PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

SEDE AMBATO

ESCUELA DE INGENIERÍA DE SISTEMAS

DISERTACIÓN DE GRADO PREVIA LA OBTENCIÓN DEL TÍTULO

DE INGENIERO DE SISTEMAS

"DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA DE GESTIÓN ADMINISTRATIVA, QUE SIMPLIFIQUE Y ESTANDARICE LOS PROCESOS DE DESPACHO DE DOCUMENTOS EN LA DIRECCION ADMINISTRATIVA DE LA PUCESA"

GARCÉS LLERENA BLADIMIR PATRICIO

DIRECTOR DE LA DISERTACIÓN

ING. MSC. PATRICIO MEDINA

AMBATO, 2006

28-1-07

CERTIFICADO

Por la presente me permito informar que la Disertación: **"Desarrollo e Implementación de un Sistema De Gestión Administrativa, que Simplifique y Estandarice los Procesos De Despacho De Documentos en la Dirección Administrativa de la Pucesa"**, elaborada por el señor: **Bladimir Patricio Garcés Llerena**, se encuentra concluida y lista para su presentación.

Particular que pongo en su conocimiento para los fines pertinentes.

Atentamente,

ING. MSC. Patricio Medina

DIRECTOR DE DISERTACION

AUTORÍA DE LA TESIS

Yo, GARCES LLERENA BLADIMIR PATRICIO, declaro que el presente trabajo de investigación que abarca al diseño, rutinas y procedimientos de programación es de propiedad absolutamente original y personal. Por consiguiente declaro que el contenido del presente proyecto es de exclusiva responsabilidad legal y académica de mi persona.

Garcés Llerena Bladimir Patricio
C.I.: 180353795-8

DEDICATORIA

El presente trabajo va dedicado a mis queridos padres, Edgar Garcés y Hortensia Llerena quienes con sus consejos, apoyo moral y económico me guiaron durante todos mis años de estudio, a mi hijo Jair razón de mis esfuerzos, a mi esposa Anita por su amor y comprensión. Sin ellos no hubiese podido lograr que mis sueños de ser un PROFESIONAL se hagan realidad.

AGRADECIMIENTO

Expreso mis más sinceros agradecimientos a mis queridos maestros, quienes sembraron la semilla del conocimiento día tras día, mención especial a los ingenieros Diego Santacruz Administrador del Centro de Computo PUCESA, Jorge Chávez y Patricio Medina Director de Tesis, quienes supieron apoyarme con sus conocimientos para llegar a ser un buen profesional.

ÍNDICE GENERAL

INDICE DE ILUSTRACIONES.

INDICES DE TABLAS

INTRODUCCION

Con el fin de alcanzar el desarrollo y fortalecimiento de la PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR SEDE AMBATO y mejorar la labor profesional de sus actividades, buscando favorecer el mejor desempeño, el sistema automático es de mucha utilidad e importancia.

En la actualidad la Dirección Administrativa de la PUCESA realiza los procesos de administración de documentos de forma manual, el cual es un proceso agitado para el responsable de esta dependencia. Al momento de acceder a la información de un docente o empleado, el encargado tarda demasiado tiempo, debido a que se deben revisar varias carpetas de información.

La presente disertación consiste en el Diseño e Implementación de un Sistema de Gestión Administrativa que Simplifique y Estandarice los Procesos de Despacho de Documentos en la PUCESA.

Con la ayuda del sistema automático la PUCESA tendrá un mejor control sobre todos quienes laboran en dicha dependencia, además de mejorar el rendimiento en el despacho de los diferentes documentos que posee la Dirección Administrativa.

1. CAPITULO I. Proyecto de la Investigación.

1.1. Planteamiento del Problema .

1.1.1. Problema.

Carencia de un Sistema de Gestión Administrativa que Simplifique y Estandarice los Procesos de Despacho de Documentos, lo cual repercute en la rapidez y eficacia de la Dirección Administrativa de la PUCESA 2005- 2006.

1.1.2. Problematización.

- Falta de Rapidez al elaborar los diferentes tipos documentos.
- No existe espacio en el disco duro del servidor Sun Enterprise 250 de la PUCESA.
- No se tiene una buena organización en los diferentes tipos de documentación.

1.2. Delimitación.

La elaboración del proyecto planteado se desarrollará exclusivamente para la Dirección Administrativa de la PUCESA en el periodo 2005- 2006. El tiempo estimado para solucionar el presente problema será de nueve meses a partir de la fecha de aprobación del plan.

El sistema contará con una base de datos independiente hecha en el motor de base de datos **MYSQL** que se alojará en el servidor SUN Enterprise 250 de la PUCESA; y se utilizará el lenguaje **PHP** (acrónimo: Hypertext Preprocessor), que es un lenguaje "Open Source" interpretado de alto nivel, especialmente creado para el desarrollo de aplicaciones Web.

Se utilizará como medio de apoyo para el desarrollo del sistema la documentación de la Dirección Administrativa de la PUCESA. El sistema estará diseñado para administrar la documentación de la PUCESA en lo que se refiere a: Contratos, Oficios (personal administrativo y docente), además emitirá

certificados de: Ingresos, Tiempo de trabajo, Subsidios e Ingresos y Tiempo de Trabajo; de una manera rápida.

1.3. Importancia y Justificación .

1.3.1. Importancia.

Con la implementación del sistema la Dirección Administrativa de la PUCESA, será un modelo a considerar por las otras Sedes además para las instituciones públicas o privadas que deseen incorporar a sus dependencias sistemas informáticos.

Para la Dirección Administrativa de la PUCESA el sistema es de suma importancia, por que ayudará al mejoramiento del desempeño de sus funciones, además de agilitar los respectivos procesos que el sistema va a contener en especial para el despacho de documentos.

A través del proyecto se alcanzará la automatización del proceso de despacho de documentos y se obtendrá en tiempo real las respuestas solicitadas al sistema.

1.3.2. Justificación.

Existen hardware y software necesarios para el desarrollo del sistema, además se cuenta con los conocimientos para el desarrollo de la aplicación; los costos son accesibles para la creación de dicho sistema.

1.4. Objetivos.

1.4.1. Objetivo General.

Desarrollar e Implementar un Sistema de Gestión Administrativa, que Simplifique y Estandarice los Procesos de Despacho de Documentos, lo cual repercute en la rapidez y eficacia del despacho de documentos en la Dirección Administrativa de la PUCESA

1.4.2. Objetivos Específicos.

- Lograr mayor rapidez para las emisiones de los documentos.
- Adquirir un disco duro nuevo para que pueda alojar la información necesaria para el sistema.
- Organizar los datos e información que se procesa en la Dirección Administrativa de la PUCESA.

1.5. Hipótesis.

Con el Sistema de Gestión Administrativa se obtendrá mayor rapidez en el acceso a la información de la documentación en la Dirección Administrativa de la PUCESA.

1.6. Aspectos Metodológicos.

1.6.1. Fundamentos Teóricos.

Este proyecto de disertación utiliza los siguientes paradigmas:

- Racionalista, se cuenta con los conocimientos e instrumentos necesarios para que la aplicación se pueda llevar a la realidad.
- Empirista, con la ayuda de este paradigma se llevará acabo la aplicación por medio de la práctica para que el proyecto sea un éxito.

1.6.2. Métodos de Investigación.

En el proceso de investigación se utilizará el método Analítico, por que se comenzará desde cero, y de acuerdo con el análisis que se llevará acabo se ira avanzando con el proyecto para lograr concluir con nuestro propósito.

Entre los diferentes tipos de investigación será utilizado el tipo bibliográfico ya que por medio de la investigación y mediante consultas se va a

alcanzar el objetivo deseado que es el de agilizar los procesos de despacho de documentos.

También se utilizara el tipo de investigación de campo por que va a realizar entrevistas con la persona que va a utilizar el sistema.

La tecnología IDE de los discos duros actuales ha sido mejorada y se le conoce como Enhaced IDE (EIDE), permitiendo mayor transferencia de datos en menor tiempo. Algunos fabricantes la denominan Fast ATA-2. Estos discos duros son más rápidos y su capacidad de almacenamiento supera un gigabyte.

Un megabyte (MB) corresponde aproximadamente a un millón de caracteres y un gigabyte (GB) tiene alrededor de mil megabytes. Los nuevos equipos traen como norma discos duros de 100 gigabytes.

SATA. Estos dispositivos son los primeros de la industria que incluyen un conjunto completo de funciones de administración e integración, diseñadas específicamente para aplicaciones y cargas de trabajo casi en línea, serie que fue fabricada en respuesta a la creciente demanda de almacenamiento de este tipo, alta capacidad y costo efectivo en el sector empresarial, con el cual se cierra la brecha de precio y rendimiento que existe entre el almacenamiento en línea y fuera de ella.

Las exigencias cada vez mayores de almacenar y administrar los datos fijos de las pesadas cuentas de correo electrónico, el aumento de las aplicaciones de imágenes digitales y los requisitos sobre las regulaciones, motivan a los gerentes de tecnología de la información a tomar en cuenta soluciones de almacenamiento casi en línea de bajo costo basadas en discos duros, como la Serie NL35 de los discos SATA.

SCSI. Los discos y controladores SCSI fueron desarrollados para servidores, 24x7, en sistemas de alta disponibilidad. Soportan largas longitudes de cable para poder conectar dispositivos externos.

La tecnología SCSI también incorpora un número de alto nivel para poder dar rendimiento y escalabilidad en servidores multiusuario, multithreading (multi-hilo) para maximizar la transferencia de datos entre la memoria y el controlador y maximizar la eficiencia de las cabezas en su desplazamiento en el disco. (Son controladores inteligentes con capacidad para mezclar y cambiar la secuencia de instrucciones recibidas de la máquina anfitrión, al objeto de mejorar el rendimiento garantizando la integridad de datos).

Ventajas de los discos SCSI:

- Elimina cualquier limitación que el PC-Bios imponga a las unidades de disco.
- El direccionamiento lógico elimina la sobrecarga que el host podría tener en manejar los aspectos físicos del dispositivo como la tabla de pistas dañadas. El controlador SCSI lo maneja.

2.1.2. Montaje de Volúmenes.

Todos los ficheros accesibles en un sistema Unix están dispuestos en un gran árbol, la jerarquía de ficheros, con la raíz en /. Estos ficheros pueden estar distribuidos sobre varios dispositivos. La orden *mount* sirve para pegar el sistema de ficheros encontrado en algún dispositivo al gran árbol de ficheros. De modo análogo pero al revés, la orden *umount* lo despegará de él de nuevo.

La forma más normal de la orden mount es:

mount -t tipo dispositivo dir

Esto le dice al núcleo que anexe el sistema de ficheros que encuentre en el dispositivo (que es del tipo *tipo*) al directorio dir. Los contenidos anteriores (si había), así como el propietario y permisos de dir se vuelven invisibles (están ocultos, tapados), y mientras este sistema de ficheros permanezca montado, el nombre de camino dir, también llamado punto de montaje, se refiere a la raíz del sistema de ficheros en dispositivo.

Tres formas de llamada no montan realmente nada:

mount -h

muestra un mensaje de ayuda;

mount -V

muestra una cadena de caracteres relativa a la versión; y un solo

mount [-t tipo]

Lista todos los sistemas de ficheros montados (del tipo tipo). El sistema de ficheros ***proc*** no está asociado a ningún dispositivo o fichero especial, y cuando se

monte, se puede emplear una palabra arbitraria, como proc, en vez de una especificación de dispositivo. (La elección personal none es menos afortunada: el mensaje de error `none busy' (nadie [está] ocupado) de umount puede confundir.)

La mayoría de dispositivos se indican mediante un nombre de fichero (de un dispositivo especial de bloques), como /dev/sda1, pero hay otras posibilidades. Por ejemplo, en el caso de un montaje por NFS, dispositivo puede ser algo como knuth.cwi.nl:/dir.

El fichero /etc/fstab puede contener renglones que describan qué dispositivos se montan usualmente dónde, empleando cuáles opciones. Este fichero se emplea de tres formas:

(i) La orden mount -a [-t tipo] (Usualmente dado en un guión de arranque) hace que todos los sistemas de ficheros mencionados en fstab (del tipo adecuado) se monten como se indique, excepto aquéllos cuya línea contenga la palabra clave noauto. Añadir la opción -F hará que mount se bifurque, de forma que los sistemas de ficheros se monten simultáneamente, en paralelo.

(ii) Cuando se monte un sistema de ficheros mencionado en fstab, basta con dar sólo el dispositivo o el punto de montaje.

(iii) Normalmente, sólo el súper usuario puede montar sistemas de ficheros. Sin embargo, cuando fstab contiene la opción user en una línea, entonces cualquiera puede montar el distema de ficheros correspondiente.

Así, dada la línea

/dev/cdrom /cdrom iso9660 ro,user,noauto,unhide
cualquier usuario puede montar el sistema de ficheros de tipo iso9660 encontrado en su CD-ROM mediante la orden

mount /dev/cdrom
mount /cdrom

Los programas ***mount*** y ***umount*** mantienen una lista de los sistemas de ficheros montados actualmente en el fichero /etc/mtab. Si no se dan argumentos a ***mount***, se muestra esta lista. Cuando el sistema de ficheros ***proc*** esté montado (digamos en /proc), los ficheros /etc/mtab y /proc/mounts tienen contenidos muy similares. El primero tiene algo más de información, como las opciones de montaje empleadas, pero no está necesariamente al día (cf. la opción -n más abajo). Es posible reemplazar /etc/mtab por un enlace simbólico a /proc/mounts, pero de esta forma se perderá algo de información, y en particular trabajar con el dispositivo de bucle será menos conveniente.

2.1.3. Sun Volumen Manager.

La función de creación automática de volúmenes (top-down volume creation) de Solaris Volume Manager incluye un nuevo comando que ayuda a los administradores de sistemas a crear configuraciones de volúmenes basadas en criterios de calidad de servicio. En lugar de particionar los discos, crear los segmentos lógicos y ensamblar los duplicados, el comando *metassist* se encarga de manejar los detalles y proporciona volúmenes lógicos funcionales basados en los criterios especificados en la línea de comandos o en los archivos de configuración de referencia.

2.2. Gestión Administrativa.

2.2.1. Funciones de la Gestión Administrativa.

Son funciones de la gestión administrativa las que continuación a se detallan:

- Programar, organizar, coordinar, ejecutar, dirigir, controlar y evaluar las actividades de los Departamentos y secciones a su cargo y que tienen relación con dotación de bienes y servicios el apoyo administrativo y servicios generales, con los recursos materiales, tecnológicos y humanos
- Proporcionar un oportuno servicio de apoyo que permita un funcionamiento apropiado de las dependencias.

- Supervisar y controlar la contratación y administración de los recursos humanos.
- Formular el Plan Anual de Adquisiciones, conforme a las políticas directrices establecidas por la entidad, las necesidades Institucionales.
- Controlar y supervisar que todos los servicios administrativos y de mantenimiento generales, se cumplan en forma eficiente y oportuna.

2.3. Ingeniería de Software.

La Ingeniería de software es una disciplina o área de la Informática, que ofrece métodos y técnicas para desarrollar y mantener software de calidad que resuelven problemas de todo tipo a bajo costo.

La ingeniería del software trata con áreas muy diversas de la Informática y de las Ciencias de la Computación, tales como construcción de compiladores, sistemas operativos o desarrollos de Intranet/Internet, abordando todas las fases del ciclo de vida del desarrollo de cualquier tipo de sistemas de información y aplicables a una infinidad de áreas tales como: negocios, investigación científica, medicina, producción, logística, banca, control de tráfico, meteorología, el mundo del derecho, la red de redes Internet, redes Intranet y Extranet, etc.

2.3.1. Definiciones.

[IEEE, 1993]. La aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación (funcionamiento) y mantenimiento del software: es decir, la aplicación de Ingeniería al software.

[Zelkovits, 1978]. Ingeniería del Software es el estudio de los principios y metodologías para desarrollo y mantenimiento de sistemas de software.

[Bauer, 1972]. Ingeniería del Software trata del establecimiento de los principios y métodos de la Ingeniería a fin de obtener software de modo rentable que sea fiable y trabaje en máquinas reales.

2.3.2. Objetivos.

- Mejorar la calidad de los productos de software.
- Definir una disciplina que garantice la producción y el mantenimiento de los productos software desarrollados en el plazo fijado y dentro del costo estimado.
- Facilitar el control del proceso de desarrollo de software.
- Suministrar a los desarrolladores las bases para construir software de alta calidad en una forma eficiente.

2.3.3. Software.

2.3.3.1. Definición

El software son las instrucciones electrónicas que van a indicar al ordenador que es lo que tiene que hacer. También se puede decir que son los programas usados para dirigir las funciones de un sistema de computación.

2.3.3.2. Características del Software.

Para poder comprender lo que es el software (y consecuentemente la Ingeniería del Software), es importante examinar las características del software que lo diferencian de otras cosas que los hombres pueden construir.

- El software se desarrolla, no se fabrica en un sentido clásico.
- El software no se estropea.
- La mayoría del software se construye a medida, en vez de ensamblar componentes existentes.

2.3.4. Ciclo de Vida.

El ciclo de vida para desarrollo de sistemas es el conjunto de actividades que los analistas, diseñadores y usuarios realizan para desarrollar e implantar un sistema de información.

Modelo de Ciclo de Vida Cascada (Bennington 1956): esta basado en el ciclo convencional de una ingeniería, el paradigma del ciclo de vida abarca las siguientes actividades:

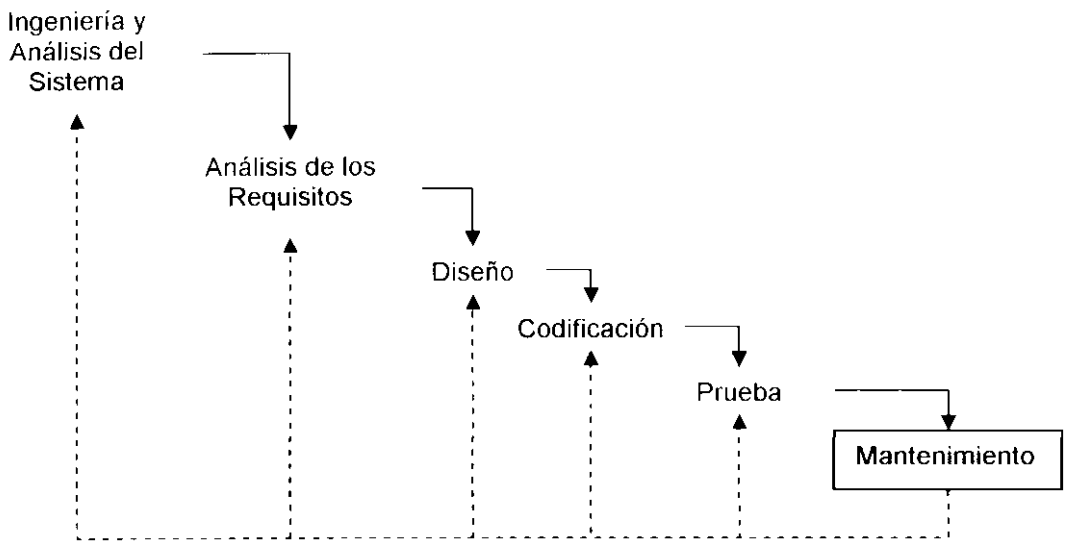


Ilustración 1. Modelo de Ciclo de Vida Cascada (Bennington 1956)

Ingeniería y Análisis del Sistema. Debido a que el software es siempre parte de un sistema mayor el trabajo comienza estableciendo los requisitos de todos los elementos del sistema y luego asignando algún subconjunto de estos requisitos al software.

Análisis de los requisitos del software. El proceso de recopilación de los requisitos se centra e intensifica especialmente en el software. El ingeniero de software (Analista) debe comprender el ámbito de la información del software, así como la función, el rendimiento y las interfaces requeridas.

Diseño. El diseño del software se enfoca en cuatro atributos distintos del programa: la estructura de los datos, la arquitectura del software, el detalle procedimental y la caracterización de la interfaz. El proceso de diseño traduce los requisitos en una representación del software con la calidad requerida antes de que comience la codificación.

2.4. Base de Datos.

2.4.1. Definición.

Base de Datos es un conjunto exhaustivo no redundante de datos estructurados organizados independientemente de su utilización y su implementación en máquina accesibles en tiempo real y compatibles con usuarios concurrentes con necesidad de información diferente y no predicable en tiempo.

2.4.2. Ventajas de las Bases de Datos.

- **Independencia de datos y tratamiento.** Cambio en datos no implica cambio en programas y viceversa (Menor coste de mantenimiento).
- **Coherencia de resultados.** Reduce redundancia. Acciones lógicamente únicas y se evita inconsistencia.
- **Mejora en la disponibilidad de datos.** No hay dueño de datos (La base de datos no es pública), ni aplicaciones ni usuarios. Guardamos descripción
- es (Ideas de catálogos almacenados).

2.4.3. Modelos de Datos.

Es una colección de herramientas conceptuales para describir los datos, las relaciones que existen entre ellos, semántica asociada a los datos y restricciones de consistencia.

2.4.4. Modelos Lógicos Basados en Objetos.

Se usan para describir datos en los niveles conceptual y de visión, es decir, con este modelo representamos los datos de tal forma como nosotros los captamos en el mundo real, tienen una capacidad de estructuración bastante flexible y permiten especificar restricciones de datos explícitamente. Existen diferentes modelos de este tipo, pero el más utilizado por su sencillez y eficiencia es el modelo Entidad-Relación.

2.4.5. Modelo Entidad Relación.

Denominado por sus siglas como: E-R; este modelo representa a la realidad a través de **entidades**, que son objetos que existen y que se distinguen de otros por sus características, por ejemplo: un alumno se distingue de otro por sus características particulares como lo es el nombre, o el número de control asignado al entrar a una institución educativa, así mismo, un empleado, una materia, etc.

Las características de las entidades en base de datos se llaman **atributos**, por ejemplo el nombre, dirección teléfono, grado, grupo, etc. Son atributos de la entidad alumno; clave, número de seguro social, departamento, etc., son atributos de la entidad empleado. A su vez una entidad se puede asociar o relacionar con más entidades a través de relaciones.

Pero para entender mejor esto, veamos un ejemplo:

Consideremos una empresa que requiere controlar a los vendedores y las ventas que ellos realizan; de este problema determinamos que los objetos o entidades principales a estudiar son el empleado (vendedor) y el artículo (que es el producto en venta), y las características que los identifican son:

Empleado:	Artículo:
Nombre	Descripción
Puesto	Costo
Salario	Clave

La relación entre ambas entidades la podemos establecer como Venta. Ahora nos falta describir como se representa un modelo E-R gráficamente, la representación es muy sencilla, se emplean símbolos, los cuales son:

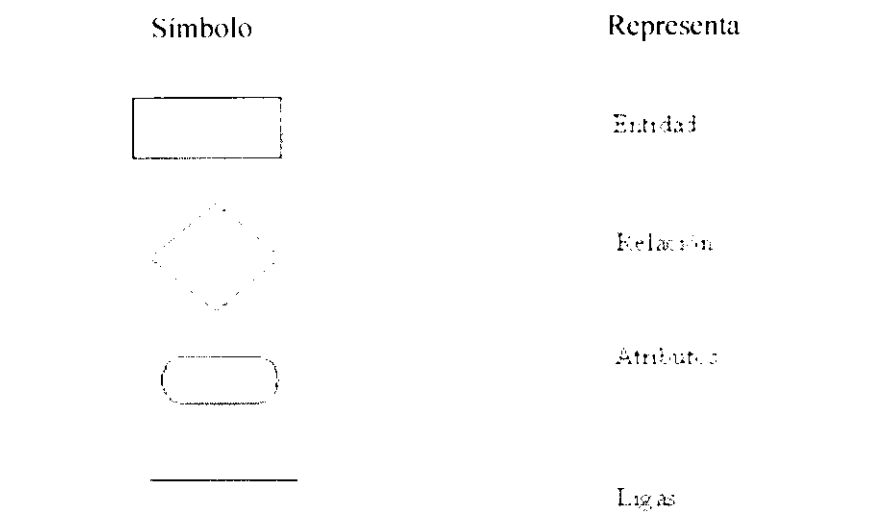


Ilustración 2. Símbolos para Representar un Modelo Entidad Relación

Así nuestro ejemplo anterior quedaría representado de la siguiente forma:

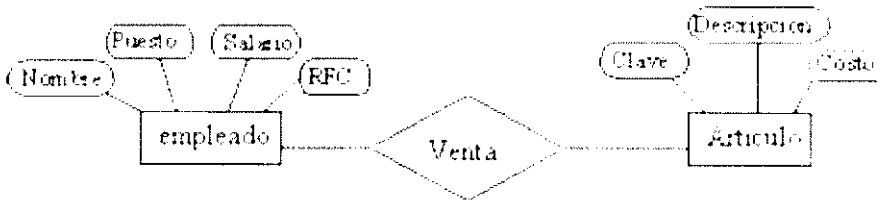


Ilustración 3. Ejemplo de Diagrama Entidad Relación

2.4.6. Modelo Lógico Basados en Registros.

Se utilizan para describir datos en los niveles conceptual y físico. Estos modelos utilizan registros e instancias para representar la realidad, así como las relaciones que existen entre estos registros (ligas) o apuntadores. A diferencia de los modelos de datos basados en objetos, se usan para especificar la estructura lógica global de la base de datos y para proporcionar una descripción a nivel más alto de la implementación.

Los tres modelos de datos más ampliamente aceptados son:

- Modelo Relacional
- Modelo de Red
- Modelo Jerárquico

2.4.6.1. Modelo Relacional.

En este modelo se representan los datos y las relaciones entre estos, a través de una colección de tablas, en las cuales los renglones (tuplas) equivalen a cada uno de los registros que contendrá la base de datos y las columnas corresponden a las características (atributos) de cada registro localizado en la tupla; considerando nuestro ejemplo del empleado y el artículo:

Tabla del empleado

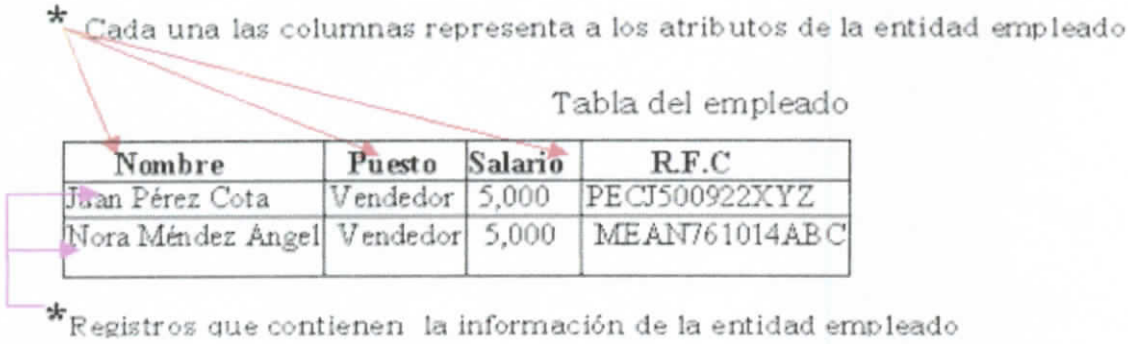


Ilustración 4. Ejemplo de Modelo Relacional de una Tabla

Tabla artículo

Clave	Descripción	Costo
C001	Colcha matrimonial	200

Ilustración 5. Ejemplo de Modelo Relacional Tabla Artículo

Ahora nace la interrogante ¿cómo se representan las relaciones entre las entidades en este modelo?.

Existen dos formas de representarla; pero para ello necesitamos definir que es una llave primaria: Es un atributo el cual definimos como atributo principal, es una forma única de identificar a una entidad. Por ejemplo, el RFC de un empleado se distingue de otro por que los RFC no pueden ser iguales.

Ahora si, las formas de representar las relaciones en este modelo son:

- Haciendo una tabla que contenga cada una de las llaves primarias de las entidades involucradas en la relación. Tomando en cuenta que la llave

primaria del empleado es su RFC, y la llave primaria del artículo es la Clave.

La relación de nuestro modelo resulta

RFC	Clave
PECJ500922XYZ	C001
MEAN761014ABC	B300

Ilustración 6. Ejemplo de la Relación de Tablas

- Incluyendo en alguna de las tablas de las entidades involucradas, la llave de la otra tabla.

Incrustamos la llave primaria del artículo en la tabla del empleado

Nombre	Puesto	Salario	R.F.C	Clave
Juan Perez Cota	Vendedor	5,000	PECJ500922XYZ	C001
Nora Méndez Angel	Vendedor	5,000	MEAN761014ABC	B300

Ilustración 7. Ejemplo de Clave Foránea

2.4.6.2. Modelo Red.

Este modelo representa los datos mediante colecciones de registros y sus relaciones se representan por medio de ligas o enlaces, los cuales pueden verse como punteros. Los registros se organizan en un conjunto de gráficas arbitrarias.

Ejemplo:

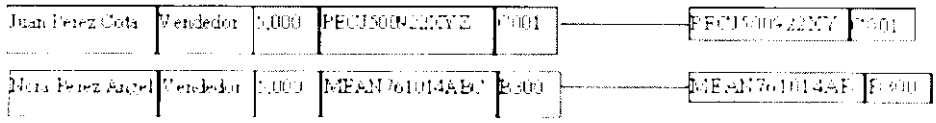


Ilustración 8. Ejemplo de Modelo Red

2.4.6.3. Modelo Jerárquico.

Es similar al modelo de red en cuanto a las relaciones y datos, ya que estos se representan por medio de registros y sus ligas. La diferencia radica en que están organizados por conjuntos de árboles en lugar de gráficas arbitrarias.

Ilustración 9. Ejemplo Modelo Jerárquico

2.5. Motor de Base de Datos MySQL.

2.5.1. Introducción.

MySQL es el más popular gestor de bases de datos relacional, es desarrollado y distribuido por MySQL AB que es una compañía comercial que desarrolla y provee servicios alrededor de MySQL.

Una vez instalado, para conectarse al servidor, generalmente se necesita facilitar un nombre de usuario MySQL cuando se lance el cliente mysql y, lo más probable, también un password. Si el servidor se está ejecutando en una máquina distinta a la que esta conectado, se necesita especificar también un nombre de host. Contacta con el administrador para averiguar qué parámetros de conexión se necesita usar para conectar (es decir, qué host, nombre de usuario y password a usar). Una vez que se conozca los parámetros adecuados, se podrá conectar de la siguiente forma:

```
shell> mysql -h host -u user -p
```

Podemos especificar el ordenador donde está el servidor de base de datos (host) y nuestro nombre de usuario. Los parámetros "-h" y "-u" indican que los parámetros a continuación son, respectivamente, el nombre del host y el nombre de usuario. El parámetro "-p" indica que se debe solicitar una clave de acceso. Cuando se trabaja con aplicaciones de gestión de bases de datos, es muy importante definir otros usuarios además del administrador. MySQL permite asignar distintos privilegios a cada usuario con el propósito de proteger la base de datos.

Para salir de una sesión del cliente de MySQL se usa el comando "Quit".

```
mysql> QUIT
Bye
```

2.5.2. Creación de Base de Datos.

Cada conjunto de relaciones que componen un modelo completo forma una base de datos. Desde el punto de vista de SQL, una base de datos es sólo un conjunto de relaciones (o tablas), y para organizarlas o distinguirlas se accede a

ellas mediante su nombre. A nivel de sistema operativo, cada base de datos se guarda en un directorio diferente.

Para crear una base de datos se usa una sentencia "CREATE DATABASE":

```
mysql> CREATE DATABASE prueba;
Query OK, 1 row affected (0.00 sec)
```

Podemos averiguar cuántas bases de datos existen en nuestro sistema usando la sentencia "SHOW DATABASES":

```
mysql> SHOW DATABASES;
+----------+
| Database |
+----------+
| mysql    |
| prueba   |
| test     |
+----------+
3 rows in set (0.02 sec)
```

Para seleccionar una base de datos se usa el comando "USE":

```
mysql> USE prueba;
Database changed
```

2.5.3. Creación de Tablas.

La sentencia "CREATE TABLE" que sirve para crear tablas. La sintaxis de esta sentencia es muy compleja, ya que existen muchas opciones y tenemos muchas posibilidades diferentes a la hora de crear una tabla.

```
mysql> USE prueba
Database changed
mysql> CREATE TABLE texto ( nombre VARCHAR(20), texto TEXT );
Query OK, 0 rows affected (0.00 sec)
```

Podemos consultar cuántas tablas y qué nombres tienen en una base de datos, usando la sentencia "SHOW TABLES":

Pero tenemos muchas más opciones a la hora de definir columnas. Además del tipo y el nombre, podemos definir valores por defecto, permitir o no que contengan valores nulos, crear una clave primaria, indexar:

sobre alguna característica especial de la columna, y entra en el apartado de documentación de la base de datos.

Claves Foráneas. En MySQL sólo existe soporte para claves foráneas en tablas de tipo InnoDB. Sin embargo, esto no impide usarlas en otros tipos de tablas. La diferencia consiste en que en esas tablas no se verifica si una clave foránea existe realmente en la tabla referenciada, y que no se eliminan filas de una tabla con una definición de clave foránea. Para hacer esto hay que usar tablas InnoDB.

2.5.4. Ingreso de Datos.

La forma más directa de insertar una fila nueva en una tabla es mediante una sentencia ***"INSERT"***. En la forma más simple de esta sentencia debemos indicar la tabla a la que queremos añadir filas, y los valores de cada columna. Las columnas de tipo cadena o fechas deben estar entre comillas sencillas o dobles, para las columnas numéricas esto no es imprescindible, aunque también pueden estar entrecomilladas.

```
mysql> INSERT INTO gente VALUES ('Fulano','1974-04-12');
Query OK, 1 row affected (0.05 sec)
```

Si no necesitamos asignar un valor concreto para alguna columna, podemos asignarle el valor por defecto indicado para esa columna cuando se creó la tabla, usando la palabra ***DEFAULT***.

Otra opción consiste en indicar una lista de columnas para las que se van a suministrar valores. A las columnas que no se nombren en esa lista se les asigna el valor por defecto. Este sistema, además, permite usar cualquier orden en las columnas, con la ventaja, con respecto a la anterior forma, de que no necesitamos conocer el orden de las columnas en la tabla para poder insertar datos.

```
mysql> INSERT INTO ciudad5 (poblacion,nombre) VALUES
    -> (7000000, 'Madrid'),
    -> (9000000, 'Paris'),
    -> (3500000, 'Berlin');
Query OK, 3 rows affected (0.05 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

2.5.5. Actualización de Datos.

Podemos modificar valores de las filas de una tabla usando la sentencia "UPDATE". En su forma más simple, los cambios se aplican a todas las filas, y a las columnas que especifiquemos.

```
UPDATE [LOW PRIORITY] [IGNORE] tbl_name
    SET col_name1=expr1 [, col_name2=expr2 ...]
    [WHERE where_definition]
    [ORDER BY ...]
    [LIMIT row_count]
```

2.5.6. Eliminación de Datos.

Para eliminar filas se usa la sentencia "DELETE".

```
DELETE [LOW PRIORITY] [QUICK] [IGNORE] FROM table_name
       [WHERE where_definition]
       [ORDER BY ...]
       [LIMIT row_count]
```

La forma más simple es no usar ninguna de las cláusulas opcionales. Pero es más frecuente que sólo queramos eliminar ciertas filas que cumplan determinadas condiciones. La forma más normal de hacer esto es usar la cláusula WHERE.

```
mysql> DELETE FROM ciudad5 WHERE clave=2;
Query OK, 1 row affected (0.00 sec)
```

Cuando queremos eliminar todas la filas de una tabla utilizamos la sentencia "TRUNCATE".

2.5.7. Selección de Datos.

Para extraer datos de una base de datos utilizaremos la sentencia "SELECT".

```
SELECT [ALL | DISTINCT | DISTINCTROW]
   expresion_select,...
   FROM referencias_de_tablas
   WHERE [condiciones]
   [GROUP BY {nombre_col | expresion | posicion}
         [ASC | DESC], ... [WITH ROLLUP]]
   [HAVING condiciones]
   [ORDER BY {nombre_col | expresion | posicion}
```

```
        [ASC | DESC] ,...]
    [LIMIT {[desplazamiento,] contador | contador OFFSET
desplazamiento}]
```

La forma más sencilla es la que hemos usado hasta ahora, consiste en pedir todas las columnas y no especificar condiciones.

```
mysql> SELECT * FROM gente;
```

Pero podemos usar una lista de columnas, y de ese modo sólo se mostrarán esas columnas:

```
mysql> SELECT nombre FROM gente;

+----------+
| nombre   |
+----------+
| Pilar    |
| Mariano  |
| Carlos   |
| Isabel   |
+----------+

4 rows in set (0.00 sec)
```

2.6. Programación Web.

2.6.1. Funcionamiento de un Sitio Web.

La arquitectura cliente-servidor hace referencia a múltiples clientes que se conectan a un servidor en forma concurrente. El servidor depende del sitio Web mientras el cliente suele ser un navegador.

En relación a una plataforma cliente-servidor que exige un protocolo que especifique de que manera se comunican o intercambian datos entre cliente y servidor, un sitio Web utiliza el protocolo HTTP que funciona encapsulado sobre el protocolo TCP/IP.

2.6.2. El protocolo HTTP.

HTTP *(HyperText Transfer Protocol)* permite la transferencia de múltiples tipos de información de una forma eficiente, haciéndolo idóneo para una red tan hetereogénea como internet, donde los formatos en que se presenta la información son muy variados e impredecibles (páginas HTML, programas, imágenes, sonidos, videos, películas Flash, futuros formatos, etc.).

La estructura del protocolo HTTP es sencilla, más o menos como la de cualquier arquitectura cliente/servidor: La máquina cliente establece una conexión (normalmente a través del protocolo de transporte TCP/IP, al puerto 80) con la máquina servidora, donde está ejecutándose un software llamado servidor web o servidor HTTP (Apache o Internet Information Server son algunos de ellos).

Una vez establecida la conexión, el cliente (por ejemplo el navegador web) envía tramas de datos que consisten en unas cabeceras especiales y una petición, que es recibida al otro lado de la conexión por el servidor HTTP. Este servidor interpretará la petición del cliente, devolviendo un resultado, que dependerá del tipo de petición recibida. Una vez que la respuesta ha sido enviada, la conexión se pierde. Es por esto que el protocolo HTTP se denomina "sin conexión", al contrario que otros como FTP que mantienen una conexión abierta continuamente. Además de ser "sin conexión", también se suele decir que es un

protocolo "sin estado", ya que no tiene en cuenta peticiones anteriores de un mismo cliente, y considera que cada petición es única e independiente del resto.

El protocolo HTTP: está basado en el modelo cliente-servidor: Un cliente HTTP abre una conexión y realiza su solicitud al servidor, el cual responde generalmente el recurso solicitado y la conexión se cierra.
El formato tanto del mensaje como de la respuesta es:

<Línea inicial>
Header-1: value-1
Header-n: value-n
<Cuerpo del mensaje (Opcional)>

La línea inicial es diferente en las solicitudes y en las respuestas. En las solicitudes van tres campos separados por un espacio en blanco: "Método recurso versión Del Protocolo" Por ejemplo: "GET /path/to/file/index.html HTTP/1.0." La línea inicial de una respuesta tiene tres campos separados por un espacio: "versión Del Protocolo código Respuesta Mensaje". Por ejemplo: "HTTP/1.0 200 OK" o bien "HTTP/1.0 404 Not Found".

Los encabezados están normados en el protocolo, e incluyen, en el caso de una solicitud, información del navegador y eventualmente del usuario cliente; en el caso de una respuesta, información sobre el servidor y sobre el recurso. El cuerpo del mensaje contiene el recurso a transferir o el texto de un error en el caso de una respuesta. En el caso de una solicitud, puede contener parámetros de la llamada archivos enviados al servidor. Actualmente viene remplazando al FTP en la transferencia de archivos.

2.6.3. Ciencias aplicadas para el Desarrollo de Aplicaciones Web.

Para desarrollar aplicaciones y dotar a las páginas web de funcionalidad existe la posibilidad de trabajar tanto en el lado del cliente como en el lado del servidor, las variantes son las siguientes:

Programación en el cliente :

- El browser envía un request.
- El Server envía un response que contiene código que el browser entiende.
- El browser interpreta el código enviado por el Server y realiza una determinada acción.

Programación en el Servidor:

- El browser envía un request.
- El Server ejecuta una aplicación que realiza una determinada acción.
- El Server envía el resultado de dicha aplicación al cliente.
- El browser muestra el resultado recibido del Server.

Esquema mixto: (programación en el cliente y en el servidor).

- El browser envía un request.
- El Server ejecuta una aplicación que realiza una determinada acción.
- El Server envía el resultado de dicha aplicación al cliente conteniendo código a interpretar por el browser.
- El browser interpreta el código enviado por el Server y realiza una determinada acción.

La programación del lado del cliente tiene como principal ventaja que la ejecución de la aplicación se encarga al cliente, con lo cual se evita recargar al servidor de trabajo. El servidor solo envía el código, y es tarea del browser interpretarlo. La gran desventaja de esta metodología es que el código que el Server envía es sensible a que cosas puede o no hacer el browser.

Programar del lado del servidor tiene como gran ventaja que cualquier cosa puede hacerse sin tener en cuenta el tipo de cliente, ya que la aplicación se ejecuta en el servidor que es un ambiente controlado. Una vez ejecutada la aplicación el resultado que se envía al cliente puede estar en un formato normalizado que

cualquier cliente puede mostrar. La desventaja reside en que el Server se sobrecarga de trabajo ya que además de servir páginas es responsable de ejecutar aplicaciones.

Programación en el cliente	Programación en le servidor
HTML	CGI(Cualquier Lenguaje)
CSS	ASP
DHTML	PHP
JavaScript	mod_perl
Java	
VBScript	

Tabla 1. Programación en Cliente-Servidor

Debido a las incompatibilidades existentes y a la posibilidad de que el usuario controle que cosas se ejecutan y cuales no, la programación del lado del cliente no es muy recomendable y debe limitarse a código altamente estándar que pueda interpretarse de cualquier forma en cualquier browser, lo cual obliga a ejecutar la gran mayoría de las aplicaciones y servicios de un sitio Web del lado del servidor.

2.6.4. Programación del Lado del Servidor.

Para el desarrollo de aplicaciones del lado del servidor existen tres grandes metodologías, utilizar el protocolo CGI, utilizar una API provista por el web-server o bien utilizar un módulo del servidor Web.

2.6.4.1. El protocolo CGI.

El protocolo CGI (Common Gateway Interface) es un estándar para comunicar aplicaciones externas con los servidores de información, tales como servidores HTTP o Web. Un documento en HTML que el daemon del Web se trae es estático, es decir, se mantiene constante: un fichero de texto que no cambia. Un programa CGI, por otro lado es ejecutado en tiempo real, así que puede generar información dinámica.

Para ejecutar una aplicación CGI el Web-server en general procede de la siguiente manera:

- Se toma el "request del browser" y los datos que se envían al server por el método post o get se pasan a variables del ambiente.
- El server redirecciona su salida estándar al browser.
- El server crea un proceso que tiene la salida Standard redireccionada.
- El server ejecuta en el proceso creado la aplicación creada.
- Se ejecuta la aplicación.

Cuando la aplicación termina de ejecutarse el proceso muere. Dentro la aplicación usa algún mecanismo para recuperar los datos enviados por el browser desde las variables de ambiente. El protocolo CGI justamente consiste en especificar de qué forma los datos enviados por el browser se conviertan en variables de ambiente, esto es general es transparente para el usuario.

De esta forma pueden realizarse aplicaciones para un Web-site en casi cualquier lenguaje, los lenguajes interpretados rápidamente ganaron terreno ya que tiene un ciclo de desarrollo en tiempo inferior a los lenguajes compilados y son más fáciles de depurar dentro del ambiente CGI.

2.6.4.2. Especificaciones del CGI.

Como un programa CGI es un ejecutable, es equivalente a dejar al mundo ejecutar un programa en el sistema operativo, que no es lo más seguro hacer. Por ello existen una serie de precauciones de seguridad que son necesarias de implementar cuando se usan programas CGI. Probablemente la que afectará al usuario típico del Web, es el hecho de que los programas CGI necesitan residir en un directorio especial, así el servidor sabe que tiene que ejecutarlo, en vez de simplemente mostrarlo por pantalla.

Este directorio está generalmente bajo el control del webmaster, prohibiendo al usuario medio crear programas CGI. Hay otros métodos para

permitir el accesos a scripts CGI, pero depende del webmaster que se de esta posibilidad. Así que deberás contactar con tu webmaster para consultar la factibilidad de permitirte un acceso a los CGI.

Al disponer de una versión del servidor HTTPd NCSA, verás un directorio denominado /cgi-bin. Este es el directorio especial antes mencionado, donde todos los programas CGI residen. Un programa CGI se puede escribir en cualquier lenguaje que permita ser ejecutado en el sistema, como:

C/C++
Fortran
PERL
TCL
Algún Shell de Unix
Visual Basic
AppleScript

2.6.4.3. Directorio CGI-BIN.

Este es un directorio especial, que contiene los scripts, configurado dentro del servidor http. El servidor conoce que este directorio contiene ejecutables que deberán ser ejecutados y su salida deberá ser enviada al navegador del cliente. No se puede simplemente crear un directorio cgi-bin, el administrador del servidor deberá configurarlo para su uso. Si no está configurado, los scripst serán cargados como simples ficheros de texto.

Algunos servidores están configurados de tal manera que los ficheros con una determinada extensión (generalmente .cgi) son reconocidos como scripts y serán ejecutados como si estuvieran en un directorio cgi-bin.
Nota: No deberemos confundirlo con html analizado (generalmente .shtml).

2.6.5. Lenguajes de Programación Web.

2.6.5.1. HTML.

(Hypertext Markup Languaje). Es el lenguaje que permite codificar o preparar documentos de hipertexto, este viene a ser lenguaje común para la construccion de una página Web. Su funcionalidad es representar cualquier clase de información que se encuentre almacenada en una página Web.

HTML no es propiamente un lenguaje de programación como C++, Visual Basic, etc., si no un sistema de etiquetas. HTML no presenta ningún compilador, por lo tanto algún error de sintaxis que se presente éste no lo detectará y se visualizará en la forma que este lo entienda.

2.6.5.2. Javascript.

Se trata de un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado.

Con Javascript podemos crear efectos especiales en las páginas y definir interactividades con el usuario. El navegador del cliente es el encargado de interpretar las instrucciones Javascript y ejecutarlas para realizar estos efectos e interactividades, de modo que el mayor recurso, y tal vez el único, con que cuenta este lenguaje es el propio navegador.

Javascript es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones, estructuras de datos complejas, etc. Además, Javascript pone a disposición del programador todos los elementos que forman la página web, para que éste pueda acceder a ellos y modificarlos dinámicamente.

2.6.5.3. VB Script.

Es un lenguaje de programación de scripts del lado del cliente, pero sólo compatible con Internet Explorer. Está basado en Visual Basic, un popular lenguaje para crear aplicaciones Windows. Tanto su sintaxis como la manera de trabajar están muy inspirados en él. Sin embargo, no todo lo que se puede hacer en Visual Basic lo podremos hacer en Visual Basic Script, pues este último es una versión reducida del primero.

El modo de funcionamiento de Visual Basic Script para construir efectos especiales en páginas Web es muy similar al utilizado en Javascript y los recursos a los que se puede acceder también son los mismos: el navegador.

2.6.5.4. DHTML.

DHTML no es precisamente un lenguaje de programación. Más bien se trata de una nueva capacidad de la que disponen los navegadores modernos, por la cual se puede tener un mayor control sobre la página que antes.

Cualquier página que responde a las actividades del usuario y realiza efectos y funcionalidades se puede englobar dentro del DHTML, pero en este caso nos referimos más a efectos en el navegador por los cuales se pueden mostrar y ocultar elementos de la página, se puede modificar su posición, dimensiones, color, etc.

DHTML nos da más control sobre la página, gracias a que los navegadores modernos incluyen una nueva estructura para visualizar en páginas Web denominada capa. Las capas se pueden ocultar, mostrar, desplazar, etc.

2.6.5.5. CSS.

(Cascading Style Sheets) es una tecnología que nos permite crear páginas web de una manera más exacta. Gracias a las CSS somos mucho más dueños de los resultados finales de la página, pudiendo hacer muchas cosas que no se podía

hacer utilizando solamente HTML, como incluir márgenes, tipos de letra, fondos, colores, etc.

Las Hojas de Estilo en Cascada se escriben dentro del código HTML de la página web, solo en casos avanzados se pueden escribir en un archivo a parte y enlazar la página con ese archivo.

2.6.5.6. Applets de Java.

Los applets de Java están programados en Java y precompilados, es por ello que la manera de trabajar de éstos varía un poco con respecto a los lenguajes de script como Javascript. Los applets son más difíciles de programar que los scripts en Javascript y requerirán unos conocimientos básicos o medios del lenguaje Java.

La principal ventaja de utilizar applets consiste en que son mucho menos dependientes del navegador que los scripts en Javascript, incluso independientes del sistema operativo del ordenador donde se ejecutan. Además, Java es más potente que Javascript, por lo que el número de aplicaciones de los applets podrá ser mayor.

Como desventajas en relación con Javascript cabe señalar que los applets son más lentos de procesar y que tienen espacio muy delimitado en la página donde se ejecutan, es decir, no se mezclan con todos los componentes de la página ni tienen acceso a ellos. Es por ello que con los applets de Java no podremos hacer directamente cosas como abrir ventanas secundarias, controlar Frames, formularios, capas, etc.

2.6.5.7. PERL.

(Practical Extracting and Reporting Languaje). Es un lenguaje de programación muy utilizado para construir aplicaciones CGI para el Web.

Perl es un lenguaje de programación interpretado, al igual que muchos otros lenguajes de Internet como Javascript o ASP. Esto quiere decir que el código de los scripts en Perl no se compila sino que cada vez que se quiere ejecutar se lee el código y se pone en marcha interpretando lo que hay escrito. Además es extensible a partir de otros lenguajes, ya que desde Perl es posible hacer llamadas a subprogramas escritos en otros lenguajes.

Perl está inspirado a partir de lenguajes como C, pero está enfocado a ser más práctico y fácil que estos últimos. Es por ello que un programador que haya trabajado con lenguaje C, ASP, etc. tendrá menos problemas en entenderlo y utilizarlo rápidamente. Una diferencia fundamental de Perl con respecto a los otros lenguajes es que no limita el tamaño de los datos con los que trabaja, el límite lo pone la memoria que en ese momento se encuentre disponible.

2.6.5.8. ASP.

(Active Server Pages). Es la tecnología desarrollada por Microsoft para la creación de páginas dinámicas del servidor. ASP se escribe en la misma página Web, utilizando el lenguaje Visual Basic Script o Javascript de Microsoft.

Con ASP es posible realizar muchos tipos de aplicaciones distintas. Permite acceso a bases de datos, al sistema de archivos del servidor y en general a todos los recursos que tenga el propio servidor. También brinda la posibilidad de comprar componentes ActiveX fabricados por distintas empresas de desarrollo de software que sirven para realizar múltiples usos, como el envío de correo, generar gráficas dinámicamente, etc.

2.6.5.9. PHP.

(Hipertext Preprocesor). Es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma, rápido, con una gran librería de funciones y mucha documentación.

PHP se escribe dentro del código HTML, lo que lo hace realmente fácil de utilizar, al igual que ocurre con ASP de Microsoft, pero con algunas ventajas como su independencia de plataforma, rapidez y seguridad. Cualquiera puede descargar a través de la página principal de PHP www.php.net y de manera gratuita, un módulo que hace que el servidor web comprenda los scripts realizados en este lenguaje. Es independiente de plataforma, puesto que existe un módulo de PHP para casi cualquier servidor web. Esto hace que cualquier sistema pueda ser compatible con el lenguaje y significa una ventaja importante, ya que permite portar el sitio desarrollado en PHP de un sistema a otro sin prácticamente ningún trabajo.

Se detallara de mejor manera este lenguaje en el apartado *2.7. Lenguaje de programación PHP.*

2.6.5.10. JSP.

(Java Server Pages). Con JSP es posible la creación de aplicaciones web que se ejecuten en variados servidores web, de múltiples plataformas, ya que Java es en esencia un lenguaje multiplataforma. Las páginas JSP están compuestas de código HTML/XML mezclado con etiquetas especiales para programar scripts de servidor en sintaxis Java. Por tanto, las JSP podremos escribirlas con nuestro editor HTML/XML habitual.

El motor de las páginas JSP está basado en los servlets de Java (programas en Java destinados a ejecutarse en el servidor.) En JSP es posible crear páginas de manera parecida a como se crean en ASP o PHP. Se generan archivos con extensión .jsp que incluyen, dentro de la estructura de etiquetas HTML, las sentencias Java a ejecutar en el servidor. Antes de que sean funcionales los archivos, el motor JSP lleva a cabo una fase de traducción de esa página en un servlet, implementado en un archivo class (Byte codes de Java). Esta fase de traducción se lleva a cabo habitualmente cuando se recibe la primera solicitud de la página .jsp, aunque existe la opción de precompilar en código para evitar ese tiempo de espera la primera vez que un cliente solicita la página.

2.6.5.11. XML.

XML, con todas las tecnologías relacionadas, representa una manera distinta de hacer las cosas, más avanzada, cuya principal novedad consiste en permitir compartir los datos con los que se trabaja a todos los niveles, por todas las aplicaciones y soportes. Así pues, el XML juega un papel importantísimo en este mundo actual, que tiende a la globalización y la compatibilidad entre los sistemas, ya que es la tecnología que permite compartir la información de una manera segura, fiable y fácil. Además, XML permite al programador y los soportes dedicar sus esfuerzos a las tareas importantes cuando trabaja con los datos, ya que algunas tareas tediosas como la validación de estos o el recorrido de las estructuras corre a cargo del lenguaje y está especificado por el estándar, de modo que el programador no tiene que preocuparse por ello.

Se observa que, XML no está sólo, sino que hay un mundo de tecnologías alrededor de él, de posibilidades, maneras más fáciles e interesantes de trabajar con los datos y, en definitiva, un avance a la hora de tratar la información, que es en realidad el objetivo de la informática en general. XML, o mejor dicho, el mundo XML no es un lenguaje, sino varios lenguajes, no es una sintaxis, sino varias y no es una manera totalmente nueva de trabajar, sino una manera más refinada que permitirá que todas las anteriores se puedan comunicar entre sí sin problemas, ya que los datos cobran sentido.

XML es interesante en el mundo de Internet y el e-bussiness, ya que existen muchos sistemas distintos que tienen que comunicarse entre sí, pero como se ha podido imaginar, interesa por igual a todas las ramas de la informática y el tratamiento de datos, ya que permite muchos avances a la hora de trabajar con ellos.

2.7. Lenguaje de Programación PHP.

2.7.1. Generalidades.

2.7.1.1. Introducción

PHP es un lenguaje open source interpretado de alto nivel diseñado para favorecer el desarrollo de sitios web dinámicos y aplicaciones para web-sites. La distribución más popular de PHP es como módulo para web-server Apache, aunque puede funcionar con las limitaciones que se conoce, como un intérprete para ejecutar aplicaciones CGI en aquellos web-servers que no lo soporten como módulos.

2.7.1.2. Generalidades.

La característica más importante de PHP es que permite combinar código HTML y código PHP en una misma página (de extención .php), por ejemplo:

```
<html>
        <head>
        <title>Ejemplo PHP</title>
        </head>
        <body>
                <?php echo "Hola, este es un ejemplo de PHP!"; ?>
        </body>
</html>
```

Se puede ver que no es lo mismo que un script CGI escrito en otro lenguaje de programación como Perl o C. En vez de escribir un programa con muchos comandos para crear una salida en HTML, escribimos el código HTML con cierto código PHP introducido en el mismo, que producirá cierta salida (en nuestro ejemplo, producir un texto). El código PHP se incluye entre etiquetas especiales de comienzo y final que nos permitirán entrar y salir del modo PHP.

Lo que distingue a PHP de la tecnología Javascript, la cual se ejecuta en la máquina cliente, es que el código PHP es ejecutado en el servidor. Si tuviésemos un script similar al de nuestro ejemplo en nuestro servidor, el cliente solamente

recibiría el resultado de su ejecución en el servidor, sin ninguna posibilidad de determinar que código ha producido el resultado recibido. El servidor web puede ser incluso configurado para que procese todos los ficheros HTML con PHP.

2.7.1.3. Seguridades.

PHP es un potente lenguaje y el interprete, tanto incluido en el servidor web como módulo o ejecutado como un binario CGI, puede acceder a ficheros, ejecutar comandos y abrir comunicaciones de red en el servidor. Todas estas características hacen que lo que se ejecute en el servidor web sea inseguro por defecto. PHP ha sido diseñado específicamente, para ser un lenguaje más seguro para escribir programas CGI, que Perl o C y con la correcta selección de las opciones de configuración del tiempo de compilación y ejecución se consigue la exacta combinación de libertad y seguridad que se necesita.

Ya que existen diferentes modos de utilizar PHP, existen multitud de opciones de configuración que permiten controlar su funcionamiento. Una gran selección de opciones garantiza que se pueda usar PHP para diferentes usos, pero también significa que existen combinaciones de estas opciones y configuraciones del servidor que producen instalaciones inseguras. Este capítulo explica las diferentes combinaciones de opciones de configuración y las situaciones donde pueden ser usadas de manera segura.

2.7.1.4. Funcionalidad.

- Funciones de calendario y manipulación de calendarios usando MCAL. (Modular Calendar Access Library).
- Programación orientada a objetos.
- Funciones para creación de archivos PDF.
- Parser de documentos XML.
- WDDX (Standard para el intercambio de estructuras de datos entre distintos lenguajes de programación utilizando XML).
- Funciones de comprensión de datos.

- Manejo de archivos DBM (Data Base Management).
- Funciones para manejo de directorios.
- Funciones de encriptación de datos.
- Funciones de acceso al filesystem.
- Funciones para manejo de FTP.
- Funciones de hashim.
- Generación dinámica de imágenes.
- Manejo de mail IMAP y POP3.
- Funciones para envío de mail.
- Funciones matemáticas.
- Acceso a base de datos (Mysql, Oracle, Postgress, Sybase, etc.) .
- Manejo de expresiones regulares.
- Manejo de Sesiones.

2.7.2. Introducción al Lenguaje.

PHP puede ser utilizado en cualquiera de los principales sistemas operativos del mercado, incluyendo Linux, muchas variantes Unix (incluido HP-UX, Solaris y OpenBSD), Microsoft Windows, Mac OS X, RISC OS y probablemente alguno más. PHP soporta la mayoría de servidores web de hoy en día, incluyendo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape e iPlanet, Oreilly Website, y muchos otros. PHP tiene módulos disponibles para la mayoría de los servidores, para aquellos otros que soporten el estándar CGI, PHP puede usarse como procesador CGI.

También tiene soporte para comunicarse con otros servicios usando protocolos tales como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (en Windows) y muchos otros. Se pueden crear raw sockets. PHP soporta WDDX para intercambio de datos entre lenguajes de programación en web. Y hablando de interconexión, puede utilizar objetos Java de forma transparente como objetos.

Tiene unas características muy útiles para el proceso de texto, desde expresiones regulares POSIX Extended ó Perl hasta parseador de documentos XML. Para parsear y acceder documentos XML, soporta los estándares SAX y

DOM. Además puede utilizar la extensión XSLT para transformar documentos XML.

2.7.2.1. Comentarios.

PHP soporta comentarios tipo 'C', 'C++' y shell de Unix. Por ejemplo:

```
<?php
echo "Esto es una prueba"; // Esto es un comentario tipo c++ para
una línea
/* Esto es un comentario multilínea
   y otra línea más de comentario*/
echo "Esto es aún otra prueba";
echo  "Una prueba final"; # Esto es un comentario tipo shell?>
```

2.7.2.2. Separación de Instrucciones.

Las instrucciones se separan igual que en C o Perl - terminando cada sentencia con un punto y coma. La etiqueta de cierre (?>) también implica el fin de la sentencia, así lo siguiente es equivalente:

```
<?php
echo "Esto es una prueba";?>
<?php echo "Esto es una prueba" ?>
```

2.7.2.3. Tipos de Datos.

PHP soporta los siguientes tipos:

- Array.
- Números en punto flotante.
- Entero.
- Objeto.
- Cadena.

El tipo de una variable normalmente no lo indica el programador; en su lugar, lo decide PHP en tiempo de ejecución dependiendo del contexto en el que se utilice esa variable.

Si se necesita obligar a que una variable se convierta a un tipo concreto, se podría forzar la variable o usar la función ***settype()***. Todas las variables en PHP se denotan utilizando el signo '$' precediendo al nombre de la variable.

2.7.2.4. Enteros.

Los enteros se puede especificar usando una de las siguientes sintaxis:

```
$a = 1234; # número decimal.
$a = -123; # un número negativo.
$a = 0123; # número octal (equivalente al 83 decimal).
$a = 0x12; # número hexadecimal (equivalente al 18 decimal).
```

2.7.2.5. Números en Punto Flotante.

Los números en punto flotante ("double") se pueden especificar utilizando cualquiera de las siguientes sintaxis:

```
$a = 1.234; $a = 1.2e3;
```

2.7.2.6. Cadenas.

Las cadenas de caracteres se pueden especificar usando uno de dos tipos de delimitadores. Si la cadena está encerrada entre dobles comillas ("), las variables que estén dentro de la cadena serán expandidas (sujetas a ciertas limitaciones de interpretación).

La segunda forma de delimitar una cadena de caracteres usa el carácter de comilla simple ("'"). Cuando una cadena va encerrada entre comillas simples, los únicos caracteres de escape que serán comprendidos son "\\"y "\'". Esto es por convenio, así que se pueden tener comillas simples y barras invertidas en una cadena entre comillas simples. Las variables no se expandirán dentro de una cadena entre comillas simples.

2.7.2.7. Array.

Los arrays actualmente actúan tanto como tablas hash (arrays asociativos) y como arrays indexados (vectores). Existen dos tipos de array como son:

- Array Unidimensionales.
- Array Multidimencionales.

Array Unidimensionales. PHP soporta tanto arrays escalares como asociativos. De hecho, no hay diferencias entre los dos. Se puede crear una array usando las funciones list() o array(), o se puede asignar el valor de cada elemento del array de manera explícita.

```
$a[0] = "abc";
$a[1] = "def";
$b["foo"] = 13;
```

También se puede crear un array simplemente añadiendo valores al array. Cuando se asigna un valor a una variable array usando corchetes vacíos, el valor se añadirá al final del array.

```
$a[] = "hola"; // $a[2] = "hola"
$a[] = "mundo"; // $a[3] = "mundo"
```

Los arrays se pueden ordenar usando las funciones ***asort()***, ***arsort()***, ***ksort()***, ***rsort()***, ***sort()***, ***uasort()***, ***usort()***, y ***uksort()*** dependiendo del tipo de ordenación que se desee. Se puede contar el número de elementos de un array usando la función ***count()***.

Se puede recorrer un array usando las funciones ***next()*** y ***prev()***. Otra forma habitual de recorrer un array es usando la función ***each()***.

Array Multidimencional. Los arrays multidimensionales son bastante simples actualmente. Para cada dimensión del array, se puede añadir otro valor [clave] al final:

```
$a[1] = $f; # ejemplos de una sola dimensión.
$a["foo"] = $f;
$a[1][2] = $f; # bidimensional
$a["foo"][2] = $f; # (se pueden mezclar indices numéricos y
asociativos.
$a[2]["bar"] = $f; # (se pueden mezclar indices numéricos y
asociativos.
$a["foo"][4]["bar"][0] = $f; # tetradimensional!
```

2.7.2.8. Objetos.

Para inicializar un objeto, se usa la sentencia new para instanciar el objeto a una variable.

```
class foo
{
function do_foo () {
echo "Doing foo.";
}
}
$bar = new foo;
$bar->do_foo();
```

2.7.2.9. Constantes.

Para definir una constante se utiliza la instrucción ***define*** de la forma

```
define("PI",3.14159265);
```

Luego las constantes pueden usarse como variables tradicionales ($PI) con la salvedad de que no se les puede asignar un valor.

2.7.2.10. Operadores.

Los operadores aritméticos en PHP también se asemejan al C:

ejemplo	Nombre	Resultado
$a + $b	Adición	Suma de $a y $b.
$a - $b	Substracción	Diferencia entre $a y $b.
$a * $b	Multiplicación	Producto de $a and $b.

ejemplo	Nombre	Resultado
$a / $b	División	Cociente de $a entre $b.
$a % $b	Módulo	Resto de $a dividido entre $b.

Tabla 2. Operadores

2.7.2.11. Asignación.

El operador básico de asignación es "=". A primera vista se podrá pensar que es el operador de comparación "igual que". Pero no. Realmente significa que el operando de la izquierda toma el valor de la expresión a la derecha, (esto es, "toma el valor de"). El valor de una expresión de asignación es el propio valor asignado. Esto es, el valor de "$a = 3"es 3. Esto permite hacer cosas curiosas como:

```
$a    ($b    4) + 5; // ahora $a es igual a 9, y $b vale 4.
```

Además del operador básico de asignación, existen los "operadores combinados" para todas las operaciones aritméticas y de cadenas que sean binarias. Este operador combinado permite, de una sola vez, usar una variable en una expresión y luego establecer el valor de esa variable al resultado de la expresión. Por ejemplo:

```
$a    ;
$a +  5; // establece $a a 8, como si hubiésemos escrito: $a    $a
+ 5;
$b    "Hola ";
$b  .  "Ah!"; // establece $b a "Hola Ah!", igual que si
hiciésemos $b    $b . "Ah!";
```

2.7.2.12. Comparación.

Los operadores de comparación, como su nombre indica, permiten comparar dos valores.

Ejemplo	Nombre	Resultado
$a == $b	Igualdad	Cierto si $a es igual a $b.
$a === $b	Identidad	Cierto si $a es igual a $b y si son del mismo tipo (sólo PHP4)
$a != $b	Desigualdad	Cierto si $a no es igual a $b.
$a < $b	Menor que	Cierto si $a es estrictamente menor que $b.
$a > $b	Mayor que	Cierto si $a es estrictamente mayor que $b.
$a <= $b	Menor o igual que	Cierto si $a es menor o igual que $b.
$a >= $b	Mayor o igual que	Cierto si $a es mayor o igual que $b.

Tabla 3. Comparación

Otro operador condicional es el operador "?:" (o ternario), que funciona como en C y otros muchos lenguajes.

```
(expr1) ? (expr2) : (expr3);
```

La expresión toma el valor *expr2* si *expr1* se evalúa a cierto, y *expr3* si *expr1* se evalúa a falso.

2.7.2.13. Operador *@*.

Cuando se antepone *@* a una expresión se suprimen los errores que la expresión pudiera generar

```
@($c = $a / $b);
```

2.7.2.14. Operadores Lógicos.

La razón de las dos variaciones de "y" y "o" es que operan con distinta precedencia.

ejemplo	Nombre	Resultado
$a and $b	Y	Cierto si tanto $a como $b son ciertos.
$a or $b	O	Cierto si $a o $b son ciertos.

ejemplo	Nombre	Resultado
$a xor $b	O exclusiva	Cierto si $a es cierto o $b es cierto, pero no ambos a la vez.
! $a	Negación	Cierto si $a no es cierto.
$a && $b	Y	Cierto si tanto $a como $b son ciertos.
$a || $b	O	Cierto si $a o $b son ciertos.

Tabla 4. Operadores Lógicos

2.7.2.15. Estructuras de Control.

Todo archivo de comandos PHP se compone de una serie de sentencias. Una sentencia puede ser una asignación, una llamada a función, un bucle, una sentencia condicional e incluso una sentencia que no haga nada (una sentencia vacía). Las sentencias normalmente acaban con punto y coma. Además, las sentencias se pueden agrupar en grupos de sentencias encapsulando un grupo de sentencias con llaves. Un grupo de sentencias es también una sentencia.

If:

```
if ($a > $b) {
print "a es mayor que b";
} elseif ($a == $b) {
print "a es igual que b";
} else {
print "a es mayor que b";
}
```

While:

```
$i = 1;
if ($i < 5) {
print "i no es lo suficientemente grande";
break;
}
$i += $factor;
if ($i < $minimum_limit) {
break;
}
print "i es correcto";
```

```
...process i...
} while(0);
```

For:

```
for ($i = 1; $i <= 10; $i++) {
print $i;
}
```

Foreach:

```
$a = array(
"uno" => 1,
"dos" => 2,
"tres" => 3,
"diecisiete" => 17
);
foreach($a as $k => $v) {
print "\$a[$k] => $v.\n";
}
```

Break:

```
$arr = array ('one', 'two', 'three', 'four', 'stop', 'five');
while (list (, $val) = each ($arr)) {
if ($val == 'stop') {
break; /* You could also write 'break 1;' here. */
}
echo "$val<br>\n";
}
```

Switch:

```
switch ($i) {
case 0:
print "i es igual a 0";
break;
case 1:
print "i es igual a 1";
break;
case 2:
print "i es igual a 2";
```

```
break;
}
```

2.7.2.16. Funciones.

Una función se define con la siguiente sintaxis:

```
function foo ($arg_1, $arg_2, ..., $arg_n) {
echo "Función de ejemplo.\n";
return $retval;
}
```

Cualquier instrucción válida de PHP puede aparecer en el cuerpo de la función, incluso otras funciones y definiciones de clases. PHP no soporta la sobrecarga de funciones, y tampoco es posible redefinir u ocultar funciones previamente declaradas.

2.7.3. Manejo de Base de Datos (MySQL).

La principal característica de PHP es su integración con diversos motores de bases de datos. PHP esta indebido para generar en forma más sencilla páginas web dinámicas a partir de información almacenada en bases de datos.

2.7.3.1. Conexión a Bases de Datos.

```
$db_link = mysql_connect ("localhost", "root", "contraseña");
```

La función realiza la conexión a la base de datos de MySQL y devuelve ***false*** si hubo algún error en la conexión o un ***link*** a la conexión a la base de datos en caso de que la conexión sea exitosa.

El link es un número que indica la sesión dentro del MySQL. Esta sesión se mantiene hasta que se finalice la conexión. Para finalizar la conexión de debe utilizar la función:

```
mysql_close();
```

Es muy importante cerrar la conexión a la base de datos una vez finalizadas las transacciones para evitar la sobrecarga en l motor de base de datos.

2.7.3.2. Sección de la Base de Datos

```
mysql_select_db (database name, db link);
```

Esta función configura cual es la base de datos que se utilizara por omisión. En este caso el link a utilizar en esta función es el link que se obtuvo al ejecutar la función ***mysql_connect.***

La función **mysql_select_db** devuelve el valor ***false*** en caso de que encuentre algún error, como por ejemplo la inexistencia de la base de datos.

2.7.3.3. Consulta a la Base de Datos.

```
$result = mysql_query (query, db link) ;
```

Nuevamente el link que se debe usar es el que se obtiene al conectarse a la base de datos. ***mysql_query*** devuelve ***false*** en caso de que la consulta no pueda ejecutarse o bien un ***result_set*** en los dos casos que devuelve algún tipo de datos como por ejemplo un select.

2.7.3.4. Cantidad de Filas Consultadas o Modificadas.

```
$cantidad = mysql_num_rows ($result);
```

Esta función devuelve la cantidad de filas que se obtuvieron luego de ejecutar una instrucción.

2.7.3.5. Obtención de Registros de una Columna.

```
$var = mysql_fetch_row ($result);
```

Toma un registro del ***result set*** y lo devuelve en un vector en el cual el elemento con índice 0 es la primera columna del registro, el elemento con índice 1 es la segunda columna, etc. Si no hay más registros que devolver devuelve ***false.*** Los valores de los campos solicitados en el ***result set*** son devueltos en un array.

2.7.4. Programación Orientada a Objetos en PHP.

2.7.4.1. Introducción.

PHP ofrece funcionalidades propias de POO. La programación orientada a objetos es una metodología de programación avanzada y bastante extendida, en la que los sistemas se modelan creando clases, que son un conjunto de datos y funcionalidades. Las clases son definiciones, a partir de las que se crean objetos. Los objetos son ejemplares de una clase determinada y como tal, disponen de los datos y funcionalidades definidos en la clase.

2.7.4.2. Conceptos de Programación Orientada a Objetos.

2.7.4.2.1. Encapsulación.

Una de las principales ventajas de la programación orientada a objetos es el concepto de encapsulación, conocido también como protección de datos, mediante el cual solo se pueden modificar lo s datos de un objeto accediendo a través de sus métodos u operaciones (interfaz del objeto). Nunca se pueden modificar directamente desde la aplicación principal.

La funcionalidad de un objeto esta sujeta a los datos que este maneja, una ventaja de usar objetos es que podemos modificar la funcionalidad de éste, añadir mejoras o corregir errores sin necesidad de cambiar su interfaz. Ya que en caso contrario un proyecto estaría sujeto a un mayor número de fallos y los cambios serían más costosos.

En la programación OO los objetos son únicos y son instancias a una clase determinada. En principio se define la clase con los atributos y métodos correspondientes y luego se crea el objeto que esta basado en una determinada

clase (esto se conoce como instancia). Se puede comparar a un objeto con una variable y la clase sería un tipo de dato definido por el programador.

La estructura mínima de una clase es la siguiente:

```
class nombre_clase {

}
```

Para que una clase sea útil, necesita atributos y operaciones. Podemos crear atributos como si de variables se trataran, con la palabra reservada var:

```
class nombre_clase {
      var $atributo1;
      var $atributo2;
}
```

Podemos crear métodos declarando funciones dentro de la definición de la clase, el siguiente código crea una clase llamada Nombre_Clase con dos operaciones que no hacen nada. Al metodo1 no le pasamos ningún parámetro y al metodo2 le pasamos dos parámetros.

```
class nombre_clase {

      function metodo1() {

      }

      function metodo2 ($param1, param2) {

      }
}
```

Para crear una instancia de un objeto debemos ejecutar el operador "new", devuelve en una variable un objeto de la clase que le estamos indicando.

```
$obj =  new nombre_clase;
```

2.7.4.2.2. Herencia.

El concepto de herencia se aplica cuando creamos una clase, que va a heredar los métodos y atributos de una ya definida, entonces la clase que hemos

3. CAPITULO III. Desarrollo del Proyecto.

3.1. Instalación y Reglas de Configuración del Disco Duro.

3.1.1. Introducción.

El servidor sun enterprise 250 soporta hasta seis unidades de disco UltraSCSI internas para su conexión sin interrupción del funcionamiento del sistema. Las unidades miden 3,5 pulgadas de anchura y 1 ó 1,6 pulgadas de altura. Todas las unidades internas están soportadas por la interfaz UltraSCSI a 40 Mbytes por segundo de la placa lógica principal del sistema. Todas las unidades se conectan a un panel posterior de circuitos impresos de seis discos que se monta en la parte posterior de la caja de discos del sistema.

La caja de discos del sistema dispone de seis indicadores luminosos verdes o amarillos, cada una de los cuales está localizado junto a un módulo de disco. Estos indicadores marcan el estado de funcionamiento y las condiciones de error asociadas a cada unidad de disco. Esta característica de diagnóstico ayuda al administrador a identificar con rapidez las unidades que precisan reparación.

Las ranuras de disco están numeradas del 0 al 5. El software Solstice DiskSuite que viene con el kit multimedia del servidor Solaris permite utilizar unidades de disco internas en diferentes configuraciones RAID. Se dispone de soporte para RAID 0 (striping), RAID 1 (duplicación), RAID 0+1 (striping y duplicación) y RAID 5 (striping con paridad intercalada). También puede configurar unidades como repuestos que se pueden utilizar sin interrupción del funcionamiento del sistema.

Para un mejor desempeño del servidor sun enterprise 250 se realizara la compra de un disco duro de la máxima capacidad que soporte dicho servidor.

3.1.2. Instalación.

La funcionalidad de conexión sin interrupción del funcionamiento del sistema de las unidades de disco internas permite la extracción e instalación de unidades mientras el sistema está en funcionamiento. Esta funcionalidad reduce

de forma significativa el tiempo de parada típico de la sustitución de unidades de disco.

El procedimiento de conexión sin interrupción del funcionamiento del sistema implica comandos de software para preparar el sistema antes de extraer una unidad de disco y para reconfigurar el entorno operativo después de instalar la unidad de repuesto. El procedimiento exacto de este tipo de conexión depende de la versión del entorno operativo Solaris en uso.

Para realizar un procedimiento de conexión de disco sin interrumpir el funcionamiento del sistema, debe conocer el nombre del dispositivo físico o lógico de la unidad que desea instalar o suprimir. Si el sistema encuentra un error de disco, a menudo puede encontrar mensajes relativos a los discos que están fallando o han fallado en la consola del sistema.

Esta información también queda registrada en el archivo o archivos ***/var/adm/messages***. Estos mensajes de error se refieren normalmente a las unidades de disco que han fallado por su nombre de dispositivo físico como por ejemplo ***/devices/pci@1f,4000/scsi@3/sd@b,0*** o por su nombre de dispositivo lógico como por ejemplo ***c0t11d0***. Además, algunas aplicaciones pueden arrojar información sobre un número de ranura de disco (del 0 al 5) o activar un indicador luminoso localizado junto a la unidad de disco misma.

3.1.3. Reglas de Configuración.

Las unidades de disco deben ser unidades estándar de sun de 3,5 pulgadas de anchura y compatibles con UltraSCSI, de 1 ó 1,6 pulgadas de altura. En el mismo sistema pueden mezclarse unidades compatibles con UltraSCSI de 1 y 1,6 pulgadas de altura. Los ID SCSI de los discos están conectados físicamente a los paneles posteriores de circuitos impresos de los discos. No es necesario definir puentes con ID SCSI en las unidades de disco mismas.

La dirección SCSI de destino (ID SCSI) de cada unidad de disco queda determinada por la ubicación de la ranura donde la unidad está conectada a su

panel posterior de circuitos impresos UltraSCSI. Las unidades de disco comparten el bus SCSI interno con los dispositivos de medios extraíbles SCSI del ensamblaje de medios extraíbles (RMA). El cable SCSI para medios extraíbles proporciona la terminación del bus SCSI interno.

3.1.4. Cuidados de Instalación.

Las placas de circuito impreso y las unidades de disco duro contienen componentes extremadamente sensibles a la electricidad estática que pueden resultar dañados por cargas normales de esta electricidad procedentes de la ropa o del entorno de trabajo. No toque los componentes ni sus partes metálicas sin adoptar las medidas de protección adecuadas contra la electricidad estática.

- Desconecte el cable de alimentación de CA de la toma de pared sólo si va a manejar la placa de distribución de potencia. El cable de alimentación de CA incluye una vía de descarga de la electricidad estática, por lo que puede dejarse enchufado durante los procedimientos de instalación o mantenimiento. La única ocasión en que debe desconectarse es cuando se va a realizar alguna operación de mantenimiento en la placa de distribución de potencia.
- Utilice una alfombrilla antiestática u otra superficie similar. Al instalar cualquier opción o realizar cualquier procedimiento de mantenimiento, coloque los componentes sensibles a la electricidad estática, como las unidades de disco, sobre una superficie antiestática. Pueden utilizarse como tal los elementos siguientes:

    - La bolsa utilizada para embalar algún componente de repuesto de Sun.
    - El material de embalaje utilizado para empaquetar algún componente de repuesto de Sun.
    - La alfombrilla de descarga electrostática de Sun (ESD), número de referencia 250-1088 (se puede pedir a los representantes comerciales de Sun).

- Cualquier alfombrilla ESD desechable que se suministre con los componentes u opciones de repuesto.

- Utilice una muñequera antiestática. Conecte un extremo de la cinta al metal de la carcasa del sistema y el otro extremo a la muñeca. Consulte las instrucciones recibidas con la muñequera.
- Retire ambos extremos de la cinta después de finalizar la instalación o el procedimiento de mantenimiento

3.1.5. Particiones del Disco Duro.

Cada disco duro constituye una unidad física distinta. Sin embargo, los sistemas operativos no trabajan con unidades físicas directamente sino con unidades lógicas. Dentro de una misma unidad física de disco duro puede haber varias unidades lógicas. Cada una de estas unidades lógicas constituye una partición del disco duro. Esto quiere decir que podemos dividir un disco duro en, por ejemplo, dos particiones (dos unidades lógicas dentro de una misma unidad física) y trabajar de la misma manera que si tuviésemos dos discos duros (una unidad lógica para cada unidad física).

Particiones y directorios. Ambas estructuras permiten organizar datos dentro de un disco duro. Sin embargo, presentan importantes diferencias:

- Las particiones son divisiones de tamaño fijo del disco duro; los directorios son divisiones de tamaño variable de la partición.
- Las particiones ocupan un grupo de cilindros contiguos del disco duro (mayor seguridad); los directorios suelen tener su información desperdigada por toda la partición.
- Cada partición del disco duro puede tener un sistema de archivos (sistema operativo) distinto; todos los directorios de la partición tienen el sistema de archivos de la partición.

Como mínimo, es necesario crear una partición para cada disco duro. Esta partición puede contener la totalidad del espacio del disco duro o sólo una parte. Las razones que nos pueden llevar a crear más de una partición por disco se suelen reducir a tres.

a. **Razones organizativas**. Considérese el caso de un ordenador que es compartido por dos usuarios y, con objeto de lograr una mejor organización y seguridad de sus datos deciden utilizar particiones separadas.

b. Instalación de más de un sistema operativo. Debido a que cada sistema operativo requiere (como norma general) una partición propia para trabajar, si queremos instalar dos sistemas operativos a la vez en el mismo disco duro (por ejemplo, Windows 98 y Linux), será necesario particionar el disco.

c. Razones de eficiencia. Por ejemplo, suele ser preferible tener varias particiones FAT pequeñas antes que una gran partición FAT. Esto es debido a que cuanto mayor es el tamaño de una partición, mayor es el tamaño del grupo y, por consiguiente, se desaprovecha más espacio de la partición.

El disco duro del servidor de la PUCESA constará de dos particiones que serán una partición principal de 10 MB y una partición lógica de 20 MB para almacenar los datos que arrojen los diferentes sistemas que maneje y administre la PUCESA.

3.2. Análisis del Sistema.

El análisis del sistema engloba a todos los elementos que van a constituir el Sistema Automático de Gestión Administrativa (SAGA), identificando las necesidades de la Dirección Administrativa de la Universidad, valorando la viabilidad del sistema, asignando funciones a hardware, software, personal; realizando un análisis técnico de las especificaciones y necesidades del sistema.

3.2.1. Identificación de las Necesidades.

La carencia de un sistema de Gestión Administrativa automatizado en la Dirección Administrativa de la PUCESA nos permite distinguir las siguientes necesidades:

- Un Sistema automatizado para la emisión de contratos.
- Un sistema automatizado para la emisión de certificados.
- Una interfase de fácil utilización .
- Un sistema estructurado de alto nivel.
- Una base de datos relacional que permita almacenar los diferentes datos en forma segura.
- Una interfase Web de fácil utilización.
- Un sistema de fácil instalación en el servidor Sun enterprise 250 multiplataforma de la universidad y de igual manera para servidores que se usen en lo posterior.

3.2.2. Estudio de Viabilidad.

Se analizara específicamente los resultados en cuatro áreas:

a) **Viabilidad Técnica.** Con la adquisición del disco duro el servidor podrá tener el espacio necesario para almacenar los datos que arroje el sistema y los demás sistemas.
b) **Viabilidad Económica.** Los costos del desarrollo e implementación del sistema son accesibles en lo que se refiere a hardware y software a utilizar.
c) **Viabilidad Legal.** Se a determinado que no existe ninguna infracción que pueda ocurrir a causa del sistema.
d) **Estudio de Alternativas.** En caso que el servidor de la universidad no este a disposición por cualquier circunstancia el sistema se podrá implementar de forma local en algún computador que cumpla con las necesidades mínimas.

Basados en estudios de viabilidad del sistema en la universidad se ha determinado que es factible el desarrollo del proyecto.

3.2.3. Análisis Técnico.

El análisis técnico abarca los conceptos del sistema recogiendo información de factibilidad, facilidad de mantenimiento, posibilidad de producción y rendimiento.

- **Análisis de Requerimientos del Servidor Sun Enterprise 250.**

    - PHP Versión 4.2.2.
    - MySQL versión 4.1.14.
    - Apache.
    - Módulos de conexión de PHP con MySQL.

- **Disponibilidad de Recursos.** Se poseen los equipos de cómputo y aplicaciones de desarrollo necesarios para la realización del sistema
- **Tecnología.** La tecnología actual permite el desarrollo eficiente de la aplicación, motivo por el cual el desarrollo del sistema no se encuentra limitado.
- **Riesgos del Desarrollo.** Se puede cumplir con todas las funciones y requerimientos que se necesita.

3.2.4. Análisis de Requerimientos.

Tras el análisis de las necesidades en la Dirección Administrativa de la PUCESA se llego a las siguientes conclusiones:

- Es posible el desarrollo del sistema de Gestión Administrativa por medio de la utilización de métodos y funciones, con lo cual se llegará al propósito del despacho de contratos y certificados de modo más eficiente.

- El front end del sistema será desarrollado en lenguaje PHP. Este lenguaje multiplataforma esta orientado a aplicaciones web. El sistema requiere soporte de programación de alto nivel orientada a objetos y procedimental, y que el lenguaje brinde la facilidad de conexión con la base de datos.
- El back end del sistema será desarrollado en MySQL, al igual que PHP también estará hospedado en el servidor de la universidad. MySQL es un potente sistema de gestión de bases de datos que con la utilización del motor del almacenamiento InnoDB, brindará la transaccionalidad de datos que el sistema requiere.

3.2.4.1. Especificaciones.

Con el propósito de que el sistema de Gestión Administrativa se desarrolle con mayor calidad, se detallaran las siguientes especificaciones:

Especificaciones del Sistema de Gestión Administrativa

Lenguajes de Programación. PHP, HTML.

Base de Datos. MySQL.

Entorno en el que va a operar. Sistemas Operativos tales como Solaris, Unix, Linux y en Browser que es el Internt explorer, además el sistema funcionará a través de Internet .

Funciones:

- Validación y control de usuarios.
  - Administrador
  - Usuario
- Mantenimiento de la base de datos.
  - Creación.
  - Modificación
  - Eliminación

- Administración de empleados.
- Asignación de contratos y certificados.
- Mantenimiento de Contratos y certificados.
- Reportes de Empleados.

Restricciones:

- El sistema será utilizado exclusivamente para el proceso de emisión de contratos y certificados.
- El sistema no realizara directamente funciones de respaldo de procesamiento de datos ni operaciones de backup y recuperación automatizado.
- El sistema necesita aproximadamente 7 Mb de espacio en disco duro para el alojamiento de las páginas web, y 5 Mb para el almacenamiento de la base de datos.

3.2.5. Análisis Estructurado.

3.2.5.1. Diagrama de Flujo de Datos Nivel 0

Representa un modelo fundamental del sistema, Se simbolizan entidades externas e internas de forma general, salidas, entradas y procesos que realiza el sistema.

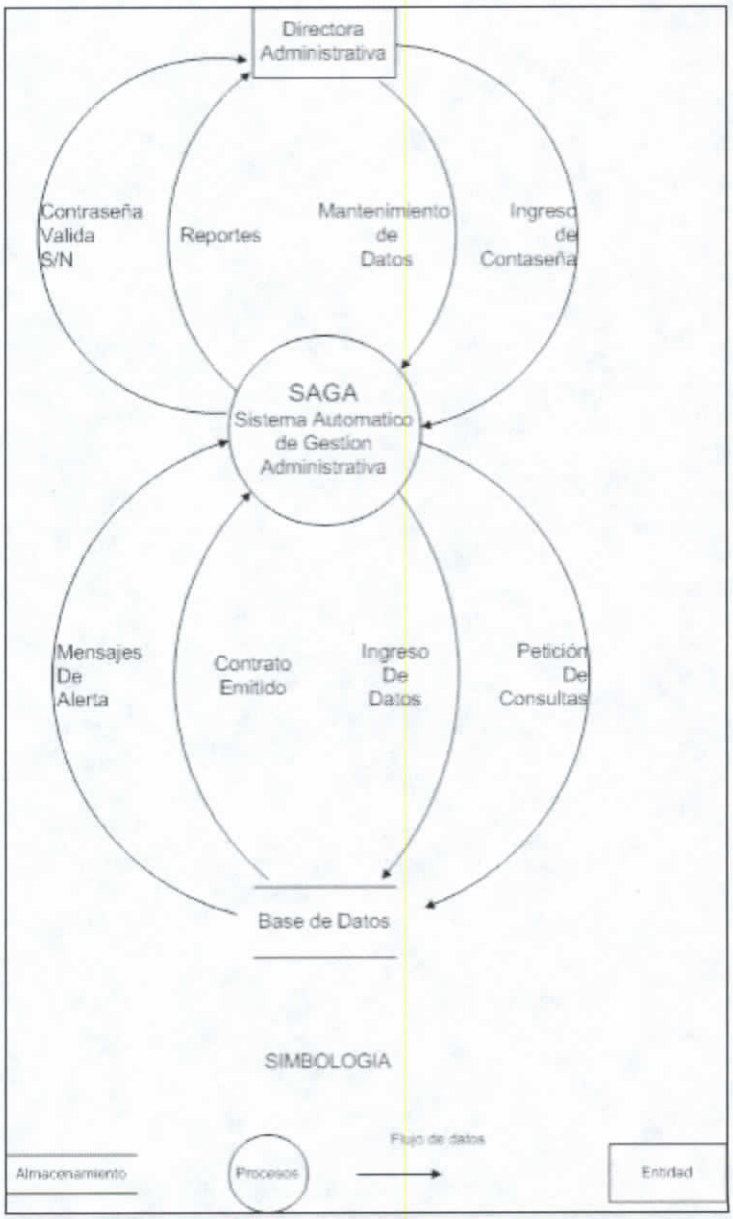


Ilustración 10. Diagrama de Flujo de Datos Nivel 0

3.2.5.2. Diagrama de Flujo de Datos Nivel 1.

En el diagrama de flujo de datos del nivel 1 se expone con mayor detalle las rutinas y procesos del sistema, así como la interacción con usuario.

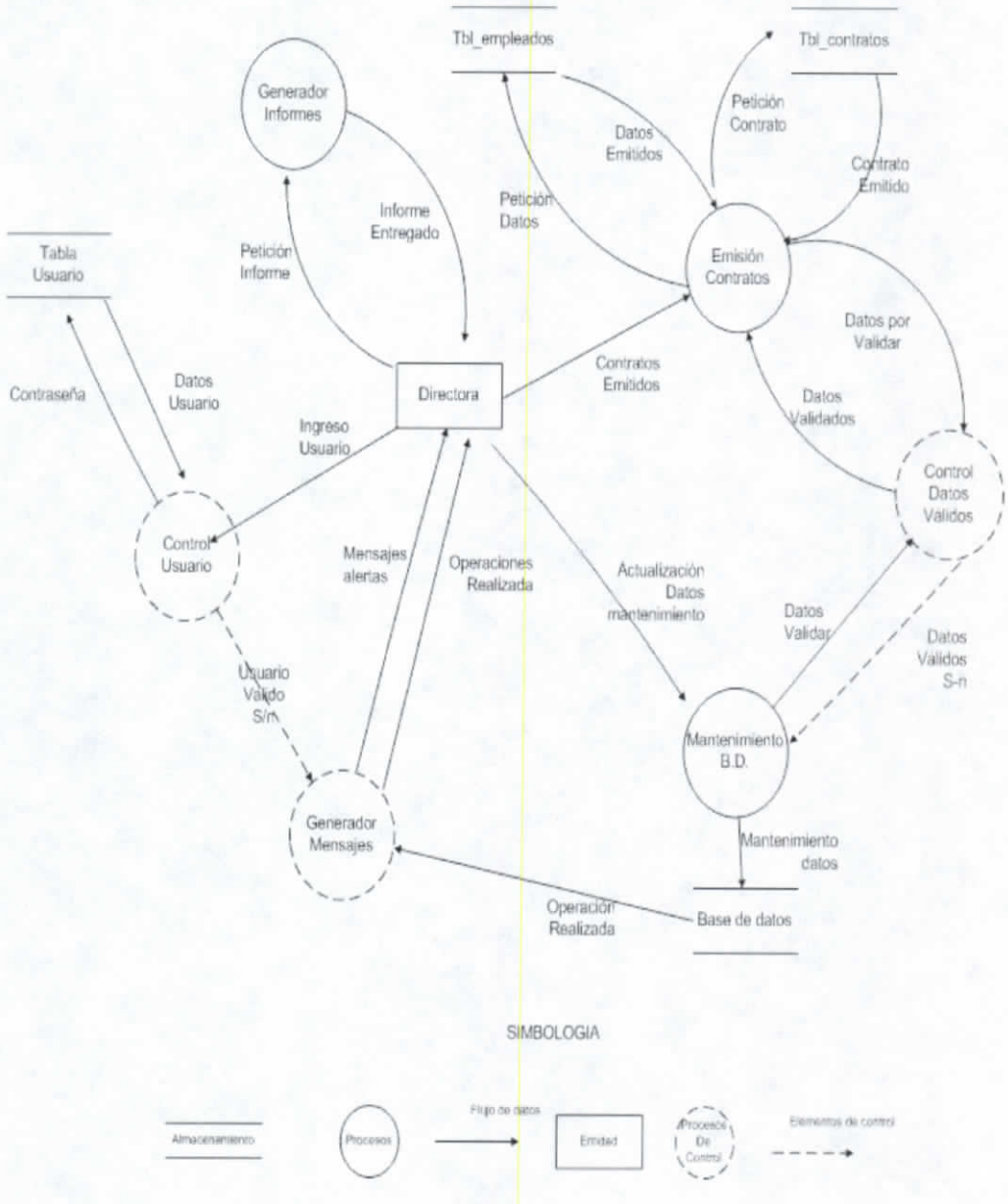


Ilustración 11. Diagrama de Flujo de Datos Nivel 1

3.2.5.3. Diagrama de Flujo de Datos Nivel 2.

En el diagrama de flujo de datos nivel 2 se encuentra especificado el proceso de la emisión de contratos y certificados que realiza el sistema.

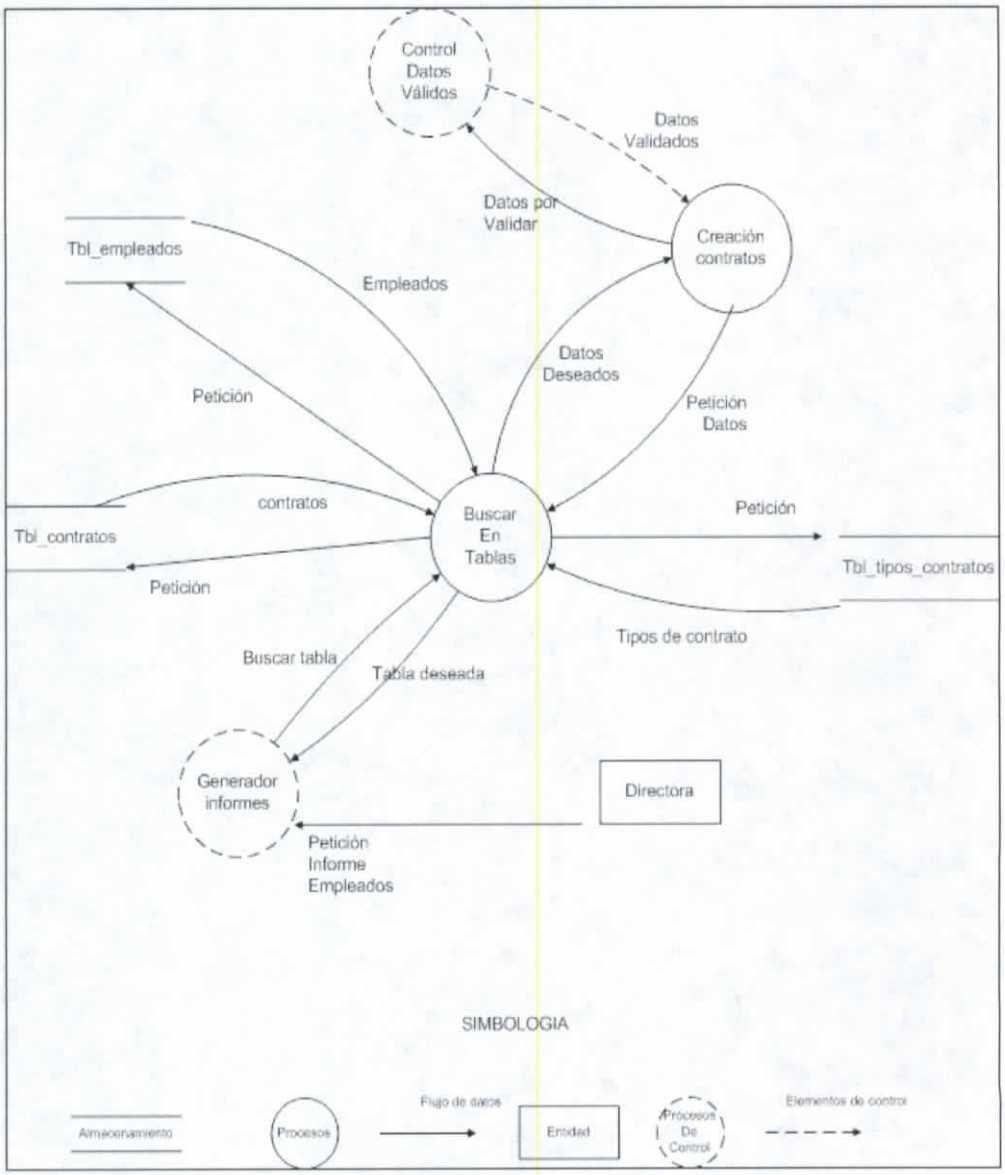


Ilustración 12. Diagrama de Flujo de Datos Nivel 2

3.2.5.4. Diagrama de Fijo de Datos Nivel 2

Generar Informes

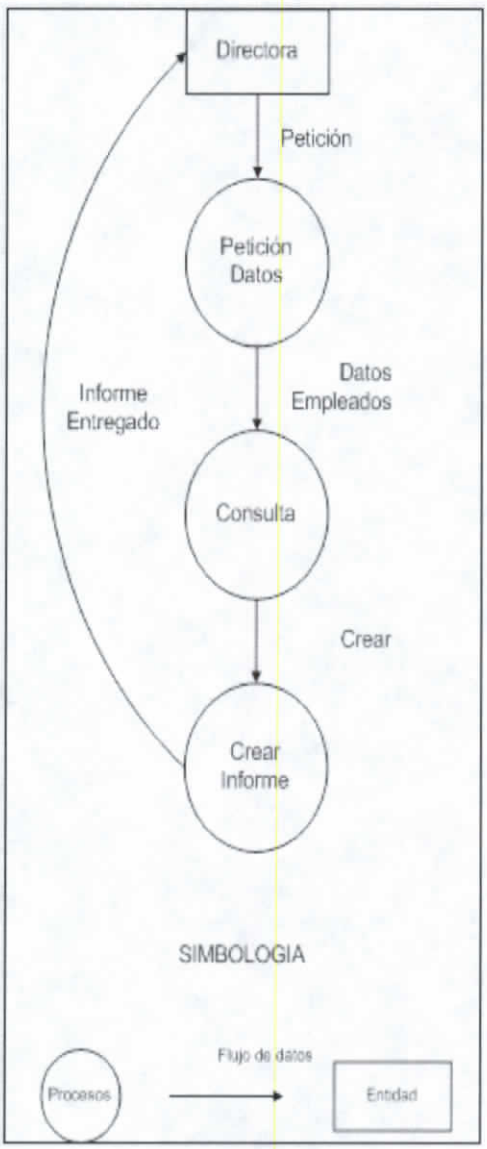


Ilustración 13. Diagrama reflujo de Datos Nivel 2 Generar Informe

3.2.5.5. Diagrama de Fijo de Datos Nivel 2

Mantenimiento a la Base de Datos

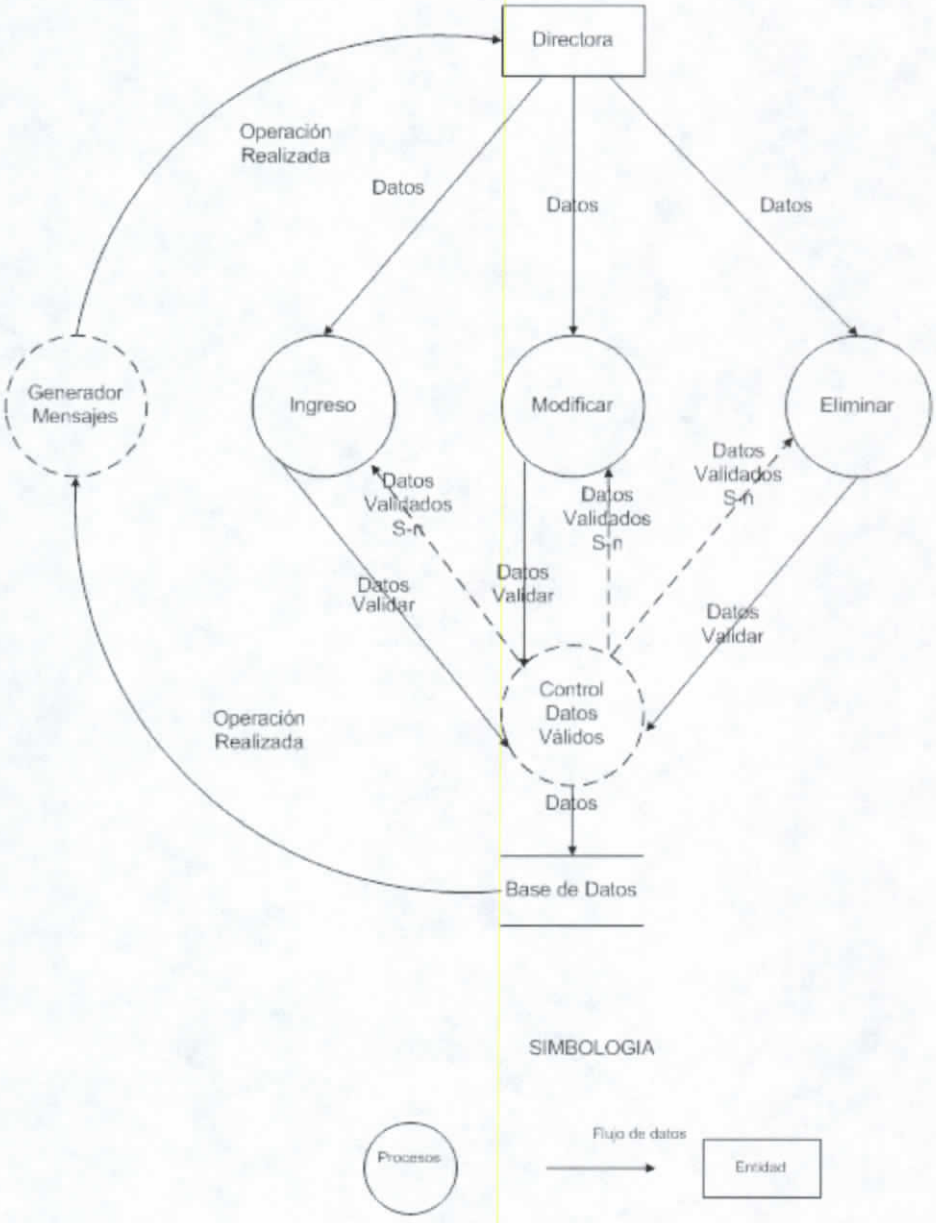


Ilustración 14. Diagrama de Flujo de Datos Nivel 2 Mantenimiento B.D.

3.3. Diseño del Sistema.

El modelado del diseño del sistema SAGA toma en cuenta los requisitos, análisis del sistema, viabilidad, necesidades, análisis técnico, especificaciones y análisis estructurado y los traduce a una representación técnica de software; una traducción lo más cercana al lenguaje de máquina del problema real.

Se desarrolla un diseño detallado tomando en cuenta refinamientos de la representación arquitectónica con una estructura de datos pormenorizada. La elaboración del diseño y calidad del software toma en consideración los siguientes puntos:

- El diseño debe ser modular y estos módulos han de ser divididos en forma lógica.
- El diseño debe tener representaciones distintas y separadas de datos y procedimientos.
- El diseño exige una organización jerárquica y un uso inteligente de control.
- El diseño debe llevar a interfases que reduzcan la complejidad de las conexiones.
- El diseño debe poder ser reproducido y debe ser conducido por los resultados del análisis de requisitos.

El diseño del sistema SAGA modela en forma detallada la base de datos, el sitio web e interfases.

3.3.1. Diseño de la Base de Datos.

El diseño de la base de datos del sistema SAGA expone en forma detallada todos los objetos que la conforman tales como entidades con sus atributos, relaciones, índices, tipo de datos y longitudes de cada columna.

Información del Modelo Físico. Detalla información básica y estadística del modelo físico de la base de datos.

Diagrama Entidad Relación. Define en forma gráfica a las entidades que conforman la base de datos del sistema SAGA con sus relaciones y atributos.

Tablas. Pormenoriza a todas las entidades con sus respectivas columnas, con claves primarias y secundarias, índices y relaciones.

Columnas. Describe cada atributo de las entidades de la base de datos, tipo de dato, si es clave primaria o secundaria si en un dato nulo o único.

Relaciones. Individualiza la relación existente en la base de datos, el tipo de relación, su tabla padre o hija y su cardinalidad.

Índices. Detalla a los índices que identifica las claves foráneas que se encuentran en una determinada entidad.

Información del Modelo Físico

Nombre de la Base de Datos	SAGA
Tablas	9
Columnas	71
Índices	6
Relaciones	6

Tabla 5. Información del Modelo Físico

Diagrama Entidad Relación

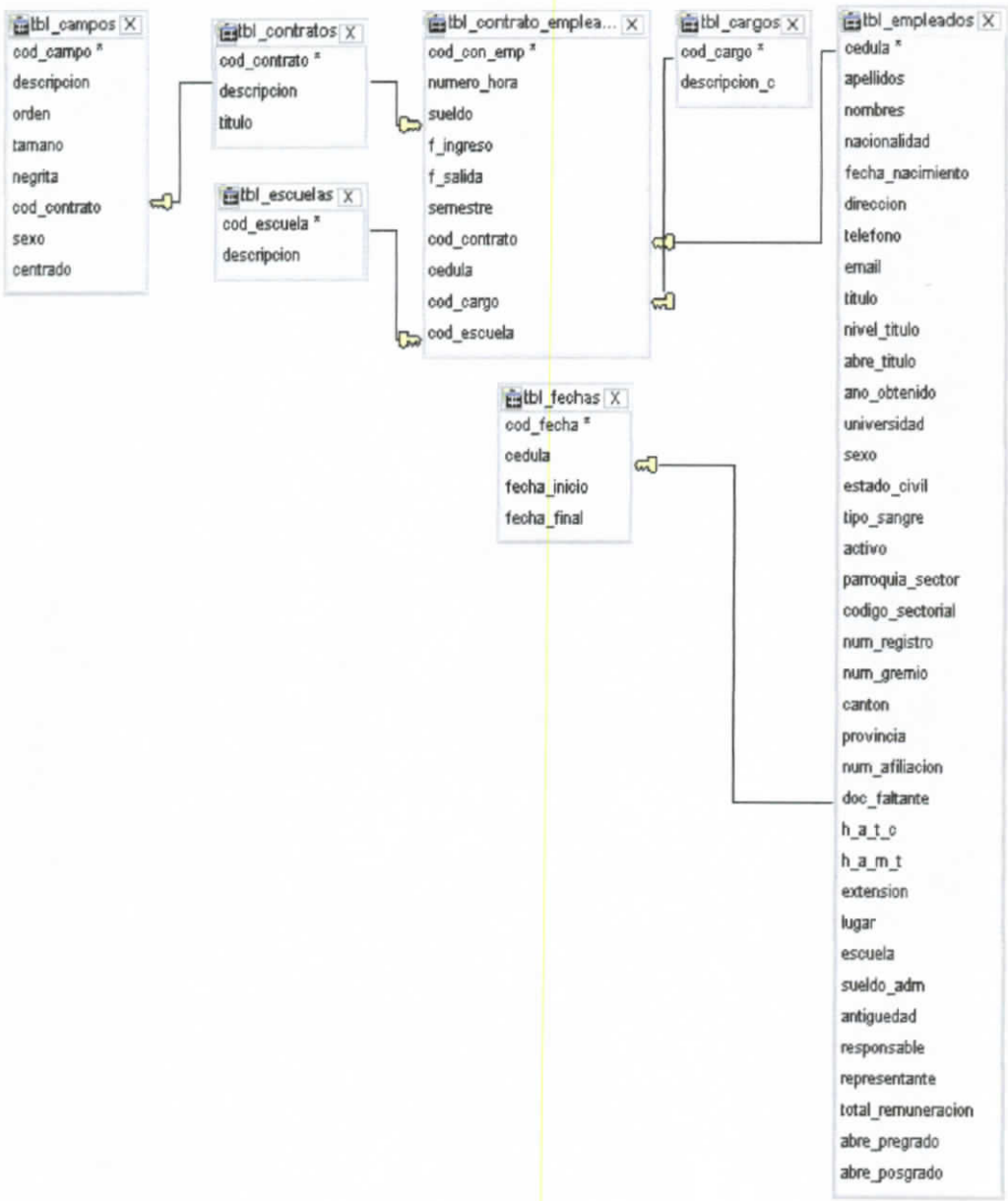


Ilustración 15. Diagrama Entidad Relación

3.3.1.1. Modelo Físico de la Base de Datos.

/*Column Information For - saga.tbl_campos*/

Field	Type	Collation	Null	Key	Default	Extra
cod_campo	int(11)	NULL		PRI	(NULL)	auto_increment
descripcion	text	latin1_swedish_ci	YES		(NULL)	
orden	int(11)	NULL	YES		(NULL)	
tamano	int(11)	NULL	YES		(NULL)	
negrita	char(1)	latin1_swedish_ci	YES		N	
cod_contrato	int(11)	NULL	YES	MUL	(NULL)	
sexo	char(3)	latin1_swedish_ci	YES		H	
centrado	char(1)	latin1_swedish_ci				

/*Column Information For - saga.tbl_cargos*/

Field	Type	Collation	Null	Key	Default	Extra
cod_cargo	int(11)	NULL		PRI	(NULL)	auto_increment
descripcion_c	varchar(50)	latin1_swedish_ci				

/*Column Information For - saga.tbl_contrato_empleado*/

Field	Type	Collation	Null	Key	Default	Extra
cod_con_emp	int(11)	NULL		PRI	(NULL)	auto_increment
numero_hora	int(11)	NULL			0	
sueldo	int(11)	NULL			0	
f_ingreso	date	NULL			0000-00-00	
f_salida	date	NULL			0000-00-00	
semestre	varchar(80)	latin1_swedish_ci				
cod_contrato	int(11)	NULL	YES	MUL	(NULL)	
cedula	varchar(11)	latin1_swedish_ci		MUL		
cod_cargo	int(11)	NULL		MUL	0	
cod_escuela	int(11)	NULL		MUL	0	

/*Column Information For - saga.tbl_contratos*/

Field	Type	Collation	Null	Key	Default	Extra
cod_contrato	int(11)	NULL		PRI	(NULL)	auto_increment
descripcion	varchar(100)	latin1_swedish_ci				
titulo	varchar(100)	latin1_swedish_ci				

/*Column Information For - saga.tbl_empleados*/

Field	Type	Collation	Null	Key	Default	Extra
cedula	varchar(11)	latin1_swedish_ci		PRI		
apellidos	varchar(25)	latin1_swedish_ci				
nombres	varchar(25)	latin1_swedish_ci				
nacionalidad	varchar(15)	latin1_swedish_ci				
fecha_nacimiento	date	NULL			0000-00-00	
direccion	varchar(70)	latin1_swedish_ci				
telefono	varchar(20)	latin1_swedish_ci				
email	varchar(30)	latin1_swedish_ci				
titulo	varchar(80)	latin1_swedish_ci				
nivel_titulo	varchar(15)	latin1_swedish_ci				
abre_titulo	varchar(10)	latin1_swedish_ci				
ano_obtenido	varchar(10)	latin1_swedish_ci				
universidad	varchar(65)	latin1_swedish_ci				
sexo	varchar(10)	latin1_swedish_ci				
estado_civil	varchar(15)	latin1_swedish_ci				
tipo_sangre	varchar(5)	latin1_swedish_ci				
activo	char(2)	latin1_swedish_ci				
parroquia_sector	varchar(25)	latin1_swedish_ci				
codigo_sectorial	int(13)	NULL			0	
num_registro	int(18)	NULL			0	
num_gremio	varchar(10)	latin1_swedish_ci				
canton	varchar(10)	latin1_swedish_ci				
num_afiliacion	int(17)	NULL	YES		(NULL)	
doc_faltante	varchar(60)	latin1_swedish_ci				
h_a_t_c	int(4)	NULL			0	
h_a_m_t	int(4)	NULL			0	
extension	varchar(30)	latin1_swedish_ci				
lugar	varchar(30)	latin1_swedish_ci				
escuela	varchar(6)	latin1_swedish_ci				
sueldo_adm	float(137,2)	NULL			0.00	
antiguedad	float(137,2)	NULL	YES		(NULL)	
responsable	float(137,2)	NULL	YES		(NULL)	
representante	float(137,2)	NULL	YES		(NULL)	
total_remuneracion	float(137,2)	NULL			0.00	
abre_pregrado	varchar(7)	latin1_swedish_ci				
abre_posgrado	varchar(7)	latin1_swedish_ci				

/*Column Information For - saga.tbl_escuelas*/

Field	Type	Collation	Null	Key	Default	Extra
cod_escuela	int(11)	NULL		PRI	(NULL)	auto_increment
descripcion	varchar(50)	latin1_swedish_ci				

/*Column Information For - saga.tbl_fechas*/

Field	Type	Collation	Null	Key	Default	Extra
cod_fecha	int(11)	NULL		PRI	(NULL)	auto_increment
cedula	varchar(11)	latin1_swedish_ci		MUL	0	
fecha_inicio	date	NULL	YES		(NULL)	
fecha_final	date	NULL	YES		(NULL)	

/*Column Information For - saga.tbl_tipo_contrato*/

Field	Type	Collation	Null	Key	Default	Extra
cod_tipo_contrato	int(11)	NULL		PRI	(NULL)	auto_increment
descripcion	varchar(100)	latin1_swedish_ci				

3.3.1.2. Diccionario de datos.

Tabla Campo

PK	Name	Data type	Size	Precision	Values	Default	Auto Increment
✓	cod_campo	INTEGER	11	0			✓
	descripcion	TEXT	65535	0			
	orden	INTEGER	11	0			
	tamano	INTEGER	11	0			
	negrita	CHAR	1	0		N	
	cod_contrato	INTEGER	11	0			
	sexo	CHAR	3	0		H	
	centrado	CHAR	1	0			

Tabla Cargo

PK	Name	Data type	Size	Precision	Values	Default	Auto Increment
✓	cod_cargo	INTEGER	11	0			✓
	descripcion_c	VARCHAR	50	0			

Tabla Contrato Empleado

PK	Name	Data type	Size	Precision	Values	Default	Auto Increment
✓	cod_con_emp	INTEGER	11	0			✓
	numero_hora	INTEGER	11	0		0	
	sueldo	INTEGER	11	0		0	
	f_ingreso	DATE	10	0		0000-00-00	
	f_salida	DATE	10	0		0000-00-00	
	semestre	VARCHAR	80	0			
	cod_contrato	INTEGER	11	0			
	cedula	VARCHAR	11	0			
	cod_cargo	INTEGER	11	0		0	
	cod_escuela	INTEGER	11	0		0	

Tabla Contrato

PK	Name	Data type	Size	Precision	Values	Default	Auto Increment
✔	cod_contrato	INTEGER	11	0			✔
	descripcion	VARCHAR	100	0			
	titulo	VARCHAR	100	0			

Tabla Empleado

PK	Name	Data type	Size	Precision	Values	Default	Auto Increment
✔	cedula	VARCHAR	11	0			
	apellidos	VARCHAR	25	0			
	nombres	VARCHAR	25	0			
	nacionalidad	VARCHAR	15	0			
	fecha_nacimiento	DATE	10	0		0000-00-00	
	direccion	VARCHAR	70	0			
	telefono	VARCHAR	20	0			
	email	VARCHAR	30	0			
	titulo	VARCHAR	80	0			
	nivel_titulo	VARCHAR	15	0			
	abre_titulo	VARCHAR	10	0			
	ano_obtenido	VARCHAR	10	0			
	universidad	VARCHAR	65	0			
	sexo	VARCHAR	10	0			
	estado_civil	VARCHAR	15	0			
	tipo_sangre	VARCHAR	5	0			
	activo	CHAR	2	0			
	parroquia_sector	VARCHAR	25	0			
	codigo_sectorial	INTEGER	13	0		0	
	num_registro	INTEGER	18	0		0	
	num_gremio	VARCHAR	10	0			
	canton	VARCHAR	10	0			
	provincia	VARCHAR	20	0			
	num_afiliacion	INTEGER	17	0			
	doc_faltante	VARCHAR	60	0			
	h_a_t_c	INTEGER	4	0		0	
	h_a_m_t	INTEGER	4	0		0	
	extension	VARCHAR	30	0			
	lugar	VARCHAR	30	0			
	escuela	VARCHAR	6	0			
	sueldo_adm	FLOAT	137	2		0.00	
	antiguedad	FLOAT	137	2			
	responsable	FLOAT	137	2			
	representante	FLOAT	137	2			
	total_remuneracion	FLOAT	137	2		0.00	
	abre_pregrado	VARCHAR	7	0			
	abre_posgrado	VARCHAR	7	0			

Tabla Escuelas

PK	Name	Data type	Size	Precision	Values	Default	Auto Increment
✔	cod_escuela	INTEGER	11	0			✔
	descripcion	VARCHAR	50	0			

Tabla Fechas

PK	Name	Data type	Size	Precision	Values	Default	Auto Increment
✔	cod_fecha	INTEGER	11	0			✔
	cedula	VARCHAR	11	0		0	
	fecha_inicio	DATE	10	0			
	fecha_final	DATE	10	0			

Tabla Tipo Contrato

PK	Name	Data type	Size	Precision	Values	Default	Auto Increment
✔	cod_tipo_contrato	INTEGER	11	0			✔
	descripcion	VARCHAR	100	0			

Tabla Usuarios

PK	Name	Data type	Size	Precision	Values	Default	Auto Increment
✔	COD_USUARIO	INTEGER	11	0			✔
	USUARIO	VARCHAR	20	0			
	CONTRASENA	VARCHAR	20	0			

3.3.2. Diseño del Sitio Web y la Interfase.

El diseño del sitio web de SAGA define la interfase del sistema de forma jerárquica revelando los usuarios y las plantillas que tienen acceso.

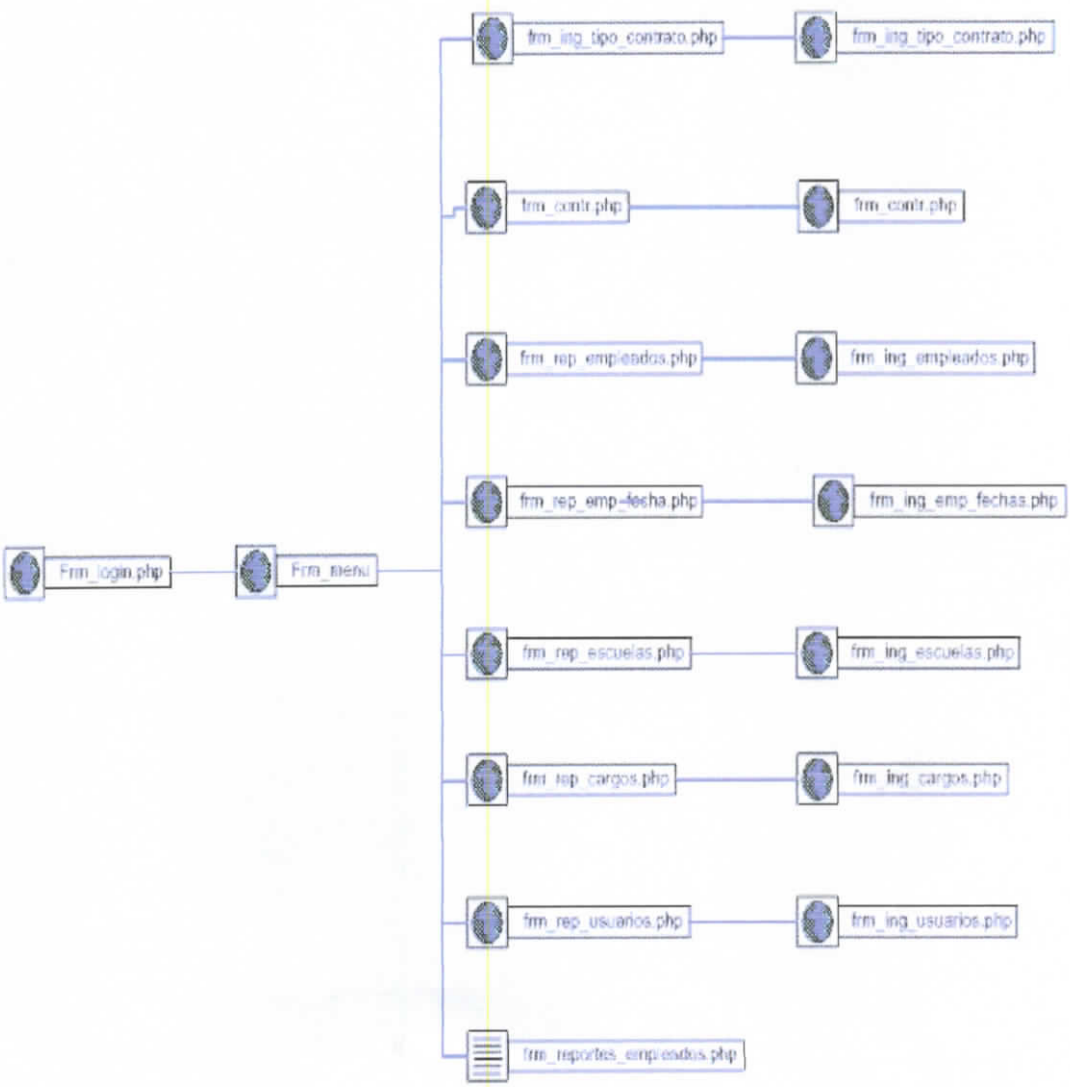


Ilustración 16. Diseño del Sitio Web y la Interfase.

A continuación se detalla cada archivo del sitio, con su extensión y una descripción.

Nombre del Archivo	Descripción
Ayuda	Carpeta de Ayuda
Imágenes	Carpeta de Imágenes
Conversor.php	Página de conversión de números a letras
encabezado.htm	Página de plantilla para el encabezado
encabezado_ini.htm	Página de plantilla para el encabezado del inicio
Estilos.css	Página de Estilos
frm_contr.php	Página de visualización de los contratos
frm_ing_campos.php	Página para el ingreso de los campos a cada contrato
frm_ing_cargos.php	Página de ingreso de los diferentes cargos
frm_ing_datos.php	Página ingreso de datos para emitir contratos
frm_ing_datos_cer.php	Página de ingreso de datos para emitir los certificados
frm_ing_emp_fechas.php	Ingreso de períodos de trabajo de los empleados
frm_ing_empleados.php	Página de ingreso de datos de los empleados
frm_ing_escuelas.php	Página de ingreso de las escuelas
frm_ing_tipo_contrato.php	Página de ingreso de los diferentes contratos
frm_ing_usuarios.php	Página de ingreso de Usuarios
frm_login.php	Pantalla de acceso de los usuarios al sistema
frm_logout.php	Página de cierre de sesión
frm_menu.php	Pantalla principal del sistema
frm_mod_camp.php	Modificar los campos de un contrato o certificado
frm_modi_cargos.php	Modificar los cargos existentes
frm_modi_contr.php	Modificar los contratos
frm_modi_empleados.php	Modificar los datos de los empleados
frm_modi_escuelas.php	Modificar las escuelas
frm_modi_fechas.php	Modificar los períodos de cada empleado
frm_modi_usuarios.php	Modificar las claves de los usuarios
frm_rep_cargos.php	Reporte de los cargos existentes
frm_rep_emp_fecha.php	Reporte de los periodos por empleado
frm_rep_empleados.php	Reporte de empleados
frm_rep_escuelas.php	Reporte de escuelas
frm_rep_tipo_contrato.php	Reporte de los tipos de contrato
frm_rep_usuarios.php	Reporte de usuarios
frm_reportes_empleados.php	Reporte de los datos de los empleados
frm_ver_cert.php	Pantalla del certificado emitido
frm_ver_contr.php	Pantalla del contrato emitido
frm_ver_reporte_emp.php	Informe de los datos de los empleados
Funcion_conectar.php	Página de conexión a la base de datos

3.4. Desarrollo del Sistema.

El desarrollo del sistema SAGA comprende la creación, codificación e interacción de los diferentes módulos entre si fundamentándose en el diseño y análisis del sistema, desarrollar la codificación del front end, generando las diferentes entidades con sus atributos para el back end y codificando los diferentes procesos, para alcanzar el perfecto funcionamiento en el proceso de emisión de contratos y certificados en la Dirección Administrativa de la PUCESA.

3.4.1. Creación de la Base de Datos.

Se a establecido una estructura SQL para la creación de la base de Datos, de las entidades con sus atributos y los valores iniciales que hospedan en el back end del sistema SAGA en el motor de base de datos MySQL la cual es la siguiente.

Creación de la Base de Datos

```
-- Creación de la Base de Datos 'saga'.
CREATE DATABASE saga;
-- Uso de la Base de Datos 'saga'
 USE saga;
```

Tabla tbl_campos

```
DROP TABLE IF EXISTS tbl_campos;
CREATE TABLE `tbl_campos` (
  `cod_campo` int(11) NOT NULL auto_increment,
  `descripcion` text,
  `orden` int(11) default NULL,
  `tamano` int(11) default NULL,
  `negrita` char(1) default 'N',
  `cod_contrato` int(11) default NULL,
  `sexo` char(3) default 'H',
  `centrado` char(1) NOT NULL default '',
  PRIMARY KEY  (`cod_campo`),
  KEY `cod_contrato` (`cod_contrato`),
```

```
  CONSTRAINT `FK_CONTRATO_1` FOREIGN KEY (`cod_contrato`)
REFERENCES `tbl_contratos` (`cod_contrato`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Tabla tbl_cargos

```
DROP TABLE IF EXISTS tbl_cargos;
CREATE TABLE `tbl_cargos` (
  `cod_cargo` int(11) NOT NULL auto_increment,
  `descripcion_c` varchar(50) NOT NULL default '',
  PRIMARY KEY  (`cod_cargo`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Tabla tbl_contratos

```
DROP TABLE IF EXISTS tbl_contratos;
CREATE TABLE `tbl_contratos` (
  `cod_contrato` int(11) NOT NULL auto_increment,
  `descripcion` varchar(100) NOT NULL default '',
  `titulo` varchar(100) NOT NULL default '',
  PRIMARY KEY  (`cod_contrato`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Tabla tbl_empleados

```
DROP TABLE IF EXISTS tbl_empleados;
CREATE TABLE `tbl_empleados` (
  `cedula` varchar(11) NOT NULL default '',
  `apellidos` varchar(25) NOT NULL default '',
  `nombres` varchar(25) NOT NULL default '',
  `nacionalidad` varchar(15) NOT NULL default '',
  `fecha_nacimiento` date NOT NULL default '0000-00-00',
  `direccion` varchar(70) NOT NULL default '',
  `telefono` varchar(20) NOT NULL default '',
  `email` varchar(30) NOT NULL default '',
  `titulo` varchar(80) NOT NULL default '',
  `nivel_titulo` varchar(15) NOT NULL default '',
  `abre_titulo` varchar(10) NOT NULL default '',
  `ano_obtenido` varchar(10) NOT NULL default '',
  `universidad` varchar(65) NOT NULL default '',
  `sexo` varchar(10) NOT NULL default '',
  `estado_civil` varchar(15) NOT NULL default '',
  `tipo_sangre` varchar(5) NOT NULL default '',
  `activo` char(2) NOT NULL default '',
  `parroquia_sector` varchar(25) NOT NULL default '',
  `codigo_sectorial` int(13) NOT NULL default '0',
  `num_registro` int(18) NOT NULL default '0',
  `num_gremio` varchar(10) NOT NULL default '',
```

```
  `canton` varchar(10) NOT NULL default '',
  `provincia` varchar(20) NOT NULL default '',
  `num_afiliacion` int(17) default NULL,
  `doc_faltante` varchar(60) NOT NULL default '',
  `h_a_t_c` int(4) NOT NULL default '0',
  `h_a_m_t` int(4) NOT NULL default '0',
  `extension` varchar(30) NOT NULL default '',
  `lugar` varchar(30) NOT NULL default '',
  `escuela` varchar(6) NOT NULL default '',
  `sueldo_adm` float(137,2) NOT NULL default '0.00',
  `antiguedad` float(137,2) default NULL,
  `responsable` float(137,2) default NULL,
  `representante` float(137,2) default NULL,
  `total_remuneracion` float(137,2) NOT NULL default '0.00',
  `abre_pregrado` varchar(7) NOT NULL default '',
  `abre_posgrado` varchar(7) NOT NULL default '',
  PRIMARY KEY  (`cedula`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Tabla tbl_escuelas

```
DROP TABLE IF EXISTS tbl_escuelas;
CREATE TABLE `tbl_escuelas` (
  `cod_escuela` int(11) NOT NULL auto_increment,
  `descripcion` varchar(50) NOT NULL default '',
  PRIMARY KEY  (`cod_escuela`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Tabla tbl_fechas

```
DROP TABLE IF EXISTS tbl_fechas;
CREATE TABLE `tbl_fechas` (
  `cod_fecha` int(11) NOT NULL auto_increment,
  `cedula` varchar(11) NOT NULL default '0',
  `fecha_inicio` date default NULL,
  `fecha_final` date default NULL,
  PRIMARY KEY  (`cod_fecha`),
  KEY `cod_empleado` (`cedula`),
  CONSTRAINT `tbl_fechas_ibfk_1` FOREIGN KEY (`cedula`) REFERENCES
`tbl_empleados` (`cedula`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Tabla tbl_usuarios

```
DROP TABLE IF EXISTS tbl_usuarios;
CREATE TABLE `tbl_usuarios` (
  `COD_USUARIO` int(11) NOT NULL auto_increment,
  `USUARIO` varchar(20) default NULL,
  `CONTRASENA` varchar(20) default NULL,
  PRIMARY KEY  (`COD_USUARIO`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

3.4.2. Codificación de Páginas Web Dinámicas.

En la codificación de las páginas web dinámicas se expondrá el código fuente de alguna de las páginas dinámicas más importantes para su mejor entendimiento.

3.4.2.1. Informe de cada uno de los Contratos.

En esta página se realiza un informe de cada uno de los tipos de contrato con sus cláusulas respectivas.

```
<?php session_start(); ?>
<?php ob_start(); ?>
<?php
require ("funcion_conectar.php");
require ("conversor.php");
?>
<?php
if (@$HTTP_SESSION_VARS["status"] <> "login") {
            header("Location:  frm_login.php");
            exit();
      }
?>
<?php
      if (@$cmd_nuevo <> "") {
            ob_end_clean();
            header("Location:
frm_ing_campos.php?cod_contrato=$cod_contrato");
            exit();
      }
?>
<?php
      if (@$HTTP_POST_VARS["checkbox"] <> "") {
      $link = conexion();
      $total_rows = $HTTP_SESSION_VARS["sv_total_rows_campo"];
      $checkbox = $HTTP_POST_VARS["checkbox"];
      for ($i=0; $i<$total_rows + 1; $i++) {
            if (isset ($checkbox[$i])) {
            $sql = "DELETE FROM tbl_campos WHERE cod_campo =
$checkbox[$i] ";
      if ($result = mysql_query($sql, $link)) {
      $HTTP_SESSION_VARS["sv_mensaje"] ="<div align='center'><font
color='#003366' class='text_correcto'><img
src='imagenes/visto.gif'>Gracias... Hemos recibido sus datos
</font></div>";
      } else {
      $HTTP_SESSION_VARS["sv_mensaje"] = "<div align='center'><font
color='##F20000'class='text_error'><img src='imagenes/errores.gif'
width='50' height='34'> Los Datos no se Guardaron </font></div>";
                              }
                        }
            }
      }
?>
<html>
<head>
<title>:: Pontificia Universidad Cat&oacute;lica Del Ecuador Sede Ambato
::</title>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link href="estilos.css" rel="stylesheet" type="text/css">
</head>
<?php
      require ("encabezado.htm");
?>
<p>
  <script type="text/javascript">
<!--
function EW_selectKey(elem) {
        var f = elem.form;
        if (!f.elements["checkbox[]"]) return;
        if (f.elements["checkbox[]"][0]) {
              for (var i=0; i<f.elements["checkbox[]"].length; i++)
                    f.elements["checkbox[]"][i].checked =
elem.checked;
        } else {
              f.elements["checkbox[]"].checked = elem.checked;
        }
}

//-->
</script><body>
  </p>
<form action="frm_contr.php" method="post" name="frm_contr">

  <TABLE width="90%"  border=0 align="center" cellPadding=5 cellSpacing=0
bgColor=#dbdbdb>
    <td width="30%" bgcolor="#666666" class="text_bold"
align="right"><FONT color=white><strong>Contrato:</strong></font></td>

      <td width="70%" valign="top">
        <?php
       $conexion = conexion();
       $sql = "SELECT cod_contrato, descripcion, titulo ";
       $sql .= "FROM tbl_contratos ";
       @$result = mysql_query($sql, $conexion);
       if (@$row = mysql_fetch_array($result)) {
       $select = "<select name='cod_contrato' class='text_bold'
onchange='location.href=\"frm_contr.php?";
             $select .= "cod_contrato=\"+frm_contr.cod_contrato.value'>";
             echo $select;
             $sw = 0;
             do {
                   $opcion = "<option value=".$row['cod_contrato'];
                   if ($row['cod_contrato'] == $cod_contrato) {
                         $opcion .= " Selected ";
                   }
                   else if (!isset ($_GET['cod_contrato']) and !isset
($cod_contrato) and $sw == 0 ) {
                         $opcion .= " Selected ";
                         $cod_contrato = $row['cod_contrato'];
                         $sw = 1;
                   }
                   $opcion .= ">".$row['titulo']."</option>";
                   echo $opcion;
             } while (@$row = mysql_fetch_array($result));
             echo'</select>';
       }
       else {
             echo '<font color="#8A3546" class="texto">Ningún Contrato
Ingresado!</font>';
             $cod_contrato = 0;
       }
       mysql_close($conexion);
?>
      </td>
          </TABLE>
```

```
          <TABLE cellSpacing=0 cellPadding=5 width="90%" bgColor=#FFFFFF
border=0 >

      <td class="texto" align=center><font size="3" color="#0033CC">
        <?php
       if (@$HTTP_SESSION_VARS["sv_mensaje"] <> "") {
              echo @$HTTP_SESSION_VARS["sv_mensaje"];
              $HTTP_SESSION_VARS["sv_mensaje"] = "";
       }
?>
         </font></td>
          </TABLE>

  <div align="center">
    <TABLE cellSpacing=0 cellPadding=5 width="90%" bgColor=#666666
class="text_bold" border=0>
      <TD width="85%"><FONT color=white><strong>Campos del
contrato</strong></font></TD>
      <TD width="10%"><FONT
color=white><strong>Eliminar</strong></font></TD>
      <td width="5%" class="texto" align=center><FONT color=white>
        <input name="checkbox" type="checkbox" class="botones"
onClick="EW_selectKey(this);">
        </font></td>
    </TABLE>
    <?php
       $link = conexion();
       $sql = "SELECT cam.cod_campo, cam.descripcion, tamano, negrita ";
       $sql .= "FROM tbl_campos cam, tbl_contratos con ";
       $sql .= "WHERE con.cod_contrato = cam.cod_contrato ";
       $sql .= "AND con.cod_contrato = $cod_contrato ";
       $sql .= "ORDER BY cam.orden ";
       $result = mysql_query($sql, $link);
       $total_rows = mysql_num_rows($result);
       $HTTP_SESSION_VARS["sv_total_rows_campo"] = $total_rows;
       if ($row = mysql_fetch_array($result)) {
              $color_sw = 1;
              $sw = 0;
              $color1 = "#e5e5e5";
              $color2 = "#dbdbdb";
              do {
              if($color_sw == 1) {
              $colorA = $color1;
              $colorB = $color2;
                    }
              echo '<TABLE cellSpacing=1 cellPadding=5 width="90%"
bgColor='.$colorB.' border=0>';
              echo '<TR bgColor='.$colorA.'>';
              $desc = '<TD width="90%" align=left><font
size="'.$row['tamano'].'">';
       if ($row['negrita'] == 'S') {
       $desc .= '<strong>'.$row['descripcion'].'</strong></font></TD>';
       } else {
              $desc .= $row['descripcion'].'</font></TD>';
                    }
       echo $desc;
       echo '<TD width="5%" class="texto"><a
href="frm_mod_camp.php?cod_campo='.$row['cod_campo'].'"><font
color="#000000" size=2>Ver</font></a></td>';
echo '<td width="5%" align=center ><input name="checkbox[]"
type="checkbox" id="'.$sw.'" value="'.$row['cod_campo'].'"></td>';
echo '</tr>';
echo '</TABLE>';
echo '<TABLE cellSpacing=0 cellPadding=0 width="90%" border=0>';
echo '<TD bgColor=#666666 colSpan=2 height=1></TD>';
echo '</TABLE>';
       $sw ++;
       if($color_sw == 1) {
```

```
            $colorA = $color2;
            $colorB = $color1;
            $color_sw = 0;
      } else {
            $color_sw = 1;
      }
            } while ($row = mysql_fetch_array($result));
            if ($total_rows == 1) {
            $mensaje = "¡Solo se ha encontrado un Campo por Contrato!";
            } else {
$mensaje = "¡Se encontraron: ".$total_rows." Campos por Contrato!";
            }
      echo '<TABLE cellSpacing=0 cellPadding=5 width="90%" border=0>';
      echo '<td height="21" colspan="5" valign="top"><font
color="#003366" class="texto">'.$mensaje.'</font></td>';
            echo '</table>';
      } else {
            echo '<TABLE cellSpacing=0 cellPadding=5 width="90%"
border=0>';
echo '<td height="21" colspan="5" valign="top"><font color="#003366"
class="texto">No se encontro ningún campo!</font></td>';
            echo '</table>';
      }
      mysql_close($link);
?>
    <br>
    <input name="cmd_nuevo" type="submit" id="cmd_nuevo" value="Nuevo
campo" class="button">
    <input name="cmd_eliminar" type="submit" id="cmd_eliminar"
value="Eliminar Seleccionados" class="button">
  </div>
</form>
<p> </p></body>
</html>
```

3.4.2.2. Código Fuente de Página Ver Contrato Emitido.

En este código fuente podrán observar la codificación de la pagina web dinámica para visualizar el contrato ya emitido, además de ver el proceso de intercambio de las palabras claves, con los respectivos datos del empleado.

```
<?php session_start(); ?>
<?php ob_start(); ?>
<?php
require ("funcion_conectar.php");
require ("conversor.php");
?>
<?php
if (@$HTTP_SESSION_VARS["status"] <> "login") {
      header("Location:  frm_login.php");
      exit();
}
?>
<?php
      if (@$HTTP_POST_VARS["checkbox"] <> "") {
      $link = conexion();
      $total_rows = $HTTP_SESSION_VARS["sv_total_rows_escuela"];
      $checkbox = $HTTP_POST_VARS["checkbox"];
      $cont = 0;
      for ($i=0; $i<$total_rows + 1; $i++) {
         if (isset ($checkbox[$i])) {
```

```
            $cont++;
            }
           }
        $cont2 = 0;
        for ($i=0; $i<$total_rows + 1; $i++) {
           if (isset ($checkbox[$i])) {
              $cont2++;
      sql = "SELECT descripcion FROM tbl_escuelas ";
           " WHERE cod_escuela = $checkbox[$i] ";
           $result = mysql_query($sql, $link);
           if ($row = mysql_fetch_array($result)) {
             do {
               if ($cont2 <> $cont -1) {
                @$descripcion .= $row['descripcion'].", ";
               } else {
                @$descripcion .= $row['descripcion']." y en ";
                }
              } while ($row = mysql_fetch_array($result));
                              }
                    }
              }

       }

?>

<?php if ($cod_contrato == "-- selccionar --" or $cod_cargo == "--
selccionar              --" or $semestre == "" or $f_inicio == ""){
$HTTP_SESSION_VARS["sv_mensaje"] = "<div align='center'><font
color='##F20000'class='text_error'><img src='imagenes/errores.gif'> Debe
Ingresar Los Datos Requeridos (*) </font></div>";
ob_end_clean();
header("Location: frm_ing_datos.php");
exit();

  }?>

<?php
      if (isset ($cod_escuela)){
      $link = conexion();
      $sql ="SELECT cod_escuela, descripcion ";
      $sql.=" FROM tbl_escuelas ";
      $sql.="where cod_escuela = $cod_escuela ";
      $result= mysql_query($sql,$link);
      if (@$row = mysql_fetch_array($result))
      {
            $cod_escuela=@$row['cod_escuela'];
            @$descripcion=@$row['descripcion'];
      }
      mysql_close($link);
}
?>
<?php
      if (isset ($cod_cargo)){
      $link = conexion();
      $sql ="SELECT cod_cargo, descripcion_c ";
      $sql.=" FROM tbl_cargos ";
      $sql.="where cod_cargo = $cod_cargo ";
      $result= mysql_query($sql,$link);
      if (@$row = mysql_fetch_array($result))
      {
            $cod_cargo=@$row['cod_cargo'];
            @$descripcion_c=@$row['descripcion_c'];
      }
      mysql_close($link);
}
?>
  <?php
```

```
        $link = conexion();
        $sql = "SELECT apellidos,nombres,cedula, abre_titulo, activo,
        sexo, abre_posgrado, abre_pregrado ";
        $sql .= "FROM tbl_empleados ";
        $sql .= "WHERE cedula = '$cedula' ";
        $result = mysql_query($sql, $link);
        if ($row = mysql_fetch_array($result)) {
              do {
                    $nombres = $row['nombres'];
                    $apellidos = $row['apellidos'];
                    $abre_titulo = $row['abre_titulo'];
                    $activo = $row['activo'];
                    $sexo = $row['sexo'];
                    $abre_pregrado = $row['abre_pregrado'];
                    $abre_posgrado = $row['abre_posgrado'];
              } while ($row = mysql_fetch_array($result));
        } else {
        echo '<div align="center"><b>No se encontro ningun
        empleado</b></div>';}
        mysql_close($link);

?>

<html>
<head>
<title>:: Pontificia Universidad Cat&oacute;lica Del Ecuador Sede Ambato
::</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body>
<div align="center"><strong>

  PONTIFICIA UNIVERSIDAD CAT&Oacute;LICA DEL ECUADOR <br>
  SEDE AMBATO</strong> </div>
<form name="form1" method="post" action="">
  <p>
    <?php
        $conexion = conexion();
        $sql = "SELECT cod_contrato, descripcion ";
        $sql .= "FROM tbl_contratos ";
        $sql .= "WHERE cod_contrato = $cod_contrato ";
        @$result = mysql_query($sql, $conexion);
        if (@$row = mysql_fetch_array($result)) {
              do {
              echo '<div
align="center"><b>'.$row['descripcion'].'</b></div>';
              @$descripcion_con = $row['descripcion'];
              } while (@$row = mysql_fetch_array($result));
        }
        else {
        echo '<div align="center"><b>No se encontro ningun
contrato</b></div>';
        }
        mysql_close($conexion);
?>
  </p>

  <?php
        $link = conexion();
        $sql = "SELECT cam.cod_campo, cam.descripcion, tamano, negrita ";
        $sql .= "FROM tbl_campos cam, tbl_contratos con ";
        $sql .= "WHERE con.cod_contrato = cam.cod_contrato ";
        $sql .= "AND con.cod_contrato = $cod_contrato ";
        $sql .= "ORDER BY cam.orden ";
        @$result = mysql_query($sql, $link);
        @$total_rows = mysql_num_rows($result);
        if (@$row = mysql_fetch_array($result)) {
```

```
            do {
            $salto = '<br>';
      @$mes = dame_nombre_mes(date(substr(@$f_inicio,5,2), time())." ";
      @$f_inicio_2 = date(substr(@$f_inicio,8,2), time())." de ".$mes."
      del ".date(substr(@$f_inicio,0,4), time());
      @$mes = dame_nombre_mes(date(substr(@$f_termina,5,2), time()))."
";
      @$f_termina_2 = date(substr(@$f_termina,8,2), time())." de
      ".$mes." del ".date(substr(@$f_termina,0,4), time());
      $resultado = convertir($numero);
      $sueldo = '<b>'.$resultado.' DOLARES '.'con '.$centavos.'/100
</b>';
      if ($sexo == 'Masculino') {
        $abre_titulo_aux = ' el '.$abre_titulo.' '.$abre_posgrado.'
      '.$abre_pregrado;
      $empleado = 'el empleado';
      $docente = 'el docente';
      $prof_may = 'El PROFESOR';
      $profesor = 'el profesor';
      $doc_may = 'EL DOCENTE';
      $emp_may = 'EL EMPLEADO';
      $profesional = 'el profesional';
      $profesi_may = 'EL PROFESIONAL';
      $lenguas = 'profesor';
      } else {
            $abre_titulo_aux = ' la '.$abre_titulo.' '.$abre_posgrado.'
            '.$abre_pregrado;
            $empleado = 'la empleada';
            $docente = 'la docente';
            $prof_may = 'LA PROFESORA';
            $profesor = 'la profesora';
            $doc_may = 'LA DOCENTE';
            $emp_may = 'LA EMPLEADA';
            $profesional = 'la profesional';
            $profesi_may = 'LA PROFESIONAL';
            $lenguas = 'profesora';
      }
            $texto = $row['descripcion'];
            @$texto = str_replace('CL02_TITULO', $abre_titulo_aux,
            $texto);
            @$texto = str_replace('CL03_APELLIDOS', $apellidos, $texto);
            @$texto = str_replace('CL04_NOMBRES', $nombres, $texto);
            @$texto = str_replace('CL05_CEDULA', $cedula, $texto);
            @$texto = str_replace('CL06_FECHA_INICIO', $f_inicio_2,
            $texto);
            @$texto = str_replace('CL07_ESCUELA', $descripcion, $texto);
            @$texto = str_replace('CL08_SUELDO', $sueldo, $texto);
            @$texto = str_replace('CL09_CARGO', $descripcion_c, $texto);
            @$texto = str_replace('CL10_SEMESTRE', $semestre, $texto);
            @$texto = str_replace('CL11_F_TERMINA', $f_termina_2,
            $texto);
            @$texto = str_replace('CL12_NUM_HORAS', $num_horas, $texto);
            @$texto = str_replace('CL13_EMPLEADO', $empleado, $texto);
            @$texto = str_replace('CL14_DOCENTE', $docente, $texto);
            @$texto = str_replace('CL15_LENGUAS', $lenguass, $texto);
            @$texto = str_replace('CL16_EMP_MAY', $emp_may, $texto);
            @$texto = str_replace('CL17_DOC_MAY', $doc_may, $texto);
            @$texto = str_replace('CL23_LEN_MAY', $lenguas, $texto);
            @$texto = str_replace('CL19_PROFESOR', $profesor, $texto);
            @$texto = str_replace('CL20_PROF_MAY', $prof_may, $texto);
            @$texto = str_replace('CL21_SALTO', $salto, $texto);
            @$texto = str_replace('CL15_PROFESIONAL', $profesional,
            $texto);
            @$texto = str_replace('CL22_PROFESI_MAY', $profesi_may,
            $texto);

      if ($row['negrita'] == 'S') {$neg = '<b>'; $neg_c = '</b>';} else
      {$neg = ''; $neg_c = '';}
```

```
        echo '<p><div align="justify">'.@$neg.$texto.@$neg_c.'</div></p>';
        } while ($row = mysql_fetch_array($result));
        } else {
              echo '<b>No se encontro ningun contrato</b>';
        }
        mysql_close($link);
?>
  <div align="center"></div>
  <table width="75%" border="0" align="center" cellspacing="0">
    <tr>
      <td width="51%"><div align="center"><strong><br>
          LA PUCESA <br>
          </strong></div></td>
      <td width="49%"><div align="center"><strong>

          <?php
        if ($profesion == 'profesional' and $sexo == 'Masculino' ) {
             echo "EL PROFESIONAL";}
        if ($profesion == 'profesional' and $sexo == 'Femenino' ) {
             echo "LA PROFESIONAL";}
        if ($profesion == 'profesor' and $sexo == 'Masculino' ) {
             echo "EL PROFESOR";}
        if ($profesion == 'profesor' and $sexo == 'Femenino' ) {
             echo "LA PROFESORA";}
        if ($profesion == 'empleado' and $sexo == 'Masculino' ) {
             echo "EL EMPPLEADO";}
        if ($profesion == 'empleado' and $sexo == 'Femenino' ) {
              echo "LA EMPLEADA";}
        if ($profesion == 'docente' and $sexo == 'Masculino' ) {
             echo "EL DOCENTE";}
        if ($profesion == 'docente' and $sexo == 'Femenino' ) {
             echo "LA DOCENTE";}
        ?>
          </strong></div></td>
    </tr>
    <tr>
      <td><div align="center">
          <p> </p>
          <p> </p>
          <p> </p>
          <p><strong>P. Dr. C&eacute;sar Gonz&aacute;lez Loor<br>
            PRORRECTOR </strong></p>
        </div></td>

      <td><p> </p>
       <p align="center"> </p>
       <p align="center"> </p>
       <p align="center"><strong><font size="3" face="Times New Roman,
       Times, serif"><strong><?php echo @$abre_titulo;?></strong></font>
       <font size="3" face="Times New Roman, Times, serif"><strong><?php
       echo strtoupper(@$nombres." ".@$apellidos)?></strong></font><br>
       C.C: <?php echo @$cedula;?></strong></p></td>
    </tr>
    <tr>

      <td height="97" colspan="2">
        <div align="center">
<p> </p>
          <p> </p>
          <p> </p>

              <?php
       if (@$descripcion_con <> "CONTRATO DE PRESTACIÓN DE SERVICIOS POR
       HONORARIOS PROFESIONALES"){
              echo "<b>INSPECTOR DEL TRABAJO</b>";
              }else{
              echo "";
              }
```

```
            ?>

           </p> </div></td>
    </tr>
  </table>
  <p> </p>
</form>
</body>
</html>
```

3.4.2.3. Código Fuente de Página de Reporte de Datos de Empleados.

En este código fuente podemos ver el proceso del reporte para sacar cada uno de los datos de los empleados conforme a la necesidad del usuario.

```
<?php session_start(); ?>
<?php ob_start(); ?>
<?php
if (@$HTTP_SESSION_VARS["status"] <> "login") {
            header("Location:  frm_login.php");
            exit();
      }
?>
<?php
require ("funcion_conectar.php");
?>

<html>
<head>
<title>:: Pontificia Universidad Cat&oacute;lica Del Ecuador Sede Ambato
::</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body>
<form name="form1" method="post" action="">
   <p> 
   <?php echo'<div
align="center"><b><br>'.@$encabezado1.'</b></br></div>';
    echo'<div align="center"><b><br>'.@$encabezado2.'</b></br></div>';
  ?>

  </p>
  <p> </p>

  <?php

       echo "<table>";
       echo "<tr>";
  if (isset ($cedula)) { echo '<td>Cédula</td>'; }
  if (isset ($nombre)) { echo '<td>Nombres</td>'; }
  if (isset ($apellido)) { echo '<td>Apellidos</td>'; }
  if (isset ($f_nacimiento)) { echo '<td>Fecha de Nacimiento</td>'; }
  if (isset ($nacionalidad)) { echo '<td>Nacionalidad</td>'; }
  if (isset ($direccion)) { echo '<td>Dirección</td>'; }
  if (isset ($telefono)) { echo '<td>Teléfono</td>'; }
  if (isset ($email)) { echo '<td>E-mail</td>'; }
  if (isset ($titulo)) { echo '<td>Titulo</td>'; }
  if (isset ($nivel_titulo)) { echo '<td>Nivel de Titulo</td>'; }
  if (isset ($abre_titulo)) { echo '<td>abreviación del Titulo</td>'; }
```

```
  if (isset ($ano_obtenido)) { echo '<td>Año Obtenido</td>'; }
  if (isset ($universidad)) { echo '<td>Universidad</td>'; }
  if (isset ($sexo)) { echo '<td>Sexo</td>'; }
  if (isset ($estado_civil)) { echo '<td>Estado Civil</td>'; }
  if (isset ($tipo_sangre)) { echo '<td>Tipo de Sangre</td>'; }
  if (isset ($activo)) { echo '<td>Activo</td>'; }
  if (isset ($parroquia_sector)) { echo '<td>Parroquia</td>'; }
  if (isset ($codigo_sectorial)) { echo '<td>Código Sectorial</td>'; }
  if (isset ($num_registro)) { echo '<td>Número de Registro</td>'; }
  if (isset ($num_gremio)) { echo '<td>Número de Gremio</td>'; }
  if (isset ($canton)) { echo '<td>Cantón </td>'; }
  if (isset ($provincia)) { echo '<td>Provincia</td>'; }
  if (isset ($num_registro)) { echo '<td>Número de Registro</td>'; }
  if (isset ($doc_faltante)) { echo '<td>Documento Faltante</td>'; }
  if (isset ($num_afiliacion)) { echo '<td>Número de Afiliación</td>'; }
  if (isset ($h_a_t_c)) { echo '<td>h_a_t_c</td>'; }
  if (isset ($h_a_m_t)) { echo '<td>h_a_m_t</td>'; }
  if (isset ($extension)) { echo '<td>Extensión</td>'; }
  if (isset ($lugar)) { echo '<td>Lugar</td>'; }
  if (isset ($sueldo_adm)) { echo '<td>Sueldo Adm.</td>'; }
  if (isset ($antiguedad)) { echo '<td>Antiguedad</td>'; }
  if (isset ($responsable)) { echo '<td>Responsable</td>'; }
  if (isset ($representante)) { echo '<td>Representante</td>'; }
  if (isset ($total_remuneracion)) { echo '<td>Ttotal</td>'; }
  if (isset ($abre_posgrado)) { echo '<td>Abrevioación Posgrado</td>'; }
  if (isset ($abre_pregrado)) { echo '<td>Abrevioación Pregrado</td>'; }
        '</b></div>';
             echo "</tr>";
?>
  <p>
  <?php

$conexion = conexion();
$sql = "SELECT * ";
$sql .= "FROM tbl_empleados ";
if ($escuela <> "todos") {$sql .= "WHERE escuela = '$escuela' ";}
@$result = mysql_query($sql, $conexion);
if (@$row = mysql_fetch_array($result)) {

do {

echo "<tr>";
 if (isset ($cedula)) { echo "<td>".$row['cedula'].'</td>'; }
 if (isset ($nombre)) { echo "<td>".$row['nombres'].'</td>'; }
 if (isset ($apellido)) { echo "<td>".$row['apellidos'].'</td>'; }
if (isset ($f_nacimiento)) { echo
"<td>".$row['fecha_nacimiento'].'</td>'; }
if (isset ($nacionalidad)){ echo "<td>".$row['nacionalidad'].'</td>'; }
if (isset ($direccion)){ echo "<td>".$row['direccion'].'</td>'; }
if (isset ($telefono)) { echo "<td>".$row['telefono'].'</td>'; }
if (isset ($email)) { echo "<td>".$row['email'].'</td>'; }
if (isset ($titulo)) { echo "<td>".$row['titulo'].'</td>'; }
if (isset ($nivel_titulo)) { echo "<td>".$row['nivel_titulo'].'</td>'; }
if (isset ($abre_titulo)) { echo "<td>".$row['abre_titulo'].'</td>'; }
if (isset ($ano_obtenido)) { echo "<td>".$row['ano_obtenido'].'</td>'; }
if (isset ($universidad)) { echo "<td>".$row['universidad'].'</td>'; }
if (isset ($sexo)) { echo "<td>".$row['sexo'].'</td>'; }
if (isset ($estado_civil)) { echo "<td>".$row['estado_civil'].'</td>'; }
if (isset ($tipo_sangre)) { echo "<td>".$row['tipo_sangre'].'</td>'; }
if (isset ($activo)) { echo "<td>".$row['activo'].'</td>'; }
if (isset ($parroquia_sector)) { echo
"<td>".$row['parroquia_sector'].'</td>'; }
if (isset ($codigo_sectorial)) { echo
"<td>".$row['codigo_sectorial'].'</td>'; }
if (isset ($num_registro)) { echo "<td>".$row['num_registro'].'</td>'; }
if (isset ($num_gremio)) { echo "<td>".$row['num_gremio'].'</td>'; }
if (isset ($canton)) { echo "<td>".$row['canton'].'</td>'; }
if (isset ($provincia)) { echo "<td>".$row['provincia'].'</td>'; }
```

```
if (isset ($num_registro)) { echo "<td>".$row['num_registro'].'</td>'; }
if (isset ($doc_faltante)) { echo "<td>".$row['doc_faltante'].'</td>'; }
if (isset ($num_afiliacion)) { echo
"<td>".$row['num_afiliacion'].'</td>'; }
if (isset ($h_a_t_c)) { echo "<td>".$row['h_a_t_c'].'</td>'; }
if (isset ($h_a_m_t)) { echo "<td>".$row['h_a_m_t'].'</td>'; }
if (isset ($extension)) { echo "<td>".$row['extension'].'</td>'; }
if (isset ($lugar)) { echo "<td>".$row['lugar'].'</td>'; }
if (isset ($sueldo_adm)) { echo "<td>".$row['sueldo_adm'].'</td>'; }
if (isset ($antiguedad)) { echo "<td>".$row['antiguedad'].'</td>'; }
if (isset ($responsable)) { echo "<td>".$row['responsable'].'</td>'; }
if (isset ($representante)) { echo "<td>".$row['representante'].'</td>';
}
if (isset ($total_remuneracion)) { echo
"<td>".$row['total_remunracion'].'</td>'; }
if (isset ($abre_posgrado)) { echo "<td>".$row['abre_pregrado'].'</td>';
}
  echo '</tr>';
} while (@$row = mysql_fetch_array($result));

echo "</table>";
}
else {

echo '<div align="center"><b>No se encontro ningun empleado</b></div>';
      }
mysql_close($conexion);
?>

  </p>
  <p> </p>
  <p> 

  <?php echo'<div align="center"><b>'.@$pie1.'</b></div>';
  echo'<div align="center"><b>'.@$pie2.'</b></div>';

  ?>

  </p>
  <p> </p>
</form>
</body>
</html>
```

3.5. Pruebas e Instalación.

3.5.1. Pruebas.

Se realizaron todas las pruebas correspondientes de forma local antes de poner en marcha el sistema para su funcionamiento.

Se realizó la emisión de contratos, certificados y reportes en presencia de la directora de la Dirección Administrativa de la PUCESA.

3.5.2. Instalación.

El disco de instalación del sistema SAGA contiene:

- Un directorio con los archivos del sitio web llamado "saga".
- Un archivo de extensión .dump donde se encuentra la base de datos llamada "saga.dump".
- Programa de instalación MySQL 4.1.14 (Para S.O. Solaris y Windows).
- Programa de instalación PHP 4.2.2 (Para S.O. Solaris y Windows).

Pasos para la Instalación del Sistema

- Como primer paso, Se procede con la instalación del compilador de PHP obligatorio que el sistema SAGA necesita. Instalamos la versión de PHP 4.2.2 con el programa de instalación que contiene dicho programa (si en el sistema operativo previamente no se encuentra instalado o existe una versión anterior).

  Luego, modificamos el archivo "php.ini" que se instaló en el sistema operativo, con la finalidad de permitir el paso de variables entre formularios; seteamos en **"On"** la opción **"register_globals"**, y grabamos la modificación.

```
Register_globals = On
```

- Ahora instalamos el motor de base de datos MySQL. Instalamos la versión de MySQL 4.1.14 con el programa de instalación del mismo (si en el sistema operativo previamente no se encuentra instalado o existe una versión anterior).
- Grabamos el sitio web que contiene los archivos del sistema en el directorio respectivo:

  - Windows . "c:/inetpub/wwwroot/saga".
  - Solaris. "/home/usuario/saga".

- Para cargar la base de datos del sistema es necesario copiar el archivo "saga.dump" en el directorio bin contenido en el directorio de MySQL.

  - Windows. "c:/mysql/bin".
  - Solaris. "/user local/mysql/bin".

Por ultimo ingresamos a MySQL, creamos la base de datos con el nombre "saga" y cargamos el archivo "saga.dump".

Shell> mysql
Mysql> **CREATE DATABASE** saga;
Mysql>**USE** saga;
Mysql>**SOURCE** saga.dump;

NOTA: El nombre de la base de datos debe ser idéntico al que se indica, caso contrario se tendrá problemas con la conexión de la base de datos.

4. CAPITULO IV. Verificación y Validación de Resultados.

4.1. Verificación.

La hipótesis planteada fue que con el Sistema de Gestión Administrativa se obtendrá mayor rapidez en el acceso a la información de la documentación en la Dirección Administrativa de la PUCESA.

En el proceso de emisión de contratos en el inicio del primer semestre 2006-2007 mediante el sistema de Gestión Administrativa, se comprobó la hipótesis planteada llegando a cumplir con los objetivos propuestos anteriormente.

El sistema reveló un ahorro de tiempo considerable, sin el Sistema de Gestión Administrativa se tardaba un mes aproximadamente para la emisión de los contratos, con el Sistema de Gestión administrativa en funcionamiento se emitieron los contratos en tres semanas en tiempo real cada contrato emitido.

La emisión de contratos con el sistema de Gestión Administrativa ayuda al mejor desempeño del personal que labora en la Dirección Administrativa de la PUCESA.

4.2. Validación .

CERTIFICACIÓN

INGENIERA ROSA PALACIOS, Directora Administrativa de la Pontificia Universidad Católica del Ecuador Sede Ambato.

CERTIFICA QUE:

La tesis de grado: "Desarrollo e Implementación de un Sistema de Gestión Administrativa, que Simplifique y Estandarice los Procesos de Despacho de Documentos en la Dirección Administrativa de la PUCESA", elaborada por el Sr. Bladimir Patricio Garcés Llerena, con C.C. 180353795-8, egresado de la Escuela de Ingeniería en Sistemas de la PUCESA, está en funcionamiento y cumple con el objetivo de facilitar la generación automática de contratos, reportes y certificados que se emiten en la Dirección Administrativa.

Es todo cuanto puedo certificar en honor a la verdad, autorizando al interesado hacer uso de este documento, el uso que estimare conveniente.

Ambato, 19 de diciembre del 2006

Ing. Rosa Palacios Gavilanes
Directora Administrativa

DIRECCION ADMINISTRATIVA

PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR
SEDE AMBATO
DIRECCIÓN ADMINISTRATIVA
Av. Manuelita Sáenz s/n
Sector El Tropezón
Apartado Postal No.18-01662
Telf: 593 3 2 414 604 ext. 107-1
Fax: 593 3 2 411 868 ext.102
rosy@pucesa.edu.ec
josue@pucesa.edu.ec
Ambato - Ecuador
www.pucesa.edu.ec

CERTIFICACIÓN

INGENIERO DIEGO SANTACRUZ, Administrador del Centro de Computo de la Pontificia Universidad Católica del Ecuador Sede Ambato.

CERTIFICA QUE:

La tesis de grado: **"Desarrollo e Implementación de un Sistema de Gestión Administrativa, que Simplifique y Estandarice los Procesos de Despacho de Documentos en la Dirección Administrativa de la PUCESA "**, elaborada por el **Sr. Bladimir patricio Garcés Llerena**, con C.C. 180353795-8 egresado de la Escuela de Ingeniería en Sistema de la PUCESA, esta en funcionamiento y cumple con el objetivo de facilitar la generación automática de contratos, reportes y certificados que se emiten en la dirección Administrativa.

Es todo en cuanto puedo certificar en honor a la verdad, autorizando al interesado hacer uso de este documento, el uso que estimare conveniente.

Ambato, 19 de diciembre del 2006

PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR
SEDE AMBATO
CENTRO DE COMPUTO

Ing. Diego Santacruz
Administrador del Centro de Computo

Av. Manuelita Sáenz s/n
Sector El Tropezón
Apartado Postal No.18-01-66
Telef: 593 3 2416 220
Telefax: 593 3 2411 868 ext.
webmaster@pucesa.edu.ec
Ambato - Ecuador
www.pucesa.edu.ec

4.3. Conclusiones.

- La utilización de nuevas herramientas de programación es de mucha utilidad para transformar las ideas en realidad, y para poder facilitar la vida del hombre en su trabajo diario.

- Se logró la automatización del proceso de emisión de contratos, en tiempo real, para un mejor desempeño del personal que labora en la Dirección Administrativa

- Existe la facilidad de manejo por parte del personal que labora en la Dirección Administrativa para el mantenimiento del sistema SAGA.

- Se obtiene un ahorro de tiempo en el proceso de emisión de contratos, el mismo que se realizaban en forma manual.

- Buscando favorecer al desarrollo de la Dirección Administrativa de la PUCESA se consideró al Sistema de Gestión Administrativa una herramienta cualitativa y valiosa en el proceso de emisión de contratos.

- Las herramientas MySQL y PHP son muy accesibles por ser gratuitos y de código abierto, además que son de fácil comprensión y de gran ayuda para la programación de páginas Web por ser multiplataforma.

4.4. Recomendaciones.

- Para un buen rendimiento del Sistema de Gestión Administrativa es necesario ingresar correctamente los datos requeridos para la emisión del los contratos.

- Para los usuarios administradores es importante revisar el manual del usuario para poder dar mantenimiento de forma adecuada a la base de datos.

- Es necesario que el administrador del sitio web y la base de datos del sistema revise el manual técnico, diccionario de datos, mapa del sitio y el proceso de instalación que este documento contiene con el propósito de mantener la integridad del sistema y poder dar mantenimiento en caso de ser necesario.

- En el proceso del montaje de la base de datos es de suma importancia mantener el nombre de la misma exactamente igual al que indica el proceso de instalación para evitar fallos en la conexión a la base de datos.

- Es necesario sacar un respaldo de la base de datos cada seis meses.

- Se recomienda realizar la compra de un nuevo servidor, debido a que ya no existen repuestos para el servidor de la universidad por ser muy antiguo.

- Se recomienda realizar la ampliación del sistema para otras dependencias de la universidad.

- Para instalar una nueva versión de sistema "SAGA" o una nueva base de datos, es necesario que todos los componentes, librerías o módulos sean soportados por el servidor web del sistema.

BIBLIOGRAFIA

LIBROS

- ANSIÓN, Juan (1998) *Educación: la mejor herencia.* Lima: Fondo Editorial de la Pontificia Universidad Católica del Perú.
- Eduardo Ramírez, *Aplicado (2001): "SQL Server 2000". Lima –Perú*
- Eduardo Ramírez, *Aplicado(2001): "Dreamweaver 4.0" Lima-Perú*
- Welling, Luke ; Thomson, Laura, *Marzo del 2005 "Desarrollo Web con PHP y MYSQL"*
- Carlos Zumbado Rodríguez; Daniel de la Cruz Heras, Noviembre 2003 "Flash, PHP y MYSQL Conceptos Dinámicos "
- Braude, Eric: *"Ingenieria de Software" Febrero 2005.*

PAGINAS WEB

http://dev.mysql.com/doc/refman/5.0/en/php.html : *Mysql PHP API , 25 de noviembre del 2005.*

http://dev.mysql.com/doc/refman/5.0/es/privilege-system.html : *el sistema de privilegis de acceso a Mysql.*

http://www.webestilo.com/php/codigo.phtml : *biblioteca del código fuente de php, mayo 15.*

http://www.webestilo.com/mysql/ : *manual de Mysql, 15 de marzo 2006.*

http://www.php.net : *la casa de php, 10 de marzo del 2006.*

http://www.php.net/manual/es/ : *manual de php, 03 de marzo del 2006.*

http://www.webestilo.com/php/articulo.phtml?art=27: *php orientado a objetos, 28 de febrero 2006.*

http://www.desarrolloweb.com/directorio/programacion/php/manuales_de_php/ : *manuales de php, 16 febrero 2006.*

http://www.desarrolloweb.com/php/: *manual de PHP, 19 enero 2006.*

http://www.desarrolloweb.com/manuales/12/: *programación en PHP, 19 de enero 2006.*

ANEXOS.

ANEXO A. Manual Técnico.

Ambito y Funciones.

AMBITO.

El Sistema de Gestión Administrativa esta dirigido a la PUCESA y tomará el nombre de SAGA.

PROBEMAS A SOLUCIONAR.

- Automatizar el proceso de emisión de contratos en la universidad.
- Emitir los contratos en tiempo real.
- Ofrecerá la facilidad de manejo y mantenimiento para los usuarios.

FUNCIONES.

Creación de Títulos.

El administrador del sistema tendrá que crear o ingresar los títulos de los contratos que desee almacenar en el sistema.

Asignación de Cláusulas a los Contratos.

Es la agrupación de rutinas y procedimientos que hacen posible que el proceso automatizado de Gestión Administrativa por el SAGA se realice correctamente.

En el proceso de asignación de cláusulas a los contratos participan varios actores principales los cuales son el titulo y las diferentes cláusulas de los contratos, también se encuentran los administradores los cuales van a revisar los resultados ingresados al sistema para su perfecto funcionamiento.

Para la asignación de cláusulas a un contrato primero seleccionamos el titulo del contrato al cual vamos a ingresar las cláusulas, con sus respectivas palabras claves dentro del texto de cada cláusula.

Emisión de Contratos.

SAGA emite el contrato que el administrador elija al momento de su generación, además de elegir el contrato, es necesario llenar los datos que requiere el sistema dependiendo del tipo de contrato para la emisión del mismo.

Emisión de Certificados.

SAGA emite el certificado que el administrador elija al momento de su generación, además es necesario llenar los datos que requiere el sistema dependiendo del tipo de certificado para la emisión del mismo.

Generación de Reportes.

SAGA proporciona al administrador reportes de los docentes y administrativos que se encuentran almacenados en la base de datos del sistema, el reporte es generado dependiendo de la necesidad del administrador.

Control del Sistema.

El sistema controla automáticamente los datos que son requeridos para cada tipo de contrato o certificado, si los datos no son ingresados correctamente el sistema dará error al momento de generar el contrato o certificado.

Control de Usuario.

El sistema posee el usuario administrador el cual tiene acceso a todas las funciones del sistema, además tiene la capacidad de dar mantenimiento a la base

de datos del sistema, procesos de administración y emisión de los contratos o certificados.

Mantenimiento de la Base de Datos.

Esta función agrupa varias tareas de mantenimiento de los datos que son necesarios para el sistema.

SRS.

1. INTRODUCCION.

1.1. OBJETIVO.

1.1.1. PROPOSITO DE DOCUMENTO

Este documento tiene la misión de dar a conocer los alcances y delimitaciones de sistema de Gestión Administrativa tiene al ser implementado en la universidad, incluye datos técnicos para el administrador del front end y back end del sistema (instalación del sistema y base de datos).

1.1.2. AUDIENCIA A LA QUE VA DIRIGIDO.

La administración del Centro de Computo de la universidad se encarga del hospedaje del sitio web de la PUCESA, por lo tanto, el presente manual técnico va dirigido especialmente a los encargados de dar mantenimiento y hospedaje al sistema.

1.2. ALCANCE.

1.2.1. IDENTIFICACION DEL PRODUCTO MEDIANTE UN NOMBRE

El sistema de Gestión Administrativa para el presente documento se le denominara SAGA.

1.2.2. QUE HACE Y QUE NO HACE EL PRODUCTO.

Sistematizará el proceso de Gestión Administrativa de la PUCESA. SAGA brinda las posibilidades de mantener un integridad de datos en la creación, modificación y actualización de la base de datos.

El sistema no realiza el respaldo de la base de datos de forma automática ya que este proceso es necesario ejecutarla de forma manual por el administrador encargado del hospedaje del sistema.

1.2.3. APLICACIONES DEL SOFTWARE: BENEFICIOS OBJETIVOS Y METAS.

Los objetivos y beneficios que presenta el sistema básicamente son de automatizar y sistematizar los procesos de gestión administrativa. Además de presentar los contratos y certificados en tiempo real.

1.3. DEFINICION, ACRONIMOS Y ABREVIATURAS.

SAGA = Sistema de Automático de Gestión Administrativa.

1.4. VISION GENERAL.

1.4.1. DESCRIPCION DEL CONTENIDO DEL RESTO DEL DOCUMENTO.

Este documento estará enfocado para el correcto funcionamiento del sistema .

1.4.2. ORGANIZACION DEL DOCUMENTO.

Estas especificaciones de requisitos software se organizan según recomendaciones del estándar [IEEE 93].

Ver Anexo C.

2. DESCRIPCION GENERAL.

2.1. PERSPECTIVA DEL DOCUMNTO.

2.1.1. INDICAR SI ES UN PRODUCTO INDEPENDIENTE O PARTE DE UN SISTEMA MAYOR.

Software dependiente de un compilador de código PHP, motor de base de datos MySQL, módulos o librerías para conexión PHP con el motor de base de datos.

2.1.2. INTERFASES DEL SISTEMA.

Interfase web interpretada por cualquier browser del lado del cliente.

2.1.3. LIMITACIONES DE MEMORIA.

Interfase. La velocidad de presentación de las pantallas del sistema depende del tipo de conexión de Internet que el cliente tenga.

Base de Datos. MySQL es un motor de base de datos pequeño, optimiza los recursos de memoria.

2.1.4. OPERACIONES.

2.1.4.1. MODO DE OPERACION DEL USUARIO ADMINISTRADOR.

Administrador. Es el encargado del mantenimiento de la base de datos del sistema, además del acceso a todas las opciones del sistema.

2.1.4.2. FUNCIONES RESPALDO DEL PROCESAMIENTO DE DATOS.

El administrador del hospedaje del sistema es el encargado de realizar las funciones de respaldo de datos de forma manual.

2.1.4.3. OPERACIONES DE BACKUP Y RECUPERACIÓN.

Interfase. La totalidad de los archivos que comprende SAGA deben hospedarse en el servidor web. En Windows copiamos toda la carpeta **"saga"** dentro del directorio **"wwwroot".** En Solaris es necesario copiarla en el directorio **"var/apache".** Para obtener un respaldo del sitio solo en necesario copiar toda la carpeta que contiene los archivos del sistema.

Base de Datos. Para obtener un respaldo de la base de datos necesariamente se lo debe realizar de forma manual mediante consola ejecutando una función propia de MySQL, y respaldándolo en un archivo **.dump.**

```
mysql>exit;
shell > bye
shell> mysqldump saga > saga.dump
```

Para recuperar una base de datos es necesario copiar el archivo de respaldo **"saga"** dentro del directorio raíz donde se encuentra MySQL, luego manualmente dentro del motor de base de dato, crear primero la base de datos y cargar luego las tabas y datos de la base con la instrucción **"source"**.

```
shell> mysql
mysql> CREATE DATABASE saga;
mysql> USE saga;
mysql> SOURCE  saga.dump;
```

2.1.5. REQUERIMIENTOS PARA ADAPTARCE A LA UBICACION.

2.1.5.1. DATOS O SECUENCIAS DE INICIALIZACION ESPECIFICOS DE CUALQUIER LUGAR, MODO DE OPERACION.

- Instalación de MySQL (versiones que soporten tablas InnoDB versión 4.1.14 o superiores).
- Instalación de PHP versión 4.2.2 o superiores.
- Windows configuración de Internet Information Server (IIS) de Windows versión 4 o superiores.
- Solaris instalación del apache y módulos de conexión de PHP con MySQL.
- Configuración del archivo php.ini para permitir al compilador de PHP el paso de variables a través de formularios.

2.1.5.2. CARACTERISTICAS QUE DEBEN SER MODIFICADAS PARA UNA INSTALACION EN PARTICULAR.

Dependiendo del sitio donde se hospeda la base de datos, los parámetros que utiliza el sistema la conexión con la base podrían cambiar. El sistema utiliza la clase conexión programada en PHP para realizar esta función, la clase posee la función de conexión *mysql_connect()* que abre una conexión a un servidor MySQL. Esta función se encuentra definida de forma local, por nombre del servidor localhost y nombre del usuario root por defecto.

```
Mysql_connect ("localhost", "root");
```

Si es necesario cambiar estos valores, la estructura de función es;

```
Mysql_connect([string server],[string username],[string
password]);
```

Las página web que debería ser cambiada es ***funcion_conectar.php***

2.2. FUNCIONES DEL PRODUCTO.

- Creación de contratos
- Creación de certificados
- Generación de reportes
- Control del sistema
- Control de usuario
- Mantenimiento de base de datos.

2.3. CARACTERISTICAS DEL USUARIO.

El administrador debe tener conocimientos de manejo de control y administración del sistema.

2.4. RESTRICCIONES.

- El sistema realizará únicamente emisión de contratos, certificados, y reportes.

2.5. SUPOSICIONES Y DEPENDENCIAS.

Depende del correcto funcionamiento del servidor web, instalación de los compiladores básicos para la funcionalidad del sistema y el motor de base de datos.

2.6. REQUISITOS PARA FUTURAS VERSIONES DEL SISTEMA.

Si se desea instalar una nueva versión del sistema o nueva base de datos, es necesario que todos los componentes, librerías o módulos sean soportados por el servidor web del sistema.

3. REQUISITOS ESPECIFICOS.

3.1. REQUISITOS DE INTERFASE EXTERNO.

3.1.1. INTERFASES DE USUARIO.

La interfase del usuario es una interfase web que cualquier browser interpreta. Esta consta de formularios, menú del usuario, navegador interno del sistema y pantalla de ayuda de todas las funciones para el usuario administrador.

3.1.2. INTERFASE DE SOFTWARE.

PHP se ejecuta y conecta del lado del servidor con MySQL.

3.2. REQUISITOS FUNCIONALES.

Windows.

- PHP versión 4.2.2
- MySQL versión 4.1.14
- Internet Information Server versión 4 o superiores.

Solaris.

- PHP versión 4.2.2
- MySQL versión 4.1.14
- Apache.

3.2.1. FLUJOS DE INFORMACIÓN.

3.2.1.1. DIAGRAMA DE FLUJO DE DATOS NIVEL 0

Ver diagrama de flujo de datos nivel 0 del capitulo 3 del desarrollo del Proyecto.

3.2.1.2. DIAGRAMA DE FLUJO DE DATOS NIVEL 1.

Ver diagrama de flujo de datos nivel 1 del capitulo 3 del desarrollo del Proyecto.

3.2.1.3. DIAGRAMA DE FUJO DE DATOS NIVEL 2.

Ver diagrama de flujo de datos nivel 1 del capitulo 3 del desarrollo del Proyecto.

3.3. REQUISITOS DE RENDIMIENTO.

Espacio en disco para la base de datos de 10 Gb.

3.4. RESTRICCIONES DEL DISEÑO.

Se encuentra optimizado para una resolución de pantalla de 800 * 600.

ANEXO B. Manual de Usuario

Acceso al Sistema

Al momento que usted ingrese al Sistema Automático de Gestión Administrativa "SAGA", le aparecerá la pantalla de bienvenida que es la que continuación le presentamos:

A continuación tenemos que ingresar el usuario y contraseña para que el sistema pueda validar los campos para el acceso.
Usted puede crear su propio usuario y contraseña a parte de administrador.

Pantalla de Bienvenida

Una ves que el Sistema haya validado su usuario podrá tener acceso al sistema y podrá ver la siguiente pantalla:

En la pantalla de bienvenida usted ya tiene acceso al menú del sistema y va a poder empezar a trabajar.

Empezar a Trabajar con SAGA

Para empezar a trabajar en el sistema ¨SAGA¨ , tenemos que acceder al menú que contiene todos los procesos, reportes y mantenimiento.

Tipos de Contrato

Para obtener el reporte de los diferentes tipos de contrato vamos a acceder al menú mantenimiento y escogemos en el menú "tipo de contrato", y nos aparecerá la siguiente ventana:

Modificar los Tipos de Contrato

Para modificar cualquier tipo de contrato nos posicionamos con el ratón o mouse del computador sobre el texto del contrato y damos un clic, y nos aparecerá la siguiente ventana :

Una ves modificado el tipo de contrato presionamos en botón enviar y se grabaran los cambios realizados en dicho campo.

Insertar un Nuevo Tipo de Contrato

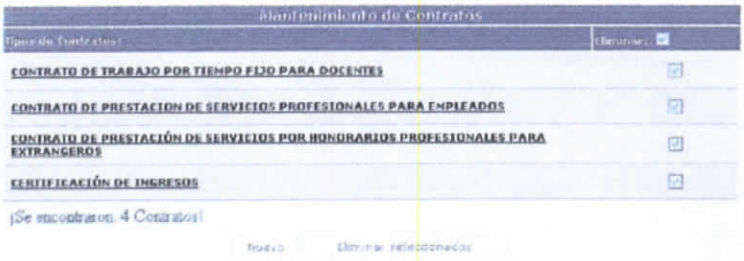
Para insertar un nuevo tipo de contrato en nuestra base de datos damos un clic en el botón nuevo del reporte de los tipos de contrato y nos aparecerá la siguiente ventana:

Para el ingreso del tipo de contrato tenemos dos campos, el campo llamado titulo va a identificar al tipo de contrato, y el campo denominado descripción se va a imprimir en el contrato debemos tener muy en cuenta esta pequeña aclaración.

Una ves digitado el texto en los dos campos damos un clic en el botón enviar para que se grabe lo ingresado en la base de datos.

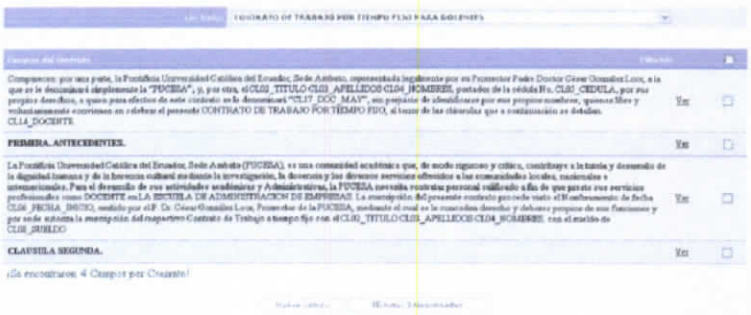
Eliminar los Tipos de Contrato

Para eliminar los diferentes tipos de contrato podemos hacerlo marcándolos uno a uno o por medio del primer check box para seleccionar todos los tipos de contratos y luego presionamos eliminar seleccionados.



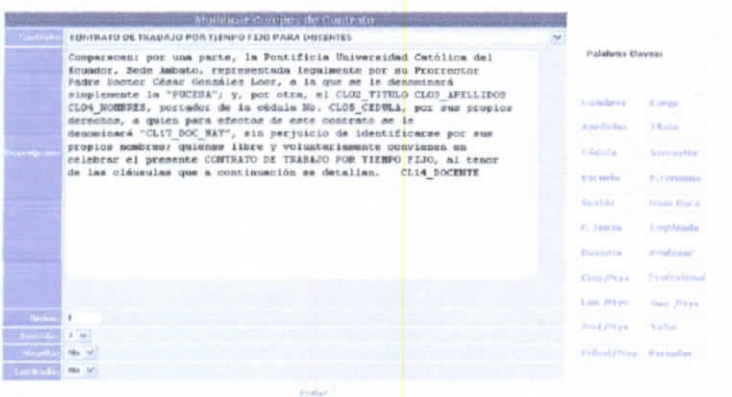
Cláusulas de los Contratos

Si se desea ver las cláusulas de cada uno de los tipos de contratos existentes lo haremos por medio de menú mantenimiento y seleccionamos el submenú "Contratos" para que se visualice el reporte de los contratos y sus cláusulas.



Modificar las Cláusulas de un Contrato

Para modificar una cláusula de un determinado tipo de contrato lo haremos dando un clic en el vínculo ver y se presentará la siguiente ventana:

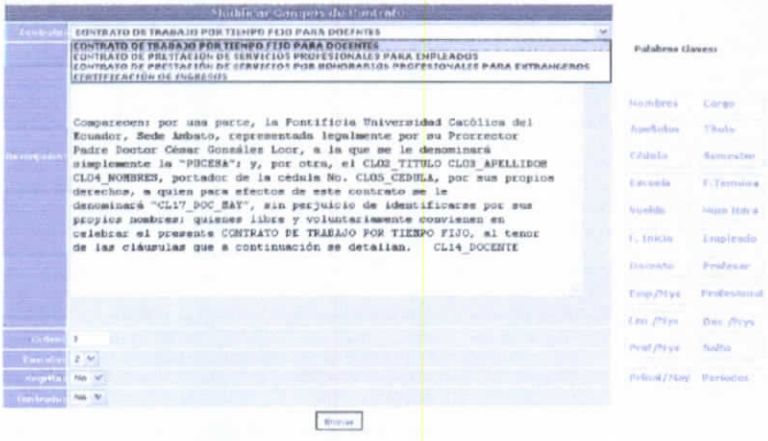


En la ventana de modificar además de aparecer la cláusula de un contrato determinado aparecerán unas palabras claves las cuales remplazarán el nombre, apellidos, cédula y otros en dicha cláusula.

También tenemos el orden el cual es en el orden que desee que la cláusula aparezca en el contrato, Tamaño, es el tamaño de la letra, Negrita sirve para el caso en que alguna cláusula requiere ser en negrita y por ultimo centrado sirve en caso que se desee que el texto vaya centrado en el contrato.

Insertar una Nueva Cláusula

Para insertar un nueva cláusula a un contrato damos un clic en el botón nuevo campo, luego elegimos el tipo de contrato al cual queremos añadir la cláusula, luego digitamos el texto de la cláusula insertando las palabras claves correspondientes:



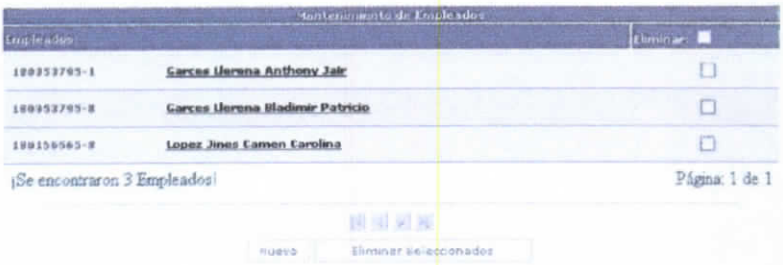
Una ves ingresado el texto presionamos enviar para que se graben los datos ingresados.

Eliminar las Cláusulas de un contrato

Para eliminar las cláusulas de un contrato podemos hacerlo marcándolos uno a uno o por medio del primer check box para seleccionar todas las cláusulas de un contrato y luego presionamos eliminar seleccionados.

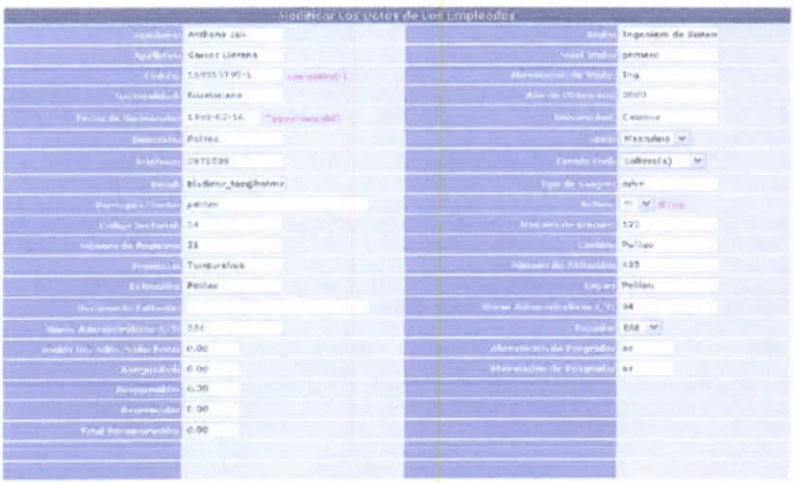
Mantenimiento de Empleado

Al igual que los procesos anteriores para acceder al reporte de los empleados lo hacemos por medio del menú mantenimiento y escogemos empleados:



Modificar los Datos Empleados

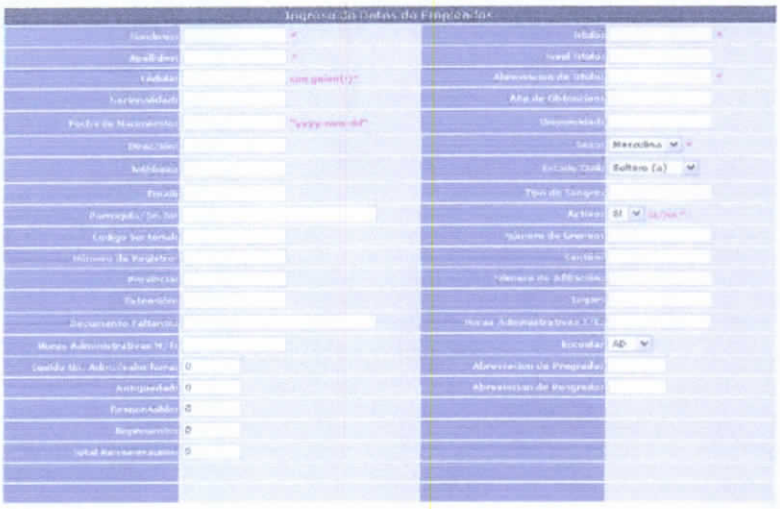
Para modificar seguimos el mismo procedimiento que en los tipos de contrato damos un clic en el nombre del empleado que queremos modificar y nos aparecerá la ventana para modificar :



Una ves modificado presionamos enviar para que se graben los datos.

Insertar un nuevo empleado

Al igual que para insertar un nuevo tipo de contrato damos un clic en el botón nuevo y nos aparecerá la ventana para ingresar los datos del empleado.



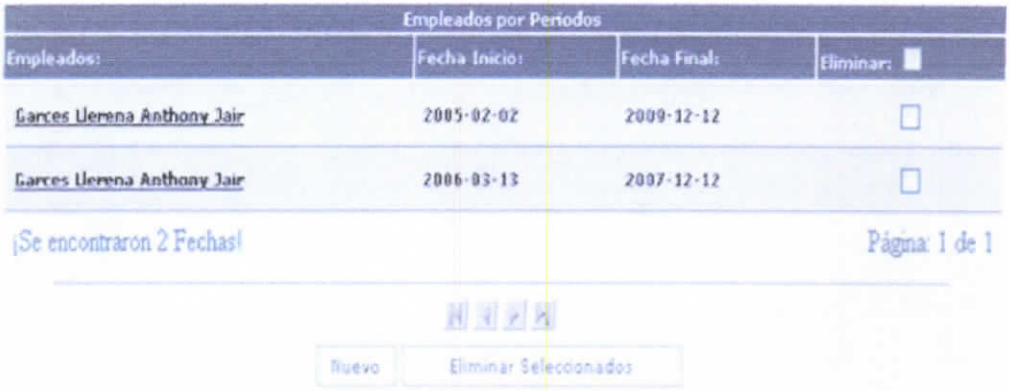
Nota: Este formulario tiene algunos campos obligatorios y están marcados con un asterisco (*).

Eliminar los Datos de un Empleado

Para eliminar los datos de los empleados podemos hacerlo marcándolos uno a uno o por medio del primer check box para seleccionar a todos los empleados que se desea eliminar y luego presionamos eliminar seleccionados.

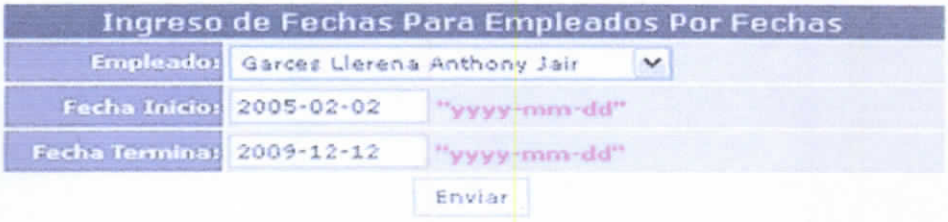
Empleado Por Fecha

Al igual que los procesos anteriores para acceder al reporte de los empleados por fecha de ingreso y salida lo hacemos por medio del menú mantenimiento y escogemos empleados por periodos:



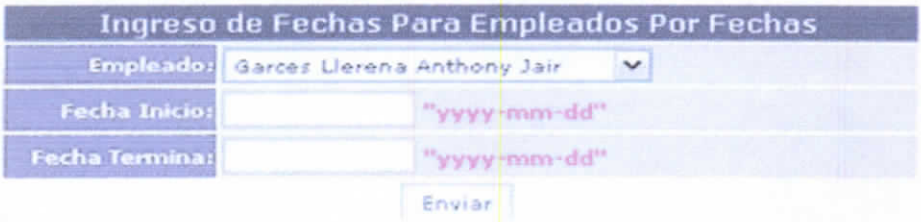
Modificar Empleado por Fechas

Para modificar seguimos el mismo procedimiento que en el formulario de empleados damos un clic en el nombre del empleado que queremos modificar y nos aparecerá la ventana para modificar :



Nuevo Empleado Por Fechas

Al igual que para insertar un nuevo empleado damos un clic en el botón nuevo y nos aparecerá la ventana para ingresar los datos.

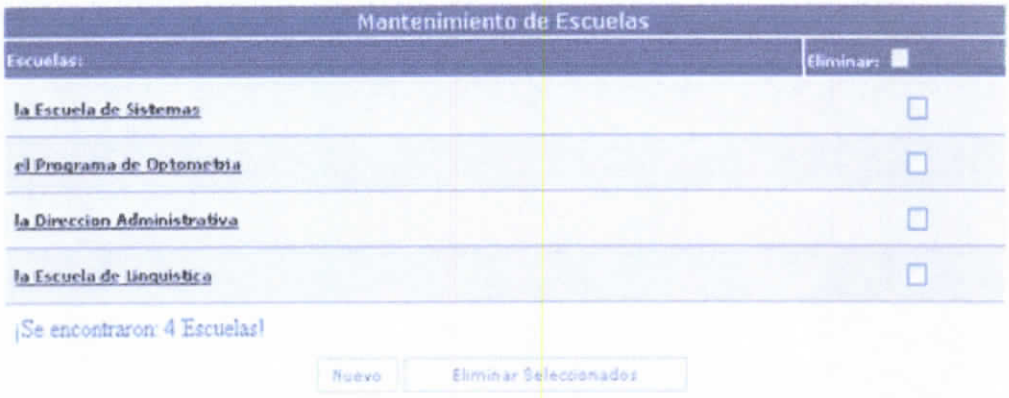


Eliminar un Empleado Por Fecha

Podemos eliminar igual que en los dos casos anteriores seleccionamos el empleado uno a uno o por medio del check box que se encuentra al inicio para marcar a todos los empleados.

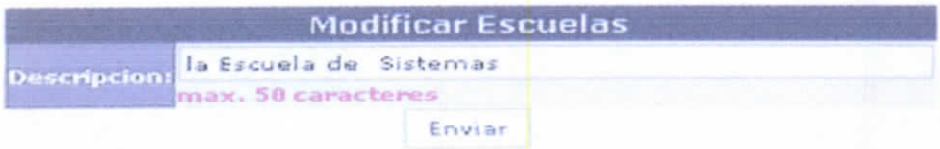
Escuelas

Para obtener el reporte de las diferentes escuelas vamos a acceder al menú mantenimiento y escogemos en el menú "escuelas", y nos aparecerá la siguiente ventana:



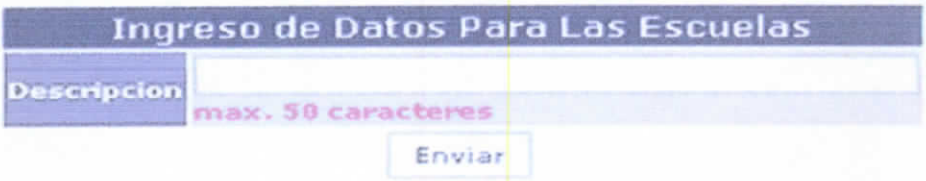
Modificar Escuelas

Para modificar una escuela realizaremos el mismo procedimiento que en los procesos anteriores, dando un clic en el nombre de la escuela para modificar la escuela seleccionada, y nos aparecerá la siguiente ventana donde podremos modificar.



Nueva Escuela

Como en los procesos anteriores damos un clic en el botón nuevo para que se aparezca la ventana para ingresar una nueva escuela.

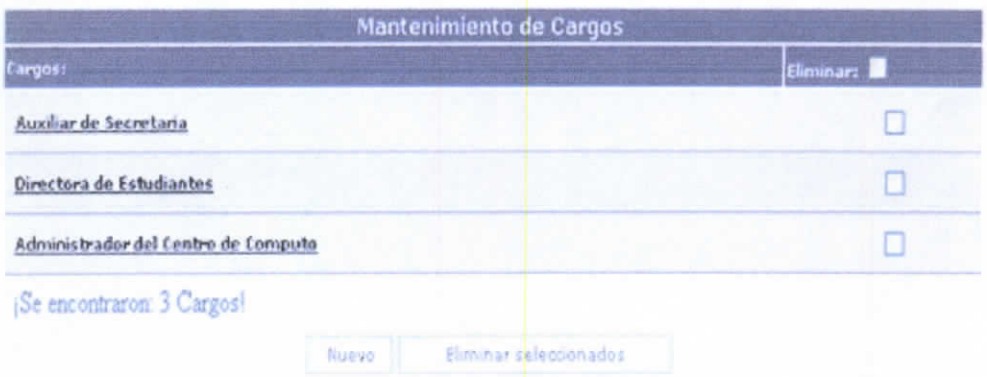


Eliminar una Escuela

Para eliminar las escuelas demos hacerlo marcándolos uno a uno o por medio del primer check box para seleccionar a todas las escuelas y luego presionamos eliminar seleccionados.

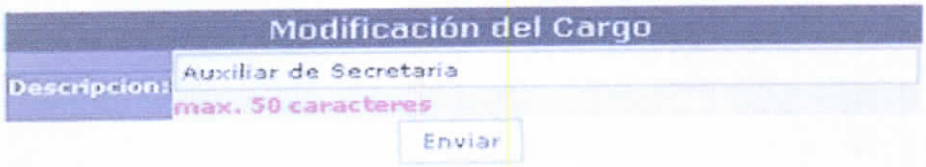
Cargos

Para obtener el reporte de las diferentes cargos vamos a acceder al menú mantenimiento y escogemos en el menú "Cargos", y nos aparecerá la siguiente ventana:



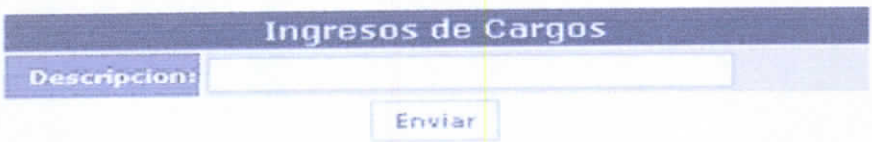
Modificar Cargos

Para modificar un cargo realizaremos el mismo procedimiento que en lo procesos anteriores, dando un clic en el campo que se desee modificar, y nos aparecerá la siguiente ventana donde podremos modificar el cargo.



Nuevo Cargo

Como en los procesos anteriores damos un clic en el botón nuevo para que se aparezca la ventana para ingresar un nuevo cargo.

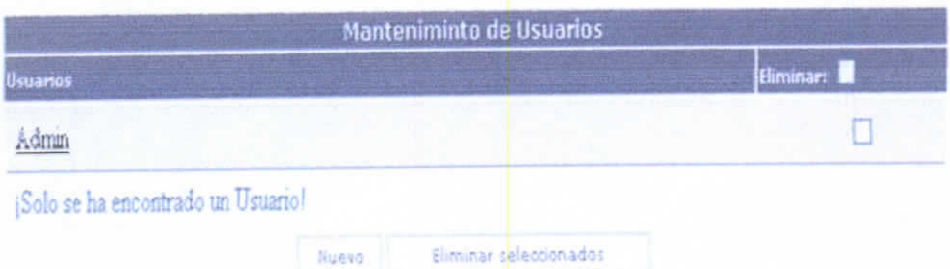


Eliminar un Cargo

Para eliminar los cargos podemos hacerlo marcándolos uno a uno o por medio del primer check box para seleccionar todos los cargos y luego presionamos eliminar seleccionados.

Usuarios

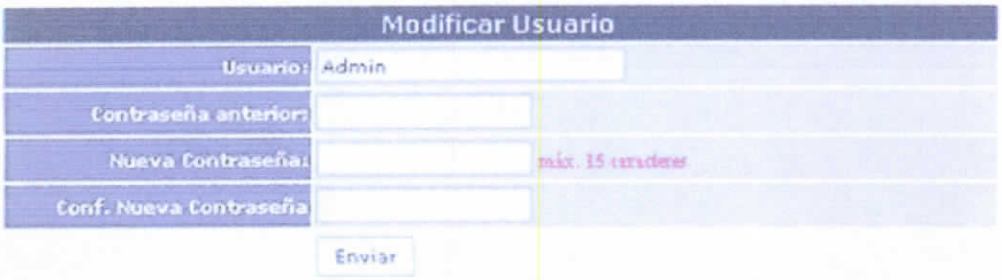
Para acceder al reporte de todos los usuarios lo hacemos por medio del menú mantenimiento y escogemos usuarios y damos un clic, y nos aparecerá la siguiente ventana:



Modificar un Usuario

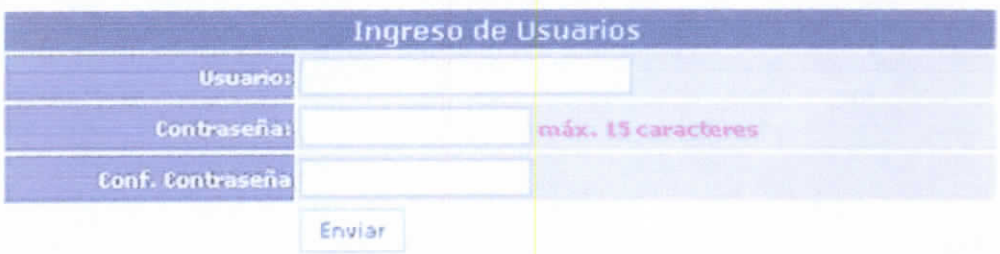
Para el mantenimiento de los usuario lo haremos al igual que en los casos anteriores al usuario que se desea modificar la clave se procede a dar un clic sobre el nombre del usuario escogido.

NOTA: La modificación únicamente se la realiza a la contraseña del usuario.



Nuevo Usuario

Al igual que en los procesos anteriores lo haremos dando un clic en el botón nuevo y aparecerá la siguiente pantalla:



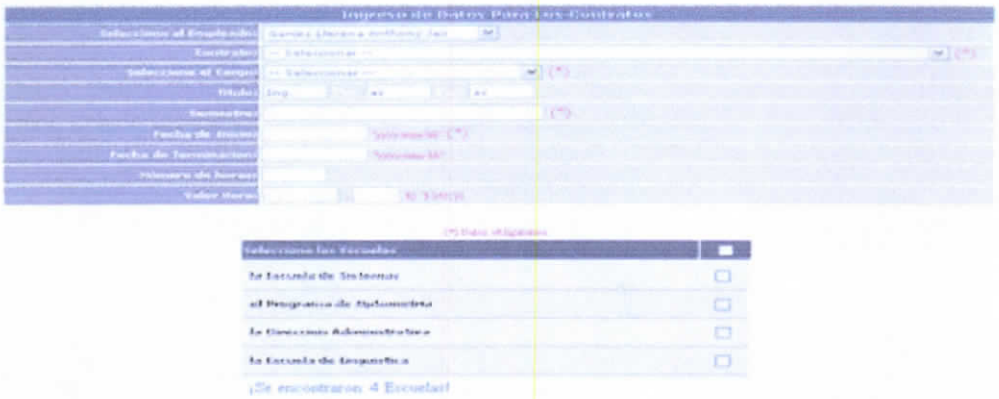
NOTA: La contraseña y la confirmación tiene que coincidir para que pueda ingresar el nuevo usuario.

Eliminar un Usuario

Para eliminar los usuarios podemos hacerlo marcándolos uno a uno o por medio del primer check box para seleccionar todos los usuarios y luego presionamos eliminar seleccionados.

Emisión de Contratos

Para la emisión de los contratos tenemos acceder por medio del menú "Procesos", luego escogemos "contratos". Tenemos que ingresar los datos requeríos (*) para poder emitir el contrato.



Una ves ingresado los datos requeridos para emitir el contrato presionamos el botón enviar y nos aparecerá el contrato listo para imprimir.

PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR
SEDE AMBATO

CONTRATO DE PRESTACION DE SERVICIOS PROFESIONALES

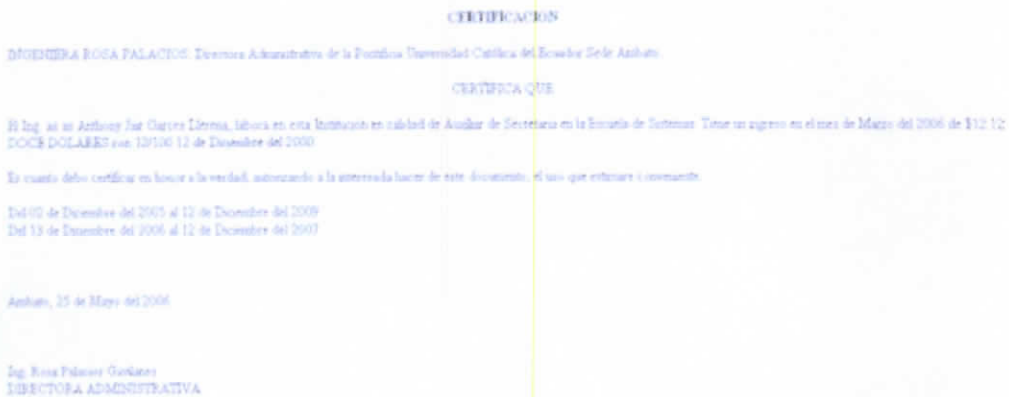
PRIMERA.- COMPARECIENTES

SEGUNDA.- ANTECEDENTES

Emisión de Certificados

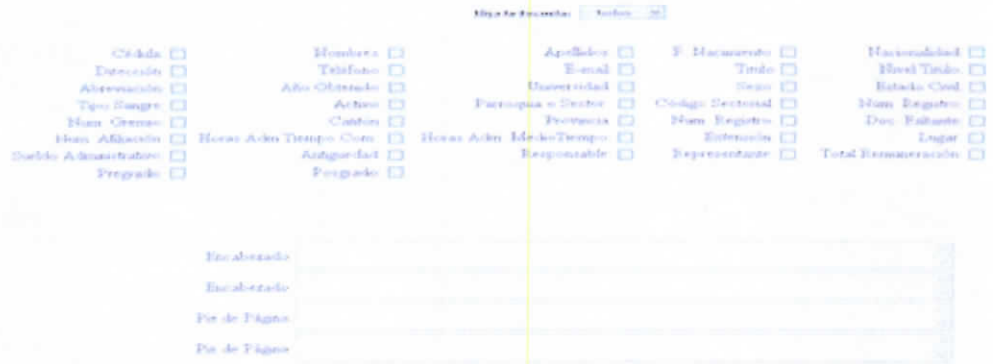
Para la emisión de los certificados tenemos acceder por medio del menú "Procesos", luego escogemos "certificados". Tenemos que ingresar los datos requeríos (*) para poder emitir el certificado.

Una ves ingresado los datos requeridos para emitir el certificado presionamos el botón enviar y nos aparecerá el certificado listo para imprimir.



Emisión de Reportes

Para la emisión de los reportes tenemos acceder por medio del menú "reportes", luego escogemos "reportes". Tenemos que ingresar los datos requeríos (*) para poder emitir el reporte.



Una ves ingresado los datos requeridos para emitir el reporte presionamos el botón enviar y nos aparecerá el reporte listo para imprimir.

Cédula	Nombres	Apellidos
180156565-8	Carmen Carolina	Lopez Jnes
180353795-1	Anthony Jair	Garces Llerena

ANEXO C. Estándar IEEE Std 830-1993.

Para esta etapa se utilizan las recomendaciones de la IEEE según su estándar 830-1993. Dichas recomendaciones describen el proceso de creación de la especificación de requisitos software.

Respecto a las recomendaciones destacamos:

- Naturaleza

  La especificación de requisitos software es una especificación para un producto software en particular, programa, o conjunto de programas que realizan ciertas funciones en una un entorno específico. Son cuestiones básicas que deben ser tratadas.:

  - Funcionalidad.
  - Interfaces externos.
  - Rendimiento.
  - Atributos: corrección, mantenimiento, seguridad, transportabilidad, etc.
  - Restricciones de diseño impuestas sobre la implementación.

- Entorno

  La especificación de requisitos software debe:

  - Definir correctamente todos los requisitos software. Un requisito software puede existir por la naturaleza de la tarea a resolver o por una características especial del proyecto.
  - No debe describir ningún diseño o detalle de implementación.
  - No debe imponer ninguna limitación adicional al software.

- Características
  - Correcto
  - No ambiguo
  - Completo
  - Consistente
  - Clasificado por importancia y/o estabilidad
  - Verificable
  - Modificable
  - Rastreable

- Evolución

  La especificación de requisitos puede evolucionar según progresa el proceso de desarrollo del software.

- Inclusión de diseño.

  La especificación de requisitos software no debe incluir generalmente cuestiones de diseño como:

  - Partición del software en módulos.
  - Asignar funciones a módulos.
  - Describir flujos de información o control entre módulos.
  - Elección de estructuras de datos.

ANEXO D. Glosario de términos.

API: (Interfaz de Programación de Aplicaciones). Grupo de procedimientos que un programa de computadora llama para acceder a un servicio específico.

BASE DE DATOS: Conjunto de datos organizados de modo tal que resulte fácil acceder a ellos, gestionarlos y actualizarlos. Una guía telefónica es un ejemplo de una base de datos

BACK END: Base de Datos.

BROWSER: Navegador.

CGI: (Common Getaway Interface). Interface de intercambio de datos estándar en www a través del cual se organiza el envío de recepción de datos entre visualizadores y programas residentes en servidores www.

C: Lenguaje de programación estructurado, de propósito general, cuyo uso está muy extendido.

C++: Lenguaje de programación orientado a objetos, basado en el lenguaje C.

COMANDO: (Command). Instrucción que un usuario da al sistema operativo de la computadora para realizar determinada tarea.

CLIENTE/SERVIDOR: Este término define la relación entre dos programas de computación en el cual uno, el cliente, solicita un servicio al otro, el servidor, que satisface el pedido.

E-MAIL: (Electronic Mail). Correo electrónico. Mensajes (normalmente privados) enviados a través de una red de ordenadores.

FTP: (File Transfer Protocol). Protocolo estándar en Internet para transferencia de ficheros

FRONT END: Interfase.

FTP: (File Transfer Protocol). Protocolo de Transferencia de Ficheros.

GB: Abreviatura de GigaByte.

HD: (Hard Disk). Disco duro

HTM: Extensión bajo MsDos de los ficheros de tipo html.

HTML: (HyperText Markup Language). El lenguaje de descripción de páginas habitual.en Internet.

HTTP: (HyperText Transfer Protocol). El protocolo usado en las páginas del www.

IDE: Uno de los estándares en conexión de discos duros y dispositivos similares

INNODB: (B-TREE o Árboles B). Los índices no proporcionan ninguna compresión del embalaje o del prefijo. Además, InnoDB también requiere una llave primaria para cada tabla. Como con BDB, aunque, si se asigna una llave primaria, MySQL proveerá un valor 64-bit.

TCP/IP: Se refieren a dos protocolos de red, que son Transmission Control Protocol (Protocolo de Control de Transmisión) e Internet Protocol (Protocolo de Internet) respectivamente.

MEGABYTE: Múltiplo del byte: un megabyte son 1.024 KiloBytes, cerca de un millón de bytes.

RESPONSE: Respuesta del servidor al cliente.

REQUEST: Petición que realiza el cliente al servidor.

SCSI: (Small Computer System Interface). Una conexión estándar para diversos dispositivos como discos duros o scanners.

SQL: (Structured Query Language). Un lenguaje estándar de consulta a bases de datos.

SRS: (Software Requirements Specification). Especificación de requisitos del Software.