PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

SEDE AMBATO

ESCUELA DE INGENIERÍA DE SISTEMAS

DISERTACIÓN DE GRADO PREVIA LA OBTENCIÓN DEL TÍTULO

DE INGENIERÍA DE SISTEMAS

"DESARROLLO E IMPLEMENTACIÓN DEL SISTEMA DE EVALUACIÓN DOCENTE VÍA PÁGINA WEB UTILIZANDO LÓGICA DIFUSA PARA LA PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR SEDE AMBATO"

CHÁVEZ ANDRADE JORGE VLADIMIR

DIRECTOR DE LA DISERTACIÓN

ING. MSC. TELMO VITERI

AMBATO, 2006

PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

SEDE AMBATO

ESCUELA DE INGENIERÍA DE SISTEMAS

DISERTACIÓN DE GRADO PREVIA LA OBTENCIÓN DEL TÍTULO DE INGENIERÍA DE SISTEMAS

"DESARROLLO E IMPLEMENTACIÓN DEL SISTEMA DE EVALUACIÓN DOCENTE VÍA PÁGINA WEB UTILIZANDO LÓGICA DIFUSA PARA LA PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR SEDE AMBATO"

DIRECTOR:

ING. MSC. TELMO VITERI

ASESOR

AUTORÍA DE LA TESIS

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

Chávez Andrade Jorge Vladimir
C.I.: 1803541125

DEDICATORIA

Este trabajo esta respetuosamente dedicado a la memoria de mi padre y mi madre. Quienes motivaron y crearon un amor incondicional en mi vida, sin ellos nada hubiese sido igual.

AGRADECIMIENTO

Es indispensable indicar que dada la vastedad y complejidad que trata este proyecto, agradecer a colegas, familiares y amigos que me han prestado ayuda con sus conocimientos. Mención especial merece Ing. Telmo Viteri, Director de la Escuela de Sistemas, Ing. Diego Santacruz, Administrador del Centro de Cómputo de la universidad. Ing. Marcelo Tamayo, Colaborador del Departamento de Sistemas de la Universidad Católica de Ibarra, con quienes he compartido no solo conocimientos técnicos sino una vida dedicada al esfuerzo personal por salir adelante frente a cualquier obstáculo.

No puedo hacer de menos constar explícitamente mi gratitud al personal administrativo, directores y profesorado de la universidad, quienes me brindaron su apoyo para iniciar y concluir el presente trabajo.

Mi más anhelado deseo es que este proyecto sirva para fortalecer y enaltecer la calidad docente de la universidad, contribuyendo con los intereses propios de los estudiantes que deseamos destacar y colaborar con la colectividad universitaria y profesional del país.

J.V.C.A.

ÍNDICE GENERAL

AUTORÍA DE LA TESIS ..... **iii**
DEDICATORIA ..... **iv**
AGRADECIMIENTO ..... **v**
ÍNDICE GENERAL ..... **vi**
ÍNDICE DE TABLAS ..... **xii**
INTRODUCCIÓN **1**
1. CAPITULO I. Proyecto de la Investigación. **2**
1.1. Planteamiento del Problema. 2
1.1.1. Problema. 2
1.1.2. Problematización. 2
1.2. Delimitación. 2
1.3. Importancia y Justificación. 3
1.3.1. Importancia. 3
1.3.2. Justificación. 3
1.4. Objetivos. 4
1.4.1. Objetivo General. 4
1.4.2. Objetivos Específicos. 4
1.5. Hipótesis. 4
1.6. Aspectos Metodológicos. 4
1.6.1. Fundamentos Teóricos. 4
1.6.2. Métodos de Investigación. 5
2. CAPITULO II. Marco Teórico. **6**
2.1. Evaluación Docente. 6
2.1.1. Introducción. 6
2.1.2. Concepto de Evaluación Docente. 6
2.1.3. Aspectos de la Evaluación. 7
2.1.3.1. Definición de Evaluación. 7
2.1.4. Características de la evaluación educativa. 7
2.1.5. Actividades de la Evaluación Universitaria. 8
2.1.6. Evaluación Cualitativa. 10
2.1.6.1. Fundamentación Teórica. 10
2.1.7. Autoevaluación. 11

2.2. Inteligencia Artificial 12
2.2.1. Introducción. 12
2.2.2. Definiciones. 12
2.2.3. Objetivos de la Inteligencia Artificial. 13
2.2.4. Ramas de la Inteligencia Artificial. 13
2.2.5. Alcance de la Inteligencia Artificial. 14
2.2.6. Sistemas Expertos. 15
2.2.6.1. Introducción. 15
2.2.6.2. Definición. 16
2.2.6.3. Ventajas de los Sistemas Expertos. 17
2.2.6.4. Limitaciones de los Sistemas Expertos. 18
2.2.7. Sistemas de Lógica Difusa. 18
2.2.7.1. Introducción. 18
2.2.7.2. Aplicaciones de Lógica Difusa. 19
2.2.7.3. Ventajas de Lógica Difusa. 20
2.2.7.4. Conjuntos Difusos. 20
2.2.7.5. Notación de Conjuntos Difusos. 21
2.2.7.6. Mapeo de Conjuntos Difusos. 22
2.2.7.7. Operaciones. 23
2.3. Ingeniería de Software. 27
2.3.1. Definiciones. 27
2.3.2. Objetivos. 28
2.3.3. Software. 28
2.3.3.1. Definición de Software. 28
2.3.3.2. Características del Software. 28
2.3.3.3. Calidad de Software. 29
2.3.3.4. Aplicaciones de Software. 30
2.3.4. Fundamentos y Técnicas de Ingeniería de Software. 32
2.3.5. Metodología. 32
2.3.6. Ciclo de Vida. 32
2.4. Base de Datos. 35
2.4.1. Definiciones. 35
2.4.2. Ventajas de las Bases de Datos. 36
2.4.3. Sistema de Gestión de la Base de Datos (SGBD). 36
2.4.4. Modelos de Datos. 36

2.4.4.1. Modelos Lógicos Basados en Objetos. 36
2.4.4.1.1. Modelo Entidad-Relación. 37
2.4.4.1.2. Modelo Orientado a Objetos. 37
2.4.4.2. Modelos Lógicos Basados en Registros. 38
2.4.4.2.1. Modelo Relacional. 38
2.4.4.2.2. Modelo de Red. 38
2.4.4.2.3. Modelo Jerárquico. 39
2.4.4.3. Modelos Físico de Datos. 39
2.5. Motor de Base de Datos MySQL. 40
2.5.1. Introducción. 40
2.5.2. Creación de Base de Datos. 40
2.5.3. Creación de Tablas. 41
2.5.4. Ingreso de Datos. 43
2.5.5. Actualización de Datos. 44
2.5.6. Eliminación de Datos. 44
2.5.7. Selección de Datos. 45
2.6. Programación Web. 46
2.6.1. Funcionamiento de un Sitio Web. 46
2.6.2. El protocolo HTTP. 46
2.6.3. Ciencias aplicadas para el desarrollo de aplicaciones Web. 47
2.6.4. Programación del lado del Servidor. 49
2.6.4.1. El protocolo CGI. 49
2.6.4.2. Uso de una API del Servidor. 50
2.6.4.3. Uso de un módulo del Web-Server. 51
2.6.5. Evolución de los Web-Sites. 51
2.6.5.1. Web-Sites de primera generación. 51
2.6.5.2. Web-Sites de segunda generación. 52
2.6.5.3. Web-Sites de tercera generación. 52
2.6.5.4. Lenguajes de Programación Web. 53
2.6.5.4.1. HTML 53
2.6.5.4.2. Javascript. 53
2.6.5.4.3. VB Script. 54
2.6.5.4.4. DHTML. 54
2.6.5.4.5. CSS. 55
2.6.5.4.6. Applets de Java. 55

2.6.5.4.7. PERL. 56
2.6.5.4.8. ASP. 56
2.6.5.4.9. PHP. 57
2.6.5.4.10. JSP. 57
2.6.5.4.11. XML. 58
2.7. Lenguaje de programación PHP. 59
2.7.1. Generalidades. 59
2.7.1.1. Introducción. 59
2.7.1.2. Generalidades. 59
2.7.1.3. Conceptos Básicos. Generación de Sitios Dinámicos con PHP. 60
2.7.1.4. Funcionalidad. 61
2.7.2. Introducción al Lenguaje. 61
2.7.2.1. Comentarios. 62
2.7.2.2. Tipos de Datos. 62
2.7.2.3. Constantes. 63
2.7.2.4. Operadores. 63
2.7.2.5. Asignación. 63
2.7.2.6. Comparaciones. 64
2.7.2.7. Operador @. 64
2.7.2.8. Operadores Lógicos. 64
2.7.2.9. Estructuras de Control. 64
2.7.2.10. Funciones. 66
2.7.3. Manejo de Base de Datos (MySQL). 67
2.7.3.1. Conexión a la Base. 67
2.7.3.2. Selección de la Base de Datos. 67
2.7.3.3. Queries a la Base de Datos. 68
2.7.3.4. Cantidad de Filas Consultadas o Modificadas. 68
2.7.3.5. Obtención de registros de una Consulta. 68
2.7.4. Programación Orientada a Objetos en PHP. 69
2.7.4.1. Introducción 69
2.7.4.2. Conceptos de Programación Orientada a Objetos. 69
2.7.4.2.1. Encapsulación. 69
2.7.4.2.2. Herencia. 70
2.7.4.2.3. Polimorfismo. 70
2.7.5. Persistencia. 71

2.7.5.1. Sesiones. ... 71
2.7.5.2. Cookies. ... 71
2.7.5.3. URL. ... 72
2.7.5.4. Sesiones en PHP. ... 73
2.7.6. Manejo de HTTP en PHP. ... 74
2.7.6.1. Headers. ... 74
3. CAPITULO III. Desarrollo del Proyecto. ... **75**
3.1. Análisis del Sistema. ... 75
3.1.1. Identificación de las necesidades. ... 75
3.1.2. Estudio de Viabilidad. ... 76
3.1.3. Análisis Técnico. ... 76
3.1.4. Análisis de Requisitos. ... 77
3.1.4.1. Especificaciones. ... 78
3.1.5. Análisis Estructurado. ... 79
3.1.5.1. Diagrama de Flujo de Datos Nivel 0. ... 79
3.1.5.2. Diagrama de Flujo de Datos Nivel 1. ... 80
3.1.5.3. Diagrama de Flujo de Datos Nivel 2. ... 81
3.2. Diseño del Sistema. ... 82
3.2.1. Diseño de la Base de Datos. ... 83
3.2.2. Diseño del Sitio Web y la Interfase. ... 90
3.2.3. Diseño de Procedimientos para el Cálculo de Evaluación. ... 94
3.3. Desarrollo del sistema. ... 97
3.3.1. Creación de la Base de Datos. ... 97
3.3.2. Codificación de Cálculos de Evaluación. ... 105
3.3.2.1. Class conexion. ... 105
3.3.2.2. Class difusidad. ... 105
3.3.2.3. Class variables. ... 106
3.3.2.4. Class centro_gravedad. ... 107
3.3.2.5. Class auto_cg. ... 107
3.3.2.6. Class evaluacion. ... 107
3.3.2.7. Class observacion. ... 108
3.4. Pruebas e Instalación. ... 109
3.4.1. Pruebas. ... 109
3.4.2. Instalación. ... 109

4. CAPITULO IV. Verificación Y Validación de Resultados. 111
4.1. Verificación. 111
4.2. Validación. 112
4.3. Conclusiones. 113
4.4. Recomendaciones. 114
ANEXOS. 115
A. Manual Técnico 115
B. Manual de Usuario. 126
BIBLIOGRAFÍA. 164
Libros: 164
Páginas Web: 165

ÍNDICE DE TABLAS

Figura 2.1: El proceso de Evaluación de Servicios..... 10
Figura 2.2: Mapeo Lógica Booleana..... 22
Figura 2.3: Mapeo Lógica Difusa. 23
Figura 2.4: Modelo de Ciclo de Vida en Cascada. 33
Figura 2.5: Modelo de Ciclo de Vida en Cascada (Variación)..... 33
Figura 2.6: Modelo de Ciclo de Vida en Cascada (Mejora iterativa)..... 34

Figura 3.1: Diagrama de Flujo de Datos Nivel 0..... 79
Figura 3.2: Diagrama de Flujo de Datos Nivel 1..... 80
Figura 3.3: Diagrama de Flujo de Datos Nivel 2..... 81
Figura 3.4: Diagrama Entidad - Relación..... 84
Figura 3.5: Mapa del Sitio..... 90

INTRODUCCIÓN

Con el fin de alcanzar y enaltecer el desarrollo y fortalecimiento de la profesión docente en su labor profesional para el mejoramiento de sus actividades, buscando favorecer al mejor aprendizaje de los estudiantes; el sistema de evaluación tiene un carácter fundamentalmente formativo y de valiosa importancia.

El presente trabajo consiste en el Desarrollo e Implementación de un Sistema de Evaluación Docente vía Página Web utilizando Lógica Difusa para la Pontificia Universidad Católica del Ecuador Sede Ambato.

Introduciendo conceptos de Lógica Difusa se va a lograr alcanzar una evaluación cualitativa a cada uno de los Docentes que conforman el Profesorado de la PUCESA, así como su rápida, fácil y eficaz utilización de este sistema por los estudiantes a través de la Pagina Web del la Universidad.

1. CAPITULO I. Proyecto de la Investigación.

1.1. Planteamiento del Problema.

1.1.1. Problema.

Carencia de un Sistema de Evaluación Docente Automatizado en la Pontificia Universidad Católica del Ecuador Sede Ambato periodo 2004 - 2005, lo que dificulta el proceso de evaluación docente.

1.1.2. Problematización.

- Ausencia de un diagnóstico sobre sistemas de evaluación docentes vía Internet en las Universidades del Ecuador
- Falta de un estudio sobre las evaluaciones docentes cualitativas mediante Sistemas Expertos.
- Carencia de una evaluación sistematizada a los docentes de la PUCESA.

1.2. Delimitación.

La elaboración del proyecto planteado se desarrollará exclusivamente para los docentes de la Pontificia Universidad Católica del Ecuador sede Ambato. El tiempo estimado para solucionar el presente problema será de nueve a diez meses a partir de la fecha aprobación del plan de disertación.

El Sistema de Evaluación contará con una base de datos propia hecha en el motor de base de datos MySQL que se alojará en el servidor SUN e250 del Centro de Cómputo de la Universidad; y se utilizará el lenguaje PHP (acrónimo: Hypertext Preprocessor), que es un lenguaje "Open Source" interpretado de alto nivel, especialmente pensado para desarrollos Web, el cual es utilizado en el mismo servidor.

Se utilizará como medio de evaluación preguntas con una calificación determinada, dichas preguntas serán recuperadas de la base de datos explicada

anteriormente; y a través de Cálculos y Métodos de Lógica Difusa procesar los resultados obtenidos y brindar la suficiente información por cada profesor.

Realizará reportes generales a Pro-Rectorado y reportes específicos por escuela a los Directores de cada una de ellas, en cuadros comparativos en categorías; así como también brindará las respectivas conclusiones y recomendaciones a cada uno de los docentes autoevaluados estableciendo su desempeño en uno de los cuatro niveles siguientes: destacado, competente, básico e insatisfactorio.

1.3. Importancia y Justificación.

1.3.1. Importancia.

Para todos los profesores, la evaluación significará un proceso de autorreflexión respecto de su desenvolvimiento profesional, lo que permitirá proyectar un camino para potenciar fortalezas y superar debilidades.

A través de este proyecto de disertación se alcanzará la automatización del proceso de Evaluación Docente y se obtendrá de una forma real y cualitativa lo que muestra cada una de las evaluaciones que se realicen.

1.3.2. Justificación.

El trabajo se justifica ya que el Sistema de Evaluación Docente realizado para la PUCESA es el primer sistema a nivel nacional que utiliza cálculos y métodos de Lógica Difusa vía Página Web y servirá de base para el desarrollo de sistemas similares en el país.

Existen los conocimientos necesarios para el desarrollo e implementación del Sistema de Evaluación; además los costos son accesibles para su elaboración; se cuenta también con el Software y Hardware necesarios para la implementación y funcionamiento y el asesoramiento de la Dirección de la Es-

cuela de Sistemas así como del Centro de Cómputo para el Hosting del Sistema de Evaluación.

1.4. Objetivos.

1.4.1. Objetivo General.

Desarrollo e Implementación de un Sistema Automatizado de Evaluación Docente vía Página Web utilizando Lógica Difusa brindando resultados verdaderos de forma confiable y en tiempo real.

1.4.2. Objetivos Específicos.

- Lograr que la PUCESA sea pionera en sistemas de evaluación docente vía Internet en las Universidades del Ecuador.
- Alcanzar mediante Sistemas Expertos un estudio sobre las evaluaciones docentes cualitativas.
- Obtener una evaluación sistematizada a los docentes de la PUCESA.

1.5. Hipótesis.

Con la utilización del Sistema de Evaluación Docente vía Página Web utilizando Lógica Difusa, se logrará sistematizar las evaluaciones de todos los docentes de la PUCESA en tiempo real.

1.6. Aspectos Metodológicos.

1.6.1. Fundamentos Teóricos.

Este proyecto de disertación utiliza los siguientes paradigmas:

- Interpretativista, ya que este muestra la comunicación de los estudiantes como un fenómeno que arrojará resultados al evaluar a los docentes.

- Crítico, ya que intenta transmitir mensajes por medio de los resultados obtenidos y conseguir resultados positivos en la educación universitaria.
- Racionalista, debido a que se cuenta con el conocimiento e instrumentos necesarios, este proyecto se puede llevar a la realidad.

1.6.2. Métodos de Investigación.

En el proceso de investigación se utilizará el método de la investigación aplicada, también conocida como activa o dinámica, corresponde al estudio y aplicación de la investigación al problema de evaluación docente definidos en la finalización de los periodos de estudio y con calificaciones que se encuentran en rangos determinados.

Entre los tipos de investigación aplicada utilizaremos la descriptiva, ya que nuestro objeto fundamental será interpretar la realidad de los resultados obtenidos por la Evaluación Docente.

También utilizaremos la Experimentación Científica ya que utilizaremos variables experimentales, bajo condiciones estrictamente comprobadas con el objeto de describir los resultados y brindar recomendaciones.

2. CAPITULO II. Marco Teórico.

2.1. Evaluación Docente.

2.1.1. Introducción.

Los cuestionarios, métodos de observación y demás instrumentos que pueden servir al proceso de evaluación deben tener condiciones de objetividad, valides, exactitud y control de los datos recopilados. Esto implica no solo la recolección de datos y un proceso interpretativista de la información recolectada; sino también la emisión de juicios de valor fundamentados.

2.1.2. Concepto de Evaluación Docente.

El concepto de evaluación docente está estrechamente ligado al de aprendizaje. Si el aprendizaje es el cambio o adquisición de una conducta que anteriormente no se tenía, la evaluación sería el intento de averiguar sistemáticamente en qué medida se han logrado las conductas formuladas en los objetivos iniciales.

La evaluación es por tanto un conjunto de estímulos (más o menos complejos) que intentan verificar la existencia de unas conductas y la medida aproximada de su desarrollo. Se trata pues, de asegurarse de forma continua de en qué medida o proporción se ha aprendido realmente lo que se pretendía enseñar.

El sentido de la evaluación docente es el comprobar y mejorar la eficacia del aprendizaje. De esto se deduce que la evaluación no es una sanción en el sentido de castigar o premiar con “notas”, sino un permanente control de calidad con la idea de mejorar el producto.

De la necesidad de evaluar (en el doble sentido que ya hemos explicado), se deriva la necesidad de las calificaciones. Así las calificaciones son necesarias para el docente como medio de verificar la eficacia de su propia enseñan-

za, para la administración educativa como criterio para la promoción de alumnos, en la orientación profesional como dato de intereses o aptitudes, para los futuros empleadores como elemento de información, y para los propios alumnos como sistema para conocer sus capacidades, limitaciones y deficiencias.

2.1.3. Aspectos de la Evaluación.

La evaluación es una actividad por la cual se precisa si un objeto logra un determinado nivel de excelencia con el propósito de tomar medidas para el mejoramiento o abandono del mismo. Es importante que la evaluación tenga fundamentos científicos al analizar la información y brindar conclusiones. Los requisitos básicos son de disponer de un conjunto de criterios de valores y tener a disposición un sistema de recolección de la información o datos de evaluación.

2.1.3.1. Definición de Evaluación.

Beeby, 1977. La evaluación supondría una recogida sistemática e interpretación de la evidencia que conduce, como parte del proceso, a un juicio de valor con vistas a la acción.

Stufflebeam y Shinkfield, 1987. La evaluación es el enjuiciamiento sistemático de la valía o el mérito de un objeto... Un estudio sistemático planificado, dirigido y realizado con el fin de ayudar a un grupo de clientes a juzgar y/o perfeccionar el valor y/o el mérito de algún objeto[1].

2.1.4. Características de la evaluación educativa.

Independiente y por ello comprometida. Independiente se refiere una evaluación que no este sometida por el poder.

[1] **Join Committee on Standards for Educational Evaluation (1981),** citado en Stufflebean y Shinkfield (1987, pp 19 y 67)

Cualitativa y no meramente cualificable. Porque los procesos que se analizan en programas educativos son muy complejos y la reducción a números suele simplificar la parte esencial de los mismos.

Práctica y no meramente especulativa. La evaluación tiene por finalidad la mejora de programas a través de su comprensión, conocimiento de su naturaleza, funcionamiento y resultados.

Procedimental, no meramente final. Esta se realiza durante todo el proceso, de esta manera su desarrollo se puede modificar y mejorar.

Asociada, no individualista. La evaluación la debe asumir un equipo no solo un individuo. Garantizando la pluralidad del enfoque al que quiere llegar la evaluación.

Los criterios de Evaluación. Criterio es la norma para juzgar o atribuir valor a una realidad. Son los criterios un elemento fundamental para poder valorar cualquier realidad educativa.

2.1.5. Actividades de la Evaluación Universitaria.

La evaluación sigue un enfoque clásico en el que predomina la necesidad de planificar y valorar el rendimiento de acuerdo con las características del contexto institucional. En otras palabras, las actividades de evaluación han de ser coherentes con el modelo tradicional de la intervención social, siendo útiles para valorar la calidad de las decisiones y acciones específicas propias de cada fase de la intervención (diagnóstico, planificación, intervención, evaluación).

Las siguientes actividades de evaluación son consideradas útiles para dirigir el cambio en los sistemas de gestión de los servicios universitarios:

Evaluación de contexto. Actividad relacionada con la identificación de usuarios, necesidades y definición de problemas. La evaluación de necesidades, ya provenga de los clientes del servicio o hayan sido originadas por las deficiencias en la gestión del mismo, constituye la actividad más importante en esta fase inicial.

Evaluación de entrada. Tiene como objetivo identificar y valorar la capacidad del sistema y de las estrategias de acción, la planificación de la logística para llevar a cabo las estrategias, y los presupuestos (recursos) que se ponen a disposición de los programas de mejora, incluidos los económicos, humanos y conceptuales, entre otros. La evaluación de la evaluabilidad, centrada específicamente en la calidad, la coherencia interna y la viabilidad del plan de mejora, es la actividad estrella en este contexto.

Evaluación de proceso. Estrechamente vinculada con la planificación y pronóstico de los defectos de la planificación, del procedimiento o de su realización, y con el seguimiento de las actividades de mejora. Cumple una función formativa y de retroalimentación de gran utilidad para corregir desviaciones del plan y reorientar la intervención en caso necesario. La evaluación de la implementación y el seguimiento son actividades nucleares en este apartado.

Evaluación de resultados (impactos). Conocida como evaluación sumativa, comprende la cuantificación y valoración de todo tipo de efectos, impactos y cambios causados por el plan de intervención (o de mejora), ya sean positivos, negativos, deseados, no deseados, inmediatos, demorados, o de cualquier otro tipo.

Evaluación comparativa. Supone una visión global del proceso de cambio, que implica el juicio estimativo de si el proceso estuvo bien orientado, si los recursos fueron adecuados y si los resultados fueron satisfactorios a un determinado coste en comparación con otras opciones que pudieron haberse intentado. Los estudios económicos de coste-beneficio y coste-eficacia resultan determinantes en este tipo de análisis.

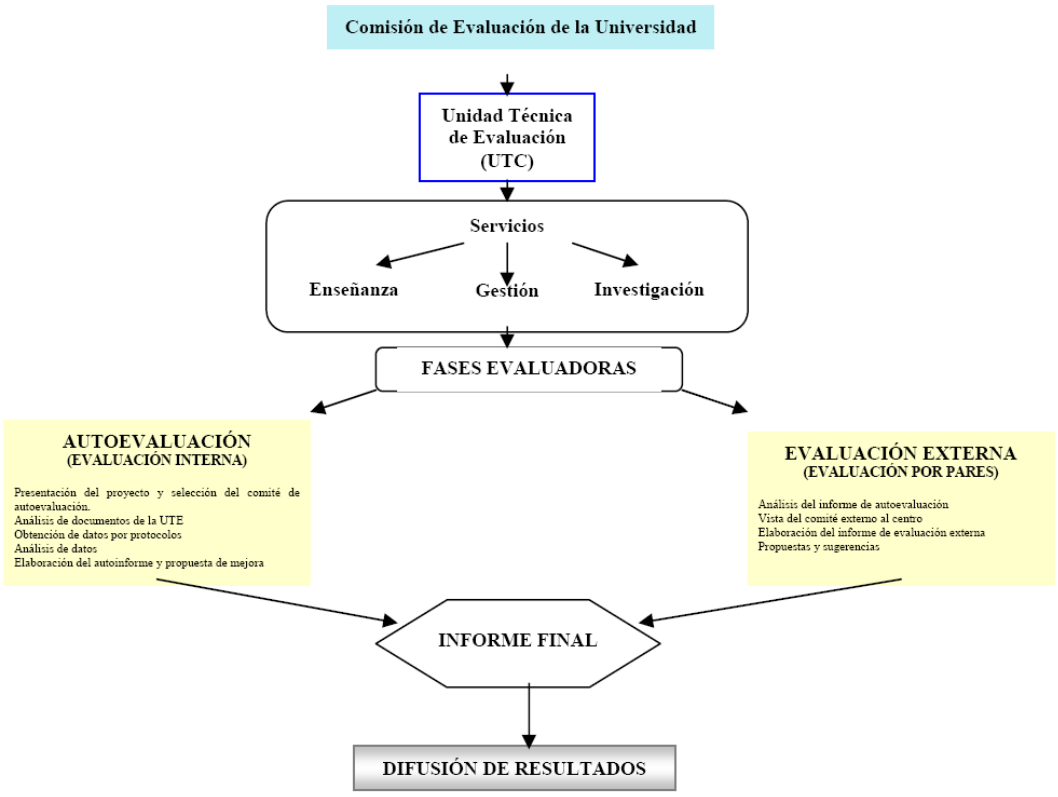


Figura 2.1: El proceso de Evaluación de Servicios[2].

2.1.6. Evaluación Cualitativa.

La evaluación cualitativa pretende describir e interpretar más que medir y predecir. La metodología cualitativa se utiliza en investigación educativa porque su misma concepción implica el estudio de procesos e interacciones entre unos sujetos y su medio, teniendo en cuenta todas las variables previstas y no previstas.

2.1.6.1. Fundamentación Teórica.

Parte del paradigma naturalista. Tiene por objetivo conocer como se producen los procesos educativos en su entorno natural, cómo se construyen tales procesos a través de las interpretaciones y las actividades de sus protagonistas.

[2] **Plan Andaluz de la Calidad de las Universidades 2001 - 2006**

Aspira a explicar los distintos aspectos y fenómenos de la realidad educativa, pero asume que esa realidad no es solamente externa, observable y cuantificable numéricamente, sino que también es dependiente de cómo los actores educativos la interpretan y vivencian.

Busca comprender el comportamiento de los sujetos implicados en un proceso, intentando captar el propio proceso en su totalidad, las interacciones y sus significados entre los sujetos con el medio ambiente, sin dejar de lado variables imprevistas.

Percibe la vida social como la creatividad compartida de los individuos. En contextos compartidos u organizaciones exentas de racionalismo que se acerquen al modelo natural.

2.1.7. Autoevaluación.

De Alba, 1991. El término autoevaluación se empezó a emplear para la evaluación del aprendizaje, entendida como un proceso de reflexión y análisis critico referido a una situación, problemática o proceso social, a partir de concepciones y valoraciones iniciales que se van reformulando en el proceso mismo.

Simona, 1985. La autoevaluación es una tendencia cada vez más extendida en el medio educativo, apoyada tanto por proyectos e iniciativas, como por acciones académicas y de formación, que tienen existencia nacional e internacional. Se trata de una tendencia contemporánea que procura la existencia de procesos pedagógicos más cooperativos, en los que la información permanente se garantice por medio del educador, mediante el análisis reflexivo de las interacciones y procesos que se dan en el aula, encaminándose más hacia el desarrollo profesional que hacia el control administrativo.

2.2. Inteligencia Artificial

2.2.1. Introducción.

La inteligencia artificial es un área de la investigación donde se desarrollan algoritmos para controlar cosas. Son muchos los estudios y aplicaciones que se han logrado con el desarrollo de esta ciencia, entre las cuales tenemos redes neuronales aplicadas al control de la calidad, control de procesos químicos, algoritmos genéticos, autómatas programables, etc.

Desde el punto de vista de ingeniería, la inteligencia artificial sirve para resolver problemas reales actuando como un conjunto de ideas para representar el conocimiento y desarrollarlo en sistemas informáticos.

Por otro lado, desde el punto de vista de ciencia, la inteligencia artificial busca la explicación de diversas clases de inteligencia.

2.2.2. Definiciones.

Rich & Knight [1994], Stuart [1996]. Estos autores definen a la inteligencia artificial como la capacidad que tienen las máquinas para realizar tareas que en el momento son realizadas por seres humanos.

Nebendah [1988], Delgado [1998]. Otros autores la definen como el campo de estudio que se enfoca en la explicación y emulación de la conducta inteligente en función de procesos computacionales basadas en la experiencia y el conocimiento continuo del ambiente.

Marr [1977], Mompin [1987], Rolston [1992]. Estos autores en sus definiciones involucran los términos de soluciones a problemas muy complejos.

2.2.3. Objetivos de la Inteligencia Artificial.

El objetivo de la inteligencia artificial básicamente trata de comprender y construir entidades inteligentes. Existen otros planteamientos como los siguientes:

- Desarrollar una máquina inteligente capaz de aprender a través de la experiencia, reconocer las limitaciones de su conocimiento, exhibir verdadera creatividad, tomar sus propias decisiones e interactuar con el medio que la rodee.
- Hacer que las computadoras sean capaces de mostrar un comportamiento que sea considerado como inteligente por parte de un observador humano.
- Elevar el coeficiente intelectual de las máquinas.
- Desarrollar las capacidades de la computadora más allá de su uso tradicional actual.

2.2.4. Ramas de la Inteligencia Artificial.

- Sistemas expertos ó sistemas basados en conocimiento.
- Visión por computadora, reconocimiento de patrones, lenguaje natural.
- Robótica, realidad virtual, edificios inteligentes y vehículos autónomos.
- Vida artificial, nanotecnología, agentes y aprendizaje de máquina.
- Sistemas Neru-difusos.
    - Lógica Difusa.
    - Redes Neurales.
- Computación evolutiva.
    - Algoritmos genéticos.
    - Programación genética.

2.2.5. Alcance de la Inteligencia Artificial.

Solución de problemas en general. Crear un sistema capaz de hallar métodos para solucionar problemas.

Percepción. Al utilizar censores y controladores en circuitos integrados, el computador es capaz de interactuar con el exterior.

Aprendizaje, demostración de teoremas, juegos. Una característica básica de la inteligencia artificial es la búsqueda de algoritmo que permitan incorporar esta capacidad a un sistema.

Lenguaje. Los lenguajes de programación tradicionales no se han adaptado bien a las aplicaciones en IA. Se han usado ampliamente lenguajes como Prolog, basados en la lógica y Lips que tiene una superior capacidad en el procesamiento de listas, pero existen muchos más lenguajes diseñados especialmente para algunas aplicaciones concretas.

Robótica. La ciencia de la robótica implica diferentes técnicas de inteligencia artificial. La idea de un robot "listo" con la capacidad de aprender por experiencia es el tema central de teorías e investigaciones en IA. Un robot, por poner un ejemplo, debe ser capaz de comunicarse en lenguaje natural y debe poder realizar tareas con iniciativa y originalidad, esto implica que el robot debe llegar a realizar, tras un periodo de aprendizaje, cosas para las cuales no estaba inicialmente programado; a diferencia de los robots que se utilizan actualmente en la aplicación industrial, los cuales no son más que meros autómatas.

Cibernética. La cibernética es una ciencia interdisciplinaria, tratando con sistemas de comunicación y control sobre organismos vivos, máquinas u organizaciones. En el cuerpo humano, el cerebro y el sistema nervioso funcionan para coordinar la información, la cual es utilizada para determinar el futuro

curso de una acción; controlar los mecanismos para la autocorrección en máquinas que sirven con un propósito similar. Este principio es conocido como retroalimentación, el cual es fundamental en el concepto de automatización.

La cibernética también se aplica al estudio de la psicología, servomecanismo, economía, neuropsicología, ingeniería en sistemas y al estudio de sistemas sociales, el término cibernética no es muy utilizado para describir por separado a un campo de estudio, y muchas de las investigaciones en el campo ahora se centran en el estudio y diseño de redes neuronales artificiales.

2.2.6. Sistemas Expertos.

2.2.6.1. Introducción.

Los sistemas expertos son la rama de la inteligencia artificial más empleada en gestión empresarial. Su origen se sitúa a mediados de los años setenta, aunque es a partir de la década de los ochenta cuando se desarrollan aplicaciones en toda su plenitud. Su objetivo es modelizar el conocimiento, representándolo en forma de símbolos. Muchos de los sistemas expertos existentes actualmente consisten en grandes bases de conocimientos, creadas para almacenar la información de que se dispone expertos humanos en varios campos y a las que se aplica una serie de reglas de manipulación expresadas en lenguajes específicos.

Los sistemas expertos forman parte de un firme y verdadero avance en inteligencia artificial. Los sistemas expertos pueden incorporar miles de reglas. Los sistemas expertos son "auto explicativo", al contrario que en los programas convencionales, en los que el conocimiento como tal está encriptado junto al propio programa en forma de lenguaje de ordenador.

Estos intentan reproducir el razonamiento humano de forma simbólica. Es un tipo de programa de aplicación informática que adopta decisiones o resuelve problemas de un determinado campo, como los sistemas de producción, las

finanzas o la medicina, utilizando los conocimientos y las reglas analíticas definidas por los expertos en dicho campo. Los expertos solucionan los problemas utilizando una combinación de conocimientos basados en hechos y en su capacidad de razonamiento. En los sistemas expertos, estos dos elementos básicos están contenidos en dos componentes separados, aunque relacionados: una base de conocimientos y una máquina de deducción, o de inferencia. La base de conocimientos proporciona hechos objetivos y reglas sobre el tema, mientras que la máquina de deducción proporciona la capacidad de razonamiento que permite al sistema experto extraer conclusiones. Los sistemas expertos facilitan también herramientas adicionales en forma de interfases de usuario y los mecanismos de explicación. Las interfases de usuario, al igual que en cualquier otra aplicación, permiten al usuario formular consultas, proporcionar información e interactuar de otras formas con el sistema. Los mecanismos de explicación, la parte más fascinante de los sistemas expertos, permiten a los sistemas explicar o justificar sus conclusiones, y también posibilitan a los programadores verificar el funcionamiento de los propios sistemas.

2.2.6.2. Definición.

Stevens (1984). Los sistemas expertos son maquinas que piensan y razonan como un experto lo haría en una cierta especialidad o campo. Por ejemplo, un sistema experto en diagnóstico médico requeriría como datos los síntomas del paciente, los resultados de análisis clínicos y otros hechos relevantes, y, utilizando estos, buscaría en una base de datos la información necesaria para poder identificar la correspondiente enfermedad. Un Sistema Experto de verdad, no solo realiza las funciones tradicionales de manejar grandes cantidades de datos, sino que también manipula esos datos de forma tal que el resultado sea inteligible y tenga significado para responder a preguntas incluso no completamente especificadas.

2.2.6.3. Ventajas de los Sistemas Expertos.

Uno de los grandes problemas a los que se enfrenta el analista es el gran caudal de información a manejar, que puede afectar negativamente a la toma de decisiones. Ante tal circunstancia los sistemas expertos pueden ser unos valiosos aliados.

Los sistemas expertos recopilan los datos, los analizan y asimilan. Convierten grandes volúmenes de datos en información útil. En este sentido conviene distinguir un sistema experto de los simples programas de ordenador: también las hojas de cálculo, bases de datos y programas convencionales manejan habitualmente información y alivian la pesada tarea de realizar cálculos numéricos pero no incluyen procedimientos de representación del conocimiento.

A continuación se describen algunas utilidades que brindan los sistemas expertos:

- Con la ayuda de un Sistema Experto, personas con poca experiencia pueden resolver problemas que requieren un conocimiento formal especializado.
- Los Sistemas Expertos pueden obtener conclusiones y resolver problemas de forma más rápida que los expertos humanos.
- Los Sistemas Expertos razonan pero en base a un conocimiento adquirido y no tienen sitio para la subjetividad.
- Se ha comprobado que los Sistemas Expertos tienen al menos, la misma competencia que un especialista humano.
- El uso de Sistemas Expertos es especialmente recomendado en las siguientes situaciones:
    - Cuando los expertos humanos en una determinada materia son escasos.

- En situaciones complejas, donde la subjetividad humana puede llevar a conclusiones erróneas.
- Cuando es muy elevado el volumen de datos que ha de considerarse para obtener una conclusión.

2.2.6.4. Limitaciones de los Sistemas Expertos.

- Pueden resultar difíciles de programar y requieren de mantenimiento complejo.
- En caso de cambios se tendría que reprogramar el sistema.
- Dificultad para manipular información no estructurada especialmente la información incompleta, inconsistente o errónea.

2.2.7. Sistemas de Lógica Difusa.

2.2.7.1. Introducción.

La lógica difusa ha surgido como una herramienta lucrativa para el control de subsistemas y procesos industriales complejos. Los seres humanos pensamos y razonamos por medio de palabras y en grados entre dos estados por ejemplo blanco y negro ó frío y caliente, etc. Estos sistemas de lógica difusa son una mejora a los sistemas experto tradicionales, en el sentido de que permiten utilizar lenguaje humano como nosotros razonamos.

La lógica difusa es básicamente una lógica multievaluativa que permite valores intermedios para poder definir evaluaciones convencionales como si/no, verdadero/falso, blanco/negro, etc. Las nociones como “casi caliente” o “poco frío” pueden formulase matemáticamente y procesarse por computadores. De esta forma se ha realizado un intento de aplicar una forma más humana de pensar en la programación de computadoras.

2.2.7.2. Aplicaciones de Lógica Difusa.

Los Sistemas difusos, pueden ser usados para estimaciones, toma de decisiones y mecanismos de Sistemas de control como son: aire acondicionado, control de automóviles y algunos trabajos inteligentes como controladores de procesos industriales.

Un área de aplicación de la lógica difusa que ha crecido considerablemente en los últimos años es la extracción de información de bases de datos. En la extracción convencional de información no se permite el tipo de flexibilidad a la que los humanos estamos acostumbrados, porque el tipo de búsqueda que se efectuaría podía pasar por alto detalles importantes que para un ser humano resultarían obvios. Por ejemplo, si queremos detectar a agentes de ventas jóvenes que sean prometedores, tendríamos que definir qué entendemos por "joven" y por "prometedor". Si, por ejemplo, para nosotros, alguien de menos de 25 años es joven, entonces podríamos pasar por alto el desempeño de vendedores de 26 años.

La lógica difusa permite definir de manera más flexible el tipo de búsqueda que deseamos realizar, y nos proporcionaría mucha mayor información que las técnicas tradicionales.

A continuación se señalan los campos o procesos donde es y no es recomendable el uso de la lógica difusa.

Recomendable para:

- Para procesos muy complejos, cuando no hay un modelo matemático simple.
- Para procesos altamente no lineales.
- Si el procesamiento del conocimiento experto puede ser desempeñado.

No es recomendable si:

- El control convencional teóricamente rinde un resultado satisfactorio.
- Existe un modelo matemático fácilmente soluble y adecuado.
- El problema no es soluble.

2.2.7.3. Ventajas de Lógica Difusa.

- Pueden ser evaluadas mayor cantidad de variables.
- Variables lingüísticas, no numéricas, son usadas simulando la vía del conocimiento humano.
- Este relaciona entradas y salidas, sin tener que entender todas las variables, permitiendo que el sistema pueda ser más confiable y estable que uno con un sistema de control convencional.
- Simplifica asignación de soluciones previas a problemas sin resolver.
- Es posible obtener prototipos, rápidamente, ya que no requiere conocer todas las variables acerca del Sistema antes de empezar a trabajar.
- El desarrollo de estos es más económico que el de sistemas convencionales, porque son más fáciles de designar.
- Simplifican la adquisición y representación del conocimiento.
- Unas pocas reglas abarcan gran cantidad de complejidades.
- Por medio de ellos es cómodo designar rápidamente un prototipo que sistemas convencionales.

2.2.7.4. Conjuntos Difusos.

La mayoría de los fenómenos que encontramos cada día son imprecisos, es decir, tienen implícito un cierto grado de difusidad en la descripción de su naturaleza. Esta imprecisión puede estar asociada con su forma, posición, momento, color, textura, o incluso en la semántica que describe lo que son. En muchos casos el mismo concepto puede tener diferentes grados de imprecisión en diferentes contextos o tiempo.

Un conjunto difuso es también una función que asocia a cada objeto del universo un valor en el intervalo [0,1]. Si **x** es un objeto en el universo y **y = C(x)** es el valor asociado a **x**, se dice que **y** es el grado de pertenencia del objeto **x** al conjunto difuso **C**.

Así pues, todo conjunto en el sentido usual es también un conjunto difuso.

El conjunto vacío **{ϕ}** coincide con la función idénticamente cero y el universo coincide con la función constante 1.

Por ejemplo, tengo el conjunto:

U ={0,1,2,3,4,5,6,7,8,9}

Y los subconjuntos:

A = {0,2,4,6,8}
B = {1,3,5,7,9}

Por lo tanto los valores o elementos del conjunto A, B tendrán un valor de 1 si pertenecen al universo y un valor de 0 si no pertenecen.

2.2.7.5. Notación de Conjuntos Difusos.

Se los puede denotar de dos maneras.

1) Como pares ordenados, en donde el primer elemento es la difusidad y el segundo elemento es el valor perteneciente al conjunto de estudio.

2) Como cociente (divisor) en donde el numerador es el que da difusidad y el denominador es el elemento del conjunto de estudio.

μ (A) = {0/C , 1/M , 1/A , 0/J}

El grado de difusidad 0, indica la carencia de aproximación al límite superior, el grado de difusidad 1, quiere decir que se encuentra sobre el límite superior o lo sobrepasa.

Nota: No puede existir difusidades menores que 0 ni mayores que 1.

2.2.7.6. Mapeo de Conjuntos Difusos.

El Mapeo es un gráfico dado por el experto en la materia y en base al cual se obtendrán las difusidades de estudio. Con la notación en forma de par ordenado de los conjuntos difusos podemos representarlos en forma gráfica en un plano cartesiano, en donde el primer elemento se graficara en el eje de las **x** y el segundo elemento en el eje de las **y**.

Tomando en cuenta la lógica clásica (*ver figura. 2.2.*), los conjuntos booleanos **µi(x)** solo pueden tomar dos valores 0 o 1.

1. **Variable:** x = Edad. Universo de discurso: 0<= x <= 100 años.
2. **Valores lingüísticos (conjuntos):**
   MJ = muy_joven,
   JO = joven,
   MA = maduro,
   VI = viejo.
3. **Definición de las funciones de pertenencia:** µi(x).

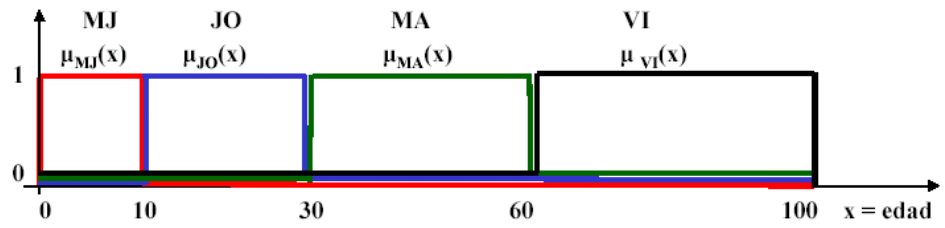


Figura 2.2: Mapeo Lógica Booleana.

µi(x) = 0 indica negación, µi(x) = 1 indica afirmación.
x = edad = 27 años : µMJ(x) = 0, µJO(x) = 1, µMA(x) = 0, µVI(x) = 0

Tomando en cuenta la lógica difusa (*ver figura. 2.3.*), los conjuntos difusos pueden tomar valores entre 0 y 1.

1. **Variable:** x = Edad. Universo de discurso: 0<= x <= 100 años.
2. **Valores lingüísticos (conjuntos):** MJ = muy_joven, JO = joven , MA = maduro, VI = viejo.
3. **Definición de las funciones de pertenencia.** µi(x).

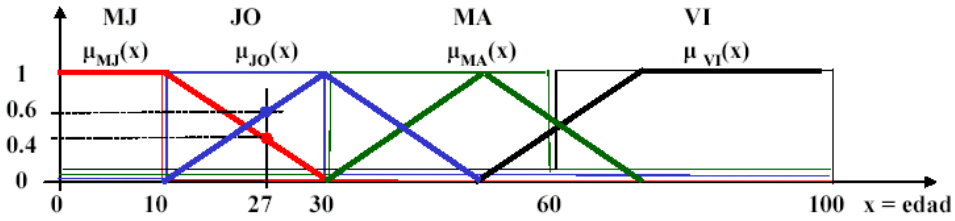


Figura 2.3: Mapeo Lógica Difusa.

x = edad = 27: µMJ(x) = 0.4, µJO(x) = 0.6, µMA(x) = 0, µVI(x) = 0

2.2.7.7. Operaciones.

Dados dos conjuntos difusos **A** y **B**, se definen las operaciones extendidas de la siguiente forma:

Producto de conjuntos difusos. Se la realiza elemento a elemento entre las partes de la operación. Los conjuntos a multiplicar tienen que tener el mismo número de elementos y referirse al mismo tipo de los mismos.

µ(A)	*	µ(B)		µ(A) * µ(B)
0,47		0,04	→	0,018
0,53		0,03	→	0,015
0,77		0,64	→	0,49
0,96		0,80	→	0,77
0,38		0,01	→	0,003
0,11		0,02	→	0,002
0,05		0,005	→	0,0002

Normalización. Se lo representa con una línea en la parte superior del conjunto a normalizar. Identificamos el mayor valor de los elementos del conjunto difuso y multiplicamos por este valor cada uno de los elementos del conjunto difuso.

μ (A)	μ (B)
0,47	0,45
0,53	0,50
0,77	0,73
0,96	0,92
0,38	0,36
0,11	0,10
0,05	0,048

Complemento del conjunto difuso. Se lo representa con un apostrofe en la parte superior derecha del conjunto a obtener el complemento. Se lo obtiene al restar 1 de cada uno de los elementos del conjunto difuso.

μ (A)	μ´ (B)
0,47	0,53
0,53	0,47
0,77	0,23
0,96	0,04
0,38	0,62
0,11	0,89
0,05	0,95

Unión de conjuntos difusos (U). Se lo obtiene tomando el mayor valor de entre las difusidades de los elementos de los conjuntos que intervienen en la operación.

μ (A)	μ (B)	μ (A) U μ (B)
0,47	0,04	0,47
0,53	0,03	0,53
0,77	0,64	0,77
0,96	0,80	0,96
0,38	0,01	0,38
0,11	0,02	0,11

Intersección (∩). Se lo obtiene tomando el menor valor de entre las difusidades de los elementos de los conjuntos que intervienen en la operación.

μ (A)	μ (B)	μ (A) ∩ μ (B)
0,47	0,04	0,04
0,53	0,03	0,03
0,77	0,64	0,64
0,96	0,80	0,80
0,38	0,01	0,01
0,11	0,02	0,02
0,05	0,005	0,005

Concentración. Es realizar una operación que permita que los valores de mayor difusidad se aproximen más hacia el valor mínimo en comparación los valores de menor difusidad. Es decir al cuadrado el valor de la difusidad.

* Encontrar la difusidad de la concentración de la honestidad.

Nombre	μ (Honestidad)	(μ (S))²
Carlos	0,32	0,1024
Milton	0,92	0,9409
José	0,8	0,64
Jorge	0,55	0,3825
Diego	0,6	0,36
Juan	0,22	0,0484

Dilatación. Es una operación que tiene un efecto contrario a la concentración y se lo consigue elevando a la potencia **(½)** la difusidad buscada.

Nombre	μ (Honestidad)	(μ (S))½
Carlos	0,32	0,5656
Milton	0,92	0,9591
José	0,8	0,8944
Jorge	0,55	0,7416
Diego	0,6	0,7745
Juan	0,22	0,4690

Intensificación Contrastante. Se utiliza dos parámetros de medida, cuando el valor de difusidad es menor que **0,5 = 2(μ (R))²** y cuando es mayor que **0,5 = (1– 2 (1 -μ(R)) ² → (1 – 2(μ´(R) ²).**

μ(R)	< 0,5 → (2(μ (R))²)	> 0,5 → (1 – 2(μ´(R)²)	Int μ(R)
0,84	-	0,9488	0,9488
0,12	0,028	-	0,028
0,03	0,0018	-	0,0018
0,15	0,045	-	0,045
0,6	-	0,68	0,68
0,43	0,369	-	0,369

Centro de Gravedad. Es una característica geométrica de cualquier objeto. El centro de gravedad es una localización media del peso de un objeto. Podemos describir totalmente el movimiento de cualquier objeto a través del espacio en los términos de la traducción del centro de gravedad del objeto a partir de un lugar a otro, y la rotación del objeto sobre su centro de gravedad si está se encontrara libre de rotar. La determinación del centro de gravedad es un proceso complicado porque la masa no se puede distribuir uniformemente, pero si el objeto tiene una línea de simetría el objeto se simplifica, para un bloque sólido de material uniforme, el centro de gravedad se encentra simplemente en la localización media de sus dimensiones físicas; para un triángulo, de altura **h**, el centro de gravedad es **h/3** y para un semicírculo de radio **r** el centro de gravedad está en **(4 * r / 3 * pi)**.

Para los objetos irregulares podemos caracterizar al centro de gravedad por un **w(x)** de la función que indique que el peso es una cierta función de la distancia **x** de una línea de referencia. Utilizamos los símbolos **"] dx de S["** para denotar la integración de una función continua, entonces el centro de gravedad se puede determinar de la siguiente manera:

$$CG = (S [x * w(x)]dx) / (S [w(x)]dx)$$

2.3. Ingeniería de Software.

La Ingeniería de Software es una disciplina o área de la informática que ofrece métodos y técnicas para desarrollar y mantener software de calidad que resuelven problemas de todo tipo a bajo costo. El proceso de Ingeniería de Software es el conjunto de actividades, métodos, prácticas y transformaciones utilizados para desarrollar software y los productos asociados: planes del proyecto, documentos de análisis/diseño, codificación, casos de prueba, manuales de usuario.

2.3.1. Definiciones.

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

Bohem, 1976. Ingeniería del Software es la aplicación practica del conocimiento científico en el diseño y construcción de programas de computadora y la documentación necesaria requerida para desarrollar, operar (funcionar) y mantenerlos.

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

Mills, 1980. La Ingeniería de Software tiene como uno de sus principales objetivos la producción de programas que cumplan las especificaciones, y que se demuestren correctos, producidos en el plazo y coste adecuados.

Meyer 1988. La Ingeniería de Software es la producción de software de calidad.

Ford, 1990. Ingeniería de Software es una forma de ingeniería que aplica los principios de la ciencia de los computadores y matemáticas para conseguir soluciones a los problemas del software de forma efectiva y económica.

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

2.3.2. Objetivos.

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

2.3.3. Software.

2.3.3.1. Definición de Software.

El software es un elemento lógico que es desarrollado con estructuras de datos que permiten a los programas manipular adecuadamente la información. El Software es la reunión de líneas de código, estructuras de datos, manual de usuario, documentación de construcción, etc.

2.3.3.2. Características del Software.

El software se desarrolla, no se fabrica. A diferencia de la construcción de hardware esta puede introducir problemas de calidad que no existen o son fácilmente corregibles en el desarrollo de software. Ambas actividades depen-

den de las personas, pero la relación entre las personas dedicadas y el trabajo realizado es completamente diferente para el software.

El software no se estropea. El software no es susceptible al entorno físico que hace que el hardware se estropee. Cada fallo en el software indica un error en el diseño o en el proceso mediante el que se tradujo el diseño a código de maquina ejecutable. Por tanto, el mantenimiento del software tiene una complejidad considerablemente mayor al mantenimiento del hardware.

La mayoría del software se construye a medida, en vez de ensamblar componentes existentes. No existen catálogos de componentes de software. Se puede comprar software ya desarrollado, pero solo como una unidad completa, no como componentes que pueden reensamblarse en nuevos programas.

2.3.3.3. Calidad de 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.

2.3.3.4. 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 forma-

to 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.

2.3.4. 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.

2.3.5. 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.

2.3.6. Ciclo de Vida.

Determina el orden de las fases del proceso software estableciendo procesos de transición de una fase a otra describiendo las actividades a realizare.

Modelo de Ciclo de Vida en Cascada. Es una visión del proceso de desarrollo como una sucesión de etapas que producen productos intermedios donde es necesario cumplir los objetivos de cada etapa para seguir con la siguiente.

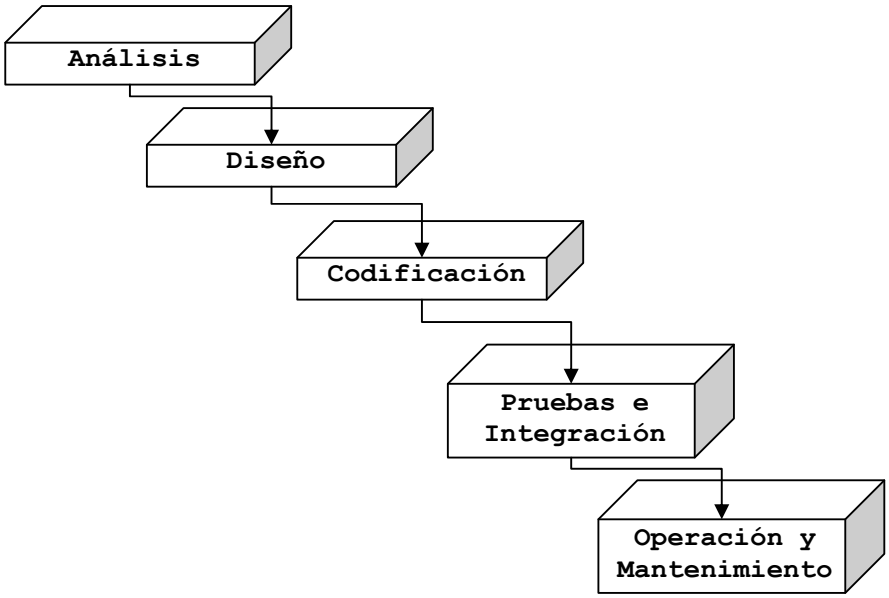


Figura 2.4: Modelo de Ciclo de Vida en Cascada.

Modelo de Ciclo de Vida en Cascada (Variación). En este modelo se añaden realimentación ente fases.

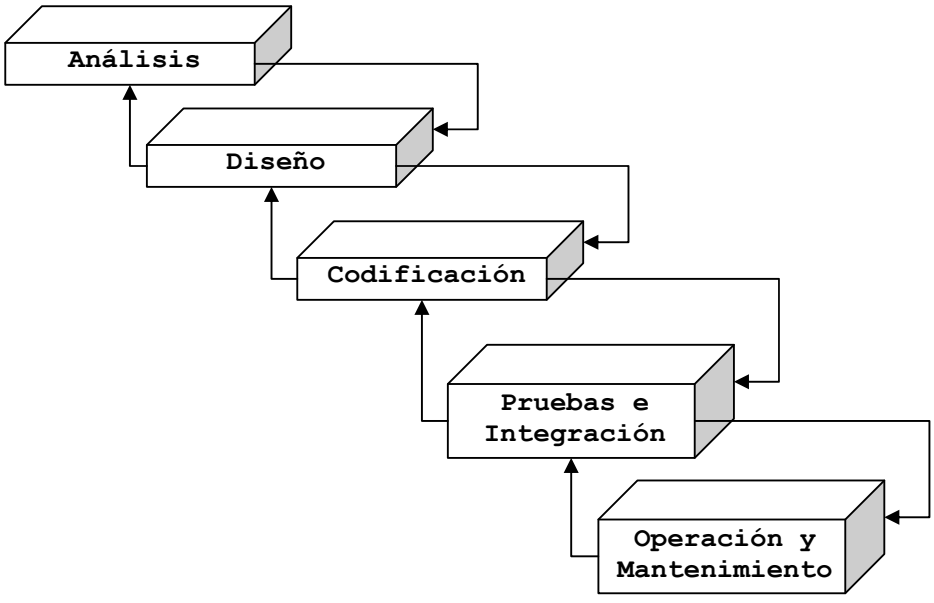


Figura 2.5: Modelo de Ciclo de Vida en Cascada (Variación).

Modelo de Ciclo de Vida en Cascada (Mejora iterativa). Los productos de las diferentes etapas se van refinando y mejorando.

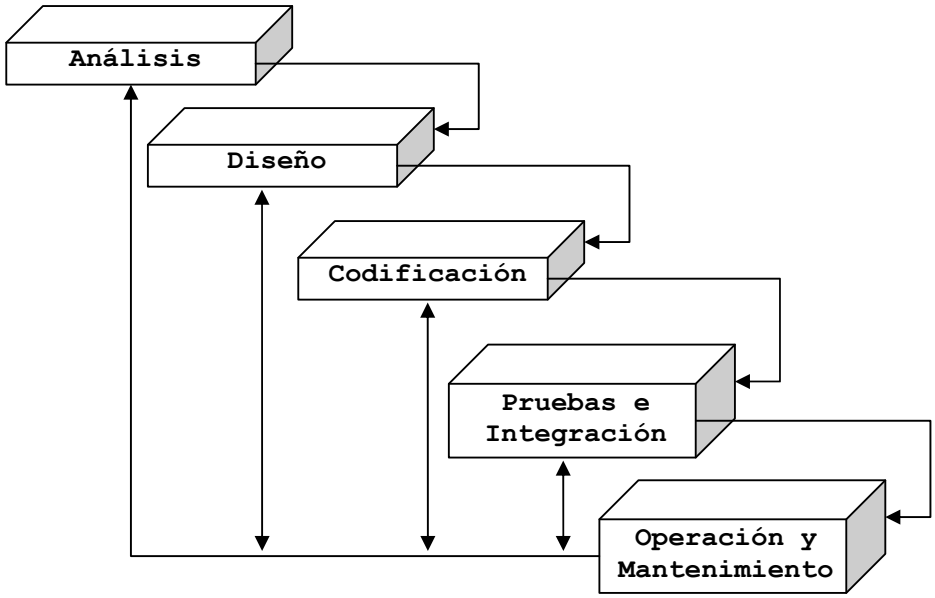


Figura 2.6: Modelo de Ciclo de Vida en Cascada (Mejora iterativa)**.**

Modelo de Ciclo de Vida Construcción de Prototipos. Es un modelo secuencial pero repetitivo. A pesar de ser rápido tiene varios inconvenientes al provocar muchas repeticiones, menor calidad y un diseño pobre.

Modelo de Ciclo de Vida Build and Fix. Se basa en el modelo de construcción de prototipos diferenciándose en que parte de un modelo ya hecho.

Modelo de Ciclo de Vida en Espiral. Representa un enfoque dirigido a prevenir riesgos para el análisis y estructuración del proceso de desarrollo de software. La ventaja es la eliminación de errores al detectar y prevenir el riesgo. La desventaja es que consume muchos recursos y sus etapas no están bien definidas.

Modelo de Ciclo de Vida Incremental. En cada etapa se obtiene un subproducto el cual es utilizado por otra.

2.4. Base de Datos.

Existen numerosas definiciones de base de datos, en resumen una base de datos es un conjunto interrelacionados y estructurados de datos organizados independientes los cuales se encuentran compartidos por diferentes usuarios.

2.4.1. Definiciones.

Martin, 1975. "Colección de datos interrelacionados almacenados en conjunto sin redundancias perjudiciales o innecesarias; su finalidad es servir a una o más aplicaciones de la mejor forma posible; los datos se almacenan de modo que resulten independientes de los programas que los usan; se emplean métodos bien determinados para incluir nuevos datos y para modificar o extraer los datos almacenados".

Deen, 1985. "Colección integrada y generalizada de datos, estructurada atendiendo a las relaciones naturales de modo que suministre todos los caminos de acceso necesarios a cada unidad de datos con objeto de poder atender todas las necesidades de los diferentes usuarios".

A. de Miguel, 1993. "Colección de datos integrados, con redundancia controlada y con una estructura que refleje las interrelaciones y restricciones existentes en el mundo real; los datos, que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de éstas, y su definición y descripción, únicas para cada tipo de datos, han de estar almacenadas junto con los mismos. Los procedimientos de actualización y recuperación, comunes y bien determinados, habrán de ser capaces de conservar la integridad, seguridad y confidencialidad del conjunto de los datos".

Date, 1995. "Una base de datos consiste en alguna colección de datos persistentes e independientes usados por una organización determinada."

2.4.2. Ventajas de las Bases de Datos.

- Integridad.
- Disponibilidad.
- Flexibilidad.
- Independencia.

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

Son sistemas que permiten la administración de los datos con el propósito de mantener la integridad de la base.

El objetivo principal de un SGBD es proporcionar un entorno que sea a la vez conveniente y eficiente para ser utilizado al extraer y almacenar información en la base de datos. Este debe permitir al administrador de la base especificar los datos que la integran, su estructura y las relaciones que existen entre ellos, las reglas de integridad semántica, los controles a efectuar antes de autorizar el acceso a la base, etc., así como las características de tipo físico y las vistas lógicas de los usuarios.

2.4.4. Modelos de Datos.

Modelo de datos es una colección de herramientas conceptuales de datos, relaciones y restricciones. Estos se dividen en tres grupos: modelos lógicos basados en objetos, modelos lógicos basados en registros y modelos físicos de datos.

2.4.4.1. Modelos Lógicos Basados en Objetos.

Se caracterizan por proporcionar capacidad de estructuración flexible permitiendo especificar restricciones de datos. Existen diferentes modelos, tales

como: el modelo entidad-relación, el orientado a objetos, el binario, el semántico de datos, y otros.

2.4.4.1.1. Modelo Entidad-Relación.

Es una percepción del mundo real, que consiste en una representación gráfica de objetos denominados entidades y la relación que existe entre ellos. Una entidad es un objeto que puede distinguirse de otro por medio de atributos. Una relación es una asociación ente varias entidades.

La estructura lógica global de una base de datos puede expresarse gráficamente por un diagrama Entidad Relación que consta de:

- Rectángulos, que representan conjuntos de entidades.
- Elipses, que representan atributos.
- Rombos, que representan relaciones entre conjunto de entidades.
- Líneas, que conectan atributos a conjuntos de entidades y conjuntos de entidades a relaciones.

Cada objeto se etiqueta con el nombre de lo que representa.

2.4.4.1.2. Modelo Orientado a Objetos.

Se basa en una colección de objetos. Un objeto contiene valores acumulados en variables de instancia dentro de él, y estos valores son objetos por si mismos. Así, los objetos contienen objetos a un nivel de anidamiento arbitrario. Un objeto también contiene partes de código que operan sobre el objeto, que se denominan métodos.

Los objetos que contienen los mismos tipos de valores y los mismos métodos se agrupan en clases. Una clase puede se vista como una definición de tipo para objetos.

La única forma en la que un objeto puede acceder a los datos de otro objeto es invocando a un método de ese otro objeto. Esto se llama envío de un mensaje al objeto. Así, la interfase de llamada de los métodos de un objeto define su parte visible externamente, la parte interna del objeto no son visibles externamente. El resultado es dos niveles de abstracción de datos.

2.4.4.2. Modelos Lógicos Basados en Registros.

Se utilizan para describir datos en los modelos conceptual y físico. A diferencia de los modelos lógicos basados en objetos, se usan para especificar la estructura lógica global de la base de datos y para proporcionar una descripción a nivel más alto de la implementación.

Cada tipo de registro define un número fijo de campos, o atributos, y cada campo normalmente es de longitud fija. La estructura más rica de estas bases de datos a menudo lleva a registros de longitud variable en le nivel físico.

Se divide en: modelos relacional, de red y jerárquico.

2.4.4.2.1. Modelo Relacional.

El modelo relacional representa los datos y relaciones entre los datos mediante una colección de tablas, cuyas columnas tienen nombres únicos.

2.4.4.2.2. Modelo de Red.

Los datos en el modelo de red se representan mediante colecciones de registros y las relaciones entre los datos se representan mediante enlaces, los cuales pueden verse como punteros.

2.4.4.2.3. Modelo Jerárquico.

El modelo jerárquico es similar al modelo de red, los datos y las relaciones se representan mediante registros y enlaces. Se diferencia del modelo de red en que los registros están organizados como colecciones de árboles.

2.4.4.3. Modelos Físico de Datos.

Los modelos físicos de datos se usan para describir datos en el nivel más bajo. Existen muy pocos modelos físicos de datos en uso, siendo los más conocidos el modelo unificador y de memoria de elementos.

2.5. Motor de Base de Datos MySQL.

2.5.1. Introducción.

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

Una vez instalado, para entrar en la consola de MySQL se requieren ciertos parámetros. Hay que tener en cuenta que el servidor es multiusuario, y que cada usuario puede tener distintos privilegios, tanto de acceso a tablas como de comandos que puede utilizar. La forma general de iniciar una sesión MySQL es:

```
mysql -h host -u usuario -p
```

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

Para salir de una sesión del cliente de MySQL se usa el comando “quit”.

```
mysql> quit
Bye
C:\mysql\bin>
```

2.5.2. Creación de Base de Datos.

Cada conjunto de relaciones que componen un modelo completo forma una base de datos. Desde el punto de vista de SQL, una base de datos es sólo

un conjunto de relaciones y tablas, y para organizarlas o distinguirlas se accede a ellas mediante su nombre. A nivel de sistema operativo, cada base de datos se guarda en un directorio diferente.

Para crear una base de datos se usa la sentencia “CREATE DATABASE”:

```
mysql> CREATE DATABASE evaluacion;
Query OK, 1 row affected (0.03 sec)
```

Para conocer la descripción de las bases de datos existentes se utiliza la sentencia “SHOW DATABASES”

```
mysql> SHOW DATABASES;
+------------+
| Database   |
+------------+
| evaluacion |
| mysql      |
| test       |
+------------+
3 rows in set (0.00 sec)
```

Para seleccionar una base de datos se usa el comando “USE”

```
mysql> USE evaluacion;
Database changed
```

2.5.3. Creación de Tablas.

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

```
mysql> CREATE TABLE gente (nombre VARCHAR (40), fecha
DATE);
Query OK, 0 rows affected (0.53 sec)
```

Para consultar las tablas existentes se usa la sentencia “SHOW TABLES”.

Existen varias opciones a la hora de definir columnas. Además del tipo y el nombre, se pueden definir valores por defecto, permitir o no que contengan valores nulos, crear una clave primaria, indexar. La sintaxis para definir columnas es:

```
nombre_col tipo [NOT NULL | NULL] [DEFAULT valor_por_defecto]
     [AUTO_INCREMENT] [[PRIMARY] KEY] [COMMENT 'string']
     [definición_referencia]
```

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 autoincrementada 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. Adicionalmente, 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.

2.5.4. Ingreso de Datos.

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

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

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

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

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

2.5.5. Actualización de Datos.

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

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

Es posible actualizar el valor de más de una columna, separándolas en la sección "SET" mediante comas.

Mediante la cláusula "WHERE" es posible establecer una condición. Solo las filas que cumplan con la condición serán actualizadas.

```
mysql> UPDATE ciudad SET poblacion = 700000
    -> WHERE nombre = 'Roma';
Query OK, 1 row affected (0.05 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

2.5.6. Eliminación de Datos.

Para eliminar filas se usa la sentencia "DELETE".

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

La forma más simple es no usar ninguna de las cláusulas opcionales. Pero es más frecuente que sólo se necesite eliminar ciertas filas que cumplan determinadas condiciones, para ello se hace uso de la cláusula “WHERE”:

```
mysql> DELETE FROM ciudad WHERE clave=2;
Query OK, 1 row affected (0.05 sec)
```

Cuando queremos eliminar todos los datos de una tabla existe una sentencia alternativa mediante la sentencia “TRUNCATE”.

2.5.7. Selección de Datos.

Para extraer datos de la base es necesario usar la sentencia “SELECT”.

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

La forma más sencilla consiste en no especificar condiciones.

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

Existe la posibilidad de realizar consultas en función de ciertas condiciones, para ello es necesario usar la cláusula “WHERE”.

```
mysql> SELECT * FROM gente WHERE nombre = "Torres";
+---------+------------+
| nombre  | fecha      |
+---------+------------+
| Torres  | 1978-06-15 |
+---------+------------+
1 row in set (0.03 sec)
```

2.6. Programación Web.

2.6.1. Funcionamiento de un Sitio Web.

La arquitectura cliente-servidor hace referencia a múltiples clientes que se conectan a un servidor en forma concurrente. El servidor depende del sitio Web mientras el cliente suele ser un navegador, por ejemplo Microsoft Internet Explorer o Netscape Navigator.

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

2.6.2. El protocolo HTTP.

Generalmente el protocolo HTTP se instancia del lado del cliente en el momento que genera un ***request*** el cual es contestado por el servidor con el envió de un ***response*** indicando si el pedido de un recurso determinado pudo ser resuelto por el servidor.

Los datos que el cliente recibe del servidor dependen del ***content-type*** declarado, por lo general los tipos más utilizados son texto plano, código html o imágenes.

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.

2.6.3. Ciencias aplicadas para el desarrollo de aplicaciones Web.

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

Programación en el cliente:

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

Programación en el servidor:

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

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

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

La programación del lado del cliente tiene como principal ventaja que la ejecución de la aplicación se encarga al cliente, con lo cual se evita recargar al servidor de trabajo. El servidor sólo envía el código, y es tarea del browser in-

terpretarlo. La gran desventaja de esta metodología es que el código que el server envía es sensible a que cosas puede o no hacer el browser. El usuario puede, por ejemplo, decidir deshabilitar una funcionalidad del browser que es necesaria para que se ejecute un determinado servicio o peor aún, browsers distintos pueden interpretar el mismo código de distintas formas. Típicamente Netscape y Microsoft, que producen los dos browser más usados del mercado, no se ponen de acuerdo sobre como se implementan diversas tecnologías en el cliente.

Programar del lado del servidor tiene como gran ventaja que cualquier cosa puede hacerse sin tener en cuenta el tipo de cliente, ya que la aplicación se ejecuta en el servidor que es un ambiente controlado. Una vez ejecutada la aplicación, el resultado que se envía al cliente puede estar en un formato normalizado que cualquier cliente puede mostrar. La desventaja reside en que el server se sobrecarga de trabajo ya que además de servir páginas es responsable de ejecutar aplicaciones. A menudo esto redunda en requisitos de hardware mayores a medida que el server ejecuta más y más servicios.

A continuación se describe un cuadro que muestra algunos lenguajes que tanto el lado del cliente como el servidor interpretan:

Programación en el Cliente	**Programación en el Servidor**
HTML	CGI (Cualquier Lenguaje)
CSS	ASP
DHTML	PHP
JavaScript	mod_perl
Java	
VBScript	

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.

2.6.4. Programación del lado del Servidor.

Para el desarrollo de aplicaciones del lado del servidor existen 3 grandes metodologías, utilizar el ***protocolo CGI***, utilizar una ***API provista por el web-server*** o bien utilizar un ***módulo del servidor Web***.

2.6.4.1. El protocolo CGI.

El protocolo CGI (Common Gateway Interface) fue creado para establecer un protocolo standard de comunicación entre el web-server y cualquier lenguaje de programación de forma tal que desde el lenguaje "x" puedan recibirse datos que el usuario envía usando el método post o get y además el resultado de la aplicación sea derivado por el web-server al browser. Típicamente para recibir datos se usa alguna biblioteca o módulo del lenguaje elegido que implementa el protocolo CGI y para enviar datos simplemente se envían al standard-output desde el lenguaje elegido y el web-server se encarga de redireccionar esto al browser. Para ejecutar una aplicación CGI el web-server en general procede de la siguiente manera:

- Se toma el "***request*** del ***browser***" y los datos que se envían al ***server*** por método ***post*** o ***get*** se pasan a variables de ambiente.
- El ***server*** redirecciona su salida standard al browser.
- El ***server*** crea un proceso que tiene la salida standard redireccionada.
- El ***server*** 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 CGI 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.

Los lenguajes no interpretados (C, C++) tienen como ventaja que requieren menos recursos del server al generarse el proceso CGI (no hace falta un interprete) y además suelen ser mucho más veloces en su ejecución (no se necesita interpretar nada), sin embargo el desarrollar y depuración suelen ser tareas muy complejas y no siempre se justifica el esfuerzo si la aplicación es pequeña. En los comienzos de la Web la gran mayoría de las aplicaciones se encontraban en la categoría chica / muy chica por lo que la eficiencia no era un factor importante y por eso los lenguajes compilados no se utilizaron demasiado.

La desventaja de las aplicaciones CGI consiste en que el server debe realizar un ***fork***, y ejecutar la aplicación o bien el intérprete de la aplicación, y este ciclo que se cumple cada vez que se ejecuta la aplicación CGI insume muchos recursos y en general es costoso en tiempo para el server. Durante muchos años este esquema no muy eficiente dominó ampliamente el mundo de las aplicaciones Web.

2.6.4.2. Uso de una API del Servidor.

Otra técnica factible consiste en utilizar una API (application programming interface) provista por el servidor Web para desarrollar aplicaciones, es decir que el web-server provee un lenguaje en el cual se pueden desarrollar aplicaciones. 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 po-

dían 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.

2.6.4.3. Uso de un módulo del Web-Server.

La tecnología más reciente para la ejecución de aplicaciones consiste en anexar a un web-server módulos que permiten al web-server interpretar un determinado lenguaje. De esta forma se logra eficiencia ya que el server no necesita crear un nuevo proceso por cada aplicación que ejecuta. Las aplicaciones son portables ya que son desarrolladas en un lenguaje standard que no depende del web-server, las aplicaciones son confiables ya que si bien pueden producir un error en el lenguaje en que están diseñadas si el módulo es sólido dichos errores no pueden comprometer al web-server. Al combinar las ventajas más importantes del ambiente CGI y un desarrollo basado en APIs evitando los inconvenientes de los mismos este esquema suele ser el más adecuado para ejecución de aplicaciones.

2.6.5. Evolución de los Web-Sites.

Desde la aparición de la Web, distintos modelos o prototipos de sitios Web fueron siendo predominantes en distintos periodos de tiempo, de acuerdo a las características comunes de los sitios predominantes en cada momento podemos diferenciar 3 generaciones de sitios Web distintos:

2.6.5.1. Web-Sites de primera generación.

En un sitio Web de primera generación las páginas se desarrollaban, se subían al servidor y el servidor se encargaba de enviar las páginas al browser, es un modelo basado en páginas estáticas, en donde predominaba el uso de texto, links a otros sitios o a otras páginas del mismo sitio. Frecuentemente se

usaban ***rules*** para separar contenidos y las páginas de gran extensión vertical con gran cantidad de texto, listas y links eran comunes. Si bien no eran visualmente atractivos estos sitios estaban enfocados a funcionar en forma veloz y entregar al usuario gran cantidad de información interrelacionada.

2.6.5.2. Web-Sites de segunda generación.

La segunda generación implicó una revolución en lo visual, a medida que los sitios web se volvían emprendimientos más comerciales que científicos el hecho de "capturar" usuarios se torno una premisa y por ello se le dio gran importancia al aspecto visual. Las páginas con "layouts" visualmente atractivos fueron más y más populares, el uso abundante de imágenes, imágenes animadas y elementos multimedia se volvió común, aparecieron páginas que controlaban los fonts, el estilo y la posición en la que los mismos se ubicaban. La gran mayoría de los sitios usaban tablas HTML para controlar la posición exacta en que los elementos aparecerían en el browser y tags nuevos en html como "font" u otros para controla la forma en la cual los mismos se verían. El concepto fue dominar la presentación de la información. Otra característica importante de esta segunda generación de Web-sites fue la aparición explosiva de más y más aplicaciones a medida. Los servicios que ofrecía un sitio se volvían factores importantes en la atracción de usuarios, chats, foros de discusión, banners, contadores y muchas aplicaciones más empezaron a aparecer y las falencias del protocolo CGI, a medida que las aplicaciones eran más grandes y la cantidad de usuarios crecía exponencialmente, comenzaron a hacerse notar.

2.6.5.3. Web-Sites de tercera generación.

La tercera generación siguió basada en lo visual, el gran cambio vino en la forma cómo se generaba la información. Las páginas estáticas que dominaban el 100% de los sitios de primera y segunda generación fueron reemplazadas por páginas dinámicas que el servidor Web generaba en el momento, a partir de información que en general se guardan en una base de datos. Estos sitios

"dinámicos" permiten actualizar la información e incluso cambiar completamente la forma en que se muestran dichos datos en velocidades asombrosas. Los sitios de tercera generación facilitaron las aplicaciones interactivas, la información en tiempo real y que día a día se ofrecieran nuevos servicios. Las aplicaciones empezaron a desarrollarse también usando otras tecnologías dejando de lado el protocolo CGI. Aplicaciones en ASP, mod_perl o PHP, mucho más poderosas y eficientes que sus padres CGI, son el standard de este tipo de sitios.

2.6.5.4. Lenguajes de Programación Web.

2.6.5.4.1. HTML

(Hypertext Markup Language). Es el lenguaje que permite codificar o preparar documentos de hipertexto, este viene a ser el lenguaje común para la construcción de una página 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 C++, Visual Basic, 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 visualizara en la forma como éste lo entienda.

2.6.5.4.2. Javascript.

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

Con Javascript se puede crear efectos especiales en las páginas y definir interactividades con el usuario. El navegador del cliente es el encargado de interpretar las instrucciones Javascript y ejecutarlas para realizar estos efectos e

interactividades, de modo que el mayor recurso, y tal vez el único, con que cuenta este lenguaje es el propio navegador.

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

2.6.5.4.3. VB Script.

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

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

2.6.5.4.4. DHTML.

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

Cualquier página que responde a las actividades del usuario y realiza efectos y funcionalidades se puede englobar dentro del DHTML, tal como efectos en el navegador que pueden mostrar y ocultar elementos de la página, se puede modificar su posición, dimensiones, color, etc.

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

2.6.5.4.5. CSS.

(Cascading Style Sheets). Es una tecnología que permite crear páginas web de una manera más exacta. Gracias a las CSS se pueden hacer muchas cosas que no se podía hacer utilizando solamente HTML, como incluir márgenes, tipos de letra, fondos, colores, etc.

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

2.6.5.4.6. Applets de Java.

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

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

Como desventajas en relación con Javascript cabe señalar que los applets son más lentos de procesar y que tienen espacio muy delimitado en la página donde se ejecutan.

2.6.5.4.7. PERL.

(Practical Extracting and Reporting Language). Es un lenguaje de programación muy utilizado para construir aplicaciones CGI para la web.

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

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

2.6.5.4.8. ASP.

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

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

2.6.5.4.9. PHP.

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

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

Se detallará de mejor manera este lenguaje en el apartado *2.7. Lenguaje de Programación PHP.*

2.6.5.4.10. JSP.

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

El motor de las páginas JSP está basado en los servlets de Java (programas en Java destinados a ejecutarse en el servidor).

2.6.5.4.11. XML.

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

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

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

2.7. Lenguaje de programación PHP.

2.7.1. Generalidades.

2.7.1.1. Introducción.

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

2.7.1.2. Generalidades.

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

```
<HTML>
<HEAD><TITLE>Hola</TITLE></HEAD>
<BODY>
Hola esta es una página de prueba. <BR />
<?php
print("Hola soy una línea generada en php <BR />");
?>
</BODY>
</HTML>
```

Este ejemplo al guardarse en un archivo de extensión .php es automáticamente parseado por el interprete de php cuando el browser envía un pedido. El ciclo es el siguiente:

- El ***browser*** envía un pedido de un archivo con extensión ***php***.
- El ***server*** analiza que la extensión del ***request*** es ***.php***, obtiene el archivo y lo envía al interprete ***php***.
- El interprete ***php*** del ***web-server*** parsea el archivo en busca de ***tags <? ?>*** y procesa todo lo que se encuentre 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 recibiría:

```
<HTML>
<HEAD><TITLE>Hola</TITLE></HEAD>
<BODY>
Hola esta es una prueba. <BR />
Hola soy una línea generada en php <BR />
</BODY>
</HTML>
```

2.7.1.3. Conceptos Básicos. Generación de Sitios Dinámicos con PHP.

Cuando el código html es dinámico se puede usar una mezcla de print y tags que abren y cierran que suele ser lo más eficiente, por ejemplo:

```
<form name="<? print ("$nombre_form")?>">
```

Una característica interesante de php es que permite realizar "includes" dentro de un script php, de esta forma se puede modularizar una página o el layout de una página en varios módulos php que se desarrollan en forma independiente, además pueden desarrollarse en php componentes reusables que luego se utilizan en otras páginas o incluso en otros sitios.

Una forma común de trabajo usando php para generar sitios dinámicos es definir templates o layouts en los cuales se divide la página en zonas o módulos que serán desarrollados en php, el layout de la página con la forma y tamaño de cada zona se puede definir sin problemas usando por ejemplo tablas de html.

2.7.1.4. Funcionalidad.

- Funciones de calendario y manipulación de calendarios usando MCAL. (Modular Calendar Access Library).
- Programación orientada a objetos.
- Funciones para creación de archivos PDF.
- Parser de documentos XML. (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 para manejo de directorios.
- Funciones de encriptación de datos.
- Funciones de acceso al filesystem.
- Funciones para manejo de FTP.
- 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.

2.7.2. Introducción al Lenguaje.

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 ";".

2.7.2.1. 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 las dos últimas variantes el comentario empieza en donde se encuentra el "//" o el "#" y termina cuando termina la línea.

2.7.2.2. Tipos de Datos.

PHP soporta los siguientes tipos de datos:

- Enteros
- Vectores
- Binarios de punto flotante
- Strings
- Objetos

En general 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 los raros casos en que esto sea necesario. Todas las variables en php se denotan utilizando el signo '**$**' precediendo al nombre de la variable.

2.7.2.3. Constantes.

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

2.7.2.4. Operadores.

Los operadores aritméticos en PHP también se asemejan al C:

```
$a + $b;  -> suma
$a - $b;  -> resta
$a++;     -> pos-incremento
++$a;     -> pre-incremento
$a--;     -> pos-decremento
--$a;     -> pre-decremento
$a * $b;  -> multiplicación
$a / $b;  -> división
$a % $b;  -> módulo
```

2.7.2.5. 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"
```

2.7.2.6. Comparaciones.

```
$a == $b;  -> true si $a igual a $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
```

2.7.2.7. Operador @.

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

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

2.7.2.8. Operadores Lógicos.

```
$a && $b; -> true si $a es verdadero y $b es verdadero
$a || $b; -> true si $a es verdadero o $b es verdadero
$a xor $b;-> or exclusivo
!$a;      -> true si $a es falso (NOT)
```

2.7.2.9. 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 ($i=0;$i<6;$i++) {
  if ($i == $b) break; -> Sale del ciclo si $i es igual a $b
}
```

Switch:

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

2.7.2.10.Funciones.

Definir subrutinas en php es sencillo:

```
function prueba ($a, $b) {
  $r = $a + $b;
  return $r;
}
```

Para invocar a la función basta con digitar:

```
$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; }
```

2.7.3. 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 PHP está indebido para generar en forma sencilla páginas Web dinámicas a partir de información almacenada en bases de datos.

2.7.3.1. 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.

2.7.3.2. 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.

2.7.3.3. Queries a la Base de Datos.

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

Nuevamente el link que se debe usar es el que se obtiene al conectarse a la base, **`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.

2.7.3.4. Cantidad de Filas Consultadas o Modificadas.

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

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

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

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

2.7.3.5. 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.

2.7.4. Programación Orientada a Objetos en PHP.

2.7.4.1. Introducción

PHP es un lenguaje hibrido donde se puede utilizar OOP o procedimientos de programación procedimental tradicional. Para largos proyectos se pude querer o necesitar usar OOP puro en PHP declarando clases y utilizando sólo objetos y clases para un proyecto. Mientras más largo es el proyecto más puede ayudar la programación orientada a objetos, ya que código es más fácil de mantener, entender y reutilizar.

2.7.4.2. Conceptos de Programación Orientada a Objetos.

2.7.4.2.1. Encapsulación.

Las propiedades de los objetos son definidas en PHP utilizando la declaración “var” dentro de la clase. Cuando se declara una propiedad la misma no tiene tipo alguno asignado, hasta que nosotros le asignemos algún valor en particular. Una propiedad puede ser un entero, un vector, un vector asociativo, e inclusive puede ser otro objeto.

Los métodos son definidos como funciones, también dentro de la clase. Y para acceder a las propiedades de la instancia de esa clase es necesario referirse a las propiedades como $this->name. En caso de no utilizar el “$this->” la variable será local al método y una vez terminada la ejecución del mismo se perderá su valor.

```
class ClaseX {
  var $x;
  function setX($v) {
   $this-> x = $v;
  }
  function getX() {
   return $this-> x;
  }
}
```

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

```
$obj = new ClaseX;
```

2.7.4.2.2. Herencia.

Los objetos de una clase hijo poseen todas las propiedades y métodos de su clase padre, más las propiedades y métodos propios.

```
class SubClaseX extends ClaseX {
  var $y;
  function setY ($v) {
    $this-> y = $v;
  }
  function getY () {
    return $this-> y;
  }
}
```

En las clases se pueden definir constructores. Los constructores son métodos que se ejecutan al momento de instanciar la clase. La característica para que un método sea el constructor de una clase es que debe tener el mismo nombre que la clase.

```
class ClaseX {
  var $x;
  function ClaseX ($y) {
    $this-> x = $y;
  }
}
```

2.7.4.2.3. Polimorfismo.

Polimorfismo se define como la habilidad de un objeto de determinar que método debe invocar para un objeto pasado como argumento en tiempo de ejecución.

2.7.5. Persistencia.

Uno de los problemas clásicos en el desarrollo de sitios Web y aplicaciones Web 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.

2.7.5.1. Sesiones.

Se suele definir como una sesión 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*** de ahora en adelante, las posibilidades se detallan a continuación:

2.7.5.2. Cookies.

Uno de los mecanismos más usados para mantener persistencia es el mecanismo de cookies. El concepto es que mediante un header del protocolo 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. Para crear un cookie en PHP se utiliza la función **setcookie**:

```
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 cookie
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.

2.7.5.3. 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 la 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 links y afea un poco la forma en la cual se muestra la URL de la página actual.

2.7.5.4. 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 PHPSESSID 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 a la función **`session_register();`**.

```
session_start();
$x = "hola";
session_register("x");
```

Notar que **`session_register`** recibe el “nombre” de la variable a registrar sin “$”. Una vez registrada una variable la misma estará disponible en todas las páginas que usen **`session_star`**t durante la sesión actual, es decir que si desde la página donde hicimos el **`session_register`** el usuario se traslada a otra página que usa **`session_start`** entonces la variable **`$x`** 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)

2.7.6. 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.

2.7.6.1. 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: url");
```

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 errores 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. CAPITULO III. Desarrollo del Proyecto.

3.1. Análisis del Sistema.

El análisis del sistema engloba a todos los elementos que van a constituir el Sistema de Evaluación Docente (SED), identificando las necesidades de la Universidad; valorando la viabilidad del sistema; asignando funciones a hardware, software, personal; realizando una análisis técnico y estructurado y, estableciendo las especificaciones y restricciones del sistema.

3.1.1. Identificación de las necesidades.

La carencia de un sistema de evaluación docente automatizado en la PUCESA, nos permite distinguir las siguientes necesidades.

- Un Sistema automatizado de Evaluación Docente.
- Un Sistema automatizado de Auto Evaluación Docente.
- Una evaluación sistematizada de los docentes de la Universidad.
- Resultados comparativos entre las evaluaciones resultantes.
- Una interfase de fácil utilización.
- Un sistema estructurado de alto nivel que brinde seguridad en los resultados de evaluación y que lo realice en tiempo real.
- Un sistema que brinde las facilidades de instalación en el Servidor SUN (multiplataforma) de la Universidad y de la misma forma en servidores que se usaren con posteridad.
- Una base de datos relacional que permita almacenar los diferentes datos y parámetros de evaluación en forma segura sin ocupar un excesivo espacio en disco y que se adapte a los requerimientos del Servidor SUN e250 donde va a estar hospedado el sistema.
- Una interfase Web de fácil utilización para los diferentes niveles de usuarios.

3.1.2. Estudio de Viabilidad.

Se analizará los resultados básicamente en cuatro áreas:

a) **Viabilidad Económica.** Los costos del desarrollo e implementación del sistema son accesibles en lo que se refiere a hardware y software a utilizarse en comparación al alto beneficio final.
b) **Viabilidad Técnica.** Existe el espacio en disco necesario para el almacenamiento de datos, páginas Web y demás componentes que utilizará el sistema hospedado en el servidor SUN e250 de la Universidad. Se posee el software y hardware básicos para la codificación del sistema así como el conocimiento suficiente para su desarrollo.
c) **Viabilidad Legal.** Se ha determinado que no existe ninguna infracción que pueda ocurrir a causa del sistema.
d) **Estudio de Alternativas.** En caso de que el servidor de la Universidad no esté a disposición por cualquier circunstancia, el sistema se podría implementar de forma local en algún computador que cumpla con las necesidades de mínimas del sistema (un servidor de base de datos MySQL, compatibilidad con PHP).

Basados en el estudio de Viabilidad del sistema en la Universidad se ha determinado que es factible el desarrollo del proyecto.

3.1.3. Análisis Técnico.

Abarca los conceptos del sistema recogiendo información de fiabilidad, facilidad de mantenimiento, posibilidad de producción y rendimiento.

a) Análisis de Requerimientos Servidor SUN e250.

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

b) Riesgos del Desarrollo. Se puede cumplir con todas las funciones y requerimientos que necesita la Universidad.

c) Disponibilidad de Recursos. Se posee los equipos de cómputo y aplicaciones de desarrollo necesarios para la realización del sistema.

d) Tecnología. La tecnología actual permite el desarrollo eficiente de la aplicación, motivo por el cual, el desarrollo del sistema no se encuentra limitado por la tecnología.

3.1.4. Análisis de Requisitos.

Tras el análisis de las necesidades de la Universidad se llego a las siguientes conclusiones:

- Es posible el desarrollo de un sistema automatizado de evaluación y auto evaluación docente, por medio de la utilizando de métodos de Lógica Difusa con el propósito de llegar a una evaluación cualitativa.
- El ***front end*** del sistema será desarrollado en lenguaje PHP. Este lenguaje multiplataforma está orientado a aplicaciones Web. El sistema requiere soporte de programación de alto nivel orientada a objetos y procedimental, y que el lenguaje brinde las facilidades de conexión con base de datos.
- El ***back end*** del sistema será desarrollado en MySQL, al igual que PHP también esta hospedada en el servidor de la Universidad. MySQL es un potente sistema de gestión de base de datos que con la ulitización del motor de almacenamiento InnoDB, brindará la transaccionalidad de datos que el sistema requiere.

3.1.4.1. Especificaciones.

Con el propósito de que el Sistema de Evaluación Docente se desarrolle con mayor calidad, se detalla las siguientes especificaciones:

Especificaciones del Sistema de Evaluación Docente	
Lenguajes de programación.	PHP, HTML, JavaScript
Base de Datos.	MySQL
Entorno en el que va a operar.	Sistemas Operativos y Browser
Funciones.	• Validación y control de los diferentes tipos de usuarios: o Administrador o Escuelas o Docentes o Estudiantes • Mantenimiento de la base de datos; el sistema debe permitir la: o creación, o modificación y o eliminación de los diferentes datos del sistema. • Administración de Docentes y Estudiantes. • Asignación de materias a los docentes. • Activar y desactivar el sistema de Evaluación Docente. • Mantenimiento de los cuestionarios de Evaluación Docente. • Realizar el proceso de evaluación y auto evaluación docente por período académico. • Reportes e informes del proceso de evaluación. • Brindar observaciones y recomendaciones a los docente evaluados.
Restricciones.	• El sistema será utilizado exclusivamente para el proceso de evaluación docente. • No será un sistema académico, lo cual implica que no deberá almacenar ninguna información innecesaria para el proceso de evaluación. • El sistema no realizara directamente funciones de respaldos de procesamiento de datos ni operaciones de backup y recuperación automatizados. Todos estos procesos quedan a cargo de la administración del sitio Web donde estará hospedado el sistema, previa capacitación para su utilización. • El sistema necesita aproximadamente de 10 Mb de espacio en disco para el alojamiento de las páginas Web y, 10 Mb para el almacenamiento de la base de datos.

3.1.5. Análisis Estructurado.

3.1.5.1. Diagrama de Flujo de Datos Nivel 0.

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

Diagrama de Flujo de Datos Nivel 0

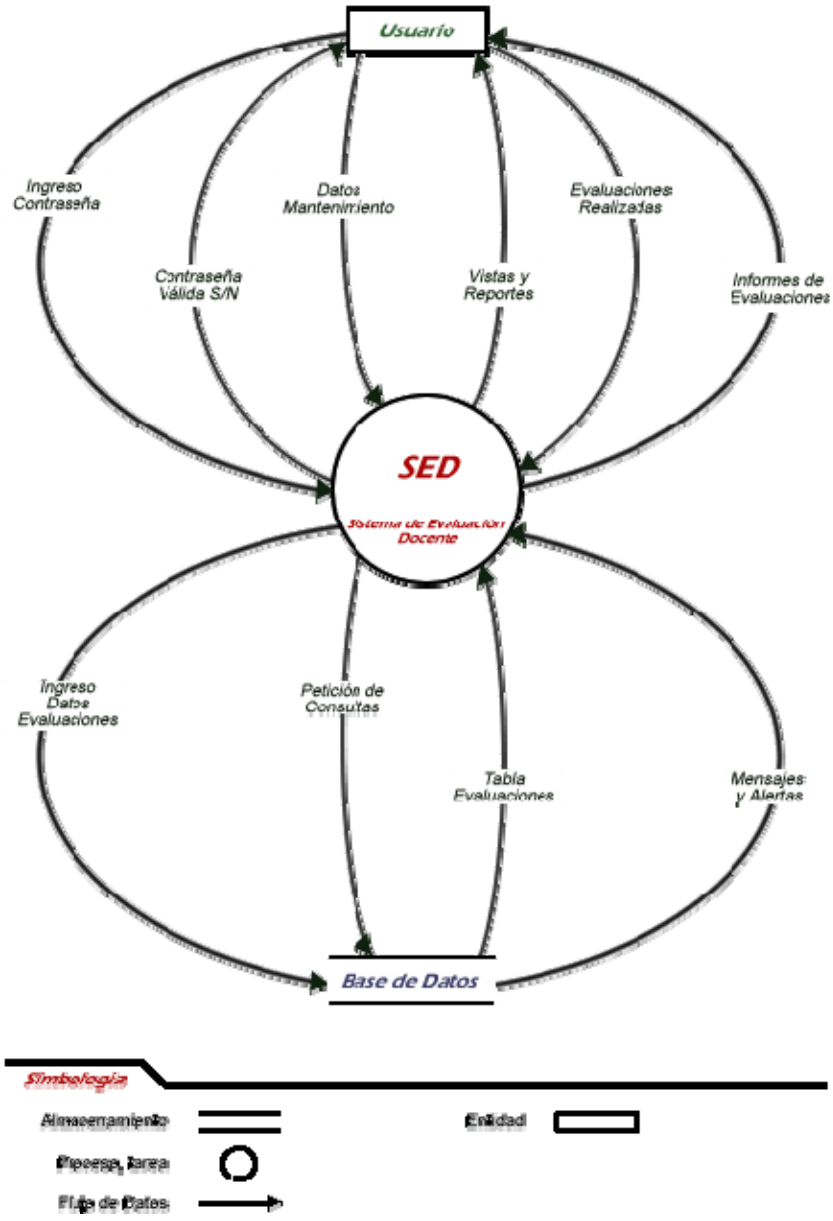


Figura 3.1: Diagrama de Flujo de Datos Nivel 0.

3.1.5.2. 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.

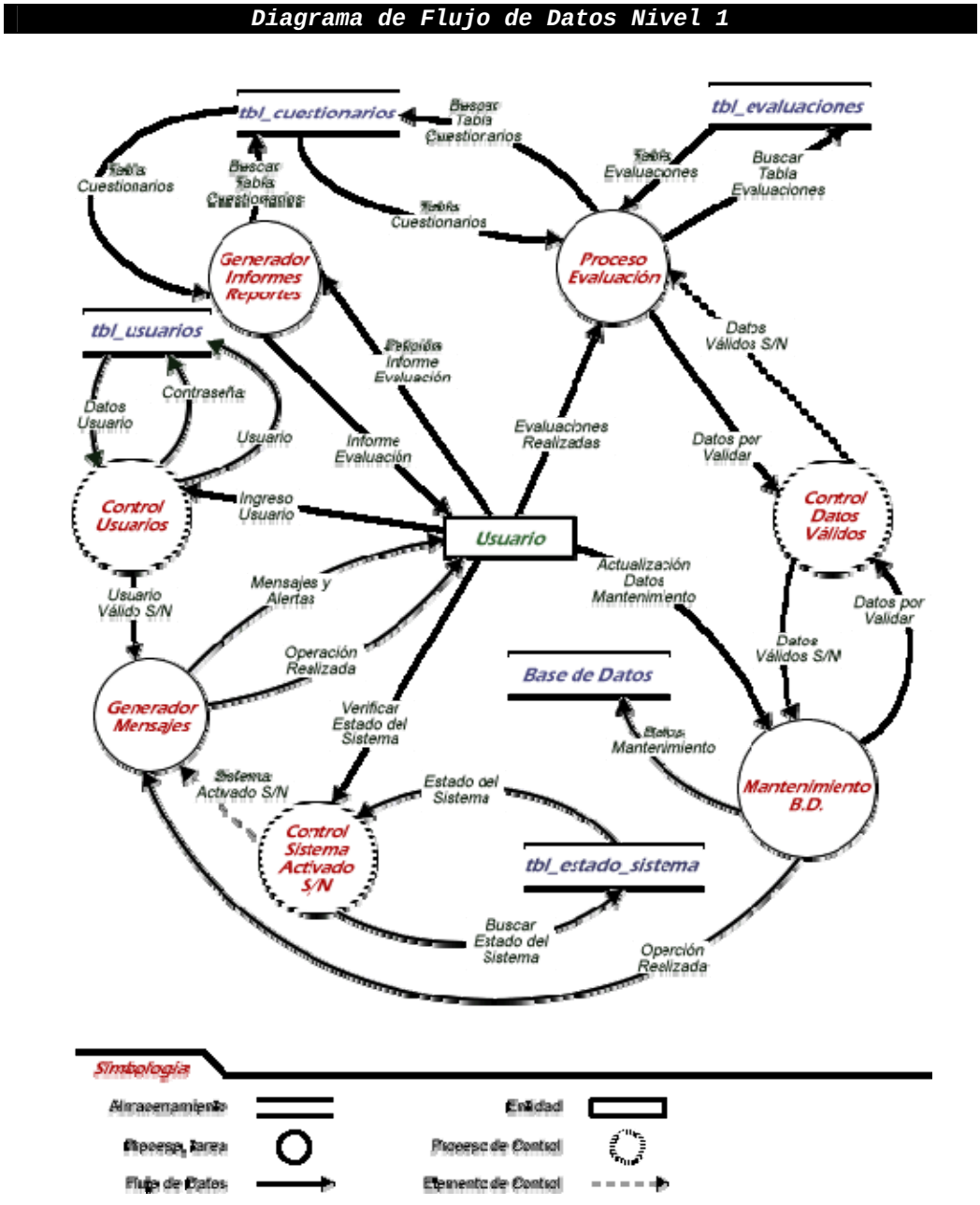


Figura 3.2: Diagrama de Flujo de Datos Nivel 1.

3.1.5.3. Diagrama de Flujo de Datos Nivel 2.

Para el diagrama de flujo de datos nivel 2 se encuentra especificado el ***proceso de evaluación*** del SED.

Diagrama de Flujo de Datos Nivel 2

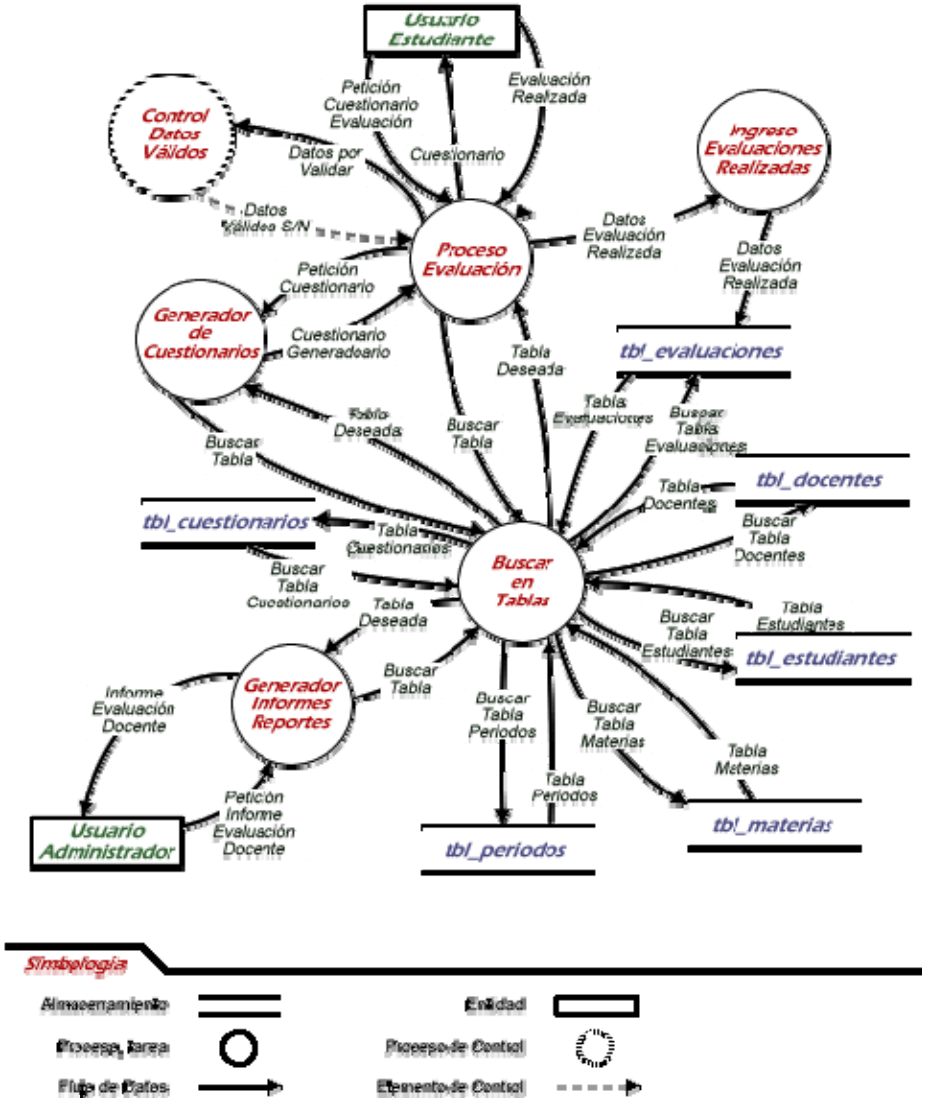


Figura 3.3: Diagrama de Flujo de Datos Nivel 2.

3.2. Diseño del Sistema.

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

Se desarrolla un diseño detallado tomando en cuenta refinamientos de la representación arquitectónica con una estructura de datos pormenorizada.

La elaboración del diseño y calidad del software toma en consideración los siguientes puntos:

- El diseño exige una ***organización jerárquica*** y un uso inteligente de control.
- El diseño debe ser ***modular*** y estos módulos han de ser divididos en forma lógica.
- El diseño debe tener ***representaciones distintas y separadas de datos y procedimientos***.
- Un diseño debe llevar a módulos que exhiban características ***funcionales independientes***.
- Un diseño debe llevar a interfases que reduzcan la complejidad de las conexiones. ***Front end con menor módulos posibles***.
- El diseño debe poder ser reproducido y debe ser conducido por los resultados del análisis de requisitos. ***Poder hacer y posible codificación***.

El diseño del sistema SED modela en forma detallada la base de datos, el sitio Web e interfases, los métodos de evaluación y procedimientos de cálculo para el Sistema de Evaluación Docente.

3.2.1. Diseño de la Base de Datos.

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

Información del Modelo Físico. Detalla información básica y estadística del modelo físico de la base de datos. Aquí se localiza el nombre de la base de datos, numero de entidades, atributos, índices y relaciones.

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

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

Columnas. Describe cada atributos de las entidad de la bases de datos, tipo de dato, si es clave primaria o secundaria, si es un dato nulo o único.

Relaciones. Individualiza las relaciones existentes en la base de datos, el tipo de relación, su tabla padre e hija y su cardinalidad.

Índices. Detalla a los índices que identifican las claves foráneas que se encuentra en una determinada entidad.

Información del Modelo Físico

Nombre Base de Datos	evaluacion
Tablas	25
Columnas	89
Índices	28
Relaciones	28

Diagrama Entidad – Relación

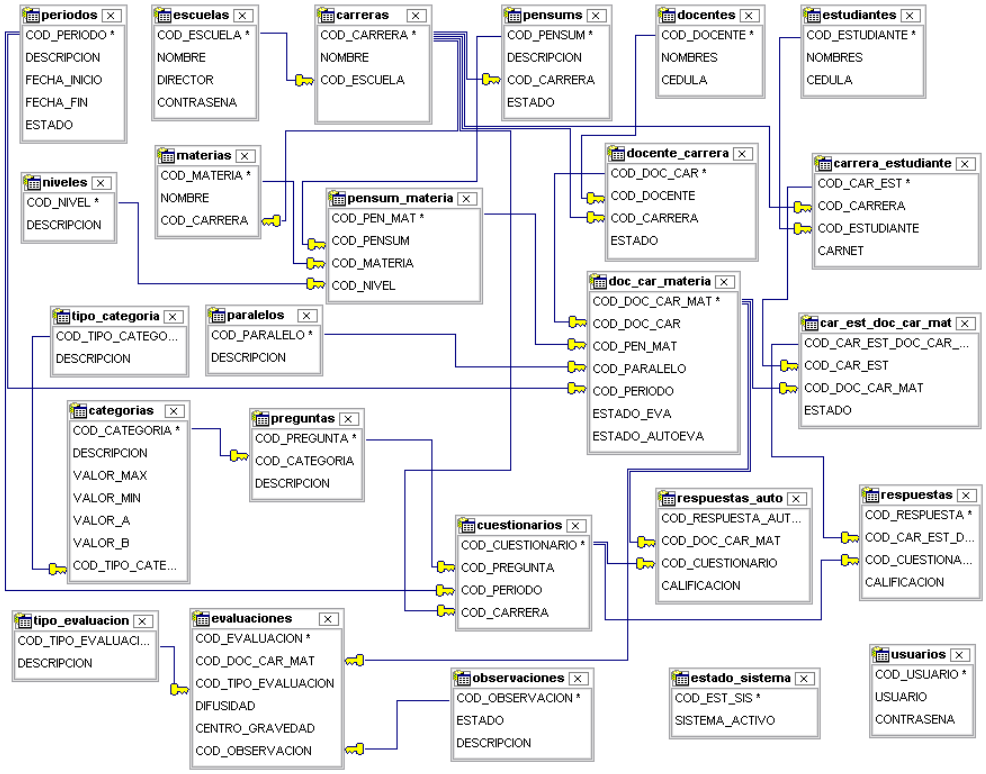


Figura 3.4: Diagrama Entidad - Relación

Tablas – Columnas – Relaciones – Índices

car_est_doc_car_mat

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_CAR_EST_DOC_CAR_MAT	Integer	si	no		
FK	COD_CAR_EST	Integer	no	no		
FK	COD_DOC_CAR_MAT	Integer	no	no		
	ESTADO	Char (1)	no	no		"N"

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_CAR_EST_DOC_CAR_MAT_27 (FK_CAR_EST_DOC_CAR_MAT_27)	car_est_doc_car_mat	respuestas	1:N
FK_ESTUDIANTES_23 (FK_ESTUDIANTES_23)	carrera_estudiante	car_est_doc_car_mat	1:N
FK_DOC_CAR_MAT_24 (FK_DOC_CAR_MAT_24)	doc_car_materia	car_est_doc_car_mat	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_CAR_EST	COD_CAR_EST	no	no	no
COD_DOC_CAR_MA	COD_DOC_CAR_MAT	no	no	no

carrera_estudiante

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_CAR_EST	Integer	si	no		
FK	COD_CARRERA	Integer	si	no		"0"
FK	COD_ESTUDIANTE	Integer	si	no		"0"
	CARNET	Varchar (10)	no	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_ESTUDIANTES_23 (FK_ESTUDIANTES_23)	carrera_estudiante	car_est_doc_car_mat	1:N
FK_CARRERAS_8 (FK_CARRERAS_8)	carreras	carrera_estudiante	1:N
FK_ESTUDIANTES_7 (FK_ESTUDIANTES_7)	estudiantes	carrera_estudiante	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_CARRERA	COD_CARRERA	no	no	no
COD_ESTUDIANTE	COD_ESTUDIANTE	no	no	no

carreras

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_CARRERA	Integer	si	no		
	NOMBRE	Varchar (50)	si	no		
FK	COD_ESCUELA	Integer	no	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_CARRERAS_8 (FK_CARRERAS_8)	carreras	carrera_estudiante	1:N
FK_CARRERAS_11 (FK_CARRERAS_11)	carreras	cuestionarios	1:N
FK_CARRERAS_4 (FK_CARRERAS_4)	carreras	docente_carrera	1:N
FK_CARRERAS_12 (FK_CARRERAS_12)	carreras	materias	1:N
FK_CARRERAS_3 (FK_CARRERAS_3)	carreras	pensums	1:N
FK_ESCUELAS_2 (FK_ESCUELAS_2)	escuelas	carreras	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_ESCUELA	COD_ESCUELA	no	no	no

categorias

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_CATEGORIA	Integer	si	no		
	DESCRIPCION	Varchar (100)	si	no		
	VALOR_MAX	Double	no	no		
	VALOR_MIN	Double	no	no		
	VALOR_A	Double	no	no		
	VALOR_B	Double	no	no		
FK	COD_TIPO_CATEGORIA	Integer	no	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_CATEGORIAS_6 (FK_CATEGORIAS_6)	categorias	preguntas	1:N
FK_TIPOS_1 (FK_TIPOS_1)	tipo_categoria	categorias	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_TIPO_CATEGORIA	COD_TIPO_CATEGORIA	no	no	no

cuestionarios

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_CUESTIONARIO	Integer	si	no		
FK	COD_PREGUNTA	Integer	no	no		
FK	COD_PERIODO	Integer	no	no		
FK	COD_CARRERA	Integer	no	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_CARRERAS_11 (FK_CARRERAS_11)	carreras	cuestionarios	1:N
FK_CUESTIONARIOS_28 (FK_CUESTIONARIOS_28)	cuestionarios	respuestas	1:N
FK_CUESTIONARIOS_26 (FK_CUESTIONARIOS_26)	cuestionarios	respuestas_auto	1:N
FK_PERIODOS_10 (FK_PERIODOS_10)	periodos	cuestionarios	1:N
FK_PREGUNTAS_9 (FK_PREGUNTAS_9)	preguntas	cuestionarios	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_PREGUNTA	COD_PREGUNTA	no	no	no
COD_PERIODO	COD_PERIODO	no	no	no
COD_CARRERA	COD_CARRERA	no	no	no

doc_car_materia

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_DOC_CAR_MAT	Integer	si	no		
FK	COD_DOC_CAR	Integer	no	no		
FK	COD_PEN_MAT	Integer	no	no		
FK	COD_PARALELO	Integer	si	no		"0"
FK	COD_PERIODO	Integer	si	no		"0"
	ESTADO_EVA	Char (1)	no	no		"N"
	ESTADO_AUTOEVA	Char (1)	no	no		"N"

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_DOC_CAR_MAT_24 (FK_DOC_CAR_MAT_24)	doc_car_materia	car_est_doc_car_mat	1:N
FK_DOC_CAR_MATERIAS_20 (FK_DOC_CAR_MATERIAS_20)	doc_car_materia	evaluaciones	1:N
FK_DOC_CAR_MATERIA_25 (FK_DOC_CAR_MATERIA_25)	doc_car_materia	respuestas_auto	1:N
FK_DOC_CAR_16 (FK_DOC_CAR_16)	docente_carrera	doc_car_materia	1:N
FK_PARALELOS_19 (FK_PARALELOS_19)	paralelos	doc_car_materia	1:N
FK_MATERIAS_17 (FK_MATERIAS_17)	pensum_materia	doc_car_materia	1:N
FK_PERIODOS_18 (FK_PERIODOS_18)	periodos	doc_car_materia	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_PERIODO	COD_PERIODO	no	no	no
COD_PARALELO	COD_PARALELO	no	no	no
COD_PEN_MAT	COD_PEN_MAT	no	no	no
COD_DOC_CAR	COD_DOC_CAR	no	no	no

docente_carrera

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_DOC_CAR	Integer	si	no		
FK	COD_DOCENTE	Integer	si	no		"0"
FK	COD_CARRERA	Integer	si	no		"0"

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_CARRERAS_4 (FK_CARRERAS_4)	carreras	docente_carrera	1:N
FK_DOC_CAR_16 (FK_DOC_CAR_16)	docente_carrera	doc_car_materia	1:N
FK_DOCENTES_5 (FK_DOCENTES_5)	docentes	docente_carrera	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_DOCENTE	COD_DOCENTE	no	no	no
COD_CARRERA	COD_CARRERA	no	no	no

docentes

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_DOCENTE	Integer	si	no		
	NOMBRES	Varchar (50)	si	no		
	CEDULA	Varchar (11)	no	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_DOCENTES_5 (FK_DOCENTES_5)	docentes	docente_carrera	1:N

escuelas

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_ESCUELA	Integer	si	no		
	NOMBRE	Varchar (50)	si	no		
	DIRECTOR	Varchar (50)	no	no		
	CONTRASENA	Varchar (15)	no	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_ESCUELAS_2 (FK_ESCUELAS_2)	escuelas	carreras	1:N

estudiantes

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_ESTUDIANTE	Integer	si	no		
	NOMBRES	Varchar (50)	si	no		
	CEDULA	Varchar (11)	si	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_ESTUDIANTES_7 (FK_ESTUDIANTES_7)	estudiantes	carrera_estudiante	1:N

evaluaciones

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_EVALUACION	Integer	si	no		
FK	COD_DOC_CAR_MAT	Integer	si	no		"0"
FK	COD_TIPO_EVALUACION	Integer	no	no		
	DIFUSIDAD	Double	no	no		
	CENTRO_GRAVEDAD	Double	no	no		
FK	COD_OBSERVACION	Integer	no	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_DOC_CAR_MATERIAS_20 (FK_DOC_CAR_MATERIAS_20)	doc_car_materia	evaluaciones	1:N
FK_OBSERVACIONES_22 (FK_OBSERVACIONES_22)	observaciones	evaluaciones	1:N
FK_TIPO_EVALUACION_21 (FK_TIPO_EVALUACION_21)	tipo_evaluacion	evaluaciones	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_OBSERVACION	COD_OBSERVACION	no	no	no
COD_DOC_CAR_MAT	COD_DOC_CAR_MAT	no	no	no
COD_TIPO_EVALUACION	COD_TIPO_EVALUACION	no	no	no

materias

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_MATERIA	Integer	si	no		
	NOMBRE	Varchar (80)	si	no		
FK	COD_CARRERA	Integer	no	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_CARRERAS_12 (FK_CARRERAS_12)	carreras	materias	1:N
FK_MATERIAS_15 (FK_MATERIAS_15)	materias	pensum_materia	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_CARRERA	COD_CARRERA	no	no	no

niveles

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_NIVEL	Integer	si	no		
	DESCRIPCION	Varchar (15)	si	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_NIVELES_13 (FK_NIVELES_13)	niveles	pensum_materia	1:N

observaciones

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_OBSERVACION	Integer	si	no		
	ESTADO	Varchar (30)	no	no		
	DESCRIPCION	Varchar (200)	no	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_OBSERVACIONES_22 (FK_OBSERVACIONES_22)	observaciones	evaluaciones	1:N

paralelos

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_PARALELO	Integer	si	no		
	DESCRIPCION	Char (1)	si	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_PARALELOS_19 (FK_PARALELOS_19)	paralelos	doc_car_materia	1:N

pensum_materia

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_PEN_MAT	Integer	si	no		
FK	COD_PENSUM	Integer	no	no		
FK	COD_MATERIA	Integer	no	no		
FK	COD_NIVEL	Integer	no	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_MATERIAS_15 (FK_MATERIAS_15)	materias	pensum_materia	1:N
FK_NIVELES_13 (FK_NIVELES_13)	niveles	pensum_materia	1:N
FK_MATERIAS_17 (FK_MATERIAS_17)	pensum_materia	doc_car_materia	1:N
FK_PENSUMS_14 (FK_PENSUMS_14)	pensums	pensum_materia	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_NIVEL	COD_NIVEL	no	no	no
COD_PENSUM	COD_PENSUM	no	no	no
COD_MATERIA	COD_MATERIA	no	no	no

pensums

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_PENSUM	Integer	si	no		
	DESCRIPCION	Varchar (50)	si	no		
FK	COD_CARRERA	Integer	no	no		
	ESTADO	Char (1)	no	no		"N"

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_CARRERAS_3 (FK_CARRERAS_3)	carreras	pensums	1:N
FK_PENSUMS_14 (FK_PENSUMS_14)	pensums	pensum_materia	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_CARRERA	COD_CARRERA	no	no	no

periodos

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_PERIODO	Integer	si	no		
	DESCRIPCION	Varchar (50)	no	no		
	FECHA_INICIO	Date	no	no		
	FECHA_FIN	Date	no	no		
	ESTADO	Char (1)	no	no		"N"

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_PERIODOS_10 (FK_PERIODOS_10)	periodos	cuestionarios	1:N
FK_PERIODOS_18 (FK_PERIODOS_18)	periodos	doc_car_materia	1:N

preguntas

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_PREGUNTA	Integer	si	no		
FK	COD_CATEGORIA	Integer	si	no		"0"
	DESCRIPCION	Varchar (200)	si	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_CATEGORIAS_6 (FK_CATEGORIAS_6)	categorias	preguntas	1:N
FK_PREGUNTAS_9 (FK_PREGUNTAS_9)	preguntas	cuestionarios	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_CATEGORIA	COD_CATEGORIA	no	no	no

respuestas

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_RESPUESTA	Integer	si	no		
FK	COD_CAR_EST_DOC_CAR_MAT	Integer	si	no		"0"
FK	COD_CUESTIONARIO	Integer	no	no		
	CALIFICACION	Double	no	no		"0"

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_CAR_EST_DOC_CAR_MAT_27 (FK_CAR_EST_DOC_CAR_MAT_27)	car_est_doc_car_mat	respuestas	1:N
FK_CUESTIONARIOS_28 (FK_CUESTIONARIOS_28)	cuestionarios	respuestas	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_CUESTIONARIO	COD_CUESTIONARIO	no	no	no
COD_CAR_EST_DOC_CAR_MAT	COD_CAR_EST_DOC_CAR_MAT	no	no	no

respuestas_auto

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_RESPUESTA_AUTO	Integer	si	no		
FK	COD_DOC_CAR_MAT	Integer	si	no		"0"
FK	COD_CUESTIONARIO	Integer	no	no		
	CALIFICACION	Double	no	no		"0"

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_CUESTIONARIOS_26 (FK_CUESTIONARIOS_26)	cuestionarios	respuestas_auto	1:N
FK_DOC_CAR_MATERIA_25 (FK_DOC_CAR_MATERIA_25)	doc_car_materia	respuestas_auto	1:N

Índices

Nombre indice	Columnas	Único	Descendiente	Clustered
COD_DOC_CAR_MAT	COD_DOC_CAR_MAT	no	no	no
COD_CUESTIONARIO	COD_CUESTIONARIO	no	no	no

tipo_categoria

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_TIPO_CATEGORIA	Integer	si	no		
	DESCRIPCION	Varchar (15)	si	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_TIPOS_1 (FK_TIPOS_1)	tipo_categoria	categorias	1:N

tipo_evaluacion

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_TIPO_EVALUACION	Integer	si	no		
	DESCRIPCION	Varchar (15)	si	no		

Relaciones

Nombre de la restricción (relación)	Tabla padre	Tabla hija	Card.
FK_TIPO_EVALUACION_21 (FK_TIPO_EVALUACION_21)	tipo_evaluacion	evaluaciones	1:N

usuarios

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_USUARIO	Integer	si	no		
	USUARIO	Varchar (30)	si	no		
	CONTRASENA	Varchar (15)	no	no		

estado_sistema

Columnas

Clave	Nombre columna	Tipo dato	No nulo	Único	Check	Por defecto
PK	COD_EST_SIS	Integer	si	no		
	SISETMA_ACTIVO	Char (1)	si	no		"S"

3.2.2. Diseño del Sitio Web y la Interfase.

El diseño del sitio Web del SED define la interfase del sistema de forma jerárquica revelando los usuarios y las pantallas a las que tienen acceso.

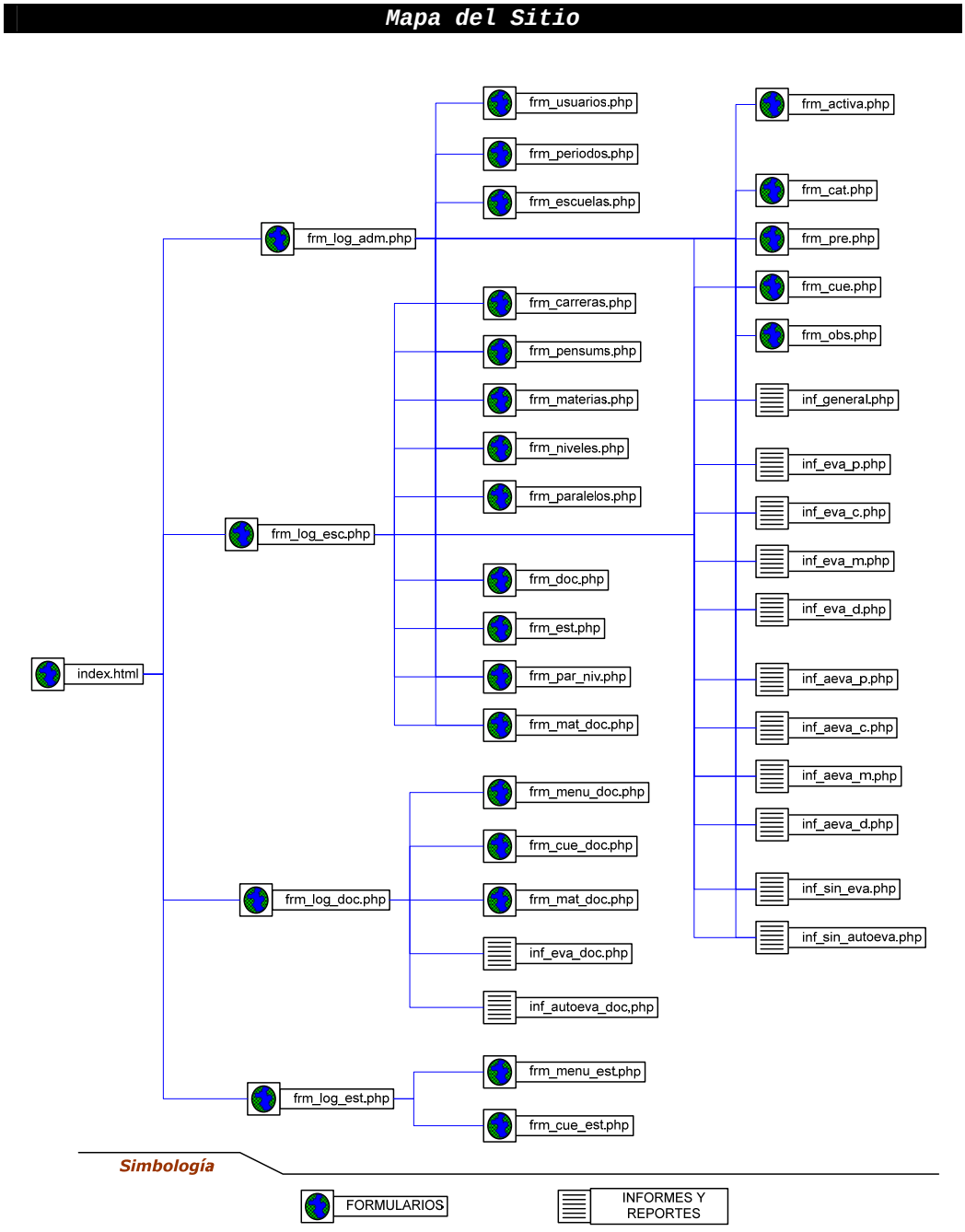


Figura 3.5: Mapa del Sitio

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

Archivos Sitio Web

Nombre del archivo	Descripción
imagenes	Carpeta de imágenes
index.html	Página de inicio del sistema
mm_menu.js	Página javascript con funciones y procedimientos
estilos.ccs	Página de estilos
clases.php	Clases, funciones y atributos para el sistema
conexion.php	Página de conexión con la base de datos
frm_acerca.php	Acerca del sistema SED
frm_activa.php	Activa y desactiva el sistema
frm_ayuda.php	Ayuda del Sistema
frm_cabeza.php	Encabezado de páginas
frm_car.php	Mantenimiento de Carreras
frm_cat.php	Mantenimiento de Categorías
frm_cue.php	Mantenimiento de Cuestionarios
frm_cue_doc.php	Cuestionarios de Autoevaluación para el Docente
frm_cue_est.php	Cuestionarios de Evaluación para el Estudiante
frm_doc.php	Mantenimiento de Docentes
frm_esc.php	Mantenimiento de Escuelas
frm_est.php	Mantenimiento de Estudiantes
frm_ing_car.php	Ingresar Carreras
frm_ing_cat.php	Ingresar Categorías
frm_ing_cue.php	Ingresar Cuestionarios
frm_ing_doc.php	Ingresar Docentes
frm_ing_esc.php	Ingresar Escuelas
frm_ing_est.php	Ingresar Estudiantes
frm_ing_m_e.php	Asignar Materias a Estudiantes
frm_ing_mat.php	Ingresar Materias
frm_ing_niv.php	Ingresar Niveles
frm_ing_p_m.php	Crear materias por paralelos
frm_ing_p_n.php	Asignar Paralelos a Niveles
frm_ing_par.php	Ingresar Paralelos
frm_ing_pen.php	Ingresar Pénsums
frm_ing_per.php	Ingresar Períodos
frm_ing_pre.php	Ingresar Preguntas
frm_ing_user.php	Ingresar Usuarios
frm_log_adm.php	Pantalla de ingreso del usuario Administrador
frm_log_doc.php	Pantalla de ingreso del usuario Docente
frm_log_esc.php	Pantalla de ingreso del usuario Escuela
frm_log_est.php	Pantalla de ingreso del usuario Estudiante
frm_logout.php	Página de logout
frm_mat.php	Mantenimiento de Materias
frm_mat_doc.php	Mantenimiento Materias - Docentes
frm_mat_est.php	Mantenimiento Materias - Estudiantes
frm_menu.php	Menú del usuario
frm_menu_doc.php	Menú del usuario Docente

frm_menu_est.php	Menú del usuario Estudiante
frm_mod_car.php	Modificar Carreras
frm_mod_cat.php	Modificar Categorías
frm_mod_doc.php	Modificar Docentes
frm_mod_esc.php	Modificar Escuelas
frm_mod_est.php	Modificar Estudiantes
frm_mod_m_d.php	Modificar Materias - Docentes
frm_mod_mat.php	Modificar Materias
frm_mod_niv.php	Modificar Niveles
frm_mod_obs.php	Modificar Observaciones
frm_mod_par.php	Modificar Paralelos
frm_mod_pen.php	Modificar Pénsums
frm_mod_per.php	Modificar Períodos
frm_mod_pre.php	Modificar Preguntas
frm_mod_user.php	Modificar Usuarios
frm_niv.php	Mantenimiento de Niveles
frm_obs.php	Mantenimiento de Observaciones
frm_par.php	Mantenimiento de Paralelos
frm_par_niv.php	Mantenimiento Niveles
frm_pen.php	Mantenimiento de Pénsums
frm_per.php	Mantenimiento de Períodos
frm_pre.php	Mantenimiento de Preguntas
frm_user.php	Mantenimiento de Usuarios
funciones.php	Funciones del sistema
inf_aeva_c.php	Informe de Autoevaluación por Carreras
inf_aeva_c_p.php	Informe de Autoevaluación por Carrera - Preguntas
inf_aeva_d.php	Informe de Autoevaluación por Docentes
inf_aeva_d_p.php	Informe de Autoevaluación por Docente - Preguntas
inf_aeva_m.php	Informe de Autoevaluación por Materia
inf_aeva_m_p.php	Informe de Autoevaluación por Materias - Pregunta
inf_aeva_p.php	Informe de Autoevaluación por Período
inf_aeva_p_p.php	Informe de Autoevaluación por Período - Pregunta
inf_autoeva_doc.php	Informe de Autoevaluación al Docente
inf_eva_c.php	Informe de Evaluación por Materias - Pregunta
inf_eva_c_p.php	Informe de Evaluación por Período
inf_eva_d.php	Informe de Evaluación por Período - Pregunta
inf_eva_d_p.php	Informe de Evaluación por Docente - Preguntas
inf_eva_doc.php	Informe de Evaluación al Docente
inf_eva_m.php	Informe de Evaluación por Materias - Pregunta
inf_eva_m_p.php	Informe de Evaluación por Período
inf_eva_p.php	Informe de Evaluación por Período - Pregunta
inf_eva_p_p.php	Informe de Evaluación por Período - Pregunta
inf_g_autoeva.php	Informa general de Autoevaluación
inf_g_eva.php	Informe general de Evaluación
inf_general.php	Informe General
inf_sin_autoeva.php	Informe Docentes sin Autoevaluación
inf_sin_eva.php	Informe Estudiantes sin Evaluación

Menú del Usuario

Menú Usuario Administrador – Usuario Escuela

Menú	Usuario Administrador	Usuario Escuela	Página
1. Mantenimiento	si	si	-
1.1. Usuarios	si	no	frm_user.php
1.2. Periodos	si	no	frm_per.php
1.3. Escuelas	si	no	frm_esc.php
1.4. Carreras	si	no	frm_car.php
1.5. Pensums	si	si	frm_pen.php
1.6. Materias	si	si	frm_mat.php
1.7. Niveles	si	si	frm_niv.php
1.8. Paralelos	si	si	frm_par.php
2. Procesos	si	si	-
2.1. Administración de	si	si	-
2.1.1. Docentes	si	si	frm_doc.php
2.1.2. Estudiantes	si	si	frm_est.php
2.2. Asignación de Datos	si	si	-
2.2.1. Paralelos a Niveles	si	si	frm_par_niv.php
2.2.2. Materias a Docentes	si	si	frm_mat_doc.php
3. Evaluación	si	si	-
3.1. Activar / Desactivar Sistema	si	no	frm_activa.php
3.2. Ingreso de Datos	si	si	-
3.2.1. Categorías	si	no	frm_cat.php
3.2.2. Preguntas	si	si	frm_pre.php
3.2.3. Cuestionarios	si	si	frm_cue.php
3.2.4. Observaciones	si	no	frm_obs.php
4. Reportes	si	si	-
4.1. Consulta General	si	si	frm_general.php
4.2. Informe Evaluación	si	si	-
4.2.1. Por Período	si	si	frm_eva_p.php
4.2.2. Por Carrera (Comparativo)	si	si	frm_eva_c.php
4.2.3. Por Materia	si	si	frm_eva_m.php
4.2.4. Por Docente	si	si	frm_eva_d.php
4.3. Informe AutoEvaluación	si	si	-
4.3.1. Por Período	si	si	frm_eva_p.php
4.3.2. Por Carrera (Comparativo)	si	si	frm_eva_c.php
4.3.3. Por Materia	si	si	frm_eva_m.php
4.3.4. Por Docente	si	si	frm_eva_d.php
4.4. Estudiantes sin Evaluación	si	si	frm_sin_eva.php
4.5. Docentes sin AutoEvaluación	si	si	frm_sin_auto.php
5. Ayuda	si	si	-
5.1. Ayuda de Evaluación Docente	si	si	frm_ayuda.php
5.2. Acerca de Evaluación Docente	si	si	frm_acerca.php
6. Cerrar Sesión	si	si	frm_logout.php

Menú Usuario Docente

Menú	Página
Menú del Docente	frm_menu_doc.php
Cerrar Sesión	frm_logout.php

Menú Usuario Estudiante

Menú	Página
Menú del Estudiante	frm_menu_est.php
Cerrar Sesión	frm_logout.php

3.2.3. Diseño de Procedimientos para el Cálculo de Evaluación.

El sistema SED utiliza Lógica Difusa que permite valores intermedios para poder definir evaluaciones convencionales con el propósito de realizar una tentativa de aplicar una forma más natural de representar el resultado de evaluación.

Se ha desarrollado cálculos aritméticos con el fin de obtener la difusidad por cada docente y el centro de gravedad para poder brindar una evaluación cualitativa. Cada variable de entrada (evaluación realizada) ingresa a un análisis matemático de Lógica Difusa utilizando el diagrama ***tipo G*** (*ver figura 3.5*) para permitir valores intermedios y proponer los resultados no solo con deducciones convencionales como si/no, verdadero/falso, 0/1; sino manifestando una grado de evaluación establecido en rangos dinámicos relativos al tipo de variable.

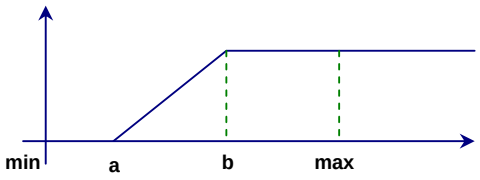


Figura 3.6: Diagrama tipo G - Variable de Entrada.

El proceso de evaluación docente en la universidad consta de 4 categorías que son: Sociabilidad Pedagógica, Gestión de Clases, Impacto Emocional y Refuerzo de Habilidades Básicas; cada una de ellas estará representada por el diagrama ***tipo G*** encontrándose constituida como un ***tipo de categoría de Entrada*** y con los valores por defecto de:

```
min = 0
max = 5
a   = 1.5
b   = 4.5
```

Estos valores se encuentran justificados en la medida que el rango de calificación para el proceso de evaluación docente que se ha utilizado se encuentra entre 0 - 5 y, tomando en cuenta que el resultado de una evaluación ***x*** con un resultado ***y*** mayor de 4.5 se considera como ***destacado*** y un resultado ***z*** menor que 1.5 como ***insatisfactorio***; resulta ineludible hallar rangos para los resultados de ***competente***, ***básico*** y ***regular*** que se encontraren entre 1.5 - 4.5.

Para ello es necesario analizar y procesar todas las evaluaciones (***u***) en un conjunto difuso entre 0 y 1 que admite infinitas alternativas mediante las siguientes fórmulas para hallar su difusidad:

$$a > u \rightarrow 0$$

$$a < u < b \rightarrow \frac{u - a}{b - a}$$

$$u > b \rightarrow 1$$

Se toma en cuenta que:

- Si un valor ***u*** es menor al valor de ***a***, su difusidad será de ***0***.
- Si un valor ***u*** es mayor al valor de ***b***, su difusidad será de ***1***.
- Si un valor ***u*** se encuentra en el intervalo de ***a*** y ***b***, será necesario realizar la operación determinada anteriormente con el propósito de encontrar su difusidad entre ***0*** y ***1***.

Con el conjunto difuso encontrado se procede a realizar las operaciones de la ***variable de Salida*** la cual utiliza el mismo diagrama ***tipo G*** con distintos atributos.

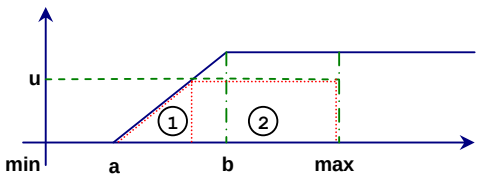


Figura 3.7: Diagrama tipo G - Variable de Salida.

Los valores por defecto para la variable de salida son:

```
min = 0              max = 100
a   = 15             b   = 95
```

Como se observa en la figura anterior (*ver figura 3.6*), la difusidad ***u***, que representa al conjunto difuso comprendido en el intervalo entre 0 y 1, intersecta con el diagrama ***tipo G*** e instaura dos figuras geométricas las cuales entran a un análisis por el sistema SED para generar un resultado por determinado criterio de búsqueda de evaluación requerida por el usuario. Para ello es necesario aplicar las siguientes fórmulas matemáticas de lógica difusa para encontrar el centro de gravedad de las figuras en conjunto:

$$base_{①} = u * (b - a)$$
$$base_{②} = max - a - x$$
$$altura_{①} = u$$
$$altura_{②} = u$$
$$brazo_{①} = a + \tfrac{2}{3}\, base_{①}$$
$$brazo_{②} = a + \tfrac{1}{2}\, base_{②} + base_{①}$$
$$area_{①} = \tfrac{1}{2}\, altura_{①} * brazo_{①}$$
$$area_{②} = altura_{②} * brazo_{②}$$
$$\sum a * b = area_{①} + area_{②}$$
$$\sum alturas = altura_{①} + altura_{②}$$

$$resultado = \frac{\sum a * b}{\sum alturas}$$

Una vez encontrado un sub resultado para el proceso de evaluación, es preciso realizar una corrección para generar una respuesta más cerca al lenguaje natural entre un rango de 30 a 50; vista la necesidad de la universidad de representar los resultados de evaluación docente de forma similar a las calificaciones que reciben los alumnos, estos resultados se distribuyen en las categorías mencionadas anteriormente en los siguientes rangos:

```
De 48.3376 a 50          Destacado
De 43.5336 a 48.3376     Competente
```

De 38.1260 a 43.5336	***Básico***
De 32.7224 a 38.1260	***Regular***
Menor de 32.7224	***Insatisfactorio***

Es necesario indicar que estos valores fluctúan en base a los valores de las variables de entrada y salida detalladas anteriormente las cuales el sistema SED toma por defecto y que se encuentran instauradas en la base de datos. Si se modifican los valores de las variables los rangos variarán dependiendo de los nuevos valores.

3.3. Desarrollo del sistema.

El desarrollo del sistema SED comprende la creación, codificación e interacción de los diferentes módulos entre si fundamentándose en el diseño y análisis del sistema, desarrollar la codificación del ***front end***, generando las diferentes entidades con sus atributos para el ***back end*** y codificando las diferentes operaciones matemáticas, con el propósito de alcanzar un perfecto funcionamiento para que el proceso de evaluación docente en la universidad se desarrolle de la mejor manera.

3.3.1. Creación de la Base de Datos.

Se ha establecido una estructura SQL para la creación de la base de datos, las entidades con sus atributos y los valores iniciales que hospeda el ***back end*** del sistema SED en el motor de base de datos ***MySQL***, la cual es la siguiente:

Creación de la Base de Datos

```
-- Creación de la Base de Datos `evaluacion`
CREATE DATABASE evaluacion;

-- Uso de la Base de Datos `evaluacion`
USE evaluacion;

-- Colocar parámetro de inicialización
SET FOREIGN_KEY_CHECKS=0;
```

Tabla 'car_est_doc_car_mat'

```
-- Estructura para la tabla `car_est_doc_car_mat`

CREATE TABLE car_est_doc_car_mat (
  COD_CAR_EST_DOC_CAR_MAT int(11) NOT NULL auto_increment,
  COD_CAR_EST int(11) default NULL,
  COD_DOC_CAR_MAT int(11) default NULL,
  ESTADO char(1) default 'N',
  PRIMARY KEY  (COD_CAR_EST_DOC_CAR_MAT),
  KEY COD_CAR_EST (COD_CAR_EST),
  KEY COD_DOC_CAR_MA (COD_DOC_CAR_MAT),
  CONSTRAINT `FK_DOC_CAR_MAT_24` FOREIGN KEY (`COD_DOC_CAR_MAT`) REFERENCES
`doc_car_materia` (`COD_DOC_CAR_MAT`),
  CONSTRAINT `FK_ESTUDIANTES_23` FOREIGN KEY (`COD_CAR_EST`) REFERENCES `car-
rera_estudiante` (`COD_CAR_EST`)
) TYPE=InnoDB;
```

Tabla 'carrera_estudiante'

```
-- Estructura para la tabla `carrera_estudiante`

CREATE TABLE carrera_estudiante (
  COD_CAR_EST int(11) NOT NULL auto_increment,
  COD_CARRERA int(11) NOT NULL default '0',
  COD_ESTUDIANTE int(11) NOT NULL default '0',
  CARNET varchar(10) default NULL,
  PRIMARY KEY  (COD_CAR_EST),
  KEY COD_ESTUDIANTE (COD_ESTUDIANTE),
  KEY COD_CARRERA (COD_CARRERA),
  CONSTRAINT `FK_CARRERAS_8` FOREIGN KEY (`COD_CARRERA`) REFERENCES `carreras`
(`COD_CARRERA`),
  CONSTRAINT `FK_ESTUDIANTES_7` FOREIGN KEY (`COD_ESTUDIANTE`) REFERENCES `es-
tudiantes` (`COD_ESTUDIANTE`)
) TYPE=InnoDB;
```

Tabla 'carreras'

```
-- Estructura para la tabla `carreras`

CREATE TABLE carreras (
  COD_CARRERA int(11) NOT NULL auto_increment,
  NOMBRE varchar(50) NOT NULL default '',
  COD_ESCUELA int(11) default NULL,
  PRIMARY KEY  (COD_CARRERA),
  KEY COD_ESCUELA (COD_ESCUELA),
  CONSTRAINT `FK_ESCUELAS_2` FOREIGN KEY (`COD_ESCUELA`) REFERENCES `escuelas`
(`COD_ESCUELA`)
) TYPE=InnoDB;

-- Datos de inicio para la tabla `carreras`

INSERT INTO  carreras VALUES (1,'Ingeniería en Sistemas',1);
INSERT INTO  carreras VALUES (2,'Ingeniería en Diseño Industrial',2);
INSERT INTO  carreras VALUES (3,'Optometría',3);
INSERT INTO  carreras VALUES (4,'Ingeniería Comercial',4);
INSERT INTO  carreras VALUES (5,'Formación Dual en Gerencia de PYMES',4);
```

Tabla 'categorias'

```
-- Estructura para la tabla `categorias`

CREATE TABLE categorias (
  COD_CATEGORIA int(11) NOT NULL auto_increment,
  DESCRIPCION varchar(100) NOT NULL default '',
  VALOR_MAX double default NULL,
  VALOR_MIN double default NULL,
  VALOR_A double default NULL,
  VALOR_B double default NULL,
  COD_TIPO_CATEGORIA int(11) default NULL,
  PRIMARY KEY  (COD_CATEGORIA),
  KEY COD_TIPO_CATEGORIA (COD_TIPO_CATEGORIA),
  CONSTRAINT `FK_TIPOS_1` FOREIGN KEY (`COD_TIPO_CATEGORIA`) REFERENCES `ti-
po_categoria` (`COD_TIPO_CATEGORIA`)
```

```
) TYPE=InnoDB;

-- Datos de inicio para la tabla `categorias`

INSERT INTO  categorias VALUES (1,'Sociabilidad Pedagógica',5,0,1.5,4.5,1);
INSERT INTO  categorias VALUES (2,'Gestion de Clases',5,0,1.5,4.5,1);
INSERT INTO  categorias VALUES (3,'Impacto Emocional',5,0,1.5,4.5,1);
INSERT INTO  categorias VALUES (4,'Refuerzo de Habilidades Básicas',5,0,1.5,4.5,1);
INSERT INTO  categorias VALUES (6,'Salida',100,0,15,95,2);
```

Tabla 'cuestionarios

```
-- Estructura para la tabla `cuestionarios`

CREATE TABLE cuestionarios (
  COD_CUESTIONARIO int(11) NOT NULL auto_increment,
  COD_PREGUNTA int(11) default NULL,
  COD_PERIODO int(11) default NULL,
  COD_CARRERA int(11) default NULL,
  PRIMARY KEY  (COD_CUESTIONARIO),
  KEY COD_PREGUNTA (COD_PREGUNTA),
  KEY COD_PERIODO (COD_PERIODO),
  KEY COD_CARRERA (COD_CARRERA),
  CONSTRAINT `FK_CARRERAS_11` FOREIGN KEY (`COD_CARRERA`) REFERENCES `carreras` (`COD_CARRERA`),
  CONSTRAINT `FK_PERIODOS_10` FOREIGN KEY (`COD_PERIODO`) REFERENCES `periodos` (`COD_PERIODO`),
  CONSTRAINT `FK_PREGUNTAS_9` FOREIGN KEY (`COD_PREGUNTA`) REFERENCES `preguntas` (`COD_PREGUNTA`)
) TYPE=InnoDB;
```

Tabla 'doc_car_materia'

```
-- Estructura para la tabla `doc_car_materia`

CREATE TABLE doc_car_materia (
  COD_DOC_CAR_MAT int(11) NOT NULL auto_increment,
  COD_DOC_CAR int(11) default NULL,
  COD_PEN_MAT int(11) default NULL,
  COD_PARALELO int(11) NOT NULL default '0',
  COD_PERIODO int(11) NOT NULL default '0',
  ESTADO_EVA char(1) default 'N',
  ESTADO_AUTOEVA char(1) default 'N',
  PRIMARY KEY  (COD_DOC_CAR_MAT),
  KEY COD_DOC_CAR (COD_DOC_CAR),
  KEY COD_PEN_MAT (COD_PEN_MAT),
  KEY COD_PERIODO (COD_PERIODO),
  KEY COD_PARALELO (COD_PARALELO),
  CONSTRAINT `FK_DOC_CAR_16` FOREIGN KEY (`COD_DOC_CAR`) REFERENCES `docente_carrera` (`COD_DOC_CAR`),
  CONSTRAINT `FK_MATERIAS_17` FOREIGN KEY (`COD_PEN_MAT`) REFERENCES `pensum_materia` (`COD_PEN_MAT`),
  CONSTRAINT `FK_PARALELOS_19` FOREIGN KEY (`COD_PARALELO`) REFERENCES `paralelos` (`COD_PARALELO`),
  CONSTRAINT `FK_PERIODOS_18` FOREIGN KEY (`COD_PERIODO`) REFERENCES `periodos` (`COD_PERIODO`)
) TYPE=InnoDB;
```

Tabla 'docente_carrera'

```
-- Estructura para la tabla `docente_carrera`

CREATE TABLE docente_carrera (
  COD_DOC_CAR int(11) NOT NULL auto_increment,
  COD_DOCENTE int(11) NOT NULL default '0',
  COD_CARRERA int(11) NOT NULL default '0',
  ESTADO char(1) default 'S',
  PRIMARY KEY  (COD_DOC_CAR),
  KEY COD_CARRERA (COD_CARRERA),
  KEY COD_DOCENTE (COD_DOCENTE),
  CONSTRAINT `FK_CARRERAS_4` FOREIGN KEY (`COD_CARRERA`) REFERENCES `carreras` (`COD_CARRERA`),
```

```
  CONSTRAINT `FK_DOCENTES_5` FOREIGN KEY (`COD_DOCENTE`) REFERENCES `docentes`
(`COD_DOCENTE`)
) TYPE=InnoDB;
```

Tabla 'docentes'

```
-- Estructura para la tabla `docentes`

CREATE TABLE docentes (
  COD_DOCENTE int(11) NOT NULL auto_increment,
  NOMBRES varchar(50) NOT NULL default '',
  CEDULA varchar(11) default NULL,
  PRIMARY KEY  (COD_DOCENTE)
) TYPE=InnoDB;
```

Tabla 'escuelas'

```
-- Estructura para la tabla `escuelas`

CREATE TABLE escuelas (
  COD_ESCUELA int(11) NOT NULL auto_increment,
  NOMBRE varchar(50) NOT NULL default '',
  DIRECTOR varchar(50) default NULL,
  CONTRASENA varchar(15) default NULL,
  PRIMARY KEY  (COD_ESCUELA)
) TYPE=InnoDB;

-- Datos de inicio para la tabla `escuelas`

INSERT INTO  escuelas VALUES (1,'Escuela de Ingeniería de Sistemas','Ing. Tel-
mo Viteri','');
INSERT INTO  escuelas VALUES (2,'Escuela de Diseño','','');
INSERT INTO  escuelas VALUES (3,'Programa de Optometría',NULL,'');
INSERT INTO  escuelas VALUES (4,'Escuela de Ingeniería Comercial y Py-
mes',NULL,'');
```

Tabla 'estado_sistema'

```
-- Estructura para la tabla `estado_sistema`

CREATE TABLE estado_sistema (
  COD_EST_SIS int(11) NOT NULL auto_increment,
  SISTEMA_ACTIVO char(1) default 'S',
  PRIMARY KEY  (COD_EST_SIS)
) TYPE=MyISAM;

-- Datos de inicio para la tabla `estado_sistema`

INSERT INTO  estado_sistema VALUES (1,'N');
```

Tabla 'estudiantes'

```
-- Estructura para la tabla `estudiantes`

CREATE TABLE estudiantes (
  COD_ESTUDIANTE int(11) NOT NULL auto_increment,
  NOMBRES varchar(50) NOT NULL default '',
  CEDULA varchar(11) NOT NULL default '',
  PRIMARY KEY  (COD_ESTUDIANTE)
) TYPE=InnoDB;
```

Tabla 'evaluaciones'

```
-- Estructura para la tabla `evaluaciones`

CREATE TABLE evaluaciones (
  COD_EVALUACION int(11) NOT NULL auto_increment,
  COD_DOC_CAR_MAT int(11) NOT NULL default '0',
  COD_TIPO_EVALUACION int(11) default NULL,
  DIFUSIDAD double default NULL,
  CENTRO_GRAVEDAD double default NULL,
  COD_OBSERVACION int(11) default NULL,
  PRIMARY KEY  (COD_EVALUACION),
  KEY COD_DOC_CAR_MAT (COD_DOC_CAR_MAT),
```

```
  KEY COD_TIPO_EVALUACION (COD_TIPO_EVALUACION),
  KEY COD_OBSERVACION (COD_OBSERVACION),
  CONSTRAINT `FK_DOC_CAR_MATERIAS_20` FOREIGN KEY (`COD_DOC_CAR_MAT`) REFER-
ENCES `doc_car_materia` (`COD_DOC_CAR_MAT`),
  CONSTRAINT `FK_OBSERVACIONES_22` FOREIGN KEY (`COD_OBSERVACION`) REFERENCES
`observaciones` (`COD_OBSERVACION`),
  CONSTRAINT `FK_TIPO_EVALUACION_21` FOREIGN KEY (`COD_TIPO_EVALUACION`) REFE-
RENCES `tipo_evaluacion` (`COD_TIPO_EVALUACION`)
) TYPE=InnoDB;
```

Tabla 'materias'

```
-- Estructura para la tabla `materias`

CREATE TABLE materias (
  COD_MATERIA int(11) NOT NULL auto_increment,
  NOMBRE varchar(80) NOT NULL default '',
  COD_CARRERA int(11) default NULL,
  PRIMARY KEY  (COD_MATERIA),
  KEY COD_CARRERA (COD_CARRERA),
  CONSTRAINT `FK_CARRERAS_12` FOREIGN KEY (`COD_CARRERA`) REFERENCES `carre-
ras` (`COD_CARRERA`)
) TYPE=InnoDB;
```

Tabla 'niveles'

```
-- Estructura para la tabla `niveles`

CREATE TABLE niveles (
  COD_NIVEL int(11) NOT NULL auto_increment,
  DESCRIPCION varchar(15) NOT NULL default '',
  PRIMARY KEY  (COD_NIVEL)
) TYPE=InnoDB;

-- Datos de inicio para la tabla `niveles`

INSERT INTO  niveles VALUES (1,'Primero');
INSERT INTO  niveles VALUES (2,'Segundo');
INSERT INTO  niveles VALUES (3,'Tercero');
INSERT INTO  niveles VALUES (4,'Cuarto');
INSERT INTO  niveles VALUES (5,'Quinto');
INSERT INTO  niveles VALUES (6,'Sexto');
INSERT INTO  niveles VALUES (7,'Septimo');
INSERT INTO  niveles VALUES (8,'Octavo');
INSERT INTO  niveles VALUES (9,'Noveno');
INSERT INTO  niveles VALUES (10,'Décimo');
INSERT INTO  niveles VALUES (11,'Nivelación');
```

Tabla 'observaciones'

```
-- Estructura para la tabla `observaciones`

CREATE TABLE observaciones (
  COD_OBSERVACION int(11) NOT NULL auto_increment,
  ESTADO varchar(30) default NULL,
  DESCRIPCION varchar(200) default NULL,
  PRIMARY KEY  (COD_OBSERVACION)
) TYPE=InnoDB;

-- Datos de inicio para la tabla `observaciones`

INSERT INTO  observaciones VALUES (1,'destacado','El docente se encuentra en
un excelente estado después de la evaluación realizada.');
INSERT INTO  observaciones VALUES (2,'competente','El docente se encuentra en
un estado Muy Bueno, puede mejorar.');
INSERT INTO  observaciones VALUES (3,'básico','El docente se encuentra en un
estado Bueno, es necesario que revise en las áreas en las que ha fallado.');
INSERT INTO  observaciones VALUES (4,'regular','El docente se encuentra en un
estado Regular, es indispensable revisar en que área esta fallando.');
INSERT INTO  observaciones VALUES (5,'insatisfactorio','El docente se encuen-
tra en un estado pésimo después de la evaluación docente, es necesario que se
comunique con el director de la escuela.');
```

Tabla 'paralelos'

```
-- Estructura para la tabla `paralelos`

CREATE TABLE paralelos (
  COD_PARALELO int(11) NOT NULL auto_increment,
  DESCRIPCION char(1) NOT NULL default '',
  PRIMARY KEY  (COD_PARALELO)
) TYPE=InnoDB;

-- Datos de inicio para la tabla `paralelos`

INSERT INTO  paralelos VALUES (1,'A');
INSERT INTO  paralelos VALUES (2,'B');
INSERT INTO  paralelos VALUES (3,'C');
INSERT INTO  paralelos VALUES (4,'D');
```

Tabla 'pensum_materia'

```
-- Estructura para la tabla `pensum_materia`

CREATE TABLE pensum_materia (
  COD_PEN_MAT int(11) NOT NULL auto_increment,
  COD_PENSUM int(11) default NULL,
  COD_MATERIA int(11) default NULL,
  COD_NIVEL int(11) default NULL,
  PRIMARY KEY  (COD_PEN_MAT),
  KEY COD_NIVEL (COD_NIVEL),
  KEY COD_PENSUM (COD_PENSUM),
  KEY COD_MATERIA (COD_MATERIA),
  CONSTRAINT `FK_MATERIAS_15` FOREIGN KEY (`COD_MATERIA`) REFERENCES `mate-
rias` (`COD_MATERIA`),
  CONSTRAINT `FK_NIVELES_13` FOREIGN KEY (`COD_NIVEL`) REFERENCES `niveles`
(`COD_NIVEL`),
  CONSTRAINT `FK_PENSUMS_14` FOREIGN KEY (`COD_PENSUM`) REFERENCES `pensums`
(`COD_PENSUM`)
) TYPE=InnoDB;
```

Tabla 'pensums'

```
-- Estructura para la tabla `pensums`

CREATE TABLE pensums (
  COD_PENSUM int(11) NOT NULL auto_increment,
  DESCRIPCION varchar(50) NOT NULL default '',
  COD_CARRERA int(11) default NULL,
  ESTADO char(1) default 'N',
  PRIMARY KEY  (COD_PENSUM),
  KEY COD_CARRERA (COD_CARRERA),
  CONSTRAINT `FK_CARRERAS_3` FOREIGN KEY (`COD_CARRERA`) REFERENCES `carreras`
(`COD_CARRERA`)
) TYPE=InnoDB;
```

Tabla 'periodos'

```
-- Estructura para la tabla `periodos`

CREATE TABLE periodos (
  COD_PERIODO int(11) NOT NULL auto_increment,
  DESCRIPCION varchar(50) default NULL,
  FECHA_INICIO date default NULL,
  FECHA_FIN date default NULL,
  ESTADO char(1) default 'N',
  PRIMARY KEY  (COD_PERIODO)
) TYPE=InnoDB;

-- Datos de inicio para la tabla `periodos`

INSERT INTO  periodos VALUES (1,'Período Académico Septiembre 2004 - Febrero
2005','2004-09-20','2005-02-25','N');
INSERT INTO  periodos VALUES (2,'Período Académico Marzo - Agosto 2005','2005-
03-21','2005-08-12','S');
INSERT INTO  periodos VALUES (3,'Período Académico Septiembre 2005 - Febrero
2006','2005-09-26','2006-02-17','N');
```

```
INSERT INTO  periodos VALUES (4,'Período Académico Marzo - Agosto 2006','2006-
03-15','2006-08-15','N');
```

Tabla 'preguntas'

```
-- Estructura para la tabla `preguntas`

CREATE TABLE preguntas (
  COD_PREGUNTA int(11) NOT NULL auto_increment,
  COD_CATEGORIA int(11) NOT NULL default '0',
  DESCRIPCION varchar(200) NOT NULL default '',
  PRIMARY KEY  (COD_PREGUNTA),
  KEY COD_CATEGORIA (COD_CATEGORIA),
  CONSTRAINT `FK_CATEGORIAS_6` FOREIGN KEY (`COD_CATEGORIA`) REFERENCES `cate-
gorias` (`COD_CATEGORIA`)
) TYPE=InnoDB;

-- Datos de inicio para la tabla `preguntas`

INSERT INTO  preguntas VALUES (1,1,'Asiste a la actividad académica en los ho-
rarios previstos');
INSERT INTO  preguntas VALUES (2,1,'Empieza la actividad académica con puntua-
lidad');
INSERT INTO  preguntas VALUES (3,1,'Trata a los estudiantes con cortesía y
respeto.');
INSERT INTO  preguntas VALUES (4,1,'Aprovecha el tiempo sin extenderse más
allá del término estipulado para la actividad académica');
INSERT INTO  preguntas VALUES (5,1,'Hay un clima de confianza propicio a la
comunicación');
INSERT INTO  preguntas VALUES (6,2,'Estimula la participación de los estudian-
tes');
INSERT INTO  preguntas VALUES (7,2,'Desarrolla de manera coherente las activi-
dades académicas');
INSERT INTO  preguntas VALUES (8,2,'Explica los criterios de evaluación que se
emplean (pruebas, exámenes, controles de lectura , monografías, informes, mo-
delos, maquetas, proyectos)');
INSERT INTO  preguntas VALUES (9,2,'Analiza con los estudiantes las califica-
ciones asignadas en las evaluaciones (pruebas, exámenes,  controle de lectura,
monografías, informes, modelos, maquetas, proyectos)');
INSERT INTO  preguntas VALUES (10,3,'Promueve un ambiente de trabajo favorable
al aprendizaje');
INSERT INTO  preguntas VALUES (11,3,'Promueve actividades académicas como con-
ductas, investigación, salidas de campo, observaciones u otras');
INSERT INTO  preguntas VALUES (12,3,'Su nivel de exigencia incentiva al apren-
dizaje');
INSERT INTO  preguntas VALUES (13,3,'Me gustaría volver a trabajar con este
profesor en el futuro');
INSERT INTO  preguntas VALUES (14,4,'Estimula el análisis y reflexión');
INSERT INTO  preguntas VALUES (15,4,'Propicia la generación de nuevas ideas,
enfoques, alternativas, opciones, soluciones, innovaciones, creaciones');
INSERT INTO  preguntas VALUES (16,4,'Adquiero nuevos conocimientos o habilida-
des');
```

Tabla 'respuestas'

```
-- Estructura para la tabla `respuestas`

CREATE TABLE respuestas (
  COD_RESPUESTA int(11) NOT NULL auto_increment,
  COD_CAR_EST_DOC_CAR_MAT int(11) NOT NULL default '0',
  COD_CUESTIONARIO int(11) default NULL,
  CALIFICACION double default '0',
  PRIMARY KEY  (COD_RESPUESTA),
  KEY COD_CAR_EST_DOC_CAR_MAT (COD_CAR_EST_DOC_CAR_MAT),
  KEY COD_CUESTIONARIO (COD_CUESTIONARIO),
  CONSTRAINT `FK_CAR_EST_DOC_CAR_MAT_27` FOREIGN KEY
(`COD_CAR_EST_DOC_CAR_MAT`) REFERENCES `car_est_doc_car_mat`
(`COD_CAR_EST_DOC_CAR_MAT`),
  CONSTRAINT `FK_CUESTIONARIOS_28` FOREIGN KEY (`COD_CUESTIONARIO`) REFERENCES
`cuestionarios` (`COD_CUESTIONARIO`)
) TYPE=InnoDB;
```

Tabla 'respuestas_auto'

```
-- Estructura para la tabla `respuestas_auto`

CREATE TABLE respuestas_auto (
  COD_RESPUESTA_AUTO int(11) NOT NULL auto_increment,
  COD_DOC_CAR_MAT int(11) NOT NULL default '0',
  COD_CUESTIONARIO int(11) default NULL,
  CALIFICACION double default '0',
  PRIMARY KEY  (COD_RESPUESTA_AUTO),
  KEY COD_DOC_CAR_MAT (COD_DOC_CAR_MAT),
  KEY COD_CUESTIONARIO (COD_CUESTIONARIO),
  CONSTRAINT `FK_CUESTIONARIOS_26` FOREIGN KEY (`COD_CUESTIONARIO`) REFERENCES
`cuestionarios` (`COD_CUESTIONARIO`),
  CONSTRAINT `FK_DOC_CAR_MATERIA_25` FOREIGN KEY (`COD_DOC_CAR_MAT`) REFER-
ENCES `doc_car_materia` (`COD_DOC_CAR_MAT`)
) TYPE=InnoDB;
```

Tabla 'tipo_categoría'

```
-- Estructura para la tabla `tipo_categoria`

CREATE TABLE tipo_categoria (
  COD_TIPO_CATEGORIA int(11) NOT NULL auto_increment,
  DESCRIPCION varchar(15) NOT NULL default '',
  PRIMARY KEY  (COD_TIPO_CATEGORIA)
) TYPE=InnoDB;


-- Datos de inicio para la tabla `tipo_categoria`

INSERT INTO  tipo_categoria VALUES (1,'Entrada');
INSERT INTO  tipo_categoria VALUES (2,'Salida');
```

Tabla 'tipo_evaluacion'

```
-- Estructura para la tabla `tipo_evaluacion`

CREATE TABLE tipo_evaluacion (
  COD_TIPO_EVALUACION int(11) NOT NULL auto_increment,
  DESCRIPCION varchar(15) NOT NULL default '',
  PRIMARY KEY  (COD_TIPO_EVALUACION)
) TYPE=InnoDB;
```

Tabla 'usuarios'

```
-- Estructura para la tabla `usuarios`

CREATE TABLE usuarios (
  COD_USUARIO int(11) NOT NULL auto_increment,
  USUARIO varchar(30) NOT NULL default '',
  CONTRASENA varchar(15) default NULL,
  PRIMARY KEY  (COD_USUARIO)
) TYPE=InnoDB;

-- Datos de inicio para la tabla `usuarios`

INSERT INTO  usuarios VALUES (1,'usuario','contrasena');
```

3.3.2. Codificación de Cálculos de Evaluación.

3.3.2.1. Class conexion.

Esta clase realiza la conexión con la base de datos del sistema.

```
  class conexion {
    var $link;
    var $base_datos;
    var $mensaje;
    var $color;

    function hacer_conexion () {
      @$this -> link = mysql_connect ("localhost", "root");
      if (!($this -> link)) {
        $this -> mensaje = "ATENCIÓN: Se presento un error en la cadena de co-
nexión!<br>Intente más tarde...";
        $this -> color = "#FF0000";
      }
      else {
        $this -> base_datos = mysql_select_db ("evaluacion", $this -> link);
        if (!($this -> base_datos)) {
          $this -> mensaje = "ATENCIÓN: Se produjo un error al seleccionar la
base de datos!<br>Intente más tarde";
          $this -> color = "#FF0000";
        }
      }
    }
  }
```

3.3.2.2. Class difusidad.

Ejecuta los procedimientos de cálculo: obtener el promedio de determinados datos y procesar la difusidad de los valores de evaluación.

```
  class difusidad {
    var $variable;
    var $contador;
    var $promedio;
    var $resultado;

    function sacar_promedio ($parametro, $calificacion) {
      @$this -> variable [$parametro] += $calificacion;
      if ($calificacion <> 0) {
        @$this -> contador [$parametro] = $this -> contador [$parametro] + 1;
        @$this -> promedio [$parametro] = $this -> variable [$parametro] /
$this -> contador [$parametro];
      }
    }
    function sacar_difusidad ($a, $b, $u) {
      if ($a >= $u) {
        $this -> resultado = 0.0001;
      }
      else if ($b <= $u) {
        $this -> resultado = 1;
      }
      else {
        $this -> resultado = ($u - $a) / ($b - $a);
      }
    }
  }
```

3.3.2.3. Class variables.

Es una clase heredada de la clase ***conexion.*** Tiene las funciones de obtener los valores de los rangos de evaluación, categorías y la observación correspondiente a cada estado resultante del proceso de evaluación.

```
class variables extends conexion {
    var $max;
    var $min;
    var $a;
    var $b;

    function sacar_valores ($cod_categoria, $cod_tipo_categoria = 2) {
      $this -> hacer_conexion ();
      $sql = "SELECT valor_max, valor_a, valor_b ";
      $sql .= "FROM categorias ";
      $sql .= "WHERE cod_categoria = $cod_categoria ";
      $result = mysql_query ($sql, $this -> link);
      if (@$row = mysql_fetch_array($result)) {
        $this -> a = $row ['valor_a'];
        $this -> b = $row ['valor_b'];
        $this -> max = $row ['valor_max'];
      }
      else if ($cod_tipo_categoria == 2){
        $this -> a = 15;  //valores por defecto
        $this -> b = 95;  //valores por defecto
        $this -> max = 100;  //valores por defecto
      }
      else {
        $this -> a = 1.5;  //valores por defecto
        $this -> b = 4.5;  //valores por defecto
        $this -> max = 5;  //valores por defecto
      }
      mysql_close ($this -> link);
    }
    function sacar_cod_categoria () {
      $this -> hacer_conexion ();
      $sql = "SELECT cod_categoria ";
      $sql .= "FROM categorias ";
      $sql .= "WHERE cod_tipo_categoria = 2 ";
      $result = mysql_query ($sql, $this -> link);
      if (@$row = mysql_fetch_array($result)) {
        $cod_categoria = $row ['cod_categoria'];
      }
      else {
        $cod_categoria = 0;
      }
      mysql_close ($this -> link);
      return $cod_categoria;
    }
    function sacar_msn_observacion ($estado) {
      $this -> hacer_conexion ();
      $sql = "SELECT estado, descripcion ";
      $sql .= "FROM observaciones ";
      $sql .= "WHERE estado = '$estado' ";
      $result = mysql_query ($sql, $this -> link);
      if (@$row = mysql_fetch_array($result)) {
        $this -> msn_estado = $row ['estado'];
        $this -> msn_observacion = $row ['descripcion'];
      }
      else {
        $this -> msn_observacion = "SIN RANGO";
      }
      mysql_close ($this -> link);
    }
  }
```

3.3.2.4. Class centro_gravedad.

Es la encargada de obtener el centro de gravedad de la difusidad obtenida del promedio de las evaluaciones realizadas y transformarlos a valores que se encuentran en rangos de evaluación definidos en la clase ***observacion***.

```
class centro_gravedad extends variables {
  var $resultado;

  function obtener_cg ($u) {
    if (!($cod_categoria = $this -> sacar_cod_categoria())) {
      $cod_categoria = 0;
    }
    $this -> sacar_valores ($cod_categoria);
    $base_tri = $u * ($this -> b - $this -> a);
    $base_rec = $this -> max - $this -> a - $base_tri;
    $altura_tri = $u;
    $altura_rec = $u;
    $cg_tri = 2 / 3 * $base_tri + $this -> a;
    $cg_rec = 1 / 2 * $base_rec + $this -> a + $base_tri;
    $area_tri = $base_tri * $altura_tri / 2;
    $area_rec = $base_rec * $altura_rec;
    $area_x_cg_tri = $cg_tri * $area_tri;
    $area_x_cg_rec = $cg_rec * $area_rec;
    $E_area_x_cg = $area_x_cg_tri + $area_x_cg_rec;
    $E_area = $area_tri + $area_rec;
    $this -> resultado = $E_area_x_cg / $E_area;
  }
}
```

3.3.2.5. Class auto_cg.

Constructor de la clase ***centro_gravedad,*** genera el valor por defecto de esta clase.

```
class auto_cg extends centro_gravedad { //constructor
  function auto_cg () {
    $this -> obtener_cg (1);
  }
}
```

3.3.2.6. Class evaluacion.

Heredada de la clase ***variables***, es la encargada de corregir el valor resultado del proceso de evaluación.

```
class evaluacion extends variables {
  var $valor_cg_corregido;

  function correccion_cg ($u) {
    $cg = new centro_gravedad;
    $cg -> obtener_cg($u);
```

```
      if (!($cod_categoria = $this -> sacar_cod_categoria())) {
        $cod_categoria = 0;  //valores por defecto
      }
      $this -> sacar_valores ($cod_categoria);
      $auto_cg = new auto_cg;
      $this -> valor_cg_corregido = $cg -> resultado + ($this -> max - $au-
to_cg -> resultado) * $u;
      $this -> valor_cg_corregido = $this -> valor_cg_corregido / 2;
    }
  }
```

3.3.2.7. Class observacion.

Emite la observación correspondiente de cada estado resultante a cada valor obtenido por el proceso de evaluación.

```
  class observacion extends evaluacion {
    var $msn_observacion;
    var $msn_estado;
    var $color;

    function sacar_observacion($evaluacion) {
      $this -> correccion_cg (0.9);
      if ($evaluacion > $this -> valor_cg_corregido) {
        $this -> sacar_msn_observacion ("destacado"); //destacado
        $this -> color = "#009900";
      }
      else {
        $this -> correccion_cg (0.65);
        if ($evaluacion > $this -> valor_cg_corregido) {
          $this -> sacar_msn_observacion ("competente"); //competente
          $this -> color = "#336699";
        }
        else {
          $this -> correccion_cg (0.4);
          if ($evaluacion > $this -> valor_cg_corregido) {
            $this -> sacar_msn_observacion ("básico"); //basico
            $this -> color = "#FF9900";
          }
          else {
            $this -> correccion_cg (0.166);
            if ($evaluacion > $this -> valor_cg_corregido) {
              $this -> sacar_msn_observacion ("regular"); //regular
              $this -> color = "#FA3058";
            }
            else {
              $this -> sacar_msn_observacion ("insatisfactorio"); //insatisf…
              $this -> color = "#FF0000";
            }
          }
        }
      }
    }
  }
```

3.4. Pruebas e Instalación.

3.4.1. Pruebas.

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

3.4.2. Instalación.

Contenido del disco de Instalación: El disco de instalación del sistema SED contiene:

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

Pasos para la Instalación del Sistema.

- Como primer paso, procedemos con la instalación del compilador de PHP obligatorio que el sistema de evaluación necesita. Instalamos la versión de PHP 4.2.2 con el programa de instalación del mismo (si en el sistema operativo previamente no se encuentra instalado o existe una versión inferior). Luego, modificamos el archivo **"php.ini"** que se instaló en el sistema operativo, con la finalidad de permitir el paso de variables entre formularios; seteamos en **"On"** la opción "**register_globals"** y grabamos la modificación.

```
register_globals = On
```

- Ahora instalamos el motor de base de datos MySQL. Instalamos la versión de MySQL 4.1.14 con el programa de instalación del mismo (si en el sis-

tema operativo previamente no se encuentra instalado o existe una versión inferior).

- Grabamos el sitio Web que contiene los archivos del sistema en el directorio respectivo:

  - Windows. "c:/inetpub/wwwroot/evaluacion".

  - Solaris. "/home/usuario/evaluacion".

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

  - Windows. "c:/mysql/bin".

  - Solaris. "/usr/local/mysql/bin".

Por último ingresamos a MySQL, creamos la base de datos con el nombre **"evaluacion"** y cargamos el archivo **"evaluacion.dump"**.

```
shell> mysql
mysql> CREATE DATABASE evaluacion;
mysql> USE evaluacion;
mysql> SET FOREIGN_KEY_CHECKS=0;
mysql> SOURCE evaluacion.dump;
```

Importante. El nombre de la base de datos debe ser idéntico al que se indica. Antes de cargar la base de datos es necesario setear en 0 la verificación de claves foráneas que tiene MySQL.

4. CAPITULO IV. Verificación Y Validación de Resultados.

4.1. Verificación.

La hipótesis planteada fue que con la utilización del Sistema de Evaluación Docente, se logrará sistematizar las evaluaciones de todos los docentes de la PUCESA en tiempo real.

En el período de evaluación efectuado del 16 al 22 de Enero de 2006 correspondiente al período académico Septiembre 2005 - Febrero 2006 mediante el SED, se comprobó la hipótesis planteada, llegando aún más lejos de los objetivos iniciales. Los resultados de las evaluaciones realizadas se los consiguió sin ningún inconveniente, desarrollándose de forma muy satisfactoria la puesta en marcha del sistema piloto de evaluación docente en la universidad.

El sistema reveló un ahorro de tiempo considerable, sin el SED este proceso tardaba 15 días aproximadamente, con el SED en funcionamiento los resultados se obtuvieron en tiempo real.

Los informes resultantes de este proceso ayudarán a Prorectorado, Dirección Académica y Directores de Escuelas a tomar decisiones para el mejoramiento académico del personal docente de la universidad.

4.2. Validación.

4.3. Conclusiones.

- La utilización de principios de Inteligencia Artificial - Lógica Difusa en el desarrollo de la aplicación refuerza los resultados del proceso de evaluación docente, transformándolos en una evaluación más cerca al lenguaje natural del hombre y consiguiendo una evaluación de forma cualitativa.

- Se logró la automatización en el plan piloto de evaluación docente en el período académico Septiembre 2005 - Febrero 2006.

- Los resultados del proceso de evaluación se consiguieron en tiempo real sin ningún inconveniente.

- Se obtuvo un ahorro de tiempo en el proceso de cálculo de los diferentes resultados de las evaluaciones que se realizaban de forma manual sin el sistema.

- Existe la facilidad de manejo por parte de los docentes y estudiantes, así como de los usuarios administradores del sistema para el mantenimiento del mismo.

- Buscando favorecer al desarrollo del profesorado de la universidad en su labor profesional y actividades académicas se consideró al Sistema de Evaluación Docente una herramienta cualitativa valiosa en el proceso de evaluación

4.4. Recomendaciones.

- Para el buen entendimiento de los resultados de evaluación, y poder brindar cualquier recomendación al profesorado, es aconsejable revisar el detalle de cada una de las evaluaciones realizadas y compararlas con la autoevaluación del docente.

- 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.

- Es importante mantener desactivado el sistema de evaluación mientras la universidad no se encuentre en proceso de evaluación con el fin de mantener la base de datos del sistema restringida a los usuarios que no intervienen en el mantenimiento de la base y solo utilizan el sistema para realizar el proceso de evaluación.

- Es necesario que el administrador del sitio Web y la base de datos del sistema revise el manual técnico, diccionario de datos, mapa del sitio y procesos de instalación que este documento contiene con el propósito de mantener la integridad del sistema y poder dar mantenimiento si es necesario.

- 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.

ANEXOS.

A. Manual Técnico

Ámbito Y Funciones

ÁMBITO

El Sistema de Evaluación Docente esta dirigido a la PUCESA y tomará el nombre de SED.

PROBLEMAS A SOLUCIONAR

- Automatizar el proceso de evaluación docente en la universidad.
- Brindará los resultados de evaluaciones y autoevaluaciones realizadas en tiempo real.
- Ofrecerá facilidad de manejo y mantenimiento para los diferentes tipos de usuarios.

FUNCIONES

Proceso de Evaluación.

Es la agrupación de rutinas y procedimientos que hacen posible que el proceso automatizado de evaluación docente por el SED se realiza correctamente.

En el proceso de evaluación participan varios actores, principalmente los docentes que van a ser evaluados y los estudiantes que los evaluarán; también se encuentran los administradores del sistema que revisaran los resultados del proceso dependiendo de un nivel de usuario (Administrador o Escuela). El administrador del sistema creará los cuestionarios de evaluación que van a ser llenados por los estudiantes cada período académico en el período de evaluación que la universidad crea conveniente. Los resultados del proceso de eva-

luación se procesarán en tiempo real almacenando en la base de datos del sistema cada cuestionario de evaluación ingresados por los estudiantes.

Asignación de Materias a Docentes

Seleccionando una carrera, nivel y paralelo, el sistema permite de manera fácil la asignación de un docente a cada materia. El sistema indica en un combo box los docentes que se encuentran ingresados en el sistema tomando en consideración la carrera a la que pertenecen y si este se encuentra activo o no.

Administración de Docentes y Estudiantes

El SED permite ingresar, modificar y eliminar docentes y estudiantes. Esta función también es la encargada de realizar el proceso de asignación de materias a cada estudiante.

Creación de Cuestionarios de Evaluación

El sistema no trabaja con cuestionarios estáticos, los cuestionarios son dinámicos y se pueden crear nuevas preguntas o modificar las existentes. Las preguntas se agrupan en categorías las cuales también se encuentran sujetas a actualización.

Generación de Reportes

El SED proporciona al administrador y escuelas reportes e informes de los resultados de evaluación y autoevaluación dependiendo del criterio de búsqueda deseado, estos pueden ser por: docente, materias, carreras o períodos académicos.

Control del Sistema

Permite controlar de forma fácil la activación del SED. Los docentes y estudiantes solo podrán ingresar al sistema cuando este se encuentre activo.

Control de Usuarios

El sistema clasifica en cuatro tipos de usuarios: administrador, escuelas, docentes y estudiante. Cada uno de ellos tiene acceso a respectivas funciones del sistema. Los administradores tienen la capacidad de dar mantenimiento a la base de datos del sistema, procesos de administración y de evaluación; los usuarios escuelas tienen acceso solo a los datos referentes a cada escuela. Los docentes y estudiantes ingresan al sistema solo para realizar el proceso de evaluación y autoevaluación respectivamente, siempre que el sistema se encuentre activo.

Mantenimiento Base de Datos

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

SRS

1. INTRODUCCIÓN

1.1. OBJETIVO

1.1.1. PROPÓSITO DEL DOCUMENTO

Este documento tiene la misión de dar a conocer los alcances y delimitaciones que el sistema de evaluación docente tiene al ser implementado en la universidad. Incluye datos técnicos para el administrador del front end y back end del sistema (instalación del sistema y respaldo de datos).

1.1.2. AUDIENCIA A LA QUE VA DIRIGIDO

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

1.2. ALCANCE

1.2.1. IDENTIFICACIÓN DEL PRODUCTO MEDIANTE UN NOMBRE

El sistema de evaluación docente para el presente documento se lo denomina **SED**.

1.2.2. QUÉ HACE Y QUÉ NO HACE EL PRODUCTO

Sistematizará el proceso de evaluación de la PUCESA. El SED brinda las posibilidades de mantener una integridad de datos en la creación, modificación y actualización de las base de datos. Genera reportes e informes dependiendo del criterio de búsqueda deseado. *Ver también Ámbito y Funciones.*

El sistema no realiza el respaldo de la base de datos de forma automática ya que este proceso es necesario ejecutarlo de forma manual por el administrador encargado del hospedaje del sistema. *Ver también Operaciones de Backup y recuperación.*

1.2.3. APLICACIONES DEL SOFTWARE: BENEFICIOS, OBJETIVOS Y METAS

Los beneficios y objetivos que presenta el sistema básicamente son de automatizar y sistematizar los procesos de evaluación docente, organizando y

simplificando los cálculos e ingresos de datos manuales que se venían realizando en la universidad, procesando los resultados en tiempo real.

1.3. DEFINICIÓN, ACRÓNIMOS Y ABREVIATURAS

SED → Sistema de Evaluación Docente

1.4. REFERENCIAS

[IEEE 93] IEEE Std 830 1993 (revision of IEEE Std 830 1984). Software engineering Standards Committee of the IEEE Computer Society.

1.5. VISIÓN GENERAL

1.5.1. DESCRIPCIÓN DEL CONTENIDO DEL RESTO DEL DOCUMENTO

Este documento estará enfocado para el correcto funcionamiento del sistema.

1.5.2. ORGANIZACIÓN DEL DOCUMENTO

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

2. DESCRIPCIÓN GENERAL

2.1. PERSPECTIVA DEL PRODUCTO.

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

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

2.1.2. INTERFASES DE SISTEMAS.

Interfase Web interpretada por cualquier browser del lado del cliente.

2.1.3. LIMITACIONES DE MEMORIA.

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

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

2.1.4. OPERACIONES.

2.1.4.1. MODO DE OPERACIÓN DE LOS DISTINTOS GRUPOS DE USUARIOS.

Administradores. Son los encargados del mantenimiento de la base de datos del sistema, administración de usuarios y acceso a todas las escuelas.

Escuelas. Acceso de datos que se refieres exclusivamente a su escuela.

Docentes. Realizan el proceso de autoevaluación docente y tiene acceso a revisas los resultados de las evaluaciones realizadas en sus respectivas materias que se encuentre asignados.

Estudiantes. Solo realizan el proceso de evaluación docente.

2.1.4.2. FUNCIONES RESPALDOS DEL PROCESAMIENTO DE DATOS.

El administrador del hospedaje del sistema es el encargado de realizar las funciones de respaldo de datos de forma manual. *Ver también Operaciones de Backup y Recuperación.*

2.1.4.3. OPERACIONES DE BACKUP Y RECUPERACIÓN.

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

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

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

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

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

2.1.5. REQUERIMIENTOS PARA ADAPTARSE A LA UBICACIÓN.

2.1.5.1. DATOS O SECUENCIAS DE INICIALIZACIÓN ESPECÍFICOS DE CUALQUIER LUGAR, MODO DE OPERACIÓN.

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

2.1.5.2. CARACTERÍSTICAS QUE DEBEN SER MODIFICADAS PARA UNA INSTALACIÓN EN PARTICULAR.

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

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

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

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

Las paginas Web que se deberían modificar son: ***conexion.php*** y ***clases.php***.

2.2. FUNCIONES DEL PRODUCTO.

- Proceso de Evaluación.
- Asignación de Materias a Docentes
- Administración de Docentes y Estudiantes
- Creación de Cuestionarios de Evaluación
- Generación de Reportes
- Control del Sistema
- Control de Usuarios
- Mantenimiento Base de Datos

Ver también Ámbito – Funciones (Pág. 150).

2.3. CARACTERÍSTICAS DEL USUARIO.

Dependiendo el tipo de usuario, estos deben tener conocimiento de manejo, control y administración del sistema.

2.4. RESTRICCIONES.

- El sistema solo podrá realizar un proceso de evaluación por período académico.
- Ya que el sistema fue creado con principios de Inteligencia artificial - Lógica Difusa, el proceso de cálculo de evaluación no puede ser modificado.

2.5. SUPOSICIONES Y DEPENDENCIAS.

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

2.6. REQUISITOS PARA FUTURAS VERSIONES DEL SISTEMA.

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

3. REQUISITOS ESPECÍFICOS

3.1. REQUISITOS DE INTERFASE EXTERNO.

3.1.1. INTERFASES DE USUARIO.

La interfase del usuario es una interfase Web que cualquier browser interpretara. Esta consta de formularios, menú de usuario, navegador y buscador interno del sistema y pantalla de ayuda de todas las funciones para los usuarios administradores y escuelas.

3.1.2. INTERFASES DE SOFTWARE.

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

3.2. REQUISITOS FUNCIONALES.

Windows.

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

Solaris.

- PHP versión 4.2.2
- MySQL versión 4.1.14
- Apache

3.2.1. FLUJOS DE INFORMACIÓN.

3.2.1.1. DIAGRAMA DE FLUJO DE DATOS NIVEL 0.

Ver diagrama de flujo de Dato Nivel 0 del Capítulo 3 del Desarrollo de Proyecto (Pág. 76).

3.2.1.2. DIAGRAMA DE FLUJO DE DATOS NIVEL 1.

Ver diagrama de flujo de Dato Nivel 1 del Capítulo 3 del Desarrollo de Proyecto (Pág. 77).

3.2.1.3. DIAGRAMA DE FLUJO DE DATOS NIVEL 2.

Ver diagrama de flujo de Dato Nivel 2 del Capítulo 3 del Desarrollo de Proyecto (Pág. 78).

3.3. REQUISITOS DE RENDIMIENTOS.

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

3.4. RESTRICCIONES DE DISEÑO.

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

B. Manual de Usuario.

INTRODUCCION

Esta guía esta pensada como instructivo a la utilización del Sistema Evaluación Docente de la PUCESA para aquellas personas que vayan a hacer uso de este. Todas las secciones le ayudarán a manejar mejor el sistema.

El proceso de Evaluación Docente es generado mediante página Web y los resultados son publicados en tiempo real. El sistema de Evaluación Docente se encuentra hospedado en el servidor Web de la PUCESA y podemos acceder de una manera fácil a través de su página: www.pucesa.edu.ec y haciendo "clic" en el vínculo de Evaluación Docente.

INGRESO AL SISTEMA

El sistema permite ingresar a 4 tipos de usuarios:

INGRESO COMO ESTUDIANTE

En la pantalla principal del Sistema de Evaluación Docente damos "clic" en el botón para el ingreso de estudiantes.



En la siguiente pantalla es necesario que el estudiante ingrese su número de cédula (con guión) para la verificación de datos del estudiante por el Sistema de Evaluación Docente.



El sistema mostrará un mensaje de "Atención" si el sistema se encuentra desactivado o la cédula se encuentra mal ingresada.

INGRESO COMO DOCENTE

En la pantalla principal del Sistema de Evaluación Docente damos "clic" en el botón para el ingreso de docentes.



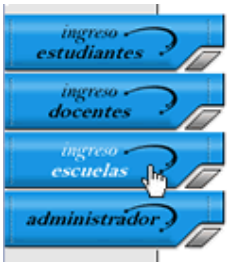
En la siguiente pantalla es necesario que el docente ingrese su número de cédula (con guión) para la verificación de datos del docente por el Sistema de Evaluación Docente.



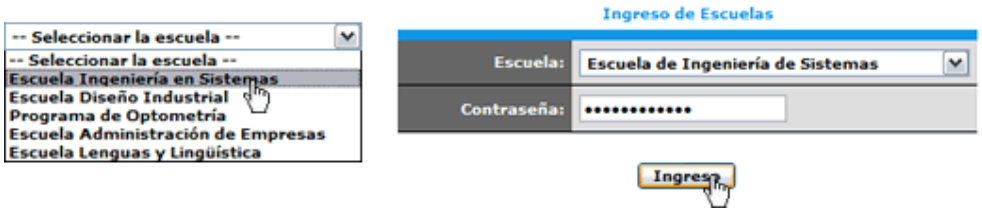
El sistema mostrará un mensaje de "Atención" si el sistema se encuentra desactivado o la cédula se encuentra mal ingresada.

INGRESO COMO ESCUELAS

En la pantalla principal del Sistema de Evaluación Docente damos "clic" en el botón para el ingreso de escuelas.



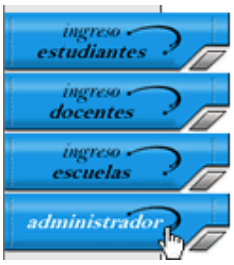
En la siguiente pantalla es necesario escoger una escuela de las existentes en el combo box donde nos aparece una leyenda que dice: "-- Seleccionar la escuela --" e ingresar su respectiva contraseña.



El sistema mostrará un mensaje de "Atención" si no se ha escogido una escuela o si la contraseña se encuentra mal ingresada.

INGRESO COMO ADMINISTRADOR

En la pantalla principal del Sistema de Evaluación Docente damos "clic" en el botón para el ingreso de los usuarios administrador.



En la siguiente pantalla es necesario que el administrador ingrese su nombre de usuario y su contraseña.



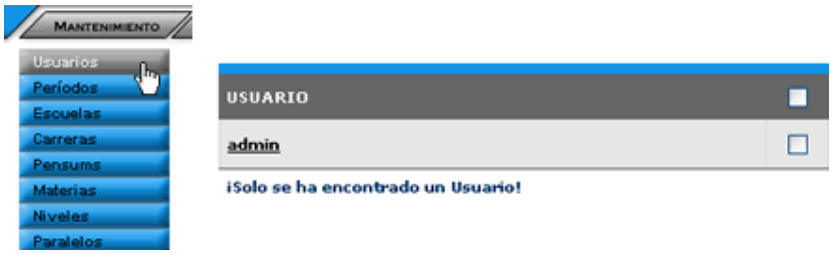
El sistema mostrará un mensaje de "Atención" si el usuario administrador o contraseña se encuentran mal ingresados.

MANTENIMIENTO

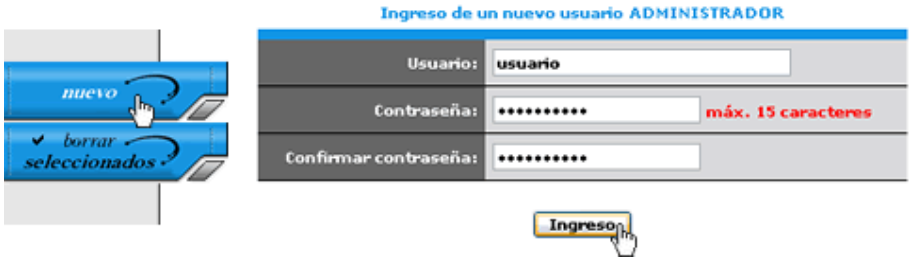
Para dar mantenimiento a la base de datos el Sistema de Evaluación Docente se encuentra clasificado en:

USUARIOS

En el menú "Mantenimiento", seleccionamos la opción de "Usuarios" para poder observar y dar mantenimiento a los usuarios existentes en el sistema.

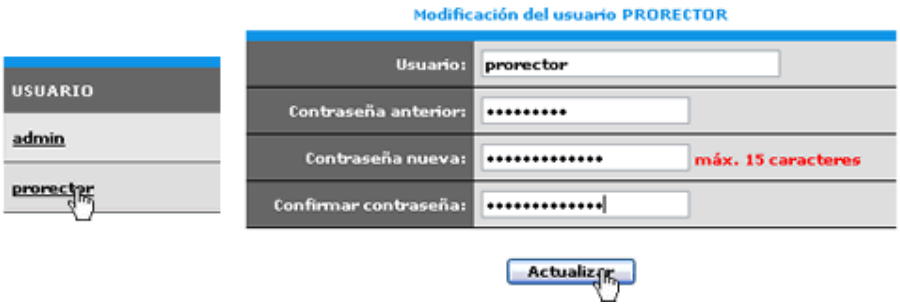


Creación de Usuarios. Si queremos ingresar un nuevo usuario damos "clic" en el botón "Nuevo", este nos enviará a un formulario de ingreso de un nuevo usuario.

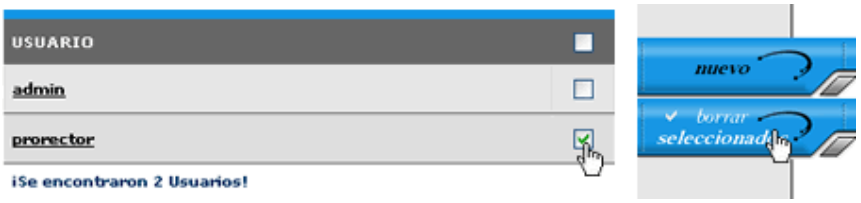


Llenamos los datos del formulario y damos "clic" en botón de ingreso, el sistema nos indicará si se ha realizado correctamente el ingreso del nuevo usuario o un mensaje de alerta si se encontraron datos mal ingresados.

Modificación de Usuarios. Para la actualización de datos de un usuario es necesario dar "clic" en el usuario a modificarse; este nos enviará a un formulario indicándonos los datos actuales del usuario.



Eliminación de Usuarios. Si queremos realizar esta operación, es necesario seleccionar el usuario a eliminarse y dar "clic" en el botón "Borrar seleccionados".



El sistema nos indicará si la eliminación se realizó correctamente o no.

PERÍODOS

En el menú "Mantenimiento", seleccionamos la opción de "Períodos" para poder observar y dar mantenimiento a los períodos académicos existentes en el sistema.

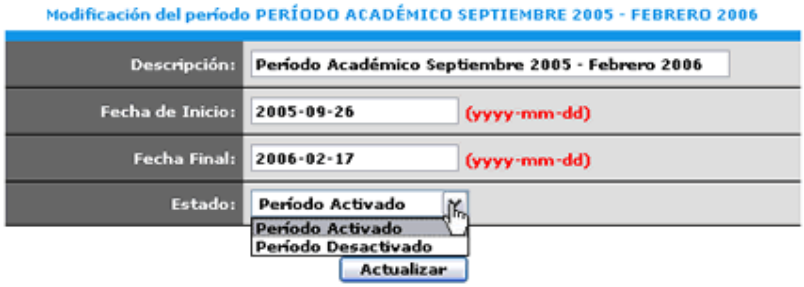


Creación de Períodos. Si queremos ingresar un nuevo período académico damos "clic" en el botón "Nuevo", este nos enviará a un formulario de ingreso de un nuevo período.



Ingresamos una descripción para el nuevo período académico, la fecha de inicio y final del período con el formato que nos indica el formulario (yyyy-mm-dd), año, mes, día; y damos "clic" en botón de ingreso, el sistema nos indicará si se ha realizado correctamente el ingreso del nuevo período o un mensaje de alerta si se encontraron datos mal ingresados.

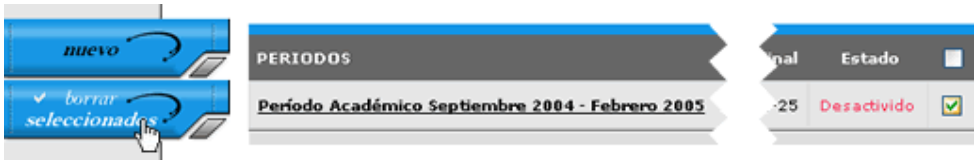
Importante. Al ingresar un nuevo período, este se crea por defecto como "Período Desactivado"; para activarlo es necesario seguir las instrucciones de "Modificación de Períodos".



Modificación de Períodos. Si queremos activar o modificar las fechas de un período es necesario dar "clic" en el período académico a modificarse; este nos enviará a un formulario indicándonos los datos actuales del período.

Importante. El sistema solo permite que exista un solo período académico activado a la vez; por consecuente al activar un período, se desactivará automáticamente el anterior período que se encontraba activado, y el sistema trabajará con el nuevo período activado.

Eliminación de Períodos. Si queremos realizar esta operación, es necesario seleccionar el período académico a eliminarse y dar "clic" en el botón "Borrar seleccionados".



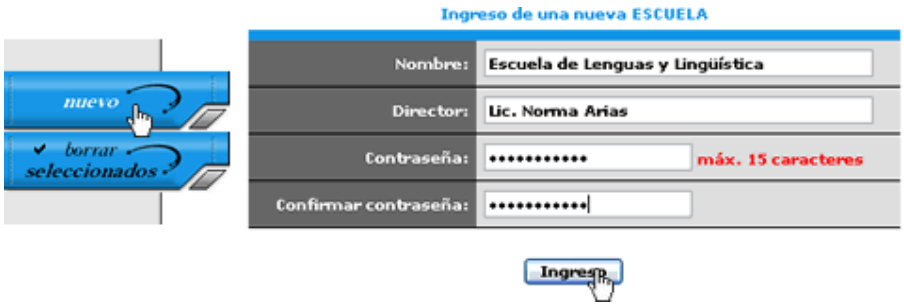
El sistema nos indicará si la eliminación se realizó correctamente o no.

ESCUELAS

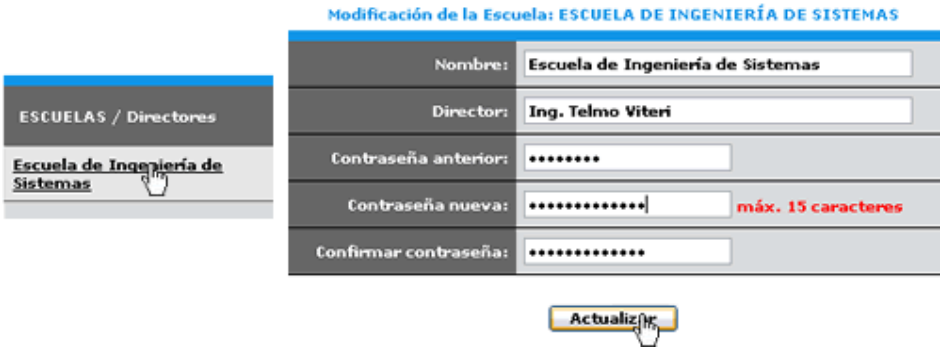
En el menú "Mantenimiento", seleccionamos la opción de "Escuelas" para poder observar y dar mantenimiento a las escuelas existentes en el sistema.



Creación de Escuelas. Si queremos ingresar una nueva escuela damos "clic" en el botón "Nuevo", este nos enviará a un formulario de ingreso de una nueva escuela.



Modificación de Escuelas. Para la actualización de datos de una escuela es necesario dar "clic" en la escuela a modificarse; esta nos enviará a un formulario indicándonos los datos actuales de la escuela.

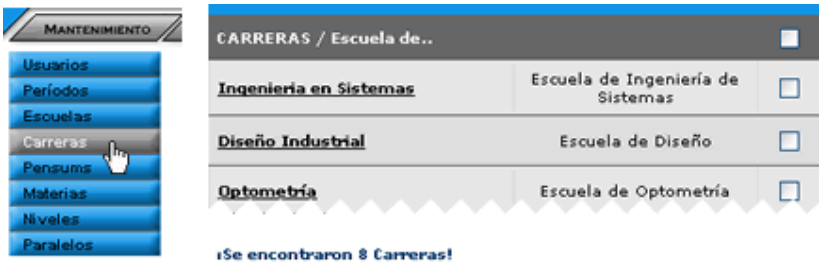


Eliminación de Escuelas. Si queremos realizar esta operación, es necesario seleccionar la escuela a eliminarse y dar "clic" en el botón "Borrar seleccionados".

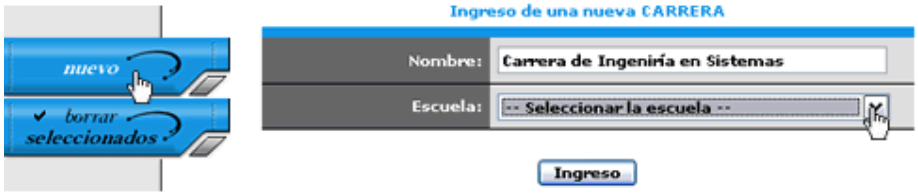
Importante. El sistema solo permite que se eliminen las escuelas que no tengan relación alguna con otro dato, es decir, si existe alguna carrera que pertenezca a la escuela a eliminarse, el sistema indicará un mensaje de "Atención" indicando que no se pudo eliminar.

CARRERAS

En el menú "Mantenimiento", seleccionamos la opción de "Carreras" para poder observar y dar mantenimiento a las carreras existentes en el sistema.

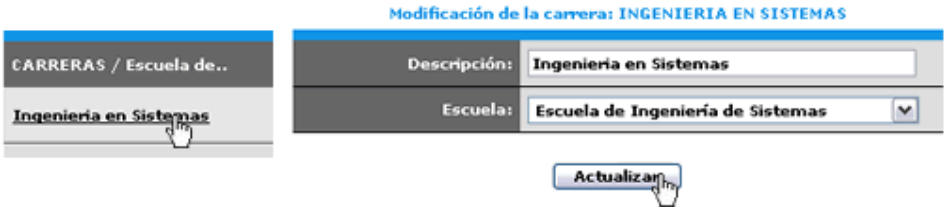


Creación de Carreras. Si queremos ingresar una nueva carrera damos "clic" en el botón "Nuevo", este nos enviará a un formulario de ingreso de una nueva carrera.

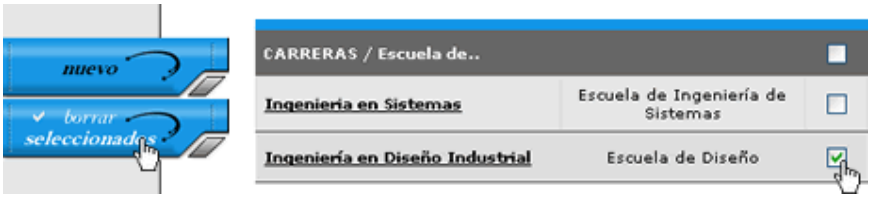


Ingresamos un nombre que identifique la carrera y seleccionamos la escuela a la que pertenece. Damos "clic" en ingresar.

Modificación de Carreras. Para la actualización de datos de una carrera damos "clic" en la carrera a modificarse; esta nos enviará a un formulario indicándonos los datos actuales de la carrera.



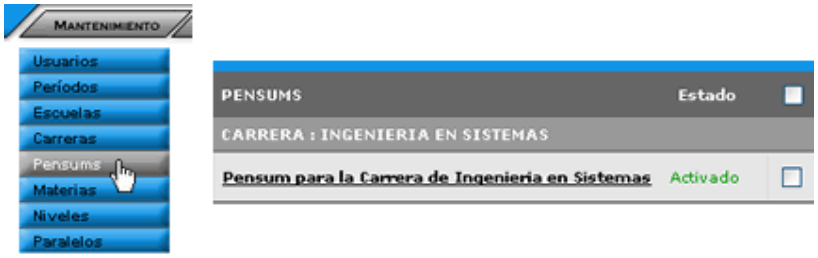
Eliminación de Carreras. Si queremos realizar esta operación, es necesario seleccionar la carrera a eliminarse y dar "clic" en el botón "Borrar seleccionados".



Importante. El sistema solo permite que se eliminen las carreras que no tengan relación alguna con otro dato, es decir, si existe alguna materia perteneciente a la carrera a eliminarse, el sistema indicará un mensaje de "Atención" indicando que no se pudo eliminar.

PENSUMS

En el menú "Mantenimiento", seleccionamos la opción de "Pénsums" para poder observar y dar mantenimiento a los pénsums existentes en el sistema.



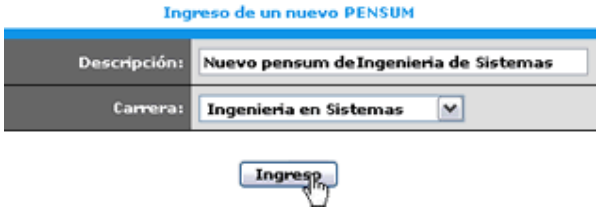
La forma de acceso a la pantalla de "Pénsums" a través del menú de "Mantenimiento" puede variar dependiendo del tipo de usuario.

Importante. Puede existir más de un pénsum por carrera pero solo uno de ellos puede estar activo a la vez.

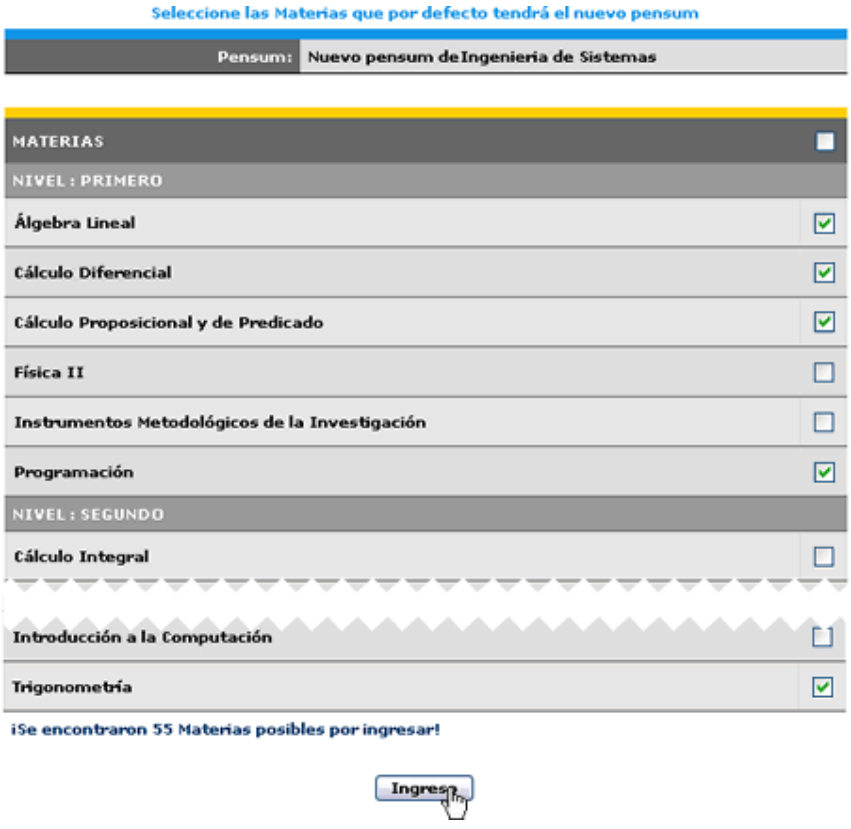
Creación de Pénsums. Si queremos crear un nuevo pénsum damos "clic" en el botón "Nuevo", este nos enviará a un formulario, donde debemos seguir los siguientes pasos:



Paso 1. Es necesario ingresar una descripción y escoger una carrera para el nuevo pénsum, presionamos "Ingreso".



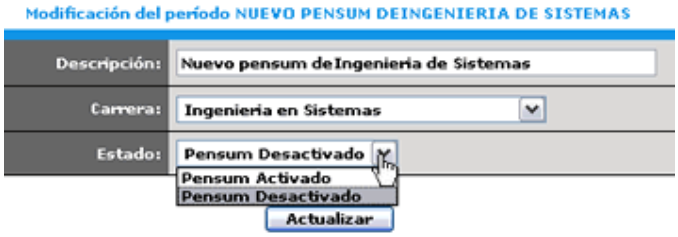
Paso 2. Si la carrera seleccionada tenía creado algún pénsum, el sistema nos brinda la posibilidad de ingresar las materias que por defecto tendrá nuestro nuevo pénsum de un listado de materias del pénsum anterior ordenado por niveles. Seleccionamos todas las materias que deseamos que tenga nuestro nuevo pénsum y pulsamos "Ingreso".



Listo, se ha creado un nuevo pénsum con las materias que por defecto seleccionamos. Si más tarde necesitamos ingresar nuevas materias a nuestro nuevo pénsum o modificar las materias existentes debemos guiarnos con los pasos del menú de ayuda de "Materias".

Importante. Al ingresar un nuevo pénsum, este se crea por defecto como "Pénsum Desactivado"; para activarlo es necesario seguir las instrucciones de "Modificación de Pénsums".

Modificación de Pénsums. Si queremos activar o modificar los datos de un pénsum es necesario dar "clic" en el pénsum a modificarse; este nos enviará a un formulario indicándonos los datos actuales del pénsum y su estado.



Importante. El sistema solo permite que exista un solo pénsum activado a la vez; por consecuente al activar un pénsum, se desactivará automáticamente el anterior pénsum que se encontraba activado, y el sistema trabajará con el nuevo pénsum activado.

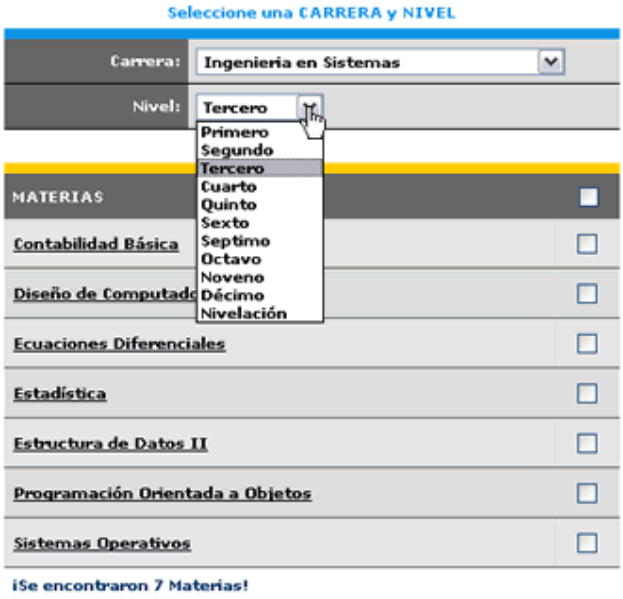
Eliminación de Pénsums. Si queremos realizar esta operación, es necesario seleccionar el pénsum a eliminarse y dar "clic" en el botón "Borrar seleccionados".

Importante. El sistema solo permite que se eliminen los pénsums que no tengan relación alguna con otro dato, es decir, si existe alguna materia perteneciente al pénsum a eliminarse, el sistema indicará un mensaje de "Atención" indicando que no se pudo eliminar.

MATERIAS

En el menú "Mantenimiento", seleccionamos la opción de "Materias" para poder observar y dar mantenimiento a las materias existentes en el sistema.

Seleccionamos la carrera y el nivel por los que deseamos realizar la búsqueda de materias.



Creación de Materias. Si queremos ingresar una nueva materia en una carrera y nivel determinados, damos "clic" en el botón "Nuevo", este nos enviará a un formulario de ingreso de una nueva materia.

Ingresamos un nombre para la nueva materia, escogemos una carrera y un nivel y presionamos "Ingresar".



Modificación de Materias. Para la actualización de datos de una materia damos "clic" en la materia a modificarse; esta nos enviará a un formulario indicándonos los datos actuales de la materia.



Eliminación de Materias. Si queremos realizar esta operación, es necesario seleccionar la materia a eliminarse y dar "clic" en el botón "Borrar seleccionados".

Importante. El sistema solo permite que se eliminen las materias que no tengan relación alguna con otro dato, es decir, si existe algún docente asignado a la materia a eliminarse, el sistema indicará un mensaje de "Atención" indicando que no se pudo eliminar.

NIVELES

En el menú "Mantenimiento", seleccionamos la opción de "Niveles" para poder observar y dar mantenimiento a los niveles existentes en el sistema.

Creación de Niveles. Si queremos ingresar un nuevo nivel damos "clic" en el botón "Nuevo", este nos enviará a un formulario de ingreso de un nuevo nivel.

Modificación de Niveles. Para la actualización de datos de un nivel es necesario dar "clic" en el nivel a modificarse; esta nos enviará a un formulario indicándonos los datos actuales del nivel.

Eliminación de Niveles. Si queremos realizar esta operación, es necesario seleccionar el nivel a eliminarse y dar "clic" en el botón "Borrar seleccionados".

Importante. El sistema solo permite que se eliminen los niveles que no tengan relación alguna con otro dato, es decir, si existe alguna materia que pertenezca al nivel a eliminarse, el sistema indicará un mensaje de "Atención" indicando que no se pudo eliminar.

PARALELOS

En el menú "Mantenimiento", seleccionamos la opción de "Paralelos" para poder observar y dar mantenimiento a los paralelos existentes en el sistema.

Creación de Paralelos. Si queremos ingresar un nuevo nivel damos "clic" en el botón "Nuevo", este nos enviará a un formulario de ingreso de un nuevo paralelo.

Modificación de Paralelos. Para la actualización de datos de un paralelo es necesario dar "clic" en el paralelo a modificarse; esta nos enviará a un formulario indicándonos los datos actuales del paralelo.

Eliminación de Paralelos. Si queremos realizar esta operación, es necesario seleccionar el paralelo a eliminarse y dar "clic" en el botón "Borrar seleccionados".

Importante. El sistema solo permite que se eliminen los paralelos que no tengan relación alguna con otro dato, es decir, si existe alguna materia que pertenezca al paralelo a eliminarse, el sistema indicará un mensaje de "Atención" indicando que no se pudo eliminar.

PROCESOS

Para el ingreso de docentes y estudiantes así como la asignación de materias a los docentes, el Sistema de Evaluación Docente cuenta con los siguientes procedimientos:

ADMINISTRACIÓN DE DOCENTES

Para acceder a la pantalla de administración de docentes debemos seleccionar del menú de "Procesos" la opción "Administración de Docentes".



A continuación se indican las partes en las que se encuentra constituida la pantalla de Administración de Docentes:

ⓐ **ComboBox de Carreras,** contiene todas las carreras existentes en el sistema a las que determinado usuario tiene acceso.

ⓑ **Botones de Mantenimiento,** tienen las funciones de creación y eliminación de los docentes del sistema.

ⓒ **Página de Docentes,** muestra los docentes existentes en la carrera seleccionada, con el número de cédula y su estado (Activo Si/No).

ⓓ **Número de Docentes encontrados,** indica el número total de docentes que se encontró en la base de datos del sistema.

ⓔ **Navegador,** tienen la función de navegar entre páginas (primera, anterior, siguiente, última). *Ver también Uso del Navegador*.

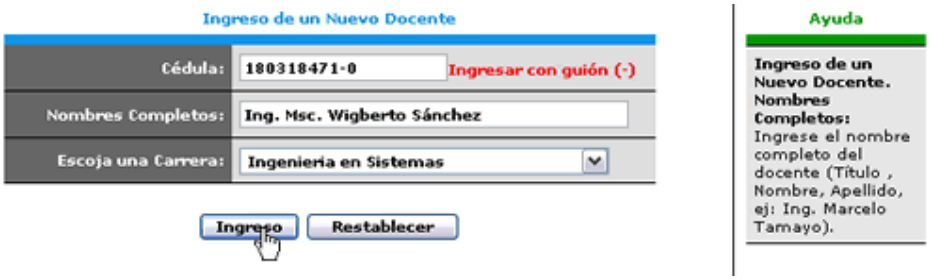
ⓕ **Página actual,** indica la página actual y el número de páginas encontradas por el sistema.

ⓖ **Buscador,** su función es la de realizar una búsqueda por algún criterio, estos criterios pueden ser por nombre, apellido, titulo o número de cédula. Ver también Uso del Buscador.

Búsqueda de Docentes. La forma más fácil de acceder a un docente determinado es mediante la ayuda del buscador. Ingrese el criterio de búsqueda, presione "Buscar"; el sistema mostrará todos los docentes que coincidan con el criterio ingresado.



Creación de Docentes. Damos "clic" en el botón "Nuevo". Como primer paso el sistema pide que se ingrese el número de cédula del docente (número de cédula con guión) para verificar si el docente se encuentra registrado y poder recuperar sus nombres automáticamente; de no encontrarse registrado, es necesario ingresar el nombre del docente y escoger una carrera, damos "clic" en el botón de ingreso.



El sistema nos indicará si se realizó correctamente el ingreso del nuevo docente o se generó algún error en el proceso.

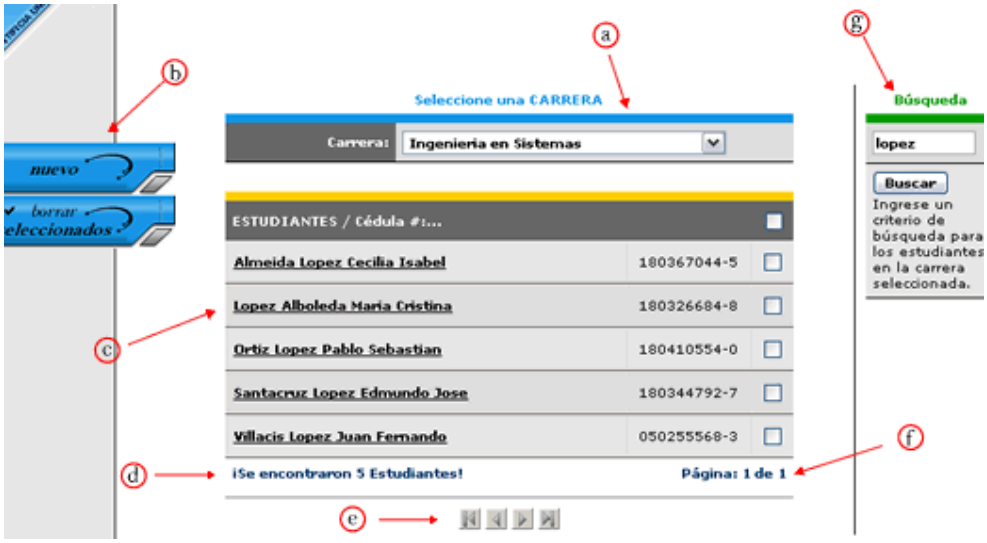
Modificación de Docentes. Para la actualización de datos de un docente damos "clic" en el docente a modificarse; esta nos enviará a un formulario indicándonos los datos actuales del docente.

Eliminación de Docentes. Si queremos realizar esta operación, es necesario seleccionar el docente a eliminarse y dar "clic" en el botón "Borrar seleccionados".

Importante. El sistema solo permite que se eliminen los docentes que no tengan relación alguna con otro dato, es decir, si existe alguna materia que se encuentre asignada al docente a eliminarse, el sistema indicará un mensaje de "Atención" indicando que no se pudo eliminar.

ADMINISTRACIÓN DE ESTUDIANTES

Para acceder a la pantalla de la administración de estudiantes debemos seleccionar del menú de "Procesos" la opción "Administración de Estudiantes".



A continuación se indican las partes en las que se encuentra constituida la pantalla:

ⓐ **ComboBox de Carreras,** contiene todas las carreras existentes en el sistema a las que determinado usuario tiene acceso.
ⓑ **Botones de Mantenimiento,** tienen las funciones de creación y eliminación de los estudiantes del sistema.
ⓒ **Página de Estudiantes,** muestra los estudiantes existentes en la carrera seleccionada, con el número de cédula.
ⓓ **Número de Estudiantes encontrados,** indica el número total de estudiantes que se encontró en la base de datos del sistema.
ⓔ **Navegador,** tienen la función de navegar entre páginas.
ⓕ **Página actual,** indica la página actual y el número de páginas encontradas por el sistema.
ⓖ **Buscador,** su función es la de realizar una búsqueda por algún criterio, estos criterios pueden ser por nombre, apellido o número de cédula. Ver también Uso del Buscador.

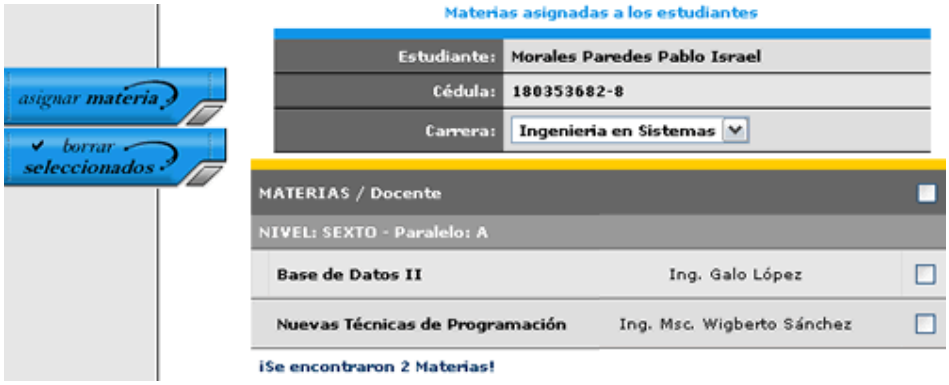
Búsqueda de Estudiantes. La forma más fácil de acceder a un estudiante determinado es mediante la ayuda del buscador. Ingrese el criterio de búsqueda (estos criterios pueden ser el nombre, el apellido o el número de cédula del estudiante), presione "Buscar"; el sistema mostrará todos los estudiantes que coincidan con el criterio ingresado.



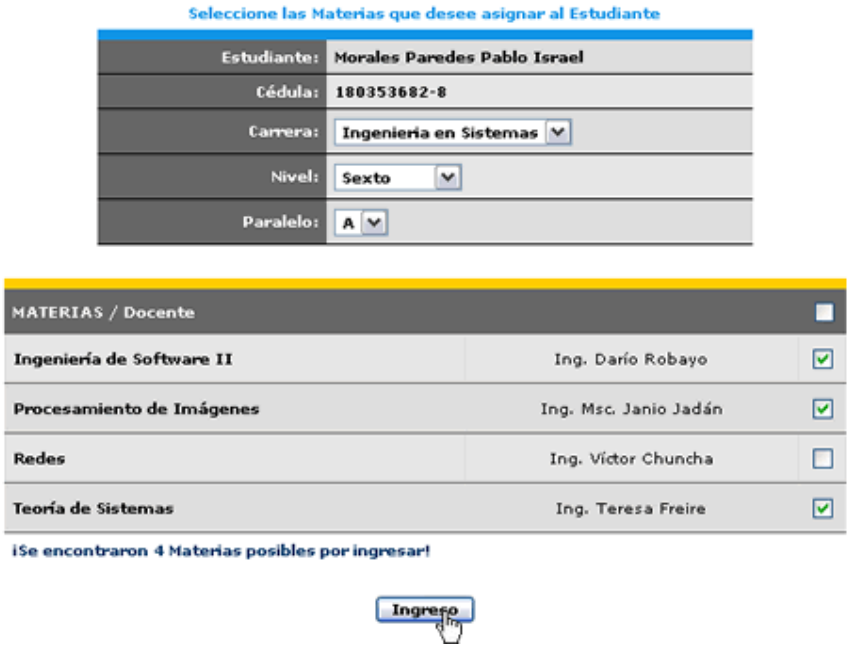
Administración de Materias de Estudiantes. Para poder asignar o eliminar materias a un estudiante debemos dar "clic" en el estudiante al cual queremos asignar o eliminar un número determinado de materias. Esta operación nos enviará a una pantalla de "Modificación del Estudiante" donde debemos dar "clic" en la carrera a la cual le pertenecen las materias que van a ser asignadas o eliminadas del estudiante.



A continuación el sistema indica una pantalla con las materias que se encuentran asignadas al estudiante en un nivel y paralelo determinado.

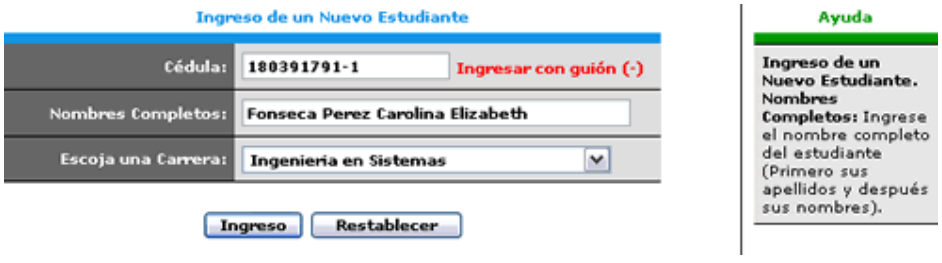


Ingreso de materias. Para ingresar nuevas materias al estudiante damos "clic" en el botón de "Asignar Materias"; el sistema nos enviará a una pantalla de selección de las posibles materias en una carrera, nivel y paralelo determinados que podrían ser ingresadas. Seleccionamos las materias que se van a asignar al estudiante. Presionamos "Ingreso".



Eliminación de Materias. Para eliminar materias asignadas al estudiante, seleccionamos la materia a eliminarse y presionamos el botón de "borrar seleccionados".

Creación de Estudiantes. Damos "clic" en el botón "Nuevo". Como primer paso el sistema pide que se ingrese el número de cédula del estudiante (número de cédula con guión) para verificar si el estudiante se encuentra registrado y poder recuperar sus nombres automáticamente; de no encontrarse registrado, es necesario ingresar el nombre del estudiante y escoger una carrera, damos "clic" en el botón de ingreso.



Si se encontró al estudiante como ya registrado, el sistema indica la pantalla de "Materias Asignadas al Estudiante" para la administración de las mismas.

Modificación de Estudiantes. Para la actualización de datos de un estudiante damos "clic" en el estudiante a modificarse; esta nos enviará a un formulario indicándonos los datos actuales del estudiante.

Eliminación de Estudiantes. Si queremos realizar esta operación, es necesario seleccionar el estudiante a eliminarse y dar "clic" en el botón "Borrar seleccionados".

 **Importante.** El sistema solo permite que se eliminen los estudiantes que no tengan relación alguna con otro dato, es decir, si existe alguna materia que se encuentre asignada al estudiante a eliminarse, el sistema indicará un mensaje de "Atención" indicando que no se pudo eliminar.

ASIGNACIÓN DE PARALELOS A NIVELES

Para acceder a la pantalla de la "Asignación de Paralelos a Niveles" debemos seleccionar del menú de "Procesos" la opción "Asignación de Datos -> Paralelos a Niveles".

Asignación de Paralelos. Presionamos el botón "Nuevo". A continuación seleccionamos una carrera, un nivel, y el paralelo a ser creado, presionamos "Crear". El sistema automáticamente creará el paralelo en el nivel seleccionado con todas las materias que le corresponden.

Eliminación de Paralelo Asignado. Si queremos realizar esta operación, es necesario seleccionar el paralelo del nivel seleccionado a eliminarse y dar "clic" en el botón "Borrar seleccionados".

 **Importante.** El sistema solo permite que se eliminen los paralelos que no tengan relación alguna con otro dato, es decir, si existe algún estudiante se en-

cuentre asignado al paralelo a eliminarse, el sistema indicará un mensaje de "Atención" indicando que no se pudo eliminar.

ASIGNACIÓN DE MATERIAS A DOCENTES

Para acceder a la pantalla de la "Asignación de Materias a Docentes" debemos seleccionar del menú de "Procesos" la opción "Asignación de Datos -> Materias a Docentes".

Asignación de Docentes. Seleccionamos la carrera, el nivel y el paralelo donde se encuentra la materia a la que se busca asignar un docente. Damos un "clic" en la materia deseada; esta nos enviará a una pantalla en donde se encuentra un ComboBox que contiene todos los docentes Activos de la carrera seleccionada. Escogemos el docente deseado y presionamos "Actualizar".

Modificación del Docente de una Materia. Si deseamos cambiar al docente de una materia, procedemos de forma similar que el paso anterior "Asignación de Docentes".

EVALUACION

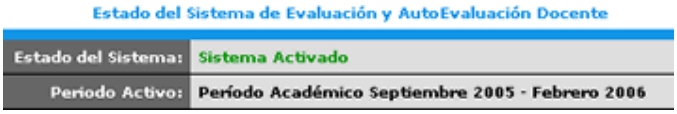
Aquí podemos ver como activar y desactivar el Sistema de Evaluación Docente, así como el ingreso de los datos necesarios para la creación de los cuestionarios de evaluación.

ACTIVAR / DESACTIVAR EL SISTEMA

Para acceder a la pantalla para activar o desactivar el sistema debemos seleccionar del menú de "Evaluación" la opción "Activar / Desactivar el sistema".



El sistema de evaluación indicará al usuario administrador el estado en que se encuentra el sistema de evaluación docente y el período académico que se encuentra activo.



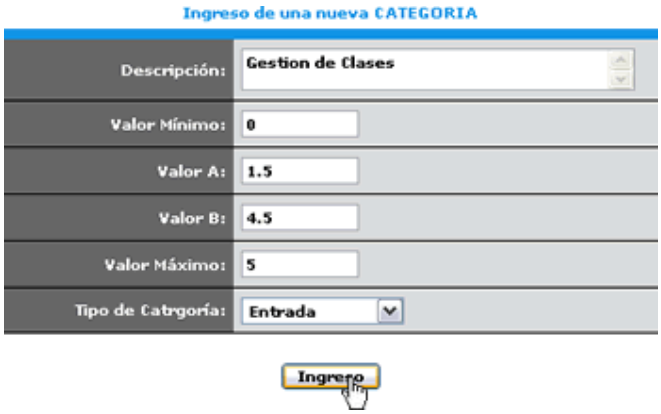
Activar/Desactivar Sistema. Necesitamos dar "clic" en el botón "Activar Sistema de Evaluación Docente" para activar el sistema y para desactivar el sistema en el botón "Desactivar el Sistema de Evaluación Docente".

Importante. Es aconsejable que el sistema solo se active en el período de evaluación docente de la universidad.

CATEGORIAS

Para poder observar y dar mantenimiento a las categorías existentes en el sistema debemos seleccionar del menú de "Evaluación" la opción "Ingreso de Datos -> Categorías".

Creación de Categorías. Si queremos ingresar una nueva categoría damos "clic" en el botón "Nuevo", este nos enviará a un formulario de ingreso de una nueva categoría.



Valores.

- Descripción. Es el nombre de la categoría que describe al grupo de preguntas que podría tener.
- Valor Mínimo. Es el valor mínimo a considerarse en el proceso de evaluación.
- Valor A y B. Son los valores que forman el rango de evaluación. Los resultados del proceso de evaluación que se encuentren dentro de estos valores serán procesados y asignados a sub rangos de evaluación para emitir un criterio más natural de los resultados obtenidos por el sistema.
- Valor Máximo. Es el valor máximo a considerarse en el proceso de evaluación.
- Tipo de Categoría. Nos indica el tipo de categoría que tomará en cuenta el sistema para la creación de cuestionarios de evaluación o para generar reportes e informes una vez terminado el proceso de evaluación.

Importante. Los valores por defecto que la categoría de entrada tiene, se encuentran por defecto en un rango entre 0 y 5. Los valores por defecto de la categoría de salida se encuentran en un rango entre 0 y 100, el sistema toma estos valores los procesa y obtiene una evaluación que va con una calificación de 30 a 50 realizando una comparativa con tipo de evaluación que se realiza a un estudiante.

Nota. Existe solo una categoría de salida. Si el sistema encuentra más de una, tomará en consideración la última categoría de salida que se encontrare en la base de datos.

Nota. Es necesario tomar en consideración que el sistema reporta la evaluación entre 30 y 50 siempre que los valores por defecto de la categoría de salida se mantengan, si estos se cambian, el rango de evaluación variará, por Ej.: si el valor máximo de la variable de salida es 200, el sistema obtendrá una evaluación entre 60 y 100 aproximadamente. Podemos guiarnos tomando en consi-

deración que el valor más alto de evaluación será la mitad del valor máximo de nuestra categoría de salida.

Modificación de Categorías. Para la actualización de datos de una categoría damos "clic" en la categoría a modificarse; esta nos enviará a un formulario indicándonos los datos actuales de la categoría.

Eliminación de Categorías. Si queremos realizar esta operación, es necesario seleccionar la categoría a eliminarse y dar "clic" en el botón "Borrar seleccionados".

 **Importante.** El sistema solo permite que se eliminen las categorías que no tengan relación alguna con otro dato, es decir, si existe alguna pregunta que pertenezca a la categoría a eliminarse, el sistema indicará un mensaje de "Atención" indicando que no se pudo eliminar.

PREGUNTAS

En el menú "Evaluación", seleccionamos la opción "Ingreso de Datos -> Preguntas" para poder observar y dar mantenimiento a las preguntas existentes en el sistema.

Creación de Preguntas. Si queremos ingresar una nueva pregunta damos "clic" en el botón "Nuevo", este nos enviará a un formulario de ingreso de una nueva pregunta. Ingresamos la descripción de la pregunta y seleccionamos una categoría a la cual esta va a pertenecer.

Modificación de Preguntas. Para la actualización de datos de una pregunta es necesario dar "clic" en la pregunta a modificarse; esta nos enviará a un formulario indicándonos los datos actuales de la pregunta.

Eliminación de Preguntas. Si queremos realizar esta operación, es necesario seleccionar la pregunta a eliminarse y dar "clic" en el botón "Borrar seleccionados".

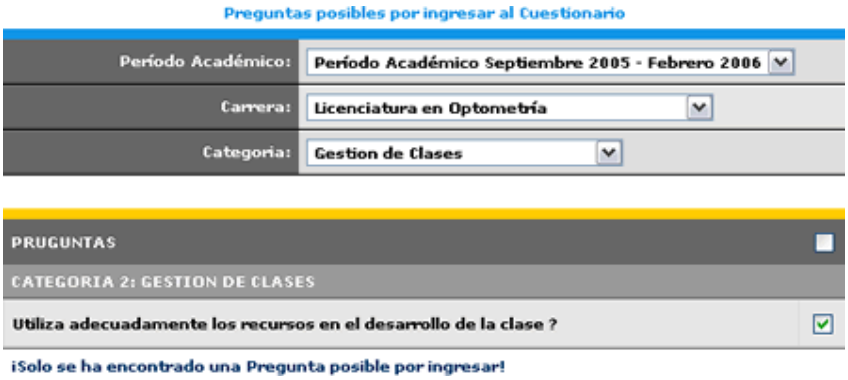
Importante. El sistema solo permite que se eliminen las preguntas que no tengan relación alguna con otro dato, es decir, si existe algún cuestionario de evaluación que contenga a la pregunta a eliminarse, el sistema indicará un mensaje de "Atención" indicando que no se pudo eliminar.

CUESTIONARIOS

En el menú "Evaluación", seleccionamos la opción "Ingreso de Datos -> Cuestionarios" para poder observar y dar mantenimiento a los cuestionarios existentes en el sistema.

Creación de Cuestionarios. Si queremos ingresar una nueva pregunta a un cuestionario en un período académico determinado, damos "clic" en el botón "Nuevo", este nos enviará a un formulario de ingreso de una nueva pregunta al cuestionario.

Paso 1. Seleccionamos un período académico, una carrera y la categoría que contiene a las preguntas que deseamos ingresar.
Paso 2. Seleccionamos las preguntas que deseamos ingresar al cuestionario y presionamos "ingresar seleccionados".



Modificación de Cuestionarios. Para la actualización de datos de un cuestionario es necesario seleccionar el período académico y la carrera que contienen al cuestionario. Damos "clic" en el botón "nuevo" si deseamos ingresar una nueva pregunta al cuestionario; si queremos por el contrario eliminar alguna pregunta del cuestionario, es necesario escoger la pregunta a eliminarse y dar "clic" en el botón "eliminar seleccionados".

OBSERVACIONES

En el menú "Evaluación", seleccionamos la opción "Ingreso de Datos -> Observaciones" para poder ver y dar mantenimiento a las observaciones existentes en el sistema.

Modificación del Estado y la Observación. Si queremos modificar la descripción del estado de las observaciones de evaluación, damos "clic" el estado a modificarse. este nos enviará a un formulario indicándonos la descripción del estado de evaluación y su observación respectiva.

 **Importante.** Es necesario mantener la jerarquía de las observaciones. Por defecto los valores del sistema se encuentran en un orden jerárquico de mayor a menor, estos estados son: destacado, competente, básico, regular e insatisfactorio. Si deseamos modificar la descripción del estado de evaluación es indispensable mantener el grado de calificación al cual cada estado representa, por Ej., si deseamos modificar la descripción del estado "destacado", la podemos remplazar por "excelente", es decir, manteniendo el grado de importancia del estado de observación.

REPORTES

El Sistema de Evaluación Docente nos brinda varios tipos de reportes e informes.

CONSULTA GENERAL

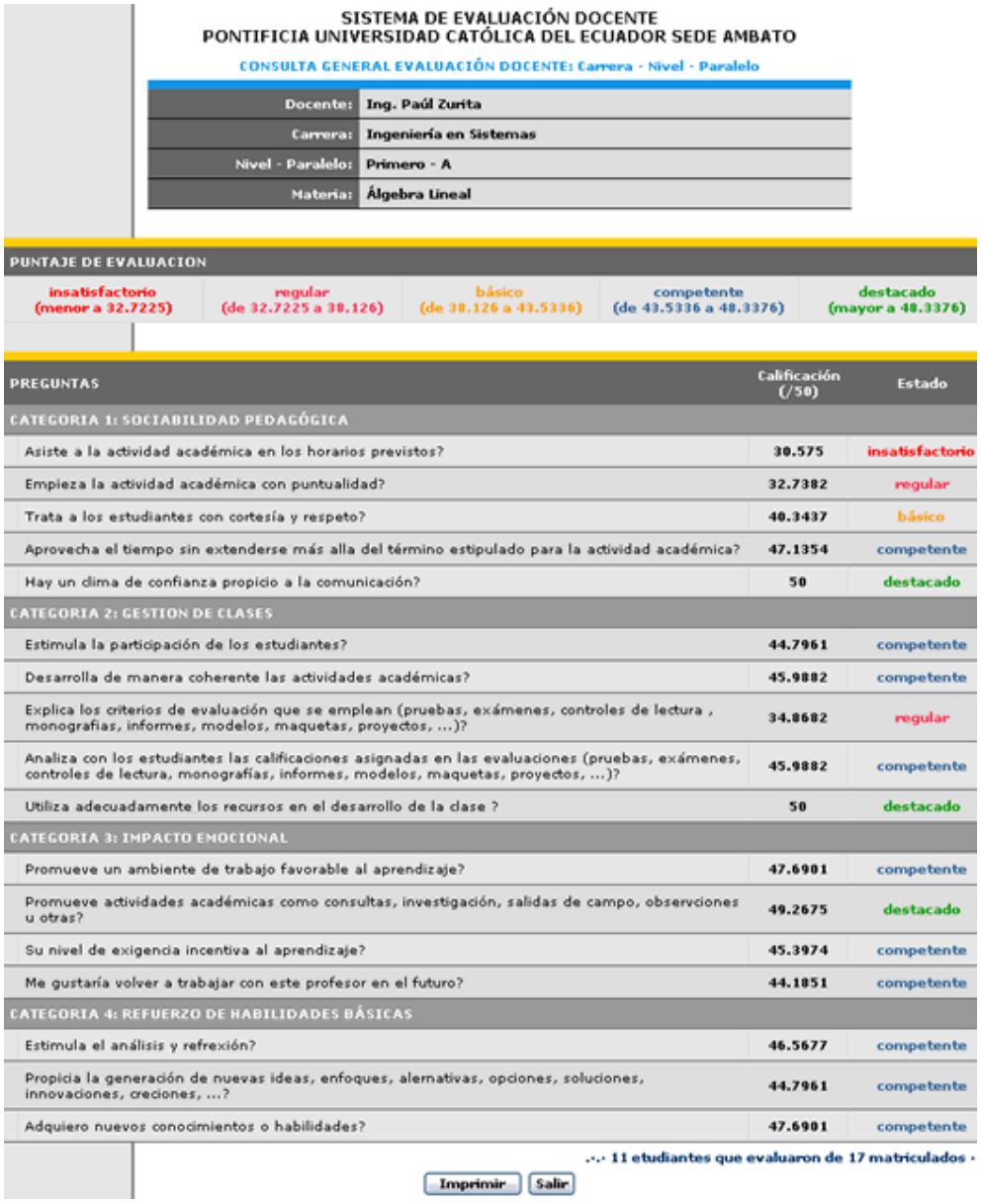
En el menú "Reportes", seleccionamos la opción "Consulta General" para poder observar y obtener los resultados de las evaluaciones existentes en el sistema.

Esta ventana muestra detalladamente el estado del proceso de evaluación en el período académico que se encuentre activo, dependiendo de la carrera, nivel y paralelo seleccionados.



Ver evaluación y autoevaluación. Para poder observar las evaluaciones y autoevaluaciones de un docente determinado, es necesario dar "clic" en el vínculo "Ver evaluación" o en "Ver autoevaluación"; estos vínculos se habilitan solo cuando el proceso de evaluación por cada materia ya ha terminado.

Consulta General. El sistema muestra un informe global de evaluación que comprende las evaluaciones de todos los estudiantes que realizaron el proceso de evaluación. El formulario consta de datos de la materia, al docente, puntaje y cuestionario de evaluación con la calificación y estado respectivos por cada pregunta.



SISTEMA DE EVALUACIÓN DOCENTE
PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR SEDE AMBATO

CONSULTA GENERAL EVALUACIÓN DOCENTE: Carrera - Nivel - Paralelo

Docente:	Ing. Paúl Zurita
Carrera:	Ingeniería en Sistemas
Nivel - Paralelo:	Primero - A
Materia:	Álgebra Lineal

PUNTAJE DE EVALUACION

insatisfactorio (menor a 32.7225)	regular (de 32.7225 a 38.126)	básico (de 38.126 a 43.5336)	competente (de 43.5336 a 48.3376)	destacado (mayor a 48.3376)

PREGUNTAS	Calificación (/50)	Estado
CATEGORIA 1: SOCIABILIDAD PEDAGÓGICA		
Asiste a la actividad académica en los horarios previstos?	30.575	insatisfactorio
Empieza la actividad académica con puntualidad?	32.7382	regular
Trata a los estudiantes con cortesía y respeto?	40.3437	básico
Aprovecha el tiempo sin extenderse más alla del término estipulado para la actividad académica?	47.1354	competente
Hay un clima de confianza propicio a la comunicación?	50	destacado
CATEGORIA 2: GESTION DE CLASES		
Estimula la participación de los estudiantes?	44.7961	competente
Desarrolla de manera coherente las actividades académicas?	45.9882	competente
Explica los criterios de evaluación que se emplean (pruebas, exámenes, controles de lectura , monografias, informes, modelos, maquetas, proyectos, ...)?	34.8682	regular
Analiza con los estudiantes las calificaciones asignadas en las evaluaciones (pruebas, exámenes, controles de lectura, monografías, informes, modelos, maquetas, proyectos, ...)?	45.9882	competente
Utiliza adecuadamente los recursos en el desarrollo de la clase ?	50	destacado
CATEGORIA 3: IMPACTO EMOCIONAL		
Promueve un ambiente de trabajo favorable al aprendizaje?	47.6901	competente
Promueve actividades académicas como consultas, investigación, salidas de campo, observciones u otras?	49.2675	destacado
Su nivel de exigencia incentiva al aprendizaje?	45.3974	competente
Me gustaría volver a trabajar con este profesor en el futuro?	44.1851	competente
CATEGORIA 4: REFUERZO DE HABILIDADES BÁSICAS		
Estimula el análisis y refrexión?	46.5677	competente
Propicia la generación de nuevas ideas, enfoques, alernativas, opciones, soluciones, innovaciones, creciones, ...?	44.7961	competente
Adquiero nuevos conocimientos o habilidades?	47.6901	competente

·-·- 11 etudiantes que evaluaron de 17 matriculados ·

Imprimir Salir

Importante. El sistema habilita el vínculo "Ver evaluación" de una determinada materia solo cuando el número de estudiantes que la han evaluado supera el 50% del total de estudiantes matriculados en ella.

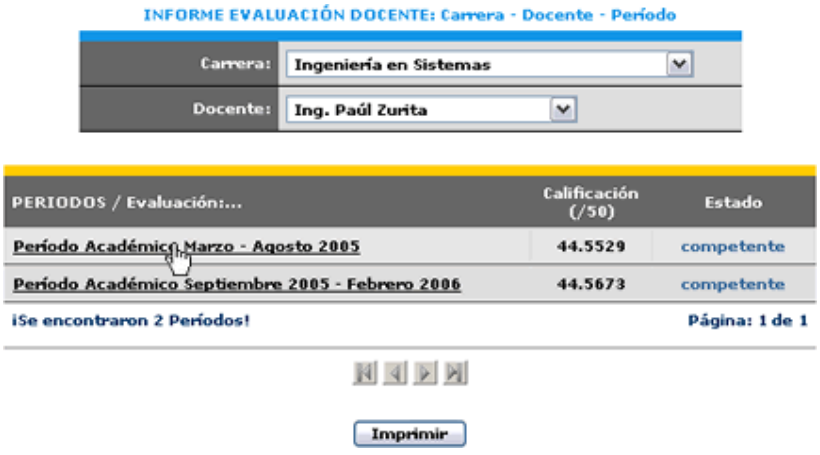
INFORME EVALUACIÓN

En el menú "Reportes", seleccionamos la opción "Informe Evaluación", para poder observar y obtener los resultados de las evaluaciones existentes en el sistema, esta se divide en cuatro categorías:

- Por Período
- Por Carrera (Comparativo)
- Por Materia
- Por Docente
-



Por Período. Este formulario nos indica los resultados de las evaluaciones existentes en el sistema por período académico en forma global. Para ver el detalle de cada una de ellas es necesario seleccionar la carrera, el docente y dar "clic" en el período académico deseado.



Por Carrera (Comparativo). Este formulario nos indica las posibles evaluaciones en el caso que un docente se encuentre registrado en más de una carrera.



Por Materia. Este formulario nos indica las materias en las cuales el docente esta registrado con su evaluación respectiva.



Por Docente. Este formulario nos muestra la evaluación global de cada uno de los docentes.



INFORME AUTOEVALUACIÓN

En el menú "Reportes", seleccionamos la opción "Informe AutoEvaluación", para poder observar y obtener los resultados de las autoevaluaciones existentes en el sistema, esta se divide en cuatro categorías idénticas a las de evaluación.

ESTUDIANTES SIN EVALUACIÓN

El sistema nos permite conocer los estudiantes que no realizaron el proceso de evaluación, para ello hay que seleccionar la opción "Estudiantes sin Evaluación" del menú "Reportes".

DOCENTES SIN AUTOEVALUACIÓN

El sistema también nos permite conocer los docente que no realizaron el proceso de auto evaluación, para ello es necesario seleccionar la opción "Docentes sin AutoEvaluación" del menú "Reportes".

PROCESO DE EVALUACIÓN

El proceso de evaluación por parte de los estudiantes se realizará solo si el Sistema de Evaluación Docente se encuentra "activado"; y el estudiante lo realizará de la siguiente forma:

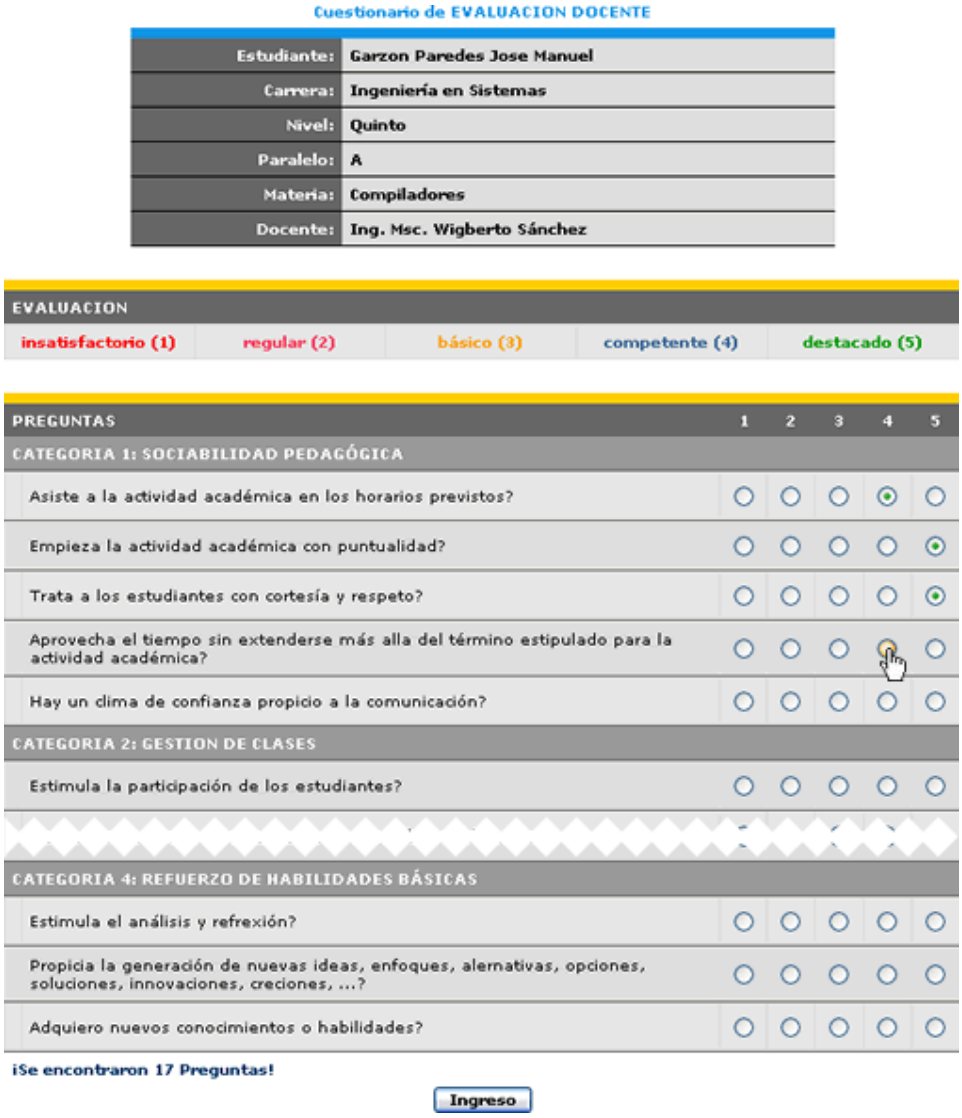
Ingreso al Sistema. El estudiante ingresa al sistema con su número de cédula. Una vez verificada la cédula, el sistema indicará una pantalla con los datos del estudiante y todas las posibles materias a las cuales el estudiante debe evaluar.



La tabla de materias se encuentra ordenada por nivel y paralelo, contiene las materias y docentes en las que el alumno se encuentra matriculado. El sistema también indica el estado de las materias y docentes: "Por evaluar" o "Evaluado".

Nota. Si el estudiante se encuentra matriculado en más de una carrera, podrá ver las materias de la carrera actualizando el formulario mediante el combo box "Carreras".

Proceso de evaluación. El estudiante da "clic" en la materia a evaluarse, esta le enviará al cuestionario respectivo de la carrera con los datos de la materia y el docente a ser evaluado.



Seleccionando uno a uno los radio button por cada pregunta, el estudiante realizará el proceso de evaluación docente de acuerdo a su criterio.

Para finalizar la evaluación, el estudiante debe dar "clic" en el botón "Ingreso" y de esta forma se finaliza el proceso de evaluación para la materia seleccionada. Este proceso es necesario repetirlo en la totalidad de materias en las cuales el estudiante se encuentra matriculado.

Importante. Para que una evaluación se tome como válida, esta deberá ser realizada con el 80% mínimo de preguntas contestadas; de no ser así, el sistema indicará un mensaje de "Atención" indicando que el número de preguntas contestadas no supera el porcentaje requerido por el sistema de evaluación docente.

Cerrar Sesión. El estudiante deberá salir del sistema cerrando su sesión, es necesario dar "clic" en el vínculo "cerrar sesión" para salir de forma segura del sistema.

PROCESO DE AUTOEVALUACIÓN

De la misma forma que el proceso de evaluación, el proceso de autoevaluación se lo realizará si el Sistema de Evaluación Docente se encuentra "activado"; y el docente lo realizará de la siguiente forma:

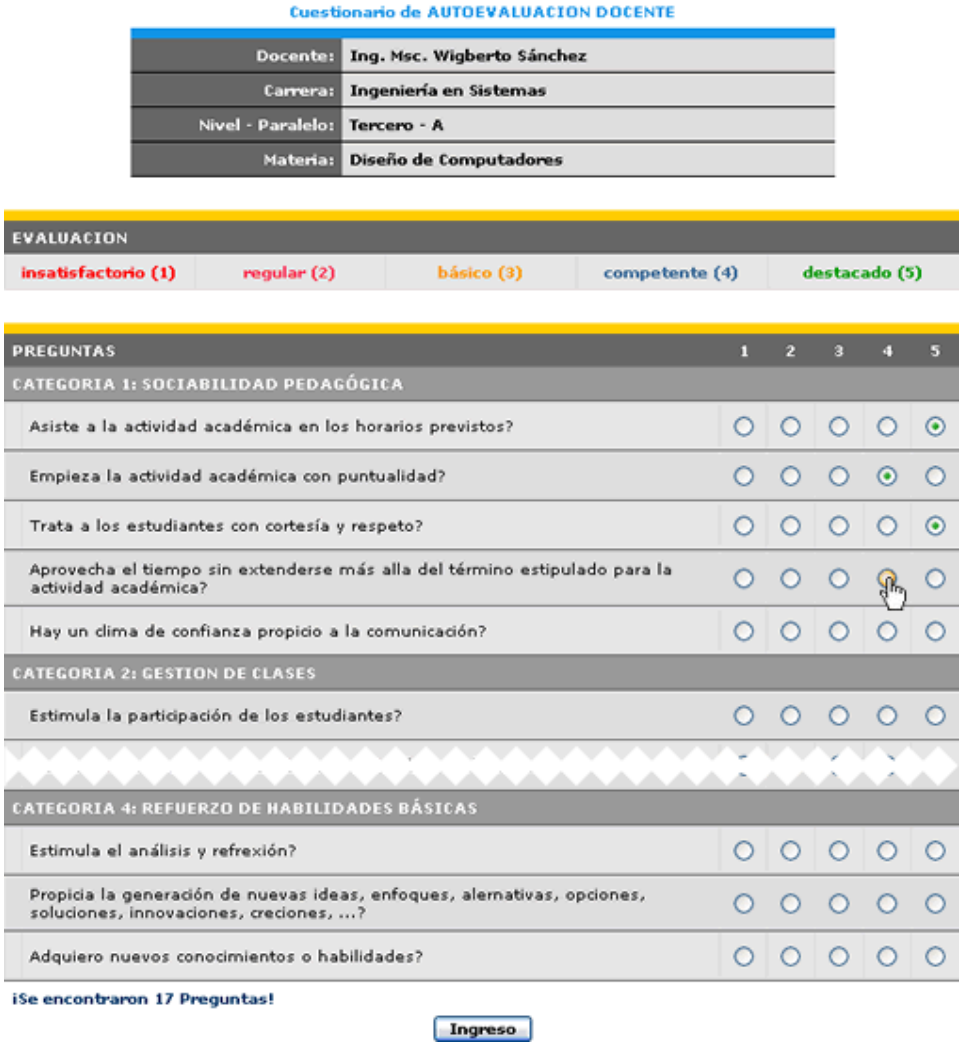
Ingreso al Sistema. El docente ingresa al sistema con su número de cédula. Una vez verificada la cédula, el sistema indicará una pantalla con los datos del docente y todas las materias en las que encuentra asignado.



La tabla de materias se encuentra ordenada por nivel y paralelo, contiene las materias en las que el docente se encuentra registrado. El sistema también indica el estado de las materias: "Por autoevaluar" o "Autoevaluado" y "Por evaluar" o "Evaluado".

Nota. Si el docente se encuentra registrado en más de una carrera, podrá ver las materias de la carrera actualizando el formulario mediante el combo box "Carreras".

Proceso de autoevaluación. El docente da "clic" en la materia por autoevaluarse, esta le enviará al cuestionario respectivo de la carrera con los datos de la materia a ser evaluada.



Seleccionando uno a uno los radio button por cada pregunta, el docente realizará el proceso de autoevaluación docente de acuerdo a su criterio.

Para finalizar la autoevaluación, el docente debe dar "clic" en el botón "Ingreso" y de esta forma se finaliza el proceso de autoevaluación para la materia seleccionada. Este proceso es necesario repetirlo en la totalidad de materias en las cuales el docente se encuentra registrado.

Importante. Para que una evaluación se tome como válida, esta deberá ser realizada con el 80% mínimo de preguntas contestadas; de no ser así, el sistema indicará un mensaje de "Atención" indicando que el número de preguntas no supera el porcentaje requerido por el sistema de evaluación.

Cerrar Sesión. El docente deberá salir del sistema cerrando su sesión, se da "clic" en el vínculo "cerrar sesión" para salir de forma segura del sistema.

USO DEL BUSCADOR

Varias pantallas del sistema poseen un buscador, el cual facilita la obtención de datos a través de un criterio de búsqueda determinado. A continuación veremos algunos criterios de búsqueda que permite el sistema.

Búsqueda de un docente. En la pantalla de administración de docentes, podemos utiliza el buscador con varios criterios, por Ej.: por el número de cédula del docente, sus apellidos o sus nombres, título.



Búsqueda de un estudiante. Podemos utilizar criterios de búsqueda como cédula del estudiante, apellidos o nombres.



USO DEL NAVEGADOR

El navegador nos permite avanzar a todas las páginas para acceder a todos nuestros datos del Sistema de Evaluación Docente.

Primera. Permite regresar a la primera página del formulario.

Anterior. Permite regresar una página anterior del formulario.

Siguiente. Permite avanzar una página del formulario.

Última. Permite avanzar a la última página del formulario.

BIBLIOGRAFÍA.

Libros:

- ADAMS, Georgia Sach. Medición y evaluación. (1983).
- MINISERIO DE EDUCACIÓN Y CULTURA. Evaluación de los aprendizajes (2004)
- CONUEP. Proyecto para el establecimiento del sistema de Evaluación y Acreditación Universitaria. (1997).
- PAGE, Mariano y VICTORIA, Jesús. De la evaluación externa de la reforma de las enseñanzas medias a la evaluación permanente del sistema educativo. (1988).
- RUIZ, José. Como hacer una evaluación de Centros Educativos. (1996).
- PEREZ, Mauricio y BUSTAMANTE, Guillermo. Evaluación Escolar. ¿Resultados o Procesos? (1996).
- GALLEGO, Rómulo. Evaluación Pedagógica y Promoción Académica. (1989).
- PRESSMAN, R. Ingeniería de Software.(2002).
- BRAUDE, Eric J. Ingeniería de Software, Una perspectiva Orientada a Objetos. (2003).
- CASTILLO, Enrique, GUITIERRES, José y HADI, Ali. Sistemas Expertos y Modelos de redes Probabilísticas. (1996)
- ELAIN, Rich. KNIGHT Kevin. Inteligencia Artificial. (1994).
- RUSELL, Stuart. Inteligencia Artificial un Enfoque Moderno. (1996.).
- HARWRYSZKIEWYCZ. Análisis y diseño de base de datos. (1994).
- BAKKEN, Stig Sæther, AULBACH, Alexander, SCHMID, Egon, WINSTEAD, Jim, WILSON, Lars Torben, LERDORF, Rasmus, ZMIEVSKI, Andrei y AHTO, Jouni. Manual de PHP. (2002).
- MySQL AB. MySQL Reference Manual. (2000).
- WELLING, Luke y THOMSON, Laura. PHP and MySQL. Web Development. (2001).

Páginas Web:

- http://www.monografias.com/trabajos7
- http://www.inf.udec.cl
- http://www.lafacu.com/apuntes/informatica
- http://www.unav.es
- http://www.php.com
- http://ww.mysql.com
- http://www.mysql-hispano.org
- http://www.arrakis.es
- http://mysql.conclase.net
- http://es.wikipedia.org
- http://www.gestiopolis.com
- http://www.gratisweb.com/comarsc/inar6.htm
- http://consc.net/online.html
- http://www.fortunecity.com/skyscraper/chaos/279/docs/sistexpertos.htm
- http://home.worldonline.es/jmariocr/introduccion.htm
- http://ciberconta.unizar.es/Biblioteca