La plataforma de hardware y el sistema operativo del cliente y del servidor no son siempre la misma. Precisamente una de las principales ventajas de esta arquitectura es la posibilidad de conectar clientes y servidores independientemente de sus plataformas.
- El concepto de escalabilidad tanto horizontal como vertical es aplicable a cualquier sistema Cliente/Servidor. La escalabilidad horizontal permite agregar más estaciones de trabajo activas sin afectar significativamente el rendimiento. La escalabilidad vertical permite mejorar las características del servidor o agregar múltiples servidores [7].

## **2.4 Multiplataforma**

Multiplataforma es un término usado para referirse a los programas, sistemas operativos, lenguajes de programación, u otra clase de software, que puedan funcionar en diversas plataformas. Por ejemplo, una aplicación multiplataforma podría ejecutarse en Windows en un procesador x86, en GNU/Linux en un procesador x86, y en Mac OS X en uno x86 (solo para equipos Apple) o en un PowerPC.

---

<sup>7</sup> [http://catarina.udlap.mx/u\\_dl\\_a/tales/documentos/lis/marquez\\_a\\_bm/capitulo5.pdf](http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/marquez_a_bm/capitulo5.pdf)

### 2.4.1 Sistemas Operativos

Un sistema operativo (SO) es el programa o conjunto de programas que efectúan la gestión de los procesos básicos de un sistema informático, y permite la normal ejecución del resto de las operaciones.

Uno de los propósitos del sistema operativo que gestiona el núcleo intermediario consiste en gestionar los recursos de localización y protección de acceso del hardware, hecho que alivia a los programadores de aplicaciones de tener que tratar con estos detalles.

La mayoría de aparatos electrónicos que utilizan microprocesadores para funcionar, llevan incorporado un sistema operativo (teléfonos móviles, reproductores de DVD, computadoras, radios, enrutadores, etc.).

Como ejemplo de sistemas operativos en ordenadores se detallan los más conocidos a continuación:

- **GNU/Linux.-** Este sistema es una versión mejorada de Unix, basado en el estándar POSIX , un sistema que en principio trabajaba en modo comandos. Hoy en día dispone de Ventanas, gracias a un servidor gráfico y a gestores de ventanas como KDE, GNOME entre muchos. Recientemente GNU/Linux dispone de un aplicativo que convierte las ventanas en un entorno 3D como por ejemplo Beryl o Compiz. Lo que permite utilizar Linux de una forma visual atractiva.



Existen muchas distribuciones actuales de Gnu/Linux (Debian, Fedora, Centos, Ubuntu, Slackware, etc) donde todas ellas tienen en común que ocupan el mismo núcleo Linux. Dentro de las cualidades de Gnu/Linux se puede caracterizar el hecho de que la navegación a través de la web es sin riesgos de ser afectada por virus, esto debido al sistema de permisos implementado, el cual no deja correr ninguna aplicación sin los permisos necesarios, permisos que son otorgados por el usuario. A todo esto se suma que los virus que vienen en dispositivos desmontables tampoco afectan al sistema, debido al mismo sistema de permisos.

**Etimología.-** El nombre GNU, **GNU's Not Unix** (GNU no es Unix), viene de las herramientas básicas de sistema operativo creadas por el proyecto GNU, iniciado por Richard Stallman en 1983 y mantenido por la FSF. El nombre Linux viene del núcleo Linux, inicialmente escrito por Linus Torvalds en 1991.

La contribución de GNU es la razón por la que existe controversia a la hora de utilizar Linux o GNU/Linux para referirse al sistema operativo formado por las herramientas de GNU y el núcleo Linux en su conjunto.

**Historia.-** El proyecto GNU, que se inició en 1983 por Richard Stallman; tiene como objetivo el desarrollo de un sistema operativo Unix completo y compuesto enteramente de software libre. La historia del núcleo Linux está fuertemente vinculada a la del proyecto GNU. En 1991 Linus Torvalds empezó a trabajar en un reemplazo no comercial para MINIX que más adelante acabaría siendo Linux.

Cuando Torvalds liberó la primera versión de Linux, el proyecto GNU ya había producido varias de las herramientas fundamentales para el manejo del sistema operativo, incluyendo un intérprete de comandos, una biblioteca C y un compilador, pero como el proyecto contaba con una infraestructura para crear su propio núcleo (o kernel), el llamado Hurd, y este aún no era lo suficiente maduro para usarse, comenzaron a usar a Linux a modo de continuar desarrollando el proyecto GNU, siguiendo la tradicional filosofía de mantener cooperatividad entre desarrolladores. El día en que se estime que Hurd es suficiente maduro y estable, será llamado a reemplazar a Linux.

Entonces, el núcleo creado por Linus Torvalds, quien se encontraba por entonces estudiando en la Universidad de Helsinki, llenó el "espacio" final que había en el sistema operativo de GNU.

**Entorno Gráfico.-** Linux puede funcionar tanto en entorno gráfico como en modo consola. La consola es común en distribuciones para servidores, mientras que la interfaz gráfica está orientada al usuario final tanto de hogar como empresarial. Un escritorio es un conjunto de elementos conformado por ventanas, iconos y similares que facilitan la utilización del computador. Los escritorios más populares en Linux, en orden alfabético son Android (debido a su éxito comercial en teléfonos móviles y tablets), GNOME, KDE, LXDE y Xfce. Intel anunció productos de consumo basados en MeeGo para mediados del 2011, por lo que es probable que este entorno tenga una creciente importancia en los próximos años.

**Como Sistema de Programación.-** La colección de utilidades para la programación de GNU es con diferencia la familia de compiladores más utilizada en este sistema operativo. Tiene capacidad para compilar C, C++, Java, Ada, entre otros muchos lenguajes. Además soporta diversas arquitecturas mediante la compilación cruzada, lo que hace que sea un entorno adecuado para desarrollos heterogéneos.

Hay varios entornos de desarrollo integrados disponibles para GNU/Linux incluyendo, Anjuta, KDevelop, Ultimate++, NetBeans IDE y Eclipse. También existen editores extensibles como Emacs o Vim. GNU/Linux también dispone de capacidades para lenguajes de guion (script).

**Argumentos a favor de GNU/Linux.-** La creciente popularidad de GNU/Linux se debe, entre otras razones, a su estabilidad, al acceso al código fuente (lo que permite personalizar el funcionamiento y auditar la seguridad y privacidad de los datos tratados), a la independencia de proveedor, a la seguridad, a la rapidez con que incorpora los nuevos adelantos tecnológicos (IPv6, microprocesadores de 64 bits), a la escalabilidad (se pueden crear clusters de cientos de computadoras), a la activa comunidad de desarrollo que hay a su alrededor, a su interoperabilidad y a la abundancia de documentación relativa a los procedimientos.

**Administración Pública.-** Hay una serie de administraciones públicas que han mostrado su apoyo al software libre, sea migrando total o parcialmente sus servidores y sistemas de escritorio, sea subvencionándolo. Como ejemplos se tiene a Alemania,

Argentina, Australia, Brasil, España, Chile, China, Cuba, México, Perú, República Dominicana, Ecuador, El Salvador, Uruguay o Venezuela.



**Figura II. 1** Logo de Linux

- **Windows Microsoft.-** A mediados de los años 80 se crea este sistema operativo, pero no es hasta la salida de (Windows 95) que se le puede considerar un sistema operativo, solo era una interfaz gráfica del (MS-DOS) en el cual se disponía de unos diskettes para correr los programas.

Hoy en día es el sistema operativo más difundido en el ámbito doméstico aunque también hay versiones para servidores como Windows NT. (Microsoft) ha diseñado también algunas versiones para superordenadores, pero sin mucho éxito. Años después se hizo el (Windows 98) que era el más eficaz de esa época. Después se crearía el sistema operativo de (Windows ME) (Windows Millenium Edition) aproximadamente entre el año 1999 y el año 2000. Un año después se crearía el sistema operativo de (Windows 2000) en ese mismo año.

Después le seguiría el sistema operativo más utilizado en la actualidad, (Windows XP) y otros sistemas operativos de esta familia especializados en las empresas. Ahora el más reciente es Windows 7 (Windows Seven) que salió al mercado el 22 de octubre del 2009, dejando atrás al (Windows Vista), que tuvo innumerables críticas durante el poco tiempo que duró en el mercado.



**Figura II. 2** Logo de Microsoft Windows

- **Mac OS.-** El lanzamiento oficial del ordenador Macintosh en enero de 1984, al precio de US \$1,995 (después cambiado a \$2,495 dólares). Incluía su sistema operativo Mac OS cuya características novedosas era una GUI (Graphic User Interface), Multitareas y Mouse. Provocó diferentes reacciones entre los usuarios acostumbrados a la línea de comandos y algunos tachando el uso del Mouse como juguete. [<sup>8</sup>]

---

<sup>8</sup> [http://es.wikipedia.org/wiki/Historia\\_de\\_los\\_sistemas\\_operativos](http://es.wikipedia.org/wiki/Historia_de_los_sistemas_operativos)  
<http://es.wikipedia.org/wiki/GNU/Linux>



**Figura II. 3** Logo de Mac OS

### **2.4.2 Lenguajes de programación**

Un lenguaje de programación es un idioma artificial diseñado para expresar computaciones que pueden ser llevadas a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana.

Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Al proceso por el cual se escribe, se prueba, se depura, se compila y se mantiene el código fuente de un programa informático se le llama programación.

También la palabra programación se define como el proceso de creación de un programa de computadora, mediante la aplicación de procedimientos lógicos, a través de los siguientes pasos:

- El desarrollo lógico del programa para resolver un problema en particular.
- Escritura de la lógica del programa empleando un lenguaje de programación específico (codificación del programa).
- Ensamblaje o compilación del programa hasta convertirlo en lenguaje de máquina.
- Prueba y depuración del programa.
- Desarrollo de la documentación.

Existe un error común que trata por sinónimos los términos 'lenguaje de programación' y 'lenguaje informático'. Los lenguajes informáticos engloban a los lenguajes de programación y a otros más, como por ejemplo HTML (lenguaje para el marcado de páginas web que no es propiamente un lenguaje de programación, sino un conjunto de instrucciones que permiten diseñar el contenido de los documentos).

Permite especificar de manera precisa sobre qué datos debe operar una computadora, cómo deben ser almacenados o transmitidos y qué acciones debe tomar bajo una variada gama de circunstancias. Todo esto, a través de un lenguaje que intenta estar relativamente próximo al lenguaje humano o natural. Una característica relevante de los lenguajes de programación es precisamente que más de un programador pueda usar un conjunto común de instrucciones que sean comprendidas entre ellos para realizar la construcción de un programa de forma colaborativa. [<sup>9</sup>]

---

<sup>9</sup> [http://es.wikipedia.org/wiki/Lenguaje\\_de\\_programaci%C3%B3n](http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n)

Existe una gran variedad de clasificación de los lenguajes de programación, aquí se detallan los más importantes lenguajes de programación orientados a objetos:

Java, C#, C++, VB.NET, Ada, Clarion, Delphi, Eiffel, Lexico (en castellano), Objective-C, Ocaml, Oz, PHP, PowerBuilder, Pitón, Ruby, Smalltalk.

A continuación se realiza una introducción al lenguaje de programación que se utilizará en la implementación de ésta investigación y a otros lenguajes que son muy utilizados en la actualidad:

**JAVA.-** Es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

Las aplicaciones Java están típicamente compiladas en un bytecode, aunque la compilación en código máquina nativo también es posible. En el tiempo de ejecución, el bytecode es normalmente interpretado o compilado a código nativo para la ejecución, aunque la ejecución directa por hardware del bytecode por un procesador Java también es posible.

La implementación original y de referencia del compilador, la máquina virtual y las bibliotecas de clases de Java fueron desarrolladas por Sun Microsystems en 1995. Desde entonces, Sun ha controlado las especificaciones, el desarrollo y evolución del lenguaje a



través del Java Community Process, si bien otros han desarrollado también implementaciones alternativas de estas tecnologías de Sun, algunas incluso bajo licencias de software libre.

Entre diciembre de 2006 y mayo de 2007, Sun Microsystems liberó la mayor parte de sus tecnologías Java bajo la licencia GNU GPL, de acuerdo con las especificaciones del Java Community Process, de tal forma que prácticamente todo el Java de Sun es ahora software libre.<sup>[10]</sup>

### **Características del Lenguaje Java**

Se detallan a continuación las principales características de Java.

- **Lenguaje simple.-** Java posee una curva de aprendizaje muy rápida. Resulta relativamente sencillo escribir applets interesantes desde el principio. Todos aquellos familiarizados con C++ encontrarán que Java es más sencillo, ya que se han eliminado ciertas características, como los punteros. Debido a su semejanza con C y C++, y dado que la mayoría de la gente los conoce aunque sea de forma elemental, resulta muy fácil aprender Java. Los programadores experimentados en C++ pueden migrar muy rápidamente a Java y ser productivos en poco tiempo.
  
- **Orientado a objetos.-** Java fue diseñado como un lenguaje orientado a objetos desde el principio. Los objetos agrupan en estructuras encapsuladas tanto sus datos como los métodos (o funciones) que manipulan esos datos. La tendencia del futuro, a la que Java

---

<sup>10</sup> [http://es.wikipedia.org/wiki/Java\\_%28lenguaje\\_de\\_programaci%C3%B3n%29](http://es.wikipedia.org/wiki/Java_%28lenguaje_de_programaci%C3%B3n%29)

se suma, apunta hacia la programación orientada a objetos, especialmente en entornos cada vez más complejos y basados en red.

- **Distribuido.-** Java proporciona una colección de clases para su uso en aplicaciones de red, que permiten abrir sockets, establecer y aceptar conexiones con servidores o clientes remotos, facilitando así la creación de aplicaciones distribuidas.
- **Interpretado y compilado a la vez.-** Java es compilado, en la medida en que su código fuente se transforma en una especie de código máquina, los bytecodes, semejantes a las instrucciones de ensamblador.

Por otra parte, es interpretado, ya que los bytecodes se pueden ejecutar directamente sobre cualquier máquina a la cual se hayan portado el intérprete y el sistema de ejecución en tiempo real (run-time).

- **Robusto.-** Java fue diseñado para crear software altamente fiable. Para ello proporciona numerosas comprobaciones en compilación y en tiempo de ejecución. Sus características de memoria liberan a los programadores de una familia entera de errores (la aritmética de punteros), ya que se ha prescindido por completo los punteros, y la recolección de basura elimina la necesidad de liberación explícita de memoria.
- **Seguro.-** Dada la naturaleza distribuida de Java, donde las applets se bajan desde cualquier punto de la Red, la seguridad se impuso como una necesidad de vital importancia. A nadie le gustaría ejecutar en su ordenador programas con acceso total a

su sistema, procedentes de fuentes desconocidas. Así que se implementaron barreras de seguridad en el lenguaje y en el sistema de ejecución en tiempo real.

➤ **Indiferente a la arquitectura.-** Java está diseñado para soportar aplicaciones que serán ejecutadas en los más variados entornos de red, desde Unix a Windows NT, pasando por Mac y estaciones de trabajo, sobre arquitecturas distintas y con sistemas operativos diversos. Para acomodar requisitos de ejecución tan variado, el compilador de Java genera bytecodes: un formato intermedio indiferente a la arquitectura diseñada para transportar el código eficientemente a múltiples plataformas hardware y software. El resto de problemas los soluciona el intérprete de Java.

➤ **Portable.-** La indiferencia a la arquitectura representa sólo una parte de su portabilidad. Además, Java especifica los tamaños de sus tipos de datos básicos y el comportamiento de sus operadores aritméticos, de manera que los programas son iguales en todas las plataformas.

Estas dos últimas características se conocen como la *Máquina Virtual Java* (JVM).

➤ **Alto rendimiento Multihebra.-** Hoy en día ya se ven como terriblemente limitadas las aplicaciones que sólo pueden ejecutar una acción a la vez. Java soporta sincronización de múltiples hilos de ejecución (*multithreading*) a nivel de lenguaje, especialmente útiles en la creación de aplicaciones de red distribuidas. Así, mientras un hilo se encarga de la comunicación, otro puede interactuar con el usuario mientras otro presenta una animación en pantalla y otro realiza cálculos.

- **Dinámico.-** El lenguaje Java y su sistema de ejecución en tiempo real son dinámicos en la fase de enlazado. Las clases sólo se enlazan a medida que son necesitadas. Se pueden enlazar nuevos módulos de código bajo demanda, procedente de fuentes muy variadas, incluso desde la Red.
  
- **Produce applets.-** Java puede ser usado para crear dos tipos de programas: aplicaciones independientes y applets. Las aplicaciones independientes se comportan como cualquier otro programa escrito en cualquier lenguaje, como por ejemplo el navegador de Web HotJava, escrito íntegramente en Java.

Por su parte, las applets son pequeños programas que aparecen embebidos en las páginas Web, como aparecen los gráficos o el texto, pero con la capacidad de ejecutar acciones muy complejas, como animar imágenes, establecer conexiones de red, presentar menús y cuadros de diálogo para luego emprender acciones, etc. [<sup>11</sup>]

**C Sharp (C#).**-Su pronunciación es *si Sharp* en inglés, es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA (ECMA-334) e ISO (ISO/IEC 23270). C# es uno de los lenguajes de programación diseñados para la infraestructura de lenguaje común.

---

<sup>11</sup> <http://www.iec.csic.es/criptonomicon/java/quesjava.html>

Su sintaxis básica deriva de C/C++ y utiliza el modelo de objetos de la plataforma .NET, similar al de Java, aunque incluye mejoras derivadas de otros lenguajes.

El nombre C Sharp fue inspirado por la notación musical, donde # (sostenido, en inglés *sharp*) indica que la nota (*C* es la nota do en inglés) es un semitono más alta, sugiriendo que C# es superior a C/C++. Además, el signo '#' viene de cuatro '+' pegados.

Aunque C# forma parte de la plataforma .NET, ésta es una API, mientras que C# es un lenguaje de programación independiente diseñado para generar programas sobre dicha plataforma. Ya existe un compilador implementado que provee el marco Mono - DotGNU, el cual genera programas para distintas plataformas como Windows, Unix y GNU/Linux.

**C++.-** Es un lenguaje de programación diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue el extender al exitoso lenguaje de programación C con mecanismos que permitan la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido.

Posteriormente se añadieron facilidades de programación genérica, que se sumó a los otros dos paradigmas que ya estaban admitidos (programación estructurada y la programación orientada a objetos). Por esto se suele decir que el C++ es un lenguaje de programación multiparadigma.

Actualmente existe un estándar, denominado ISO C++, al que se han adherido la mayoría de los fabricantes de compiladores más modernos. Existen también algunos intérpretes, tales como ROOT.

Una particularidad del C++ es la posibilidad de redefinir los operadores (sobrecarga de operadores), y de poder crear nuevos tipos que se comporten como tipos fundamentales.

El nombre C++ fue propuesto por Rick Mascitti en el año 1983, cuando el lenguaje fue utilizado por primera vez fuera de un laboratorio científico. Antes se había usado el nombre "C con clases". En C++, la expresión "C++" significa "incremento de C" y se refiere a que C++ es una extensión de C.

**Visual Basic .NET (VB.NET).**- Es un lenguaje de programación orientado a objetos que se puede considerar una evolución de Visual Basic implementada sobre el framework .NET. Su introducción resultó muy controvertida, ya que debido a cambios significativos en el lenguaje VB.NET no es compatible hacia atrás con Visual Basic, pero el manejo de las instrucciones es similar a versiones anteriores de Visual Basic, facilitando así el desarrollo de aplicaciones más avanzadas con herramientas modernas.

La gran mayoría de programadores de VB.NET utilizan el entorno de desarrollo integrado Microsoft Visual Studio en alguna de sus versiones (Visual Studio .NET, Visual Studio .NET 2003 o Visual Studio 2005), aunque existen otras alternativas, como SharpDevelop (que además es libre).

Al igual que con todos los lenguajes de programación basados en .NET, los programas escritos en VB .NET requieren el Framework .NET o Mono para ejecutarse. [<sup>12</sup>]

## **2.5 Creación de scripts en Linux**

Definición de Script: Un script es un archivo que incluye un conjunto de comandos. Son ejecutados desde la primera línea hasta la última (de forma secuencial).

### **2.5.1 ¿Qué es el SHELL SCRIPT?**

Un Shell Script es un script para la shell de comandos (terminal). Para crear un script basta con un editar un fichero nuevo y en el nombre poner .sh Ejemplo: HolaMundo.sh

Una vez creado, se empieza a editarlo. Se puede utilizar un editor de textos gráfico como gedit o un editor en terminal como vim, nano o emacs. En la primera línea del script se debe indicar que shell que va a usar ( /bin/bash/ , /usr/bin/perl , etc ) Aunque da igual la que se use lo importante es el contenido:

```
#!/bin/bash
```

```
#! Se conoce con el nombre de Sha Bang.
```

Se denomina “sha-bang” a la secuencia `#!` con la que se inician los scripts. Su función es indicarle al sistema que se trata de un conjunto de comandos para que sean interpretados. En realidad, es un número mágico de dos bytes. El número mágico es un marcador especial

---

<sup>12</sup> [http://es.wikipedia.org/wiki/Visual\\_Basic\\_.NET](http://es.wikipedia.org/wiki/Visual_Basic_.NET)

para indicar el tipo de archivo, en este caso, indica que se trata de un script de shell ejecutable.

Para introducir comentarios se debe poner #. Por cada línea que se desee poner un comentario, lo primero que debe tener es #. Es importante añadir comentarios comentando la utilidad del script o las variables que se crean.

### **2.5.2 ¿Cómo ejecutar un SCRIPT?**

Antes de poder ejecutarlo, se debe dar permisos de ejecución (+x) por ello, se hace uso del comando chmod y se da permisos de ejecución, si se desea, se pueden dar a todos los usuarios y grupos.

chmod 755 /ruta\_del\_script Para el usuario propietario

chmod 777 /ruta\_del\_script Para cualquier usuario

Una vez hecho todo lo anterior, se usa:

```
./nombredelscript.sh
```

Pero también se puede usar si es un shell script:

```
sh nombredelscript.sh
```

### **2.5.3 Código de un SCRIPT**

Ya se tiene el script creado, se le ha puesto la cabecera y se ha cambiado los permisos, ya solo falta meter el código. Se va a empezar construyendo lo esencial para ir desarrollando estructuras más complejas: Lo primero es saber cómo dar valor a una variable. Es tan sencillo como poner:



nombre\_variable=valor

Si se quiere guardar la salida de un programa solo se tiene que poner entre tildes invertidas:

nombre\_variable=`comando` También hay un comando que lee por teclado las variables (read). Para ponerlo es: read [opciones] nombre\_variable1 nombre\_variable2 nombre\_variableN ejemplo: read -p "Introduce el nombre y los apellidos" nombre

apellidos. Tiene muchas opciones pero estas son las más importantes y usadas:

-n num\_car: Número máximo de caracteres que se puede introducir por teclado

-p "frase": Muestra por pantalla una frase para indicar que se debe introducir

-d "delimitador": Se especifica cuál va a ser el delimitador, es decir se dice que el delimitador será ";" pues todo lo que venga antes de un ";" lo cogerá una variable y todo lo que venga después de ese delimitador hasta el próximo ";" será receptado por otra variable.

Cuando se desea utilizar el valor de una variable en el código, se refiere a éste como:

\$nombre\_variable. Estos son algunos de los ejemplos de esta primera parte. Para ejecutarlos sólo hay que crear un archivo .sh. A lo largo de los ejemplos se introducen algunos comandos básicos de Linux.

### **HolaMundo.sh**

```
#!/bin/bash
```

```
clear
```

```
echo "Hola mundo ,este es mi primer script"
```

### **ScriptUno.sh**

```
#!/bin/bash

clear

nombre="Perico"

apellidos="Palotes"

echo "Te llamas $nombre $apellidos"
```

### **Fecha.sh**

```
#!/bin/bash

clear

fecha=`date | cut -d " " -f 1,2,3`

hora=`date | cut -d " " -f 4`

echo "Hoy es $fecha y son las $hora"
```

## **2.5.4 Depuración**

Depuración de programas es el proceso de identificar y corregir errores de programación. En inglés se le conoce como debugging, ya que se asemeja a la eliminación de bichos (bugs), manera en que se conoce informalmente a los errores de programación. Se dice que el término bug proviene de la época de las computadoras de bulbos, en las cuales los problemas se generaban por los insectos que eran atraídos por las luces y estropeaban el equipo.

Depurar el código sirve para ver como se ejecuta paso por paso el script, que valores toman sus variables, si se ha cometido un fallo saber en qué parte del código ha sido, etc. No es

algo que se deba hacer obligatoriamente por cada script realizado (sería demasiado pesado) pero facilita mucho a la hora de buscar el problema que surja. Hay dos formas:

**Primera:** bien en la línea `#!/bin/bash -x | -v`

`-x` → Muestra las instrucciones antes de ejecutarlas por consola y sustituyendo las variables

`-v` → Presenta todo pero sin sustituir las variables

**Segunda:** Mediante set

Cambiando las opciones de ejecución de la shell a lo largo del script, también con las opciones `-x | -v`

### 2.5.5 Estructuras condicionales

La estructura básica de una condición sería:

if condición

then

comando1

...

else

comando1

...

fi

Como se ve la condición si (if) se cierra con su correspondiente fi que al parecer un juego de palabras es fácil de recordar. Entonces si la condición se cumple entraría por el then, en caso de que no, por el else. Pero este es un método simple, luego se tiene uno más complejo con if anidados, sería:

```
if condición
then
comando1
elif condición
then
comando
elif condición
then
...
fi
```

Ahora lo que hace es evaluar la condición, si es verdadera entra por el then, pero si no y se da el caso de otra condición entraría por el elif, los elif no se cierran, solamente el fi final corresponde a la apertura del if.

La condición es cualquier cosa que de un return (que devuelva) algo que sea 0 o verdadero.

Nótese el uso del archivo /dev/null como archivo vacío para comprobar algunas condicionales.

Se observa cómo se puede hacer un script para comprobar que realmente la ruta es un directorio.

```
CompruebaDirectorio.sh
```

```
#!/bin/bash
```

```
clear
```

```
if `cd /tmp/prueba/ >/dev/null`
```

```
then
```

```
echo "Pues sí, es un directorio y contiene..."
```

```
ls -l
```

```
else
```

```
echo "Pues va a ser que no es un directorio"
```

```
fi
```

Este script es un claro ejemplo de cómo comprobar si un usuario y un grupo existen en el sistema, se ve también el uso que se le da al elif.

### **ExisteGrupoUsuario.sh**

```
#!/bin/bash
```

```
clear
```

```
read -p "Introduce usuario... " user
```

```
read -p "Introduce grupo... " group
```

```
if `grep -e "^$user:.*" /etc/passwd >/dev/null`
```

```
then
```

```
if `grep -e "^$group:.*" /etc/group >/dev/null`
```

```
then
```

```
echo "Usuario y grupo ya existen en el sistema"
```

```
elif `grep -e "^$group:.*" /etc/group >/dev/null`
```

```
then
```

```
echo "usuario no existe, grupo si!!"
```

```
else
```

```
echo "Ni grupo ni usuario existen"
```

```
fi
```

### **Condicionales case**

Se estructuran de la siguiente forma:

```
case expresion in
caso1) comandos ;;
caso2) comandos ;;
*) comandos ;;
esac
```

Para aquellos que sepan de programación, funciona igual que los switch, esta estructura sirve para dependiendo de la expresión se cumple un determinado caso, es decir, según sea el valor de esa expresión se hará un caso u otro. Esto se ve mejor con un ejemplo:

### **tecla.sh**

```
#!/bin/bash
clear
read -n 1 -p "Pulsa una tecla " tecla
case $tecla in
[a-zA-Z]) echo "Ha introducido una letra" ;;
[0-9]) echo "Ha introducido un numero" ;;
*) echo "Ha introducido un caracter especial" ;;
esac
```

Este script sirve para comprobar que tecla se ha pulsado, primero pide por teclado la tecla que será guardada en la variable tecla, y se usa en el case. Si la tecla es una letra se hará todos los comandos que hayan en el caso [ a-z,A-Z ]( ya que una letra puede ser cualquiera, hace falta abarcar a todas las posibilidades por eso se pone el intervalo de a-z y A-Z ), si la tecla es un número se hará todos los comandos que haya en el caso [0-9 ]( ya que puede ser cualquier número, hace falta abarcar a todas las posibilidades por eso se pone el intervalo de 0-9 ) y si la tecla no es un número ni una letra, se ejecutan todos los comandos del caso \*.

Se pone el \* cuando no se va a cumplir el resto de casos. Es muy importante saber que las dobles comas ( ;; )se pone obligatoriamente al final de cada caso ya que marcan el final de ese caso, sino el programa no funcionará. Cuando se haga una condicional se debe poner al final el esac ya que es el cierre del case. Se suele utilizar mucho esta estructura para mostrar un menú al usuario. Ejemplo:

### **ejemploMenu.sh**

```
#!/bin/bash

clear

echo "1.Ejemplo de menu uno"

echo "2.Ejemplo de menu dos"

read -n 1 -p "Introduce una opcion" opcion

case $opcion in

1) exit 1 ;;
```

2) exit 2 ;;

\*) echo "No se ha introducido ni un dos ni un uno" ;;

esac

### **2.5.6 Comando test**

Este comando sirve para expresar condiciones y evaluarlas, si son correctas origina códigos de salida = 0 y si son falsas = 1. El comando pretende abreviar un poco en algunos casos, por eso se suele utilizar su forma corta:

test expresiones —> [ expresión ]

Hay que tener en cuenta que la forma es:

[(espacio)expresión(espacio)]

Ya que si no se ponen los espacios en blanco daría lugar a error. Un ejemplo de su uso:

test -f /home/alumno && echo "Existe directorio". La salida será:

Existe Directorio

En la forma resumida se puede escribir:

[ -f /home/albertjh/googleearth ] && echo "fichero existe..."

fichero existe...

### **Expresiones test**

Estas son algunas de las más comunes:

Comprobación directorios:

-f /ruta/nombre -> Comprueba si es un fichero normal

-l /ruta/nombre -> Comprueba si es un enlace suave



- d /ruta/ -> Comprueba que existe el directorio
- x /ruta/nombre -> Comprueba si es un ejecutable
- u /ruta/nombre -> Comprueba si tiene activados los permisos suid
- g /ruta/nombre -> Comprueba si tiene activados los permisos sgid
- s /ruta/nombre -> Comprueba que su tamaño es mayor a 0

#### Comprobación de cadenas:

- “cadena” = “cadena2” -> Comprueba si son iguales
- z cadena -> Comprueba si está vacía
- “cadena” != “cadena2” -> Comprueba que son diferentes

#### Comprobación de expresiones numéricas:

- exp -eq exp2 -> Comprueba si son iguales
- exp -ge exp2 -> Comprueba si  $exp \geq exp2$
- exp -ne exp2 -> Comprueba si exp distinto de exp2
- exp -gt exp2 -> Comprueba si  $exp > exp2$
- exp -le exp2 -> Comprueba si  $exp \leq exp2$
- exp -lt exp2 -> Comprueba si  $exp < exp2$

#### Para concatenar expresiones a evaluar:

- o = OR
- a = AND
- != NOT

Algunos ejemplos del uso de test:

**numeros.sh**

```
#!/bin/bash
```

```
clear
```

```
read -d “,” -p “Introduce dos números separados por comas ” num1 num2
```

```
if [ -z $num1 -o -z $num2 ]
```

```
then
```

```
echo “Debes introducir dos números, por favor”
```

```
elif [ $num1 -eq $num2 ]
```

```
then
```

```
echo “Los números son iguales”
```

```
elif [ $num1 -gt $num2 ]
```

```
then
```

```
echo “El $num1 > que $num2”
```

```
fi
```

Nota: Algunos de los ejemplos de este documentos han sido extraídos de “Diario de un Linuxero” [13].

---

<sup>13</sup> <http://www.ucm.es/info/aulasun/archivos/SCRIPTS.pdf>

## **2.6 Demonios o Servicios**

Un demonio, daemon o dæmon (de sus siglas en inglés Disk And Execution MONitor), es un tipo especial de proceso informático no interactivo, es decir, que se ejecuta en segundo plano en vez de ser controlado directamente por el usuario.

Este tipo de programas se ejecutan de forma continua (infinita), vale decir, que aunque se intente cerrar o matar el proceso, este continuará en ejecución o se reiniciará automáticamente dependiendo del proceso. Todo esto sin intervención de terceros y sin dependencia de consola alguna.

### **2.6.1 Historia de los demonios**

El origen de la palabra daemon (demonio), se encuentra en la antigua Grecia y la figura del daimon, un espíritu interior, equivalente a un "ángel protector" que guiaba y protegía a los hombres.

Los programas demonios reciben este nombre en los sistemas UNIX. En otros sistemas existen procesos similares como los TSRs de MS-DOS o los servicios de Windows.

Según una investigación realizada por Richard Steinberg, la palabra fue utilizada en 1963 por primera vez, en el área de la informática, para denominar a un proceso que realizaba backups en unas cintas, este proceso se utilizó en el proyecto MAC del MIT y en una computadora IBM 7094, dicho proyecto estaba liderado por Fernando J. Corbato, quien afirma que se basó en el daemon de James Maxwell, este daemon era una especie de

vigilante que residía en medio de un recipiente dividido en dos, lleno de moléculas, el vigilante o daemon se encargaba de permitir, dependiendo de la velocidad de la molécula, que éstas pasaran de un lado al otro, los daemons de las computadoras actúan muy similar al daemon de Maxwell, pues realizan acciones según el comportamiento y algunas condiciones del sistema.

### **2.6.2 Características de los demonios**

Los demonios suelen tener las siguientes características:

- No disponen de una "interfaz" directa con el usuario, ya sea gráfica o textual.
  
- No hacen uso de la entradas y salidas estándar para comunicar errores o registrar su funcionamiento, sino que usan archivos del sistema en zonas especiales (/var/log/ en los UNIX más modernos) o utilizan otros demonios especializados en dicho registro como el syslogd.

Por ejemplo, una máquina que alberga un servidor web utilizará un demonio httpd (HTTP Daemon) para ofrecer el servicio y que los visitantes a dicha web puedan acceder. Otro ejemplo son los demonios "cronológicos" como cron, que realizan tareas programadas como mantenimiento del sistema en segundo plano [<sup>14</sup>].

---

<sup>14</sup> [http://es.wikipedia.org/wiki/Demonio\\_%28inform%C3%A1tica%29](http://es.wikipedia.org/wiki/Demonio_%28inform%C3%A1tica%29)

Los daemons (o demonios) no son más que un proceso que se ejecuta en segundo plano. Estos demonios ejecutan diferentes funciones y proporcionan ciertos servicios, pero sin la interacción del usuario; son procesos de los que no "se nota" su ejecución.

Los demonios pueden ser iniciados al arrancar el sistema, al entrar en un runlevel (nivel de ejecución) determinado, o simplemente cuando el usuario los inicie. Se verá entonces de qué modo se pueden controlar los demonios (iniciarlos, pararlos, etc) y cómo se puede hacer que se inicien automáticamente.

### **2.6.3 Iniciando demonios manualmente, directorio init.d**

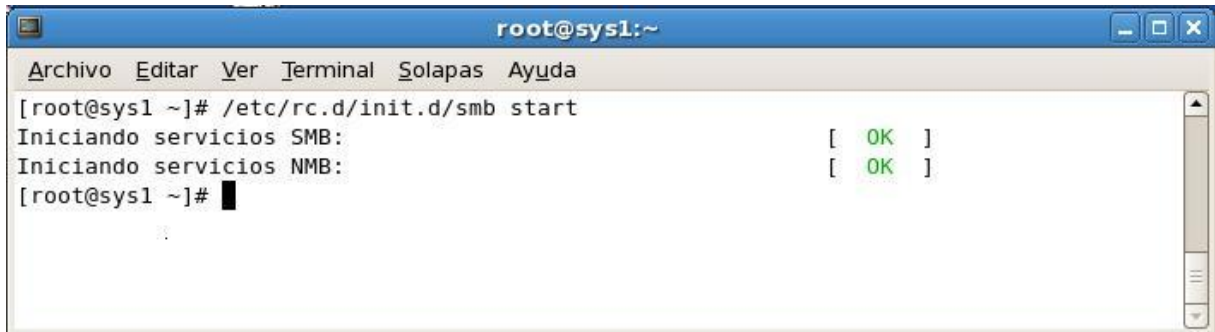
Dentro de esta carpeta ubicada en /etc o en /etc/rc.d dependiendo de la distribución, se encuentran una serie de scripts que permiten iniciar/detener la gran mayoría de los servicios/servidores que estén instalados en el equipo. Estos scripts están programados de tal manera que la mayoría reconoce los siguientes argumentos:

- start
- stop
- restart
- status

Los argumentos son autodescriptivos, y tienen permisos de ejecución, entonces siendo root es posible iniciar un servicio de la siguiente manera, por ejemplo samba:

```
#> /etc/rc.d/init.d/smb start
```

Iniciando servicios SMB            [OK]



```
root@sys1:~  
Archivo Editar Ver Terminal Solapas Ayuda  
[root@sys1 ~]# /etc/rc.d/init.d/smb start  
Iniciando servicios SMB: [ OK ]  
Iniciando servicios NMB: [ OK ]  
[root@sys1 ~]#
```

**Figura II. 4** Inicia el servicio Samba desde una ruta determinada

Solo que hay que cambiar start por stop | restart | status para detenerlo, reiniciarlo (releer archivos de configuración) o checar su estatus. Ahora bien si se está parado dentro del directorio se puede hacer así.

```
#> pwd
```

```
/etc/rc.d/init.d
```

```
#> ./smb stop
```

```
Apagando los servicios SMB [OK]
```



```
root@sys1:/etc/rc.d/init.d  
Archivo Editar Ver Terminal Solapas Ayuda  
[root@sys1 init.d]# pwd  
/etc/rc.d/init.d  
[root@sys1 init.d]# ./smb stop  
Apagando los servicios SMB: [ OK ]  
Apagando los servicios NMB: [ OK ]  
[root@sys1 init.d]#
```

**Figura II. 5** Apaga el servicio Samba desde una ruta determinada

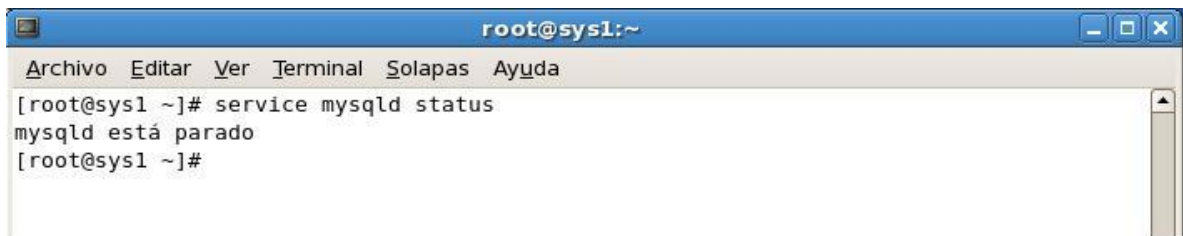
Se trata de tan solo un script así que con el permiso de ejecución (x) se puede ejecutar con ./ seguido del nombre del servicio, sin espacios y después el argumento que se necesite, iniciarlo, detenerlo etc.

#### 2.6.4 Comando service

En varias distribuciones, como Fedora o RedHat, existe el comando service, este comando permite también iniciar y/o detener servicios, de hecho funciona exactamente igual a como si se escribiera la ruta completa hacía el directorio init.d, con service se indica de la siguiente manera:

```
#> service mysql status
```

```
Checking for service MySQL: stopped
```




**Figura II. 6** Detiene el servicio Samba con el comando service

Si se desea iniciarlo:

```
#> service mysql start
```

Starting service MySQL [OK]



```
root@sys1:~  
Archivo Editar Ver Terminal Solapas Ayuda  
[root@sys1 ~]# service mysqld start  
Iniciando MySQL: [ OK ]  
[root@sys1 ~]#
```

**Figura II. 7** Inicia un servicio con el comando service

### 2.6.5 Iniciando servicios desde el arranque del sistema

En muchos casos es conveniente que un servidor o servicio inicien junto con el arranque del equipo en si, por ejemplo el servidor web Apache o alguna base de datos, esto es para que estén disponibles todo el tiempo y no se requiera de intervención del administrador para iniciarlos.

En Linux, a diferencia de otros sistemas operativos, es posible configurarlo en base a niveles de ejecución (run levels), cada nivel de ejecución (en la mayoría de las distribuciones son 7), inicia o detiene (Start o Kill) ciertos servicios. Estos niveles son los siguientes:

- 0 Detener o apagar el sistema
- 1 Modo monousuario, generalmente utilizado para mantenimiento del sistema
- 2 Modo multiusuario, pero sin soporte de red
- 3 Modo multiusuario completo, con servicios de red



- 4 No se usa, puede usarse para un inicio personalizado
- 5 Modo multiusuario completo con inicio gráfico ( X Window)
- 6 Modo de reinicio (reset)

Por ejemplo el nivel 0, que apaga el equipo, mata o detiene a todos los procesos del sistema, todos los servicios, lo mismo hace el 6 con la diferencia que después inicia un script que permite reiniciar el sistema. El nivel por omisión o por default del sistema está definido en `/etc/inittab`, en la línea `initdefault`:

```
$> grep initdefault /etc/inittab
```

```
id:5:initdefault:
```

```
# runlevel 0 is System halt (Do not use this for initdefault!)
```

```
# runlevel 6 is System reboot (Do not use this for initdefault!)
```

La salida es de una distribución OpenSuse 10. Como se puede apreciar existe una línea que comienza con `id`, seguido por `5` que indicaría entonces el nivel por default del sistema, basta con cambiar este valor con cualquier editor, reiniciar el equipo y se entraría a otro nivel, por ejemplo el `3` que tiene una funcionalidad completa pero sin sistema de ventanas X Window (ideal para equipos obsoletos o con pocos recursos de hardware). Nótese que en el listado previo aparecen dos líneas de comentarios, las que empiezan con `#`, que hacen la advertencia de no usar nunca los niveles `0` y `6` de `initdefault`, ya que jamás arrancaría el equipo. Se tendría que usar un método de rescate. (He aquí la importancia de jamás dejar

una terminal abandonada con una sesión de root abierta ya que podría llegar un bromista y rápidamente cambiar la línea init default a 6 por ejemplo, y después estaría muy divertido viendo a un pobre administrador novato tratándose de explicar por qué diablos el equipo se resetea hasta el infinito sin nunca entrar a una pantalla de login).

El nivel de ejecución actual del sistema puede ser consultado con `who -r` y con el comando `runlevel`, este último muestra dos números el primero es el nivel previo en que se estaba y el segundo el nivel actual, si no se ha cambiado de nivel aparece una N.

### **2.6.6 Comando chkconfig**

Lo anterior puede resultar engorroso para más de alguno pero es la única manera si no se tienen herramientas gráficas o de línea de comandos para configurar los servicios más ágilmente, y precisamente una de estas herramientas de línea de comandos es `chkconfig` que permite configurar que servicios arrancan/detienen en cada nivel de ejecución. Aclaración: `chkconfig` no inicia ni detiene servicios al momento (excepto aquellos bajo `xinetd`), tan solo crea o elimina precisamente los enlaces de los que se habló en el punto anterior de una manera más amigable. Si lo que se quiere es iniciar o detener el servicio en tiempo real o manualmente hay que usar `service` o directamente el script con su argumento conveniente tal como se explicó anteriormente. Ejemplos que muestran como trabajar con `chkconfig`.

Con la opción --list da una lista completa de todos los servicios instalados y para cada nivel si arrancará (on) al entrar a ese nivel o se detendrá (off) o simplemente no se iniciara.

Nótese que al final de la lista vienen los servicios que dependen del súper servidor xinetd.

```
#> chkconfig --list
apache2                0:off 1:off 2:off 3:off 4:off 5:off 6:off
bluetooth              0:off 1:off 2:off 3:off 4:off 5:off 6:off
cron                   0:off 1:off 2:on  3:on  4:off 5:on  6:off
cups                   0:off 1:off 2:on  3:on  4:off 5:on  6:off
dhcpcd                 0:off 1:off 2:off 3:off 4:off 5:off 6:off
ldap                   0:off 1:off 2:off 3:off 4:off 5:off 6:off
mysql                  0:off 1:off 2:off 3:off 4:off 5:off 6:off
named                  0:off 1:off 2:off 3:off 4:off 5:off 6:off
network                0:off 1:off 2:on  3:on  4:off 5:on  6:off
nfs                    0:off 1:off 2:off 3:on  4:off 5:on  6:off
quotad                 0:off 1:off 2:off 3:off 4:off 5:off 6:off
smb                    0:off 1:off 2:off 3:on  4:off 5:on  6:off
vmware                 0:off 1:off 2:on  3:on  4:off 5:on  6:off
xinetd                 0:off 1:off 2:off 3:on  4:off 5:on  6:off
zebra                  0:off 1:off 2:off 3:off 4:off 5:off 6:off
xinetd based services:
  rsync:                off
  sane-port:            off
  servers:              off
  services:             off
  swat:                 on
  systat:               off
  tftp:                 off
  time:                 off
  time-udp:             off
  vnc:                  off
  vsftpd:               off
```

Si se indica como argumento el nombre de algún servicio retornará su estatus:

```
#> chkconfig smb
smb on
```

Sin ningún argumento dará una lista con el status de todos los servicios:

```
#> chkconfig
httpd off
mysql on
smb on
```

Se desea que el servidor web apache (httpd) inicie cuando se entra en el nivel 5, entonces se usa la opción --level:

```
#> chkconfig --level 5 httpd on
```

La base de datos MySQL no se desea que inicie en los niveles 3 y 5, solo hasta que el administrador decida arrancarla:

```
#> chkconfig --level 35 mysql off [15]
```

## 2.7 Hebras

Una hebra es un punto de ejecución de un proceso. Un proceso tendrá una o más hebras. Las hebras representan un método software para mejorar el rendimiento de los sistemas operativos reduciendo el recargo por el cambio de contexto entre procesos. Las hebras de un mismo proceso compartirán recursos, como memoria, archivos. Las hebras asumirán el papel de los procesos como unidad de planificación. [<sup>16</sup>]

Las hebras tendrán un flujo separado de control (o punto de ejecución) y tienen su propia pila y estado hardware. Como todos los recursos (excepto la CPU) son gestionados por el proceso, la comunicación entre sus hebras será mucho más rápida y eficiente, porque todas las hebras de un proceso comparten un mismo espacio de memoria.

---

<sup>15</sup> [http://www.linuxtotal.com.mx/index.php?cont=info\\_admon\\_003](http://www.linuxtotal.com.mx/index.php?cont=info_admon_003)

<sup>16</sup> <http://gsm.gui.uva.es/revista/login/15/process.html>.

Cuando se realice un cambio de contexto entre hebras de procesos diferentes se realizará un cambio de contexto entero.

Con las hebras se hacen a los sistemas operativos mucho más rápidos, pero siempre habrá que tener cuidado con la planificación entre hebras, ya que varias hebras podrán acceder a cualquier variable compartida y puede haber problemas de inconsistencia de datos.

### **2.7.1 Hebras en java**

Java es el único lenguaje (de los “populares”) que incluye de forma natural la concurrencia, de manera que el programador puede definir hebras de ejecución.

Cada hebra es una porción del programa que puede ejecutarse concurrentemente con otras hebras. Esta capacidad se denomina multihebra o multihilo.

Ejemplos de hebras:

- En un gestor de descargas FTP, los archivos se descargan en varias partes simultáneamente. La transferencia de cada parte del archivo la gestiona una hebra distinta.
- El recolector de basura de Java es una hebra del intérprete que libera automáticamente la memoria que ya no se utiliza.

Hay dos posibilidades de cómo se puede implementar hebras en java:

**Crear una clase que extienda la clase Thread.-** Se implementa el objeto que realizará el cómputo, y se sobrecarga el método run() de la clase Thread: Ejemplo:

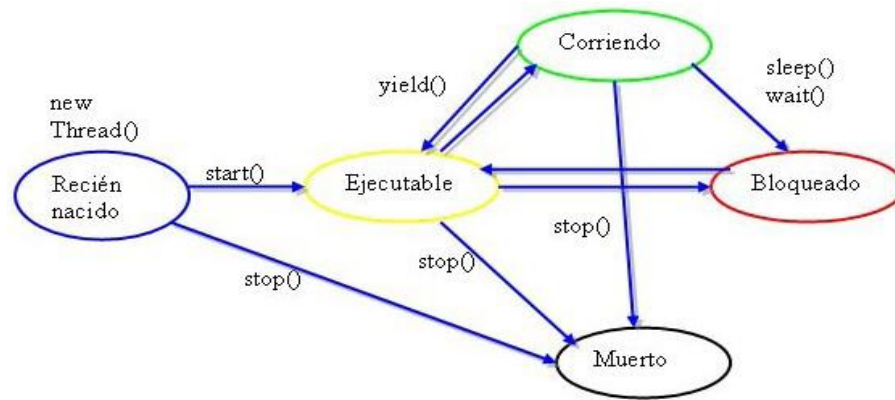
```
class HebraDePrueba extends Thread  
  
    {  
  
        Código  
  
    }
```

**Implementar la interfaz Runnable.-** Una interfaz en Java no es más que una declaración de los métodos que ha de tener una clase que implemente esa interfaz (algo similar a una clase abstracta). Cuando se quiere paralelizar la ejecución de una clase que ya deriva de otra distinta de Thread (por ejemplo, TelefonoMovil), se debe implementar el interfaz Runnable en esa clase. Implementar el interfaz Runnable permite ejecutar en una hebra el código contenido en el método run. Ejemplo:

```
class ObjetoRunnable implements Runnable  
  
    {  
  
        Código  
  
    }
```

### **2.7.2 Estado de una hebra**

Una hebra puede tener diversos estados, los cuales se muestran en el siguiente gráfico y a continuación se detalla la secuencia de su ciclo de vida:



**Figura II. 8** Estados de un thread

### **Recién nacido**

Como se muestra en el gráfico para el estado recién nacido se ha reservado memoria para el thread y se han inicializado sus variables, pero no se puesto todavía en cola de ejecución. Dicho de otro modo: se ha creado un objeto de tipo Thread. En este estado le pueden ocurrir dos cosas: que pase a cola de ejecución, mediante el método Star(), o que se mate sin llegar a ejecutarse nunca, mediante el método stop().

### **Ejecutable**

El thread está listo para ser ejecutado, se halla en cola de ejecución junto a los demás threads. Periódicamente la CPU cambia de thread de ejecución, cogiendo el thread que estaba ejecutando y poniéndolo en la cola de ejecución y escogiendo un nuevo thread de la cola de ejecución. Los threads llevan asociada una prioridad, cuando la CPU ha de elegir un thread de la cola de ejecución para ver cuál ejecuta, elige el de mayor prioridad y de haber varios los va ejecutando secuencialmente.

## **Corriendo**

El thread está siendo ejecutado en el procesador. Hay tres posibles causas por las que un thread que está corriendo pueda liberar el control del procesador:

- Se acaba su tiempo de ejecución y vuelve a la cola de procesos, pasando al estado ejecutable.
- Se ejecuta el método `sleep()` o `wait()`, pasando a estado bloqueado.
- Termina su cuerpo de ejecución, o se invoca a su método `stop`, por lo que pasa al estado muerto.

## **Bloqueado**

En este estado el thread no está en la cola de ejecución, está esperando a que ocurra un determinado evento o transcurra un determinado plazo de tiempo antes de poder acceder a la cola de ejecución.

## **Muerto**

Como su nombre indica el thread deja de existir. Puede ocurrir bien porque ha terminado su cuerpo de ejecución o bien porque ha sido matado mediante el método `stop ()`. [<sup>17</sup>]

## **2.8 Concurrencia**

Procesamiento Concurrente es la situación que se obtiene al hacer una instantánea del sistema, es decir; varios procesos se encuentran en un estado intermedio entre su estado inicial y final.

---

<sup>17</sup> (<http://javahispano.org>). Página 123 de 125



Programación Concurrente es el conjunto de notaciones que se usan para expresar paralelismo y técnicas que se usan para resolver posibles conflictos entre los procesos. [18]

Dos procesos serán concurrentes cuando la primera instrucción de uno de ellos se ejecuta después de la primera del otro y antes de la última.

## **2.9 Servicios no Orientados a la Conexión**

El servicio no orientado a la conexión se concibió con base en el sistema postal. Cada mensaje (carta) lleva la dirección completa de destino y cada una se encamina a través del sistema, independiente de los demás. En general, cuando se envían dos mensajes al mismo destino, el primero que se envíe será el primero en llegar. Sin embargo, es posible que el que se envió primero se dilate tanto que le segundo llegue primero.

Cada paquete debe llevar la dirección destino, y con cada uno, los nodos de la red deciden el camino que se debe seguir. Existen muchas técnicas para realizar esta decisión, como por ejemplo comparar el retardo que sufriría en ese momento el paquete que se pretende transmitir según el enlace que se escoja.

Características:

- Cada mensaje puede ser encaminado independientemente.
- No se garantiza que los mensajes lleguen en el mismo orden en que son enviados.
- Requiere menos ancho de banda, debido a que no utiliza técnicas para detectar o corregir errores. Esto no necesariamente implica que la comunicación es poco

---

<sup>18</sup> <http://trevinca.ei.uvigo.es/~nrufino/so/Teoria/Tema2.pdf>

confiable. La detección y corrección de errores puede efectuarse en otras capas en referencia al modelo OSI.

- Ejemplo: Correo electrónico, discos virtuales.

En resumen se muestra el siguiente cuadro y el significado de las siglas CLNS es Servicios no Orientados a Conexión. [<sup>19</sup>]

**Tabla II. I** Características Servicios no Orientados a la Conexión

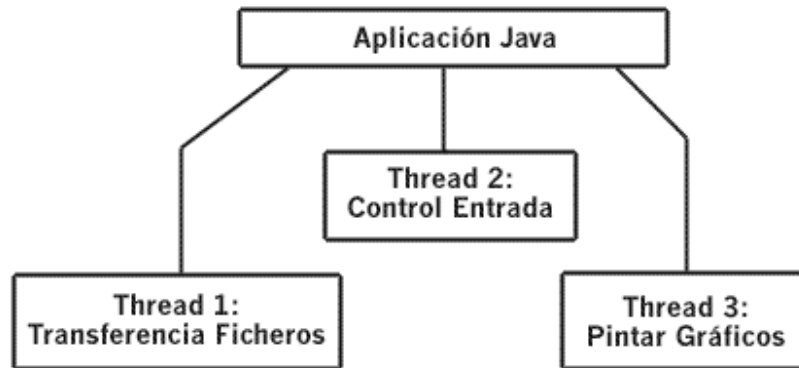
	<b>Red CLNS (datagramas)</b>
<b>Establecimiento conexión</b>	No Orientado a Conexión, no es necesaria la conexión
<b>Direccionamiento</b>	Conmutación de circuitos, encaminamiento independiente para cada paquete, dirección completa de origen y destino
<b>Información de estado</b>	No se conserva información de estado
<b>Routing</b>	La ruta es independiente para cada datagrama
<b>Efecto de fallo en un router</b>	Se pierden los paquetes que se están transmitiendo
<b>Calidad de servicio</b>	Difícil control de paquetes
<b>Control de congestión</b>	Difícil, pero es más flexible ante congestiones en la red
<b>Control de errores y flujo</b>	Fácil

---

<sup>19</sup> <http://es.wikipedia.org/wiki/CLNS>

## 2.10 Multihebras.

Considerando el entorno multithread (multihilo), cada thread (hilo, flujo de control del programa) representa un proceso individual ejecutándose en un sistema. A veces se les llama procesos ligeros o contextos de ejecución. Típicamente, cada hilo controla un único aspecto dentro de un programa, como puede ser supervisar la entrada en un determinado periférico o controlar toda la entrada/salida del disco. Todos los hilos comparten los mismos recursos, al contrario que los procesos, en donde cada uno tiene su propia copia de código y datos (separados unos de otros). Gráficamente, los hilos (threads) se parecen en su funcionamiento a lo que muestra la figura siguiente:



**Figura II. 9** Funcionamiento de los hilo

Hay que distinguir *multihilo* (multithread) de *multiproceso*. El *multiproceso* se refiere a dos programas que se ejecutan "*aparentemente*" a la vez, bajo el control del Sistema Operativo. Los programas no necesitan tener relación unos con otros, simplemente el hecho de que el usuario desee que se ejecuten a la vez.

*Multihilo* se refiere a que dos o más tareas se ejecutan "*aparentemente*" a la vez, dentro de un mismo programa.

Se usa "*aparentemente*" en ambos casos, porque normalmente las plataformas tienen una sola CPU, con lo cual, los procesos se ejecutan en realidad "*concurrentemente*", sino que comparten la CPU. En plataformas con varias CPU, sí es posible que los procesos se ejecuten realmente a la vez.

Tanto en el multiproceso como en el multihilo (multitarea), el Sistema Operativo se encarga de que se genere la ilusión de que todo se ejecuta a la vez. Sin embargo, la multitarea puede producir programas que realicen más trabajo en la misma cantidad de tiempo que el multiproceso, debido a que la CPU está compartida entre tareas de un mismo proceso.

Además, como el multiproceso está implementado a nivel de sistema operativo, el programador no puede intervenir en el planteamiento de su ejecución; mientras que en el caso del multihilo, como el programa debe ser diseñado expresamente para que pueda soportar esta característica, es imprescindible que el autor tenga que planificar adecuadamente la ejecución de cada hilo, o tarea. [<sup>20</sup>]

## **2.11 Multiprocesos**

Dos o más unidades de proceso ejecutando cada una de ellas uno o más procesos. Cada unidad trabaja en un conjunto de instrucciones o en diferentes partes del mismo proceso

---

<sup>20</sup> <http://www.itapizaco.edu.mx/paginas/JavaTut/froufe/parte10/cap10-1.html>

Multiproceso es tradicionalmente conocido como el uso de múltiples procesos concurrentes en un sistema en lugar de un único proceso en un instante determinado. Como la multitarea que permite a múltiples procesos compartir una única CPU, múltiples CPUs pueden ser utilizados para ejecutar múltiples hilos dentro de un único proceso.

El multiproceso para tareas generales es, a menudo, bastante difícil de conseguir debido a que puede haber varios programas manejando datos internos (conocido como estado o contexto) a la vez. Los programas típicamente se escriben asumiendo que sus datos son incorruptibles.

Sin embargo, si otra copia del programa se ejecuta en otro procesador, las dos copias pueden interferir entre sí intentando ambas leer o escribir su estado al mismo tiempo. Para evitar este problema se usa una variedad de técnicas de programación incluyendo semáforos y otras comprobaciones y bloqueos que permiten a una sola copia del programa cambiar de forma exclusiva ciertos valores.

Las computadoras que tienen más de un CPU son llamadas multiproceso. Un sistema operativo multiproceso coordina las operaciones de las computadoras multiprocesadoras. Ya que cada CPU en una computadora de multiproceso puede estar ejecutando una instrucción, el otro procesador queda liberado para procesar otras instrucciones simultáneamente.

Al usar una computadora con capacidades de multiproceso se incrementa su velocidad de respuesta y procesos. Casi todas las computadoras que tienen capacidad de multiproceso ofrecen una gran ventaja. [<sup>21</sup>]

## **2.12 Georeferenciación**

La georeferenciación es un neologismo que refiere al posicionamiento con el que se define la localización de un objeto espacial (representado mediante punto, vector, área, volumen) en un sistema de coordenadas y datos determinados. Este proceso es utilizado frecuentemente en los Sistemas de Información Geográfica.

La georeferenciación, en primer lugar, posee una definición tecno científica, aplicada a la existencia de las cosas en un espacio físico, mediante el establecimiento de relaciones entre las imágenes de raster o vector sobre una proyección geográfica o sistema de coordenadas.

Por ello la georeferenciación se convierte en central para los modelados de datos realizados por los Sistemas de Información Geográfica (SIG). [<sup>22</sup>]

## **2.13 Sistemas de Información Geográfica (SIG)**

A continuación se detalla qué es un Sistema de Información Geográfica (SIG), su importancia, componentes, y lo referente a la Geolocalización.

---

<sup>21</sup> <http://nayar.uan.mx/~iavalos/OS.htm>

<sup>22</sup> <http://es.wikipedia.org/wiki/Geolocalizaci%C3%B3n>

### **2.13.1 ¿Que es un SIG?**

Un Sistema de Información Geográfica (SIG o GIS, en su acrónimo inglés (Geographic Information System) es una integración organizada de hardware, software y datos geográficos diseñada para capturar, almacenar, manipular, analizar y desplegar en todas sus formas la información geográficamente referenciada con el fin de resolver problemas complejos de planificación y gestión. También puede definirse como un modelo de una parte de la realidad referido a un sistema de coordenadas terrestre y construido para satisfacer unas necesidades concretas de información.

En el sentido más estricto, es cualquier sistema de información capaz de integrar, almacenar, editar, analizar, compartir y mostrar la información geográficamente referenciada. En un sentido más genérico, los SIG son herramientas que permiten a los usuarios crear consultas interactivas, analizar la información espacial, editar datos, mapas y presentar los resultados de todas estas operaciones.

La tecnología de los Sistemas de Información Geográfica puede ser utilizada para investigaciones científicas, la gestión de los recursos, gestión de activos, la arqueología, la evaluación del impacto ambiental, la planificación urbana, la cartografía, la sociología, la geografía histórica, el marketing, la logística por nombrar unos pocos.

Por ejemplo, un SIG podría permitir a los grupos de emergencia calcular fácilmente los tiempos de respuesta en caso de un desastre natural, el SIG puede ser usado para encontrar los humedales que necesitan protección contra la contaminación, o pueden ser utilizados

por una empresa para ubicar un nuevo negocio y aprovechar las ventajas de una zona de mercado con escasa competencia. [<sup>23</sup>]

Un sistema de computador capaz de mantener y usar datos con localizaciones exactas en una superficie terrestre.

Un sistema de información geográfica, es una herramienta de análisis de información. La información debe tener una referencia espacial y debe conservar una inteligencia propia sobre la topología y representación.

En general un SIG debe tener la capacidad de dar respuesta a las siguientes preguntas:

- ¿Dónde está el objeto A?
- ¿Dónde está A con relación a B?
- ¿Cuántas ocurrencias del tipo A hay en una distancia D de B?
- ¿Cuál es el valor que toma la función Z en la posición X?
- ¿Cuál es la dimensión de B (Frecuencia, perímetro, área, volumen)?
- ¿Cuál es el resultado de la intersección de diferentes tipos de información?
- ¿Cuál es el camino más corto (menor resistencia o menor costo) sobre el terreno desde un punto (X1, Y1) a lo largo de un corredor P hasta un punto (X2, Y2)?
- ¿Qué hay en el punto (X, Y)?

---

<sup>23</sup> <http://www.monografias.com/trabajos/gis/gis.shtml>



- ¿Qué objetos están próximos a aquellos objetos que tienen una combinación de características?
- ¿Cuál es el resultado de clasificar los siguientes conjuntos de información espacial?
- Utilizando el modelo definido del mundo real, simule el efecto del proceso P en un tiempo T dado un escenario S.

### **2.13.2 Importancia de los SIG**

Las soluciones para muchos problemas frecuentemente requieren acceso a varios tipos de información que sólo pueden ser relacionadas por geografía o distribución espacial. Sólo la tecnología SIG permite almacenar y manipular información usando geografía, analizar patrones, relaciones, y tendencias en la información, todo con el interés de contribuir a la toma de mejores decisiones.

### **2.13.3 Componentes de los SIG**

Los programas de SIG proveen las funciones y las herramientas necesarias para almacenar, analizar y desplegar la información geográfica. Los principales componentes de los programas son:



**Figura II. 10** Elementos que intervienen en un SIG.

### **Hardware**

Los SIG corren en un amplio rango de tipos de computadores desde equipos centralizados hasta configuraciones individuales o de red, una organización requiere de hardware suficientemente específico para cumplir con las necesidades de aplicación.

### **Software**

Los programas SIG proveen las herramientas y funcionalidades necesarias para almacenar, analizar y mostrar información geográfica, los componentes principales del software SIG son:

- Sistema de manejo de base de datos.
- Una interfaz gráfica de usuarios (GUI) para el fácil acceso a las herramientas.

- Herramientas para captura y manejo de información geográfica.
- Herramientas para soporte de consultas, análisis y visualización de datos geográficos.

Actualmente la mayoría de los proveedores de software SIG distribuyen productos fáciles de usar y pueden reconocer información geográfica estructurada en muchos formatos distintos. Además existen organizaciones públicas y privadas que distribuyen software SIG libre.

La captura de gran cantidad de información cartográfica utiliza sistemas automatizados de digitalización como los dispositivos de exploración. Estos minimizan el trabajo manual y aseguran resultados coherentes y repetibles cada vez que se examina un mapa. Aunque la exploración es más rápida que la digitalización, solo pueden someterse a ese proceso los mapas de buena calidad e incluso así, el resultado del producto no es por lo general tan satisfactorio.

Además, una vez digitalizado un mapa puede reproducirse y transformarse a voluntad o de acuerdo a las necesidades establecidas por el usuario.

### **Información**

El componente más importante para un SIG es la información. Se requieren de buenos datos de soporte para que el SIG pueda resolver los problemas y contestar a preguntas de la forma más acertada posible.

La consecución de buenos datos generalmente absorbe entre un 60 y 80 % del presupuesto de implementación del SIG, y la recolección de los datos es un proceso largo que frecuentemente demora el desarrollo de productos que son de utilidad. La información producida solo tiene el valor de los datos introducidos previamente. Una información incorrecta o insuficiente introducida en el SIG produciría respuestas incorrectas o insuficientes, por muy perfeccionada o adaptada al usuario que pueda ser la tecnología. Los datos geográficos y alfanuméricos pueden obtenerse por recursos propios u obtenerse a través de proveedores de datos. Mantener organizar y manejar los datos debe ser política de la organización.

### **Personal**

Las tecnologías SIG son de valor limitado sin los especialistas en manejar el sistema y desarrollar planes de implementación del mismo. Sin el personal experto en su desarrollo, la información se desactualiza y se maneja erróneamente, el hardware y el software no se manipula en todo su potencial. Cuando se define un SIG se tiende a limitar a equipos y programas como el sistema completo, relegando tal vez el elemento más primordial: El talento humano que hace funcionar eficazmente todo el sistema.

### **Métodos**

Para que un SIG tenga una implementación exitosa debe basarse en un buen diseño y reglas de actividad definidas, que son los modelos y practicas operativas exclusivas en cada organización.



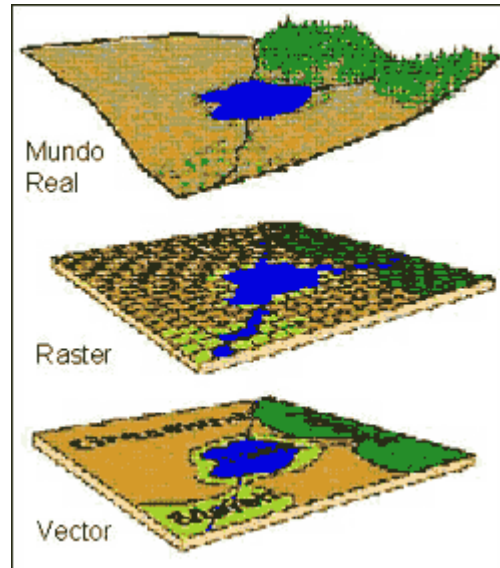
**Figura II. 11** Esquema componentes básicos de un Sistema de Información Geográfica.

La recolección de información y la introducción de la misma en el sistema, requiere de una gran calidad de diseño y trabajo, una capacitación intensiva y un control frecuente para vigilar la calidad. En otras palabras, además de contar con equipos y programas adecuados para realizar el trabajo, la utilización eficaz del SIG requiere contar con personal suficientemente capacitado, así como con servicios de planificación, organización y supervisión, que permitan mantener la calidad de los datos y la integridad de los productos finales.

#### **2.13.4 Funcionamiento de los SIG**

La información espacial contiene una referencia geográfica explícita como latitud y longitud o una referencia implícita como domicilio o código postal. Las referencias implícitas pueden ser derivadas de referencias explícitas mediante geocodificación. Los

SIG funcionan con dos tipos diferentes de información geográfica: el modelo vector y el modelo raster.



**Figura II. 12** Modelo Raster

El modelo raster ha evolucionado para modelar tales características continuas. Una imagen raster comprende una colección de celdas (píxel) de una grilla más como un mapa o una figura escaneada. Ambos modelos para almacenar datos geográficos tienen ventajas y desventajas únicas y los SIG modernos pueden manejar varios tipos.

En el modelo vector, la información sobre puntos, líneas y polígonos se almacena como una colección de coordenadas x, y. La ubicación de una característica puntual, pueden describirse con un sólo punto x, y. Las características lineales, pueden almacenarse como un conjunto de puntos de coordenadas x, y. Las características poligonales, pueden almacenarse como un circuito cerrado de coordenadas.

El modelo vector es extremadamente útil para describir características discretas, pero menos útil para describir características de variación continua. [24]

## **2.14 Sistemas de Gestión de Bases de Datos**

Los sistemas de gestión de bases de datos (en inglés database management system, abreviado DBMS) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.

El propósito general de los sistemas de gestión de bases de datos es el de manejar de manera clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante para una organización.

### **2.14.1 Objetivos de los Sistemas de Gestión de Bases de Datos**

Existen distintos objetivos que deben cumplir los SGBD:

- **Abstracción de la información.** Los SGBD ahorran a los usuarios detalles acerca del almacenamiento físico de los datos. Da lo mismo si una base de datos ocupa uno o cientos de archivos, este hecho se hace transparente al usuario. Así, se definen varios niveles de abstracción.

---

<sup>24</sup> [http://es.wikipedia.org/wiki/Requerimiento\\_\(sistemas\)](http://es.wikipedia.org/wiki/Requerimiento_(sistemas))

- **Independencia.** La independencia de los datos consiste en la capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.
  
- **Consistencia.** En aquellos casos en los que no se ha logrado eliminar la redundancia, será necesario vigilar que aquella información que aparece repetida se actualice de forma coherente, es decir, que todos los datos repetidos se actualicen de forma simultánea. Por otra parte, la base de datos representa una realidad determinada que tiene determinadas condiciones, por ejemplo que los menores de edad no pueden tener licencia de conducir. El sistema no debería aceptar datos de un conductor menor de edad. En los SGBD existen herramientas que facilitan la programación de este tipo de condiciones.
  
- **Seguridad.** La información almacenada en una base de datos puede llegar a tener un gran valor. Los SGBD deben garantizar que esta información se encuentra segura de permisos a usuarios y grupos de usuarios, que permiten otorgar diversas categorías de permisos.
  
- **Manejo de transacciones.** Una transacción es un programa que se ejecuta como una sola operación. Esto quiere decir que luego de una ejecución en la que se produce una falla es el mismo que se obtendría si el programa no se hubiera ejecutado. Los SGBD proveen mecanismos para programar las modificaciones de los datos de una forma mucho más simple que si no se dispusiera de ellos.



- **Tiempo de respuesta.** Lógicamente, es deseable minimizar el tiempo que el SGBD demora en proporcionar la información solicitada y en almacenar los cambios realizados.

### **2.14.2 Ventajas**

Los Sistemas de Gestión de bases de datos brindan algunas ventajas a continuación se detallan algunas de éstas:

- Simplifican la programación de equipos de consistencia.
- Manejando las políticas de respaldo adecuadas, garantizan que los cambios de la base serán siempre consistentes sin importar si hay errores correctamente, etc.
- Organizan los datos con un impacto mínimo en el código de los programas.
- Disminuyen drásticamente los tiempos de desarrollo y aumentan la calidad del sistema desarrollado si son bien explotados por los desarrolladores.
- Usualmente, proveen interfaces y lenguajes de consulta que simplifican la recuperación de los datos.

### **2.14.3 Inconvenientes**

Se pueden presentar inconvenientes entre los cuales se tiene:

- Típicamente, es necesario disponer de una o más personas que administren la base de datos, de la misma forma en que suele ser necesario en instalaciones de cierto porte disponer de una o más personas que administren los sistemas operativos. Esto puede llegar a incrementar los costos de operación en una empresa. Sin embargo hay que balancear este aspecto con la calidad y confiabilidad del sistema que se obtiene.

- Si se tienen muy pocos datos que son usados por un único usuario por vez y no hay que realizar consultas complejas sobre los datos, entonces es posible que sea mejor usar una hoja de cálculo.
- Complejidad: el software muy complejo y las personas que vayan a usarlo deben tener conocimiento de las funcionalidades del mismo para poder aprovecharlo al máximo.
- Tamaño: la complejidad y la gran cantidad de funciones que tienen hacen que sea un software de gran tamaño, que requiere de gran cantidad de memoria para poder correr.
- Coste del hardware adicional: los requisitos de hardware para correr un SGBD por lo general son relativamente altos, por lo que estos equipos pueden llegar a costar gran cantidad de dinero.

#### **2.14.4 Productos SGBD**

Se los ha separado en grupos los que se detallan a continuación:

##### **SGBD libres:**

- MySQL
- PostgreSQL (<http://www.postgresql.org> Postgresql) Licencia BSD
- SQLite
- DB2 Express-C
- Apache Derby

##### **SGBD no libres:**

- MySQL Advantage Database
- dBase

- FileMaker
- Fox Pro
- IBM DB2: Universal Database (DB2 UDB)
- IBM Informix
- Microsoft Access
- Microsoft SQL Server
- NexusDB
- Open Access
- Oracle
- Paradox
- PervasiveSQL
- Progress (DBMS)
- Sybase ASE
- WindowBase

### **SGBD no libres y gratuitos**

- Microsoft SQL Server Compact Edition.
- Sybase ASE Express Edition para Linux (edición gratuita para Linux)
- Oracle Express Edition 10 (solo corre en un servidor, capacidad limitada). [<sup>25</sup>]

---

<sup>25</sup> [http://es.wikipedia.org/wiki/Sistema\\_de\\_gesti%C3%B3n\\_de\\_bases\\_de\\_datos](http://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos)

A continuación se realiza una introducción al Sistema de Gestión de Base de Datos que se utiliza en la implementación de ésta investigación:

### **2.14.5 MySQL**

Es un sistema de gestión de bases de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB desde enero de 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009 desarrolla MySQL como software libre en un esquema de licenciamiento dual.

Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C.

Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y los derechos de autor del código están en poder del autor individual, MySQL es patrocinado por una empresa privada, que posee el copyright de la mayor parte del código.

Esto es lo que posibilita el esquema de licenciamiento anteriormente mencionado. Además de la venta de licencias privativas, la compañía ofrece soporte y servicios. Para sus operaciones contratan trabajadores alrededor del mundo que colaboran vía Internet. MySQL AB fue fundado por David Axmark, Allan Larsson y Michael Widenius.

#### **2.14.5.1 Lenguajes de Programación compatibles con MySQL**

Existen varias APIs que permiten, a aplicaciones escritas en diversos lenguajes de programación, acceder a las bases de datos MySQL, incluyendo C, C++, C#, Pascal, Delphi (via dbExpress), Eiffel, Smalltalk, Java (con una implementación nativa del driver de Java), Lisp, Perl, PHP, Python, Ruby, Gambas, REALbasic (Mac y Linux), FreeBASIC, y Tcl; cada uno de estos utiliza una API específica. También existe una interfaz ODBC, llamado MyODBC que permite a cualquier lenguaje de programación que soporte ODBC comunicarse con las bases de datos MySQL. También se puede acceder desde el sistema SAP, lenguaje ABAP.

#### **2.14.5.2 Plataformas**

MySQL funciona sobre múltiples plataformas, incluyendo las que se listan a continuación:

- GNU/Linux
- AIX
- BSD
- FreeBSD
- HP-UX
- Mac OS X
- NetBSD
- Novell Netware
- OpenBSD
- OS/2 Warp
- QNX

- SGI IRIX
- Solaris
- SunOS
- SCO OpenServer
- SCO UnixWare
- Tru64
- eBD
- Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP, Windows Vista, Windows 7 y Windows Server (2000, 2003 y 2008).
- OpenVMS. [<sup>26</sup>]

---

<sup>26</sup> <http://es.wikipedia.org/wiki/MySQL>

# **CAPÍTULO III**

## **ANÁLISIS COMPARATIVO ENTRE LOS TIPOS DE SERVIDORES**

### **3.1 Introducción**

En la actualidad es muy importante conocer la ubicación geoespacial de un objeto o bien mueble como un vehículo o incluso de los seres queridos como nuestros hijos por lo que se ha visto la necesidad de implementar una aplicación que funcione como una puerta de enlace la misma que utiliza para recibir la información enviada por GPS, la información es almacenada en una base de datos.

Se ha encontrado dos formas de realizar la implementación de la puerta de enlace, el objetivo es hallar la forma de desarrollo más eficiente para su correcto funcionamiento.

En este capítulo se implementa el análisis comparativo entre los tipos de servidores no orientados a la conexión, multihebras y multiprocesos, se lista los parámetros a tomarse en cuenta para realizar la comparación estadística, el desarrollo de los prototipos de pruebas, análisis comparativo, se explica los puntajes alcanzados e interpretación, resultado del análisis y finalmente se realiza la Comprobación de la Hipótesis.

### **3.2 Determinar los parámetros a comparar**

Como se dijo en los antecedentes los parámetros para la comparación van a ser dos en éste caso los que se anotan a continuación:

- CPU
- Memoria Ram

Para el correcto funcionamiento de un computador, es necesario que el procesador en conjunto con la cantidad de Memoria Ram disponible responda de una manera eficiente y rápida.

### **3.3 Desarrollo de los prototipos de pruebas**

Se detalla el proceso que se siguió para la implementación de los prototipos de servidores a comparar como también un programa cliente que se necesitó para realizar dichas pruebas.

Como se explicó en los antecedentes para el desarrollo de los prototipos de pruebas se utilizó el lenguaje de programación Java por ser software libre lo que implica no tener costo de licencias además de ser multiplataforma y el almacenamiento de los datos se lo hizo en el motor de bases de datos MySQL.



Para el desarrollo del prototipo de servidor multiprocesos y multihilos se creó un programa el mismo que actúa como servidor, está activo en un puerto determinado (el puerto es configurable) para receptar los paquetes enviados por los GPSs, el programa servidor multiprocesos invoca otro programa cada vez que llegue un paquete, en cambio el servidor multihilos crea una hebra cada vez que llegue un paquete, cada servidor al finalizar el proceso asume la tarea de descomponer la trama y almacenar en la base de datos.

Para las pruebas se implementó y utilizó una aplicación cliente, la misma que necesita como datos de entrada número de peticiones que se desean enviar y en qué tiempo se lo requiere, finalmente va a leer de un archivo el texto a enviar.

### **3.4 Análisis Comparativo**

A continuación se detallan los recursos, parámetros y técnicas utilizadas, en las pruebas efectuadas para la selección del servidor que mejores resultados obtenga para su completo desarrollo.

Las características físicas del computador en las que se desarrollaron las pruebas de los servidores multiprocesos y multihilos son las siguientes:

- CPU: AMD Turion X2 Dual Core.
- Memoria Ram: 2 GB.

Para la obtención de los resultados las pruebas que se aplicaron fueron pruebas de carga, esto se refiere a cómo responde cada uno de los prototipos en el consumo de los parámetros

comparados, ya que se envió una cierta cantidad de paquetes cada hora para sacar el valor de consumo de los recursos del servidor (computador).

Se utilizó para estos fines una distribución de probabilidad que se denomina Distribución T de Student con la que se determinó las diferencias entre dos medias muestrales y para la construcción del intervalo de confianza el cual mostró la diferencia entre las medias de las dos poblaciones.

El valor de desconfianza con el que se trabajó en la distribución T de Student es del 5%, con éste valor se pudo obtener un intervalo de confianza de acuerdo a los valores de cada uno de los parámetros comparados.

### **3.5 Puntajes Alcanzados**

Para la toma de datos se trabajó en el Sistema operativo Linux, en el cual se ejecutó el comando `SAR -U 5 720` con el que se obtuvieron los datos que se muestran en las siguientes tablas, de las cuales se tomó el campo %idle para realizar los cálculos del uso de CPU. Si %idle tiende a cero = CPU sobrecargado.

**Tabla III. II** Datos Consumo CPU enviando 10000 paquetes/hora

<b>Enviando 10000 paquetes/hora</b>							
<b>MULTIHEBRAS</b>							
MEDIA	CPU	%user	%nice	%system	%iowait	%steal	%idle
	All	6,48	1,36	16,72	3,24	0	<b>71,84</b>
<b>MULTIPROCESOS</b>							
MEDIA	CPU	%user	%nice	%system	%iowait	%steal	%idle
	All	13,67	1,59	14,39	0,25	0	<b>70,09</b>

**Tabla III. III** Datos Consumo CPU enviando 20000 paquetes/hora

<b>Enviando 20000 paquetes/hora</b>							
<b>MULTIHEBRAS</b>							
MEDIA	CPU	%user	%nice	%system	%iowait	%steal	%idle
	All	7,48	1,53	14,27	0,25	0	<b>76,48</b>
<b>MULTIPROCESOS</b>							
MEDIA	CPU	%user	%nice	%system	%iowait	%steal	%idle
	All	8,55	1,4	13,84	0,48	0	<b>75,73</b>

**Tabla III. IV** Datos Consumo CPU enviando 40000 paquetes/hora

<b>Enviando 40000 paquetes/hora</b>							
<b>MULTIHEBRAS</b>							
MEDIA	CPU	%user	%nice	%system	%iowait	%steal	%idle
	All	6,92	1,01	13,9	0,32	0	<b>77,86</b>
<b>MULTIPROCESOS</b>							
MEDIA	CPU	%user	%nice	%system	%iowait	%steal	%idle
	All	6,75	1,41	12,97	0,17	0	<b>78,7</b>

**Tabla III. V** Datos Consumo CPU enviando 60000 paquetes/hora

<b>Enviando 60000 paquetes/hora</b>							
<b>MULTIHEBRAS</b>							
MEDIA	CPU	%user	%nice	%system	%iowait	%steal	%idle
	All	9,59	1,24	22,86	0,58	0	<b>65,73</b>
<b>MULTIPROCESOS</b>							
MEDIA	CPU	%user	%nice	%system	%iowait	%steal	%idle
	All	9,33	1,5	21,51	0,39	0	<b>67,26</b>

**Tabla III. VI** Datos Consumo CPU enviando 80000 paquetes/hora

<b>Enviando 80000 paquetes/hora</b>							
<b>MULTIHEBRAS</b>							
MEDIA	CPU	%user	%nice	%system	%iowait	%steal	%idle
	All	9,03	1,45	30,91	0,32	0	<b>66,3</b>
<b>MULTIPROCESOS</b>							
MEDIA	CPU	%user	%nice	%system	%iowait	%steal	%idle
	All	10,03	1,51	22,19	0,35	0	<b>65,92</b>

En la toma de datos para obtener el consumo de la memoria RAM se ejecutó el comando SAR -r 5 720 y se obtuvieron los datos que se muestran en las siguientes tablas tomando el valor del campo %menused para realizar los respectivos cálculos.

**Tabla III. VII** Datos Consumo Memoria RAM enviando 10000 paquetes/hora

<b>Enviando 10000 paquetes/hora</b>							
<b>MULTIHEBRAS</b>							
	kbmenfree	Kbmenused	%menused	kbbuffers	kbcached	kbcommit	%commit
MEDIA	81148	952560	<b>92,15</b>	47062	536064	570860	34,06
<b>MULTIPROCESOS</b>							
	kbmenfree	Kbmenused	%menused	kbbuffers	kbcached	kbcommit	%commit
MEDIA	50605	983103	<b>95,1</b>	149710	376706	611593	36,49

**Tabla III. VIII** Datos Consumo Memoria RAM enviando 20000 paquetes/hora

<b>Enviando 20000 paquetes/hora</b>							
<b>MULTIHEBRAS</b>							
	kbmenfree	Kbmenused	%menused	kbbuffers	kbcached	kbcommit	%commit
MEDIA	84231	949477	<b>91,85</b>	30895	552957	572227	34,14
<b>MULTIPROCESOS</b>							
	kbmenfree	Kbmenused	%menused	kbbuffers	kbcached	kbcommit	%commit
MEDIA	48104	985604	<b>95,35</b>	104517	472214	565600	33,74

**Tabla III. IX** Datos Consumo Memoria RAM enviando 40000 paquetes/hora

<b>Enviando 40000 paquetes/hora</b>							
<b>MULTIHEBRAS</b>							
	Kbmenfree	Kbmenused	%menused	kbbuffers	kbcached	kbcommit	%commit
MEDIA	71624	962084	<b>93,07</b>	27243	561045	595908	35,55
<b>MULTIPROCESOS</b>							
	Kbmenfree	Kbmenused	%menused	kbbuffers	kbcached	kbcommit	%commit
MEDIA	44385	989323	<b>95,71</b>	39186	569624	603737	36,02

**Tabla III. X** Datos Consumo Memoria RAM enviando 60000 paquetes/hora

<b>Enviando 60000 paquetes/hora</b>							
<b>MULTIHEBRAS</b>							
	kbmenfree	Kbmenused	%menused	kbbuffers	kbcached	kbcommit	%commit
MEDIA	27174	1006534	<b>97,37</b>	102005	495282	583284	34,8
<b>MULTIPROCESOS</b>							
	kbmenfree	Kbmenused	%menused	kbbuffers	kbcached	kbcommit	%commit
MEDIA	25081	1008627	<b>97,57</b>	114634	475319	590705	35,24

**Tabla III. XI** Datos Consumo Memoria RAM enviando 80000 paquetes/hora

<b>Enviando 80000 paquetes/hora</b>							
<b>MULTIHEBRAS</b>							
	kbmenfree	Kbmenused	%menused	kbbuffers	kbcached	Kbcommit	%commit
MEDIA	69420	964288	<b>93,28</b>	127738	401253	593100	35,38
<b>MULTIPROCESOS</b>							
	kbmenfree	Kbmenused	%menused	kbbuffers	kbcached	Kbcommit	%commit
MEDIA	31993	1001715	<b>96,91</b>	140186	406513	611519	36,48

Luego de realizadas las pruebas de rendimiento donde se puso cada uno de los prototipos a recibir datagramas UDP, se obtuvieron los valores que a continuación muestran en la Tabla III.XII y Tabla III XIII, después de efectuar el estudio estadístico descriptivo de dichos datos se alcanzaron los resultados como se describe en el Anexo I.

En dicho Anexo I se detallan los valores alcanzados entre las dos medias poblacionales, varianza, desviación estándar, intervalo de confianza, se detalla las fórmulas aplicadas en el

proceso; el concepto de las medidas de variabilidad se describe en la sección de Glosario de Términos.

### **Servidor Multihebras**

Los valores que se muestran en la Tabla III XII indican el consumo de CPU y de memoria RAM, por ejemplo cuando se envió 10000 paquetes/hora el CPU estuvo libre o desocupado un 71,84 % y se consumió el 92,15 % de memoria RAM.

El valor de la media en el uso de CPU que muestra la tabla III. XII indica que este parámetro estuvo libre en un 71,64 % como promedio, lo cual muestra que éste recurso no corría el peligro de colapsar, más bien su consumo fue moderado ya que se consumió solo un 28.36% de su capacidad total.

Por otra parte el valor de la media en el uso de memoria RAM que muestra la tabla III. XII indica que este parámetro fue consumido en un 93,54 % como promedio, lo cual muestra que este recurso estuvo siendo utilizado casi en su totalidad ya que solo un 6,46% de su capacidad total se podía utilizar.

**Tabla III. XII** Datos Multihebras

	<b>CPU</b>	<b>MEMORIA RAM</b>
	<b>% idle</b>	<b>% menused</b>
<b>enviando 10000 p/h</b>	71,84	92,15
<b>enviando 20000 p/h</b>	76,48	91,85
<b>enviando 40000 p/h</b>	77,86	93,07
<b>enviando 60000 p/h</b>	65,73	97,37
<b>enviando 80000 p/h</b>	66,3	93,28
<b>MEDIA</b>	<b>71,642</b>	<b>93,544</b>

### **Servidor Multiprocesos**

Los valores que se muestran en la tabla III. XIII indican el consumo de CPU y de memoria RAM, por ejemplo cuando se envió 10000 paquetes / hora el CPU estuvo libre un 70,09 % y se consumió el 95,1 % de memoria RAM.

El valor de la media en el uso de CPU que muestra la tabla III. XIII indica que este parámetro estuvo libre en un 71,54% como promedio, eso expresa claramente que este recurso no se encontró en peligro de colapsar, más bien su consumo fue moderado ya que se consumió solo un 28.46 % de su capacidad total.

Por otra parte el valor de la media en el uso de memoria RAM que muestra la tabla III XIII indica que este parámetro fue consumido en un 96,128 % como promedio, eso indica que



este recurso estuvo siendo ocupado casi en su totalidad ya que solo un 3,872 % de su capacidad total estuvo disponible.

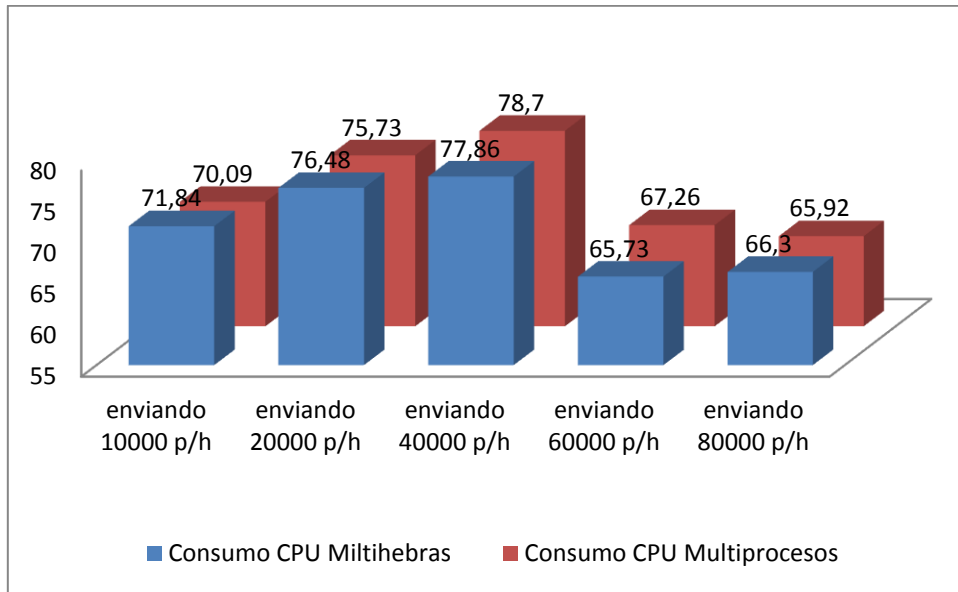
**Tabla III. XIII** Datos Multiproceso

	<b>CPU</b>	<b>MEMORIA RAM</b>
	<b>% idle</b>	<b>% menused</b>
<b>enviando 10000 p/h</b>	70,09	95,1
<b>enviando 20000 p/h</b>	75,73	95,35
<b>enviando 40000 p/h</b>	78,7	95,71
<b>enviando 60000 p/h</b>	67,26	97,57
<b>enviando 80000 p/h</b>	65,92	96,91
<b>MEDIA</b>	<b>71,54</b>	<b>96,128</b>

### **3.6 Interpretación**

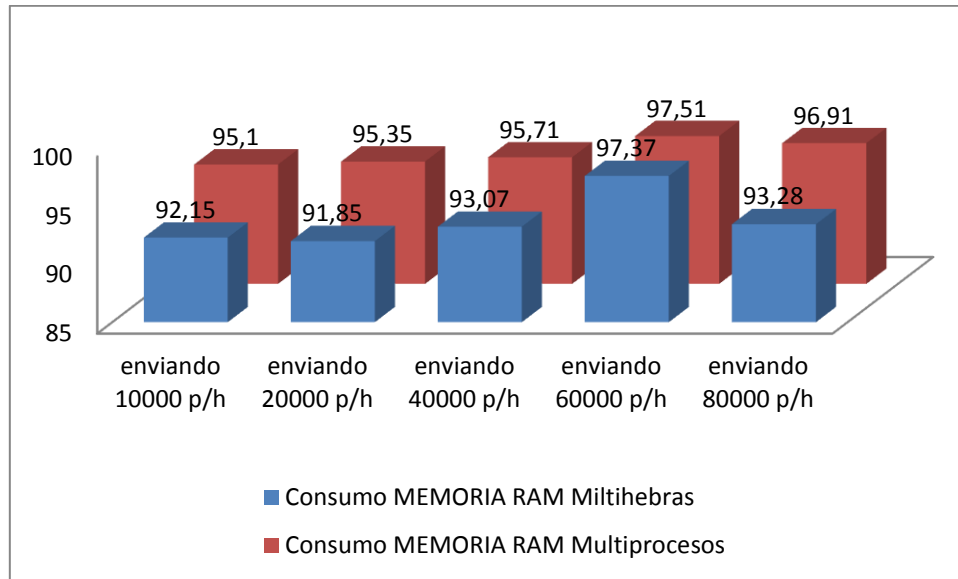
**Conclusión 1:** Estadísticamente se ha llegado a la conclusión que no existe diferencia significativa en el consumo de CPU entre las dos formas de implementar la Puerta de Enlace ya que los valores hallados (medias del consumo de cpu) se encuentran dentro del intervalo de confianza con un error admisible de  $\pm 4,96$ .

A continuación se muestra una gráfica con la comparación de los valores obtenidos.



**Figura III. 13** Comparación consumo de CPU

**Conclusión 2 :** Estadísticamente se ha llegado a la conclusión que si existe diferencia significativa en el consumo de memoria RAM entre las dos formas de implementar la Puerta de Enlace ya que uno de los valores hallados (medias del consumo de memoria RAM) no se encuentran dentro del intervalo de confianza con error admisible de  $\pm 0,0211$ . A continuación se muestra una gráfica con la comparación de los valores obtenidos.



**Figura III. 14** Comparación consumo de Memoria RAM

### 3.7 Resultado del análisis

**Conclusión final:** Se ha utilizado Estadística Descriptiva y Estadística Inferencial para poder hacer el análisis y escoger el servidor más eficiente; en el consumo de CPU no se ha encontrado una diferencia significativa, pero los datos obtenidos con el siguiente parámetro comparado que es memoria RAM demostraron que el servidor multihebras realiza un menor consumo por lo que se ha elegido el mismo para el completo desarrollo del proyecto.

### 3.8 Comprobación de la Hipótesis

Hipótesis: El estudio estadístico de los tipos de servidores no orientados a la conexión, multihebras y multiprocesos, permitirá seleccionar el más eficiente para el desarrollo de una puerta de enlace con los equipos de rastreo móvil de Enfora.

Causa: El estudio estadístico de los tipos de servidores no orientados a la conexión, multihebras y multiprocesos.

Efecto: Selecciona el más eficiente para el desarrollo de una puerta de enlace con los equipos de rastreo móvil de Enfora.

**Tabla III. XIV Operacionalización conceptual**

<b>VARIABLE</b>	<b>TIPO</b>
Tipos de Servidores No orientados a la conexión	Independiente
Consumo de CPU	Dependiente
Consumo de Memoria RAM	Dependiente

**Tabla III. XV Operacionalización metodológica**

<b>VARIABLE</b>	<b>INDICADORES</b>	<b>TECNICAS</b>	<b>FUENTE DE VERIFICACIÓN</b>
Tipos de Servidores no orientados a la conexión	#funcionamiento #tiempo de ejecución #envío de paquetes	Observación	Prototipo Multihebras Prototipo Multiproceso
Consumo de CPU	#rendimiento	Experimentación Observación	Envío de paquetes
Consumo de Memoria RAM	# rendimiento	Experimentación Observación	Envío de paquetes

Para la selección del servidor más eficiente se ha tomado aquel que menor uso del CPU y/o de la memoria RAM requiera.

De los datos que se describen en el Anexo I se obtiene que el intervalo de confianza para el consumo de CPU sea  $IC = [66,68; 76,6]$  y el intervalo de confianza para el consumo de memoria RAM sea  $IC = [91,43; 95,65]$ ; los valores obtenidos de multihebras y multiprocesos son:

**Tabla III. XVI Datos Finales**

	<b>Multihebras</b>	<b>Multiprocesos</b>
<b>CPU</b>	71,642	71,54
<b>Memoria RAM</b>	93,54	96,128

Dado que en el uso de CPU no existen estadísticamente diferencias significativas, entonces este parámetro no ha sido determinante para escoger el tipo de servidor a implementar.

Por otra parte, dado que existen estadísticamente diferencias significativas con el consumo de memoria RAM se ha seleccionado el servidor que menos uso de éste recurso tuvo durante las pruebas (servidor multihebras).

Del estudio estadístico de los tipos de servidores no orientados a la conexión, multihebras y multiprocesos, seleccionamos como el más eficiente (el que menos memoria RAM utilizó) el servidor multihebras para el desarrollo de la puerta de enlace con los equipos de rastreo móvil de Enfora.

# **CAPÍTULO IV**

## **IMPLEMENTACIÓN DEL SERVIDOR**

### **4.1 Introducción**

Como se mencionó en la Justificación para la implementación del servidor se utilizó la metodología ágil XP la cual consta de varias fases, las mismas que se detallan a continuación:

- Planificación.
- Diseño.
- Codificación.
- Pruebas.

### **4.2 Planificación**

En la planificación se desarrolla una pequeña descripción del sistema, se detalla los requerimientos tanto funcionales como no funcionales, se escriben historias de usuario, iteraciones y la velocidad del proyecto.

#### **4.2.1. Descripción del Sistema**

La tesis a implementar es el estudio estadístico de tipos de servidores, para la comunicación con plataformas de rastreo móvil, en la implementación de una Puerta de Enlace para equipos Enfora.

Se desarrolló la Puerta de Enlace la misma que cumple la función de recibir la información que emitan los GPS para su procesamiento y/o almacenamiento, una aplicación adicional que realiza tareas de administración (configuración) de la Puerta de Enlace.

La Puerta de Enlace se desarrolló utilizando el lenguaje de programación Java, con lo cual se logró un servidor multiplataforma (instalar en varios sistemas operativos como Windows, Linux, etc.), de la misma forma para el almacenamiento de la información se utilizó el motor de base de datos MySQL por su rapidez y poco consumo de recurso hardware, además a esto, tanto Java como MySQL no tienen un costo adicional a la empresa por la utilización de licencias por ser software libre.

#### **4.2.2 Requerimientos**

Para el desarrollo de la Puerta de Enlace, se necesita cumplir con los siguientes requerimientos:

### **Requerimientos Funcionales**

Los requerimientos funcionales son una descripción de lo que un sistema debe hacer. Los requerimientos denotan algo que el sistema entregado debe ser capaz de realizar. A continuación se detallan los requerimientos del sistema:

#### **Módulo de Recepción de Datos**

1. Recibir información de los GPS
2. Obtener e interpretar a su formato original los campos del datagrama.
3. Almacenar los campos en la base de datos y en un archivo de texto plano.

#### **Módulo de Envío de Datos**

4. Seleccionar un dispositivo GPS para transmitir un comando AT requerido.

#### **Módulo de Administración**

5. Manipular estados de funcionamiento de la Puerta de Enlace (encender, apagar, reiniciar).
6. Mostrar información de los vehículos registrados agrupados por empresa y refrescar el sistema, dado un tiempo (en segundos).
7. Mostrar información de los GPS que han registrado determinado evento.

### **Requerimientos No Funcionales**

Especifica algo sobre el propio sistema, y cómo debe realizar sus funciones. Algunos ejemplos de aspectos solicitables son la disponibilidad, escalabilidad, una interfaz amigable, etc.



**Disponibilidad.-** Se debe tener en cuenta que el sistema deberá estar disponible durante las 24 horas del día.

**Escalable.-** Presentar un sistema flexible ya que se pueden incrementar nuevas funciones de acuerdo a los requerimientos por el crecimiento de la empresa.

**Interfaz Amigable.-** El sistema debe contar con una interfaz comprensible y de fácil manejo para el usuario.

#### **4.2.3 Historias de Usuario**

El objetivo de las historias de usuario es describir un caso de uso en dos o tres líneas con terminología del cliente (de hecho, se supone que deben ser escritos por el mismo).

A continuación se describen algunas historias de usuario que fueron implementadas en el desarrollo de la Puerta de Enlace.

La primera historia de usuario requiere recibir la información que los GPS's envían, para esto es necesario que el servidor (hardware y software) debe estar encendido y funcionando.

La segunda historia de usuario requiere obtener e interpretar a su formato original los campos del datagrama, para lo cual se requiere el Evolution Binary Format (documento para interpretar el tamaño y significado de los campos que son enviados por los GPS).

Las historias de usuario en su totalidad se encuentran desarrolladas en el Anexo II.

#### **4.2.4 Iteraciones**

Se detalla a continuación el plan de iteraciones y el plan de entrega.

### **Plan de Iteraciones**

El desarrollo se divide en iteraciones, en cada una de las cuales se eligen las historias de usuario a desarrollar y las tareas de desarrollo.

Se eligieron siete historias de usuario para su desarrollo las mismas que tienen un tiempo de implementación de cuatro semanas cada una. El plan de iteraciones se describe en el Anexo III.

### **Plan de Entrega**

Desde la iteración número 1 hasta la iteración 4 es el módulo de Recepción de datos el cual cumple la función de recibir la información de los GPS la misma que es transformada a hexadecimal para posteriormente obtener e interpretar en código ascii los campos del datagrama los cuales finalmente son almacenados en una base de datos MySQL y en un archivo respectivamente esto tuvo una duración de 4 meses.

La iteración número 5 ayuda a enviar un comando AT a un GPS específico cualquiera que se desee. La duración de esta iteración fue de un mes.

Desde la iteración numero 6 hasta la iteración 9 es el modulo de Administración permite realizar las siguientes funciones:

Manipular estados de funcionamiento del Gateway server (encender, apagar, reiniciar).

Mostrar información de los vehículos registrados agrupados por empresa y refrescar el sistema dado un tiempo (en segundos).

Mostrar información de los GPS que encuentran en estado de pánico y refrescar el sistema dado un tiempo (en segundos).

Este módulo se desarrolló en aproximadamente 4 meses.

#### **4.2.5 Velocidad del proyecto**

Cada iteración se desarrolló en cuatro semanas con lo que se concluye que el proyecto se culminó en aproximadamente nueve meses.

### **4.3 Diseño**

A continuación se detalla el contenido que se desarrolló según la metodología XP en cada una de las subetapas del diseño.

#### **4.3.1 Diseños simples**

La metodología XP sugiere que hay que conseguir diseños simples y sencillos. Hay que procurar hacerlo todo lo menos complicado posible para conseguir un diseño fácilmente entendible e implementable que a la larga costará menos tiempo y esfuerzo desarrollar.

Se desarrollaron métodos pequeños y simples para utilizarlos en métodos complejos que requieran su uso para la ejecución de tareas con mayor grado de dificultad, los métodos simples pueden ser llamados en métodos de la misma clase o en métodos de clases distintas para optimizar recursos y facilitar el desarrollo del proyecto.

### **4.3.2 Riesgos**

Teniendo en cuenta que la explotación de un riesgo causa daños o pérdidas financieras o administrativas a una empresa u organización, se tiene la necesidad de poder estimar la magnitud del impacto del riesgo a que se encuentra expuesta mediante la aplicación de controles.

Por lo que para este proyecto se ha realizado una Gestión de Riesgos lo cual consta de la identificación de los mismos.

Una vez identificados los riesgos se realizó su descripción, categorización, análisis de sus posibles consecuencias, se los ubicó en un rango de probabilidad de ocurrencia, determinación del impacto, determinación de la exposición al riesgo, determinación de la prioridad del riesgo y finalmente la hoja de gestión de riesgos en la cual constan los valores hallados anteriormente y una posible solución a ellos, esto se realizó de los riesgos cuya exposición se cataloga entre media y alta.

La descripción de los riesgos que se encontraron se detalla en el Anexo IV

### **4.3.3 Funcionalidad extra**

En la implementación de la puerta de enlace no se añadió ninguna funcionalidad extra aunque se piense que en un futuro será utilizada, ya que sólo el 10% de la misma es utilizada, lo que implica que el desarrollo de ella es un desperdicio de tiempo y recursos.

#### **4.3.4 Tarjetas C.R.C**

Para el desarrollo de la puerta de enlace se utilizó las tarjetas C.R.C (Class, Responsibilities and Collaboration) ya que permiten al programador centrarse y apreciar el desarrollo orientado a objetos olvidándose de los malos hábitos de la programación clásica.

Las tarjetas C.R.C representan objetos, la clase a la que pertenece el objeto se puede escribir en la parte de arriba de la tarjeta, en una columna a la izquierda se pueden escribir las responsabilidades u objetivos que debe cumplir el objeto y a la derecha, las clases que colaboran con cada responsabilidad. Las tarjetas C.R.C se encuentran detalladas en el Anexo V.

#### **4.4 Codificación**

La codificación debe hacerse atendiendo a estándares de codificación ya creados. Programar bajo estándares mantiene el código consistente y facilita su comprensión y escalabilidad.

Por lo que en el desarrollo de la Puerta de Enlace se siguió un formato de codificación como por ejemplo el nombre de las clases comienzan con las letras cls y el nombre representativo de la clase como clsPaquete la cual es la que tiene la funcionalidad principal de transformar el paquete recibido a lo que necesita el usuario.

Los nombre de los métodos dentro de dicha clase son nombres que describen la función que realizan como el método ToHexadecimal () que devuelve transformados los símbolos que llega del GPS hacia la Puerta de Enlace a formato hexadecimal o el método sendCmds ()

que envía los comandos AT que se encuentran en un determinado archivo a través del socket abierto hacia el GPS.

#### **4.5 Pruebas**

Uno de los pilares de la metodología X.P es el uso de test para comprobar el funcionamiento de los códigos que vayamos implementando.

Por lo que para la clase clsServidorUDP que es la que recibe la información en un determinado puerto y llama a la clase clsPaquete para que almacene dicha información se ejecutó una aplicación cliente que envía paquetes a la Puerta de Enlace para comprobar dicha funcionalidad.

Otra prueba que se realizó es la descomposición y transformación del paquete enviado por un GPS a texto que se pueda entender para luego ser almacenado en la base de datos implementada; esto y algunas funciones adicionales las ejecuta la clase clsPaquete con algunos de sus métodos como: DescomponerTrama63(), ToHexadecimal(), ToDecimal (), InsertarDB(), etc.

##### **4.5.1 Pruebas de aceptación**

Los test mencionados anteriormente sirven para evaluar las distintas tareas en las que ha sido dividida una historia de usuario. Para asegurar el funcionamiento final de una determinada historia de usuario se deben crear "Pruebas de aceptación".

Al ser las distintas funcionalidades de la aplicación no demasiado extensas, no se hicieron test que analicen partes de las mismas, sino que las pruebas se realizaron para las funcionalidades generales que debe cumplir el programa especificado en la descripción de requisitos.

Las pruebas de aceptación que se realizaron fueron en función a las características que debe cumplir el sistema y son las que se describen a continuación:

- El sistema debe tener la capacidad de recibir los paquetes que son enviados por los GPS y transformarlos en su formato original esto lo podemos comprobar cuando estos sean almacenados en el archivo rds.log que se encuentra en el directorio var/log/rdsd/
- La información debe ser insertada en una tabla de la base de datos, en caso de ocurrir un error la sentencia SQL es almacenada en un archivo de texto.
- Se debe realizar el envío de un comando AT a un dispositivo en particular el mismo que se comprueba con el envío de un paquete de confirmación por parte del GPS.
- El estado de la Puerta de Enlace (encendido, apagado) se puede comprobar con el comando SERVICE RDSD STATUS desde la consola de administración del servidor.
- La información de los vehículos registrados agrupados por empresa y los GPS que han registrado determinado evento se lo puede observar en el módulo de administración.

## CONCLUSIONES

- En el estudio estadístico que se realizó entre los tipos de servidores no orientados a la conexión multihebras y multiprocesos no existió diferencias significativas en el consumo de CPU pero si en el consumo de Memoria RAM con 93,544% para el servidor multihebras y 96,128% para el servidor multiprocesos con un margen de error del 5%, concluyéndose que por el menor consumo de Memoria RAM el servidor multihebras es el más eficiente.
- Se determinó estadísticamente y se desarrolló el servidor no orientado a la conexión más eficiente para los equipos Enfora.
- Al implementar los prototipos de los servidores no orientados a la conexión por multihebras y multiprocesos, en el primero se invoca a una clase de la misma aplicación para que realice el proceso de descomposición y/o almacenamiento de la información y en el segundo se invoca a una aplicación diferente para que realice el mismo proceso.
- Por medio del desarrollo de los prototipos se pudo comparar los parámetros como: el consumo de Memoria RAM y del CPU, realizando pruebas en tiempo de ejecución.



- El estudio estadístico efectuado entre los dos tipos de servidores no orientados a la conexión multihebras y multiprocesos dio como resultado que si existen diferencias significativas en el consumo de uno de los recursos comparados, lo que ayudó en el desarrollo de la investigación.
  
- Aplicando la técnica t-Student con los datos obtenidos en las pruebas de carga que se realizaron se pudo encontrar los intervalos de confianza con los cuáles se obtuvo como resultado que el más eficiente por el menor consumo de Memoria RAM es el servidor multihebras.
  
- El desarrollo de la puerta de enlace de los dispositivos de rastreo móvil de Enfora se logró mediante la utilización de herramientas de software libre las cuales son: MySQL como motor de base de datos, Java el lenguaje de programación y el sistema operativo Linux Centos, representando una gran ventaja en costo y tiempo de desarrollo.

## **RECOMENDACIONES**

- La definición de los parámetros para la comparación de los servidores se la debe realizar previo a un análisis para determinar las ventajas de desarrollo de cada tipo, tomando en cuenta el uso que se les va a dar en lo posterior.
- El desarrollo de prototipos debe abarcar tareas básicas de funcionamiento de las que debe realizar el proyecto final para de esta manera realizar la comparación de los servidores.
- Para el desarrollo de la Puerta de Enlace se requiere de conocimientos técnicos en el funcionamiento de los dispositivos Enfora.
- Para la utilización de MySQL se necesitan estudios previos acerca de la gestión y administración de base de datos.

## **RESUMEN**

El Estudio estadístico de tipos de servidores para la comunicación con plataformas de rastreo móvil en la implementación de una Puerta de Enlace para equipos Enfora, se realizó con el fin de determinar el tipo de servidor más eficiente; para la empresa Rastreo Directo Satelital (RDS) que radica en la ciudad de Ambato.

Se aplicó el método estadístico utilizando la distribución t de Student para la comparación del consumo de CPU y Memoria RAM de los prototipos de servidores no orientados a la conexión multihebras y multiprocesos los cuales se implementaron en Java bajo el sistema operativo Linux (Centos) en los que se realizaron pruebas de carga de 10000, 20000, 40000, 60000, 80000 paquetes/hora comparándose las medias obtenidas, no existiendo diferencias significativas en el consumo de CPU pero si en el consumo de Memoria RAM con 93,544% para el servidor multihebras y 96,128% para el servidor multiprocesos con un margen de error del 5%.

Concluyéndose que por el menor consumo de Memoria RAM el servidor multihebras es el más eficiente, eligiéndose éste para desarrollar en su totalidad la Puerta de Enlace utilizando la metodología ágil de desarrollo XP e implementándose una aplicación web para su administración.

Se recomienda en la definición de los parámetros para la comparación de los servidores se la debe realizar previo a un análisis para determinar las ventajas de desarrollo de cada tipo, tomando en cuenta el uso que se les va a dar en lo posterior.

## **SUMMARY**

The statistical analysis of mobile-tracking platform communication server types in the gateway implementation for Enfora equipment was carried out to determine the most efficient server for the enterprise Rastreo Directo Satelital (RDS) located in the city of Ambato.

Statistical method was applied by using the t distribution from Student to compare the CPU consumption with RAM memory of the server prototypes that are not intended to the multithreading and multiprocessing connection to be implemented in Java considering the operative system Linux (Centos) in which load tests of 10000, 20000, 40000, 60000, 80000 packets/hour were carried out. It was also necessary to compare to gotten measures. There were no meaningful differences in the CPU consumption but RAM memory consumption with 93,544% for the multithread server and 96,128% for the multiprocessing server with an error margin of 5%.

It is concluded that the most efficient server is multithread one because it uses less RAM memory. It was chosen to develop the gateway fully by using the fast XP development methodology and using a web application for its running.

It is recommended the server comparison must be done with a prior analysis to determine the advantages for each type, taking into account further usage.

## **GLOSARIO DE TÉRMINOS**

### **AVL**

El sistema de Rastreo Vehicular Automatizado (RVA) o AVL, acrónimo de Automatic Vehicle Location, se aplica a los sistemas de localización remota en tiempo real, basados generalmente en el uso de un GPS y un sistema de transmisión que es frecuentemente un módem inalámbrico.

### **CPU**

La Unidad Central de Procesamiento o CPU (por el acrónimo en inglés de Central Processing Unit), o simplemente el procesador o microprocesador, es el componente del computador y otros dispositivos programables, que interpreta las instrucciones contenidas en los programas y procesa los datos.

### **Desviación Estándar**

La desviación estándar o desviación típica ( $\sigma$ ) es una medida de centralización o dispersión para variables de razón (ratio o cociente) y de intervalo, de gran utilidad en la estadística descriptiva.

Se define como la raíz cuadrada de la varianza. Junto con este valor, la desviación típica es una medida (cuadrática) que informa de la media de distancias que tienen los datos respecto de su media aritmética, expresada en las mismas unidades que la variable.

## **Intervalo de Confianza**

Se llama intervalo de confianza en estadística a un par de números entre los cuales se estima que estará cierto valor desconocido con una determinada probabilidad de acierto. Formalmente, estos números determinan un intervalo, que se calcula a partir de datos de una muestra, y el valor desconocido es un parámetro poblacional. La probabilidad de éxito en la estimación se representa por  $1 - \alpha$  y se denomina nivel de confianza. En estas circunstancias,  $\alpha$  es el llamado error aleatorio o nivel de significación, esto es, una medida de las posibilidades de fallar en la estimación mediante tal intervalo.<sup>1</sup>

El nivel de confianza y la amplitud del intervalo varían conjuntamente, de forma que un intervalo más amplio tendrá más posibilidades de acierto (mayor nivel de confianza), mientras que para un intervalo más pequeño, que ofrece una estimación más precisa, aumentan sus posibilidades de error. Su fórmula es:

$$IC = \bar{x} \pm z(S\bar{x}) \quad \Rightarrow \text{donde} \quad S\bar{x} = \frac{S}{\sqrt{5}}$$

*Z = ValorDeDesconfianza(5%)  
DeLaDistribuciónTdeStudent*

## **Java**

La palabra Java, por sí misma, se refiere habitualmente al lenguaje de programación Java, que fue diseñado para usar con la Plataforma Java. Los lenguajes de programación se encuentran fuera del ámbito de lo que es una “plataforma”, aunque el lenguaje de programación Java es uno de los componentes fundamentales de la plataforma Java. El propio lenguaje y el entorno en tiempo de ejecución suelen considerarse una única entidad.

## **Media**

En matemáticas y estadística, la media aritmética (también llamada promedio o simplemente media) de un conjunto finito de números es igual a la suma de todos sus valores dividida entre el número de sumandos. Cuando el conjunto es una muestra aleatoria recibe el nombre de media muestral siendo uno de los principales estadísticos muestrales.

Expresada de forma más intuitiva, podemos decir que la media (aritmética) es la cantidad total de la variable distribuida a partes iguales entre cada observación.

## **Metodología XP**

La programación extrema es una metodología de desarrollo ligera (o ágil) se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software.

## **MySQL**

Es un sistema de gestión de bases de datos relacional, licenciado bajo la GPL de la GNU. Su diseño multihilo le permite soportar una gran carga de forma muy eficiente. MySQL fue creada por la empresa sueca MySQL AB, que mantiene el copyright del código fuente del servidor SQL, así como también de la marca.

Aunque MySQL es software libre, MySQL AB distribuye una versión comercial de MySQL, que no se diferencia de la versión libre más que en el soporte técnico que se

ofrece, y la posibilidad de integrar este gestor en un software propietario, ya que de no ser así, se vulneraría la licencia GPL.

Este gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración.

## **OSI**

Siglas que significan Open Systems Interconnection o Interconexión de Sistemas Abiertos. Es un modelo o referente creado por la ISO para la interconexión en un contexto de sistemas abiertos. Se trata de un modelo de comunicaciones estándar entre los diferentes terminales y host. Las comunicaciones siguen unas pautas de siete niveles preestablecidos que son Físico, Enlace, Red, Transporte, Sesión, Presentación y Aplicación.

## **Protocolo FTP**

Es el acrónimo de File Transfer Protocol, cuyo significado es Protocolo de Transferencia de Archivos. Como su nombre indica, se trata de un protocolo de comunicación remota para transferir ficheros entre ordenadores.

## **Protocolo UDP**

UDP son las siglas de Protocolo de Datagrama de Usuario (en inglés User Datagram Protocol) un protocolo sin conexión que, como TCP, funciona en redes IP.



## **Sistemas de Información Geográfica**

Un Sistema de Información Geográfica (SIG o GIS, en su acrónimo inglés) es una integración organizada de hardware, software y datos geográficos diseñado para capturar, almacenar, manipular, analizar y desplegar en todas sus formas la información geográficamente referenciada con el fin de resolver problemas complejos de planificación y gestión. También puede definirse como un modelo de una parte de la realidad referido a un sistema de coordenadas terrestre y construido para satisfacer unas necesidades concretas de información. En el sentido más estricto, es cualquier sistema de información capaz de integrar, almacenar, editar, analizar, compartir y mostrar la información geográficamente referenciada. En un sentido más genérico, los SIG son herramientas que permiten a los usuarios crear consultas interactivas, analizar la información espacial, editar datos, mapas y presentar los resultados de todas estas operaciones.

## **Sistema de Posicionamiento Global**

El GPS (Global Positioning System: sistema de posicionamiento global) o NAVSTAR-GPS es un sistema global de navegación por satélite (GNSS) que permite determinar en todo el mundo la posición de un objeto, una persona o un vehículo con una precisión hasta de centímetros (si se utiliza GPS diferencial), aunque lo habitual son unos pocos metros de precisión.

## **Varianza**

En teoría de probabilidad, la varianza ( $\sigma^2$ ) de una variable aleatoria es una medida de su dispersión definida como la esperanza del cuadrado de la desviación de dicha variable respecto a su media.

Está medida en unidades distintas de las de la variable. Por ejemplo, si la variable mide una distancia en metros, la varianza se expresa en metros al cuadrado. La desviación estándar, la raíz cuadrada de la varianza, es una medida de dispersión alternativa expresada en las mismas unidades. Su fórmula es:

$$S^2 = \frac{\sum (x_i - \bar{x})^2}{n - 1}$$

# BIBLIOGRAFÍA

## REFERENCIAS WEB GENERAL

### 1. CLIENTE (INFORMÁTICA)

[http://es.wikipedia.org/wiki/Cliente\\_\(inform%C3%A1tica\)](http://es.wikipedia.org/wiki/Cliente_(inform%C3%A1tica))  
2010-03-15

### 2. COMO HACER SCRIPTS EN LINUX

<http://www.ucm.es/info/aulasun/archivos/SCRIPTS.pdf>  
2010-03-16

### 3. CONCURRENCIA

<http://trevinca.ei.uvigo.es/~nrufino/so/Teoria/Tema2.pdf>  
2010-03-15

### 4. DEMONIOS LINUX

[http://es.wikipedia.org/wiki/Demonio\\_%28inform%C3%A1tica%29](http://es.wikipedia.org/wiki/Demonio_%28inform%C3%A1tica%29)  
[http://www.linuxtotal.com.mx/index.php?cont=info\\_admon\\_003](http://www.linuxtotal.com.mx/index.php?cont=info_admon_003)  
2010-03-16

### 5. GEOLOCALIZACIÓN

<http://www.idg.es/computerworld/articulo.asp?id=167772>  
2010-03-15

### 6. GEOREFERENCIACIÓN

<http://es.wikipedia.org/wiki/Geolocalizaci%C3%B3n>  
2010-08-31

### 7. HEBRAS (INFORMÁTICA)

<http://gsm.gui.uva.es/revista/login/15/process.html>  
<http://javahispano.org>  
2010-03-15

### 8. LENGUAJE DE PROGRAMACIÓN

[http://es.wikipedia.org/wiki/Lenguaje\\_de\\_programaci%C3%B3n](http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n)  
<http://www.iec.csic.es/criptonomicon/java/quesjava.html>  
[http://es.wikipedia.org/wiki/Visual\\_Basic\\_.NET](http://es.wikipedia.org/wiki/Visual_Basic_.NET)  
2010-03-16

## **9. MULTITHILOS**

<http://www.itapizaco.edu.mx/paginas/JavaTut/froufe/parte10/cap10-1.html>  
2010-08-31

## **10. MySQL**

<http://es.wikipedia.org/wiki/MySQL>  
2010-08-31

## **11. PUERTA DE ENLACE**

[http://es.wikipedia.org/wiki/Puerta\\_de\\_enlace](http://es.wikipedia.org/wiki/Puerta_de_enlace)  
2010-03-15

## **12. QUÉ ES UN SERVIDOR**

<http://www.masadelante.com/faqs/servidor>.  
<http://es.wikipedia.org/wiki/Servidor>  
2010-08-04

## **13. REQUERIMIENTOS SISTEMA**

[http://es.wikipedia.org/wiki/Requerimiento\\_\(sistemas\)](http://es.wikipedia.org/wiki/Requerimiento_(sistemas))  
2010-08-31

## **14. SISTEMA DE GESTIÓN DE BASES DE DATOS**

[http://es.wikipedia.org/wiki/Sistema\\_de\\_gesti%C3%B3n\\_de\\_bases\\_de\\_datos](http://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos)  
2010-08-31

## **15. SISTEMA DE INFORMACIÓN GEOGRÁFICA**

[http://es.wikipedia.org/wiki/Sistema\\_de\\_Informaci%C3%B3n\\_Geogr%C3%A1fica](http://es.wikipedia.org/wiki/Sistema_de_Informaci%C3%B3n_Geogr%C3%A1fica)  
<http://www.monografias.com/trabajos/gis/gis.shtml>  
2010-06-14

## **16. SISTEMAS OPERATIVOS**

[http://catarina.udlap.mx/u\\_dl\\_a/tales/documentos/lis/marquez\\_a\\_bm/capitulo5.pdf](http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/marquez_a_bm/capitulo5.pdf)  
[http://es.wikipedia.org/wiki/Historia\\_de\\_los\\_sistemas\\_operativos](http://es.wikipedia.org/wiki/Historia_de_los_sistemas_operativos)  
<http://es.wikipedia.org/wiki/GNU/Linux>  
2010-03-15

# **ANEXOS**

# **ANEXO I**

## **Desarrollo de los Cálculos Probabilísticos**

En este anexo se muestra las fórmulas aplicadas y los valores que se establecieron para tomar la decisión de cuál de las dos formas de implementar un servidor es más eficiente.

Los valores que se encontraron son la media, la varianza, desviación estándar e intervalo de confianza de cada una de las formas de desarrollo (multiprocesos, multihilos) así como de los parámetros a comparar (CPU, Memoria RAM).

En resumen con los valores que se realizó los cálculos son los que se muestran en las siguientes tablas.

### **S. Multihebras (CPU)**

	<b>CPU</b>	<b>MEMORIA RAM</b>
	<b>% idle</b>	<b>% menused</b>
<b>enviando 10000 p/h</b>	71,84	92,15
<b>enviando 20000 p/h</b>	76,48	91,85
<b>enviando 40000 p/h</b>	77,86	93,07
<b>enviando 60000 p/h</b>	65,73	97,37
<b>enviando 80000 p/h</b>	66,3	93,28
<b>MEDIA</b>	<b>71,642</b>	<b>93,544</b>

### Media

$$\bar{x} = 71,64$$

### Varianza

$$S^2 = \frac{\sum (x_i - \bar{x})^2}{n - 1}$$

$$S^2 = \frac{0,04 + 23,43 + 38,69 + 34,93 + 28,52}{5 - 1}$$

$$S^2 = \frac{125,61}{4}$$

$$S^2 = 31,40$$

### Desviación Estándar

$$S = \sqrt{S^2}$$

$$S = \sqrt{31,40}$$

$$S = 5,6$$

### Intervalo de Confianza

$$IC = \bar{x} \pm z(S\bar{x}) \quad \Rightarrow \text{donde}$$

$$S\bar{x} = \frac{S}{\sqrt{5}} = \frac{5,6}{2,24} = 2,33$$

$Z = \text{ValorDeDesconfianza}(5\%)$   
*DeLaDistribuciónTdeStudent*

$$IC = 71,64 \pm 2,13(2,33)$$

$$IC = 71,64 \pm 4,96$$

$$IC = [66,68; 76,6]$$



## S. Multihebras (Memoria Ram)

### Media

$$\bar{x} = 93,54$$

### Varianza

$$S^2 = \frac{\sum (x_i - \bar{x})^2}{n - 1}$$

$$S^2 = \frac{1,93 + 2,86 + 0,22 + 14,67 + 0,07}{5 - 1}$$

$$S^2 = \frac{19,75}{4}$$

$$S^2 = 4,94$$

### Desviación Estándar

$$S = \sqrt{S^2}$$

$$S = \sqrt{4,94}$$

$$S = 2,22$$

### Intervalo de Confianza

$$IC = \bar{x} \pm z(S\bar{x}) \quad \text{donde} \Rightarrow \quad S\bar{x} = \frac{S}{\sqrt{5}} = \frac{2,22}{2,24} = 0,99$$

*Z = ValorDeDesconfianza(5%)  
DeLaDistribuciónTdeStudent*

$$IC = 93,54 \pm 2,13(0,99)$$

$$IC = 93,54 \pm 2,11$$

$$IC = [91,43;95,65]$$

## S. Multiprocesos (CPU)

	CPU	MEMORIA RAM
	% idle	% menused
enviando 10000 p/h	70,09	95,1
enviando 20000 p/h	75,73	95,35
enviando 40000 p/h	78,7	95,71
enviando 60000 p/h	67,26	97,57
enviando 80000 p/h	65,92	96,91
<b>MEDIA</b>	<b>71,54</b>	<b>96,128</b>

### Media

$$\bar{x} = 71,54$$

### Varianza

$$S^2 = \frac{\sum (x_i - \bar{x})^2}{n - 1}$$

$$S^2 = \frac{2,10 + 17,56 + 51,27 + 18,32 + 31,58}{5 - 1}$$

$$S^2 = \frac{120,83}{4}$$

$$S^2 = 30,21$$

### Desviación Estándar

$$S = \sqrt{S^2}$$

$$S = \sqrt{30,21}$$

$$S = 5,5$$

## Intervalo de Confianza

$$S\bar{x} = \frac{S}{\sqrt{5}} = \frac{5,5}{2,24} = 2,46$$

$$IC = \bar{x} \pm z(S\bar{x}) \quad \Rightarrow \text{donde}$$

$Z = \text{ValorDeDesconfianza}(5\%)$   
*DeLaDistribuciónTdeStudent*

$$IC = 71,54 \pm 2,13(2,46)$$

$$IC = 71,54 \pm 5,24$$

$$IC = [66,3;76,78]$$

## S. Multiprocesos (Memoria Ram)

### Media

$$\bar{x} = 96,128$$

### Varianza

$$S^2 = \frac{\sum (x_i - \bar{x})^2}{n - 1}$$

$$S^2 = \frac{1,01 + 0,60 + 0,17 + 2,08 + 0,78}{5 - 1}$$

$$S^2 = \frac{4,64}{4}$$

$$S^2 = 1,16$$

### Desviación Estándar

$$S = \sqrt{S^2}$$

$$S = \sqrt{1,16}$$

$$S = 1,08$$

## Intervalo de Confianza

$$S_{\bar{x}} = \frac{S}{\sqrt{5}} = \frac{1,08}{2,24} = 0,48$$

$$IC = \bar{x} \pm z(S_{\bar{x}}) \quad \Rightarrow \text{donde}$$

*Z = ValorDeDesconfianza(5%)  
DeLaDistribuciónTdeStudent*

$$IC = 96,128 \pm 2,13(0,48)$$

$$IC = 96,128 \pm 1,02$$

$$IC = [95,108;97,148]$$

# **ANEXO II**

## **Historias de Usuario**

<b>Historia de Usuario</b>	
<b>Número Tarea: 01</b>	
<b>Nombre Tarea:</b> Recibir información de los GPS	
<b>Tipo de Tarea :</b> Desarrollo	
<b>Prioridad de Negocio:</b> Alta	<b>Prioridad de Desarrollo:</b> Alta
<b>Programador Responsable:</b> Angel Montesdeoca	
<b>Descripción:</b> El servidor (hardware y software) debe estar encendido y funcionando, se procede a esperar el envío de un datagrama desde los GPS hacia el mismo.	

<b>Historia de Usuario</b>	
<b>Número Tarea: 02</b>	
<b>Nombre Tarea:</b> Obtener e interpretar a su formato original los campos del datagrama.	
<b>Tipo de Tarea :</b> Desarrollo	
<b>Prioridad de Negocio:</b> Alta	<b>Prioridad de Desarrollo:</b> Alta
<b>Programador Responsable:</b> Angel Montesdeoca	
<b>Descripción:</b> La información que se ha interpretado a hexadecimal deberá ser seleccionada de acuerdo al Evolution Binary Format (documento para interpretar el tamaño y significado de los campos que son enviados por los GPS) y transformada a su formato original.	

<b>Historia de Usuario</b>	
<b>Número Tarea: 03</b>	
<b>Nombre Tarea:</b> Almacenar los campos en la base de datos y en un archivo de texto plano.	
<b>Tipo de Tarea :</b> Desarrollo	
<b>Prioridad de Negocio:</b> Alta	<b>Prioridad de Desarrollo:</b> Alta
<b>Programador Responsable:</b> Angel Montesdeoca	
<b>Descripción:</b> Una vez obtenidos los atributos del GPS serán almacenados en una base de datos MySQL y los datos como IP y puerto se guardarán en un archivo de texto plano.	

<b>Historia de Usuario</b>	
<b>Número Tarea: 04</b>	
<b>Nombre Tarea:</b> Seleccionar un dispositivo GPS para transmitir un comando AT requerido.	
<b>Tipo de Tarea :</b> Desarrollo	
<b>Prioridad de Negocio:</b> Alta	<b>Prioridad de Desarrollo:</b> Alta
<b>Programador Responsable:</b> Angel Montesdeoca	
<b>Descripción:</b> Seleccionar la IP y el PUERTO de un GPS para enviar un comando AT.	

<b>Historia de Usuario</b>	
<b>Número Tarea: 05</b>	
<b>Nombre Tarea:</b> Manipular estados de funcionamiento del Gateway server (encender, apagar, reiniciar).	
<b>Tipo de Tarea :</b> Desarrollo	
<b>Prioridad de Negocio:</b>	<b>Prioridad de Desarrollo:</b>
<b>Programador Responsable:</b> Angel Montesdeoca	
<b>Descripción:</b> El servidor debe estar en constante funcionamiento en caso de que el administrador crea conveniente se realizarán las respectivas alternativas.	

<b>Historia de Usuario</b>	
<b>Número Tarea: 06</b>	
<b>Nombre Tarea:</b> Mostrar información de los vehículos registrados agrupados por empresa y refrescar el sistema, dado un tiempo (en segundos).	
<b>Tipo de Tarea :</b> Desarrollo	
<b>Prioridad de Negocio:</b>	<b>Prioridad de Desarrollo:</b>
<b>Programador Responsable:</b> Angel Montesdeoca	
<b>Descripción:</b> Visualizar la información (ID, Alias, Status, Odómetro, etc.) si una empresa tiene a su disposición varios vehículos estos se mostrarán de forma agrupada.	

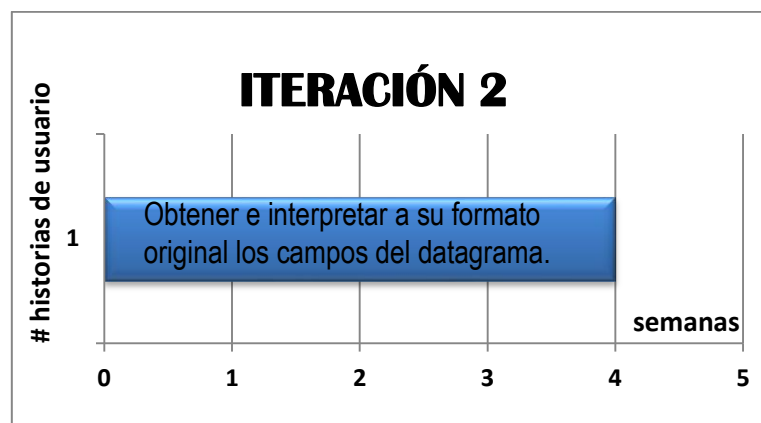
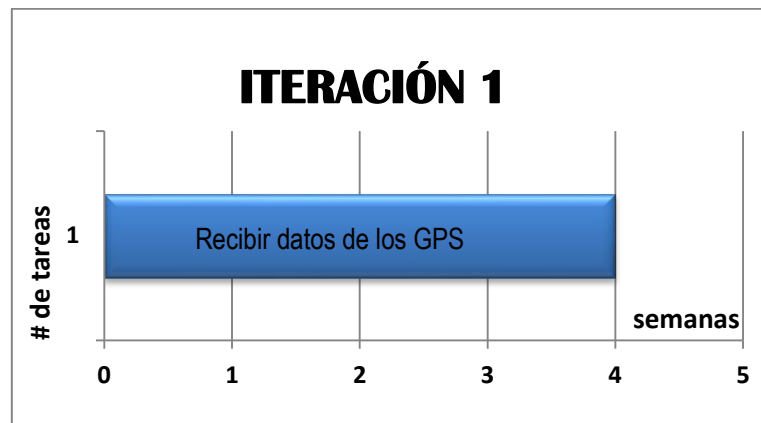
<b>Historia de Usuario</b>	
<b>Número Tarea: 07</b>	
<b>Nombre Tarea:</b> Mostrar información de los GPS que han registrado determinado evento	
<b>Tipo de Tarea :</b> Desarrollo	
<b>Prioridad de Negocio:</b>	<b>Prioridad de Desarrollo:</b>
<b>Programador Responsable:</b> Angel Montesdeoca	
<b>Descripción:</b> Mostrar la información de los vehículos dado un evento que el usuario escoja.	



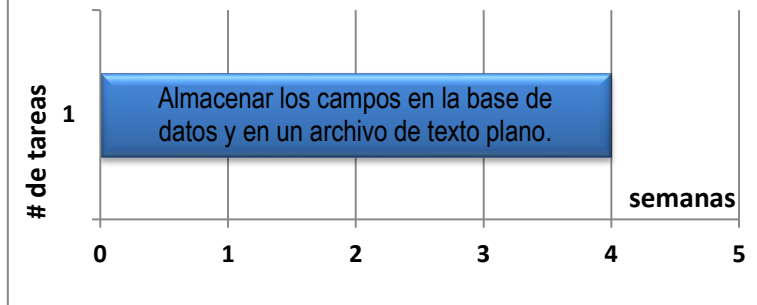
# **ANEXO III**

## **Plan de Iteraciones**

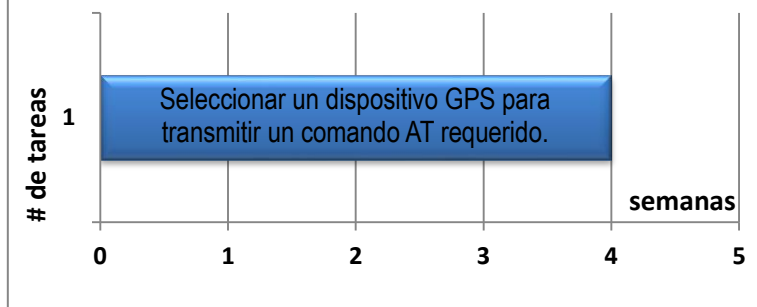
En cada una de las iteraciones se muestra el número de tareas que se va a llevar a cabo y el tiempo en el que se desarrollará el cual está dado en semanas. El tiempo para el desarrollo de cada una de las iteraciones es de 4 semanas por iteración.



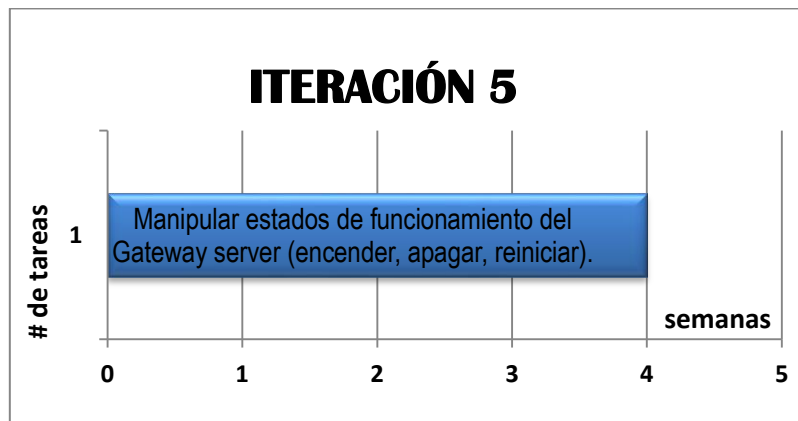
### ITERACIÓN 3



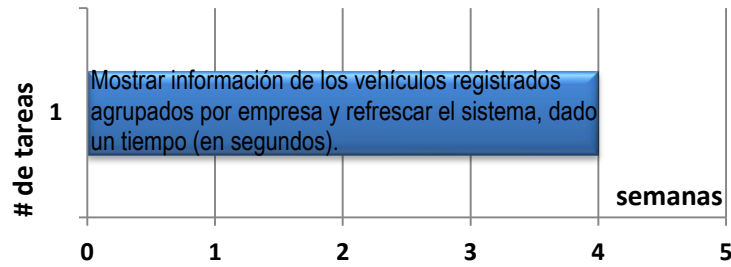
### ITERACIÓN 4



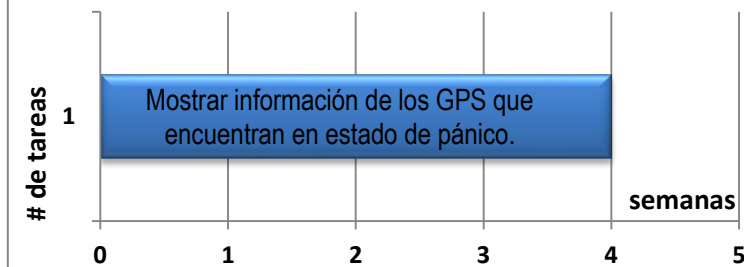
### ITERACIÓN 5



## ITERACIÓN 6



## ITERACIÓN 7



# **ANEXO IV**

## **Gestión de Riegos**

## DESCRIPCION DE LOS RIESGOS

ID	DESCRIPCIÓN DEL RIESGO	CAT	CONSECUENCIA
R1	Error en la conexión con la base de datos.	R Proyecto	Los datos no son almacenados.
R2	Los usuarios modifican los requerimientos del proyecto durante el desarrollo del software.	R Proyecto	Retraso del proyecto, trabajo baldío del Software.
R3	La interfaz es poco amigable para el usuario.	R Técnico	Pérdida de tiempo en la capacitación al personal que va a utilizar el SW.
R4	El SW tiene errores en su programación.	R Técnico	Retraso del proyecto.
R5	Falta de control de seguridad en la Base de Datos.	R Técnico	Sabotaje de la información confidencial de la empresa.
R6	No existan los archivos de configuración.	R Proyecto	El sistema no pueda entrar en funcionamiento.
R7	El servidor no cumpla con las características hardware mínimas para el funcionamiento del sistema.	R Técnico	El sistema no cumpla con su propósito al 100%.

Rango de probabilidad	Descripción	Valor
1% - 33%	Baja	1
34% - 67%	Media	2
68% - 99%	Alta	3

Identificación	Probabilidad		
	%	Valor	Probabilidad
R1	70	3	Alta
R2	50	2	Media
R3	30	1	Baja
R4	10	1	Baja
R5	10	1	Baja
R6	5	1	Baja
R7	20	1	Baja

### Determinación del impacto

Impacto	Retraso	Impacto técnico	Impacto del Costo	Valor
Bajo	1 semana	Ligero efecto en el desarrollo del proyecto	< 1%	1
Moderado	2 semanas	Moderado efecto en el desarrollo del proyecto	< 5%	2
Alto	3 semanas	Severo efecto en el desarrollo del proyecto	< 10%	3
Crítico	1 mes	Proyecto no puede ser culminado	> =10%	4

Identificación	Impacto	
	Valor	Impacto
R1	1	Bajo
R2	4	Critico
R3	2	Moderado
R4	1	Bajo
R5	3	Alto
R6	1	Bajo
R7	1	Bajo

**Determinación de la exposición al riesgo**

Exposición	Valor	Color
Baja	1 o 2	Verde
Media	3 o 4	Amarillo
Alta	Mayor a 4	Rojo

Impacto \ Probabilidad	Bajo =1	Moderado =2	Alto =3	Crítico =4
Alta = 3	3	6	9	12
Media = 2	2	4	6	8
Baja = 1	1	2	3	4



Identificación	Exposición al riesgo	
	Valor	Exposición
R1	3	Media
R2	8	Alta
R3	2	Baja
R4	1	Baja
R5	3	Media
R6	1	Baja
R7	1	Baja

### Determinación de la prioridad del riesgo

ID	DESCRIPCION	EXPOSICION	PRIORIDAD
R2	Los usuarios modifican los requerimientos del proyecto durante el desarrollo del software.	8	
R1	Error en la conexión con la base de datos.	3	
R5	Falta de control de seguridad en la base de datos.	3	
R3	La interfaz es poco amigable	2	
R4	El Software tiene errores en su programación.	1	
R6	No existen los archivos de configuración.	1	
R7	El servidor no cumple con las características hardware mínimas para el funcionamiento del sistema.	1	

Línea de Corte

<b>HOJA DE GESTIÓN DEL RIESGO</b>			
<b>ID DEL RIESGO:</b> R2		<b>FECHA:</b> 12/10/2010	
<b>Probabilidad:</b> Baja <b>Valor:</b> 2	<b>Impacto:</b> Crítico <b>Valor:</b> 4	<b>Exposición:</b> Alta <b>Valor:</b> 8	<b>Prioridad:</b> 1
<b>DESCRIPCIÓN:</b> Los usuarios modifican los requerimientos del proyecto durante el desarrollo del software.			
<b>REFINAMIENTO:</b>  <b>Causa:</b> El usuario no pone en claro todos los requerimientos del software.  <b>Consecuencia:</b> Retraso del proyecto, trabajo baldío del Software.			
<b>REDUCCIÓN:</b> Aplicar correctamente las técnicas de recopilación de información.			
<b>SUPERVISIÓN:</b> Planificar reuniones continuas con usuarios y desarrolladores antes de la implementación del proyecto hasta que los requerimientos estén totalmente definidos.			
<b>GESTIÓN:</b> Planificar reuniones periódicas durante el desarrollo del proyecto.			
<b>ESTADO ACTUAL:</b>			
Fase de reducción iniciada:		<input checked="" type="checkbox"/>	
Fase de supervisión iniciada:		<input type="checkbox"/>	
Gestionando el riesgo:		<input type="checkbox"/>	
<b>RESPONSABLES:</b> Desarrollador.			

<b>HOJA DE GESTIÓN DEL RIESGO</b>			
<b>ID DEL RIESGO:</b> R1		<b>FECHA:</b> 12/10/2010	
<b>Probabilidad:</b> Alta <b>Valor:</b> 3	<b>Impacto:</b> Bajo <b>Valor:</b> 1	<b>Exposición:</b> Alta <b>Valor:</b> 3	<b>Prioridad:</b> 2
<b>DESCRIPCIÓN:</b> Error en la conexión con la base de datos			
<b>REFINAMIENTO:</b>			
<b>Causas:</b> El servicio de MySQL (demonio) deje de funcionar.			
<b>Consecuencias:</b> La base de datos no almacena todos paquetes enviados por los GPS.			
<b>REDUCCIÓN:</b> Verificar si el servicio (MySQL) fue instalado correctamente.			
<b>SUPERVISIÓN:</b> Consultar si la información se esta almacenando en el archivo de texto rds.sql			
<b>GESTIÓN:</b> Reiniciar el servicio.			
<b>ESTADO ACTUAL:</b>			
Fase de reducción iniciada:	<input checked="" type="checkbox"/>		
Fase de supervisión iniciada:	<input type="checkbox"/>		
Gestionando el riesgo:	<input type="checkbox"/>		
<b>RESPONSABLES:</b> Desarrollador.			

## HOJA DE GESTIÓN DEL RIESGO

**ID DEL RIESGO:** R5

**FECHA:** 12/10/2010

**Probabilidad:** Baja  
**Valor:** 1

**Impacto:** Alta  
**Valor:** 3

**Exposición:** Media  
**Valor:** 3

**Prioridad:** 3

**DESCRIPCIÓN:** Falta de control de seguridad en la base de datos.

### REFINAMIENTO:

**Causa:** Problemas en la implementación y desarrollo.

**Consecuencia:** Pérdida de tiempo, realizar pruebas de funcionamiento

**REDUCCIÓN:** Ejecutar pruebas durante el desarrollo del software

**SUPERVISIÓN:** Detectar problemas durante el avance del software y corregirlos

**GESTIÓN:** Realizar un informe de todos los errores encontrados en el transcurso del desenvolvimiento de la aplicación.

### ESTADO ACTUAL:

Fase de reducción iniciada:



Fase de supervisión iniciada:



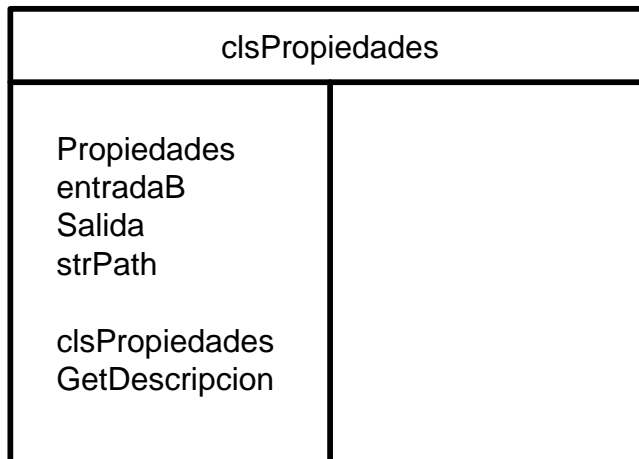
Gestionando el riesgo:



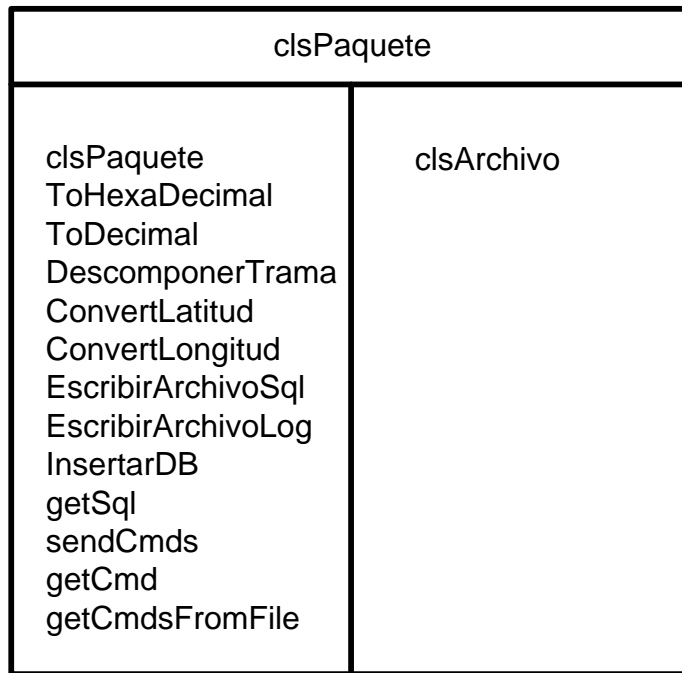
**RESPONSABLES:** Desarrollador.

# **ANEXO V**

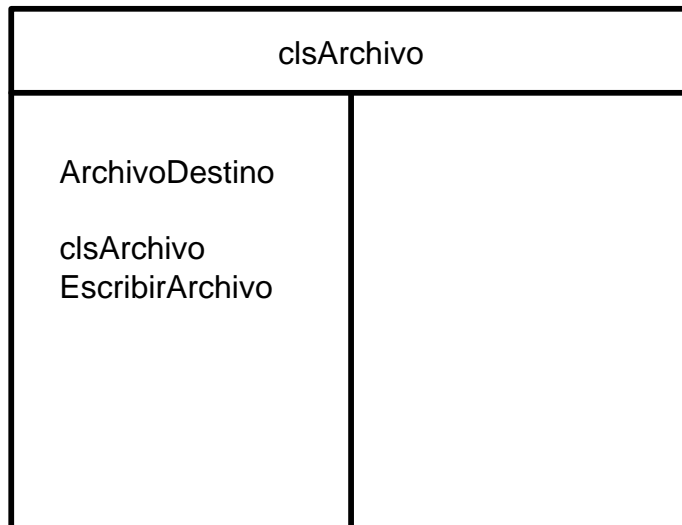
**Tarjetas C.R.C**



La tarjeta CRC de la clase clsPropiedades muestra los atributos con los que cuenta estos son: entradaB, salida, strPath, las mismas que son inicializadas en el constructor clsPropiedades y el método más relevante es GetDescripcion el cual nos ayuda a obtener los valores que necesitamos para el correcto funcionamiento de la Puerta de Enlace que se encuentran en un archivo de configuración. Esta clase no necesita de la colaboración de ninguna clase adicional.



La tarjeta CRC de la clase clsPaquete muestra los métodos que posee para realizar su objetivo que es el de descomponer, transformar, almacenar y enviar comandos hacia los GPS's. Además muestra que ésta clase depende de la otra como lo es la clase clsArchivo porque puede establecer algunos parámetros como la ubicación de los archivos donde va a escribir los diferentes eventos a acciones que se presenten.



La tarjeta CRC de la clase `clsArchivo` entre lo más importante muestra el método `EscribirArchivo` el mismo que permite escribir los errores o lo que designe el método en el cual es llamado.



# **ANEXO VI**

## **Manual de Usuario**

## **Introducción**

La empresa Rastreo Directo Satelital (RDS) se dedica al rastreo vehicular, por medio de GPS instalado en los mismos, ubicándolos en un mapa para mejorar la experiencia de los usuarios del sistema.

Para el análisis de la información que los GPS envían, RDS utiliza una Puerta de Enlace del cual, en el presente manual, detallaremos el proceso de instalación y manipulación.

## **Presentación**

La Puerta de Enlace es una aplicación que tiene la función de recibir la información (datagramas) enviados por los GPSs, los mismos que van a ser decodificado para luego ser almacenado en una Base de Datos, donde se encuentra por ejemplo, la posición geoespacial (latitud, longitud, altitud), velocidad, números de satélites, etc. Además, permite el envío de órdenes a ser cumplidas por los equipos instalados en los vehículos.

## **Requisitos mínimos del sistema**

El computador donde se ejecute el sistema Puerta de Enlace, para su correcto funcionamiento, debe cumplir los siguientes requisitos mínimos:

### Requisitos Hardware

- Memoria Ram de 1 GB
- Disco duro 120 GB

### Requisitos Software

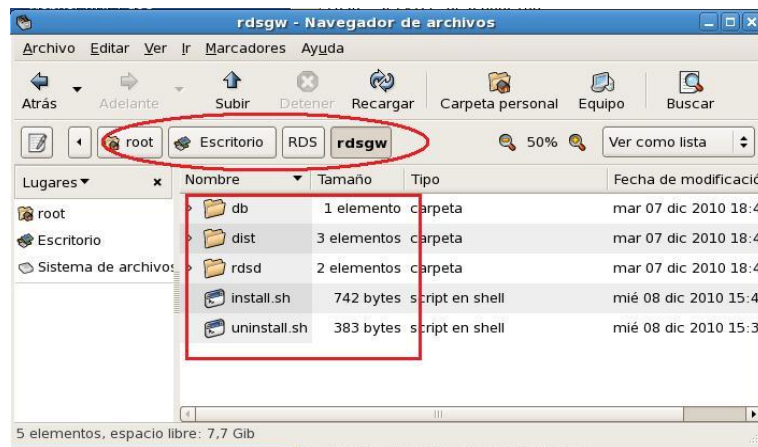
- Sistema Operativo Linux (Centos 5.x)
- Máquina Virtual de Java
- Servidor de páginas Web Apache 2.2
- Motor de Base de Datos MySQL 5.0
- Lenguaje de programación PHP 5.1 o posterior
- Sistema Administrador de Contenidos Joomla 1.5.17

## Instalación de la Puerta de Enlace

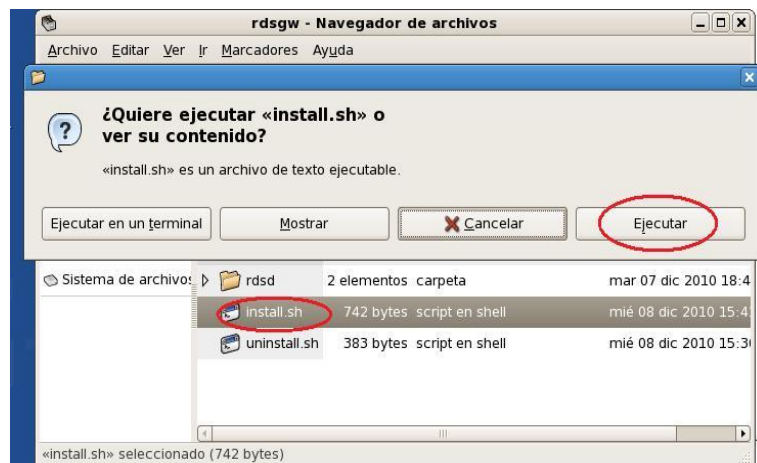
A continuación se detallan las formas para la instalación del servidor puerta de enlace para su correcto funcionamiento.

Forma1: Estos pasos los pueden utilizar usuarios comunes

- Se debe ubicar en la carpeta donde se tenga el instalador de la Puerta de Enlace.



- Una vez dentro de la carpeta del instalador ejecutar el archivo install.sh dando doble click, posteriormente a dar click en el botón ejecutar.



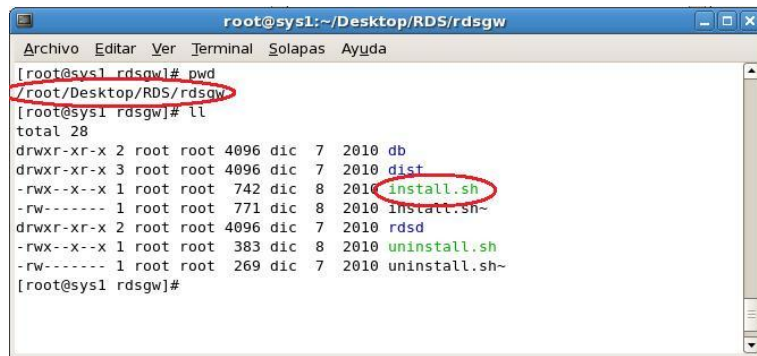
- Se comprueba que la instalación de la Puerta de Enlace se ejecutó, esto lo podemos realizar poniendo el comando `service rdsd status` la misma que debe responder “`rdsd está parado`” como muestra la siguiente figura.



```
root@sys1:~  
Archivo Editar Ver Terminal Solapas Ayuda  
[root@sys1 ~]# service rdsd status  
rdsd está parado  
[root@sys1 ~]#
```

Forma2: Este procedimiento es recomendado para usuario que tengan un conocimiento básico sobre el sistema operativo Linux.

- Abrir una consola de comandos y situarse en la carpeta donde se encuentre el instalador de la Puerta de Enlace, en este caso está en `/root/Desktop/RDS/rdsgw`.



```
root@sys1:~/Desktop/RDS/rdsgw  
Archivo Editar Ver Terminal Solapas Ayuda  
[root@sys1 rdsgw]# pwd  
/root/Desktop/RDS/rdsgw  
[root@sys1 rdsgw]# ll  
total 28  
drwxr-xr-x 2 root root 4096 dic 7 2010 db  
drwxr-xr-x 3 root root 4096 dic 7 2010 dist  
-rwx--x--x 1 root root 742 dic 8 2010 install.sh  
-rw----- 1 root root 771 dic 8 2010 install.sh~  
drwxr-xr-x 2 root root 4096 dic 7 2010 rdsd  
-rwx--x--x 1 root root 383 dic 8 2010 uninstall.sh  
-rw----- 1 root root 269 dic 7 2010 uninstall.sh~  
[root@sys1 rdsgw]#
```

- Ejecutar el archivo `install.sh` de la siguiente forma



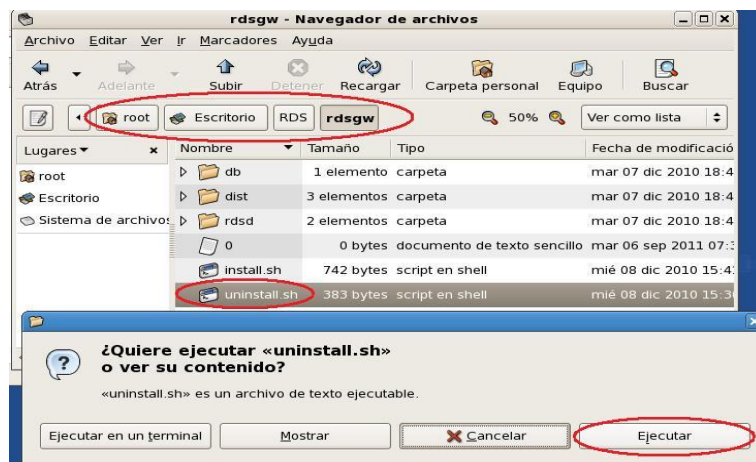
```
root@sys1:~/Desktop/RDS/rdsgw  
Archivo Editar Ver Terminal Solapas Ayuda  
[root@sys1 rdsgw]# ./install.sh  
[root@sys1 rdsgw]#
```

- Listo se ha finalizado el proceso de instalación.

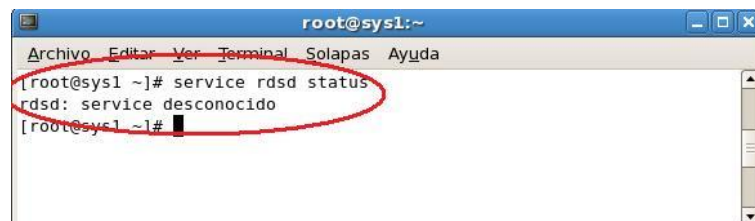
## Desinstalación de la Puerta de Enlace

Se detalla la forma como se puede desinstalar la Puerta de Enlace de una manera sencilla.

- Ubicarse en el directorio donde se encuentra el desinstalador, dar doble click en `uninstall.sh` y posteriormente dar click en ejecutar.



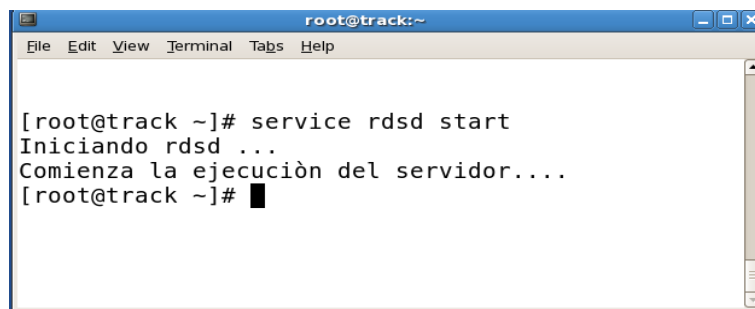
- Listo verificamos que todo este correcto como muestra la imagen.



## Cómo trabajar con RDS GWServer

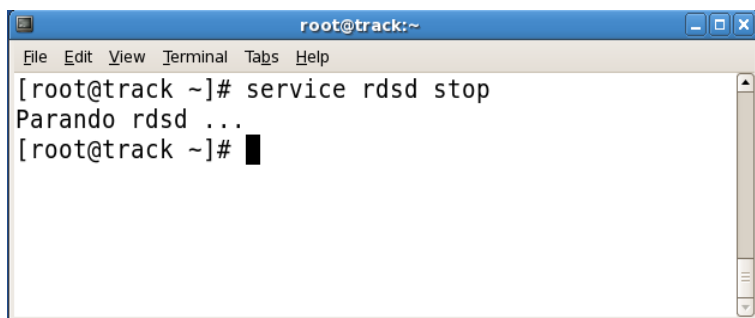
En ésta sección se detallan las opciones del demonio rdsd para ejecutarlo, detenerlo, reiniciarlo y para conocer en qué estado se encuentra el mismo.

***service rdsd start*** Opción para levantar el servicio, con lo que la puerta de enlace comenzará su correcto funcionamiento.



```
root@track:~  
File Edit View Terminal Tabs Help  
  
[root@track ~]# service rdsd start  
Iniciando rdsd ...  
Comienza la ejecución del servidor....  
[root@track ~]# █
```

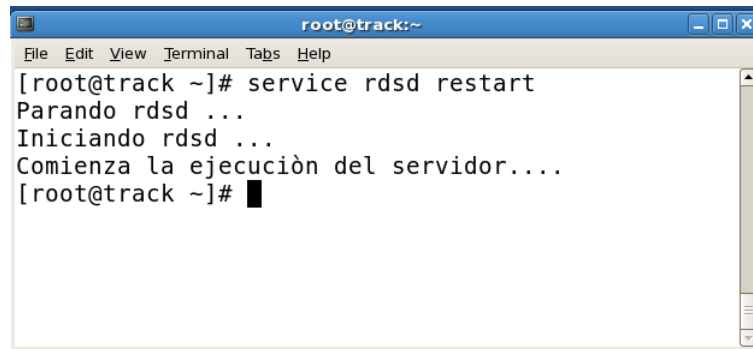
***service rdsd stop*** Opción para detener el servicio.



```
root@track:~  
File Edit View Terminal Tabs Help  
  
[root@track ~]# service rdsd stop  
Parando rdsd ...  
[root@track ~]# █
```

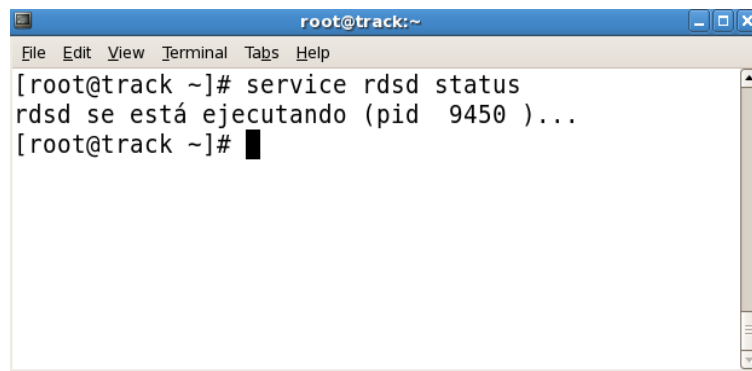
***service rdsd restart*** Opción que se debe ejecutar cuando necesitamos que el servicio tome algunos valores que hayamos modificado en el archivo de configuración rdsd.conf previamente instalado.





```
root@track:~  
File Edit View Terminal Tabs Help  
[root@track ~]# service rdsd restart  
Parando rdsd ...  
Iniciando rdsd ...  
Comienza la ejecución del servidor....  
[root@track ~]# █
```

*service rdsd status* Aquí se puede observar en el estado (en ejecución o detenido) en el que se encuentre la puerta de enlace.



```
root@track:~  
File Edit View Terminal Tabs Help  
[root@track ~]# service rdsd status  
rdsd se está ejecutando (pid 9450 )...  
[root@track ~]# █
```

*service rdsd version* Aquí se puede observar que versión de puerta de enlace se encuentra instalada en la máquina servidor.

## **Análisis de eventos del RDS GWServer**

Cuando la Puerta de Enlace es ejecutada, su función es recibir los datagramas emitidos por los GPSs, los mismos que son almacenados en el directorio `/var/log/rdsd/rds.log`, si así está definido en el archivo de configuración `/etc/rds.conf`, así como en el archivo `/var/log/rds/rds.sql` que se almacenan aquellas consultas SQL que por algún motivo no se puedan ejecutar correctamente.

A continuación se detalla el formato de los archivos donde se almacenan los eventos generados por rdsd

- `/var/log/rdsd/rds.log` el mismo que va a contener la fecha/hora, dirección ip:puerto del origen del datagrama, longitud y la trama convertida a hexadecimal.
- `/var/log/rdsd/rds.sql` va a contener la fecha/hora, el error por el se produjo y la sentencia SQL a insertarse en la base de datos si estos por algún inconveniente no lograrse su objetivo, esto para su posterior inserción.

## Interfaz de Usuario

En el diseño de la interface del usuario se siguieron principios de diseño con el objetivo de que sea amigable y de fácil uso para los que utilicen el sistema.

A continuación se detallan las opciones que posee la Interfaz de Usuario para su manipulación.

### Monitoreo

- Muestra datos de todos los vehículos que cuenten con el servicio de rastreo.
- Muestra los vehículos, agrupados por el nombre del propietario.
- Tiene la opción de refrescar la pantalla cada cierto tiempo.

### Forma Específica (de un automóvil)

- Muestra los datos del vehículo.
- Muestra la información geoespacial reportada por el GPS.
- Muestra la información registrada del propietario del vehículo.

Se cuenta con una interfaz de usuario la misma que va a tener una presentación similar a las pantallas que se muestran a continuación:

En la figura 1 se presenta la información de los vehículos (id\_vehículo, alias, fecha/hora, odómetro) que han contratado el servicio de rastreo en la empresa RDS.

Vehículo	Alias	Fecha Hora	Odómetro	Comando
1224	DISCO 02 TEA-349 PP	2011-10-27 16:14:29	199483	.....
1268	DISCO 03 TAE-761	2011-10-27 16:11:49	34108	.....
1203	DISCO 04 TEA-536	2011-10-27 16:13:52	66388	.....
1272	DISCO 05 TEA-538	2011-10-27 16:14:58	147425	.....
1259	DISCO 06 NEA-159	2011-10-27 16:12:14	171282	.....
1216	DISCO 07 TEA-770	2011-10-27 16:13:48	38022	.....
1261	DISCO 08 NEA-158	2011-10-27 15:22:26	161570	.....
1211	DISCO 09 TEA-557	2011-10-27 16:13:53	82308	.....
1202	DISCO 10 TEA-607	2011-10-27 16:14:51	68901	.....
1215	DISCO 11 TEA-771	2011-10-27 16:12:17	42571	.....
1208	DISCO 12 TEA-605	2011-10-27 16:09:47	0	.....
1213	DISCO 13 TEI-1008	2011-10-27 16:11:52	29423	.....
1260	DISCO 14 NEA-154	2011-10-27 16:11:29	244982	.....
1227	DISCO 15 TEA-391	2011-10-27 16:15:37	98502	.....
1225	DISCO 16 TEA-407	2011-10-27 16:13:25	88335	.....
1281	DISCO 17 TEA-440	2011-10-27 15:51:15	0	.....
1205	DISCO 18 TEA-538	2011-10-27 16:11:08	82188	.....
1249	DISCO 19 TEA-539	2011-10-27 16:15:59	8122	.....
1206	DISCO 20 TEI-1004	2011-10-27 16:12:11	24277	.....
1256	DISCO 21 TEA-560	2011-10-27 16:13:45	196257	.....
1246	DISCO 22 TEA-561	2011-10-27 16:13:40	101899	.....

Figura 1 Información General

En la figura 2 se muestra que se puede enviar comandos a los vehículos a través de los GPSs para que estos realicen ciertas actividades como bloqueo, desbloqueo del motor del automóvil.

Vehículo	Alias	Fecha Hora	Odómetro	Comando
1224	DISCO 02 TEA-349 PP	2011-10-27 16:14:29	199483	<ul style="list-style-type: none"> <li>Pantallas</li> <li>Apertura</li> <li>Bloqueo</li> <li>Motor</li> <li>Desbloqueo</li> <li>Bloqueo</li> <li>Otras</li> <li>Reportarse</li> </ul>
1268	DISCO 03 TAE-761	2011-10-27 16:11:49	34108	.....
1203	DISCO 04 TEA-536	2011-10-27 16:13:52	66388	.....
1272	DISCO 05 TEA-538	2011-10-27 16:14:58	147425	.....
1259	DISCO 06 NEA-159	2011-10-27 16:12:14	171282	.....
1216	DISCO 07 TEA-770	2011-10-27 16:13:48	38022	.....
1261	DISCO 08 NEA-158	2011-10-27 15:22:26	161570	.....
1211	DISCO 09 TEA-557	2011-10-27 16:13:53	82308	.....
1202	DISCO 10 TEA-607	2011-10-27 16:14:51	68901	.....
1215	DISCO 11 TEA-771	2011-10-27 16:12:17	42571	.....
1208	DISCO 12 TEA-605	2011-10-27 16:09:47	0	.....
1213	DISCO 13 TEI-1008	2011-10-27 16:11:52	29423	.....
1260	DISCO 14 NEA-154	2011-10-27 16:11:29	244982	.....
1227	DISCO 15 TEA-391	2011-10-27 16:15:37	98502	.....
1225	DISCO 16 TEA-407	2011-10-27 16:13:25	88335	.....
1281	DISCO 17 TEA-440	2011-10-27 15:51:15	0	.....
1205	DISCO 18 TEA-538	2011-10-27 16:11:08	82188	.....
1249	DISCO 19 TEA-539	2011-10-27 16:15:59	8122	.....
1206	DISCO 20 TEI-1004	2011-10-27 16:12:11	24277	.....
1256	DISCO 21 TEA-560	2011-10-27 16:13:45	196257	.....
1246	DISCO 22 TEA-561	2011-10-27 16:13:40	101899	.....

Figura 2 Envío de comandos

En la figura 3 se muestra los eventos por los que se pueden agrupar a los vehículos

	Hora	Odometro	Comando
Vehículos con [Código] Parado reportado	16:14:26	10043	
Vehículos con [Código] Pánico reportado	16:11:49	3409	
Vehículos con [Código] Pánico reportado	16:13:52	6638	
Vehículos con [Código] Encendido reportado	16:14:58	14705	
Vehículos con [Código] Apagado reportado	16:13:14	17182	
Vehículos con [Código] Desconexión Batena reportado	16:13:48	3813	
Vehículos con [Código] Puerta Cerrada reportado	13:23:26	16130	
Vehículos con [Código] Puerta Abierta reportado	16:13:53	8239	
Vehículos con [Código] Exceso de Velocidad reportado	16:14:51	6961	
Vehículos con [Código] Pánico reportado	16:13:17	4251	
Vehículos con [Código] Alarma Disparada reportado	20:11:00	16:09:47	0
Vehículos con [Código] Alarma Activada reportado	20:11:00	16:11:52	2943
Vehículos con [Código] Alarma Desactivada reportado	20:11:00	16:11:29	29982
Vehículos con [Código] Alarma Activada reportado	20:11:00	16:15:57	9852
Vehículos con [Código] Alarma Desactivada reportado	20:11:00	16:13:25	8831
Vehículos con [Código] Alarma Activada reportado	20:11:00	15:11:15	0
Vehículos con [Código] Alarma Desactivada reportado	20:11:00	16:11:08	8188
Vehículos con [Código] Alarma Activada reportado	20:11:00	16:15:59	8122
Vehículos con [Código] Alarma Desactivada reportado	20:11:00	16:12:11	24277
Vehículos con [Código] Alarma Activada reportado	20:11:00	16:13:45	396257
Vehículos con [Código] Alarma Desactivada reportado	20:11:00	16:13:40	501899

Figura 3 Eventos producidos