PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL ECUADOR
SEDE AMBATO

ESCUELA DE INGENIERÍA EN SISTEMAS

Tema:

"SISTEMA DE MONITOREO DE EGRESADOS DE LA PUCESA"

Autores:

SANDRA MICHAELA ALEMÁN GUERRA
PABLO EDUARDO JARAMILLO ZAMORA

Asesor:

ING. Darío Robayo

Ambato – Ecuador
Julio 2007

PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR
SEDE AMBATO

ESCUELA DE INGENIERÍA EN SISTEMAS

Tema:

SISTEMA de MONITOREO de EGRESADOS de la PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR SEDE AMBATO

Autores:

SANDRA MICHAELA ALEMÁN GUERRA
PABLO EDUARDO JARAMILLO ZAMORA

Asesor:

_____
ING. Darío Robayo

Ambato – Ecuador
Julio 2007

DECLARACIÓN DE AUTENTICIDAD
Y RESPONSABILIDAD

Yo, Sandra Michaela Alemán Guerra portador de la cédula de ciudadanía No. 1803219458 declaro que los resultados obtenidos en la investigación que presento como informe final, previo la obtención del Título de "SISTEMA DE MONITOREO DE EGRESADOS DE LA PUCESA" son absolutamente originales, auténticos y personales.

En tal virtud, declaro que el contenido, las conclusiones y los efectos legales y académicos que se desprenden del trabajo propuesto son de exclusiva responsabilidad legal y académica del autor.

Sandra Michaela Alemán Guerra
CI. 1803219458

DECLARACIÓN DE AUTENTICIDAD
Y RESPONSABILIDAD

Yo, Pablo Eduardo Jaramillo Zamora portador de la cédula de ciudadanía No. 180324643 declaro que los resultados obtenidos en la investigación que presento como informe final, previo la obtención del Título de "SISTEMA DE MONITOREO DE EGRESADOS DE LA PUCESA" son absolutamente originales, auténticos y personales.

En tal virtud, declaro que el contenido, las conclusiones y los efectos legales y académicos que se desprenden del trabajo propuesto son de exclusiva responsabilidad legal y académica del autor.

Pablo Eduardo Jaramillo Zamora
CI. 1803241643

AGRADECIMIENTO

Muchas han sido las personas que de manera directa o indirecta nos han ayudado en la realización de esta tesis. Queremos dejar constancia de todas ellas y agradecerlas con sinceridad su participación.

Queremos extender nuestra gratitud a la Pontificia Universidad Católica del Ecuador Sede Ambato, ya que es allí donde hemos adquirido conocimientos que nos han llevado a culminar nuestra carrera

Agradecemos principalmente al Ing, Dario Robayo nuestro director de tesis cuyo invaluable apoyo e interés hicieron posible la realización de esta tesis. De igual manera al Ing. Santiago Acurio, director de la Escuela de Sistemas que nos brindo su apoyo incondicional.

Desde luego, no podemos dejar de agradecer a nuestras familias, su constante apoyo y su inmenso cariño han sido la base fundamental para llevar a feliz término esta etapa de nuestras vidas.

Gracias infinitas a todos

DEDICATORIA

Este trabajo de tesis esta enteramente dedicado a mis padres, hermanos, a mi esposo y a mis hijas.

Gracias por confiar en mí y brindarme todo su apoyo, es obvio que sin ustedes este sueño nunca hubiera podido ser alcanzado. Sencillamente ustedes son la base de mi vida tanto emocional como profesional; realmente no hay palabras que logren expresar mi profundo agradecimiento.

Sandra

DEDICATORIA

Agradesco a mis padres y a mi hermano por todo el apoyo brindado recompensado en este trabajo, a todas las personas que me apoyaron y alentaron. Con lo que concluyo una etapa más de mi vida. Gracias.

Pablo

TABLA DE CONTENIDOS

Ilustraciones.

INTRODUCCIÓN

En vista del crecimiento estudiantil de LA PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR SEDE AMBATO y de la falta de control de los Egresados y Graduados de la misma, se ha pensado en una práctica solución muy acorde a las nuevas tendencias de manejo de información que es vía Internet, esto es con la implementación de un *Sistema de Monitoreo de los Egresados* de la Universidad, dentro de la Página Web que posee la PUCESA, para que los egresados puedan tener actualizados sus datos, tanto personales como profesionales. De esta manera la Escuela de Sistemas tendrá un sistema de información mas detallado de lo que están realizando sus ex – alumnos, así como también, la capacitación de los mismos.

Así de igual forma se pueden tener los datos personales, tanto de los egresados como de los graduados de la PUCESA, con el propósito de tener sus datos actualizados por medio de la página Web de la PUCESA con una clave personal.

Con esta página se facilitará la consulta y búsqueda de los datos de los egresados de la PUCESA, manteniéndose el contacto con los ex alumnos de la Universidad y su información profesional.

1. CAPITULO I. Proyecto de la Investigación.

1.1. Planteamiento del Problema.

1.1.1. Problema

Carencia de un "Sistema de Monitoreo de los Egresados de La Pontificia Universidad Católica del Ecuador Sede Ambato" que proporcione información estadística de los mismos en el periodo 2006 – 2007.

1.1.2. Problematización

- Falta de información de los egresados de la PUCESA
- Carencia de información estadística de los egresados
- Inexistencia de información del nivel académico obtenido por los egresados.

1.2. Delimitación

El campo en el que se desarrolla la información del sistema es en la misma página Web de la PUCESA. El tiempo utilizado para la elaboración del sistema ha sido de diez meses.

El Sistema de Monitoreo de los Egresados está orientado a ampliar la información de los ex alumnos de la PUCESA por medio de su página Web.

El sistema cuenta con una base de datos propia con el motor de base de datos MySQL que se alberga en el servidor SUN e250 del Centro de Cómputo de la Universidad.

Se utiliza el lenguaje PHP (acrónimo: Hypertext Preprocessor), el mismo que es un lenguaje "Open Source" interpretado como de alto nivel, el que es utilizado en la página Web de la Universidad, especializado para las mismas.

El sistema está orientado a ampliar la información de los ex alumnos de la PUCESA por medio de su página, el mismo que podrá ingresar por medio de su número de cédula de identidad y llenar sus datos respectivos con el propósito de incrementar la información de los egresados y graduados de las diferentes Escuelas de la PUCESA, de esta manera conocer los logros alcanzados académicamente.

El Módulo esta compuesto de lo siguiente:

- Información de ex alumnos.
- Datos personales y profesionales de los ex alumnos.
- Datos estadísticos.

1.3. Importancia y Justificación

1.3.1. Importancia

A nivel Nacional, el presente proyecto servirá de base para el desarrollo de Sistemas de información de ex alumnos.

A nivel local, servirá como un estudio de las necesidades reales, que requieren las instituciones educativas, adaptándose a las nuevas exigencias de capacitación e información que estas demanden, debido al crecimiento de la localidad.

El presente proyecto de disertación es importante ya que permitirá que los egresados de la PUCESA, tengan actualizados sus datos tanto personales como laborales para opciones de empleos.

1.3.2. Justificación

El Sistema de Monitoreo de los Egresados de la Pontificia Universidad Católica del Ecuador Sede Ambato, es de gran utilidad, pues utiliza cálculos estadísticos para de esta manera acceder a la información de los egresados, vía Página Web.

Existen los conocimientos necesarios para el desarrollo e implementación del Sistema de Monitoreo de los Egresados de la Pontificia Universidad Católica del Ecuador Sede Ambato; además los costos son accesibles para su elaboración; se cuenta también con el Software y Hardware necesarios para la implementación y funcionamiento, el asesoramiento de la Dirección de la Escuela de Sistemas así como del Centro de Cómputo para el Hosting del Sistema de Monitoreo de los Egresados.

1.4. Objetivos

1.4.1. General

Desarrollar un "Sistema de Monitoreo de los Egresados de la Pontificia Universidad Católica del Ecuador Sede Ambato", para incrementar la información estadística de los egresados de las diferentes Escuelas.

1.4.2. Objetivos Específicos

- Incrementar la información de los egresados y graduados de las diferentes Escuelas de la PUCESA.
- Alimentar al sistema de Monitoreo de Egresados con datos estadísticos.
- Implementar este sistema para que los ex alumnos llenen los datos personales y de esta manera conocer los logros alcanzados académicamente

1.5. Hipótesis

Con la implementación del "Sistema de Monitoreo de los Egresados de La Pontificia Universidad Católica del Ecuador Sede Ambato" y creada la información estadística de los egresados y graduados de la PUCESA, podrán tener actualizado sus datos, tanto personales como profesionales.

1.6. Aspectos Metodológicos

1.6.1. Fundamentos Teóricos

Este proyecto de disertación se ha basado en los siguientes paradigmas:

- **Paradigma Racionalist**a: debido a que el planteamiento puede llevarse a la realidad y existen los mecanismos y medios necesarios para su plena realización las cuales brindarán las facilidades necesarias para cumplir el objetivo.
- **Paradigma Pragmatista:** en base a un conjunto de ideas teóricas y la práctica de las mismas, nos ayudará a resolver el problema.

- **Paradigma Crítico:** ya que intenta transmitir mensajes por medio de los resultados obtenidos y conseguir efectos positivos en la educación universitaria.

1.6.2. Métodos de Investigación

Los métodos de investigación usados son: El Método Lógico Inductivo Completo, ya que se conoce el objeto total de estudio y además la lógica inductiva ayudara a plantear mejor el problema; el Método de Inducción por Simple Enumeración por que se conocen los elementos a estudiar además de que estos no son infinitos ni inconmensurables.

También se uso la Investigación Aplicada porque el Sistema de Monitoreo de los Egresados de la Pontificia Universidad Católica del Ecuador Sede Ambato será incluido en la página Web de la PUCESA.

2. Capítulo II Marco Teórico

2.1.Software

2.1.1. Definición

Conjunto de instrucciones (programas de computadoras) que cuando se ejecutan proporcionan la función y los rendimientos deseados con estructuras de datos que permiten a los programas manipular adecuadamente la información con sus documentos que describen la operación y el uso de programas.

2.1.2. Características del Software

El software no se desarrolla, se fabrica. A diferencia de la construcción de hardware, esta puede introducir problemas de calidad que no existen o son fáciles de corregir en el desarrollo del software. Ambas de estas actividades dependen de las personas, pero la relación entre personal dedicadas y el trabajo realizado es completamente diferente para el software.

El software no se estropea. El software no es tangible al entorno físico lo que hace que el software no sufra daños. Cada error del software significa un error en el diseño o en el proceso mediante el cual se tradujo del diseño a código de maquina ejecutable. Por tanto el mantenimiento del mismo es mucho más importante y complejo que el del hardware.

El software si se deteriora. El software al igual que el hardware sufren deterioro por el tiempo y el uso.

El software se construye a medida en vez de ensamblarse. Se puede adquirir software ya desarrollado pero solo como una unidad completa, mas no como componentes que pueden ser reaplazados en los nuevos programas.

2.2. Ingeniería del Software

La Ingeniería de software es una disciplina o un rama de la informática que crea y mantiene las aplicaciones de software, aplicando tecnologías y prácticas de las ciencias computacionales, manejo de proyectos en el ámbito de la creación de aplicaciones; Por otra parte se exige que el software sea eficaz y barato tanto en el desarrollo como en la compra con una serie de características como fiabilidad, facilidad de mantenimiento y de uso, eficiencia, codificación, casos de prueba, manuales técnicos y de usuarios.

2.2.1. Conceptos y Definiciones.

Barry Bohem 1976. "Ingeniería de software es la aplicación práctica del conocimiento científico en el diseño y construcción de programas para computadoras y su mantenimiento".

Lewis 1994. "Ingeniería de software es la suma total de los programas de computadora, procedimientos, reglas, la documentación asociada y los datos que pertenecen a un sistema de cómputo. Un producto de software es un producto diseñado para un usuario".

Cota 1994. "Ingeniería de Software es un enfoque sistemático del desarrollo, operación, mantenimiento y retiro del software, es decir, se consi-

dera que la Ingeniería de Software es la rama de la ingeniería que aplica los principios de la ciencia de la computación y las matemáticas para lograr soluciones costo-efectivas (eficaces en costo o económicas) a los problemas de desarrollo de software".

Jacobson 1998. El proceso de ingeniería de software se define como "un conjunto de etapas parcialmente ordenadas con la intención de logra un objetivo, en este caso, la obtención de un producto de software de calidad".El proceso de desarrollo de software "es aquel en que las necesidades del usuario son traducidas en requerimientos de software, estos requerimientos transformados en diseño y el diseño implementado en código, el código es probado, documentado y certificado para su uso operativo". Concretamente "define quién está haciendo qué, cuándo hacerlo y cómo alcanzar un cierto objetivo"

IEEE 1993. "La aplicación de un enfoque sistemático disciplinado para el desarrollo, operación, mantenimiento y eliminación del software".

2.2.2. Objetivo de la Ingeniería del Software.

- Encontrar procesos o metodologías predecibles y repetibles que mejoren la productividad y la calidad del software.
- Facilitar el proceso y control para el desarrollo del software.
- Mejorar las herramientas para los desarrolladores de software para mejorar la calidad y eficiencia.

- La construcción y desarrollo de proyectos se aplican métodos y técnicas para resolver los problemas, la informática aporta herramientas y procedimientos sobre los que se apoya la ingeniería de software.

2.2.2.1. Mitos del Software

Mitos de Gestión.

- Se tiene información de los estándares para el desarrollo del software y saber que sistema construir. Lo cual no es suficiente toda la información sin utilizarla.
- Si fallamos en la planificación se podrá incluir mas programadores para resolver el problema del tiempo de entrega del proyecto.

Mitos del Cliente

- Una declaración de los objetivos es suficiente para el desarrollo.
- Los requerimientos del software pueden cambiar constantemente y se puede solucionar facialmente por que el software es flexible.

Mitos de los Desarrolladores

- Una ves que hemos escrito el programa y lo hacemos funcionar, nuestro trabajo termina.
- Hasta que el programa no este corriendo no hay forma de probar su calidad.

2.2.2.2. Calidad del Software

La calidad del software trata la combinación de varios factores: Corrección. El software debe cumplir sus tareas y especificaciones exactamente. Para ello es necesario realizar especificaciones precisas y usar métodos de corrección convencional.

Robustez. Es la capacidad de los sistemas para reaccionar apropiadamente ante condiciones anormales. Se caracteriza por funcionar fuera de los límites de su especificación.

Extensibilidad. Es la facilidad de adaptación del software a cambios de **especificación. Para que ello ocurra es necesario que el software se caracterice** por su simplicidad de diseño y la descentralización de módulos.

Reutilización. Es la capacidad de los elementos del software de servir para el desarrollo de aplicaciones diferentes. Es recomendable evitar reinventar soluciones a problemas que ya han sido encontrados. Eficiencia. La eficiencia del software consiste en que este debe aprovechar al máximo los recursos de hardware. Facilidad de Uso. El sistema debe facilitar el manejo para todo tipo de usuario.

Oportunidad. El software debe estar desarrollado para cuando los usuarios lo necesiten o antes si es posible.

Verificabilidad. Es la facilidad para preparar procedimientos de prueba

Integridad. La integridad consiste en la capacidad para que el sistema se proteja contra modificaciones y accesos no autorizados Economía. Tiene que encontrarse dentro el presupuesto.

2.2.2.3. Aplicaciones de Software.

Software de Sistemas. Es un conjunto de programas que han sido escritos para ser utilizados por otros programas. Se caracteriza por una fuerte interacción con el hardware de la computadora; una gran utilización por múltiples usuarios; una operación concurrente que requiere una planificación, una compartición de recursos y una sofisticada gestión de procesos; unas estructuras de datos complejas y múltiples interfases externas.

Software de Tiempo Real. Es el que mide, analiza y controla sucesos del mundo real conforme van ocurriendo. Entre los elementos que lo conforman se incluyen: un componente de adquisición de datos que recolecta y da formato a la información recibida del entorno externo, un componente de análisis que transforma la información recibida del entorno externo, un componente de análisis que transforma la información según lo requiera la aplicación, un componente de control/salida que responda al entorno externo y un componente de monitorización que coordina todos los demás componentes, de forma tal que pueda mantenerse la respuesta en tiempo real.

Software de Gestión. El procesamiento de información comercial constituye la mayor de las áreas de aplicación del software. Las aplicaciones reestructuran los datos existentes para facilitar las operaciones comerciales o gestionar la toma de decisiones. Además de las tareas convencionales de procesamiento de datos, las aplicaciones de software de gestión también realizan cálculo interactivo.

Software de Ingeniería y Científico. Se caracteriza por los algoritmos de manejo de números. Las aplicaciones abarcan la astronomía, vulcanología, biología molecular, fabricación automática, etc.

***Software Empotrado*.** Reside en memoria de solo lectura y se utiliza para controlar productos y sistemas de los mercados industriales y de consumo. El software empotrado puede ejecutar funciones muy limitadas y curiosas o suministrar una función significativa y con capacidad de control.

Software de Inteligencia Artificial. El software de inteligencia artificial hace uso de algoritmos no numéricos para resolver problemas complejos para los que no son adecuados el cálculo o el análisis directo. El área más activa de la inteligencia artificial es la de los sistemas expertos, también llamados sistemas basados en el conocimiento.

2.2.3. Fundamentos y Técnicas de Ingeniería de Software.

- **Métodos.** Definen el cómo construir el software desde el punto de vista técnico.

- **Herramientas.** Proporcionan un soporte automático o semi-automático para los métodos.
- **Procedimientos.** Definen la secuencia en la que se aplican los métodos, cómo usar las herramientas, las entregas que se requieren, controles de seguimiento y calidad, guías para facilitar la labor de gestores y desarrolladores, etc.

2.2.3.1. Metodología.

Es el conjunto de métodos con el propósito de formalizar y optimizar los procesos de desarrollo de software definiendo qué hacer, cómo y cuándo durante todo el transcurso del proyecto. Algunas ventajas de usar una metodología:

- Proporcionar la tarea de planificación, control y seguimiento del proyecto.
- Optimizar el uso de recursos, mejorando la relación costo-beneficio.
- Facilitar la evaluación de resultados y cumplimiento de los objetivos.
- Optimizar el conjunto y cada una de las fases del proceso de desarrollo.
- Garantizar un determinado nivel de calidad en el producto final y generar confianza en los plazos de tiempo fijados en la definición del proyecto.
- Aunque existen diferentes métodos de diseño del software, los más importantes, desarrollados y utilizados son el diseño orientado a flujo de datos y el diseño orientado a objetos.

- **Diseño orientado a flujo de datos**

La metodología de diseño orientada a flujo de datos se centra en el movimiento y transformación de los datos a medida que se mueven por el sistema.

Considera los datos de entrada al sistema, estudia todos los caminos que recorren y su transformación hasta que alcanzan la salida. Estas metodologías son consideradas clásicas o convencionales frente a las más novedosas orientadas a objeto.

Durante la fase de diseño se realizan una serie de actividades cuyo objetivo es pasar de las ideas más o menos informales recogidas en la fase de análisis a definiciones detalladas y precisas. Estas actividades son:

- ***Diseño de datos*** Transforma el modelo de datos obtenido en el análisis en las estructuras de datos necesarias para el sistema. La base para esta actividad son el Diagrama Entidad-Relación (DER) y el diccionario de datos.
- ***Diseño arquitectónico*** Define la estructura modular del sistema mostrando la relación entre los principales elementos estructurales del sistema. Se parte de los diagramas de flujo de datos obtenidos en el análisis.
- ***Diseño de la interfaz*** Describe cómo se comunica el software con otros sistemas y con los usuarios del mismo.

- ***Diseño procedimental*** En esta actividad se transforman los elementos estructurales en una descripción procedimental de los

componentes. La base para esta actividad son los requisitos funcionales obtenidos en la fase de análisis.

Todas estas actividades se desarrollan en un proceso iterativo mediante el cual se traducen los requisitos en una representación del software. Se parte de un alto nivel de abstracción refinándolo en cada iteración hasta obtener un nivel lo suficientemente bajo para que se pueda pasar a la fase de implementación.

El modelo de diseño se puede representar como una pirámide, que es un objeto muy estable. En la base se sitúa una amplia capa muy estable con el diseño de los datos. En las capas medias se sitúan el diseño arquitectónico y de interfaz, dejando el pico de la pirámide para el diseño procedimental. La estabilidad de este modelo se basa en situar en las capas bajas los elementos que menos cambian durante el desarrollo del software como es la estructura de los datos, mientras que en la punta de la pirámide se sitúan los elementos que más cambiarán como son las funciones y procedimientos del sistema. El modelo de diseño se muestra en la figura 1.

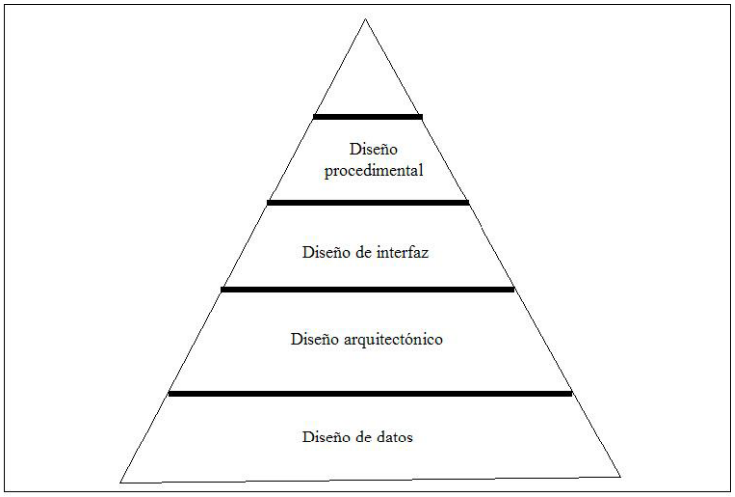


Figura 1. Modelo de diseño orientado a flujo de datos

- ***Diseño Orientado a Objetos***

El diseño orientado a objetos transforma el modelo de análisis obtenido mediante el análisis orientado a objetos en un modelo de diseño para la construcción del software. Las componentes principales del sistema se organizan en módulos denominados subsistemas. Los datos y las operaciones que los manipulan están encapsulados en objetos, una forma modular que es el bloque de construcción básico de un sistema orientado a objetos (OO).

El diseño OO descansa en cuatro importantes conceptos del diseño: abstracción, ocultación de la información, independencia funcional y modularidad.

En el diseño OO se pueden definir cuatro capas básicas necesarias para obtener un modelo completo del sistema, estas capas son:

- ***La capa de subsistemas*** Contiene una representación de los subsistemas que permiten al software cumplir con los requisitos definidos en la etapa de análisis.
- ***La capa de clases y objetos*** Contiene la jerarquía de clases y objetos del sistema.
- ***La capa de mensajes*** Contiene los detalles que permiten que los objetos se comuniquen entre si. También se establecen las interfaces del sistema.
- ***La capa de responsabilidades*** Contiene las estructuras de datos y el diseño algorítmico para los atributos y operaciones de los objetos.

Al igual que en la metodología orientada a flujo de datos, también se pueden representar estas capas en una pirámide cuya base sería la capa de diseño de subsistemas, situándose sobre ella la de clases y objetos, la de mensajes y la de responsabilidades. El modelo de diseño orientado a objetos puede verse en la Figura 2.

Hay que resaltar que aunque los métodos convencionales y orientados a objetos utilizan notaciones y heurísticas diferentes para derivar el modelo de diseño del modelo de análisis, se pueden establecer correspondencias entre cada elemento del análisis convencional y una o mas capas del análisis OO.

Así el diseño OO aplica diseño de datos cuando se representan los atributos, diseño de interfaz cuando se desarrolla el modelo de paso de mensajes y diseño procedimental al diseñar las operaciones. Sin embargo el diseño arquitectónico es diferente ya

que el diseño OO no exhibe una estructura jerárquica, de hecho la estructura del diseño OO tiene más que ver con las colaboraciones entre objetos que con el flujo de datos y control.

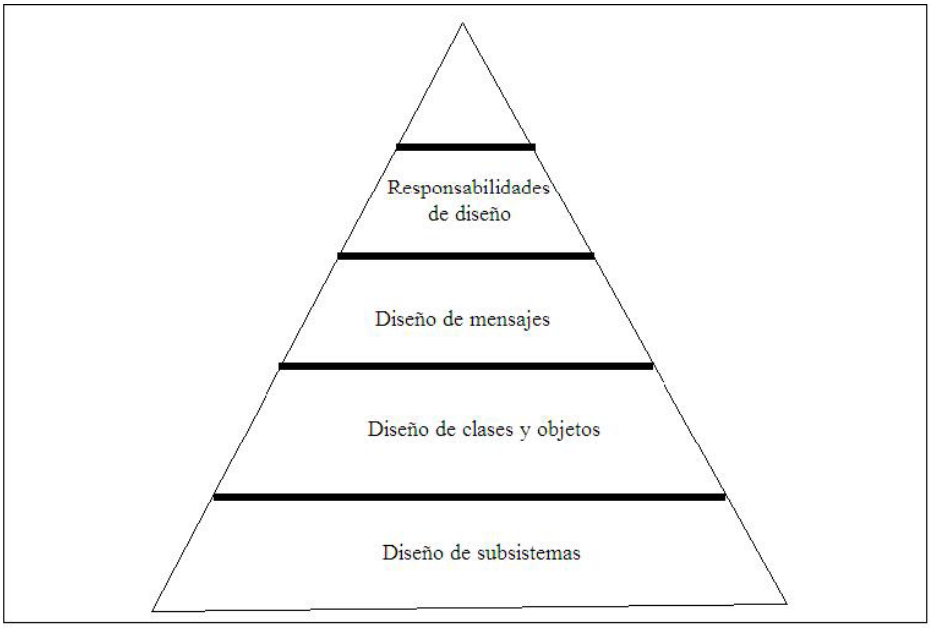


Figura 2. Modelo de diseño OO[1]

2.2.3.2. Ciclo de Vida.

Todo proyecto de ingeniería tiene unos fines ligados a la obtención de un producto, proceso o servicio que es necesario generar a través de diversas actividades. Algunas de estas actividades pueden agruparse en fases porque globalmente contribuyen a obtener un producto intermedio, necesario para continuar hacia el producto final y facilitar la gestión del proyecto.

[1] Ingenieria de Software. Presuman 1997

Sin embargo, la forma de agrupar las actividades, los objetivos de cada fase, los tipos de productos intermedios que se generan, etc. pueden ser muy diferentes dependiendo del tipo de producto o proceso a generar y de las tecnologías empleadas.

2.2.3.2.1. Requerimientos y Análisis

Para obtener los requisitos debe detallar en un documento que funciones debe realizar el sistema, sin analizar las formas de implementarlo. Esta etapa es interactiva entre las dos partes (cliente y equipo de desarrollo), con discusión de los puntos involucrados y termina con un acuerdo y aceptación de los mismos.

Partiendo de los Requisitos, debe crearse una completa especificación, esto de representa el contrato entre ambas partes, y será la entidad utilizada para realizar la VALIDACION.

Los requisitos definen las características de una solución aceptable.

La ESPECIFICACION es la parte que se "entrega", y da a ambas partes una descripción de la solución computacional propuesta.

Las especificaciones convierten los requisitos a una descripción que permite VER que aspecto tendrá y como se comportara el sistema.

En lo que se refiere a los programas de simulación, es conveniente que se muestre al usuario una versión reducida del mismo, que presente las pantallas de ingreso y egreso de datos, para discutir los cambios a introducir ante de entrar en la etapa de diseño.

2.2.3.2.2. Elementos del Ciclo De Vida

Un ciclo de vida para un proyecto se compone de fases sucesivas compuestas por tareas planificables. Según el modelo de ciclo de vida, la sucesión de fases puede ampliarse con bucles de realimentación, de manera que lo que conceptualmente se considera una misma fase se pueda ejecutar más de una vez a lo largo de un proyecto, recibiendo en cada pasada de ejecución aportaciones de los resultados intermedios que se van produciendo (realimentación).

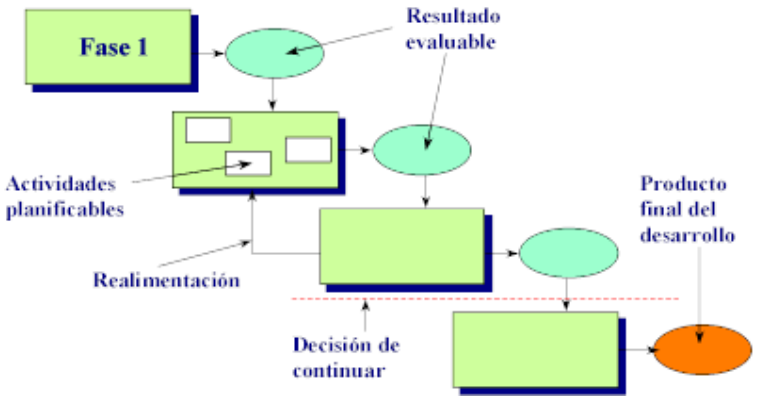


Figura 3. Ciclo de Vida

Para un adecuado control de la progresión de las fases de un proyecto se hace necesario especificar con suficiente precisión los resultados evaluables, o sea, productos intermedios que deben resultar

de las tareas incluidas en cada fase. Normalmente estos productos marcan los hitos entre fases.

A continuación presentamos los distintos elementos que integran un ciclo de vida:

- **Fases.** Una fase es un conjunto de actividades relacionadas con un objetivo en el desarrollo del proyecto. Se construye agrupando tareas (actividades elementales) que pueden compartir un tramo determinado del tiempo de vida de un proyecto. La agrupación temporal de tareas impone requisitos temporales correspondientes a la asignación de recursos (humanos, financieros o materiales).

Cuanto más grande y complejo sea un proyecto, mayor detalle se necesitará en la definición de las fases para que el contenido de cada una siga siendo manejable. De esta forma, cada fase de un proyecto puede considerarse un "micro-proyecto" en sí mismo, compuesto por un conjunto de micro-fases.

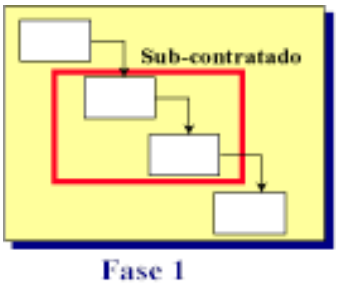


Figura 4. Fases del Ciclo de Vida

Otro motivo para descomponer una fase en sub-fases menores puede ser el interés de separar partes temporales del proyecto que se subcontraten a otras organizaciones, requiriendo distintos procesos de gestión.

Cada fase viene definida por un conjunto de elementos observables externamente, como son las actividades con las que se relaciona, los datos de entrada (resultados de la fase anterior, documentos o productos requeridos para la fase, experiencias de proyectos anteriores), los datos de salida (resultados a utilizar por la fase posterior, experiencia acumulada, pruebas o resultados efectuados) y la estructura interna de la fase.

Figura 5. Esquema general de operación de una fase

- Entregables ("deliverables"). Son los productos intermedios que generan las fases. Pueden ser materiales (componentes, equipos) o inmateriales (documentos, software). Los entregables permiten evaluar la marcha del proyecto mediante comprobaciones de su adecuación o no a los requisitos funcionales y de condiciones de realización previamente establecidos.

Cada una de estas evaluaciones puede servir, además, para la toma de decisiones a lo largo del desarrollo del proyecto.

2.2.3.2.3. Tipos de Modelo de Ciclo de Vida

Las principales diferencias entre distintos modelos de ciclo de vida están en:

- El alcance del ciclo dependiendo de hasta dónde llegue el proyecto correspondiente. Un proyecto puede comprender un simple estudio de viabilidad del desarrollo de un producto, o su desarrollo completo o, llevando la cosa al extremo, toda la historia del producto con su desarrollo, fabricación, y modificaciones posteriores hasta su retirada del mercado.
- Las características (contenidos) de las fases en que dividen el ciclo. Esto puede depender del propio tema al que se refiere el proyecto (no son lo mismo las tareas que deben realizarse para proyectar un avión que un puente), o de la organización (interés de reflejar en la división en fases aspectos de la división interna o externa del trabajo).
- La estructura de la sucesión de las fases que puede ser lineal, con prototipado, o en espiral. Veámoslo con más detalle:

Ciclo de vida lineal

Es el más utilizado, siempre que es posible, precisamente por ser el más sencillo. Consiste en descomponer la actividad global del proyecto en fases que se suceden de manera lineal, es decir, cada una se realiza una sola vez, cada una se realiza tras la anterior y antes que la siguiente. Con un ciclo lineal es fácil dividir las tareas entre equipos sucesivos, y prever los tiempos (sumando los de cada fase).

Requiere que la actividad del proyecto pueda descomponerse de manera que una fase no necesite resultados de las siguientes (realimentación), aunque pueden admitirse ciertos supuestos de realimentación correctiva. Desde el punto de vista de la gestión (para decisiones de planificación), requiere también que se sepa bien de antemano lo que va a ocurrir en cada fase antes de empezarla.

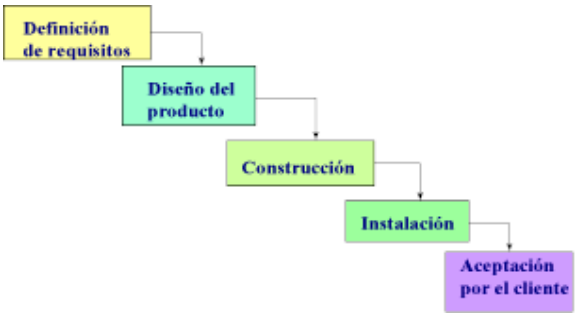


Figura 6. Ejemplo de ciclo lineal para un proyecto de construcción

Ciclo de Vida con Prototipado

A menudo ocurre en desarrollos de productos con innovaciones importantes, o cuando se prevé la utilización de tecnologías nuevas o poco probadas, que las incertidumbres sobre los resultados realmente alcanzables, o las ignorancias sobre el comportamiento de las tecnologías, impiden iniciar un proyecto lineal con especificaciones cerradas.

Si no se conoce exactamente cómo desarrollar un determinado producto o cuáles son las especificaciones de forma precisa, suele recurrirse a definir especificaciones iniciales para hacer un prototipo, o sea, un producto parcial (no hace falta que contenga funciones que se consideren triviales o suficientemente probadas) y provisional (no se va a fabricar realmente para clientes, por lo que tiene menos restricciones de coste y/o prestaciones). Este tipo de procedimiento es muy utilizado en desarrollo avanzado.

Figura 7. Ciclo Prototipazo

Ciclo de vida en espiral

El ciclo de vida en espiral puede considerarse como una generalización del anterior para los casos en que no basta con una sola evaluación de un prototipo para asegurar la desaparición de incertidumbres y/o ignorancias. El propio producto a lo largo de su desarrollo puede así considerarse como una sucesión de prototipos que progresan hasta llegar a alcanzar el estado deseado. En cada ciclo (espirales) las especificaciones del producto se van resolviendo paulatinamente.

A menudo la fuente de incertidumbres es el propio cliente, que aunque sepa en términos generales lo que quiere, no es capaz de definirlo en todos sus aspectos sin ver como unos influyen en otros. En estos casos la evaluación de los resultados por el cliente no puede esperar a la entrega final y puede ser necesaria repetidas veces.

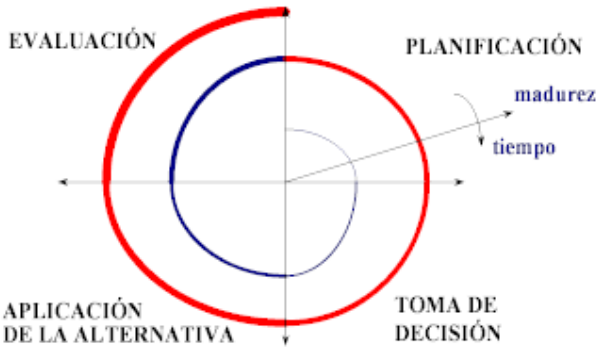


Figura 8. Ciclo de Vida en Espiral

Fases del Ciclo de Vida del Software

Análisis

- **Entrada**
  - Conocimiento del dominio de la aplicación, actividades de los usuarios, mercado, etc.
- **Actividades**
  - Identificar las necesidades del usuario
  - Análisis de viabilidad
  - Determinar los requisitos de la aplicación
- **Salida**
  - Documento de especificación de requisitos del software (ERS)

Diseño

- **Entrada**
  - Documento de especificación de requisitos del software
- **Actividades**
  - Establecer una(s) estrategia(s) de solución
  - Análisis de alternativas
  - Formalizar la solución
  - Descomponer y organizar la aplicación
  - Fijar descripciones de cada modulo
- **Salida**
  - Documento de diseño del software

Codificación

- **Entrada**
  - Documento de diseño del software
- **Actividades**
  - Creación del código fuente
  - Pruebas de unidades
- **Salida**
  - Código de módulos, probado

Validación

- **Entrada**
  - Código de módulos, probado
  - Documento de especificación de requisitos del software (validación)
- **Actividades**
  - Pruebas de integración
  - Pruebas de validación
- **Salida**
  - Aplicación completa, lista para usar

Aseguramiento de calidad del software[2]

[2] CUEVA, Juan Manuel "CALIDAD DEL SOFTWARE", Pág. 4

- El aseguramiento de calidad del software es el conjunto de actividades planificadas y sistemáticas necesarias para aportar la confianza en que el producto (software) satisfará los requisitos dados de calidad.
- El aseguramiento de calidad del software se diseña para cada aplicación antes de comenzar a desarrollarla y no después.
- Algunos autores prefieren decir garantía de calidad en vez de aseguramiento.
  - Garantía, puede confundir con garantía de productos
  - Aseguramiento pretende dar confianza en que el producto tiene calidad
- El aseguramiento de calidad del software está presente en
  - Métodos y herramientas de análisis, diseño, programación y prueba
  - Inspecciones técnicas formales en todos los pasos del proceso de desarrollo del software
  - Estrategias de prueba multiescala
  - Control de la documentación del software y de los cambios realizados
  - Procedimientos para ajustarse a los estándares (y dejar claro cuando se está fuera de ellos)
  - Mecanismos de medida (métricas)
  - Registro de auditorias y realización de informes

- Actividades para el aseguramiento- de calidad del software
  - Métricas de software para el control del proyecto
  - Verificación y validación del software a lo largo del ciclo de vida
  - Incluye las pruebas y los procesos de revisión e inspección
  - La gestión de la configuración del software

2.3. Bases de Datos

Una base de datos se puede definir como un conjunto de información relacionada que se encuentra agrupada ó estructurada. La base de datos es un sistema formado por un conjunto de datos almacenados en discos que permiten el acceso directo a ellos y un conjunto de programas que manipulen ese conjunto de datos.

2.4. Definiciones

Michel Adiba & Claude Delobel 1976. "Base de Datos es una colección de información que ha sido creada para satisfacer uno o más objetivos precisos".

James Martin 1977. "Una Base de Datos es la colección de ocurrencias de múltiples tipos de registro, conteniendo relaciones entre los registros, datos agregados y elementos de datos".

C.J. Date, 1981. ".Una Base de Datos es la colección de datos operacionales almacenados que son usados por el sistema de aplicaciones de una determinada empresa".

Howe, 1983. "Base de Datos es una colección no redundante de datos compatible entre diferentes aplicaciones".

Paul Jones & Robert Curtice, 1988. "BD es una colección de información organizada que a través de mecanismos, facilita el uso de la información".

En consecuencia, existen cuatro conceptos involucrados; estos son:

1. Coherencia, asociado a la validez de los datos.
2. Integridad, sobre el total de la información a representar.
3. Seguridad, como garantía de los datos en su representación.
4. Confidencialidad, otorgando acceso de acuerdo a los niveles de la organización.

2.4.1. Características de las Base de Datos

- Conjunto (colección) de datos.
- Datos interrelacionados y estructurados.
- Redundancia controlada.
- Independencia de datos y de procesos.

- Soporta múltiples usuarios y múltiples aplicaciones.
- La actualización y recuperación de datos debe asegurar Integridad, Seguridad y Confidencialidad de los datos.

2.4.2. Sistema de Gestión de la Base de Datos (SGBD)

Son Sistemas desarrollados que hace posible acceder a datos integrados que atraviesan los límites operacionales, funcionales u organizacionales de una empresa.

Objetivos en el uso de un sistema de gestión de base de datos:

- Oportunidad, asociado a la eficiencia y eficacia.
- Disponibilidad, permitiendo la accesibilidad de datos
- Consistencias (oportunidad + disponibilidad), como calidad de datos
- Evolución, para adaptarse al entorno
- Integridad, en el nivel de los datos así como el sistema.

Objetivos del sistema de gestión de base de datos que podemos identificar son:

- Independencia de datos
- Accesibilidad limitada
- Datos al día y sin redundancias
- Consistencia
- Interfaz única
- Entrada directa a los datos
- Recuperación por diferentes accesos

- Función completa de interrogantes
- Estandarización
- Seguridad

2.4.3. Los modelos de datos.

En el proceso de abstracción que conduce a la creación de una base de datos desempeña una función prioritaria el modelo de datos. El modelo de datos, como abstracción del universo de discurso, es el enfoque utilizado para la representación de las entidades y sus características dentro de la base de datos, y puede ser dividido en tres grandes tipos:

2.4.3.1.Modelos lógicos basados en objetos:

Los dos más extendidos son el modelo entidad-relación y el orientado a objetos.

Se basa en una percepción del mundo compuesta por objetos, llamados entidades, y relaciones entre ellos. Las entidades se diferencian unas de otras a través de atributos. El orientado a objetos también se basa en objetos, los cuales contienen valores y métodos, entendidos como órdenes que actúan sobre los valores, en niveles de anidamiento. Los objetos se agrupan en clases, relacionándose mediante el envío de mensajes. Algunos autores definen estos modelos como "modelos semánticos".

2.4.3.2.Modelos lógicos basados en registros:

El más extendido es el relacional, mientras que los otros dos existentes, jerárquico y de red, se encuentran en retroceso. Estos modelos se usan para especificar la estructura lógica global de la base de datos, estructura-

da en registros de formato fijo de varios tipos. El modelo relacional representa los datos y sus relaciones mediante tablas bidimensionales, que contienen datos tomados de los dominios correspondientes. El modelo de red está formado por colecciones de registros, relacionados mediante punteros o ligas en grafos arbitrarios. El modelo jerárquico es similar al de red, pero los registros se organizan como colecciones de árboles. Algunos autores definen estos modelos como "modelos de datos clásicos".

2.4.3.3. Modelos físicos de datos

Muy poco usados, son el modelo unificador y el de memoria de elementos. Algunos autores definen estos modelos como "modelos de datos primitivos".

2.5. Motor de base de Datos – MYSQL

Los sistemas de administración de datos, son programas utilizados para almacenar y recuperar datos. Un programa de administración de base de datos tiene un número de componentes importantes. El interfaz de usuario es el componente que controla interacción entre el usuario y el programa. También controla la forma unos datos de usuario de vistas a través de formularios e informes. El motor de base de datos es el componente que en realidad controla los datos. El usuario normalmente no interactúa directamente con el motor de base de datos.

2.5.1. Introducción.

MySQL, es la más popular base de datos SQL de código abierto, proporcionada por MySQL AB. MySQL AB es una compañía comercial que

construye su negocio proporcionando los servicios alrededor de la base de datos de MySQL.

MySQL es un sistema de dirección de base de datos. Una base de datos es una colección estructurada de datos. Puede ir desde algo como una simple lista de compras a una galería de arte o las inmensas cantidades de información en una red corporativa. Para agregar, acceda, y los datos del proceso guardaron en una base de datos de la computadora, usted necesita un sistema de dirección de base de datos como MySQL. Desde que las computadoras están muy excelentes en el manejo de millones de datos, la dirección de la base de datos juega un papel central en la computación, como las utilidades autosuficientes o como las partes de otras aplicaciones.

MySQL es un sistema de dirección de base de datos correlativo.
Una base de datos correlativa guarda los datos en tablas separadas en lugar de poner todos los datos en un solo lugar. Esto agrega velocidad y flexibilidad. Las Tablas se unen por relaciones definidas lo que hace posible combinar los datos de varias tablas.

MySQL es el Software de la Código Abierto. Código Abierto hace posible que cualquiera lo use y modifique. Cualquiera puede descargar MySQL de la Internet y puede usarlo sin pagar algo. Alguien tan inclinado puede estudiar el código de la fuente y puede cambiarlo para encajar sus necesidades.

2.5.1.1.Características (versión 4.0).

Inicialmente, MySQL carecía de elementos considerados esenciales en las bases de datos relacionales, tales como integridad referencial y transaccio-

nes. A pesar de ello, atrajo a los desarrolladores de páginas Web con contenido dinámico, justamente por su simplicidad.

Poco a poco los elementos de los que carecía MySQL están siendo incorporados tanto por desarrollos internos, como por desarrolladores de software libre. Entre las características disponibles en las últimas versiones se puede destacar:

- Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente.
- Disponibilidad en gran cantidad de plataformas y sistemas.
- Diferentes opciones de almacenamiento según si se desea velocidad en las operaciones o el mayor número de operaciones disponibles.
- Transacciones y claves foráneas.
- Conectividad segura.
- Replicación.
- Búsqueda e indexación de campos de texto.

2.5.1.2.Características de la versión 5.0.22

- Un amplio subconjunto de ANSI SQL 99, y varias extensiones.
- Soporte a multiplataforma
- Procedimientos almacenados
- Triggers
- Cursors
- Vistas actualizables

- Soporte a VARCHAR
- INFORMATION_SCHEMA
- Modo Strict
- Soporte X/Open XA de transacciones distribuidas; transacción en dos fases como parte de esto, utilizando el motor InnoDB de Oracle
- Motores de almacenamiento independientes (MyISAM para lecturas rápidas, InnoDB para transacciones e integridad referencial)
- Transacciones con los motores de almacenamiento InnoDB, BDB Y Cluster; puntos de recuperación(savepoints) con InnoDB
- Soporte para SSL
- Query caching
- Sub-SELECTs (o SELECTs anidados)
- Indexing y buscando campos de texto completos usando el motor de almacenamiento MyISAM
- Embedded database library
- Soporte completo para Unicode
- Conforme a las reglas ACID usando los motores InnoDB, BDB y Cluster
- Shared-nothing clustering through MySQL Cluster

2.5.1.3.Características distintivas.

Las siguientes características son implementadas únicamente por MySQL:

- Múltiples motores de almacenamiento (MyISAM, Merge, InnoDB, BDB, Memory/heap, MySQL Cluster, Federated, Archive, CSV, Bla-

ckhole y Example en 5.x), permitiendo al usuario escoger la que sea más adecuada para cada tabla de la base de datos.

- Agrupación de transacciones, reuniendo múltiples transacciones de varias conexiones para incrementar el número de transacciones por segundo.

2.5.2. Creación y Uso de una Base de Datos.

Todos los conjuntos que forman un modelo completo forman una base de datos. Para crear una base de datos de usa la sentencia "CREATE DATABASE", aquí un ejemplo:

```
"CREATE DATABASE [IF NOT EXISTS] db_name"
Mysql> CREATE DATABASE egresados;
```

CREATE DATABASE crea una base de datos con el nombre dado. Se dan reglas para los nombres de la base de datos aceptables. Un error ocurre si la base de datos ya existe y usted no especificó IF NOT EXISTS.

Las bases de datos en MySQL se implementan como directorios que contienen archivos que corresponden a las Tablas en la base de datos, Porque no hay ninguna tabla en una base de datos cuando se crea inicialmente, la sentencia CREATE DATABASE crea un directorio bajo el directorio de MySQL.

2.5.3. Creación y Eliminación de Tablas.

La tabla, un contenedor estructurado de datos, es el concepto básico en una base de datos correlativa. Antes de que se pueda empezar a agregar los

datos a una tabla, se debe definir la estructura de la tabla. Considerando el siguiente diseño:

```
+---------------------------------+
|             people              |
+-------------+-------------------+
| name        | char(10) not null |
| address     | text(100)         |
| id          | int               |
+-------------+-------------------+
```

No sólo hace que la tabla contenga los nombres de las columnas, pero también contiene los tipos de cada campo así como cualquier información adicional que los campos puedan tener. Los datatype de un campo especifican qué tipo de datos puede sostener cada campo. Los datatypes de SQL son similares a los datatypes en otros lenguajes de programación.

La sintaxis general para la creación de las tablas es:

```
CREATE TABLE nombre_tabla (columna_nombre1 type [modifiers]
                    [, columna_name2 type [modifiers]]
)
```

Una columna es la unidad individual de datos dentro de una tabla. Una tabla puede tener cualquier número de columnas, pero las tablas grandes pueden ser ineficaces. Aquí es donde el buen diseño de la base de datos, se vuelve una habilidad importante. Creando las tablas propiamente normalizadas, usted puede unir las tablas para realizar una sola búsqueda de los datos en más de una tabla.

Como la mayoría de las cosas en la vida, la destrucción es muy más fácil como la creación. La sentencia para eliminar una tabla de la base de datos es:

```
DROP TABLE nombre_tabla
```

Esta sentencia quitará completamente todos los rastros de la tabla de la base de datos. MySQL quitará todos los datos dentro de la tabla destruida. Si no se tiene algún respaldo de la mesa, no se podrá recuperar ningún dato con esta acción.

2.5.4. Drop Procedure y Drop Function

Este comando se usa para borrar un procedimiento o función almacenado. Esto es, la rutina especificada se borra del servidor. Debe tener el permiso ALTER ROUTINE para las rutinas desde MySQL 5.0.3. Este permiso se otorga automáticamente al creador de la rutina.

```
DROP {PROCEDURE | FUNCTION} [IF EXISTS] sp_name
```

La cláusula IF EXISTS es una extensión de MySQL . Evita que ocurra un error si la función o procedimiento no existe. Se genera una advertencia que puede verse con SHOW WARNINGS.

2.5.5. Show Create Procedure y Show Create Function[3].

```
SHOW CREATE {PROCEDURE | FUNCTION} sp_name
```

Este comando es una extensión de MySQL. Similar a SHOW CREATE TABLE, retorna la cadena exacta que puede usarse para recrear la rutina nombrada.

```
mysql> SHOW CREATE FUNCTION test.hello\G
```

[3] www.MySQLhispano.org

```
*************************** 1. row ***************************
       Function: hello
       sql_mode:
Create Function: CREATE FUNCTION `test`.`hello`(s CHAR(20))
RETURNS CHAR(50)
RETURN CONCAT('Hello, ',s,'!')
```

Con MySQL, se puede especificar más de una tabla para eliminarla, separando los nombres de la tabla con comas. Por ejemplo, DROP TABLE estudiantes, materias, se eliminaran las dos tablas nombradas.

2.5.6. Show Procedure Status y Show Function Status

```
SHOW {PROCEDURE | FUNCTION} STATUS [LIKE 'pattern']
```

Este comando es una extensión de MySQL . Retorna características de rutinas, como el nombre de la base de datos, nombre, tipo, creador y fechas de creación y modificación. Si no se especifica un patrón, le lista la información para todos los procedimientos almacenados, en función del comando que use.

```
mysql> SHOW FUNCTION STATUS LIKE 'hello'\G
*************************** 1. row ***************************
           Db: test
         Name: hello
         Type: FUNCTION
      Definer: testuser@localhost
     Modified: 2004-08-03 15:29:37
      Created: 2004-08-03 15:29:37
Security_type: DEFINER
      Comment:
```

Valores Nulos. Al definir cada columna se puede decidir si contendrá o no valores nulos. Aquellas columnas que son o forman parte de una clave primaria no pueden contener valores nulos. La opción por defecto es que se permitan valores nulos, "NULL", y para que no se permitan, se usa "NOT NULL".

Valores por Defecto. Cada columna también puede ser definida con valores por defecto. El valor por defecto se asignará de forma automática a una columna cuando no se especifique un valor determinado al añadir filas.

Claves primarias. Se puede definir una clave primaria sobre una columna, usando la palabra clave "KEY" o "PRIMARY KEY". Sólo puede existir una clave primaria en cada tabla, y la columna sobre la que se define una clave primaria no puede tener valores NULL. Si esto no se especifica de forma explícita, MySQL lo hará de forma automática.

Columnas Auto Incrementadas. Si al insertar una fila se omite el valor de la columna auto incrementada o si se inserta un valor nulo para esa columna, su valor se calcula automáticamente, tomando el valor más alto de esa columna y sumándole una unidad. Esto permite crear, de una forma sencilla, una columna con un valor único para cada fila de la tabla. Generalmente, estas columnas se usan como claves primarias. MySQL está optimizado para usar valores enteros como claves primarias.

Comentarios. Adicional ni ente, al crear la tabla, es posible añadir un comentario a cada columna. Este comentario sirve como información adicional 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.

2.5.7. Manejo de Datos.

2.5.7.1.Ingreso de Datos.

Los datos ingresados a una tabla es uno de los conceptos más sinceros en MySQL. Se lo realiza con la sentencia "INSERT":

```
INSERT INTO nombre_tabla (columna1, columna2, ..., columnaN)
VALUES (valor1, valor2, ..., valorN)
```

Al insertar los datos en los campos numéricos, se puede insertar el valor como es; para todos los otros campos, se deben envolverlos en una sola cita. Por ejemplo, para insertar una fila de datos en una tabla de direcciones, se podría seguir el orden siguiente:

```
INSERT INTO direcciones (nombre, dirección, ciudad, provincia, teléfono, edad)
VALUES('Aleman', '123 cevallos', 'Ambato', 'Tungurahua',
       '2855-1234', 26)
```

Además, si no necesario asignar un valor concreto para las columnas, se podrá asignar un valor por defecto --` \' by default—

2.5.7.2.Actualización de Datos.

La inserción de nuevas filas en una base de datos es solo el comienzo del uso de la base de datos, a menos que la base de datos sea "solo de lectura", se necesitará también hacer los cambios periódicos a los datos. El estándar de modificación para MySQL es el siguiente:

```
UPDATE nombre_tabla
SET columna1=valor1, columna2=valor2, ..., columnaN=valorN
[WHERE where_definition]
```

Bajo MySQL, el valor que se asigna a una columna debe se un literal del tipo de dato de la columna. MySQL, en el contraste, le permite que calcule el valor asignado, incluso se podrá calcular el valor basado en un valor en otra columna:

```
UPDATE año
SET fin_año = inicio_año+5
```

Esta sentencia pone el valor en la columna fin _año igual al valor en la columna inicio_año más 5 para cada fila en esa mesa.

2.5.7.3. Borrado.

Anulando los datos es un funcionamiento muy delicado. Se podrá especificar la tabla de la cual se quiere anular, seguida por simplemente la cláusula WHERE que identifica las filas que se quiere anular:

```
DELETE FROM table_name [WHERE clause]
```

Como con otros comandos que aceptan la cláusula WHERE, la cláusula WHERE es optativa. En este caso se omite la cláusula WHERE, se anulará todos los archivos en la tabla.

2.5.7.4.SQL Embebido.

Es el método de combinación de códigos en un lenguaje de programación con la capacidad de manipulación de base de datos por SQL.

Permite a los programadores integrar sentencias SQL en programas escritos en Pascal, Cobol, C, etc.

Las sentencias SQL integradas son sentencias SQL escritas dentro del código de programación y son preprocesadas por un preprocesador SQL antes de que la aplicación sea compilada.

2.6. Programación Web

2.6.1. Introducción

La Web nació alrededor de 1989 a partir de un proyecto del CERN, en el que Tim Berners-Lee construyó el prototipo que dio lugar al núcleo de lo que hoy es la World Wide Web. La intención original era hacer más fácil el compartir textos de investigación entre científicos y permitir al lector revisar las referencias de un artículo mientras lo fuera leyendo.

La funcionalidad elemental de la Web se basa en tres estándares: El *Localizador Uniforme de Recursos* (URL), que especifica cómo a cada página de información se asocia una "dirección" única en donde encontrarla; el *Protocolo de Transferencia de Hipertexto* (HTTP), que especifica cómo el navegador y el servidor intercambian información en forma de peticiones y respuestas, y el *Lenguaje de Marcación de Hipertexto* (HTML), un método para codificar la información de los documentos y sus enlaces.

La World Wide Web permite una manera más organizada el acceso a la información disponible en Internet, presentando una interfaz amigable con el usuario.

El surgimiento de la World Wide Web ha ayudado a un crecimiento considerable de Internet en la actualidad. Compañías pequeñas, empresas grandes, estados, gobiernos de distintos países, universidades, bibliotecas, están presentes en Internet.

2.6.2. Funcionamiento de un Sitio Web.

La estructura del 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*" o "*browser*" como por ejemplo Microsoft Internet Explorer o Netscape Navigator; mediante los Navegadores podemos acceder a hojas de calculo, base de datos, vídeo, sonido y todas las posibilidades más avanzadas. Pero el diseño de páginas debe mantener un equilibrio entre utilizar todas las capacidades y la posibilidad de ser leídas por cualquier tipo de Navegador.

Para la plataforma cliente-servidor que necesita de un protocolo que especifique de que manera se comunican o intercambian datos entre cliente y servidor, se utiliza el protocolo HTTP para sitio Web, que funciona encapsulado sobre el protocolo TCP/IP.

2.6.3. Protocolo HTTP.

El protocolo de Transferencia de Hipertexto HTTP *(HyperText Transfer Protocol*) es el protocolo usado en cada transacción de la Web del lado del cliente. El hipertexto es el contenido de las páginas Web y el protocolo de transferencia es el sistema mediante el cual se envían las peticiones de acce-

der a una página Web y la respuesta de esta, remitiendo la información que se verá en pantalla. También sirve el protocolo para enviar información adicional en ambos sentidos, cliente-servidor, como formularios con mensajes y otros similares.

El protocolo HTTP es un protocolo sin estado; 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 como sigue:

```
<Linea inicial>
Header-1: value-1
...
Header-n: value-n

<Cuerpo del mensaje (Opcional)>
```

De este modo el cliente y el servidor se comunican con varios ciclos de *request-response* como sean necesarios. Se debe indicar que por cada pedido del cliente se debe iniciar una nueva conexión debido a que el protocolo HTTP no admite que en una misma conexión se realicen más de un pedido al servidor, por ejemplo, en una página HTML simple con tres imágenes es normal que se efectúen cuatro conexiones al servidor: una para la página y mas tarde una por cada imagen.

HTTP es un protocolo sin estado, es decir, que no guarda ninguna información sobre conexiones anteriores. Al finalizar la transacción todos los datos se pierden. Por esto se popularizaron las cookies, que son pequeños ficheros guardados en el propio ordenador que puede leer un sitio Web al establecer conexión con él y de esta forma reconocer a un visitante que ya estuvo

en ese sitio anteriormente. Gracias a esta identificación, el sitio Web puede almacenar gran número de información sobre cada visitante.

2.6.4. Ciencias Aplicadas para el Desarrollo de Aplicaciones Web.

Para el desarrollo de aplicaciones y dotar a las páginas Web de funcionalidad, existe la posibilidad de trabajar en el lado del cliente y en el lado del servidor con las siguientes variantes:

Programación en el cliente:

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

Programación en el servidor:

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

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

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

- El browser interpreta el código enviado por el servidor y realiza una determinada acción.

La programación del lado del cliente tiene la ventaja que la ejecución de la aplicación se encarga al cliente con lo cual se evita sobre cargar al servidor de trabajo. El servidor sólo envía el código y el browser lo interpreta. La gran desventaja de esta metodología es que el código que el servidor envía sea sensible a que el browser no las pueda interpretar. El usuario puede decidir deshabilitar una funcionalidad del browser que es necesaria para que se ejecute un determinado servicio o también los distintos tipos de browsers pueden interpretar el mismo código de distintas formas, así sucede con Netscape y Microsoft Explorer

La Programación del lado del servidor tiene como gran ventaja que se pueden hacerse varias tareas 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 servidor se sobrecarga de trabajo, ya que, además de servir páginas es responsable de ejecutar aplicaciones. A menudo esto redunda en los requisitos de hardware mayores a medida que el servidor ejecuta más y más servicios.

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 Standard 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.

2.6.5. Programación del lado del Servidor.

Un servidor Web se encarga de mantenerse a la espera de peticiones HTTP llevada a cabo por un cliente HTTP que solemos conocer como navegador. El navegador realiza una petición al servidor y éste le responde con el contenido que el cliente solicita. Para el desarrollo de aplicaciones del lado del servidor existen 3 grandes metodologías: el *protocolo **CGI***, utilizar una *API provista por el Web- Server* o bien utilizar un *módulo del servidor Web*

2.6.5.1.Protocolo CGI.

Common Gateway Interface (Interfaz Común de Pasarela») es una importante tecnología de la World Wide Web que permite a un cliente o browser solicitar datos de un programa ejecutado en un servidor Web. CGI especifica un estándar para transferir datos entre el cliente y el programa. Es un mecanismo de comunicación entre el servidor Web y una aplicación externa.

Las aplicaciones CGI fueron una de las primeras maneras prácticas de crear contenido dinámico para las páginas Web. En una aplicación CGI, el servidor Web pasa las solicitudes del cliente a un programa externo. La salida de dicho programa es enviada al cliente en lugar del archivo estático tradicional.

CGI ha hecho posible la implementación de funciones nuevas y variadas en las páginas Web, de tal manera que esta interfaz rápidamente se volvió un estándar, siendo implementada en todo tipo de servidores Web.

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 servidor por método post o get se pasan a variables de ambiente.
- El servidor redirecciona su salida Standard al browser.
- El servidor crea un proceso que tiene la salida Standard redireccionada.
- El servidor ejecuta en el proceso creado la aplicación deseada.
- Se ejecuta la aplicación.

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

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

Esta tecnología tiene la ventaja de correr en el servidor cuando el usuario lo solicita por lo que es dependiente del servidor y no de la computadora del usuario.

Un documento HTML es estático, lo que significa que existe en un estado constante; es un archivo de texto que no cambia. Un script CGI por otro lado, es ejecutado en tiempo real, lo que permite que regrese información dinámica. Por ejemplo, digamos que quieres conectar tus bases de datos de Unix al World Wide Web para permitir que las personas de todo el mundo la manipulen. Básicamente, lo que debes hacer es crear un script CGI que será ejecutado por el servidor para transmitir información al motor de la base de datos, recibir los resultados y mostrárselos al cliente. Este es un ejemplo sencillo que muestra donde el CGI tiene sus orígenes

.Los programas que maneja el CGI pueden estar compilados en diferentes lenguajes de programación. El más popular para el desarrollo de contenidos Web es el lenguaje Perl de distribución gratuita, aunque también podemos mencionar: C, C++ y Java.

El funcionamiento de esta tecnología es muy sencilla. Los scripts residen en el servidor, donde son llamados, ejecutados y regresa información de vuelta al usuario.

Para una mejor aclaración presentamos la siguiente gráfica:

Figura 9. Protocolo CGI

2.6.5.2.Uso de una API del Servidor.

Otras de las técnicas es el uso de una API (Application Programming Interface - Interfaz de Programación de Aplicaciones) provista por el servidor Web para desarrollar aplicaciones, es decir que el Web-Server provee un lenguaje en el cual se pueden desarrollar aplicaciones, es un conjunto de especificaciones de comunicación entre componentes software.

Representa un método para conseguir abstracción en la programación, generalmente (aunque no necesariamente) entre los niveles o capas inferiores y los superiores del software. Uno de los principales propósitos de una API consiste en proporcionar un conjunto de funciones de uso general, por ejemplo, para dibujar ventanas o iconos en la pantalla. De esta forma, los programadores se benefician de las ventajas de la API haciendo uso de su funcionalidad, evitándose el trabajo de programar todo desde el principio. Las APIs asimismo son abstractas: el software que proporciona una cierta API generalmente es llamado la implementación de esa API.

Este esquema, como podemos apreciar, es mucho más eficiente que el anterior ya que el Web-Server es el encargado de ejecutar 'las aplicaciones en forma directa sin necesidad de crear un proceso. Las desventajas son sin embargo importantes: en primer lugar las aplicaciones creadas en este marco no son portables ya que sólo pueden ejecutarse en un Web-Server determinado, esto es una gran desventaja frente a las aplicaciones CGI que podían una vez desarrolladas ejecutarse en cualquier servidor. La segunda gran desventaja es que frecuentemente un error de programación de

una aplicación podría ocasionar que el Web-Server deje de funcionar, genere un error, se cuelgue, pierda memoria u otros problemas. Esto ocasiona que este tipo de aplicación no sea confiable.

2.6.5.3. Uso de Módulos en el Web-Server.

La ejecución de aplicaciones consiste en anexar a un Web-Server módulos que permitan interpretar un determinado lenguaje. De esta forma se logra rapidez en los procesos ya que el servidor Web no necesita crear un nuevo proceso para cada aplicación que requiera. Las aplicaciones son portables puesto que son desarrolladas en un lenguaje Standard.

Estas aplicaciones son confiables y en el caso de que puedan producir un error, este se da por lo general en el lenguaje en que están diseñadas, pero si el módulo es sólido dichos errores no pueden comprometer al Servidor Web.

2.6.6. Lenguaje de Programación Web

2.6.6.1. HTML

El HTML, acrónimo inglés de HyperText Markup Language (lenguaje de marcas hipertextuales), Es un lenguaje de marcación diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas Web. Su funcionalidad es la de representar cualquier clase de información que se encuentre almacenada en una página Web.

HTML no es propiamente un lenguaje de programación como Visual Basic, Java, etc., sino 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 como esté entendida.

2.6.6.2. JAVASCRIPT

JavaScript es un lenguaje interpretado, es decir, que no requiere compilación, orientado a las páginas Web, con una sintaxis semejante a la del lenguaje Java y el Lenguaje C.

Al contrario que Java, JavaScript no es un lenguaje orientado a objetos propiamente dicho, ya que no dispone de Herencia, es más bien dirigida por eventos.

Estará listo para actuar en cuanto un evento (un clic en un botón, por ejemplo) sea ejecutado. Aún así javascript implementa una sencilla interfaz de objetos/propiedades/métodos.

2.6.6.3. 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 la sintaxis como la manera de trabajar están basados en Visual Basic, sin embargo, no todo lo se puede hacer en está se lo puede hacer en VBScript, ya que este ultimo 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.

2.6.6.4. DHTML

DHTML (Dynamic HyperText Markup Language) 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.

2.6.6.5. CSS

(Cascading Style Sheets, CSS) Es una tecnología que permite crear páginas Web de una manera exacta. Gracias a este lenguaje podemos realizar varias cosas que antes no se podían, como incluir márgenes, diferentes tipos de letras, fondos, colores, etc.

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

2.6.6.6. PERL

PERL (Practical Extracting and Reporting Lenguaje) es un lenguaje de propósito general originalmente desarrollado para la manipulación de texto y que ahora es utilizado para un amplio rango de tareas incluyendo administración de sistemas, desarrollo Web, programación en red, desarrollo de GUI y más.

Se previó que fuera práctico (facilidad de uso, eficiente, completo) en lugar de hermoso (pequeño, elegante, mínimo). Sus principales características son que es fácil de usar, soporta tanto la programación estructurada como la programación orientada a objetos y la programación funcional, tiene incorporado un poderoso sistema de procesamiento de texto y una enorme colección de módulos disponibles.

2.6.6.7. ASP

ASP (Active Server Pages, páginas de servidor activas) es una tecnología de Microsoft para hacer scripts del lado del servidor. ASP se escribe en la misma página Web, Utilizando el lenguaje Visual Basic Script o Javascript

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 envió de correo, generar gráficas dinámicas, etc.

2.6.6.8. PHP

Es un lenguaje de programación usado generalmente para la creación de contenido para sitios Web. Las siglas significan "PHP Hypertext Preprocessor" (inicialmente PHP Tools, o, Personal Home Page Tools), y se trata de un lenguaje interpretado usado para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios Web.

El fácil uso y la similitud con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores experimentados crear aplicaciones complejas con una curva de aprendizaje muy suave. También les permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones y prácticas.

Debido al diseño de PHP, también es posible crear aplicaciones con una interfaz gráfica para el usuario (también llamada GUI). También es independiente de plataforma, puesto que existe un modulo de PHP para casi cualquier servidor Web. Esto hace que cualquier sistema pueda ser compatible con el lenguaje lo cual es 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

2.6.6.9. JSP

Java Pages Server (JSP), en el campo de la Informática, es una tecnología para crear aplicaciones Web

La JSP, es una tecnología Java que permite a los programadores generar contenido dinámico para Web, en forma de documentos HTML, XML, o de otro tipo. Las JSP's permite al código Java y a algunas acciones predefinidas ser incrustadas en el contenido estático del documento Web.

El Motor de las páginas JSP está basado en los servlets de Java (Programas en Java destinados a ejecutarse en el servidor)

2.6.6.10. XML

XML, (eXtensible Markup Language), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML.

XML es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que

permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

El XML se creó para que cumpliera varios objetivos:

- Que fuera idéntico a la hora de servir, recibir, y procesar la información del HTML para aprovechar toda la tecnología implantada de este.
- Que fuera normal y conciso desde el punto de vista de los datos y la manera de guardarlos.
- Que fuera extensible, para que lo puedan utilizar en todos los campos del conocimiento.
- Que fuese fácil de leer y editar e implementar.
- Que fuese fácil de implantar, programar y aplicar a los distintos sistemas.

2.6.7. Lenguaje de Programación PHP

2.6.7.1.Introducción

El lenguaje PHP es un lenguaje de programación de estilo clásico, con esto se quiere decir que es un lenguaje de programación con variables, sentencias condicionales, bucles, funciones. No es un lenguaje de marcas como podría ser HTML[4], XML[5] o WML[6]. Está más cercano a JavaScript o a C.

Pero a diferencia de Java o JavaScript que se ejecutan en el navegador, PHP se ejecuta en el servidor, por eso nos permite acceder a los re-

[4] HyperText Markup Language
[5] eXtensible Markup Language
[6]Wireless Markup Language

cursos que tenga el servidor como por ejemplo una base de datos. El programa PHP es ejecutado en el servidor y el resultado enviado al navegador. El resultado es normalmente una página HTML pero igualmente podría ser una pagina WML.

Figura 10. Proceso de paginas PHP

Al ser PHP un lenguaje que se ejecuta en el servidor no es necesario que el navegador lo soporte, es independiente del navegador, pero sin embargo para que las páginas PHP funcionen, el servidor donde están alojadas debe soportar PHP.

2.6.7.1.1. Generalidades

Una de las características más importantes de PHP es que permite combinar el código HTML y el código PHP en una misma página (de extensión php), así:

```
<html>
<Head><Title> PAGINA </Head></Title>
<Body>
Página de pruebas. <br/>
<?php
print("Esta línea es generada en php <br/>");
?>
</body>
```

```
</html>
```

Para este ejemplo se debe guardar en un archivo de extensión "php", el cual se genera automáticamente por el interprete de PHP cuando el browser[7] envía un pedido. Para este pedido se genera el siguiente ciclo:

- El browser envía un pedido de un archivo con extensión php.
- El Servidor analiza que la extensión del request[8] es .php, obtiene al archivo y lo envía al interprete php.
- El interprete php del Web-Server divide el archivo en busca de tags <? ?> y procesa todo lo que se encuentra entre dichos tags (puede haber varias apariciones de los tags en un mismo archivo), todo aquello que esta fuera de los tags se envía al browser sin interpretar.
- El resultado combinado de aquello que no debe interpretarse y el resultado del código interpretado se envía al browser.

En el ejemplo el browser recibirá:

```
<html>
<Head><Title> PAGINA </Head></Title>
<Body>
Página de pruebas. <br/>
Esta línea es generada en php <br/>
</body>
</html>
```

[7] Navegador de Internet
[8] Respuesta del Navegador

2.6.7.2.Generación de Sitios Dinámicos con PHP.

El lenguaje PHP nos permite realizar "includes" dentro de un script PHP y de esta forma se puede modularizar la página o el layout de las páginas en varios módulos, también se pueden aplicar componentes re usables de la misma página en otras.

Para la generación de sitios dinámicos es con la definición de templates.

2.6.7.3.Funcionalidad.

- Funciones de calendario y manipulación de calendarios usando MCAL. (Modular Calendar Access Library).
- Programación orientada a objetos.
- Funciones para manejo de directorios.
- Funciones de encriptación de datos.
- Funciones de acceso al filesystem.
- Funciones para manejo de FTP.
- Funciones para creación de archivos PDF.
- Parser de documentos XML. (eXtensible Markup Language, es un formato de información para el intercambio de documentos estructurado en la "Web").
- WDDX (Standard para el intercambio de estructuras de datos entre distintos lenguajes de programación utilizando XML).
- Funciones de compresión de datos.
- Manejo de archivos DBM. (Data Base Management).
- Funciones de hashing.

- Generación dinámica de imágenes.
- Manejo de cuentas de mail IMAP y POP3.
- Funciones para envió de mail.
- Funciones matemáticas.
- Acceso a bases de datos (Mysql, Oracle, Postgress, Sybase, etc.).
- Manejo de expresiones regulares.
- Manejo de sesiones.

2.6.7.4.Introducción al Lenguaje PHP

PHP es un lenguaje no posicional, por lo que no importa la columna en la cual se comience a escribir el código. Tampoco influye sobre el código la cantidad de saltos de línea que se coloquen, ni la cantidad de espacios.

La forma en la que se separan las distintas sentencias es mediante la utilización de ";". En PHP cada sentencia debe finalizar con ";". Se puede escribir más de una sentencia en la misma línea siempre y cuando las mismas se encuentren separadas con "; ".

2.6.7.5. Tipos de Datos.

PHP soporta los siguientes tipos de datos:

- Vectores
- Enteros
- Strings
- Binarios de punto flotante

- Objetos

El tipo de dato de una variable no es decidido por el programador sino que lo decide el lenguaje en tiempo de ejecución, la instrucción ***settype*** puede usarse para forzar el tipo de dato de una variable en ciertos casos en que esto sea necesario. Todas las variables en php se denotan utilizando el signo '$' precediendo al nombre de la variable.

2.6.7.6.Comentarios.

En PHP hay 3 formas distintas de incluir comentarios:

- Al estilo de C en donde el comentario empieza y termina delimitado por barra asterisco y asterisco barra.

```
/* comentario */
```

- Usando doble barra

```
// comentario
```

- Usando el signote número

```
# comentario
```

En los dos últimos casos, las variantes del comentario empieza en donde se encuentra el *"/1"* o el "#" y termina cuando termina la línea.

2.6.7.7. Operadores.

`$a + $b;` suma

`$a -- $b;` resta

`$a * $b;` multiplicación

$a / $b; división

$a % $b; módulo

$a++; pos-incremento

++$a; pre-incremento

$a --; pos-decremento

--$a; pre-decremento

2.6.7.8.Asignación.

$a=5; Asigna 5 a $a

$a=$b; Asigna el valor de $b a $a

$b = ($c=6); Asigna 6 a $c y a $b

$a += 5; Suma y asigna 5 a $a

$x.= "hola"; Concatena $x con "hola"

2.6.7.9. Constantes.

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

```
Define("PI",3.14151692)
```

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

2.6.7.10. Comparaciones.

$a == $b; true si $a igual $b

$a == $b; true si $a igual a $b y además son del mismo tipo

$a >= $b; mayor o igual

$a <= $b; menor o igual

$a !=$b; true si a y b son distintos

2.6.7.11. Operador @.

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

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

2.6.7.12. Operadores Lógicos.

$a && $b; true si $a es verdadero y $b es verdadero

$a || $h; true si $a es verdadero o $b es verdadero

$a xor $b; *or* exclusivo

!$a; true si $a es falso *(NOT)*

2.6.7.13. Estructuras de Control.

- **If:**

```
if (expresión) sentencia;
if (expresión) {
   sentencias;
}
if (expresión) {
   sentencias;
}else{
   sentencias;
if (expresión) {
   sentencias;
}elseif(expresión){
   sentencias;
}else(expresión){
   sentencias;
}
```

- **While:**

```
while (expresión)
   sentencias;
}
do {
   sentencias;
} while(expresión)
```

- **For:**

```
for ($i = 0; $i < 5 $i++) {
print ("$i");
}
```

- **Foreach:**

```
foreach ($vector as $variable) {
   sentencias;
}
foreach ($vector as $clave => $valor) {
   sentencias;
}
```

- **break;**

```
for ($i=O;$i<6;$i++) {
if ($i == $b) break; -> Sale del ciclo si $i es gua1 a $b
}
```

- **Switch:**

```
switch ($variable) {
   case valor:
         sentencias;
   break;
case valor2:
   sentencias;
   break;
default:
   sentencias;
   break;
}
```

2.6.7.14. Funciones.

Para la definición de subrutinas en PHP se emplea lo siguiente:

```
function prueba ($a, $b) {
   $r $a + Sb;
   returri $r;
}
```

Para llamar a la función solo se digitara:

```
$x = prueba (4,6);
```

Los parámetros que recibe la función pueden ser enteros, flotantes, strings, vectores u objetos, es decir, cualquiera de los tipos de datos soportado por PHP. El valor devuelto por la función también puede ser cualquier tipo de datos de PHP.

Es posible asignar un valor default a los parámetros que recibe una función de forma tal que cuando se invoca la función si se ignora el parámetro el mismo es asignado al default.

```
function prueba ($a=2, $b=3, $c=5) {sentencias;}
```

2.6.7.15. Manejo de Base de Datos (MySQL).

Una de las características más importantes de PHP es su integración con diversos motores de base de datos. El lenguaje PHP está indebido para generar en forma sencilla páginas Web dinámicas a partir de información almacenada en bases de datos.

2.6.7.16. Conexión a la Base.

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

La función realiza la conexión a la base MySQL y devuelve ***false*** si hubo algún error en la conexión o un ***link*** a la conexión a la base 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 se 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 el motor de la base de datos.

2.6.7.17. Selección 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 se encuentre algún error, como por ejemplo la inexistencia de la base de datos.

2.6.7.18. Queries a la Base de Datos.

```
$resuit = mysql_query (query, db_link);
```

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

2.6.7.19. Cantidad de Filas Consultadas o Modificadas.

```
$cantidad = mysq_num_rows ($resuit);
```

Esta función devuelve la cantidad de filas que se obtuvieron luego de ejecutar una instrucción de consulta como por ejemplo la función select.

```
$cantidad = mysq_affected_rows (db_link);
```

Devuelve cuantos registros fueron afectados por un query con insert, update, o delete.

2.6.7.20. Obtención de registros de una Consulta.

```
$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 primer columna del registro, el elemento con índice 1 es la segunda columna, etc. Si no hay más registros por devolver devuelve *false*. Los valores de los campos solicitados en el *result set* son devueltos en forma de *array*.

2.6.7.21. Persistencia.

En el desarrollo de sitios Web y aplicaciones Web, uno de los problemas clásicos es la perdida de persistencia cuando el usuario pasa de una página a otra. Debido a las características de diseño del protocolo HTTP que fuerza una nueva conexión y desconexión por cada request no es posible saber quien está accediendo a que página o en que lugar esta cada usuario del site. Mantener persistencia a lo largo de la navegación del sitio ha sido uno de los temas más complejos e importantes en el desarrollo de aplicaciones Web.

2.6.7.22. Sesiones.

Las sesiones se suelen definir al tiempo en el que un usuario determinado se encuentra navegando en el sitio Web, dependiendo de la definición podemos decir que si el usuario no navega por el site durante una cierta cantidad de minutos ha terminado su sesión en el sitio y a partir de allí cuando vuelve a ingresar lo hace en una nueva sesión.

El concepto de sesión es útil porque es posible asociar a cada sesión un identificador único de forma tal de registrar la actividad del usuario en el site y mantener persistencia utilizando únicamente este identificador, el problema pasa a ser como mantener la persistencia del session-id.

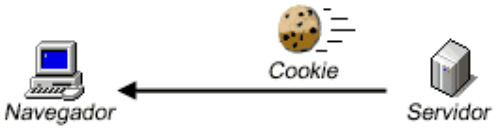
2.6.7.23. Cookies.

Uno de los mecanismos más usados para mantener persistencia es el mecanismo de cookies. El concepto es que mediante un header del proto-

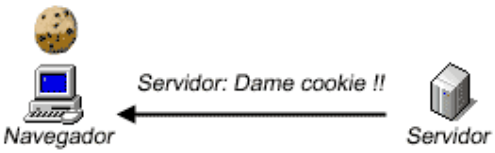
colo HTTP el server pueda almacenar información en el cliente. Cuando el server envía un header con un cookie el browser, si acepta cookies, guarda la información enviada en un archivo de texto con un formato especial. Cada vez que el browser solicita una página del dominio que envió la cookie reenvía la cookie al site, de esta forma es posible mantener persistencia. La información que puede guardarse en una cookie esta limitada por lo que habitualmente se utiliza la misma para mantener el identificador de sesión del usuario almacenándose el resto de los datos necesarios en el servidor usando el session-id de la cookie como clave.

Funcionamiento

La cookie es enviada al navegador desde el servidor y si este la acepta permanece en él.



Las páginas piden la cookie al navegador...



El navegador las envía, permitiendo la identificación del usuario por parte del servidor.

A continuación vamos a ver como usar las cookies para nuestro beneficio.

Para crear un cookie en PHP se utiliza la función setcookie de la siguiente manera:

int = setcookie (nombre, valor, expiración, path, dominio);

nombre. Nombre de la cookie a setear
valor. Valor que contendrá la cookie
expiración. Fecha de vencimiento de la 000kie
path. En general no se usa, suele setearse en "/"
dominio. Dominio para el cual el cookie es válido

La función devuelve verdadero si pudo setearse la cookie o falso en caso contrario, por ejemplo si el browser no acepta cookies.

2.6.7.24. URL.

Otro método posible para pasar información desde una página a otra es mediante el URL de la página en cuestión, usando el URL se pueden pasar datos de una página a otra usando el query string de esta forma:

```
http: //dominio/path?query_string
```

Donde `query_string` es de la forma:

```
variable = valor&variable2=valor2&variable3 = valor3... etc...
```

De esta forma podríamos hacer un manejo similar al anterior pero pasando el session_id usando el URL en lugar de usando cookies, la desventaja de este método es que todos los links deben generarse dinámicamente en PHP para agregar a la dirección del link el valor del cookie de la forma:

```
<a href="http://dominio/path?session=<?print ("$session_id");?>">
```

El funcionamiento es similar, no requiere que el browser tenga habilitados cookies pero altera la forma en que se escriben los Iinks y afea un poco la forma en la cual se muestra la URL de la página actual.

2.6.7.25. Sesiones en PHP.

PHP soporta en forma nativa desde el lenguaje el concepto de sesiones, en PHP se pueden manejar sesiones en forma transparente al usuario, el lenguaje define una constante PHPSESSTD con el identificador de cada sesión y se encarga de propagar el mismo usando cookies o bien el URL de la página en caso de que los cookies estén deshabilitados.

Para usar variables de sesión en PHP es necesario invocar la función **session_start()**; en el comienzo del script, esta función se encarga de recuperar el session_id en la constante $PHPSESSID y si la sesión no existe crea una nueva.

PHP dispone además de funciones para registrar variables dentro de una sesión, esto se hace llamando ala función **session_register();.**

```
Session_start();
$x = "hola";
Session_registerÇx");
```

Notar que **sessionregister** recibe el "nombre" de la variable a registrar sin "$". Una vez registrada una variable la misma estará disponible en todas las páginas que usen **session start** durante la sesión actual, es decir que si desde la página donde hicimos el **session_register** el usuario se traslada a otra página que usa **session_start** entonces la variable sx automáticamente estará definida y con el valor **"hola"** ya que es una variable registrada

dentro de la sesión. Cualquier tipo de variable de PHP puede registrarse con este método, incluyendo objetos (sólo las propiedades del objeto se registran, no los métodos)

2.6.7.26. Manejo de HTTP en PHP.

Uno de los temas más importantes en todo lenguaje de programación usado para generación dinámica de sitios Web y aplicaciones Web es el manejo del protocolo HTTP, conexiones, métodos GET y POST, uploads, headers, cache y demás alternativas. Todas estas funciones están bien soportadas en php de tal forma de tener desde el lenguaje un control completo sobre la forma en que el server interactúa con el browser.

2.6.7.27. Headers.

Una de las funciones más importantes de PHP es la función header(); que sirve para enviar al browser un determinado header http.

Por default en cuanto un script PHP usa una función de salida o transmite algo al browser php envía el header "Content-Type: text/html" al browser. Por eso es importante destacar que la función header solo puede usarse ANTES de realizar cualquier tipo de salida al browser.

```
header ('Location: un")
```

Este es un header http que sirve para redireccionar al browser a otra página, script o URL, es muy útil para scripts php que procesan datos recibidos desde un formulario o similar y luego en función de los datos redireccionan al browser a una página que por ejemplo puede mostrar erro-

res en el ingreso de datos, aceptar los datos o simplemente volver a la página que llamo al script.

Este tipo de redirección sólo puede usarse si no se emitió ninguna salida al browser, si ya se emitió una salida al browser y es necesario redireccionar la página debemos generar desde PHP código JavaScript que transmita al browser y el browser si es capaz de interpretar JavaScript podrá redireccionarse a la página pasada.

3. Capítulo III Desarrollo del Proyecto

3.1. Análisis del Sistema

El análisis del Sistema de Monitoreo de Egresados de la PUCESA se a obtenido identificando las necesidades de la Universidad, tomando en cuenta las necesidades de la misma y la viabilidad del proyecto; de esta manera se ha realizado el análisis técnico con el que se ha obtenido las especificaciones y las restricciones del sistema.

3.1.1. Identificación de las Necesidades

Carencia de un Sistema de Monitoreo de los Egresados que proporcione información estadística de los mismos con las siguientes necesidades:

- Un sistema automatizado de información de los egresados de la Universidad.
- Carencia de un sistema de información estadística de los egresados.
- Inexistencia de información del nivel académico obtenido por los egresados.
- Una base de datos relacional que permita almacenar la mayor cantidad de información con respecto a los egresados de la PUCESA en el ámbito profesional y académico.

3.1.2. Estudio de Viabilidad.

Se ha tomado en cuenta los estudios en las siguientes áreas:

a) **Económica:** Los costos para el desarrollo e implementación del sistema son accesibles tanto en el ámbito del hardware y software.

b) **Técnica:** El servidor SUN e250 cuenta con todos los componentes necesarios para la implementación y codificación del sistema, al igual que el espacio en disco para el almacenamiento de los datos y pagina Web.

c) **Legal:** Se ha determinado que no existe ninguna infracción legal que pueda incurrir en el desarrollo del sistema.

d) **Alternativas:** En caso de que el servidor de la universidad no este disponible por cualquier circunstancia, el sistema se podrá implementar de forma local en cualquier equipo de que cumpla con las necesidades mínimas del sistema (un servidor de base de datos MySQL, compatibilidad con PHP)

3.2. Análisis Técnico

3.2.1. Análisis de Requisitos del Sistema de Monitoreo de los Egresados.

- Es posible el desarrollo del sistema de monitoreo de Egresados para la recopilación de información de los egresados y graduados de forma automática vía Web.

- El *front end*[9]se desarrolló en lenguaje PHP que es un lenguaje multiplataforma encaminado a las aplicaciones Web, brindando todas las herramientas para la conexión con la base de datos.
- Todo el *back end*[10] del sistema será implementado en MySQL, el cual está alojado en el servidor de la universidad. MySQL.es un popular sistema de dirección de base de datos que conjuntamente con el motor de almacenamiento InnoDB se compilará los datos que el sistema requiera.

3.2.2. Análisis de Requerimientos del Servidor SUN e250.

- MySQL 4.1.14
- PHP 4.2.2
- Apache

3.2.3. Análisis técnico

- Módulos de conexión de PHP con MySQL.
- **Riesgos del desarrollo**. Se puede cumplir con todas las funciones y requerimientos que necesite la universidad.
- **Disponibilidad de recursos**. Se dispone de los equipos de cómputo y aplicaciones de desarrollo necesarios para la realización del sistema,
- **Tecnología**. La tecnología actual permite el desarrollo eficiente del sistema.

[9] Lenguaje del sistema
[10] Fondo del sistema

3.2.4. Especificaciones.

Para el desarrollo del sistema se ha tonado en cuenta las siguientes especificaciones:

Especificaciones del Sistema de Monitoreo de Egresados	
Leguaje de programación	PHP
Base de datos	MySQL
Entorno operativo:	sistemas operativos y browser
Funciones	Validación y control de los usuarios • Administrador • Escuelas • Egresados Mantenimiento de las base de datos. • Creación • Modificación y • Eliminación Reportes y datos estadísticos de la información obtenida desde el sistemas
Restricciones	• Este sistema será utilizado únicamente para el control de los egresados de la PUCESA • El sistemas no realizará funciones de respaldo de los procesamientos de datos ni operaciones de backup y recuperación automatizados, todos estos procesos quedan a cargo de los administradores del sitio Web.

3.3. Análisis de Procesos

3.3.1. 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 básicos.

Diagrama de Flujo de Datos. Nivel 0

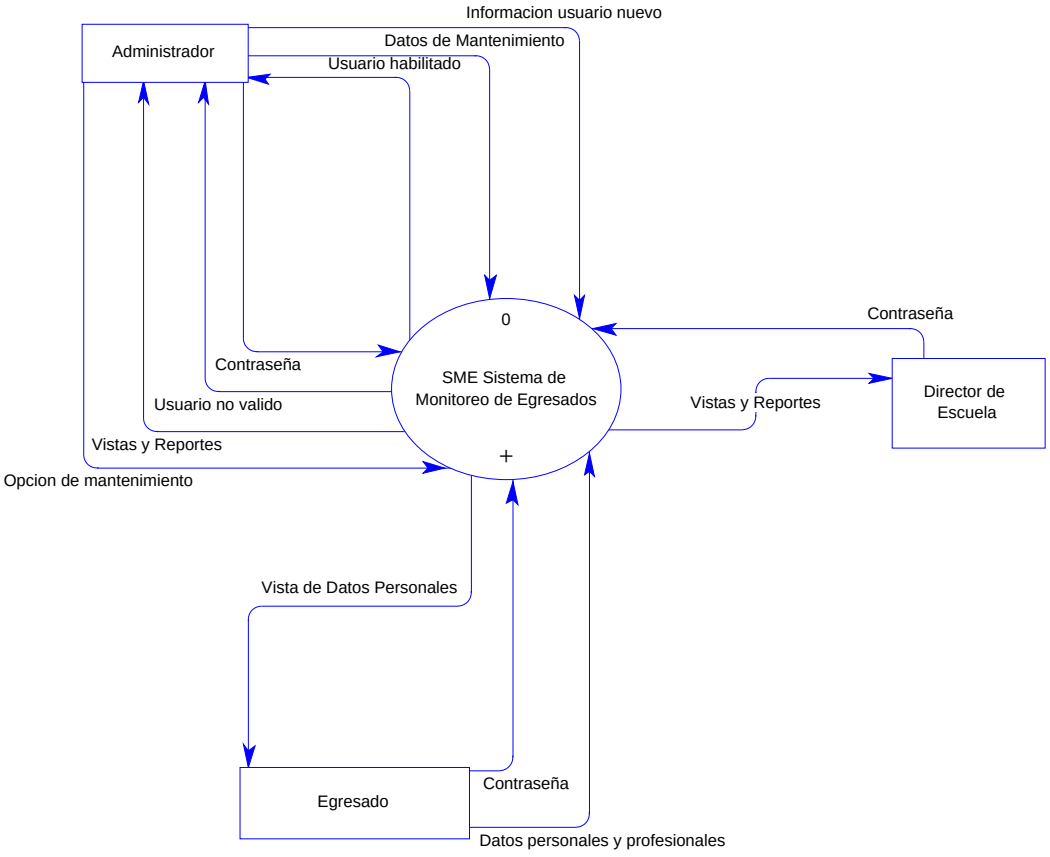


Figura 11. DFD 0 del Sistema de monitoreo de Egresados

3.3.2. Diagrama de Flujo de Datos Nivel 1

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

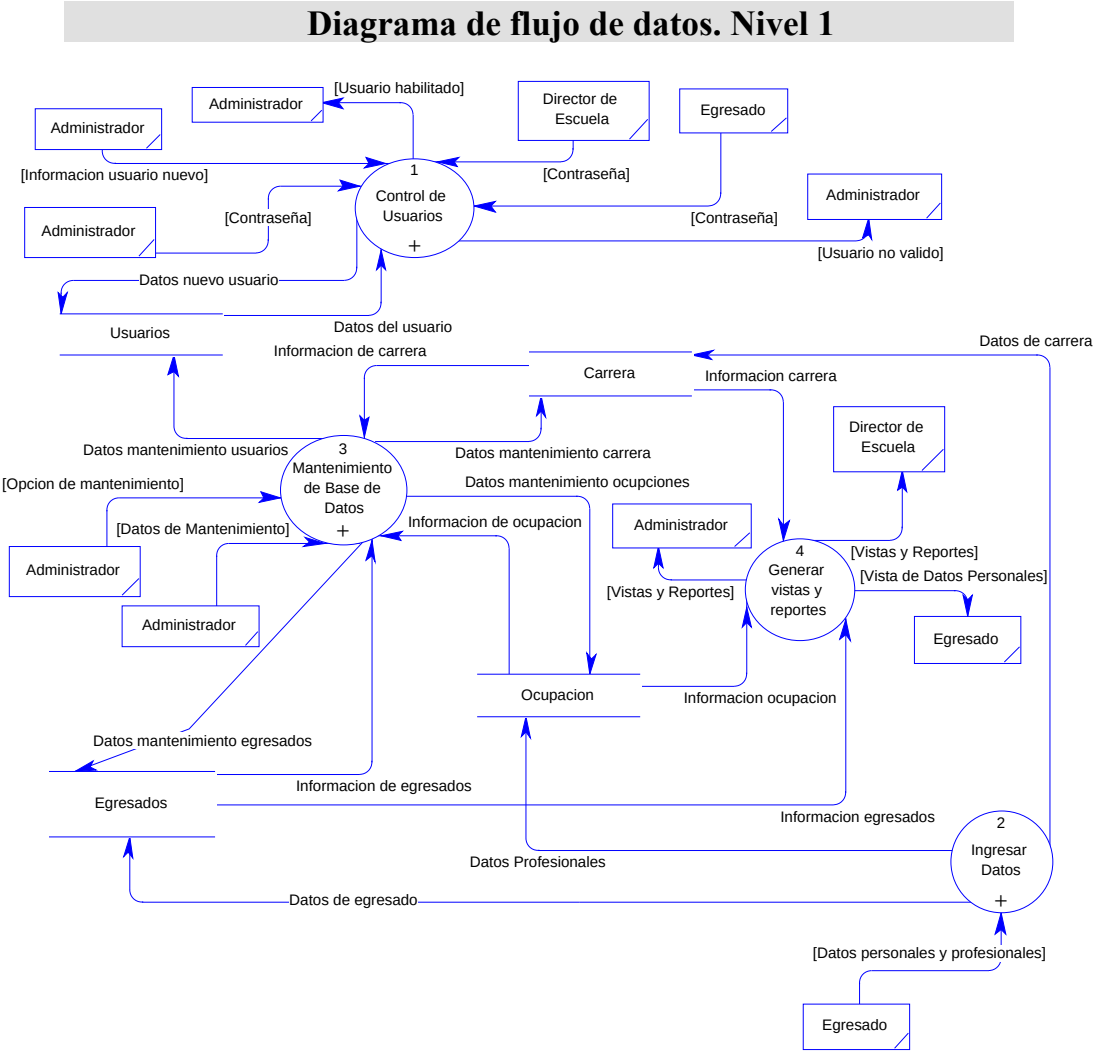


Figura 12. DFD 1 de Sistema de Monitoreo de Egresados

3.3.3. Diagrama de Flujo de Datos

En los Diagramas siguientes de Flujo de Datos se muestra los de Control de Usuarios, Ingreso de Datos y Manteniemiento de la base de datos, donde se encuentra especificado los procesos del SME.

Diagrama de Flujo de Datos de Control de Usuarios

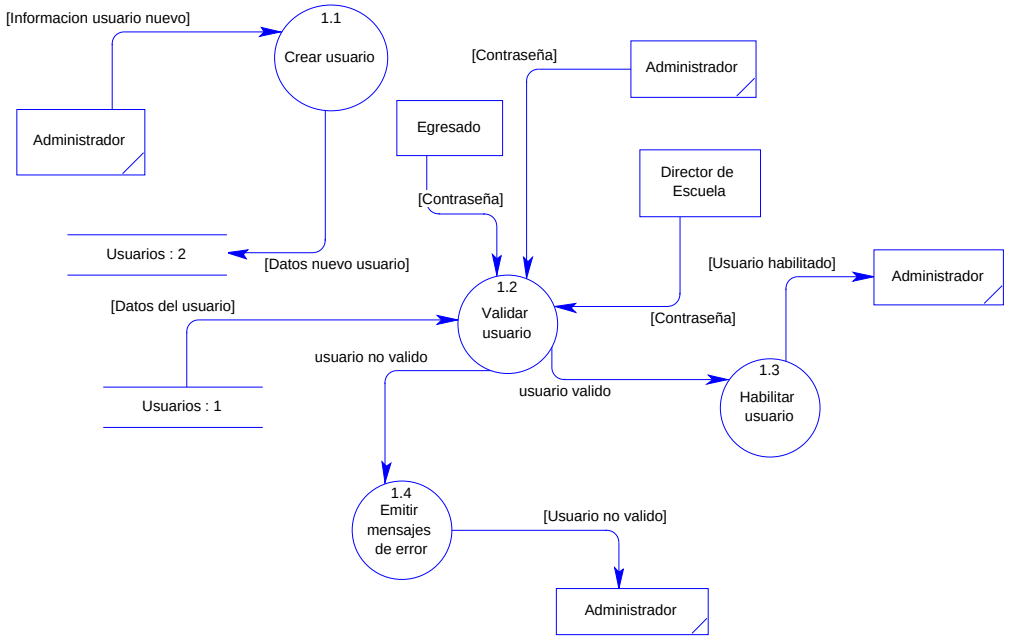


Figura 13. DFD 1.1 de Control de Usuarios

Diagrama de Flujo de Datos de Ingreso de Datos

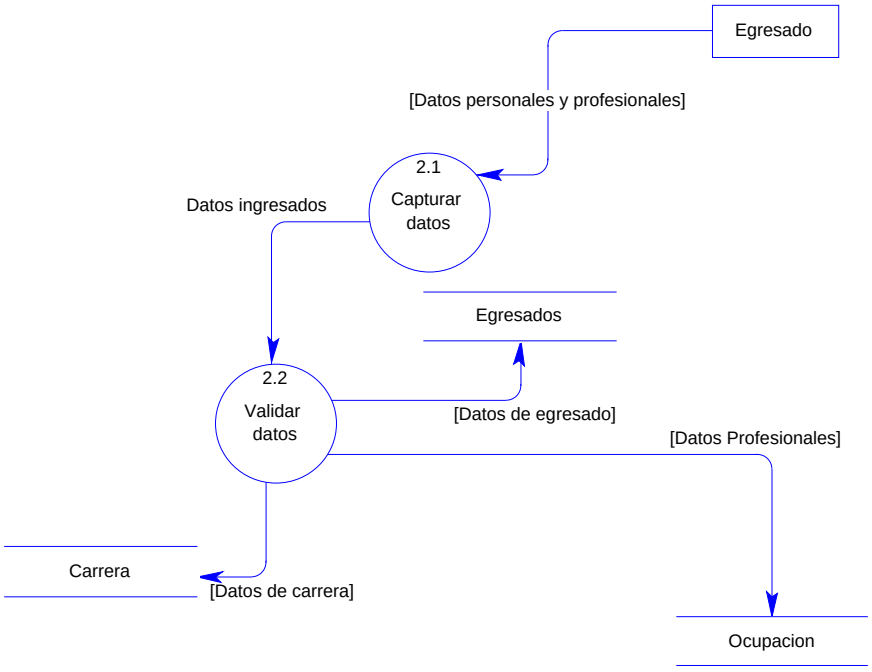


Figura 14. DFD Ingreso de Datos

Mantenimiento de Base de Datos

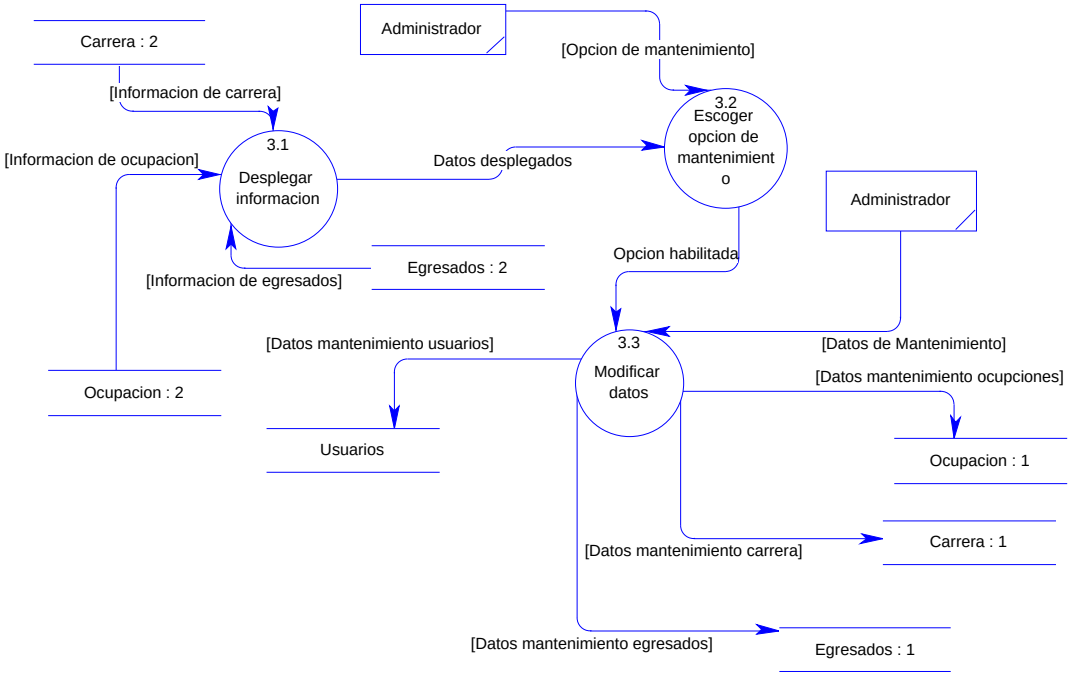


Figura 15 DFD Mantenimiento de Base de Datos

3.4. Diseño del Sistema.

Para el diseño del SME se tomaron en cuenta los requisitos del sistema, necesidades, viabilidad, análisis técnico, especificaciones y análisis estructurado para la realización técnica del software y su respectiva implementación.

La elaboración del sistema requiere de una completa organización jerárquica, al igual que debe contar con un diseño de interfaces sencilla y funcional reduciendo la complejidad en las conexiones, todo esto con el fin de hacer posible la codificación del sistema.

Para el diseño del sitio se ha modelado en forma detallada la base de datos, la interfaz y procedimientos de cálculos para el Sistema de Monitoreo.

3.4.1. Diseño de la Base de Datos.

Para el diseño de la base de datos del sistema SME, se detallan los objetos que la conforman como son las entidades con sus atributos, relaciones, índices, longitud de cada columna y los tipos de datos.

Modelo Físico de la Base de Datos. Se detalla la información básica y estadística de la base de datos que conforma el sistema, como su nombre, número de tablas, columnas, índices y relaciones.

Diagrama Entidad Relación. Explica de manera gráfica a las entidades con sus relaciones y atributos de la base de datos del sistema.

Tablas. Describe a las entidades con sus columnas, claves primarias y secundarias, índices y relaciones.

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

Relaciones. Individualiza las relaciones existentes en la ase de datos, el tipo de relación que posee, la tabla padre e hijo y su cardinalidad.

Índices. Detalla a los índices que identifican las claves foráneas que se encuentran en una determinada identidad.

DIAGRAMA DE LA BASE DE DATOS

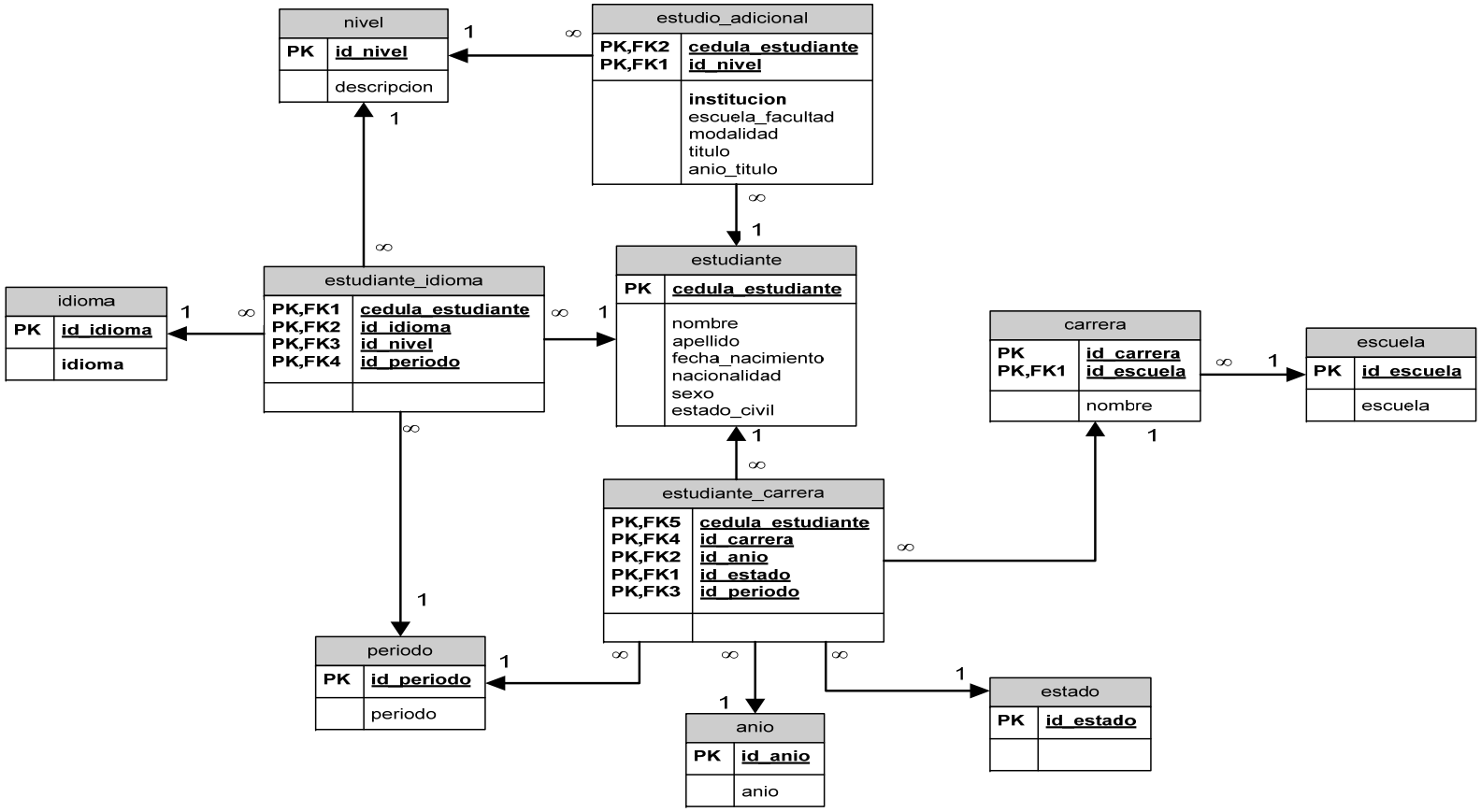


Figura 16. Diagrama de la Base de Datos

INFORMACION DEL MODELO FISICO

Tabla	Registros	Tipo	Tamaño
anio	2	MyISAM	2.0 KB
capacitacion	3	MyISAM	2.1 KB
carrera	2	MyISAM	1.1 KB
escuela	0	MyISAM	1.0 KB
estado	2	MyISAM	2.0 KB
estudiante	3	MyISAM	2.2 KB
estudiante_carrera	3	MyISAM	1.1 KB
estudiante_idioma	3	MyISAM	1.1 KB
estudio_adicional	3	MyISAM	1.1 KB
experiencia	3	MyISAM	1.1 KB
idioma	3	MyISAM	3.1 KB
nivel	2	MyISAM	1.0 KB
ocupacion	2	MyISAM	2.0 KB
Periodo	2	MyISAM	2.0 KB
usuario	5	MyISAM	2.2 KB
15 tabla(s)	**38**	**--**	**25.1 KB**

ANIO

Campo	Tipo	Nulo	Predeterminado
id_anio	varchar(25)	No	
anio	varchar(10)	No	

CAPACITACION

Campo	Tipo	Nulo	Predeterminado
cedula_estudiante	varchar(11)	No	
Institución	varchar(15)	No	
ciudad	varchar(15)	No	
tema	varchar(30)	No	
duracion	decimal(3,0)	No	0
fecha_inicio	date	No	0000-00-00
fecha_fin	date	No	0000-00-00

CARRERA

Campo	Tipo	Nulo	Predeterminado
id_carrera	varchar(25)	No	
nombre	varchar(30)	No	
id_escuela	varchar(25)	No	

ESCUELA

Campo	Tipo	Nulo	Predeterminado
id_escuela	varchar(25)	No	
escuela	varchar(30)	No	

ESTADO

Campo	Tipo	Nulo	Predeterminado
id_estado	varchar(25)	No	
estado	varchar(20)	No	

ESTUDIANTE

Campo	Tipo	Nulo	Predeterminado
cedula_estudiante	varchar(11)	No	
nombre	varchar(30)	No	
apellido	varchar(30)	No	
fecha_nacimiento	date	No	0000-00-00
nacionalidad	varchar(25)	No	
sexo	varchar(20)	No	
estado_civil	varchar(20)	No	

ESTUDIANTE_CARRERA

Campo	Tipo	Nulo	Predeterminado
cedula_estudiante	varchar(11)	No	
id_carrera	varchar(25)	No	
id_periodo	varchar(25)	No	
id_anio	varchar(25)	No	
id_estado	varchar(25)	No	

ESTUDIANTE_IDIOMA

Campo	Tipo	Nulo	Predeterminado
cedula_estudiante	varchar(11)	No	
id_idioma	varchar(25)	No	
nivel	varchar(10)	No	

ESTUDIO_ADICIONAL

Campo	Tipo	Nulo	Predeterminado
cedula_estudiante	varchar(11)	No	
id_nivel	varchar(25)	No	
Institución	varchar(25)	No	
escuela_facultad	varchar(25)	No	
modalidad	varchar(25)	No	
titulo	varchar(30)	No	
anio_titulo	varchar(4)	No	

EXPERIENCIA

Campo	Tipo	Nulo	Predeterminado
cedula_estudiante	varchar(11)	No	
id_ocupacion	varchar(25)	No	
fecha_inicio	date	No	0000-00-00
fecha_fin	date	No	0000-00-00
lugar	varchar(25)	No	
descripcion	varchar(30)	No	

IDIOMA

Campo	Tipo	Nulo	Predeterminado
id_idioma	varchar(25)	No	
idioma	varchar(20)	No	

NIVEL

Campo	Tipo	Nulo	Predeterminado
id_nivel	varchar(25)	No	
descripcion	varchar(30)	No	

OCUPACION

Campo	Tipo	Nulo	Predeterminado
id_ocupacion	varchar(25)	No	
descripcion	varchar(30)	No	

PERIODO

Campo	Tipo	Nulo	Predeterminado
id_periodo	varchar(25)	No	
periodo	varchar(20)	No	

USUARIO

Campo	Tipo	Nulo	Predeterminado
cedula	varchar(11)	No	
password	varchar(6)	No	
nickname	varchar(6)	No	
Nombre	varchar(20)	No	
apellido	varchar(20)	No	
nivel	Int(11)	No	0

3.4.2. Diseño del Sitio Web

El diseño del sitio Web del SME se detalla de manera jerárquica, con los usuarios y las paginas que tienen acceso los mismos.

3.4.2.1.Arquitectura del Sitio

3.4.2.1.1. Diseño de la interfaz

Para realizar el diseño de la interfaz del sistema se tienen en cuenta las posibles conexiones que tendrá ésta, tanto las internas entre sus módulos o funciones, como las externas con los usuarios que manejan la información.

3.4.2.1.2. Interfaz interna

La interfaz interna hace referencia al flujo de información que fluye entre los módulos del sistema. En los diagramas de flujo de datos obtenidos en la fase de análisis de procesos estaba representado este flujo como objetos de datos que entraban o salían de los procesos de dichos diagramas.

En nuestro caso la interfaz interna consistirá en especificar para cada función del sistema los datos que acepta como entrada, los datos que produce como salida, las tablas de la base de datos que utiliza o actualiza y las posibles excepciones o mensajes de cualquier tipo que pueden producirse durante la ejecución de la función.

Todos estos datos que constituyen la interfaz interna del sistema se recogen en la especificación detallada de las funciones que se realiza en la Figura 15.

3.4.2.1.3. Interfaz externa

La interfaz externa hace referencia a las conexiones del sistema con los diferentes productores o consumidores de información externos al programa y que no son humanos.

En nuestro caso, al fijarse en el diagrama de contexto obtenido en el análisis (DFD0), se verá que la única entidad que requiere una interfaz externa es la pantalla del ordenador.

Todos los datos que entran o salen del sistema lo harán a través de la interfaz mostrada en la pantalla. Para poder introducir y visualizar datos sera necesario habilitar una casilla de texto para cada campo de las tablas de la base de datos, cada casilla irá acompañada de una etiqueta identificando dicho campo. Además será necesario habilitar una lista para mostrar los resultados de consultas consistentes en listados.

Las órdenes se le darán al sistema mediante pulsación en los diferentes links.

3.4.2.1.4. Interfaz de usuario

A los usuarios del sistema se les supone el conocimiento informático mínimo para saber rellenar campos con los datos adecuados y para moverse entre páginas Web.

Jerarquia de Usuarios: Tenemos tres tipos de usuarios los mismos que detallamos a continuación

- ***Administrador***: Tiene acceso total en el sistema

- ***Visualidador***: Tiene acceso solo a los reportes y busquedas
- ***Egresado***: Tiene acceso solo a la Ficha de sus datos personales y profesionales

3.4.2.1.5. Prototipos de las páginas Web

Con los requisitos de las interfaces interna, externa y de usuario comentados previamente se puede bosquejar el aspecto que tendrían las páginas Web del sistema.

Se muestran a continuación unos prototipos en los que sólo se pretende mostrar los elementos que deben contener, no el aspecto final que tendrán.

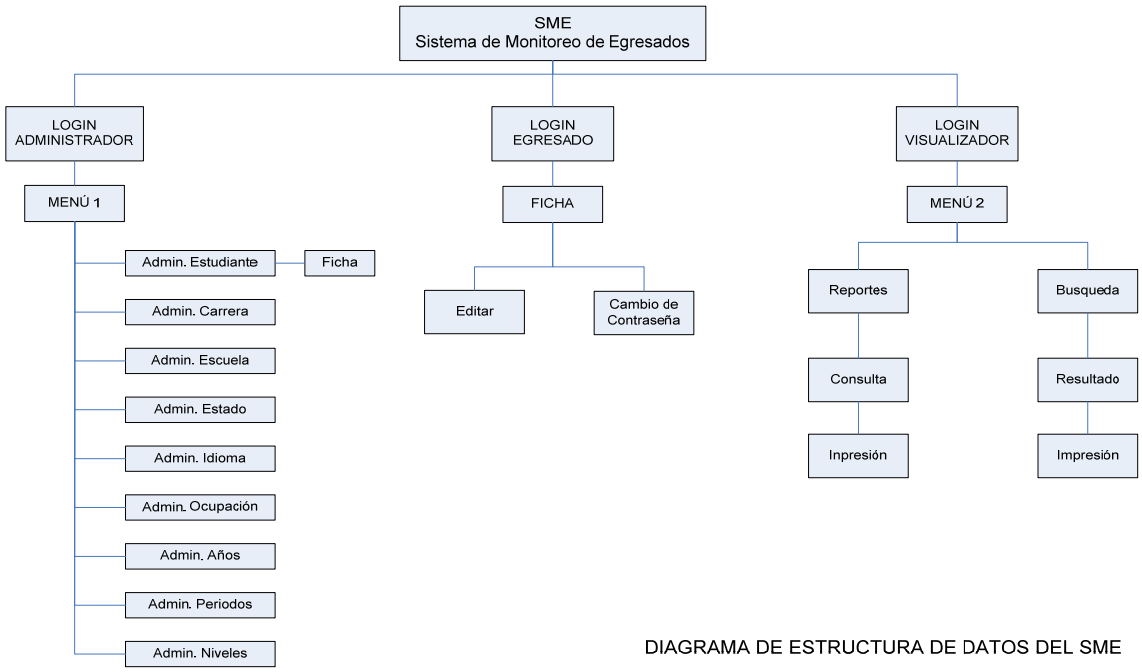


Figura 17. Diagrama de estructura del SME

3.4.3. Desarrollo del Sitio Web

En el desarrollo del SME comprende la creación, codificación e interacción de los módulos que comprenden para el funcionamiento del sistema, de igual manera el diseño y desarrollo del Font End para generar las entidades y atributos para el desarrollo del Back End con ello codificar las diferentes funciones para alcanzar el óptimo funcionamiento del sistema.

Se utilizó Macromedia Dreamweaver 8, para el diseño y la codificación del sitio tanto en HTML como en PHP, utilizando el motor de base de datos MySQL. Se empezó diseñando la página principal de ingreso de usuarios, haciéndola de fácil uso y agradable a la vista del usuario.

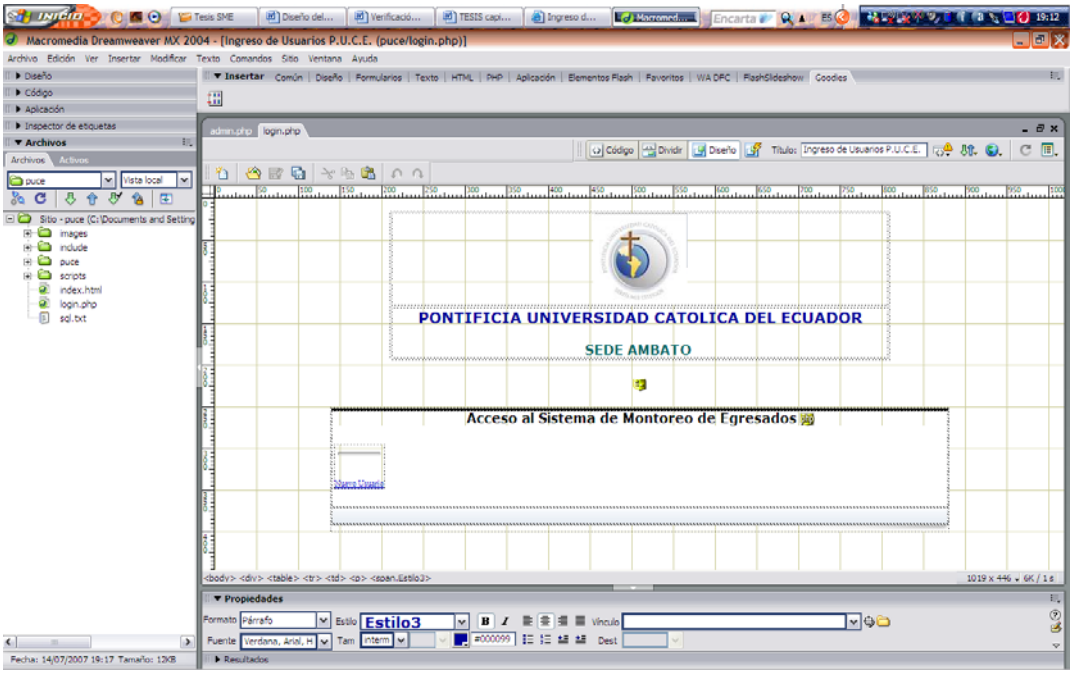


Figura 18. Diseño Pagina Inicio

Se utilizó una función en PHP para la conexión a la base de datos, la cual es llamada desde cada una de las páginas que necesitan el acceso a los datos, dicha función presenta el siguiente código:

```
<?php
function Conectarse()
{
  if (!($link=mysql_connect("localhost","root","")))
  {
    echo "Error conectando a la base de datos";
    exit();
  }
  if (!mysql_select_db("puce",$link))
  {
    echo "Error seleccionando la base de datos";
    exit();
  }
  return $link;
}

$link=Conectarse();

mysql_close($link); //cierra la conexion
?>
```

Para el ingreso a la página tenemos 3 tipos de usuarios: administrador, visualizador y usuario.

En el administrador tiene acceso al menú con las opciones de búsqueda, eliminación y creación de usuarios al igual que la información de los estudiantes.

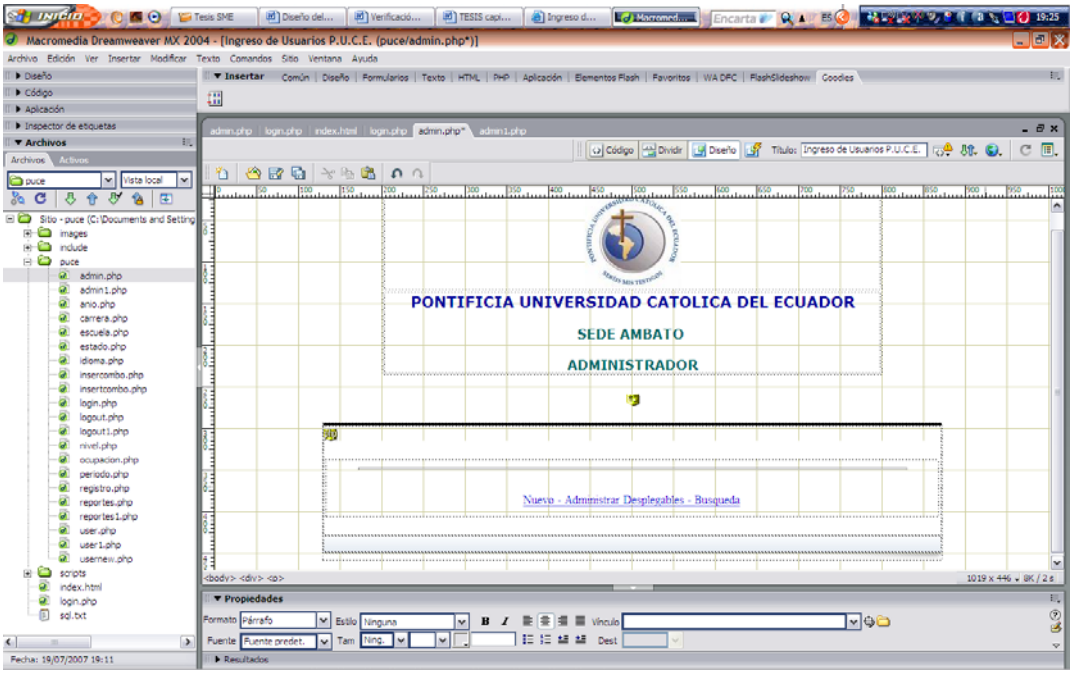


Figura 19. Diseño página Administrador

El script para la creación de un nuevo usuario se presenta a continuación:

```
function check_estudiante( form, is_new )
{
//          client = form.slt_branch.options[form.slt_branch.selectedIndex].value;
usuario = '';
if ( is_new == 1 )
          usuario = form.usuario.value;
password = form.password.value;
password1 = form.password1.value;

cedula = form.cedula.value;
nombre = form.nombre.value;
apellido = form.apellido.value;
fecha_nacimiento = form.fecha_nacimiento.value;
nacionalidad = form.nacionalidad.value;
```

```
head = 'Tiene que llenar todos los campos\n\n';
msg = '';
if(usuario=='' && is_new == 1)
	msg += "Debe introducir un Usuario\n";
if(password=='' && is_new == 1)
	msg += "Debe introducir un Password\n";
else if(password!='' && password!=password1 && is_new == 0)
	msg += "Los Passwords no coinciden\n";
if(cedula=='')
	msg += "Debe introducir un Cédula\n";
if(nombre=='')
	msg += "Debe introducir un Nombre\n";
if(apellido=='')
	msg += "Debe introducir un Apellido\n";
if(fecha_nacimiento=='')
	msg += "Debe introducir una Fecha de nacimiento\n";
if(nacionalidad=='')
	msg += "Debe introducir una Nacionalidad\n";


if (msg=='')
{
	return confirm( 'Confrime el registro de los siguientes datos\n\n' + 'Cédula: '
+ cedula + '\nNombre: ' + nombre + '\nApellido: ' + apellido + '\nFecha de Nacimien-
to: ' + fecha_nacimiento + '\nNacionalidad: ' + nacionalidad );
}
else
{
	alert(head + msg);
	return false;
}

}
```

Por otra parte para la eliminación de los datos del los usuarios se presenta a continuación en la siguiente función:

```
function delete()
	{
		$sql_carrera = "delete from estudiante_carrera where cedu-
la_estudiante='" . $this->cedula . "'";
		$sql_usuario = "delete from experiencia where cedula='" . $this-
>cedula . "'";
		$sql_experiencia = "delete from experiencia where cedu-
la_estudiante='" . $this->cedula . "'";
		$sql_capacitacion = "delete from capacitacion where cedu-
la_estudiante='" . $this->cedula . "'";
		$sql_estudios = "delete from estudio_adicional where cedu-
la_estudiante='" . $this->cedula . "'";
		$sql_idioma = "delete from estudiante_idioma where cedu-
la_estudiante='" . $this->cedula . "'";
```

```
                        $sql = "delete from estudiante where cedula_estudiante='" . $this-
        >cedula . "'";
                        $this->mysql->query( $sql );
                        $this->mysql->query( $sql_experiencia );
                        $this->mysql->query( $sql_capacitacion );
                        $this->mysql->query( $sql_estudios );
                        $this->mysql->query( $sql_idioma );
                        $this->mysql->query( $sql_usuario );
                        $this->mysql->query( $sql_carrera );
                }
```

Para guardar los datos del los usuarios se presenta a continuación en la siguiente función:

```
        function save()
                {
//                      $new_id = $this->mysql->get_new_id( "estudiantes" );

                        if ( $this->is_new )
                        {
                                $sql = "insert into estudiante values( '" . $_POST["cedula"] .
"', '" . $_POST["nombre"] . "', '" . $_POST["apellido"] . "', '" . $_POST["fecha_nacimiento"] . "',
'" . $_POST["nacionalidad"] . "', '" . $_POST["sexo"] . "', '" . $_POST["estado_civil"] . "' )";
                                $sql_experiencia = "insert into experiencia values( '" .
$_POST["cedula"] . "', '" . $_POST["id_ocupacion"] . "', '" . $_POST["exp_fecha_inicio"] . "', '"
. $_POST["exp_fecha_fin"] . "', '" . $_POST["lugar"] . "', '" . $_POST["descripcion"] . "' )";
                                $sql_capacitacion = "insert into capacitacion values( '" .
$_POST["cedula"] . "', '" . $_POST["institucion"] . "', '" . $_POST["ciudad"] . "', '" .
$_POST["tema"] . "', '" . $_POST["duracion"] . "', '" . $_POST["fecha_inicio"] . "', '" .
$_POST["fecha_fin"] . "' )";
                                $sql_estudios = "insert into estudio_adicional values( '" .
$_POST["cedula"] . "', '" . $_POST["id_nivel"] . "', '" . $_POST["est_institucion"] . "', '" .
$_POST["escuela_facultad"] . "', '" . $_POST["modalidad"] . "', '" . $_POST["titulo"] . "', '" .
$_POST["anio_titulo"] . "' )";
                                $sql_idioma = "insert into estudiante_idioma values( '" .
$_POST["cedula"] . "', '" . $_POST["id_idioma"] . "', '" . $_POST["idioma_nivel"] . "' )";
                                $sql_carrera = "insert into estudiante_carrera values( '" .
$_POST["cedula"] . "', '" . $_POST["id_carrera"] . "', '" . $_POST["id_periodo"] . "', '" .
$_POST["id_anio"] . "', '" . $_POST["id_estado"] . "' ) ";
                                $sql_usuario = "insert into usuario values( '" .
$_POST["cedula"] . "', '" . $_POST["password"] . "', '" . $_POST["usuario"] . "', '" .
$_POST["nombre"] . "', '" . $_POST["apellido"] . "', '2' )";
                        }
                        else
                        {
                                $sql = "Update estudiante SET nombre='" .
$_POST["nombre"] . "', apellido='" .$_POST["apellido"] . "', fecha_nacimiento='"
.$_POST["fecha_nacimiento"] . "', nacionalidad='" .$_POST["nacionalidad"] . "', sexo='"
.$_POST["sexo"] . "', estado_civil='" .$_POST["estado_civil"] . "' where cedula_estudiante='"
. $this->cedula . "'";
                                $sql_experiencia = "Update experiencia SET
id_ocupacion='" .  $_POST["id_ocupacion"] . "', fecha_inicio='" .
$_POST["exp_fecha_inicio"] . "', fecha_fin='" .  $_POST["exp_fecha_fin"] . "', lugar='" .
$_POST["lugar"] . "', descripcion='" .  $_POST["descripcion"] . "' where cedula_estudiante='"
. $_POST["cedula"] . "'";
```

```
                $sql_capacitacion = "Update capacitacion SET institucion='"
.  $_POST["institucion"] . "', ciudad='" .  $_POST["ciudad"] . "', tema='" .  $_POST["tema"] . "',
duracion='" .  $_POST["duracion"] . "', fecha_inicio='" .  $_POST["fecha_inicio"] . "', fe-
cha_fin='" .  $_POST["fecha_fin"] . "' where cedula_estudiante='" . $_POST["cedula"] . "'";
                $sql_estudios = "Update estudio_adicional SET id_nivel='" .
$_POST["id_nivel"] . "', institucion='" .  $_POST["est_institucion"] . "', escuela_facultad='" .
$_POST["escuela_facultad"] . "', modalidad='" .  $_POST["modalidad"] . "', titulo='" .
$_POST["titulo"] . "', anio_titulo='" .  $_POST["anio_titulo"] . "' where cedula_estudiante='" .
$_POST["cedula"] . "'";
                $sql_idioma = "Update estudiante_idioma SET id_idioma='" .
$_POST["id_idioma"] . "', nivel='" .  $_POST["idioma_nivel"] . "' where cedula_estudiante='" .
$_POST["cedula"] . "'";
                $sql_carrera = "Update estudiante_carrera SET
id_carrera='" .  $_POST["id_carrera"] . "', id_periodo='" .  $_POST["id_periodo"] . "',
id_anio='" .  $_POST["id_anio"] . "', id_estado='" .  $_POST["id_estado"] . "' where cedu-
la_estudiante='" . $_POST["cedula"] . "'";
                $password = ( $_REQUEST["password"] )? ", password='" .
$_REQUEST["password"] . "'": "";
                $sql_usuario = "Update usuario SET nombre='" .
$_REQUEST["nombre"] . "', apellido='" . $_REQUEST["apellido"] . "' " . $password . " where
cedula='" . $_REQUEST["cedula"] . "'";
            }

            $this->mysql->query( $sql );
            $this->mysql->query( $sql_experiencia );
            $this->mysql->query( $sql_capacitacion );
            $this->mysql->query( $sql_estudios );
            $this->mysql->query( $sql_idioma );
            $this->mysql->query( $sql_carrera );
            $this->mysql->query( $sql_usuario );
        }
```

Para visualizar los datos del los egresados, el reporte general se presenta a continuación en la siguiente función:

```
function get_list()
        {
            $sql = "select * from estudiante Order By nombre";
            $rst = $this->mysql->query( $sql );

            $retval = '
            <table border="0" align="center">
                <tr>
                    <td align="left"><a
href="?mode=frm">Nuevo</a></td>
                </tr>
                <tr>
                    <td class="table_title"
align="center">Nombre</td>
                    <td class="table_title"
align="center">Apellido</td>
```

```
                                        <td class="table_title"
align="center">Sexo</td>
                                        <td class="table_title" align="center">Estado
Civil</td>
                                        <td class="table_title"
align="center">Nacionalidad</td>
                                        <td class="table_title" colspan="3"
align="center">Acci&oacute;n</td>
                                </tr>';
                        if ( mysql_num_rows( $rst ) > 0 )
                        {
                                while( $row = mysql_fetch_object( $rst ) )
                                {
                                        $class = ( $first )? "table_high_row": "ta-
ble_low_row";
                                        $retval .= '
                                        <tr>
                                                <td class="' . $class . '">' . $row-
>nombre . '</td>
                                                <td class="' . $class . '">' . $row-
>apellido . '</td>
                                                <td class="' . $class . '">' . $row->sexo .
'</td>
                                                <td class="' . $class . '">' . $row-
>estado_civil . '</td>
                                                <td class="' . $class . '">' . $row-
>nacionalidad . '</td>
                                                <td class="' . $class . '"><a
href="?mode=details&cedula=' . $row->cedula_estudiante . '" ti-
tle="editar">Detalles</a></td>
                                                <td class="' . $class . '"><a
href="?mode=frm&cedula=' . $row->cedula_estudiante . '" title="editar"><img
src="http://' . $_SERVER['HTTP_HOST'] . '/puce/images/admin/button_edit.png"
border=0 alt="editar"></a></td>
                                                <td class="' . $class . '"><a
href="?mode=del&cedula=' . $row->cedula_estudiante . '" title="eliminar" on-
Click=" return confirm(\'Confirme eliminación de:\\n\\n '  . $row->apellido . ' ' .
$row->nombre . '\')"><img src="http://' . $_SERVER['HTTP_HOST'] .
'/puce/images/admin/button_drop.png" border=0 alt="eliminar"></a></td>
                                        </tr>';
                                        $first = !$first;
                                }
                        }
                        else
                        {
                                $retval .= '
                                        <tr>
                                                <td colspan="8" class="table_low_row"
align="center">No existen estudiantes registrados!!!</td>
```

```
                                        </tr>';
                        }
                        $retval .= '
                        </table>';
                        return $retval;
                }
```

Para visualizar el formulario de búsqueda para los reportes se presenta a continuación en la siguiente función:

```
function get_report_form()
                {
                        $retval = '
                        <form name="frm_report" action="" method="post">
                                <table align="center">
                                        <tr>
                                                <td colspan="2" align="center"
class="table_title">Reportes</td>
                                        </tr>
                                        <tr>
                                                <td
class="table_title">Per&iacute;odo:</td>
                                                <td>
                                                        <select name="periodo">';
                        $sql_periodo = "select * from periodo";
                        $rst_periodo = $this->mysql->query( $sql_periodo );
                        if ( mysql_num_rows( $rst_periodo ) > 0 )
                        {
                                while( $row_periodo = mysql_fetch_object(
$rst_periodo ) )
                                {
                                        $selected = ( $row_periodo->id_periodo ==
$_REQUEST["periodo"] )? " selected": "";
                                        $retval .= '
                                                                <option value="' .
$row_periodo->id_periodo . '" ' . $selected . '>' . $row_periodo->periodo;
                                }
                        }
                        $retval .= '
                                                        </select>
                                                </td>
                                        </tr>
                                        <tr>
                                                <td class="table_title">Estado:</td>
                                                <td>
                                                        <select name="estado">';
                        $sql_estado = "select * from estado";
```

```
                        $rst_estado = $this->mysql->query( $sql_estado );
                        if ( mysql_num_rows( $rst_estado ) > 0 )
                        {
                                while( $row_estado = mysql_fetch_object( $rst_estado
) )
                                {
                                        $selected = ( $row_estado->id_estado ==
$_REQUEST["estado"] )? " selected": "";
                                        $retval .= '
                                                                <option value="' .
$row_estado->id_estado . '" ' . $selected . '>' . $row_estado->estado;
                                }
                        }
                        $retval .= '
                                                        </select>
                                                </td>
                                        </tr>
                                        <tr>
                                                <td class="table_title">Carrera:</td>
                                                <td>
                                                        <select name="carrera">
                                                                <option value="">Todas';
                        $sql_carrera = "select * from carrera";
                        $rst_carrera = $this->mysql->query( $sql_carrera );
                        if ( mysql_num_rows( $rst_carrera ) > 0 )
                        {
                                while( $row_carrera = mysql_fetch_object(
$rst_carrera ) )
                                {
                                        $selected = ( $row_carrera->id_carrera ==
$_REQUEST["carrera"] )? " selected": "";
                                        $retval .= '
                                                                <option value="' .
$row_carrera->id_carrera . '" ' . $selected . '>' . $row_carrera->nombre;
                                }
                        }
                        $retval .= '
                                                        </select>
                                                </td>
                                        </tr>
                                        <tr>
                                                <td colspan="2" align="center"><input
name="report" type="submit" value="Reporte"></td>
                                        </tr>
                                </table>
                        </form>';
                        return $retval;
                }
```

Para visualizar el reporte se presenta a continuación en la siguiente función:

```
function get_result_report()
{
	$carrera = ( $_REQUEST["carrera"] )? "and carre-
ra.id_carrera='" . $_REQUEST["carrera"] . "'": "";
//	$sql = "select COUNT( DIS-
TINCT(estudiante_carrera.cedula_estudiante) ) from estudiante_carrera inner join
periodo on estudiante_carrera.id_periodo=periodo.id_periodo inner join estado on
estudiante_carrera.id_estado=estado.id_estado inner join carrera on estudian-
te_carrera.id_carrera=carrera.id_carrera where periodo.periodo='" .
$_REQUEST["periodo"] . "' and estado.estado='" . $_REQUEST["estado"] . "' " .
$carrera;
	$sql_total = "select COUNT( DIS-
TINCT(estudiante_carrera.cedula_estudiante) ) as total from estudiante_carrera inner
join periodo on estudiante_carrera.id_periodo=periodo.id_periodo inner join estado
on estudiante_carrera.id_estado=estado.id_estado inner join carrera on estudian-
te_carrera.id_carrera=carrera.id_carrera where periodo.id_periodo='" .
$_REQUEST["periodo"] . "' " . $carrera;
	$rst_total = $this->mysql->query( $sql_total );
	$row_total = mysql_fetch_object( $rst_total );

	$sql_estado = "select * from estado where id_estado='" .
$_REQUEST["estado"] . "'";
	$rst_estado = $this->mysql->query( $sql_estado );
	$row_estado = mysql_fetch_object( $rst_estado );

	$sql = "select COUNT( DIS-
TINCT(estudiante_carrera.cedula_estudiante) ) as num from estudiante_carrera inner
join periodo on estudiante_carrera.id_periodo=periodo.id_periodo inner join estado
on estudiante_carrera.id_estado=estado.id_estado inner join carrera on estudian-
te_carrera.id_carrera=carrera.id_carrera where periodo.id_periodo='" .
$_REQUEST["periodo"] . "' and estado.id_estado='" . $_REQUEST["estado"] . "' " .
$carrera;
	$rst = $this->mysql->query( $sql );
	$row = mysql_fetch_object( $rst );


	if ( !$_REQUEST["carrera"] )
	{
		$sql_carrera = "select * from carrera";
		$rst_carrera = $this->mysql->query( $sql_carrera );
		$retval = '
		<table align="center" border=1>
			<tr>
				<td class="table_title">Carrera</td>
```

```
                                        <td class="table_title">Total de Alumnos</td>
                                        <td class="table_title">Alumnos ' . $row_estado->estado . '</td>
                                        <td class="table_title">Porcentaje ' . $row_estado->estado . '(%)</td>
                                    </tr>';
                                while( $row_carrera = mysql_fetch_object( $rst_carrera ) )
                                {
                                        $sql_por_carrera = "select COUNT( DISTINCT(estudiante_carrera.cedula_estudiante) ) as num from estudiante_carrera inner join periodo on estudiante_carrera.id_periodo=periodo.id_periodo inner join estado on estudiante_carrera.id_estado=estado.id_estado inner join carrera on estudiante_carrera.id_carrera=carrera.id_carrera where periodo.id_periodo='" . $_REQUEST["periodo"] . "' and estado.id_estado='" . $_REQUEST["estado"] . "' and carrera.id_carrera='" . $row_carrera->id_carrera . "'";
                                        $rst_por_carrera = $this->mysql->query( $sql_por_carrera );
                                        $row_por_carrera = mysql_fetch_object( $rst_por_carrera );

                                        $sql_total_por_carrera = "select COUNT( DISTINCT(estudiante_carrera.cedula_estudiante) ) as total from estudiante_carrera inner join periodo on estudiante_carrera.id_periodo=periodo.id_periodo inner join estado on estudiante_carrera.id_estado=estado.id_estado inner join carrera on estudiante_carrera.id_carrera=carrera.id_carrera where periodo.id_periodo='" . $_REQUEST["periodo"] . "' and carrera.id_carrera='" . $row_carrera->id_carrera . "'";
                                        $rst_total_por_carrera = $this->mysql->query( $sql_total_por_carrera );
                                        $row_total_por_carrera = mysql_fetch_object( $rst_total_por_carrera );

                                        $retval .= '
                                        <tr>
                                                <td align="left">' . $row_carrera->nombre . '</td>
                                                <td align="center">' . $row_total_por_carrera->total . '</td>
                                                <td align="center">' . $row_por_carrera->num . '</td>';
                                        $row_total_por_carrera->total = ( $row_total_por_carrera->total == 0 )? 1: $row_total_por_carrera->total;
                                        $retval .= '
                                                <td align="right">' . number_format( ( ( $row_por_carrera->num * 100 ) / $row_total->total ), 2, ".", "," ) . ' %</td>
                                        </tr>';
                                }
```

```
                        $retval .= '
                                <tr>
                                        <td>Todas</td>
                                        <td align="center">' . $row_total->total .
'</td>
                                        <td align="center">' . $row->num .
'</td>
                                        <td align="right">' . number_format( ( (
$row->num * 100 ) / $row_total->total ), 2, ".", "," ) . ' %</td>
                                </tr>
                        </table>';
                }
                else
                {
                        $sql_carrera = "select * from carrera where
id_carrera='" . $_REQUEST["carrera"] . "'";
                        $rst_carrera = $this->mysql->query( $sql_carrera );
                        $row_carrera = mysql_fetch_object( $rst_carrera );
                        $retval = '
                        <table align="center" border=1>
                                <tr>
                                        <td class="table_title">Carrera</td>
                                        <td class="table_title">Total de Alum-
nos</td>
                                        <td class="table_title">Alumnos ' .
$row_estado->estado . '</td>
                                        <td class="table_title">Porcentaje ' .
$row_estado->estado . '(%)</td>
                                </tr>
                                <tr>
                                        <td>' . $row_carrera->nombre . '</td>
                                        <td align="center">' . $row_total->total .
'</td>
                                        <td align="center">' . $row->num .
'</td>
                                        <td align="right">' . number_format( ( (
$row->num * 100 ) / $row_total->total ), 2, ".", "," ) . ' %</td>
                                </tr>
                        </table>';

                }

                return $retval;
        }
```

3.4.4. Creación de la Base de Datos.

Se ha creado la base de datos con una estructura SQL con las entidades y atributos, así como los valores iniciales que tendrá el back end del sistema.

CREACION DE LA BASE DE DATOS

```
-- phpMyAdmin SQL Dump
-- version 2.7.0-pl1
-- http://www.phpmyadmin.net
--
-- Servidor: localhost
-- Tiempo de generación: 02-07-2007 a las 15:00:48
-- Versión del servidor: 4.1.14
-- Versión de PHP: 5.0.5
--
-- Base de datos: `puce`
--
CREATE DATABASE `puce` DEFAULT CHARACTER SET utf8 COLLATE
utf8_general_ci;
USE puce;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `anio`
--

DROP TABLE IF EXISTS `anio`;
CREATE TABLE IF NOT EXISTS `anio` (
  `id_anio` varchar(25) NOT NULL default '',
  `anio` varchar(10) NOT NULL default '',
  PRIMARY KEY  (`id_anio`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `anio`
--

INSERT INTO `anio` VALUES ('1', '2006');
INSERT INTO `anio` VALUES ('2', '2007');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `capacitacion`
--

DROP TABLE IF EXISTS `capacitacion`;
CREATE TABLE IF NOT EXISTS `capacitacion` (
  `cedula_estudiante` varchar(11) NOT NULL default '',
  `institucion` varchar(15) NOT NULL default '',
  `ciudad` varchar(15) NOT NULL default '',
  `tema` varchar(30) NOT NULL default '',
  `duracion` decimal(3,0) NOT NULL default '0',
  `fecha_inicio` date NOT NULL default '0000-00-00',
```

```
  `fecha_fin` date NOT NULL default '0000-00-00',
  KEY `cedula_estudiante_2` (`cedula_estudiante`),
  FULLTEXT KEY `cedula_estudiante` (`cedula_estudiante`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `capacitacion`
--

INSERT INTO `capacitacion` VALUES ('1', 'ESPOCH', 'Riobamba', 'Ju-
lio', 55, '2007-02-06', '2007-06-30');
INSERT INTO `capacitacion` VALUES ('2', 'ESPOCH', 'Riobamba', 'XXX',
2, '2007-06-27', '2007-06-28');
INSERT INTO `capacitacion` VALUES ('3', 'ESPOCH', 'Riobamba', 'XXX',
22, '2007-06-29', '2007-06-30');
-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `carrera`
--

DROP TABLE IF EXISTS `carrera`;
CREATE TABLE IF NOT EXISTS `carrera` (
  `id_carrera` varchar(25) NOT NULL default '',
  `nombre` varchar(30) NOT NULL default '',
  `id_escuela` varchar(25) NOT NULL default ''
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `carrera`
--

INSERT INTO `carrera` VALUES ('1', 'Sistemas', 'FIE');
INSERT INTO `carrera` VALUES ('2', 'Administración de Empresas',
'FADE');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `escuela`
--

DROP TABLE IF EXISTS `escuela`;
CREATE TABLE IF NOT EXISTS `escuela` (
  `id_escuela` varchar(25) NOT NULL default '',
  `escuela` varchar(30) NOT NULL default '',
  PRIMARY KEY  (`id_escuela`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `escuela`
--


-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `estado`
--
```

```
DROP TABLE IF EXISTS `estado`;
CREATE TABLE IF NOT EXISTS `estado` (
  `id_estado` varchar(25) NOT NULL default '',
  `estado` varchar(20) NOT NULL default '',
  PRIMARY KEY  (`id_estado`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `estado`
--

INSERT INTO `estado` VALUES ('1', 'Egresado');
INSERT INTO `estado` VALUES ('2', 'Graduado');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `estudiante`
--

DROP TABLE IF EXISTS `estudiante`;
CREATE TABLE IF NOT EXISTS `estudiante` (
  `cedula_estudiante` varchar(11) NOT NULL default '',
  `nombre` varchar(30) NOT NULL default '',
  `apellido` varchar(30) NOT NULL default '',
  `fecha_nacimiento` date NOT NULL default '0000-00-00',
  `nacionalidad` varchar(25) NOT NULL default '',
  `sexo` varchar(20) NOT NULL default '',
  `estado_civil` varchar(20) NOT NULL default '',
  PRIMARY KEY  (`cedula_estudiante`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `estudiante`
--

INSERT INTO `estudiante` VALUES ('1', 'Julio', 'Cabezas Samaniego',
'1978-07-28', 'ecuatoriana', 'Masculino', 'Soltero(a)');
INSERT INTO `estudiante` VALUES ('2', 'Isabel', 'León', '1980-04-
07', 'ecuatoriana', 'Femenino', 'Soltero(a)');
INSERT INTO `estudiante` VALUES ('3', 'Narciza', 'Riera', '1982-06-
01', 'ecuatoriana', 'Femenino', 'Soltero(a)');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `estudiante_carrera`
--

DROP TABLE IF EXISTS `estudiante_carrera`;
CREATE TABLE IF NOT EXISTS `estudiante_carrera` (
  `cedula_estudiante` varchar(11) NOT NULL default '',
  `id_carrera` varchar(25) NOT NULL default '',
  `id_periodo` varchar(25) NOT NULL default '',
  `id_anio` varchar(25) NOT NULL default '',
  `id_estado` varchar(25) NOT NULL default ''
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `estudiante_carrera`
```

```
--

INSERT INTO `estudiante_carrera` VALUES ('1', '1', '1', '2', '1');
INSERT INTO `estudiante_carrera` VALUES ('2', '2', '1', '1', '1');
INSERT INTO `estudiante_carrera` VALUES ('3', '2', '1', '1', '2');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `estudiante_idioma`
--

DROP TABLE IF EXISTS `estudiante_idioma`;
CREATE TABLE IF NOT EXISTS `estudiante_idioma` (
  `cedula_estudiante` varchar(11) NOT NULL default '',
  `id_idioma` varchar(25) NOT NULL default '',
  `nivel` varchar(10) NOT NULL default ''
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `estudiante_idioma`
--

INSERT INTO `estudiante_idioma` VALUES ('1', '2', 'ultimo');
INSERT INTO `estudiante_idioma` VALUES ('2', '1', 'No se');
INSERT INTO `estudiante_idioma` VALUES ('3', '1', 'noveno');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `estudio_adicional`
--

DROP TABLE IF EXISTS `estudio_adicional`;
CREATE TABLE IF NOT EXISTS `estudio_adicional` (
  `cedula_estudiante` varchar(11) NOT NULL default '',
  `id_nivel` varchar(25) NOT NULL default '',
  `institucion` varchar(25) NOT NULL default '',
  `escuela_facultad` varchar(25) NOT NULL default '',
  `modalidad` varchar(25) NOT NULL default '',
  `titulo` varchar(30) NOT NULL default '',
  `anio_titulo` varchar(4) NOT NULL default ''
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `estudio_adicional`
--

INSERT INTO `estudio_adicional` VALUES ('1', '2', 'La vida', 'ni
se', 'xxx', 'Ing', '2007');
INSERT INTO `estudio_adicional` VALUES ('2', '2', 'ESPOCH', 'XXXX',
'XXX', 'XXXX', '2006');
INSERT INTO `estudio_adicional` VALUES ('3', '', '', '', '', '',
'');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `experiencia`
--
```

```
DROP TABLE IF EXISTS `experiencia`;
CREATE TABLE IF NOT EXISTS `experiencia` (
  `cedula_estudiante` varchar(11) NOT NULL default '',
  `id_ocupacion` varchar(25) NOT NULL default '',
  `fecha_inicio` date NOT NULL default '0000-00-00',
  `fecha_fin` date NOT NULL default '0000-00-00',
  `lugar` varchar(25) NOT NULL default '',
  `descripcion` varchar(30) NOT NULL default ''
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `experiencia`
--

INSERT INTO `experiencia` VALUES ('1', '1', '2000-02-01', '2005-04-
25', 'Riobamba', 'lo nuevo');
INSERT INTO `experiencia` VALUES ('2', '1', '2007-06-05', '2007-06-
30', 'Riobamba', 'esta es la descripcioh de mi e');
INSERT INTO `experiencia` VALUES ('3', '', '0000-00-00', '0000-00-
00', '', '');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `idioma`
--

DROP TABLE IF EXISTS `idioma`;
CREATE TABLE IF NOT EXISTS `idioma` (
  `id_idioma` varchar(25) NOT NULL default '',
  `idioma` varchar(20) NOT NULL default '',
  PRIMARY KEY  (`id_idioma`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `idioma`
--

INSERT INTO `idioma` VALUES ('1', 'Español');
INSERT INTO `idioma` VALUES ('2', 'Inglés');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `nivel`
--

DROP TABLE IF EXISTS `nivel`;
CREATE TABLE IF NOT EXISTS `nivel` (
  `id_nivel` varchar(25) NOT NULL default '',
  `descripcion` varchar(30) NOT NULL default ''
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `nivel`
--

INSERT INTO `nivel` VALUES ('1', 'Primero');
INSERT INTO `nivel` VALUES ('2', 'Segundo');
```

```
-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `ocupacion`
--

DROP TABLE IF EXISTS `ocupacion`;
CREATE TABLE IF NOT EXISTS `ocupacion` (
  `id_ocupacion` varchar(25) NOT NULL default '',
  `descripcion` varchar(30) NOT NULL default '',
  PRIMARY KEY  (`id_ocupacion`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `ocupacion`
--

INSERT INTO `ocupacion` VALUES ('1', 'Profesor');
INSERT INTO `ocupacion` VALUES ('2', 'Estudiante');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `periodo`
--

DROP TABLE IF EXISTS `periodo`;
CREATE TABLE IF NOT EXISTS `periodo` (
  `id_periodo` varchar(25) NOT NULL default '',
  `periodo` varchar(20) NOT NULL default '',
  PRIMARY KEY  (`id_periodo`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `periodo`
--

INSERT INTO `periodo` VALUES ('1', '2006-2007');
INSERT INTO `periodo` VALUES ('2', '2007-2008');

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `usuario`
--
DROP TABLE IF EXISTS `usuario`;
CREATE TABLE IF NOT EXISTS `usuario` (
  `cedula` varchar(11) NOT NULL default '',
  `password` varchar(6) NOT NULL default '',
  `nickname` varchar(6) NOT NULL default '',
  `nombre` varchar(20) NOT NULL default '',
  `apellido` varchar(20) NOT NULL default '',
  `nivel` int(11) NOT NULL default '0',
  PRIMARY KEY  (`cedula`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--
-- Volcar la base de datos para la tabla `usuario`
--
```

```
INSERT INTO `usuario` VALUES ('1', 'cambio', 'jubaco', 'Julio', 'Ca-
bezas Samaniego', 3);
INSERT INTO `usuario` VALUES ('1000', 'admin', 'admin', 'Administra-
dor', 'Administrador', 1);
INSERT INTO `usuario` VALUES ('1001', 'visual', 'visual', 'Vizuali-
zador', 'Vizualizador', 2);
INSERT INTO `usuario` VALUES ('2', 'xx', 'xx', 'Isabel', 'León', 2);
INSERT INTO `usuario` VALUES ('3', 'yyy', 'YYY', 'Narciza', 'Riera',
2);
```

3.5. Pruebas e Instalación

3.5.1. Pruebas

Se realizaron todas las pruebas necesarias de forma local antes de la puesta en marcha del sistema por la PUCESA

3.5.1.1. Pruebas de caja negra

Las pruebas de caja negra de la aplicación consistirán en analizar el comportamiento de las diferentes funciones del sistema con datos de entrada correctos e incorrectos y comprobar que responde de manera adecuada, es decir, con los resultados de la operación cuando los datos son correctos y con mensajes de error adecuados cuando son incorrectos.

Como la aplicación funciona básicamente con la pulsación en los link, será a estas pulsaciones a las que se les apliquen las pruebas.

A continuación se muestran las pruebas de caja negra realizadas junto con los resultados obtenidos.

PRUEBAS DE ACCESO AL SISTEMA	
PRUEBA REALIZADA	**RESULTADO OBTENIDO**
Pulsar el icono de ingreso a "SME"	Se muestra la pantalla de contraseña
Pantalla de Contraseña – NickName mal	Mensaje de error de usuario
Pantalla de Contraseña NickName - bien Contraseña - Mal	Mensaje de error de Contraseña
Pantalla de Contraseña NickName – bien Contraseña – Bien	Acceso a la pagina de: ➢ Administrador ➢ Visualizador ➢ Egresado

PRUEBAS DE ACCESO A LA PAGINA DEL ADMINISTRADOR	
PRUEBA REALIZADA	**PRUEBA REALIZADA**
Pulsar "Admin. Estudiante"	Se muestra una busqueda ya sea por Apellido o por Cédula, se puede acceder a Nuevo o vista preliminar del listado,
Busqueda de egresado sin nombre en busqueda y pulsar el boton Buscar	Se muestra el listado general de los egresados, en donde podemos ver los Detalles del usuario, Editar o Eliminar
Pulsar "Menu Principal"	Permite regresar a la pagina principal del Administrador, de cualquiera de las paginas del Administrador
Pulsar "Admin. Carreras"	Accede a la página de Administración de

	Carreras, donde se puede crear nuevo, eliminar o editar
Pulsar Nuevo en admin. Carreras	Accede al formulario de Carreras
Pulsar "Admin. Escuelas"	ccede a la página de Administración de Escuelas, donde se puede crear nuevo, eliminar o editar
Pulsar Nuevo en admin. Escuelas	Accede al formulario de escuelas
Pulsar "Admin. Estado"	ccede a la página de Administración de Estado, donde se puede crear nuevo, eliminar o editar
Pulsar Nuevo en admin. Estado	Accede al formulario de estado
Pulsar "Admin. Idioma"	ccede a la página de Administración de Idioma, donde se puede crear nuevo, eliminar o editar
Pulsar Nuevo en admin. Idioma	Accede al formulario de idioma
Pulsar "Admin. Ocupación"	ccede a la página de Administración de Ocupación, donde se puede crear nuevo, eliminar o editar
Pulsar Nuevo en admin. Ocpación	Accede al formulario de ocupación
Pulsar "Admin. Años"	ccede a la página de Administración de Años, donde se puede crear nuevo, eliminar o editar
Pulsar Nuevo en admin. Años	Accede al formulario de años
Pulsar "Admin. Periodos"	ccede a la página de Administración de

	Periodos, donde se puede crear nuevo, eliminar o editar
Pulsar Nuevo en admin. Periodos	Accede al formulario de periodos
Pulsar "Admin. Nivel"	ccede a la página de Administración de Nivel, donde se puede crear nuevo, eliminar o editar
Pulsar Nuevo en admin. Nivel	Accede al formulario de nivel

PRUEBAS DE ACCESO A LA PAGINA DEL VISUALIZADOR	
PRUEBA REALIZADA	**PRUEBA REALIZADA**
Ingresar Nick y contraseña de Visualizador	Ingresa al Menu Principal de la pagina de Visualizador
Pulsar "Reportes"	Ingresa a la pagina de parámetros para desplegar el reporte
Escoger los parámetros y Pulsar el botón "Reporte"	Nos despliega el Listado de todos los usuarios que tengan las caracteristicas especificadas
Pulsar "Vista Previa" en el Reporte	Visualiza el Reporte como queda antes de imprimir
Pulsar "Imprimir" en la Vista Previa del Reporte	Nos despliega la ventana para escoger impresora e imprimir el informe deseado
Pulsar "Ir a Menu Principal"	Nos premite regresar a la pagina principal del Visualizador
Pulsar "Buscar Estudiantes"	Ingresa a la pagina de parámetros para

	desplegar el reporte
Escoger los parámetros y Pulsar el botón "Reporte"	Nos despliega el Listado de todos los nombres de los usuarios que tengan las caracteristicas especificadas.
Pulsar "Detalle"	Nos permite acceder a los datos Generales del Egresado o Graduado
Pulsar "Vista Previa" en el Busq. Estudiante	Visualiza el Reporte como queda antes de imprimir
Pulsar "Imprimir" en la Vista Previa del la Busq. estudiante	Nos despliega la ventana para escoger impresora e imprimir el informe deseado

PRUEBAS DE ACCESO A LA PAGINA DEL EGRESADO	
PRUEBA REALIZADA	**PRUEBA REALIZADA**
Ingresar Nick y contraseña del Egresado	Ingresa al Reporte de la Ficha de Datos de los egresados
Pulsar "Editar" en la Ficha de los Egresados	Ingresa a la Ficha de Datos del egresado en forma de Edicion, que nos permite cambiar algun o todos los datos
Pulsar el botón "Guardar"	Despliega un mensaje y permite guardar todos los cambios realizados de la Ficha del Usuario
Pulsar "Regresar"	Permite ir al reporte de la Ficha de Datos
Pulsar "Salir"	Permite salir a la pagina de inicio del sistema

3.5.2. Instalación

Contenido del disco de instalación:

El del disco de instalación del sistema SME contiene:

- Un directorio con los archivos del sitio web llamado **"PUCE"**
- Un archivo de extensión .dump donde se encuentra la base de datos llamada "puce.dump"
- Programa de instalación appserve 2.5.5 (para SO Windows y Solaris)

Pasos para la Instalación del Sistema.

- Como primer paso, procedemos a la instalación del **appserve 2.5.5**, el mismo que nos instala la versión de PHP 4.1.7 y el motor de base de datos MySQL 4.1.14
- Grabamos el sitio Web que contiene los archivos del sistema en el directorio respectivo:
    - Windows. "D:\web\puce"
- Para cargar las base de datos del sistema es necesario copiar el archivo **puce.dump** en el directorio del **appserve**

4. CAPITULO IV. Verificación y Validación de Resultados

4.1. Comprobación de la Hipótesis

Con la implementación del "Sistema de Monitoreo de los Egresados de La Pontificia Universidad Católica del Ecuador Sede Ambato" y creada la información estadística de los egresados y graduados de la PUCESA, podrán tener actualizado sus datos, tanto personales como profesionales mediante el uso de esta página Web.

Se demuestra la hipótesis mediante el método lógico Modus Ponendo Ponens.

A = Variable Independiente
B = Variable Dependiente
A = Implementación del SME
B = Obtención de datos de los egresados

$$\begin{array}{l} \mathbf{A \rightarrow B} \\ \mathbf{A} \\ \hline \mathbf{B} \end{array}$$

Esto quiere decir que:

Al implementar un Sistema de Monitoreo Egresados de la Pontificia Universidad Católica del Ecuador Sede Ambato, se está contribuyendo con el control de egresados para el mejoramiento de la calidad académica de la Universidad.

4.2. Validación.

4.3. Conclusiones

- La utilización de esta pagina nos permite tener un informe detallado acerca de la vida profesional tanto de egresados como de graduados de la PUCESA.
- Los datos obtenidos a través del sistema serán precisos ya que cada ex alumno ingresa su propia información personal y profesional.
- Las autoridades de la PUCESA pueden llevar un monitoreo de sus ex alumnos y utilizar esta información para la realización de análisis y estudios con el fin de realizar proyectos que beneficien a la institución.
- A través de los reportes datos del sitio web se podrá valorar que porcentaje de alumnos escogieron de manera correcta su carrera y si se encuentra laborando en la misma.
- Buscando favorecer la calidad de la enseñanza y disminuir la desocupación y deserción de los estudiantes en las distintas carreras y así también ofrecer más y mejores carreras.
- Se realiza el diseño utilizando las técnicas y modelos seleccionados para btener las funciones necesarias para satisfacer los requisitos del sistema.
- Se estudian las distintas tecnologías existentes para el desarrollo Web y e eligen las mas convenientes para el presente trabajo que son: Apache como ervidor Web, MySQL como gestor de bases de datos y PHP como lenguaje de implementación.

4.4. Recomendaciones

- Para obtener buenos resultados con el monitoreo de egresados, es aconsejable tener informados a los alumnos de visitar la pagina web de la Universidad, para que de esta manera se informen de las opciones que tiene la misma.
- Para los usuarios administradores y por escuelas es importante revisar el manual de usuario para poder dar mantenimiento de forma adecuada a la base de datos.
- En el proceso de montaje de la base de datos, es de suma importancia mantener el nombre de la base de datos exactamente igual al que indica el proceso de instalación.
- Es importante mantener activo el Sistema de Monitoreo de egresados de la PUCESA con el fin de que los estudiantes tengan acceso las 24 horas y de esta manera recolectar la mayor cantidad de datos posibles.
- Para un mejor resultado del sistema, es recomendable informar o hacer un requisito indispensable que los estudiantes actualizen sus datos en la paguina.

5. BIBLIOGRAFÍA

Libros:

- KROENKE, David M. "Procesamiento de Bases de Datos". Editorial Prentince Hall , Quinta Edición, 1996.
- CHAKRABARTI, Soumen (2003): "Mining the Web". San Francisco, CA.
- ANSIÓN, Juan (1998) Educación: la mejor herencia. Lima: Fondo Editorial de la Pontificia Universidad Católica del Perú.
- Ángel (Comp.) "El examen: textos para su historia y debate" México: UNAM.
- DE KETELE J-H y ROEGIERS, X. (1995) "*Metodología para la recogida de la Información"*. Madrid, La Muralla.
- PHP COOKBOOK. Sklar, David; Trachtenberg, Adam (O'Reilly Vlg. GmbH & Co.)
- PHP 5
  Cabezas Granado, Luis Miguel (Ed. Anaya Multimedia)
- GUÍA AVANZADA PROGRAMACIÓN DE PHP PARA WINDOWS Andrew Stopford (Pearson Educación)
- PHP, Christopher Cosentino (Pearson Educación)
- CREACIÓN DE APLICACIONES WEB CON PHP 4, Maribel Martínez Moyano; Till Gerken; Tobias Ratschiller (Pearson Educación)
- PHP BLACK BOOK (El libro negro de PHP), Peter Moulding
- GUÍA DE APRENDIZAJE: MYSQL Larry Ullman (Pearson Educación)
- WEB DATABASE APPLICATIONS WITH PHP AND MYSQL. Lane, David; Williams, Hugh E. (O'Reilly Vlg. GmbH & Co.)
- MYSQL 5. Gutiérrez, Juan Diego (Ed. Anaya Multimedia)

Páginas Web:

http://www.cristalab.com
http://www.quasimondo.com
http://www.codearchive.com
http://www.php.net
http://dev.mysql.com
http://www.mysql_hispano.org
http://www.webestilo.com/php
http://www.webestilo.com/php
http://www.webestilo.com/mysql/
http://www.webestilo.com/php/articulo.phtml?art=48
http://www.webestilo.com/php/articulo.phtml?art=27
http://dev.mysql.com
http://www.webestilo.com/php
http://www.wikipedia.org
http://zonaphp.com/
http://www.programacionweb.net/

Anexos

A. Manual de usuario

A.1. Introducción

En este manual se dará una detallada explicación del funcionamiento y del correcto uso del "Sistema de Monitoreo de egresados de la PUCESA".

Este sitio Web ha sido diseñado con la finalidad de facilitar el manejo de información del estado de los egresados de la PUCESA, así los estudiantes egresados y graduados puedan actualizar sus datos personales y profesionales desde la pagina Web de la universidad de esta manera la Escuela de Sistemas tendrá un sistema de información mas detallado de lo que están realizando sus ex – alumnos, así como también, la capacitación de los mismos.

Así mismo podemos tener los datos personales, tanto de los egresados como de los graduados de la PUCESA, con el propósito de tener los datos actualizados de los mismos que lo harán por medio de la página Web de la PUCESA con una clave personal.

Con esta página se facilitara la consulta y búsqueda de los datos de los egresados de la PUCESA, manteniéndose el contacto de los ex alumnos de la Universidad y la información profesional de los mismos.

A.2. Instalación

Para una instalación correcta de este sistema se necesita de los siguientes requerimientos de Hardware y Software indicados a continuación, los que incluimos en el paquete del software.

A.3. Pasos para la Instalación del Sistema.

- Como primer paso, procedemos a la instalación del **appserve 2.5.5**, el mismo que nos instala la versión de PHP 4.1.7 y el motor de base de datos MySQL 4.1.14
- Grabamos el sitio Web que contiene los archivos del sistema en el directorio respectivo:
- Windows. "D:\web\puce"
- Para cargar las base de datos del sistema es necesario copiar el archivo **puce.dump** en el directorio del **appserve**

A.4. Ingreso al Sistema

El sistema nos permite el ingreso a 3 niveles:

Se accede al sistema abriendo la páginaWeb "www.pucesa.edu.ec". En esta página se un Clic y se ingresara el nombre de usuario según queramos acceder como Administrador, Visualisador o Usuario. A continuación será necesario introducir un nombre de usuario y una contraseña válida. Si los datos introducidos son erróneos se mostrará un mensaje de error, si son correctos se accederá a la página correspondiente al tipo de usuario en cuestión.

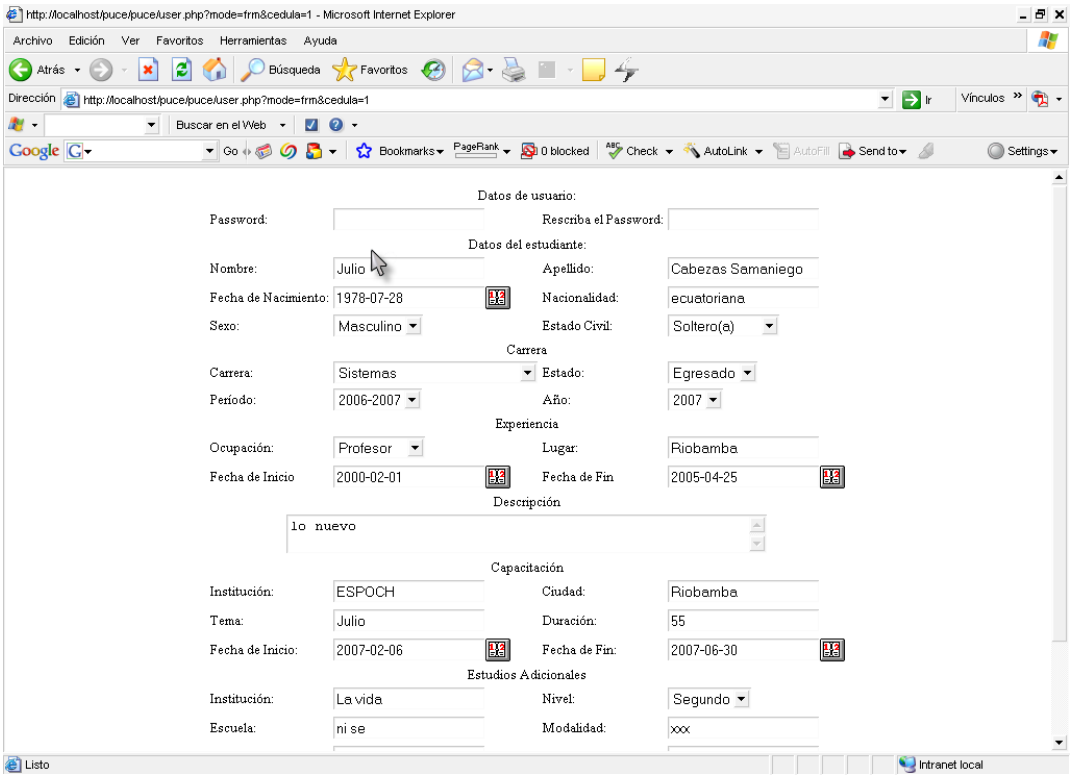
A.5. Ingreso como Estudiante.

En la pantalla de inicio se ingresa con el nombre de usuario y su clave que será su cedula de identidad, tanto para el usuario y la contraseña, la cual la podrá modificar mas adelante.

Luego del ingreso del estudiante, el mismo podrá ver sus datos existentes y modificarlos según las actualizaciones del estudiante.

Datos del estudiante:			
Cédula:	1		
Nombre:	Julio	Apellido:	Cabezas Samaniego
Fecha de Nacimiento:	1978-07-28	Nacionalidad:	ecuatoriana
Sexo:	Masculino	Estado Civil:	Soltero(a)
Experiencia			
Ocupación:	Profesor	Lugar:	Riobamba
Fecha de Inicio	2000-02-01	Fecha de Fin	2005-04-25
Descripción			
lo nuevo			
Capacitación			
Institución:	ESPOCH	Ciudad:	Riobamba
Tema:	Julio	Duración:	55
Fecha de Inicio:	2007-02-06	Fecha de Fin:	2007-06-30
Estudios Adicionales			
Institución:	La vida	Nivel:	Segundo
Escuela:	ni se	Modalidad:	xxx
Titulo:	Ing	Año del Titulo:	2007
Idioma			
Idioma:	Ingles	Nivel:	ultimo
Carrera			
Carrera:	Sistemas	Estado:	Egresado
Período:	2006-2007	Año:	2007
Editar			

Dando clic en el link editar el estudiante podrá modificar sus datos personales, password y datos profesionales del mismo.



A.6. Ingreso como Administrador

En la pantalla de inicio se ingresa como administrador como nombre de usuario y la clave que se le proporcione al administrador.

En la pantalla de administrador, se podrá encontrar un menú con las opciones de búsqueda, eliminación y creación de usuarios al igual que la información de los estudiantes.

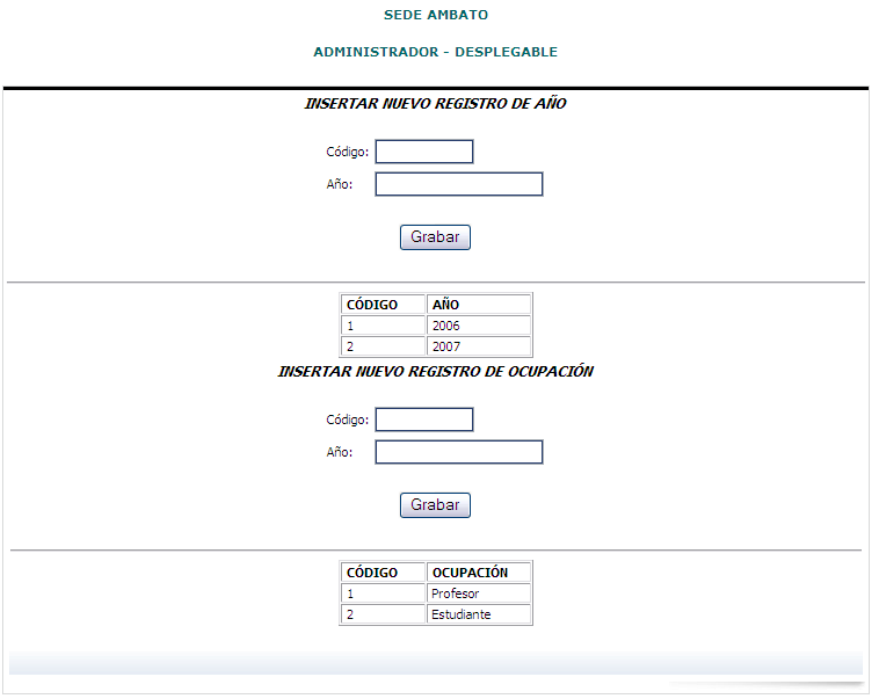
En el menú NUEVO del administrador, podemos crear un nuevo usuario, así como también podemos ver la lista de los estudiantes ingresados al sistema, los mismos que podemos editar , eliminar , o visualizar los detalles del cada usuario

El administrador podrá modificar los datos de los estudiantes dando un clic en el link de detalles y luego en editar. Así mismo podrá eliminarlos con un clic en el botón eliminar.

En el menú ADMINISTRADOR DE DESPLEGABLES, se puede ingresar todos los datos que el usuario debe ingresar por defecto como son:

- Periodo
- Año
- Carrera

- Ocupación
- Idioma
- Nivel



En el menú de Búsqueda podemos realizar búsquedas por:

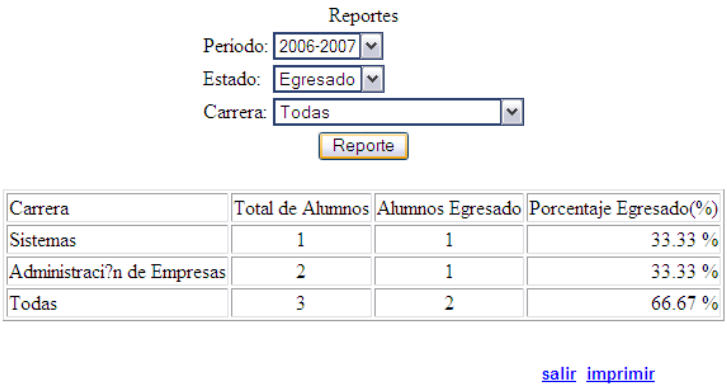
- Carrera
- Apellido
- Periodo
- Estado
- Cedula

A.7. Ingreso como Visualizador

En la pantalla de inicio se ingresa como visualizador el mismo que tendrá acceso solo a los reportes y búsqueda de egresados por el menú de BUSQUEDA.

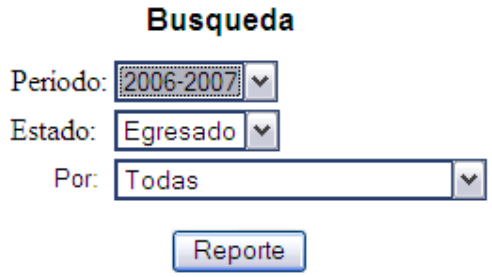
En la pantalla de reportes se podrá visualizar por los siguientes campos:

- Periodo
- Estado
- Carrera
- Estado laboral



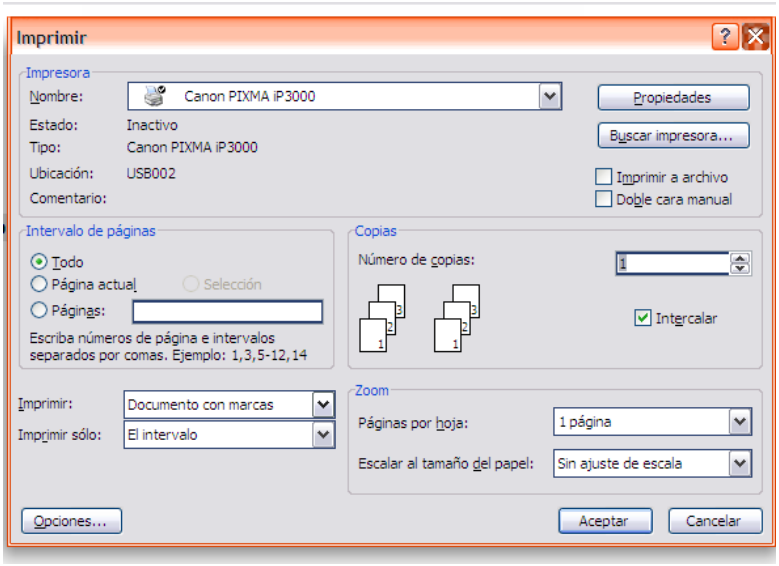
En el menú de Búsqueda podemos realizar búsquedas por:

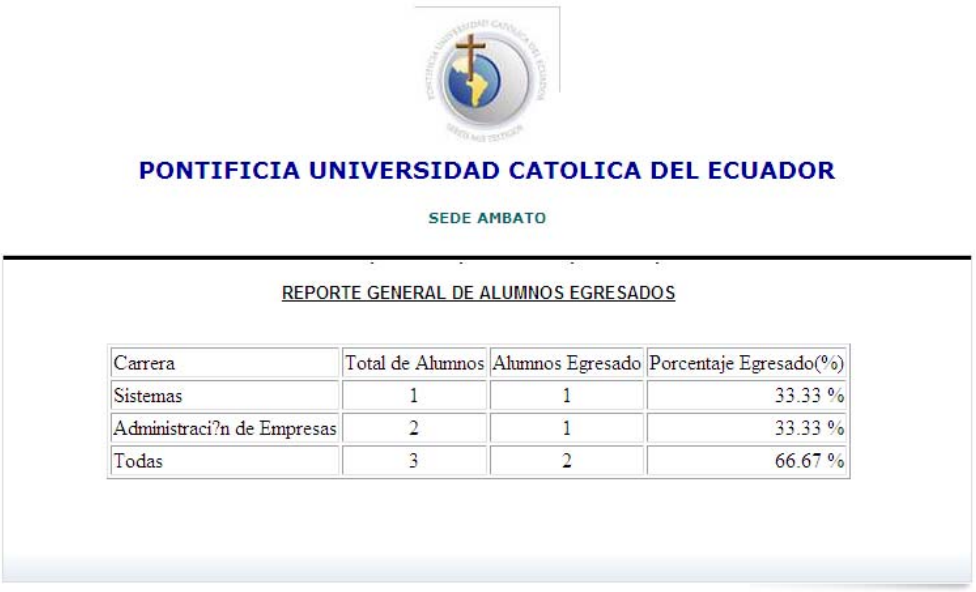
- Carrera
- Apellido
- Periodo
- Estado
- Cedula



A.8. Impresión de Reportes

Para la impresión de los reportes se dará un clic en el botón imprimir donde se despliega la siguiente ventana y se imprimirán los resultados del reporte.





PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR

SEDE AMBATO

REPORTE GENERAL DE ALUMNOS EGRESADOS

Carrera	Total de Alumnos	Alumnos Egresado	Porcentaje Egresado(%)
Sistemas	1	1	33.33 %
Administraci?n de Empresas	2	1	33.33 %
Todas	3	2	66.67 %

B. Manual de instalacion

En este manual se incluyen las instrucciones para instalar, configurar y usar los distintos programas y paquetes software utilizados en la aplicación.

Estos son:

- Appserve 2.5.5.
- Apache Utilizado como servidor de páginas Web.
- PHP Utilizado como lenguaje de programación de páginas Web.
- Un directorio con los archivos del sitio web llamado "PUCE"
- Un archivo de extensión .dump donde se encuentra la base de datos llamada "puce.dump"
- Programa de instalación appserve 2.5.5 (para SO Windows y Solaris)

B.1. Pasos para la Instalación del Sistema.

- Como primer paso, procedemos a la instalación del **appserve 2.5.5**, el mismo que nos instala la versión de PHP 4.1.7 y el motor de base de datos MySQL 4.1.14
- Grabamos el sitio Web que contiene los archivos del sistema en el directorio respectivo:
  - Windows. "D:\web\puce"
  - Solaris. "\usr\local\mysql\bin"
- Para cargar las base de datos del sistema es necesario copiar el archivo **puce.dump** en el directorio del **appserve**

C. Archivos del Sistema.

C.1. MySQL

sql.txt. Archivo que contiene la creación de la base de datos y de las tablas y unos datos mínimos para cada tabla. Este archivo se cargará una sola vez al instalar la aplicación.

C.2. PHP

Todos los archivos de la aplicación tienen extensión .php y se sitúan en el directorio " Windows. "D:\web\puce" o Solaris. "\usr\local\mysql\bin"" que a su vez se sitúa en el directorio raíz del servidor.

Los archivos en cuestión son:

- admin.php
- anio.php
- buscar2.php
- buscar.php
- carrera.php
- escuela.php
- estado.php
- idioma.php
- login.php
- logout.php
- menu2.php
- menu.php
- nivel.php
- ocupacion.php
- periodo.php
- registro.php
- reportes.php
- user.php

D. Glosario de Términos

API. Application Programming Interface. Interfaz de Programación de Aplicaciones.

ASP. Acitve Server Pages. Paginas de Servidor Activas.

CGI. Common Gateway Interface, Interfaz Comun de Pasarela.

BD Base de Datos

DBM. Data Base Management.

DER Diagrama Entidad-Relación, utilizado en el modelo de análisis estructurado.

DFD Diagrama de Flujo de Datos, utilizado en el modelo de análisis estructurado.

ERS. Especificaiocnes de Requisitos del Software

HTML HyperText Markup Language, Lenguaje de marcación de hypertexto.

HTTP HyperText Transfer Protocol, Protocolo de Transferencia de Hypertexto.

JSP. Java Server Pages

MCAL. Modular Calendar Access Library.

Página Web Documento digital que contiene información específica de un tema en particular y que es almacenado en algún sistema de cómputo que se encuentre conectado a la red mundial de información denominada Internet, de tal forma que este documento pueda ser consultado por cualquier persona que se conecte a esta red mundial de comunicaciones y que cuente con los permisos apropiados para hacerlo. Una página Web es la unidad básica del World Wide Web.

PERL. Practical Extracting and Reporting Lenguaje.

SGBD. Sistema de Gestion de Base de Datos.

Sitio Web Conjunto de archivos electrónicos y páginas Web referentes a un tema en particular, que incluye una página inicial de bienvenida, generalmente denominada home page, con un nombre de dominio y dirección en Internet específicos.

SME. Sistema de Monitoreo de Egresados

SQL Strudtured Querry Language, lenguaje estructurado de consultas para bases de datos..

TCP. Transmission Control Protocol, Protocolo de Control de Transmisión

URL. Uniform Resource Locutor, Lcalizador Uniforme de Recurso.

Web World Wide Web, o simplemente Web, es el universo de información accesible a través de Internet.

XHTML eXtensible HyperText Markup Language (lenguaje extensible de marcado de hipertexto).

XML eXtensible Markup Language, Lenguaje de Marcado Extensible.

XSL eXtensible Stylesheet Language, (lenguaje extensible de hojas de estilo), lenguaje para transformar, formatear y presentar la información de un documento XML en un medio específico.