

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMA ADMINISTRADOR Y CREADOR DE TUTORIALES

T E S I S

QUE PARA OBTENER EL TITULO DE:
INGENIERO EN COMPUTACIÓN

P R E S E N T A N

CHÁVEZ LEAL ELSA IBETH
RAMÍREZ VÁZQUEZ DARINKA

DIRECTOR DE TESIS: M.C. MA. JAQUELINA LÓPEZ BARRIENTOS



MÉXICO, D.F., 2005

Deseo darles las gracias a las personas que han estado a mi lado en el trayecto de mi vida:

- **DIOS**

  Por darme la oportunidad de terminar mis estudios universitarios, por no dejarme sola cuando pensé era así y mostrarme las cosas por las cuales vale la pena seguir adelante esforzándose.

  Subir los remos y dejarse llevar con los ojos cerrados.
  Abrir los ojos y encontrarse vivo: se repitió el milagro.
  Anda, levántate y olvida esta ribera oculta en que has desembarcado. (Sain)

- **FAMILIARES**

  A ti mamá por darme tu apoyo y comprensión en todo momento, por darme alientos para continuar y solidarizarte cuando te necesité. Gracias en verdad, por ti soy lo que soy.

  A mi papá, hermana, cuñado, sobrinos y tía, que también estuvieron a mi lado apoyándome.

- **PERSONALES**

  Adrián, por tu paciencia, apoyo, enseñanzas y tiempo. Por encontrar la forma de hacerme sentir bien y estar a mi lado siempre.
  Darinka, principalmente por tu amistad, tolerancia, apoyo y entusiasmo.
  Oscar, Francisco, Israel, Armando por estar cerca de algún modo y hacerme saber que cuento con ustedes en todo momento.
  Ricardo, Fabiola y Nancy, sé que aún cuando la distancia y actividades nos separan, estamos juntos en pensamiento.
  Y a mis demás amigos que me han ayudado de alguna otra forma.

- **UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO**

  Por haberme dado la oportunidad de estudiar en sus aulas, por todas las oportunidades culturales, estudiantiles, personales y sociales que me dio.
  Profesora Jaquelina, por su tiempo, paciencia, apoyo, optimismo y amistad.
  Alfredo, por apoyarme académica y moralmente.
  Profesores, por transmitir sus conocimientos.

Elsa Ibeth Chávez Leal

Quiero agradecer primeramente a Dios por darme salud, paciencia y determinación para alcanzar una de las metas más importantes de mi vida.

Dedico este trabajo a mi mamá, **Ma. Margarita Vázquez** agradeciéndole por todo su amor, paciencia, comprensión, consejos e incondicional apoyo, gracias mamá porque todo lo que soy te lo debo a ti, porque se el gran esfuerzo que hiciste para sacarme adelante y ver nuestro sueño hecho realidad, te amo mamá.

A mi familia y hermanos, **Antonio** y **Rodrigo** que siempre me han apoyado en todo lo que me propongo, gracias por su comprensión y disposición para ayudarme.

A mi amiga, **Elsa** por sus consejos, paciencia y comprensión en los momentos más difíciles, por haber tenido determinación para terminar juntas este proyecto a pesar de todos los problemas que se nos presentaron. Sin ti no lo hubiera terminado. Gracias.

A mis compañeros de trabajo por su comprensión, apoyo y orientación para la elaboración del sistema propuesto, por compartir su conocimiento conmigo y ayudarme cuando tuve algún problema, gracias a todos.

A la **M.C. Jaquelina López Barrientos** por su tiempo, dedicación y orientación que nos proporciono durante el desarrollo de este proyecto, muchas gracias.

A la **Universidad Nacional Autónoma de México** por permitir que mi formación profesional fuese aquí y darme la oportunidad de ser parte de esta gran casa de estudios, me siento orgullosa de pertenecer a esta gran institución educativa.

ÍNDICE

Introducción ---- I

Capítulo 1
Psicopedagogía orientada al desarrollo de Software ---- 1
1.1. Aprendizaje ---- 2
1.2. Estrategias de aprendizaje ---- 2
1.2.1. Estrategias de procesamiento ---- 2
1.2.2. Estrategias de producción ---- 3
1.3. Medición del aprendizaje ---- 3
1.3.1. Retención y olvido ---- 4
1.4. Enfoques sobre aprendizaje ---- 4
1.4.1. Enfoque de Piaget ---- 4
1.4.2. Enfoque de Collins y Duguid ---- 5
1.4.3. Enfoque de Thorndike ---- 5
1.4.4. Enfoque de Pavlov ---- 6
1.4.5. Enfoque de Skinner ---- 7
1.4.6. Reducción de necesidades de Hull ---- 7
1.4.7. Conclusiones ---- 7
1.5. Las estrategias de aprendizaje y las computadoras ---- 8
1.6. Contribución pedagógica de las tecnologías de computación ---- 9
1.6.1. Funciones pedagógicas del modo herramienta ---- 9
1.6.2. Funciones pedagógicas del modo comunicacional ---- 9
1.7. Tipografía ---- 10
1.7.1. Normas generales de tipografía ---- 10
1.8. Colorimetría ---- 12
1.8.1. Normas sobre los colores ---- 13
1.8.2. Lenguaje de los colores ---- 13

Capítulo 2
Marco Teórico e Ingeniería de Software ---- 15
2.1. Marco teórico ---- 16
2.2. Ingeniería del software ---- 17
2.3. Paradigmas de la ingeniería del software ---- 18
2.3.1. El ciclo de vida modificado ---- 19
2.3.1.1. Elementos del ciclo de vida modificado ---- 19
2.3.1.2. Diagrama del ciclo de vida modificado ---- 20
2.3.1.3. Ventajas del ciclo de vida modificado ---- 20
2.3.1.4. Desventajas del ciclo de vida modificado ---- 20
2.3.2. El ciclo de vida en V ---- 21
2.3.2.1. Elementos del ciclo de vida en V ---- 21
2.3.2.2. Diagrama del ciclo de vida en V ---- 21
2.3.2.3. Ventajas del ciclo de vida en V ---- 22
2.3.2.4. Desventajas del ciclo de vida en V ---- 22
2.3.3. El ciclo de vida clásico ---- 22
2.3.3.1. Elementos del ciclo de vida clásico ---- 22
2.3.3.2. Diagrama del ciclo de vida clásico ---- 23
2.3.3.3. Ventajas del ciclo de vida clásico ---- 23
2.3.3.4. Desventajas del ciclo de vida clásico ---- 23

2.3.4. El modelo en espiral ---- 24
2.3.4.1. Elementos del modelo en espiral ---- 24
2.3.4.2. Diagrama del modelo en espiral ---- 25
2.3.4.3. Ventajas del modelo en espiral ---- 25
2.3.4.4. Desventajas del modelo en espiral ---- 25
2.3.5. Prototipo ---- 26
2.3.5.1. Elementos del prototipo ---- 26
2.3.5.2. Diagrama del prototipo ---- 27
2.3.5.3. Ventajas del prototipo ---- 27
2.3.5.4. Desventajas del prototipo ---- 27
2.3.6. Técnicas de cuarta generación ---- 28
2.3.6.1. Elementos de las técnicas de cuarta generación ---- 28
2.3.6.2. Diagrama de las técnicas de cuarta generación ---- 28
2.3.6.3. Ventajas de las técnicas de cuarta generación ---- 29
2.3.6.4. Desventajas de las técnicas de cuarta generación ---- 29
2.4. Selección del modelo ---- 29
2.5. Comparativo de paradigmas ---- 30

Capítulo 3
Análisis previo al diseño del sistema ---- 32
3.1. Fases de un sistema ---- 33
3.1.1. Definición del sistema ---- 33
3.1.2. Fase de desarrollo ---- 33
3.1.3. Validación, pruebas y mantenimiento ---- 34
3.1.4. Implantación y evaluación ---- 34
3.1.5. Análisis del sistema ---- 34
3.1.6. Análisis de requerimientos ---- 35
3.2. Análisis y especificación estructurada ---- 36
3.2.1. Diagrama de flujo de datos ---- 36
3.2.2. Diccionario de datos ---- 37
3.2.3. Diagrama de entidad-relación ---- 37
3.3. Diseño del software ---- 38
3.3.1. Modularidad ---- 39
3.3.2. Cohesión ---- 40
3.3.3. Acoplamiento ---- 41
3.4. La calidad en los sistemas ---- 41
3.4.1. Normas del aseguramiento de la calidad en los sistemas ---- 42
3.4.2. Factores que afectan la calidad del software ---- 43
3.4.3. Características métricas en la calidad del software ---- 43
3.5. Pruebas del software ---- 44
3.5.1. Características de las pruebas ---- 44
3.5.2. Pasos en las pruebas de los sistemas ---- 44
3.5.3. Pruebas de validación ---- 45

Capítulo 4
Análisis del sistema ---- 47
4.1. Planteamiento del problema ---- 48
4.2. Solución ---- 48
4.2.1. Módulo administrador de contraseñas ---- 49
4.2.2. Módulo de creación y modificación de tutoriales ---- 51

4.2.3. Módulo de creación y modificación de cuestionarios ---- 51
4.2.4. Módulo de consulta de tutoriales ---- 52
4.2.5. Módulo de aplicación de exámenes ---- 52
4.2.6. Módulo de evaluación de exámenes ---- 53
4.2.7. Módulo de consultas de exámenes ---- 53
4.3. Lenguajes de programación ---- 54
4.3.1. CGI (Common Gateway Interface) ---- 55
4.3.1.1. Funcionamiento ---- 55
4.3.1.2. Ventajas ---- 56
4.3.1.3. Desventajas ---- 57
4.3.2. ASP (Active Server Pages) ---- 57
4.3.2.1. Funcionamiento ---- 58
4.3.2.2. Ventajas ---- 59
4.3.2.3. Desventajas ---- 60
4.3.3. Perl (Practical Extraction and Report Language) ---- 60
4.3.3.1. Funcionamiento ---- 60
4.3.3.2. Ventajas ---- 62
4.3.3.3. Desventajas ---- 62
4.3.4. Java ---- 62
4.3.4.1. Funcionamiento ---- 63
4.3.4.2. Ventajas ---- 64
4.3.4.3. Desventajas ---- 65
4.3.5. PHP (Hypertext Preprocessor) ---- 65
4.3.5.1. Funcionamiento ---- 66
4.3.5.2. Ventajas ---- 67
4.3.5.3. Desventajas ---- 67
4.3.6. Elección del lenguaje de programación ---- 67
4.4. Bases de datos ---- 68
4.4.1. Access ---- 68
4.4.2. Oracle ---- 70
4.4.3. MySQL ---- 71
4.4.4. SQL Server ---- 72
4.4.5. Elección de la base de datos ---- 74
4.4.6. DTS (Data Transformation Services) ---- 75
4.5. Editor ---- 77
4.5.1. AreaHTML ---- 77

Capítulo 5
Diseño y Desarrollo del Sistema ---- 80
5.4. Diagrama de flujo ---- 81
5.4.1. Administrador ---- 82
5.4.2. Desarrollador ---- 83
5.4.3. Profesor ---- 84
5.4.4. Coordinador ---- 85
5.4.5. Alumno ordinario ---- 86
5.4.6. Alumno extraordinario ---- 88
5.5. Modelo de datos ---- 89
5.5.1. Diagrama entidad-relación ---- 90
5.6. Diccionario de datos ---- 90
5.6.1. Catálogos ---- 90

5.6.2. Tablas ---------- 92
5.6.3. Vistas ---------- 95
5.7. Pruebas del sistema ---------- 95
5.8. Mantenimiento ---------- 96
5.9. Sistema administrador y creador de tutoriales ---------- 96

Conclusiones ---------- 111

Apéndice ---------- 114

Bibliografía ---------- 126

INTRODUCCIÓN

La computadora proporciona un contexto en el cual se puede llevar el proceso de aprendizaje. Para que esto suceda, es necesario que las ventajas de las computadoras estén adaptadas a las necesidades de los educandos, de tal modo que su presencia se torne en un agente fundamental para la Enseñanza-Aprendizaje.

Como resultado de los avances informáticos, los recursos que proporciona la computadora (Voz, Audio y Video) se han incorporado como nuevas alternativas de apoyo en la comprensión y aprendizaje de temas específicos.

Las herramientas informáticas educativas que se crean día con día (Tutoriales en línea, Enciclopedias digitales, Simulaciones, etc.) pueden ser aprovechadas por el alumno en su totalidad, si se toma en cuenta su apariencia y contenido. Eso puede ser la diferencia entre que el alumno lea o no lo que está escrito y retenga o no la información. Para ello, es indispensable basarnos en la Psicopedagogía.

La importancia de aprovechar los recursos informáticos en la educación es proporcionar asistencia al alumno para adquirir o reforzar conocimientos, y permitirle al profesor dedicar más tiempo a los alumnos que necesitan atención personal y orientación en la materia.

El sistema Administrador y creador de tutoriales tiene como objetivo el ser una herramienta básica tanto de los profesores como y principalmente de los alumnos de la carrera de Ingeniería en Computación, el sistema puede contener toda la información que el alumno requiere a lo largo de la carrera y acceder a ella por medio de INTERNET.

La tecnología avanza rápidamente y es indispensable el poder contar con herramientas que apoyen el desarrollo intelectual de los alumnos y que puedan ver la información que requieran cuando lo deseen.

CAPÍTULO 1



PSICOPEDAGOGÍA ORIENTADA AL DESARROLLO DE SOFTWARE

Es necesario crear conciencia que en la realización de un software, se debe de tener al menos un panorama general de conocimientos sobre los enfoques psicopedagógicos referentes al proceso de aprendizaje.

En este capítulo, se describen los aspectos de estudio y las corrientes teóricas que pueden ser utilizadas para un mejor desarrollo de un software educativo.

Hay un apartado de colorimetría que sirve de apoyo para el desarrollador, en cuanto a los colores de las pantallas, letras, botones, etc. del software a crear.

Existe un apartado de tipografía donde se muestra una visión general de los tipos de letras, el tamaño, espacios entre ellas, etc. que son recomendables usar para que el software sea sencillo de leer y visualizar.

En conjunto, en este capítulo se presentan herramientas de apoyo para un buen diseño en cuanto a colores y tipos de letra, formatos y recomendaciones para que el contenido sea lo más concreto posible, fácil de comprender y aprender.

1.1. APRENDIZAJE

Definiciones:

- Adquirir el conocimiento de una cosa por medio del estudio o experiencia.
- Psicología: Modificación en la forma de reaccionar de un organismo frente a una situación experimentada de antemano.

En base a las definiciones anteriores, la de Gordon Cross (Psicólogo) es la más completa ya que define el aprendizaje como "un cambio permanente en la conducta como resultado de la experiencia", ya que los cambios permanentes de la conducta son consecuencia de la práctica y de la experiencia.

1.2. ESTRATEGIAS DE APRENDIZAJE

Se denomina estrategias de aprendizaje a aquellos procedimientos, mecanismos y estrategias que el aprendiz aplica, conciente o inconscientemente, para el mejoramiento de su proceso de enseñanza-aprendizaje. Influyen en el procesamiento, almacenamiento, recuperación y aplicación de conocimientos y destrezas en un sentido amplio.

Concretamente, ejercen influencia sobre la atención, el registro de estímulos, las funciones de memoria activa, sobre la memoria permanente y sobre los efectores de respuestas.

Por su forma de funcionar se clasifican las estrategias de aprendizaje como:

- Estrategias de procesamiento.
- Estrategias de producción.

1.2.1. Estrategias de procesamiento

Las estrategias de procesamiento son aquellas que utiliza el aprendiz para asegurar un proceso de captación e ingreso de nueva información, destrezas, etc., y su exitoso almacenamiento en la memoria permanente. Estas estrategias incluyen las:

- Estrategias de atención.
- Estrategias de elaboración verbal a través del uso de:
  - La repetición.
  - El parafraseo.
  - El uso de preguntas.
  - La agrupación o combinación selectiva de estructuras y redes.
- Elaboración de imágenes
- La comparación.

- La inferencia.
- La aplicación o ensayos futuros.

Al hacerse preguntas acerca de lo que uno está aprendiendo, el proceso de transformación se profundiza.

También se incluye en este rubro la formación de información relacionada con episodios, es decir, memorias de eventos que ocurrieron en una cierta secuencia y en un cierto lugar o momento en la historia del alumno. Los episodios normalmente son compuestos de imágenes.

La comparación es una actividad básica en la conducta del ser humano y también una estrategia lógica para facilitar el aprendizaje. Comparar incluye contrastar, usar analogías, metáforas, símiles y otras formas de resaltar similitudes y diferencias.

1.2.2. Estrategias de producción

Cuando es necesario recuperar la información almacenada para algún uso, comienzan a funcionar las estrategias de producción. Estas estrategias pueden ser desarrolladas y aplicadas a cuatro principales formas de producción:

- Recuperación y uso de información específica.
- Generalización o transferencia de información o aplicación de habilidades a nuevas situaciones.
- Identificación, representación y resolución de problemas aplicando lo aprendido.
- Desarrollo y aplicación de la creatividad en la generación de respuestas.

La educación es el crecimiento de las capacidades inherentes de las personas, por ello, el fin de la adquisición de información en la escuela debe ser en parte el aprender cómo generar o encontrar la información cuando ésta se necesita, es desarrollar una actitud buscante e investigadora.

1.3. MEDICIÓN DEL APRENDIZAJE

La variable independiente en el estudio del aprendizaje es la práctica y la variable dependiente es lo que hace al sujeto que aprenda.

Una medida más precisa y más útil de la ejecución, es el tiempo que se toma la respuesta en ocurrir.

Otra medida de la ejecución respecto a los experimentos de aprendizaje son el tamaño, la magnitud o la amplitud de la respuesta.

1.3.1. Retención y olvido

El olvido es la deterioración de la ejecución sin práctica alguna, reforzada o no-reforzada.

La retención es la persistencia de los cambios que se aprenden en el comportamiento sin práctica, y el olvido es la falta de persistencia de dichos cambios. Los métodos principales para medir la retención son la reproducción, el reconocimiento y el reaprendizaje.

Las condiciones de adquisición de conocimientos pueden operar sólo a través de la cantidad de aprendizaje, incluyendo el reaprendizaje, el significado y la velocidad de adquisición.

1.4. ENFOQUES SOBRE APRENDIZAJE

A lo largo de la historia han existido numerosas aportaciones para el mejor aprendizaje del ser humano, a continuación se presentan algunos enfoques de los siguientes autores: Piaget, Collins, Duguid, Thorndike, quienes se considera son los precursores que han hecho más aportaciones en el ámbito de la educación desde el punto de vista psicológico.

1.4.1. Enfoque de Piaget

Piaget define que la inteligencia implica una adaptación biológica, un equilibrio entre el individuo y su medio ambiente y una serie de operaciones mentales que permiten ese equilibrio. El desarrollo se halla influido por cuatro factores:

1. Las estructuras físicas limitan ciertos aspectos del desarrollo cognoscitivo y hacen que otras sean posibles.
2. La experiencia física exige el adquirir el conocimiento de los objetos observándolos directamente. La experiencia lógico-matemática demanda una coordinación interna de las acciones de un individuo que en un comienzo se hallan realizadas teniendo en cuenta los objetos, pero que más adelante no requieren este apoyo físico.
3. La transmisión social se refiere a la adquisición de conocimiento mediante ciertas técnicas como la lectura o la instrucción.
4. El equilibrio supone una serie de procesos autorreguladores que conducen de una manera progresiva hacia estados de equilibrio cada vez más eficaces.

En base a la teoría de Piaget enumeramos algunos puntos importantes que pueden orientar las técnicas educativas:

1. Manipulando cosas se aprende.
2. Si la experiencia es moderadamente novedosa, capta la atención y se aprende mejor.
3. El pensamiento progresa a lo largo de una serie de etapas para adquirir un conocimiento donde tiene puntos débiles (No aprendidos).
4. La interacción social, especialmente cuando se centra en una experiencia física relevante, promueve el desarrollo intelectual.

La contribución de Piaget no ha consistido en elaborar nuevas ideas pedagógicas, sino en proporcionar un amplio conjunto de datos y de teorías que brindan una base sólida para una mejor orientación a las escuelas.

1.4.2. Enfoque de Collins y Duguid

Collins y Duguid propusieron un enfoque de aprendizaje cognitivo basado en:

- *Conocimiento de un dominio específico*. Incluye conceptos, hechos y procedimientos que identifican una asignatura determinada y que están contenidos en los libros, se explican en las clases y se ponen en práctica en distintos ejercicios.
- *Estrategias heurísticas*. Son técnicas y enfoques para la realización de tareas que pueden ser consideradas como “Trucos de la profesión”.
- *Estrategias de control*. Controlan el proceso mediante el cual se realiza una tarea. Incluyen componentes de autorregulación, análisis y corrección.
- *Estrategias de aprendizaje*. Son estrategias para aprender cualquiera de los componentes señalados anteriormente. El conocimiento de cómo se aprende va desde estrategias generales para explorar un nuevo dominio hasta estrategias más específicas, para ampliar o reestructurar los conocimientos en la solución de problemas o en la realización de tareas complejas.

1.4.3. Enfoque de Thorndike

Thorndike, formuló una teoría de conexión E-R (Estimulo-Respuesta) del aprendizaje: a cada estímulo E se asocia una respuesta R que, unidas, forman una conexión o lazo. En el proceso del aprendizaje, los nexos E-R se van estableciendo, fortaleciendo y organizando.

Estímulo es por definición cualquier cosa que pueda excitar uno de los mecanismos receptores del organismo, y respuesta es cualquier aspecto de la conducta con la que podamos trabajar.

Las tres leyes más importantes del aprendizaje propuestas por Thorndike son:

1. La *ley del efecto.* La conexión E-R se refuerza si sus consecuencias son satisfactorias o premiadas y se debilita si sus consecuencias son molestadas o se castigan.
2. La *ley del ejercicio.* Con la práctica o repetición un lazo o conexión se fortalece o del mismo modo, se debilita al no usarlo. Cuanto mayor es la frecuencia y más recientes son las acciones, se tiende a repetirlas y se fortalecen.
3. La *ley de la disposición.* Con la satisfacción y la recompensa como opuestas a la frustración y molestia dependen de que la persona que aprende esté o no preparado para hacerlo. Un lazo o conexión se fortalece o debilita según el ajuste mental del que aprende, y su preparación y disposición. La pérdida de la disposición no se recobra fácilmente.

Thorndike propuso también cinco leyes secundarias y éstas tienen relevancia para la educación:

1. El que aprende ha de ser flexible y variar su modo cuando trata de resolver un problema hasta que por ensayo y error consigue el éxito.
2. La actitud o forma mental del que aprende es significativa para determinar su conducta y éxito en el aprendizaje.
3. El que aprende selecciona los puntos fundamentales y esenciales de un argumento o problema y desecha los detalles irrelevantes.
4. El que aprende responde al identificar elementos comunes de ambas.
5. El que aprende responde a un estímulo dado, luego por asociación hace la trasferencia a estímulos totalmente diferentes.

Estas leyes fueron consideradas a pesar de su enfoque rígido y mecanicista del aprendizaje, en el que el ensayo, error, la práctica, el aprendizaje memorístico y la formulación de hábitos tienen más significado que la comprensión, la experiencia, la discriminación y la motivación.

1.4.4. Enfoque de Pavlov

El fisiólogo ruso Pavlov expuso la “generalización del estímulo”, que se basa en que un organismo puede condicionarse respondiendo a estímulos que son semejantes en carácter.

En el condicionamiento clásico, la respuesta o reflejo que provoca un estímulo son comúnmente conocidos como “conducta de respuesta”.

La memorización verbal de asignaturas académicas es un tópico apropiado para estudiar al abordar el aprendizaje, como lo es el crecimiento emocional y el desarrollo personal.

1.4.5. Enfoque de Skinner

El término técnico usado por Skinner para describir la recompensa es "refuerzo"; el refuerzo positivo incluye recompensas, el refuerzo negativo incluye el castigo. Naturalmente si no hay más recompensas, se tiende a disminuir su actividad y la respuesta desaparece gradualmente por un proceso llamado de "extinción".

Mientras que la actividad de refuerzo lleva a aprender y la de no refuerzo resulta en extinción, no son iguales ni opuestas porque cuando se recobra espontáneamente la actividad original se reaprende rápidamente y el olvido total es raro.

1.4.6. Reducción de necesidades de Hull

Hull puso gran énfasis en la ley del efecto y del refuerzo. Interpretó refuerzo en término de reducción de la necesidad.

La conclusión más importante a la que ha llegado la revisión de los principios educativos, que se ha estado efectuando en los últimos años, se expresa en términos de que el estudiante debe *aprender a aprender.*

Entre los aspectos más sobresalientes que comprende la expresión *aprender a aprender* están los siguientes:

- El estudiante, mediante su propia experiencia, tiene que determinar su capacidad y efectividad de trabajo.
- El estudiante debe familiarizarse, por propio esfuerzo, con las diferentes etapas que componen el proceso del aprendizaje de cualquier tema estructurado de la ciencia.

1.4.7. Conclusiones

Con base en todos los conceptos que explica cada especialista, se puede afirmar que el aprendizaje ocurre cuando el alumno siente la necesidad de aprender, cuando siente un uso intelectual o práctico para lo que hay que aprender.

Cuando el alumno tiene la libertad de explorar y de aprender lo que le interesa, puede poner en el aprendizaje toda su habilidad innata, su energía y su entusiasmo.

El aprendizaje y la memorización son resultado de la formación en la mente de asociaciones simples de palabras, ideas, objetos y conceptos: las experiencias más agradables se retienen más tiempo; las dolorosas y desagradables, se olvidan.

La visión constructivista de la educación, que entronca con el trabajo de Piaget, defiende que lo importante de la educación es ayudar a los estudiantes a construir su propia comprensión de las cosas.

John Dewey menciona que la actitud más importante que puede formarse es la actitud investigadora, el deseo de un continuo aprender.

1.5. LAS ESTRATEGIAS DE APRENDIZAJE Y LAS COMPUTADORAS

La experiencia demuestra uno de los posibles usos positivos de las computadoras en la educación es la destrezas del más alto orden. El uso de la computadora, concebida como medio para fines educativos, puede ser:

- Una manera de despertar o activar ciertas operaciones mentales relevantes para el aprendizaje, como las estrategias cognitivas y afectivas.
- Una manera de enseñar ciertos sistemas de códigos, símbolos o procesos para que los alumnos puedan aprovechar mejor los mensajes de los dispositivos.
- Una manera de acceder a símbolos que pueden ser internacionalizados, organizados, esquematizados y utilizados como herramientas mentales, es decir, como estrategias cognitivas.

Características o formas para los programas que pretenden enseñar o desarrollar las estrategias de aprendizaje.

- Crear expectativas en el alumno y estimular su planificación.
- Despertar y distinguir la atención del alumno hacia los aspectos más relevantes el programa.
- Asegurar situaciones de aprendizaje significativas, es decir, episodios que contienen sentido y valor para el alumno.

Fomentar el procesamiento de nueva información, vía uso de preguntas, y mostrando una estructura bien desarrollada en redes apropiadas a la asignatura.

- Aprovechar las posibilidades de usar imágenes cuando son apropiadas.
- Pedir y desarrollar la comparación, la inferencia y el uso de reglas lógicas.
- Estimular y guiar la generalización y la transferencia de lo aprendido, vía diversos ejemplos prácticos.
- Modelar el uso de la resolución analítica y ofrecer diversos ejemplos y problemas para resolver.
- Ofrecer diversas situaciones de resolución de problemas.

- Proveer retroalimentación frecuente.
- Asegurar autoeficacia.
- Dosificar la cantidad de contenidos necesarios de acuerdo con las necesidades del aprendizaje.

1.6. CONTRIBUCIÓN PEDAGÓGICA DE LAS TECNOLOGÍAS DE COMPUTACIÓN

La función pedagógica o instruccional se refiere, a la relación que existe entre determinados componentes de la situación instruccional y los cambios de comportamiento que ocurren a nivel del estudiante, o sea, el aprendizaje.

1.6.1. Funciones pedagógicas del modo herramienta

La computadora actúa como un recurso de trabajo altamente eficiente para resolver determinado tipo de tarea.

Esa tarea podría llevarse a cabo también por medios humanos, pero a través de grandes esfuerzos y de un tiempo prolongado. La eficiencia adicional con que se logra la tarea varia de acuerdo con las características de cada medio, mientras más integrado sea el paquete, mayor será el número de funciones pedagógicas que puede desempeñar.

1.6.2. Funciones pedagógicas del modo comunicacional

La búsqueda, procesamiento y almacenamiento de información de todo tipo es una de las grandes ventajas del modo comunicacional.

Un usuario puede recibir a través de estos sistemas la información a base de un curso, y explorar por su cuenta los elementos adicionales que requiera para suplir lagunas o realizar sus tareas individuales.

Las formas de comunicación por medio de las cuales se puede aprender en base a la computadora son esquemas, iconos, animaciones y cuadros:

- De medio escrito:
  - Aprendizaje de información de todo tipo.
  - Desarrollo de la expresión.
  - Desarrollo de habilidades para el análisis y síntesis de textos.

- De la interacción y cooperación de los grupos:
  - Aprendizaje tutorial.
  - Apoyo motivacional de los estudiantes a distancia.
  - Desarrollo de un juicio crítico.
  - Solución participativa de problemas.
  - Oportunidades de aprendizaje incidental.

- De los medios audiovisuales
  - Valor motivacional añadido.
  - Sustitución de la experiencia directa.
  - Presentación de conocimientos abstractos mediante imágenes.

1.7. TIPOGRAFÍA

La tipografía se ocupa de estructurar y organizar el lenguaje visual. El diseño de tipos se ocupa de la creación de las unidades que deben organizarse, los caracteres que constituyen un tipo. El diseño de la forma del tipo y la tipografía están encaminados a transmitir un mensaje.

1.7.1. Normas generales de tipografía

Estas normas comunes son de aplicación en todos los textos propios de la publicación.

1. Los tipos de letra usados en las informaciones y títulos (Georgia, Eurostyle Roman y sus variedades) no podrán ser usados en la publicidad.
2. Los textos, ilustraciones o líneas no pueden sobrepasar los límites del margen de cada página, con excepción de las imágenes que se extiendan por una doble página o espacios publicitarios.
3. Las líneas de composición no sobrepasarán los 80 caracteres, incluyendo los espacios. Se exceptúan textos breves como pies de foto, sumarios, fotonoticias, etc.
4. El carácter visual de la publicación es la causa de que los titulares deban ser vistosos y de gran tamaño en el cuerpo tipográfico.
5. Números que faciliten al lector la identificación de una enumeración. Estos caracteres irán en negrita, y separados del texto por un espacio de cuarto de cuadratín.
6. Los ladillos, pies de páginas, fechas o firmas que empleen un tipo o cuerpo de letra distinto al del texto no necesitan signos de puntuación. Las sangrías se componen de dos cuadratines en blanco. El primer párrafo de cada texto no se sangra, al igual que los que siguen a un ladillo.
7. Los ladillos se emplazarán entre la cuarta línea de cada columna y la antepenúltima, para no ir al principio ni al final de cada bloque de textos.
8. Los textos compuestos en bandera a la izquierda, derecha o ambas no parten palabra a final de línea. Los titulares o ladillos no pueden llevar palabras partidas a final de líneas.
9. La partición de palabra a final de líneas es obligatoria en todos los demás textos de la publicación.
10. En las citas no se debe emplear “que” para introducirlas en estilo directo. Para expresar las palabras que tal como fueron dichas se usan signos ortográficos como los dos puntos y las comillas. En el estilo indirecto no se usan estos signos, y se introducen las citas mediante el “que”.
11. El uso de la cursiva se limita a casos especiales. No se debe abusar de esta variedad tipográfica.

12. Los títulos de discos, libros, películas, obras de teatro, publicaciones, etc. se escribirán en cursiva, además irán también en cursiva los títulos de parte de éstos, como canciones, capítulos, poemas, secciones, artículos, etc. En caso de aparecer en el mismo párrafo, el título de la obra irá en cursiva y el de la parte irá entre comillas simples. Por ejemplo: la canción `she came in through the bathroom window´ forma parte del disco Abbey Road de los Beatles.
13. Neologismos. Si su uso es muy reciente y su significado no está claro para buena parte de la audiencia.
14. Nombres. Se usa para los apodos si acompañan al nombre real. En caso contrario, el apodo sigue las directrices para nombres propios. Por ejemplo, bruce Springsteen, The Boss. Para diminutivos no se usará la cursiva.
15. No se usará la cursiva para seguidores de un determinado grupo, estilo, miembro de una tribu urbana, etc. Si el término no tiene un uso muy extenso, se explicará en su primer uso. Por ejemplo: los bakalas se dirigen a la discoteca; los raperos siguen ciegamente los principios de incorrección.
16. Los nombres comunes de lugares o accidentes geográficos van en redonda, incluso si son extranjeros: Calle de Venecia, Plaza Nueva, Picaddilly Circus, Leicester Square, etc.
17. Marcas. No se escribirán en cursivas, pero sí aquellos ejemplos de productos concretos conocidos por su singularidad. Por ejemplo, un avión DC-10 pero el Enola Gay.
18. Titulares. En críticas, artículos de opinión, editoriales, etc. Se diferencian así de los títulos de textos informativos o interpretativos, que irán en redonda.
19. Siglas. Irán siempre en redonda, aún cuando su forma extendida haya de ir en cursiva o se encuentren dentro de un enunciado en cursiva.
20. Los signos de puntuación inmediatos a una palabra en cursiva usarán también esta variedad. Así, siguiendo a una palabra cursiva en tamaño 10, se empleará una coma en cursiva a tamaño 10.
21. En el cuerpo de texto se emplea para los nombres propios de personas o grupos, en la primera vez que a parece en cada texto.
22. La entradilla de un artículo o reportaje irá también en cursiva, en la misma fuente del texto.

Algunos consejos para realizar presentaciones con la ayuda de un software adecuado son:

- Puntualizar conceptos de forma sencilla, clara y concisa.
- Limitar la información contenida a los puntos más importantes (el exceso de texto puede ser un distractor para el alumno).
- Utilizar y variar el uso de los recursos (audio, vídeos, imágenes, etc.).
- Desarrollar una estructura que sea utilizada a lo largo del curso.
- Estandarizar a lo largo de la presentación el tipo y tamaño de letra, los colores y posición de los elementos gráficos.
- Colores, tipos de letras, efectos: la regla es "NO ABUSAR". Lo mejor es escoger dos o tres colores, tipos de letras y efectos de animación, y usarlos a lo largo de la presentación. Que no parezca catálogo de fuentes o de colores. Se recomienda emplearlos según la estructura de la presentación

(títulos con determinado color y fuente, cuerpo de texto con otro color y fuente, conceptos importantes con otro tamaño, etc.).
- Utilizar la combinación de letras mayúsculas y minúsculas. El uso de las mayúsculas se recomienda sólo para palabras o frases muy cortas (texto destacado).
- Usar letras preferentemente planas y con un tamaño igual para el contenido de los temas (por ejemplo: Arial, Gill Sans, Helvetica, Univers, entre otras).
- Utilizar el estilo cursivo para resaltar palabras claves dentro del texto.
- Utilizar colores contratantes entre fondos y textos.
- Evitar, en lo posible, el uso de colores muy saturados (brillantes) que hacen difícil la legibilidad en los monitores (especialmente el rojo).

1.8. COLORIMETRÍA

Los colores poseen una fuerza psicológica. El color es un símbolo que expresa ideas, sentimientos y emociones.

El color es una sensación del ojo humano o del cerebro producida por estímulos luminosos en la retina o reacciones del sistema nervioso.

La mirada humana percibe la forma antes que el color, por lo tanto si con la forma logramos el primer impacto el color es fundamental para retener la atención y actuar sobre la capacidad reflexiva y emocional del individuo.

Los colores se agrupan en:
- Cálidos (rojos, amarillos, anaranjados, amarillos-verdosos), los cuales producen un efecto alegre y vivo. Siendo más estimulantes y excitantes a medida que se acercan al rojo.
- Fríos (azules, azules-verdosos y violetas), los cuales producen un efecto tranquilo, sedante, silencioso y fresco; y a medida que se acercan al negro, más fríos y deprimentes son.

Las luces de color pueden transferir mucha mayor viveza e intensidad a una superficie que si sobre esta misma superficie se aplican sustancias cubrientes para animarla. Al mismo tiempo, las luces sirven para cambiar las reacciones emotivas relacionadas específicamente con cada color.

El orden de visibilidad del rojo sobre otros colores es el siguiente:

1. Sobre azul
2. Sobre amarillo oro
3. Sobre naranja rosado
4. Sobre amarillo verdoso
5. Sobre gris

La clasificación de contrastes de un estudio inglés citado por Santiago de García, titulado de la escuela oficial de periodismo, es la siguiente:

1. Negro sobre blanco
2. Negro sobre amarillo
3. Rojo sobre blanco
4. Verde sobre blanco
5. Blanco sobre rojo
6. Amarillo sobre negro
7. Amarillo sobre violeta
8. Blanco sobre azul
9. Rojo sobre amarillo
10. Azul sobre blanco
11. Blanco sobre negro
12. Verde sobre rojo
13. Rojo sobre verde
14. Negro sobre rojo

1.8.1. Normas sobre los colores

1. No debe dedicarse una gran superficie a un color luminoso –amarillo, naranja- salvo que se desee atraer violentamente la atención.

2. No debe componerse con colores abigarrados. Normalmente basta con dos colores base: uno cálido y otro frío, uno más violento que el otro; los demás colores, muy disminuidos, se animaran por contraste con los colores base.

3. Las diferencias de luminosidad son más fuertes que las de tonalidad. (es decir, si queremos destacar algo lo lograremos mejor con un color de la misma familia muy claro, que con otro totalmente distinto pero igualmente oscuro. Por ejemplo destacara más un naranja claro sobre un rojo oscuro que un azul oscuro sobre ese mismo rojo.)

1.8.2. Lenguaje de los colores

- Verde. Comunica tranquilidad, sensación de descanso, esperanza e incluso euforia.
- Azul. Los colores cálidos se valorizan sobre él. En consecuencia resulta muy adecuado para fondos. La combinación de azul con blanco posee una visibilidad y una fuerza considerablemente impactantes.
- Violeta. Produce cierta sensación de incomodidad y melancolía. Puede resultar desagradable a la vista.
- Rojo. Es el color dominante que se puede combinar para producir efectos no tan violentos a la vista.

- Naranja. Cuando contiene poco rojo (calabaza) comunica cierta sensación de placidez a medida que aumenta en el rojo (anaranjado bermellón) la violencia se hace casi excesiva. Combinado con rojos y marrones produce armonías ricas y de fuerte impacto, pero que pronto pueden producir cansancio.
- Amarillo. Es un color cálido, alegra la vista. Comunica luz, y a la vez cierto sosiego de medio día.
- Marrón. Color de cierta aburrida solidez y contrapesa los colores activos, permitiendo que nuestra mirada descanse. Es adecuado para fondos.
- Gris. Es considerado como el típico color de fondo, puesto que cualquier color va bien sobre el. El gris claro que casi se confunde con el blanco suaviza el violento contraste de colores vivos sobre un blanco puro. Da mayor reposo y serenidad a nuestra composición.
- Negro. Cualquier color colocado sobre el negro aumenta su visibilidad, a su vez el negro, colocado sobre otro color, se aprovecha de la luminosidad de éste, teniendo inconvenientes como: letras muy delgadas en negro sobre un color luminoso puede resultar ilegible. No es recomendable para fondos.
- Blanco. En éste color como fondo los colores pierden luz y tono. El blanco no toma reflejos del color que esta junto a el, sino de su complementario: al lado del azul aparece anaranjado, al lado del rojo aparece verdoso.

CAPÍTULO 2



MARCO TEÓRICO E INGENIERÍA DE SOFTWARE

En un principio el desarrollo de Software se hacía sin usar metodologías o técnicas, esto debido a que los sistemas eran pequeños y de muy escasa complejidad.

A medida que avanzó la tecnología, las necesidades crecieron provocando que los sistemas de software no fueran de suficiente calidad, iniciándose una rama de la computación que se ha hecho indispensable en la última década, ésta es conocida como "Ingeniería de Software".

La cual ha ayudado a solucionar problemas en lo referente al análisis, diseño, desarrollo y mantenimiento de sistemas administrativos, los de apoyo a las tomas de decisiones o las de aplicaciones más comunes como son los de sistemas de monitoreo o los de control de procesos.

Para el desarrollo e implantación de tales sistemas es necesario realizar una serie de procesos que parten desde la comprensión del problema hasta la puesta en marcha del producto final.

En éste capítulo, se presenta el marco teórico y algunos de los paradigmas utilizados en el desarrollo de un software, tales como: El ciclo de vida clásico, Prototipo, Modelo en espiral y Técnicas de cuarta generación.

2.1. MARCO TEÓRICO

La forma para desarrollar software desde que se inició la computación, a través de los años ha sido de forma lenta, en un principio se trató de aplicar el análisis y diseño de otras especialidades, dando como resultado que el software fuera de mala calidad.

Pasaron varios años antes de que los desarrolladores de software, se dieran cuenta que se requería de métodos y/o técnicas que ayudaran éstos a crear un software de mejor calidad.

“Desgraciadamente el avance tecnológico es tan rápido que superó los avances alcanzados y pronto esas técnicas pasaron a ser obsoletas, por ésta y otras causas hubo la necesidad de crear una nueva rama de la ingeniería, la Ingeniería en Software". [2]

Esta ingeniería, “cada año es actualizada para poder cumplir con los requisitos que el hardware solicita, a pesar de esto aún no ha sido posible desarrollar software que explote al máximo el hardware, han aparecido nuevas especialidades como "Ingeniero en Bases de Datos", tratando de alcanzar el avance de la tecnología, exigiendo que los estudiosos se actualicen para que esta meta se llegue a conseguir”. [2]

La Humanidad ha entrado a la Era de la Información y el Conocimiento y la Educación no queda excluida de esta realidad.

La informática educativa es un nuevo campo que emerge de la interdisciplina que se da entre la Informática y la Educación para dar solución a tres problemas básicos:

- Aplicar Informática en Educación.
- Aplicar Educación en Informática.
- Asegurar el desarrollo del propio campo.

Actualmente, la participación de la Informática en la Educación se ha extendido por todo el mundo, principalmente en el uso de sus herramientas tecnológicas más representativas, entre las que se encuentran todos los dispositivos que permiten manejar información como son: computadoras, televisores, videocaseteras, micrófonos, videocámaras, módems, scanners, impresoras, etc.

2.2. INGENIERÍA DE SOFTWARE

En la actualidad, la ingeniería de software comprende modelos conceptuales y herramientas de trabajo que hacen posible la solución de cualquier problema en el proceso de desarrollo y mantenimiento de software.

Algunas definiciones formales del concepto de Ingeniería de Software son las siguientes:

"La ingeniería de software es el establecimiento y uso de sólidos principios de ingeniería orientados a obtener software económico, fiable y que funcione de una manera eficiente en un equipo de cómputo." (1)

"Ingeniería de software trata del establecimiento de los principios y métodos de la ingeniería a fin de obtener software de modo rentable que sea fiable y trabaje en máquinas reales." (2)

El uso de la Ingeniería de software es multidisciplinario, utiliza modelos matemáticos para el análisis de algoritmos, ingeniería para estimar costos y definir los métodos, herramientas y procedimientos para el desarrollo, administración para definir la planeación, organización, requerimientos, riesgos y control en los procesos.

Los objetivos que persigue la Ingeniería de software son:

- Mejorar la calidad de los productos de software.
- Aumentar la productividad y trabajo de los ingenieros de software.
- Facilitar el control del proceso de desarrollo del 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.

La ingeniería de software es una tecnología multicapa, Fig. 2.1. los enfoques de esta tecnología se basan en una organización de calidad, para mejorar los procesos y conducir un desarrollo más robusto que constituye la base de la Ingeniería de software.



Fig. 2.1 Tecnología multicapa.

La forma en que la Ingeniería de software nos apoya en la informática es conociendo sus cuatro elementos principales:

1. *Métodos*. Nos enseñan cómo construir técnicamente el software y abarcan los siguientes puntos:

   - Planificación y estimación de proyectos.
   - Análisis de requerimientos del sistema y del software.
   - Diseño de estructuras de datos.
   - Diseño de programas y procedimientos algorítmicos.
   - Codificación.
   - Prueba.
   - Mantenimiento.

2. *Herramientas***.** Son el soporte automático o semiautomático de aplicación de los métodos. Existen herramientas para cada uno de los métodos anteriores. Cuando la información creada por una herramienta puede ser utilizada por otra, el sistema establecido es llamado Ingeniería del Software Asistida por Computadora.

3. *Procedimientos.* Son la aplicación de los métodos utilizando las herramientas. También pueden ser los elementos de unión entre los métodos y las herramientas, estos procedimientos, definen la secuencia en la que se aplican los métodos.

4. *Enfoques de calidad***.** Nos indica la secuencia en que se aplicarán los métodos, procedimientos y herramientas.

Cuando todos estos procesos se integran se establece un sistema de soporte para el desarrollo de software llamado Ingeniería de Software Asistida por Computadora.

2.3. PARADIGMAS DE LA INGENIERÍA DE SOFTWARE

La ingeniería de software propone utilizar una metodología para el diseño de software que sea flexible, de bajo costo y confiable. A estas metodologías se les conoce como paradigmas y nos permiten situar el proceso de desarrollo del sistema en diferentes etapas, cada metodología tiene características diferentes ya que se emplean de acuerdo a la naturaleza del proyecto.

En la implementación del Software, uno de los logros ha sido identificar de manera precisa la forma que adopta el proceso de desarrollo, donde los modelos como: Prototipo, Espiral, Técnicas de Cuarta Generación y los de Ciclo de Vida (Clásico, Modificado, en Cascada y en V); plantean el desarrollo y explotación de una aplicación de Software.

2.3.1. El Ciclo de Vida Modificado

Este modelo nos proporciona más flexibilidad entre cada una de las fases, permitiendo interacciones entre ellas, según sean necesarias. Se efectúan cambios en los requerimientos y se disminuyen los efectos de errores en cada etapa del desarrollo.

Abarca toda la vida del sistema, comenzando con su concepción por medio del análisis y finalizando con el mantenimiento

2.3.1.1. Elementos del Ciclo de Vida Modificado

Está formado por las siguientes fases:

1. *Análisis*. Consiste en analizar las necesidades de los usuarios potenciales del software para determinar qué debe hacer el sistema a desarrollar, y de acuerdo con ello, escribir una especificación precisa de dicho sistema.

   Este planteamiento es necesario porque el software tiene que interactuar con el hardware, con personas y posiblemente con bases de datos o alguna otra fuente de información.

2. *Diseño*. Consiste en descomponer y organizar el sistema en elementos componentes que puedan ser desarrollados por separado para aprovechar las ventajas de la división del trabajo y poder hacer el trabajo en equipo. El resultado del diseño es la colección de especificaciones de cada elemento componente, a fin de obtener la calidad requerida antes de que se comience con la codificación.

3. *Codificación.* En esta fase se programa cada elemento componente por separado (se escribe el código fuente de cada uno). Se harán pruebas y ensayos para garantizar en lo posible que dicho código funciona.

4. *Integración* y prueba del sistema completo. Se debe de ir combinando todos los elementos componentes.

5. *Mantenimiento*. Durante la explotación del software es necesario realizar cambios ocasionales, para corregir errores no detectados con anterioridad o para introducir mejoras en él, teniendo en este caso que rehacer parte de los trabajos anteriores.

2.3.1.2. Diagrama del Ciclo de Vida Modificado

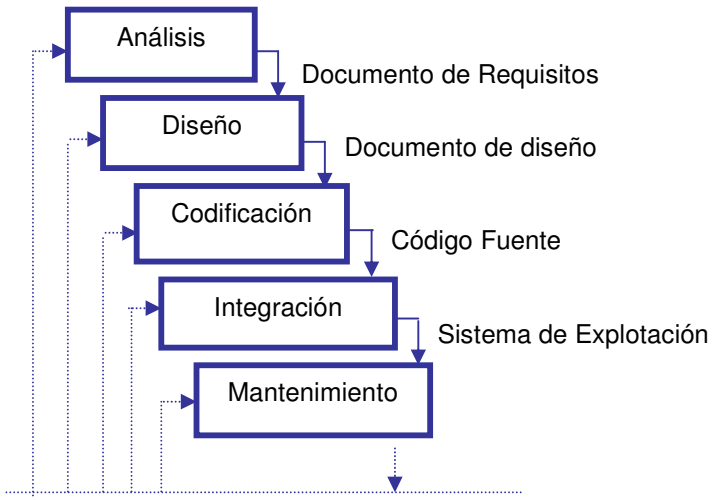


Fig. 2.2 Modelo de ciclo de vida modificado.

Como se aprecia en la figura 2.2, es notable que los pasos de este modelo sean muy similares a los pasos genéricos aplicables a todos los paradigmas de la ingeniería de sistemas por lo que resulta ser uno de los más completos.

2.3.1.3. Ventajas del Ciclo de Vida Modificado

El modelo de ciclo de vida modificado presenta las siguientes ventajas:

- Se tiene un seguimiento secuencial del proyecto, esto permite tener un orden durante el desarrollo del software.
- Es el modelo más completo de todos los modelos pertenecientes al ciclo de vida, ya que permite regresar a alguna de las fases que lo conforman, siempre y cuando se haya pasado una vez por todos los niveles.
- Permite acceder a algún nivel específico si se requiere, para hacerle modificaciones cuando el sistema está terminado.
- Se desarrolla un sistema que por las diferentes etapas en las que estuvo envuelto, promete ser el óptimo y el adecuado para las necesidades del cliente.

2.3.1.4. Desventajas del Ciclo de Vida Modificado

Pero como todo modelo, también cuenta con sus desventajas como las siguientes:

- Siempre ocurre la iteración y crea problemas en la aplicación de esta metodología.
- Una versión desarrollada del programa no estará disponible hasta mucho tiempo después de haber iniciado el proyecto, por lo que el cliente debe esperar.

2.3.2. El *ciclo de vida en V*

El modelo en V es análogo al modelo en cascada, pero se da especial importancia a la visión jerarquizada que se va teniendo de los distintas partes del sistema a medida que se avanza en el desarrollo.

2.3.2.1. Elementos del ciclo de vida en V

Éste modelo, tiene los mismos elementos de los modelos pertenecientes al ciclo de vida, pero a excepción de la última fase: análisis, diseño, codificación, integración y explotación.

En las líneas horizontales se representa el avance en el desarrollo y hacia arriba, corresponde al nivel de detalle con el que se trabaja en cada fase.

2.3.2.2. Diagrama del ciclo de vida en V

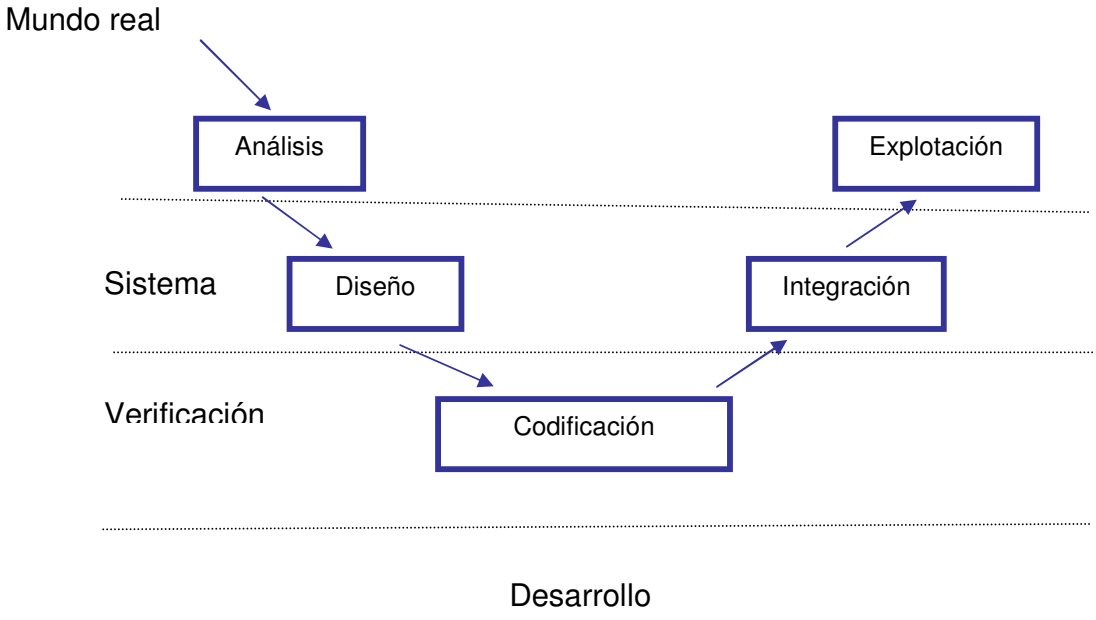


Fig. 2.3 Modelo de ciclo de vida en V.

2.3.2.3. Ventajas del ciclo de vida en V

El ciclo de vida en V presenta las siguientes ventajas:

- Se tiene un seguimiento secuencial del proyecto, esto permite tener un orden durante el desarrollo del software.

- Las fases iniciales en la rama izquierda son descendientes (se van descomponiendo en elementos cada vez más sencillos hasta llegar a las sentencias del lenguaje de programación).
- Se pueden establecer en un modelo en V el número de fases que se considere necesario.

2.3.2.4. Desventajas del ciclo de vida en V

El ciclo de vida en V presenta las siguientes desventajas:

- Se tiene un crecimiento incontrolable, debido a que se pueden añadir el número de fases requeridas.
- Una falla o error detectado hasta el final del desarrollo y no durante el mismo, puede ser desastroso.
- El desarrollo de un proyecto raramente sigue la secuencia que el modelo propone.
- No se puede acceder a una fase específica sin tener que pasar por todos los niveles que conforman éste modelo.

2.3.3. El Ciclo de Vida Clásico

Éste paradigma es el más antiguo y más ampliamente usado en la ingeniería del software.

2.3.3.1. Elementos del Ciclo de Vida Clásico

Esta formado por las mismas etapas que el modelo modificado como se aprecian en la figura. 2.4 y son: Ingeniería del Sistema, Análisis, Diseño, Codificación, Prueba y Mantenimiento.

2.3.3.2. Diagrama del Ciclo de Vida Clásico

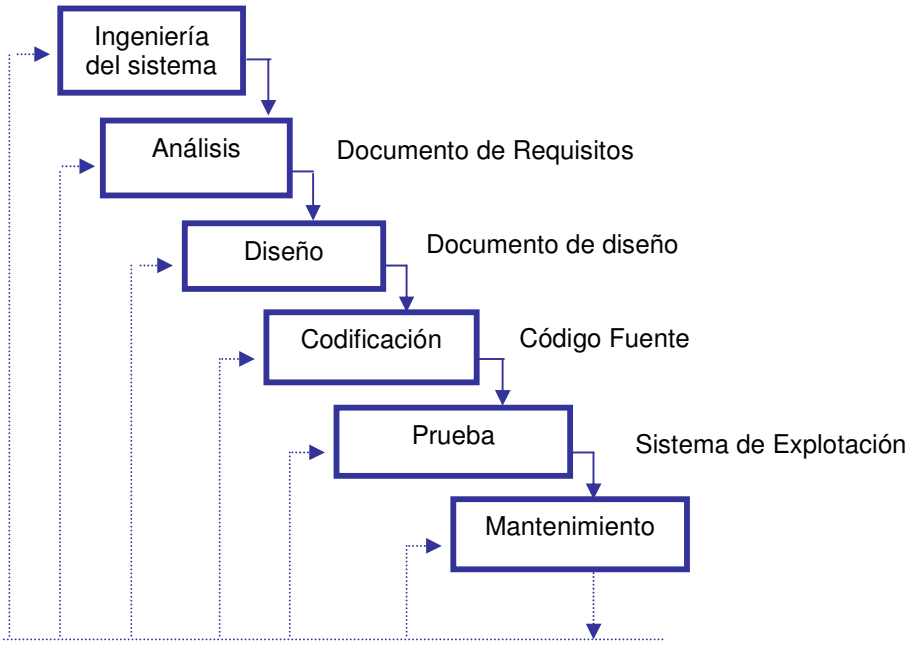


Fig. 2.4 El Ciclo de Vida Clásico

2.3.3.3. Ventajas del Ciclo de Vida Clásico

El modelo de ciclo de vida clásico presenta las siguientes ventajas:

- Se tiene un seguimiento secuencial del proyecto, esto permite tener un orden durante el desarrollo del software.
- Se desarrolla un sistema que por las diferentes etapas en las que estuvo envuelto, promete ser el óptimo y el adecuado para las necesidades del cliente.

2.3.3.4. Desventajas del Ciclo de Vida Clásico

Pero como todo modelo, también cuenta con sus desventajas como las siguientes:

- Siempre ocurre la iteración y crea problemas en la aplicación de esta metodología.
- Una versión desarrollada del programa no estará disponible hasta mucho tiempo después de haber iniciado el proyecto, por lo que el cliente debe esperar.
- Se deben tener todos los requisitos específicos del cliente al iniciar el desarrollo del software.

2.3.4. El Modelo en Espiral

Este modelo contiene las mejores características del ciclo de vida clásico y de prototipo, además de contar con el análisis de riesgos.

2.3.4.1. Elementos del Modelo en Espiral

Como se aprecia en la figura 2.5, en cada ciclo de la espiral (toda una vuelta en la espiral) se realiza una parte del desarrollo total, siguiendo la secuencia de las fases indicadas:

1. *Planificación.* Establece el contexto del desarrollo y decide qué parte del mismo se abordara en ese ciclo de la espiral.
2. *Análisis de riesgo*. Analiza alternativas para contar con menores riesgos en el desarrollo del software, así como la identificación y resolución de los mismos.
3. *Evaluación.* se analizan los resultados de Ingeniería generalmente con el cliente.

De la figura 2.5 se tiene:

1. Análisis de Riesgos.
2. Diversos Prototipos.
3. Simulación y Modelos
4. Concepto de Operación
5. Requerimientos de Software. Validación de Requerimientos.
6. Diseño, validación y verificación de Software.
7. Detalles de Diseño.
8. Implementación del Código.
9. Diversos test para el Código (unificación, integración, aceptación e implementación).
10. Planes de integración, requerimientos y ciclos de vida.

2.3.4.2. Diagrama del Modelo en Espiral

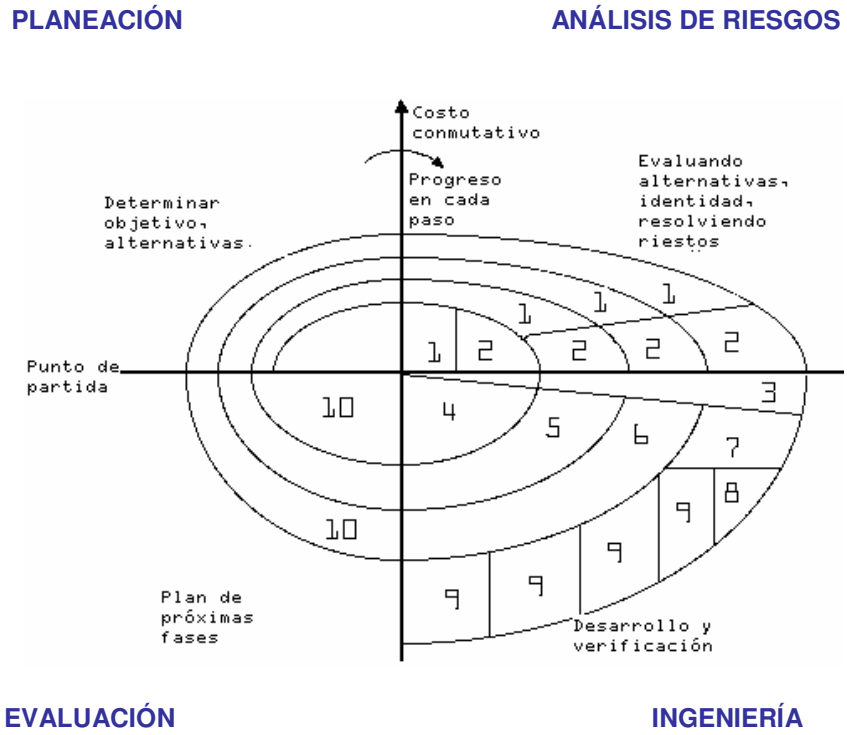


Fig. 2.5 Modelo en Espiral

2.3.4.3. Ventajas del Modelo en Espiral

Algunas ventajas que presenta el modelo en espiral son las siguientes:

- La identificación de riesgos asociados con cada una de las alternativas, y las distintas maneras de resolverlo (Estos son el centro del modelo).
- La división de los Proyectos implica que existe un acuerdo para los cambios que hay que realizar para terminar el proyecto en función de lo que se hace, aprendiendo desde el principio del proyecto.

2.3.4.4. Desventajas del Modelo en Espiral

El ciclo de vida en V presenta las siguientes desventajas:

- Es difícil convencer al cliente de que el enfoque evolutivo es controlable.
- Requiere una considerable habilidad para la valoración de riesgos.
- Si no se delimita bien el número de iteraciones, la espiral crecerá de manera desordenada y sin saber en qué momento se detendrá.

2.3.5. Prototipo

Esta metodología presenta un diseño rápido sobre la representación de los aspectos del software visibles al usuario, el modelo que se crea puede diseñarse de la siguiente forma:

- Un prototipo en papel o un modelo basado en computadora que describa la interacción hombre-máquina.
- Un prototipo que implemente algunos subconjuntos de funciones requeridas en el programa.
- Un programa existente que ejecute parte o toda la función deseada, pero que tenga características que deban ser mejoradas en el programa final que se dará al cliente.

2.3.5.1. Elementos del Prototipo

Como en todos los métodos de desarrollo de software, la construcción de prototipos comienza con la recolección de requisitos. Luego se produce un diseño rápido, el cual muestra al usuario los métodos de entrada, los formatos de salida y las diversas funciones del programa, según los requisitos.

A partir de este diseño se construye el prototipo, el cual es evaluado por el cliente, y con base en estas evaluaciones mejora el software. De esta manera, se genera un proceso interactivo en el que el prototipo es mejorado una y otra vez hasta que satisfaga las necesidades del cliente.

Este paradigma de prototipo se utiliza muchas veces con la finalidad de establecer un conjunto de requerimientos formales, que más adelante son traducidos en la producción de paradigmas mediante el uso de métodos y técnicas de ingeniería de programación.

2.3.5.2. Diagrama del Prototipo

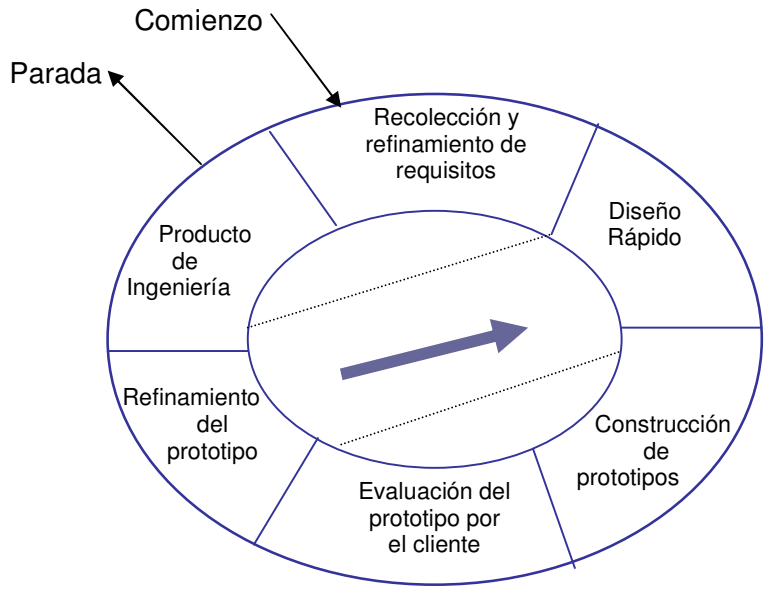


Fig. 2.6. Prototipo

2.3.5.3. Ventajas del Prototipo

- El cliente puede ver a corto plazo una primera versión de su sistema.
- Al tener un prototipo del proyecto del software, se pueden detectar con mayor facilidad las características faltantes, proponerse nuevas ideas, mejorar el diseño y en general ampliar la perspectiva del proyecto que se quiere desarrollar.

2.3.5.4. Desventajas del prototipo

- El cliente ve una versión del sistema en la cual no se consideran los aspectos de calidad ni mantenimiento del software a largo plazo.
- Cuando se le informa que el producto debe ser reconstruido, el cliente cree que presenta fallas porque no comprende que haciendo pequeñas modificaciones al prototipo, éste se ajustará a las nuevas especificaciones y será puesto en ejecución.

2.3.6. Técnicas de Cuarta Generación

Las técnicas de cuarta generación (T4G), son aquellas herramientas de software que facilitan al que desarrolla el software, la especificación de alguna característica del software a alto nivel. Además de que la herramienta genera de forma automática el código fuente.

Un entorno para el desarrollo de software que soporte las T4G, puede incluir al menos algunas de las siguientes herramientas:

- Generación de informes.
- Manipulación de datos.
- Interacción y definición de pantallas.
- Generación de código.
- Facilidades gráficas de alto nivel.
- Facilidades de hoja de cálculo.

2.3.6.1. Elementos de las técnicas de cuarta generación

Como se muestra en la Fig. 2.7, los componentes de T4G son:

1. *Recolección de requisitos.* En teoría, el cliente debe describir los requisitos, para que posteriormente se pueda construir un prototipo operativo.
2. *Estrategia de diseño.* Se desarrolla una estrategia de diseño para el sistema (como en algunos de los paradigmas anteriores), incluso si se utiliza las T4G. El uso de T4G sin diseño causará efectos como: poca calidad, mantenimiento pobre, mala aceptación por el cliente, etc.
3. *Prueba.* Para transformar un T4G en un producto, se debe hacer una prueba completa, documentación completa y debe ser construido el software de tal forma que permita un mantenimiento sin complicaciones.

2.3.6.2. Diagrama técnicas de cuarta generación

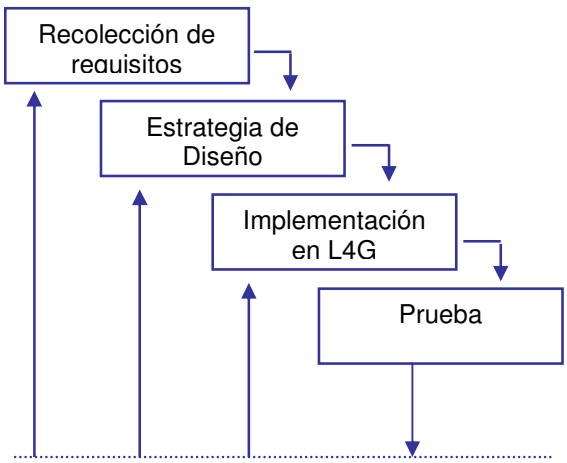


Fig. 2.7. Técnicas de cuarta generación

2.3.6.3. Ventajas de las técnicas de cuarta generación

- En general se reduce el tiempo de desarrollo del software y mejora en la productividad de las personas que construyen el mismo.
- Creación de un software mediante el uso de más de una herramienta de T4G y que como resultado proporcione un software atractivo, fácil de manejar y eficiente.
- El uso de T4G para grandes trabajos de desarrollo de software requiere el mismo tiempo que para trabajos chicos o medianos, pero el tiempo perdido, que se compensa mediante la codificación.

2.3.6.4. Desventajas de las técnicas de cuarta generación

- Algunas de las herramientas de T4G, no son más fáciles de utilizar que los lenguajes de programación.
- El mantenimiento de grandes sistemas de software desarrollados mediante T4G en ocasiones es cuestionable.

2.4. SELECCIÓN DEL MODELO

Para seleccionar un paradigma, se debe de tomar en cuenta el problema que se desea resolver y el modelo (o modelos) que nos ofrezca la solución adecuada.

El mismo problema se puede resolver por diferentes paradigmas, pero solo uno de ellos cubrirá nuestras necesidades y ofrecernos algunas ventajas para facilitarnos el desarrollo.

En la figura 2.8, se presenta una tabla en la que se comparan algunos de los elementos de los paradigmas mencionados anteriormente: Ciclo de vida clásico, Prototipo, Espiral y Lenguajes de cuarta generación.

En el modelo en espiral, se recomiendan darle tres vueltas como mínimo esto lo convierte en un modelo impráctico para el desarrollo de nuestro software, además un punto importante es que el cliente proporciona los requerimientos a lo largo del desarrollo del modelo, por todo eso, no creemos conveniente utilizarlo.

El prototipo no lo consideramos como la solución óptima, ya que es utilizado cuando se tiene poco tiempo y se requiere una interfaz gráfica, es decir, presentar físicamente un modelo del proyecto a desarrollar pero este se desecha y se reinicia con los pasos del modelo.

El modelo de Ciclo de Vida no es el apropiado porque limita la corrección de posibles errores. Es necesario tener los requerimientos al inicio y pasar por cada etapa del modelo solo una vez.

El modelo de Técnicas de Cuarta Generación, podría ser una de las soluciones, porque utilizaríamos software específico que reduciría el tiempo de desarrollo. Se convierte en un paradigma impráctico porque además de aprender a programar en los lenguajes propios de éste modelo es recomendable que en conjunto se trabaje con otro de los mencionados.

El modelo de Ciclo de Vida se adecua a una de nuestras necesidades ya que el cliente debe proporcionar desde el inicio los requerimientos, pero el problema que presenta es que el software tendría que quedar funcionando al haber pasado por cada una de sus etapas porque no existe la posibilidad de regresar a alguna de ellas.

En el modelo de Ciclo de Vida en V, los niveles de este paradigma cubren casi todas las etapas de los modelos vistos anteriormente, permite la corrección de posibles errores en máximo dos etapas anteriores y el cliente debe proporcionar los requerimientos del sistema antes de comenzar su desarrollo.

Analizando las ventajas y desventajas de cada modelo, decidimos utilizar el Ciclo de vida Modificado para el desarrollo del software.

2.5 Comparativo de Paradigmas

Etapa	Ciclo de Vida Clásico	Prototipo	Espiral	T4G	Descripción
Seguimiento secuencial	Si	Si	Si	Si	Al terminar una etapa pasa a la consecutiva
Requerimientos del software	Todos únicamente al inicio	Al inicio y después verlo	Inicio e intermedio	Inicio (idealmente)	Necesidades del cliente
Etapa de riesgos	No	No	Si	No	Posibles problemas de costos ambos y de software
Etapa de ingeniería	Si (recolección y refinamiento de requisitos)	Si (recolección y refinamiento de requisitos)	Si	No	Desarrollo del producto de “siguiente nivel”
Etapa de análisis	Si	No	Si	No	Requisitos del software (función, rendimiento, interfaces, etc.)
Etapa de diseño	Si	Si (rápido)	Si	Si	Los requisitos se traducen en representación visual del software
Etapa de codificación	Si	A veces	Si	No	El diseño en lenguaje de máquina
Etapa de prueba	Si	No	Si	No	Detectar posibles errores y verificar el funcionamiento del software

Etapa de mantenimiento	Si	No	Si	No	Método aplicado para que el software siga operando adecuadamente
Acceder a un nivel específico	Si	No	No	No	Ingresar a un nivel requerido del paradigma
Crecimiento controlable	Si	Si	Si	Si	Una vez que se tienen todos los requerimientos (o la mayoría) del cliente y un buen diseño
Construcción prototipo	No	Si	Si	Si (operativo)	Una primera versión del software (papel, visual, maqueta)
Reducción de tiempos en el desarrollo	No	No	No	Si	Tiempo transcurrido al desarrollar el software
Mejora en la productividad del personal	No	No	No	Si	Mayor número de actividades realizadas por desarrolladores del software
Evaluación del software por el cliente	Al final	Al final	Durante el desarrollo del software	Al final	Verificación de los requisitos y eficiencia del SW

Fig. 2.8. Tabla comparativa de paradigmas

CAPÍTULO 3



ANÁLISIS PREVIO AL DISEÑO DEL SISTEMA

En este capítulo, se muestran las fases que se recomiendan seguir en el desarrollo de un sistema, tales como son la validación, las pruebas, el análisis del sistema, el análisis de los requerimientos, diccionario de datos, etc.

También se presenta en el análisis y las especificaciones estructuradas el diagrama de flujo de los datos con sus elementos: el proceso, el flujo, el almacén y el terminador.

Para la creación de tablas de las bases de datos, se necesita del diagrama de entidad-relación con sus cuatro componentes principales: tipos de objetos, relaciones, indicadores asociativos de tipo objeto, indicadores de subtipo/supertipo.

3.1. FASES DE UN SISTEMA

Para el desarrollo de un proyecto de software, se deben considerar las siguientes etapas:

- Definición del sistema
- Desarrollo
- Validación, pruebas y mantenimiento
- Implantación y evaluación
- Análisis del sistema
- Análisis de requerimientos

3.1.1. Definición del sistema

Esta etapa inicia cuando se identifica un problema, se obtienen y se definen los requisitos principales del sistema y del software, así como las necesidades, alcances y limitaciones del proyecto. Selección de la metodología a seguir y las posibles herramientas de apoyo para su realización.

Al identificar un problema se debe pensar en qué información se necesita procesar, cómo se requiere y a qué necesidades debe dar solución. Para identificar estos aspectos es necesario tomar en cuenta los siguientes pasos:

1. Desarrollar un enunciado del problema por resolver, en el cual, debe emplearse terminología del área a la que pertenezca el sistema que se requiere desarrollar. También se debe incluir una descripción de la situación actual, restricciones del problema y de las metas que se lograrán.

2. Justificar una estrategia de solución computarizada para el problema.

3. Identificar las funciones por realizar, las restricciones, el subsistema de equipo electrónico. El subsistema del producto de programación y el de personal.

4. Determinar los objetivos y requisitos en el nivel del sistema para el proceso de desarrollo y los productos finales.

5. “Establecer criterios de alto nivel para la aceptación del sistema” [1].

3.1.2. Fase de desarrollo

En el proceso de desarrollo de software, las necesidades del usuario son traducidas en requerimientos de software, estos requerimientos transformados en diseño y el diseño implementado en código; el código es probado, documentado y certificado para su uso operativo.

Se define quién está haciendo qué, cuándo hacerlo y cómo alcanzar un cierto objetivo, un plan a seguir, se especifica las características principales y se fundamentan las herramientas a utilizar.

En esta fase se realiza una lista de pasos llamado algoritmo en donde se va verificando que el problema se resuelve como se necesita. Es la primera parte más difícil del proceso de solución del problema, ya que se debe revisar que el algoritmo está correcto y se puede traducir a un lenguaje de programación antes de continuar. Se auxilia de técnicas como pseudocódigo y diagramas de flujo.

3.1.3. Validación, pruebas y mantenimiento

En esta etapa se prueba el programa completo y se verificar que trabaje como se esperaba. Se prueban cada una de las funciones, primero por separado y luego en conjunto; probar el programa completo con distintos conjuntos de datos de prueba, en caso de que haya errores repetir las pruebas hasta la satisfacción de los requerimientos.

Para poder asegurar que el software no tiene fallas, es necesario utilizarlo de forma experimental, es decir, que su funcionamiento sea de acuerdo con las especificaciones y en la forma en que los usuarios esperan que lo haga. Las entradas son un conjunto de datos de prueba para que realice el procesamiento y después se examinan los resultados.

Es fundamental que las pruebas realizadas al sistema, las generen personas ajenas al mismo, para asegurarse que las mismas sean completas e imparciales y por otra, que el software sea más confiable. Esto con la finalidad de que los desarrolladores puedan depurar los errores que arroje el sistema antes que se implante y dependan de él.

3.1.4. Implantación y evaluación

La implantación es el proceso de verificar e instalar el software desarrollado en equipos nuevos, además de entrenar a los usuarios, instalar la aplicación y construir todos los archivos de datos necesarios para su uso.

Los sistemas de información deben mantenerse siempre al día, la implantación es un proceso de constante evolución. La evaluación de un sistema se lleva a cabo para identificar puntos débiles y fuertes.

3.1.5. Análisis del sistema

El análisis de un sistema debe efectuarse para comprender lo que se quiere obtener como producto final. Cuando la entidad es algo físico podemos construir un modelo igual en cuanto a la forma, pero a una escala menor; sin embargo, cuando la entidad se va a construir es un software, el modelo debe ser diferente

por tratarse de modelar la información que opera en el software y las funciones que permiten que ocurran las operaciones del sistema.

3.1.6. Análisis de requerimientos

Durante el análisis de los requerimientos del software, se elaboran modelos del sistema a obtener. Los modelos se enfocan en lo que debe hacer el sistema, no en cómo lo hace. En los modelos pueden usarse notaciones gráficas que nos describen información, procesamiento, comportamiento del sistema y otras características con diferentes símbolos reconocibles; algunas veces los modelos pueden ser sólo texto.

El modelo ayuda al analista a interpretar la función, la información y el comportamiento del sistema, lo cual hace más sencilla y sistemática la función de análisis de requisitos. Se convierte en el centro de interés y clave para verificar consistencia de la especificación. Además, proporciona al diseñador una representación clara que le permite implementarla.

En el modelo de análisis se logran tres objetivos:

1. Proporcionar de manera clara y concreta lo que requiere el usuario (cliente).
2. Establecer un punto de partida para la creación de un diseño de software.
3. Definir un subconjunto del total de requisitos, que se puedan validar al obtener el software.

3.2. ANÁLISIS Y ESPECIFICACIÓN ESTRUCTURADA

3.2.1. Diagrama de flujo de datos

El diagrama de flujo de datos es un modelo que representa el flujo de la información y las transformaciones que se aplican a los datos al moverse desde la entrada hasta la salida.

Se puede utilizar el diagrama de flujo de datos para representar un sistema o un software a cualquier nivel de abstracción. Los diagramas de flujo de datos pueden ser divididos en niveles que representan un mayor flujo de información y un mayor detalle funcional.

Un diagrama de flujo de datos lo podemos tener en dos categorías:

- Nivel cero o Modelo fundamental o Modelo de contexto.
- Nivel uno.

“Un diagrama de flujo de datos de nivel cero”, “representa al elemento de software completo como una sola burbuja con datos de entrada y de salida representados por flechas de entrada y de salida, respectivamente.

Al particionar el diagrama de flujo de datos de nivel cero para mostrar más detalles aparecen representados procesos, burbujas, almacén de datos y caminos de flujo de información adicionales, lo que conlleva a un diagrama de flujo de datos de nivel uno, en este diagrama se pueden contener cinco o seis burbujas con flechas interconectadas, y cada uno de los procesos representados en el nivel uno es una subdivisión del sistema general". [3.1]

Para evitar construir diagramas de flujo de datos confusos es necesario tomar en cuenta cuatro componentes:

1. *El proceso.* Es representado frecuentemente por un círculo, un óvalo o un rectángulo como muestra la Fig. 3.1, representa las partes del sistema que realizan una función de transformación de entradas en salidas. Generalmente se nombran con una frase verbo-objeto:



Fig. 3.1 Símbolo de proceso en un diagrama de flujo de datos.

2. *El Flujo.* Es representado por una flecha que entra o sale de un proceso, como se muestra en la Fig. 3.2, se refiere al movimiento de bloques o datos de una a otra parte del sistema. Generalmente se nombran con el significado del paquete de datos que se mueve a lo largo del flujo. Puede existir flujo en ambas direcciones.

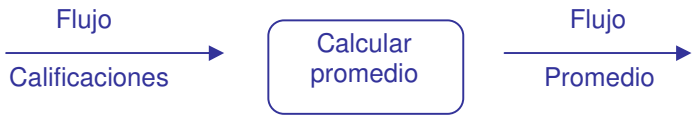


Fig. 3.2 Símbolo de Flujo en un diagrama de flujo de datos.

3. *El almacén.* La Fig. 3.3, muestra el símbolo con el cual se representa este elemento, se refiere a colecciones de paquetes de información en reposo como una base de datos. Generalmente su nombre es el plural del que se utiliza para los paquetes de datos que entran y salen del almacén por medio de flujos.



Fig. 3.3 Símbolo de almacén en un diagrama de flujo de datos.

4. *El terminador.* Representado por un rectángulo Fig. 3.4, se refiere a entidades externas con las cuales el sistema se comunica y que se encuentra fuera del control del sistema, puede ser otro sistema o el usuario. Generalmente se nombran con una palabra significativa para el usuario.



Fig. 3.4 Símbolo de terminador en un diagrama de flujo de datos.

3.2.2. Diccionario de datos

El modelo de análisis acompaña las representaciones de objetos (entidades) de datos, funciones y control; en cada representación, las entidades y/o elementos de control desempeñan un papel importante.

Es necesario proporcionar un enfoque organizado para representar las características de cada entidad de datos y elementos de control. Esto se realiza con el diccionario de datos; esta importante notación de modelado ha sido definida de la siguiente manera:

El diccionario de datos es un conjunto central de tablas y vistas de la base de datos de sólo lectura, contiene definiciones precisas que permite unificar criterios entre el usuario y el analista.

Un diccionario de datos proporciona la siguiente información:

- Nombre de los usuarios.
- Alias o nombre opcional.
- Privilegios (permisos) que tienen los usuarios.
- Nombre de los objetos (tablas, vistas, índices, sinónimos, clusters, secuencias, procedimientos, tigre, etc.).
- Defaults o información adicional.
- Espacio ocupado por los objetos.

Durante la operación de la Base de Datos, el DBMS (Sistema Manejador de Base de Datos) lee el diccionario de Datos (DD) para ver los objetos y ver quién los accede, así como para actualizar estructuras del desarrollo de la base de datos; ya que si un usuario crea una tabla, se generan registros de la nueva tabla en el diccionario de datos, sus columnas, sus segmentos y privilegios que el usuario tiene en la tabla.

3.2.3. Diagrama entidad-relación

El diagrama entidad-relación es un modelo que se basa en la percepción del mundo real y describe la distribución de los datos de un sistema.

Representa los objetos llamados entidades y las relaciones entre ellas. Además, indica la estructura lógica general de la base de datos de manera gráfica.

Hay cuatro componentes principales en un diagrama entidad-relación los cuales se listan a continuación:

1. *Tipos de objetos*. Se representa con una caja rectangular. Modela una colección o conjunto de objetos del mundo real cuyos miembros individuales tiene cada uno las siguientes características:

   - Identificarse de manera única por algún medio.
   - Juega un papel necesario en el sistema que se construye.
   - Puede describirse por uno o más datos, llamados atributos.

2. *Relaciones*. Una relación establece el conjunto de conexiones entre dos o más objetos, y se representa por medio de un rombo.

3. *Indicadores asociativos de tipo objeto.* Representa algo que funciona como objeto y como relación. Es algo acerca del cual se desea mantener alguna información.

4. *Indicadores de subtipo/supertipo*. Consisten en tipos de objeto de una o más categorías conectados por una relación. Los subtipos se conectan por medio de una relación sin nombre y el supertipo se conecta a la relación con una línea que contiene una barra. El supertipo denota los datos que se aplican a todos los subtipos.

Una vez concluido el modelo entidad-relación, se deben traducir las entidades a tablas, nombradas de acuerdo a los objetos que representan; los cuales poseen atributos (características propias).

Estas tablas se utilizan para guardar datos (campos y registros) relacionados al objeto, siendo cada columna un campo y cada fila un registro.

3.3. DISEÑO DE SOFTWARE

La etapa del diseño durante el desarrollo de un sistema es importante, ya que nos da un panorama del software (producto) a obtener, mediante las especificaciones y características deseadas y expuestas por el cliente. Sino contamos con un buen diseño, tenemos el riesgo de tener un sistema totalmente ineficiente.

El diseño estructurado es una excelente opción, ya que traduce los requerimientos del usuario a una representación gráfica del sistema, obteniendo un diagrama de módulos independientes que interactúan entre sí para lograr las funciones solicitadas por el usuario.

Algunas de las características relacionadas con la conceptualización del diseño son:

"Un diseño debe exhibir una organización jerárquica que haga uso inteligente del control entre los componentes del software.

Un diseño debe ser modular, esto es, el software debe estar dividido de forma lógica en elementos que realicen funciones y subfunciones específicas.

Un diseño debe contener representaciones distintas y separadas de los datos y de los procedimientos.

Un diseño debe llevar a módulos que exhiban características funcionales independientes.

Un diseño debe llevar a interfaces que reduzcan la complejidad de las conexiones entre los módulos y el entorno exterior.

Un diseño debe obtenerse mediante un método que sea reproducible y que esté reducido por la información obtenida durante el análisis de los requisitos del software" [3.2].

3.3.1. Modularidad

Esta característica se refiere a la división de los sistemas en partes pequeñas, a las cuales se les llama módulos, estos son independientes entre si por lo que pueden ser manejados por separado, siendo sencillo seguir el flujo de control de instrucciones al igual que la localización de las variables utilizadas.

Su objetivo es reducir el tiempo de generar programas y, en consecuencia, se obtiene un beneficio en el costo, ya que éste se ve disminuido.

La modularidad se logra mediante afinaciones constantes de los procedimientos verificando de lo general a lo particular.

El ocultamiento de la información se presenta cuando diseñamos de tal forma que, los procedimientos y datos que lo integran, sean inaccesibles por otros módulos, que no necesiten información.

Dentro del diseño estructurado existen los siguientes tipos de módulos:

1. *Secuénciales.* Su ejecución es sin interrupción aparente.
2. *Incrementales.* Su ejecución puede interrumpirse y reestablecerse en el punto donde se interrumpió.
3. *Paralelos.* Su ejecución es simultánea con otros módulos.

4. *Independientes.* Su ejecución es de forma separada a los otros módulos. La independencia funcional se da como resultado de la modularidad mediante el desarrollo de módulos con una función bien definida.

3.3.2. Cohesión

Es la medición de la fuerza de asociación de los componentes internos de los módulos y se puede ver como una extensión del ocultamiento de la información.

Nos sirve para verificar la fuerza funcional de un módulo individual al realizar una sola función dentro del sistema y después transmitir el resultado a otro módulo.

El tipo de cohesión puede ser distinto en función de la fuerza de unión de los componentes dentro del módulo siendo la mejor la más fuerte. A continuación se muestran de la más débil (la menos deseada) a la más fuerte (la más deseada).

1. *Cohesión coincidental.* Se presenta cuando los elementos internos de un módulo aparentemente no tienen relación, ya que ésta es muy débil.

2. *Cohesión lógica*. Existe una relación entre los elementos internos del módulo en cuanto a sus actividades, ya que una puede complementar o ser consecuencia de otra.

3. *Cohesión temporal*. Se da cuando se ejecutan en un momento dado todos los elementos de un módulo sin necesidad de un parámetro o lógica.

4. *Cohesión en la comunicación.* En este tipo, es independiente la función de cada elemento, se presenta si estas funciones son alimentadas por la misma entrada de datos.

5. *Cohesión secuencial.* En este tipo de cohesión cada salida de un elemento es la entrada de otro elemento de manera ordenada.

6. *Cohesión funcional*. Es la más deseada por su fuerte, la idea es de trabajo en equipo por parte de los elementos al estar enfocados a llevar a cabo la misma función.

7. *Cohesión informal.* Ocurre cuando el módulo contiene una estructura de datos compleja, en sí es la abstracción de dichos datos.

3.3.3. Acoplamiento

Es la medición de la fuerza de interconexión entre los módulos, la cual está influida por la complejidad de la interfaz y el tipo de comunicación que existe entre ellos; si es la mejor complejidad, la comprensión es más sencilla.

También podemos tener niveles de acoplamiento que se presentan a continuación del más fuerte (el menos deseable) al más débil (el más deseable):

1. *Acoplamiento por contenido*. Se da cuando un módulo modifica los valores o las instrucciones de otro módulo.

2. *Acoplamiento por zonas compartidas*. Se presenta si una zona es utilizada por varios módulos, debido a que se encuentran ligados mediante zonas globales.

3. *Acoplamiento por control.* Se presenta cuando un módulo controla la secuencia de otro por contener el paso de banderas de control y de parámetros en forma global.

4. *Acoplamiento por zona de datos .* De manera similar a la de las zonas compartidas, pero en este caso los elementos globales son compartidos en forma selectiva entre las diversas rutinas que requieren los datos.

5. *Acoplamiento por datos*. Ocurre cuando se tiene un módulo con la lista de parámetros, los cuales se pasan a los elementos de las rutinas. Este tipo de acoplamiento es el más deseado ya que si sufre variación solo se modifican los detalles internos de los módulos, sin tener que modificar las rutinas que usan los módulos modificados.

3.4. LA CALIDAD EN LOS SISTEMAS

La situación actual en el desarrollo de software, exige la creación de nuevos productos o servicios con un tiempo de desarrollo mínimo, pero con requisitos estrictos de calidad y seguridad.

Las técnicas de desarrollo de software utilizando prototipos, ayudan a reducir el tiempo de desarrollo, a realizar los sistemas de forma concurrente, y a validarlos desde un momento más temprano del ciclo de producción. De esta forma, se pueden cumplir los requisitos de calidad de los clientes y mostrarles una versión del sistema en un tiempo inmediato.

El aseguramiento de la calidad en el desarrollo y en la implantación de aplicaciones de software, amerita un análisis fino y un estudio detallado con el fin de prevenir o anticipar las posibles dificultades o problemas, que se puedan presentar en la realización del mismo.

En el proyecto de sistemas se comienza por la definición. Se continúa luego por el diseño o un plan que puede ser general o detallado, pasa a un desarrollo o bien a una adaptación de lo existente, se hace una prueba o bien una instalación y, finalmente, se hace una serie de revisiones después de la instalación.

Una desviación durante el proceso del desarrollo del sistema significa puntos menos en la calidad total del proyecto y obviamente en el nivel de satisfacción del usuario que disfrutará del desarrollo y de la aplicación que se le esté implantando.

3.4.1. Normas del aseguramiento de la calidad de los sistemas

Para garantizar que un desarrollo de software tenga calidad es necesario revisar todo el entorno del proceso del sistema para su realización, como son: la gente, los productos o herramientas a utilizar, el dinero con el que se cuenta y por supuesto el tiempo disponible.

Es necesario partir de unos integrantes sanos y de alta calidad que permitan obtener el más excelente desarrollo informático, como resultado de una buena preparación del producto desarrollado o instalado.

Si los desarrolladores no son los adecuados, o si los productos o herramientas de trabajo son limitados, poco confiables o bien no poseemos el suficiente presupuesto o el tiempo apremia y no es posible terminar todo lo que se necesita en el límite fijado, muy probablemente, cualquiera que sea el esfuerzo que hagamos por realizarlo, no obtendremos software de calidad.

Adicionalmente, para lograr una mejor calidad del producto de software se contemplan las siguientes actividades.

- Aplicar métodos y técnicas de control de calidad.
- Realizar revisiones técnicas formales.
- Apegarse a los estándares establecidos.
- Elaborar formas de control de cambios y modificaciones al sistema.
- Realizar pruebas al sistema.
- Realizar mediciones y estimaciones de calidad.
- Elaborar registros e informes concisos.

Elementos para lograr una calidad aceptable en el desarrollo de software.

- Métodos y herramientas de análisis, diseño, codificación y pruebas.
- Revisiones técnicas formales que se aplican en las etapas de desarrollo del proyecto.
- Elaboración de estrategias para realizar pruebas escalonadas.
- Control de la documentación de los cambios realizados al sistema.
- Procedimientos que fijen los estándares utilizados en el sistema.
- Mecanismos para medir la información.

3.4.2. Factores que afectan la calidad del software

1. *Corrección*. El grado en que un programa realiza satisfactoriamente las especificaciones establecidas por los clientes.
2. *Fiabilidad.* El grado en que se espera que los resultados del programa realicen sus funciones con la precisión requerida por el cliente.
3. *Eficiencia*. La cantidad de recursos de computadora y de código requeridos por el programa para llevar a cabo las funciones.
4. *Integridad*. El grado que se controla el acceso al sistema o a los datos por personal no autorizado.
5. *Facilidad de uso*. El esfuerzo requerido para aprender, trabajar, preparar la entrada e interpretar la salida de un programa.
6. *Facilidad de mantenimiento.* El esfuerzo requerido para localizar y arreglar un error en el programa.
7. *Facilidad de prueba*. El esfuerzo requerido para probar un programa de forma que se asegure que realiza las funciones requeridas.
8. *Portabilidad.* El esfuerzo necesario para transferir el programa desde un hardware y/o entorno de sistemas a otro.
9. *Reusabilidad.* El grado en que un programa (o parte de un programa) se pueden utilizar en otras aplicaciones.

3.4.3. Características métricas en la calidad del software

1. *"Facilidad de auditoría.* La facilidad con que se puede establecer con los estándares establecidos.
2. *Exactitud.* La precisión de los cálculos y el control del programa.
3. *Completitud*. El grado en que se ha conseguido la total implementación de las funciones requeridas.
4. *Concisión. Lo* compacto que es el programa en término de líneas de código.
5. *Consistencia.* El uso de un diseño uniforme y de técnicas de documentación a lo largo del desarrollo del proyecto.
6. *Estandarización en los datos.* El uso de estructuras de datos y de tipos estándar a lo largo de todo el programa.
7. *Tolerancia de errores.* El daño que se produce cuando el programa encuentra un error.
8. *Eficiencia en la ejecución*. El rendimiento en tiempo de ejecución de un programa.
9. *Modularidad.* La independencia funcional de los componentes del programa.
10. *Facilidad de operación.* La facilidad para operar el programa.
11. *Seguridad*. La disponibilidad de mecanismos para controlar o proteger los programas y la información.
12. *Simplicidad.* El grado en que un programa puede ser entendido si dificultad.
13. *Facilidad de seguimiento.* La posibilidad de seguir paso a paso las instrucciones del programa hacia atrás, hasta llegar a los requerimientos." [3.1]

3.5. PRUEBAS DEL SOFTWARE

La prueba es uno de los pasos de la ingeniería de software que define una serie de estrategias posibles para garantizar que, al usarse por primera vez el sistema, se encuentre libre de problemas.

Algunas veces las pruebas ocasionan que nos demos cuenta que no se tiene el producto esperado; sin embargo, son necesarias para obtener un software de calidad.

3.5.1. Características de las pruebas

En el proceso de aplicación de pruebas se pueden establecer algunas normas características para que el objetivo de la prueba se cumpla.

- La prueba es un proceso que se inicia a nivel de módulo y trabaja “hacia fuera”, es decir, hacia la integración del sistema.
- Se aplican diferentes tipos de prueba apropiados en cada caso para diferentes momentos.
- La prueba la lleva a cabo el desarrollador de software y en el caso de grandes proyectos la realiza un grupo de prueba independiente.
- Se realiza la prueba y depuración, aunque son actividades diferentes, es importante incluir la depuración en cualquier estrategia de prueba.
- Decimos que una prueba es exitosa cuando se tiene una alta probabilidad de mostrar algún error, que no se había descubierto hasta el momento.

3.5.2. Pasos en la prueba de los sistemas

Las pruebas constan de una serie de tres pasos, que incluyen la dirección a la cual se enfocan las pruebas y se ilustra en la Fig. 3.4.

“Un sistema de programación sigue los siguientes tres pasos:

1. *Prueba de unidad.* Se centra en cada módulo individual, asegurándose que cada uno de ellos funcione correctamente como unidad. Se utilizan caminos específicos de la estructura de control del módulo para asegurar un alcance completo y una detección máxima de errores.

2. *Pruebas de integración.* Se deben integrar los módulos para formar el sistema completo. Aquí la prueba se dirige a todos los aspectos asociados con el problema de verificar y construir el programa. Después de que el sistema se ha integrado, se realizan un conjunto de pruebas de alto nivel donde se verifican los criterios de validación.
   La prueba de validación debe proporcionar la seguridad de que el sistema satisface todos los requisitos funcionales, de comportamiento y rendimiento.

3. En el último paso de la prueba una vez validado el software, se debe combinar con otros elementos del sistema, como pueden ser el hardware, usuarios, base de datos. La prueba del sistema verifica que cada elemento encaja de forma adecuada y que se alcanza la funcionalidad y el rendimiento del sistema en su totalidad" [2.1]

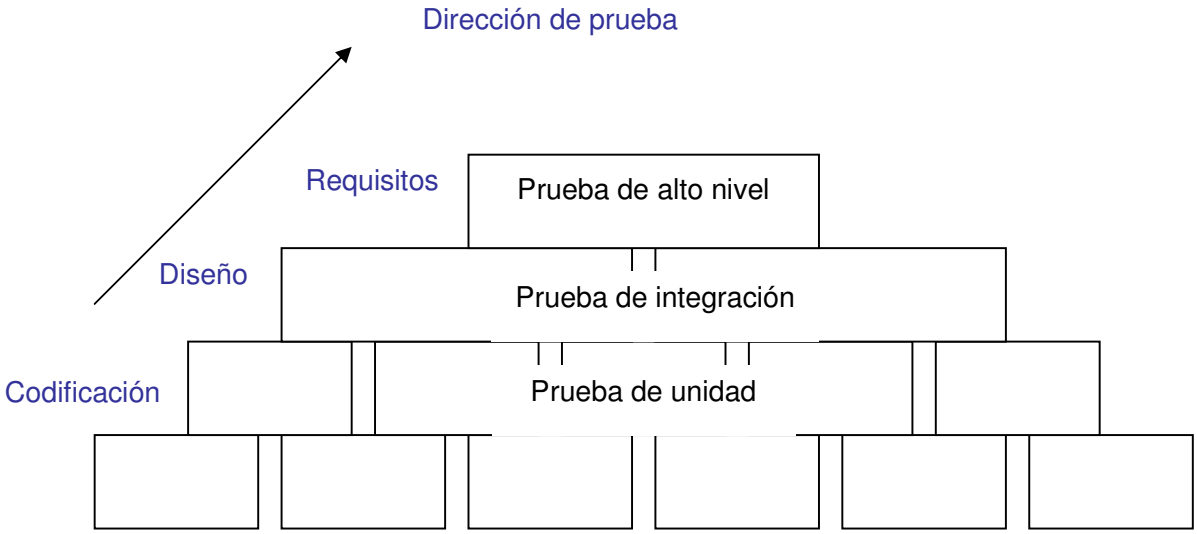


Fig. 3.4 Pasos de Pruebas

3.5.3. Pruebas de validación

Un software debe contemplar una prueba de validación al procesar datos y ejecutar funciones, de manera que podamos visualizar que el sistema arroja los resultados esperados y si no es así, revisar que sucede.

La validación del software se consigue mediante una serie de pruebas que demuestran la conformidad de los requisitos. Se determina un plan de prueba, el cual consiste en trazar las pruebas que se han de llevar a cabo.

Tanto el plan como el procedimiento estarán diseñados para asegurar que se satisfacen los requisitos funcionales, que se alcanzan todos los requisitos de rendimiento, que la documentación es correcta.

Una vez que se lleva a cabo cada prueba de validación, pueden ocurrir dos condiciones: las características de funcionamiento o de rendimiento están de acuerdo con las especificaciones y son aceptables, otra que se dan cuenta que las especificaciones no se cubren y se encuentran deficiencias.

Un elemento importante en el proceso de validación es el repaso de la configuración del sistema, en donde se intenta asegurar que los elementos que componen el software se han desarrollado de forma correcta, y que están bien detallados para facilitar la fase de mantenimiento que más adelante será necesaria.

También se realizan pruebas de aceptación, para permitir que el usuario final valide todos los requisitos.

Existen también pruebas llamadas alfa y beta, en los casos en que el software sea empleado por un grupo de usuarios; el objetivo de esta prueba es detectar errores que sólo el usuario final puede detectar.

La prueba alfa es realizada por un cliente en el lugar de desarrollo y un entorno controlado y la prueba beta se lleva a cabo en uno o más lugares en donde se encuentres los usuarios finales.

CAPÍTULO 4



ANÁLISIS DEL SISTEMA

En éste capitulo, se muestra el análisis del sistema que se desea crear, con base en las necesidades de los alumnos y Profesores, definimos lo que el sistema a desarrollar debe proporcionarles.

En el planteamiento del problema, se define la necesidad de crear un sistema el cual esté disponible y sea una herramienta de apoyo para los alumnos y Profesores de la Facultad de Ingeniería, específicamente para las asignaturas impartidas en la carrera de Ingeniería en Computación.

Se explican los siete módulos en los cuales se divide el sistema que son: administrador de contraseñas, creación y modificación de tutoriales, creación y modificación de cuestionarios, consulta de tutoriales, aplicación de exámenes, evaluación de exámenes, y consulta de exámenes.

4.1. PLANTEAMIENTO DEL PROBLEMA

En la Facultad de Ingeniería, por semestre se aplican generalmente tres exámenes departamentales a los alumnos inscritos en alguna asignatura del primero al cuarto semestre.

Esos exámenes son elaborados por un grupo de profesores de la Coordinación de cada asignatura con base al temario vigente, que cada profesor entrega a sus alumnos e imparte la asignatura con los temas contenidos en el mismo.

El alumno muchas ocasiones no cuenta con los conocimientos mínimos necesarios de las asignaturas, por esa razón no aprueba los exámenes. No cuenta con una herramienta que le sirva como apoyo para la comprensión, aprendizaje, reforzamiento y análisis de los temas vistos en clase, en el tiempo que requiera para realizarlo y que esa herramienta esté disponible en todo momento.

4.2. SOLUCIÓN

Por lo anterior, se pretende crear un sistema creador de tutoriales y evaluador de exámenes departamentales y extraordinarios, para fines prácticos, se toman las asignaturas impartidas en la carrera de Ingeniería en Computación de la Facultad de Ingeniería.

El objetivo del sistema a realizar es:

- Proporcionar a los alumnos una herramienta donde puedan encontrar toda la información de las asignaturas que cursen.
- Que la herramienta esté disponible en la Facultad de Ingeniería y fuera de ella.
- Brindarles a los profesores material adicional y básico para uso bibliográfico en la impartición de sus asignaturas.
- Los alumnos puedan realizar en el semestre que cursen, los exámenes departamentales y/o extraordinarios.

De esa forma se puede evaluar el conocimiento general de los alumnos, ya que para poder aprobar los exámenes deben contar con los conocimientos mínimos requeridos para ser ingenieros de calidad y poder resolver eficientemente los problemas a los que se enfrentarán en su vida profesional.

Para que los alumnos puedan encontrar los temas deseados de las asignaturas correspondientes a la carrera de Ingeniería en Computación, se crearán tutoriales que los contengan, basándose en los planes de estudios vigentes.

La solución que se encontró es:

Realizar un sistema Web para que pueda ser accedido desde cualquier lugar que cuente con acceso a INTERNET.

En dicho sistema se pretende que se puedan crear tutoriales por asignatura, a los cuales tengan acceso un grupo definido de profesores para su modificación y actualización. Así como a crear el cuestionario correspondiente a ese tutorial.

Los alumnos inscritos a las asignaturas o a los exámenes extraordinarios tendrán acceso a ver el tutorial publicado así como a contestar el cuestionario correspondiente el cual se les podrá tomar en cuenta como examen departamental o extraordinario según sea el caso al que pertenezcan.

Los profesores podrán realizar consultas por asignatura de los alumnos que contestaron el cuestionario y su calificación.

Para ello se propone que el sistema cuente con los siguientes módulos:

1. Administrador de Contraseñas.
2. Creación y Modificación de Tutoriales.
3. Creación y Modificación de Cuestionarios.
4. Consulta de Tutoriales.
5. Aplicación de exámenes.
6. Evaluación de Exámenes.
7. Consulta de Exámenes.

4.2.1. Módulo Administrador de Contraseñas

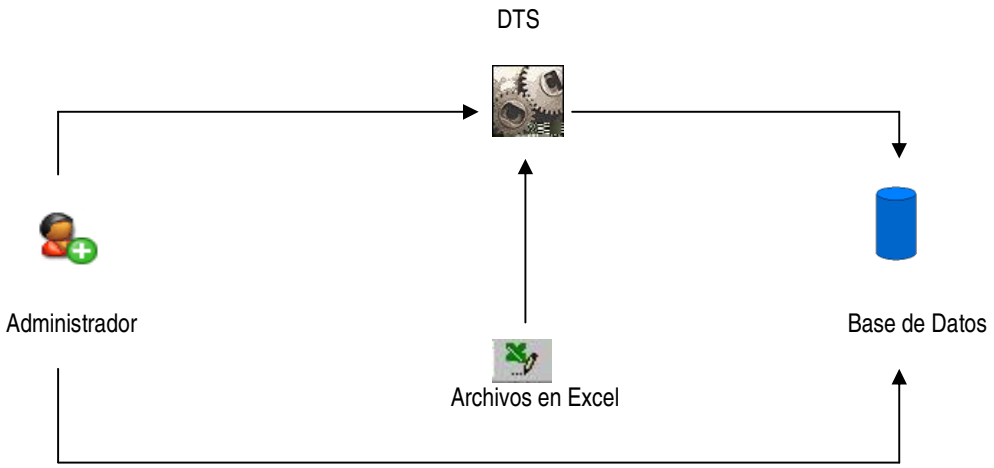
Este módulo se encargará de subir los datos de los alumnos inscritos a las asignaturas o a exámenes extraordinarios al servidor en donde se encuentre la aplicación. La autenticación del alumno, será por su usuario, contraseña y número de cuenta.

Solo podrá acceder a este módulo el administrador del sistema dando privilegios de administrador a quien él considere pertinente.

En este módulo también se podrán dar de alta usuarios que estén asignados para crear, modificar o borrar algún tutorial o cuestionario existente.

Cabe mencionar que el usuario y contraseña que se le proporcione es personal e intransferible. El mal uso que se haga de este será responsabilidad del usuario.

Esquema



El administrador del sistema correrá un DTS (Servicio de Transformación de Datos) el cual tomará los datos de un archivo de Excel previamente colocado en una carpeta definida, con este proceso se poblará una tabla en la base de datos utilizada para obtener el usuario y contraseña para la autenticación de los alumnos por semestre.

También el administrador podrá dar de alta a los profesores que se lo soliciten para que puedan crear tutoriales y cuestionarios dentro del sistema, así como darles el privilegio que el considere pertinente.

Habrá cinco tipos de usuarios, los cuales tienen diferentes privilegios según se describe a continuación:

1. *Alumnos*. Podrán ver los tutoriales de las asignaturas que estén inscritos y resolver el examen de dichas asignaturas solo una vez.
   Los alumnos se dividen en dos grupos:
   - Alumnos ordinarios que son los alumnos inscritos en las asignaturas hasta dos veces
   - Alumnos extraordinarios son aquellos que presentarán el examen extraordinarios y no están inscritos en la asignatura
2. *Coordinador del Área*. Podrá ver todos los tutoriales, cuestionarios y hacer cualquier consulta.
3. *Desarrollador.* Podrá modificar los tutoriales y cuestionarios que haya creado, ver todos los tutoriales y hacer cualquier consulta.
4. *Profesor*. Solo podrán ver los tutoriales, cuestionarios y consultas de las asignaturas que imparta.

   Los Profesores se dividen en dos grupos:
   - El desarrollador que es el que se involucra en la realización del tutorial y cuestionarios, por lo que pasa a ser desarrollador.

- El Profesor que esta en la plantilla del personal de la Universidad pero que no se involucra en la realización de los mismos.

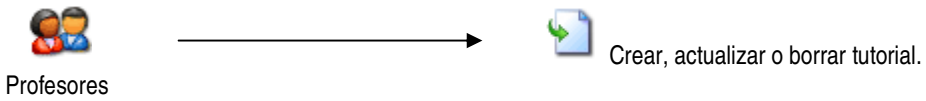
5. *Administrador de contraseñas*. Podrá dar de alta, baja y modificar usuarios, podrá dar de alta asignaturas y grupos relacionados a estas, así como consultar los tutoriales, cuestionarios y realizar cualquier consulta.

4.2.2. Módulo de Creación y Modificación de Tutoriales

Solo podrá ser accesado por profesores que hayan solicitado un usuario y una contraseña. Se podrán crear, modificar o dar de baja el contenido y titulo de los capítulos, subcapitulos, temas y subtemas de un tutorial.

El capitulo, subcapitulo, tema o subtema solo podrá ser modificado por un usuario a la vez, este debe de indicar si autoriza su publicación en la Web para poder mostrarlo, el tutorial debe de tener un máximo de 15 capítulos y debe de contener al menos una bibliografía.

Esquema



Los profesores autorizados podrán crear y modificar el contenido de los tutoriales creados por ellos y solo un usuario a la vez podrá modificar los capítulos de algún tutorial.

4.2.3. Módulo de Creación y Modificación de Cuestionarios

Solo podrán acceder los profesores que hayan creado el tutorial del cual se deriva este cuestionario. Dicho cuestionario solo podrá ser modificado por un profesor a la vez.

Se creará solo un cuestionario por tutorial de opción múltiple con 3 ó 5 opciones de respuesta, con 30 preguntas cada uno, se debe indicar la fecha de publicación del cuestionario, dicha fecha se podrá cambiar según se requiera.

Esquema

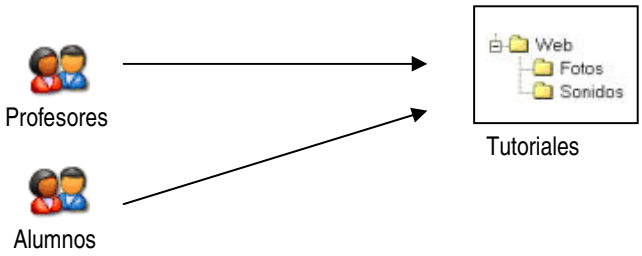


En este módulo solo un usuario a la vez podrá crear o modificar el cuestionario correspondiente al tutorial que ha creado previamente, también podrá modificar la fecha de publicación del cuestionario según lo requiera.

4.2.4. Módulo de Consulta de Tutoriales

Podrán acceder los alumnos inscritos a las asignaturas, los alumnos inscritos a los exámenes extraordinarios y los profesores que cuenten con usuario y contraseña. Solo podrán ver el contenido del tutorial, no podrán modificarlo.

Esquema

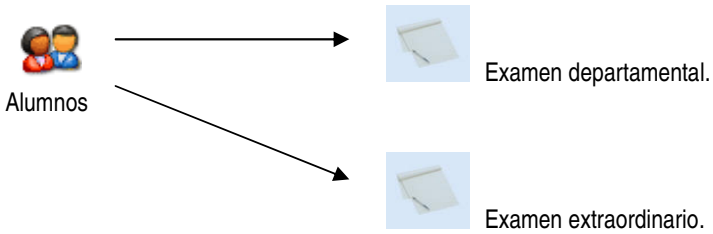


4.2.5. Módulo de Aplicación de Exámenes

En este módulo solo podrán acceder los alumnos, estos solo podrán contestar el examen una sola vez, teniendo un tiempo máximo de 1 hora para exámenes departamentales y 2 horas para exámenes extraordinarios, el número de preguntas será de 10 para departamentales y 30 para extraordinarios, el usuario no podrá regresarse a la pregunta anterior para evitar que cambie su respuesta.

Las preguntas que se presentan en el examen serán aleatorias, por lo que cada examen aplicado será diferente.

Esquema

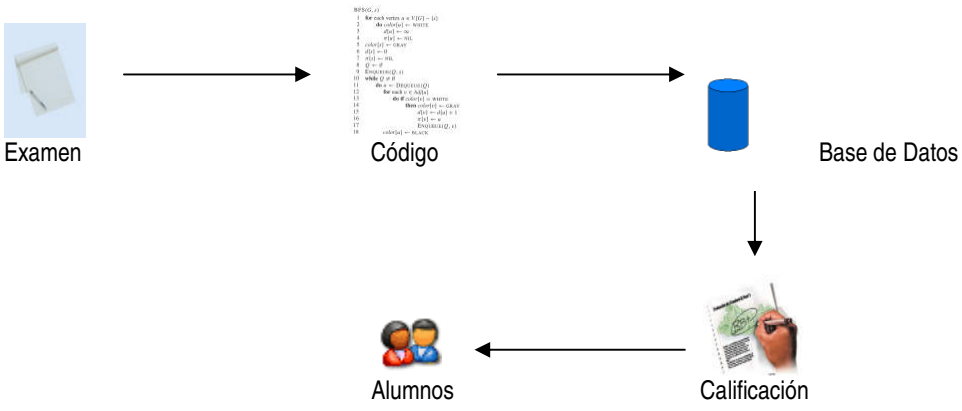


Solo podrán entrar los alumnos que estar inscritos a la asignatura a la cual corresponde ese examen, siempre y cuando este se encuentre en la fecha de publicación, dichos exámenes solo se podrán resolver una vez por cada alumno por lo que se le darán todas las indicaciones antes de comenzar a contestar el examen, así como el tiempo que tiene para resolverlo.

4.2.6. Módulo de Evaluación de Exámenes

En este módulo se ingresaran a la base de datos los números de las preguntas que se realizaron en el examen y se les dará un porcentaje a cada pregunta dependiendo del número de preguntas generadas para obtener su calificación final, ésta se le proporcionara al alumno al terminar su examen.

Esquema

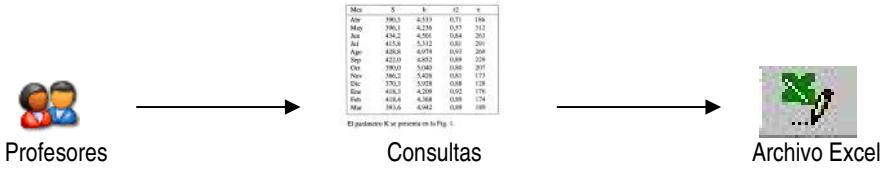


4.2.7. Módulo de Consultas de Exámenes

En este módulo los profesores que cuenten con un usuario y contraseña podrán hacer consultas por alumno, tipo de examen y por asignatura de los usuarios que contestaron el examen, podrán obtener su calificación, la asignatura a la que están inscritos y el examen que se le aplicó a cada alumno.

Esta información podrá ser exportada a un formato en Excel.

Esquema



Únicamente los profesores que tienen privilegios para crear tutoriales podrán acceder a este módulo en donde pueden consultar la información de los alumnos que realizaron el examen ya sea por alumno, tipo de examen o por asignatura, esta información se presentara en una tabla la cual podrá ser exportada a un formato de Excel.

4.3. LENGUAJES DE PROGRAMACIÓN

Las páginas Web son texto ASCII escrito en el lenguaje HTML World Wide Web y los exploradores de los clientes mediante el protocolo HTTP (HyperText Transfer Protocol).

La investigación y desarrollo en el campo de la creación y mantenimiento de páginas Web es uno de los más dinámicos en el ámbito de INTERNET, debido a sus consecuencias comerciales y de utilización de la red.

Inicialmente las páginas Web eran estáticas, en el sentido de que, a efectos de usuario, el único proceso realizado era el de visualización de sus contenidos (escritos en lenguaje HTML) por parte del explorador del cliente.

Las páginas estáticas se siguen utilizando ampliamente debido a que forman la base necesaria para la presentación de datos en muchos tipos de situaciones. También influye decisivamente la sencillez con que se pueden crear, instalar y manejar.

Surge la necesidad de una interacción mayor entre los usuarios y el sistema que soporta las páginas Web, así como el reunir y procesar las peticiones del cliente con el fin de ofrecerle información mejor dirigida, escogida y elaborada.

Por lo anterior, se demanda la creación de alguna fase de procesamiento al intercambio de información entre los usurarios y el servidor de páginas Web, con lo que se introduce el concepto de páginas Web activas o dinámicas.

Otra importante ventaja se basa en el ahorro de comunicaciones (ancho de banda) que se puede experimentar en muchas aplicaciones que ejecuten procesos en el cliente, y eviten de ésta manera realizar continuos traspasos de información con el servidor de Web.

Las páginas activas en el cliente se basan actualmente en tecnologías muy dependientes del explorador y la plataforma del usuario, de esta manera, hoy en día, un explorador de Netscape no interpretaría aplicaciones (scripts) de Visual Basic. Los ActiveX sólo funcionan con Windows, etc.

A continuación haremos un análisis de los posibles lenguajes de programación que podemos utilizar para la creación de Páginas Dinámicas y cual de ellos nos conviene más para el desarrollo de nuestro sistema:

- CGI
- ASP
- PERL
- JAVA
- PHP

4.3.1. CGI (Common Gateway Interface)

Es el sistema más antiguo que existe para la programación de las páginas dinámicas de servidor. El CGI cambió la forma de manipular información en el Web.

Es un método para la transmisión de información hacia un compilador instalado en el servidor. Su función principal es la de añadir una mayor interacción a los documentos Web que por medio del HTML se presentan de forma estática, tiene la ventaja de correr en el servidor cuando el usuario lo solicita por lo que es dependiente del servidor y no de la computadora del usuario.

Los programas que maneja el CGI pueden estar compilados en diferentes lenguajes de programación.

El más popular para el desarrollo de contenidos Web es el lenguaje Perl de distribución gratuita, aunque también podemos mencionar: C, C++ y Java.

4.3.1.1. Funcionamiento

El funcionamiento básico de un programa CGI es parecido al utilizado para las páginas dinámicas del servidor, con algunas particularidades:

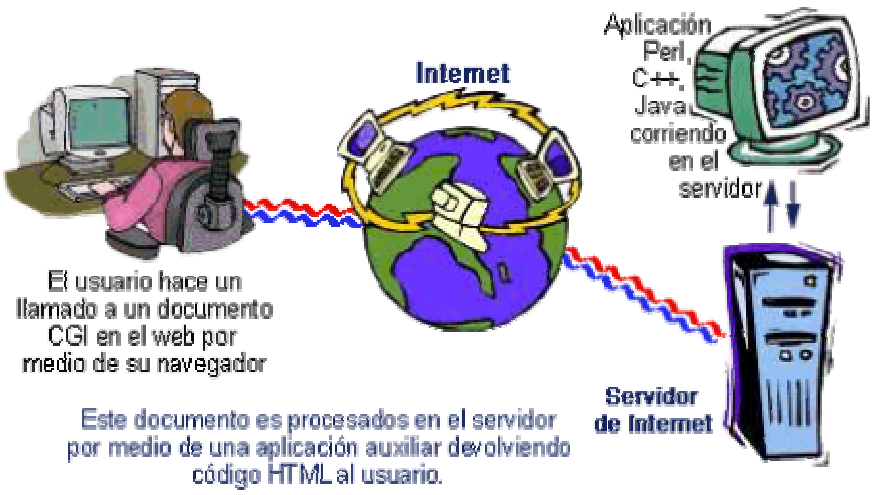
1. Se realiza una petición http, a la que pueden acompañar datos que nos han llegado por un formulario o bien a través de la URL.
2. El servidor ejecuta los programas CGI a los que se accede y trabaja con los recursos necesarios para llevar a cabo las acciones, como por ejemplo bases de datos.

3. El programa CGI va escribiendo en la salida estándar el resultado de la ejecución del CGI, que incluye etiquetas HTML, ya que lo que se escribe es una página Web.

En general, podemos decir que el funcionamiento de esta tecnología es muy sencillo. Los scripts residen en el servidor, donde son llamados, ejecutados y regresa información de vuelta al usuario.

El CGI es utilizado comúnmente para contadores, bases de datos, motores de búsqueda, formularios, generadores de correo electrónico automático, foros de discusión, chats, comercio electrónico, mapas de imágenes, juegos en línea y otros.

Para una mejor aclaración presentamos la siguiente gráfica:



4.3.1.2. Ventajas

Algunas ventajas de los CGI son las siguientes:

- Los scripts se programan en lenguajes potentes y rápidos como C, C++ o Perl, esto hace que las aplicaciones no consuman tantos recursos.
- Como los scripts se ejecutan sobre cualquier servidor y sólo reciben código HTML (ya sea por el método POST o GET) y eventualmente devuelve código también en HTML, esto los hace absolutamente independientes del tipo de servidor que se desee utilizar, por tanto: son portables.

4.3.1.3. Desventajas

Algunas desventajas de la programación en CGI son las siguientes:

- Los resultados se escriben directamente con el CGI, así que el código del programa se mezcla con el del HTML haciendo difícil su comprensión y mantenimiento.
- Cada programa CGI que se pone en marcha lo hace en un espacio de memoria propio. Así, si tres usuarios ponen en marcha un CGI a la vez se multiplicará por tres la cantidad de recursos que ocupe ese CGI. Esto significa una grave ineficiencia.
- Si el CGI es enviado por el método GET, ya que no permite manejar grandes cantidades de información y por ser una dirección, los Caches de los navegadores pueden guardar los resultados, lo que dificulta el poder acceder nuevamente a la aplicación.
- La posibilidad de que el CGI sea engañado por la entrada del usuario para que ejecute comandos imprevistos, pudiendo llegar a causar graves daños en el servidor.
- La posibilidad de revelar innecesariamente información acerca del servidor, que permitirá al atacante conocer mejor la configuración del sistema y estar así mejor equipado para buscar posibles agujeros por los que colarse.
- Uno de los aspectos más delicados de la programación con CGI, es la llamada a otros programas o comandos del sistema desde la aplicación CGI.
- Si se especifica la máxima longitud de un campo, el usuario podría enviar más caracteres de los prefijados, manipulando el formulario o invocando al CGI directamente.
- El usuario puede ver los datos presentes en campos ocultos.
- Si se utilizan cookies para mantener el estado de la sesión en el servidor, el programa CGI podría recibir cookies que nunca creó.

4.3.2. ASP (Active Server Pages)

ASP o lenguaje de páginas activas de Microsoft esta englobado dentro de los lenguajes ISS (Include Server Side) de 2ª generación, es decir, que ASP es una evolución de los CGI's (Common Gateway Interface).

No define un lenguaje de programación con sus sentencias de control, sus estructuras de almacenamiento, sino que define una serie de objetos de servidor, los cuales tienen una serie de métodos que podremos utilizar para cosas como acceso a base de datos, lectura de ficheros, etc.

ASP está basado en dos lenguajes de script, como son JavaScript y VBScript para su desarrollo. Sus páginas son agrupadas en la categoría de lenguajes de script (guión), contienen además de los tags de HTML habituales en las páginas Web, fragmentos de código que el servidor resolverá antes de enviarlo al navegador

Una página ASP es en esencia una página HTML que contiene código que será ejecutado del lado del servidor y procesado por el Web Server antes de que el cliente reciba la información. Es decir que el cliente lo que recibe es código HTML puro. Esa programación se puede combinar a su vez con XML, COM y HTML.

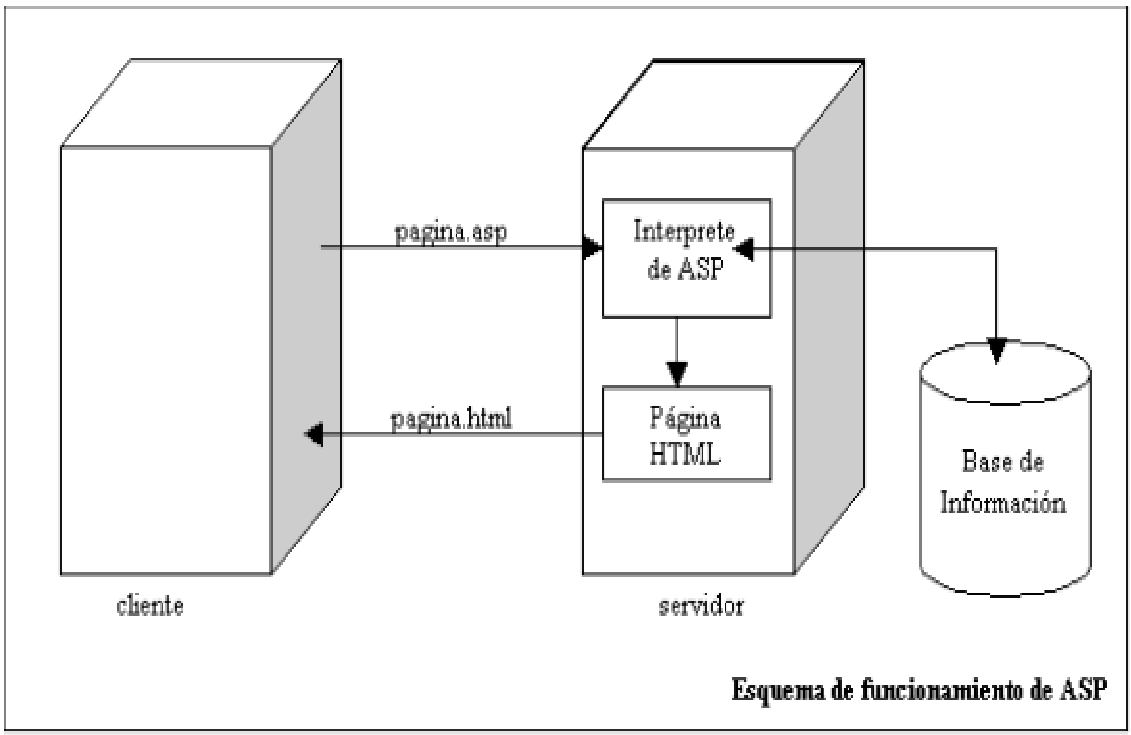
De esta forma, cada vez que el cliente solicita una archivo .asp del servidor, se ejecuta primero el código dentro de dicha página y luego se envía el resultado formateado de dicha página al cliente. Así pues, dentro de ese código tanto se pueden utilizar elementos de XML como se pueden hacer llamadas a componentes COM para acceder a otro tipo de información.

4.3.2.1. Funcionamiento

El esquema de funcionamiento de ASP sería:

- Una máquina cliente realiza una petición de una página ASP.
- Esta petición llega a una maquina servidor la cual interpreta el código de esa página ASP.
- Dicho código puede tener accesos a ficheros o bases de datos (Base de Información).
- El resultado de interpretar la página ASP es una página HTML, la cual se le envía al usuario. Es decir, el usuario no llega a ver el código ASP, sino que ve el resultado de interpretar dicho código: una página HTML.

Para una mejor aclaración presentamos la siguiente gráfica:



Esquema de funcionamiento de ASP

En conclusión podríamos decir que una aplicación en ASP tiene como objetivo diseñar una página Web. Todas las salidas de información que se realicen en una pagina ASP serán de código HTML o texto.

Como hemos dicho antes, la respuesta a la petición de una página ASP es una página HTML. Es por ello que dentro de una página ASP podemos encontrarnos con:

- Código ASP, devolveremos aquel código que sea susceptible de cambiar, o el encargado de acceder a una base de datos.
- Código HTML, partes del código HTML que permanezcan inmutables, esas partes de código se incluirán sin más, como si de una página HTML se tratara.

Ambos códigos se mezclarán dentro de la página ASP sin ningún orden.

4.3.2.2. Ventajas

Algunas ventajas de ASP son las siguientes:

- La facilidad para conectar con una Base de datos y extraer datos de la misma dinámicamente visualizándolos en el navegador.
- Puede conectarse a gestores de Base de datos SQL, Access, Oracle, o cualquier otro motor que disponga de driver ODBC.
- Se puede tener código embebido dentro del código HTML.
- Utiliza scripts (VBscript y JavaScript)
- Compatibilidad con otros lenguajes de programación
- El cliente no tiene acceso nunca al código que se ejecuta.
- Las páginas se ejecutan dentro del propio servidor Web (IIS) en lo que se denomina IN-PROC. Es decir, no salen a tomar recursos de memoria fuera del servidor Web; lo cual permite un mejor aprovechamiento del sistema de hebras con el que cuenta Windows.
- Óptima reutilización de objetos por parte del propio administrador del IIS, obtenemos un entorno bastante más optimizado en lo que concierne a utilización de recursos.
- Las Páginas Activas permiten a su vez el acceso a bases de datos a través de objetos que administran esas llamadas. Por tanto, tampoco se desperdician recursos a ese nivel.
- ASP sólo tiene soporte nativo para los servidores IIS y Personal Web Server, que son los dos servidores Web para sistemas Microsoft, el primero con tecnología NT y el segundo para sistemas Windows 98 y similares.
- El IIS a partir de Windows XP, viene incluido, así que se reducen costos de licencias.

4.3.2.3. Desventajas

Algunas desventajas de ASP son las siguientes:

- Creado para funcionar sobre servidores Windows NT con IIS (Internet Information Server).
- Existencias de demasiados sitios ya establecidos con scripts CGI´s aún funcionales.
- Su portabilidad está limitada a aplicaciones bajo Windows.

4.3.3. Perl (Practical Extraction and Report Language)

Lenguaje inspirado en otras herramientas de UNIX como son: sed, grep, awk, c-shell, para la administración de tareas propias de sistemas UNIX.

Éste lenguaje es utilizado muy a menudo con el sistema UNIX; aunque hoy se hayan efectuado cambios sobre otras muchas plataformas (Windows 9x/NT, OS/2, MS-DOS, etc.), el entorno natural para desarrollar y utilizar script en Perl es UNIX, donde es posible integrar y hacer interaccionar los script con los numerosísimos instrumentos incluidos en el sistema (sed, awk, grep, mail etc.) y obtener lo mejor del lenguaje.

Perl es un lenguaje de programación, estructurado en el lenguaje C, está diseñado para facilitar la manipulación de textos, archivos y procesos. Prácticamente, sirve para todo, todas las tareas de administración de UNIX se pueden simplificar con un programa en Perl.

4.3.3.1. Funcionamiento

La forma del lenguaje facilita la programación. Esto hace también que se utilice para hacer prototipos rápidos de algún algoritmo que queremos ver funcionar antes que tomarnos la molestia de programarlo en un lenguaje más eficiente, como el C++. Y últimamente ha encontrado su aplicación en la escritura de CGI o scripts ejecutados desde páginas de la World Wide Web.

En general, los programas en Perl se ejecutan en el servidor, como todos los programas CGI, a diferencia de otros programas ejecutados por el cliente (generalmente un navegador como el INTERNET Explorer de Microsoft o el Navigator), como aquellos escritos en JavaScript o Java.

Además existen extensiones al Apache (mod Perl) que permiten ejecutar directamente programas en Perl desde el servidor de HTTP.

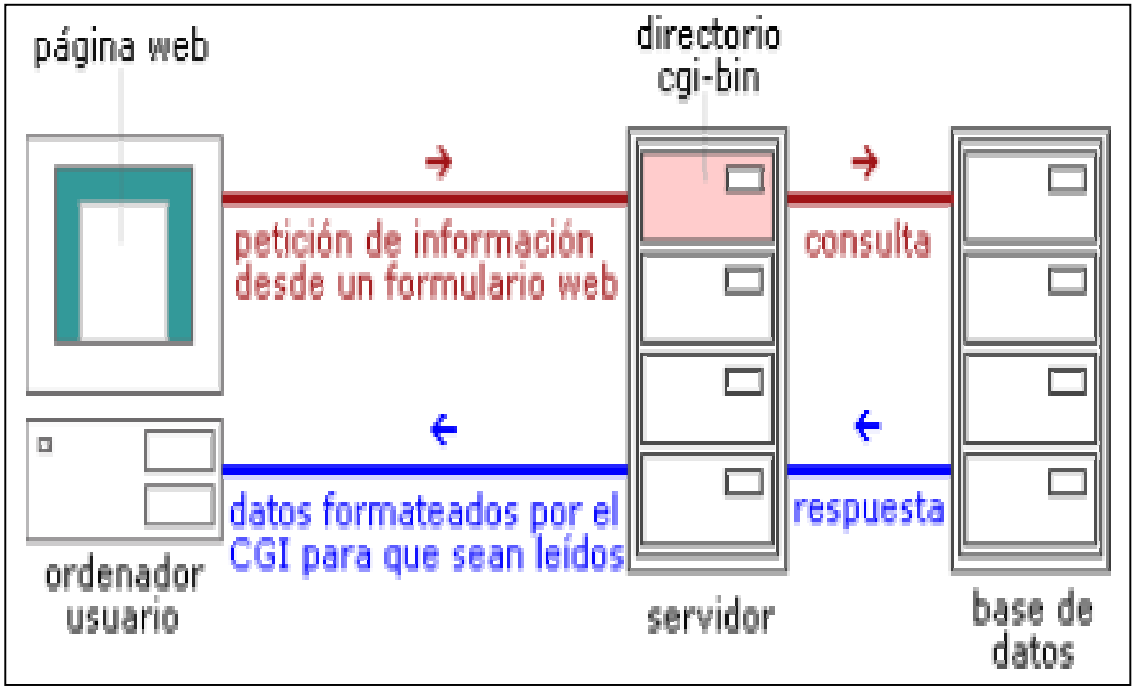
Mediante una serie de módulos adicionales, tales como el DBD o el ODBC, Perl puede servir para acceder a bases de datos, desde BD gratuitas como MySQL hasta el Microsoft SQL Server usando ODBC.

Es un lenguaje interpretado que tiene varias utilidades, pero está principalmente orientado a la búsqueda, extracción y formateado de ficheros de tipo texto. También es muy usado para manejo y gestión de procesos (estado de procesos, conteo y extracción de parámetros característicos, etc.).

Algunas de las principales funciones del lenguaje PERL son:

- Permite construir filtros, para seleccionar información desde un archivo cualquiera, empleando expresiones regulares de búsqueda.
- Permite una comunicación interactiva, por medio de comandos ejecutados en el sistema operacional.
- Al igual que cualquier script de los diferentes interpretadores de comandos (Shells), permite almacenar y ejecutar sin problema un script en particular desde el sistema operativo.
- Puede emplearse en cualquier línea de un programa estructurado en cualquier ambiente, sin problemas de ejecución.
- Permite la manipulación de la estructura de cualquier archivo en el ambiente Unix. Toda la traducción Lógica de Oracle Libraries, se estructuró en scripts elaborados en PERL, sus resultados fueron excelentes y la traducción fue exitosa.

Para una mejor aclaración presentamos la siguiente gráfica:



4.3.3.2. Ventajas

Algunas ventajas de Perl son las siguientes:

- Construcción de pequeños programas que pueden ser usados como filtros para obtener información de ficheros, realizar búsquedas, etc.
- Se puede utilizar en varios entornos, como puede ser Windows 95, OS/2, entre otros, sin realizar cambios de código, siendo únicamente necesario la introducción del interprete PERL correspondiente a cada sistema operativo.
- También es uno de los lenguajes mas utilizados en la programación de CGI scripts, que son guiones o scripts que utilizan el interfase CGI, para intercambio de información entre aplicaciones externas y servicios de información. Como ejemplo de ello tenemos los programas de búsqueda usados por el browser Netscape.
- El mantenimiento y depuración de un programa en PERL es mucho más sencillo que la de cualquier programa en C.

4.3.3.3. Desventajas

Algunas desventajas de Perl son las siguientes:

- Es lento para algunas aplicaciones, como programación a bajo nivel, escribiendo un ``driver'' para una aplicación o corriendo modelos numéricos de cálculo intensivo. Si bien se pueden insertar subrutinas FORTRAN o C en Perl, teniendo lo mejor de los dos mundos, pero con algo más de complejidad.
- La libertad que se le otorga al programador puede significar que el resultado sea un programa ilegible. Si no se escribe con cuidado puede llegar a ser difícil de leer. De hecho hay un concurso de Perl ofuscado.
- No se pueden compilar programas Perl. Aunque actualmente se está desarrollando un compilador que realice esta tarea y hay uno comercial disponible para la plataforma Windows. Este tema no es tan crítico como suena, los programas Perl no correrán mucho más rápidos cuando se compilen, la única ventaja está en la desaparición de la fase inicial de compilación al correr la aplicación.
- Utiliza muchos recursos de la máquina. Esto significa que no es tan ligero como un programa en C, pero en la práctica es ligero comparado con la potencia de computación de las computadoras actuales.

4.3.4. Java

Java se creó como parte de un proyecto de investigación para el desarrollo de software avanzado para una amplia variedad de dispositivos de red y sistemas embebidos. La meta era diseñar una plataforma operativa sencilla, fiable, portable, distribuida y de tiempo real.

Para la creación de Java, se extrajeron decisiones de diseño y arquitectura de una amplia variedad de lenguajes como Eiffel, SmallTalk, Objetive C y Cedar/Mesa. Creado por SUN Microsystems

El resultado es un lenguaje que se ha mostrado ideal para desarrollar aplicaciones de usuario final seguras, distribuidas y basadas en red en un amplio rango de entornos desde los dispositivos de red embebidos e INTERNET.

4.3.4.1. Funcionamiento

El fichero de código fuente puede ser escrito mediante cualquier editor Ascii convencional o, para mayor comodidad, con el editor suministrado con el paquete del lenguaje Java.

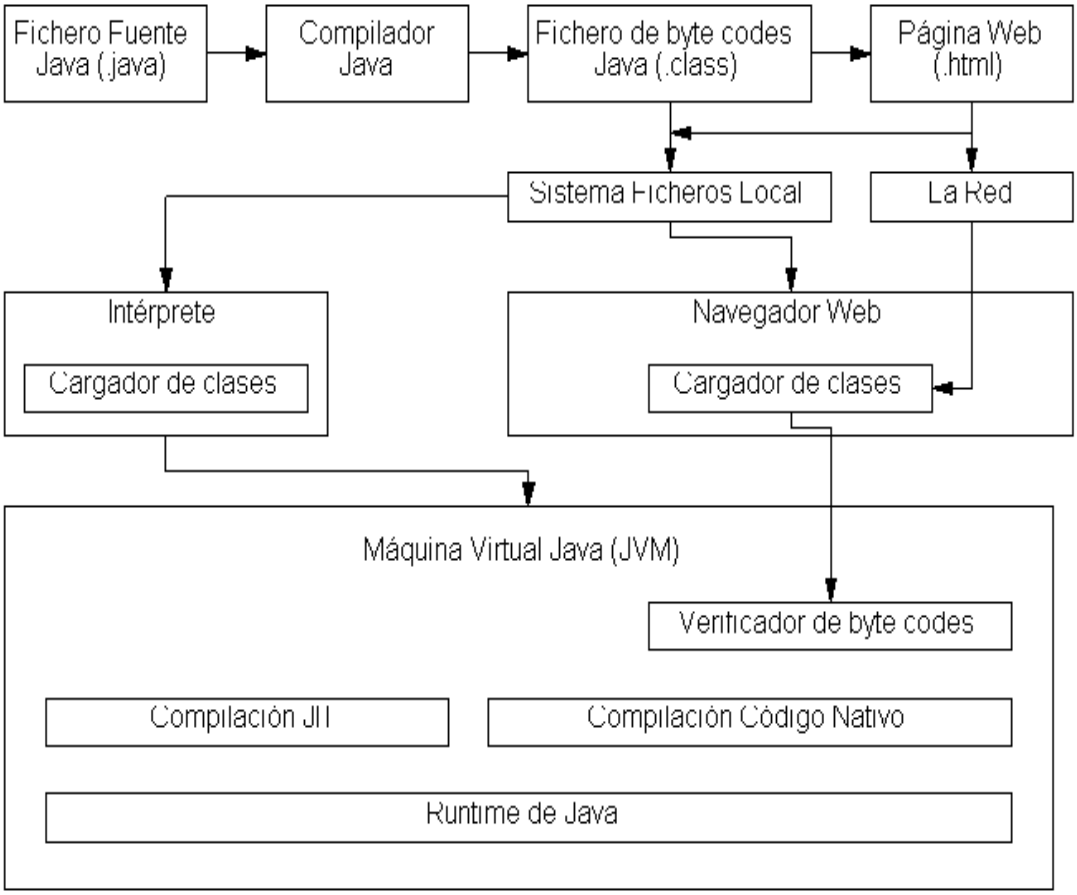
Una vez creado el fichero .java con el código del programa, se compila, generándose un fichero intermedio con los bytecodes, de extensión .class.

Una vez generado el fichero .class, éste ya puede ser interpretado en cualquier máquina virtual de Java. Tratándose de applets, este fichero se descargará típicamente desde la Red. Para indicárselo al navegador se utilizan unas etiquetas especiales del lenguaje HTML.

Esta página Web en la que se encuentra embebida la applet la habremos bajado desde la Red o bien la hemos cargado desde el disco duro. En cualquier caso, para ver la applet funcionando hace falta un navegador capaz de ejecutar Java, como los muy conocidos Netscape y Microsoft Internet Explorer.

Java posee una curva de aprendizaje muy rápida. Resulta relativamente sencillo escribir applets interesantes desde el principio. Todos aquellos familiarizados con C++ encontrarán que Java es más sencillo, ya que se han eliminado ciertas características, como los punteros.

Para una mejor aclaración presentamos la siguiente gráfica:



4.3.4.2. Ventajas

Algunas ventajas de Java son las siguientes:

- Lenguaje de propósito general.
- Lenguaje Orientado a Objetos.
- Lenguaje multiplataforma: Los programas Java se ejecutan sin variación (sin recompilar) en cualquier plataforma soportada (Windows, UNIX, Mac, entre otros).
- Lenguaje interpretado: El intérprete a código máquina (dependiente de la plataforma) se llama Java Virtual Machine (JVM). El compilador produce un código intermedio independiente del sistema denominado bytecode.
- Lenguaje gratuito
- API distribuida con el J2SE muy amplia. Código fuente de la API disponible.

4.3.4.3. Desventajas

Algunas desventajas de Java son las siguientes:

- Se compila e interpreta (doble tiempo).
- No está estandarizada
- Más complicado de usar porque es orientado a objetos y la manera de escribir los programas es más rígida.
- Se debe rescribir debido a que pueden diferir de las aplicaciones nativas
- La velocidad y el consumo de recursos es mucho mayor si los usurarios no poseen una conexión adsl (y a veces ni aun así).
- Es antiestético, por lo tanto poco didáctico
- Hay diferentes tipos de soporte técnico para la misma herramienta, por lo que el análisis de la mejor opción se dificulta.
- Para manejo a bajo nivel deben usarse métodos nativos, lo que limita la portabilidad.
- El diseño de interfaces gráficas con awt y swing no es simple.
- Existen herramientas como el JBuilder que permiten generar interfaces gráficas de manera sencilla, pero tienen un costo adicional y no es totalmente compatible con otros lenguajes.

4.3.5. PHP (Hypertext Preprocessor)

PHP es un lenguaje de programación de alto nivel, el cual se ejecuta en los servidores Web, que permite crear contenido dinámico en páginas HTML ya que forma parte de éstas páginas y es open source.

Open Source Software (Software de fuentes abiertas), se refiere al modelo de desarrollo y de distribución del software desarrollado cooperativamente.

En vez de que el código del sistema o de cada uno de los programas sea un secreto celosamente guardado por la empresa que lo produce, éste es puesto a disposición del público, para que puedan modificar, mejorar o corregir.

Es multiplataforma, funciona tanto para Unix (con Apache) como para Windows (con Microsoft INTERNET Information Server) de forma que el código que se haya creado para una de ellas no tiene porqué modificarse al pasar a la otra. La sintaxis que utiliza, la toma de otros lenguajes muy extendidos como C y Perl.

4.3.5.1. Funcionamiento

En ningún caso se envía código PHP al navegador, por lo que todas las operaciones realizadas son transparentes para el usuario, al que le parecerá que está visitando una página HTML que cualquier navegador puede interpretar.

El funcionamiento es bastante simple:

- Escribir páginas HTML pero con el código PHP dentro.
- Guardar la página en el servidor Web.
- Un navegador solicita una página al servidor.
- El servidor interpreta el código PHP.

- El servidor envía el resultado del conjunto de código HTML y el resultado del código PHP que también es HTML.

Para una mejor aclaración presentamos la siguiente gráfica:



4.3.5.2. Ventajas

Entre las ventajas podemos citar:

- Está disponible para los principales sistemas operativos.
- Soporta en la orientación a objeto. Clases y herencia.
- El análisis léxico para recoger las variables que se pasan en la dirección lo hace PHP de forma automática. Librándose el usuario de tener que separar las variables y sus valores.
- Se puede incrustar código PHP con etiquetas HTML.
- Permite trabajar inicialmente o cambiar a otra Base de Datos (MySQL, Access, SQL Server, Oracle, etc.).
- Además PHP tiene un conjunto de funciones de seguridad que previenen la inserción de órdenes dentro de una solicitud de datos.
- Se puede hacer todo lo que se pueda transmitir por vía HTTP.
- Tiene soporte para diferentes protocolos.

4.3.5.3. Desventajas

Así como se han presentado las ventajas de PHP, también hay ciertas desventajas las cuales se desglosan a continuación:

- Todo el trabajo lo realiza el servidor y no delega al cliente. Por tanto puede ser más ineficiente a medida que las solicitudes aumenten de número.
- La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.
- La orientación a objetos es aún muy deficiente para aplicaciones grandes.
- Si se trabaja con MySQL para aplicaciones grandes, su funcionamiento es deficiente.

PHP consume mas recursos (y por tanto es ligeramente mas lento) llamando y ejecutando una función que ejecutando código que encuentra en línea, embebido en el script. Por eso no tiene sentido usar funciones salvo que efectivamente vayas a utilizar esa porción de código varias veces.

4.3.6. Elección del lenguaje de programación

Con base en las ventajas y desventajas que se vieron previamente y para los fines que se persigue en la creación del sistema, se decide que la programación sea por medio de ASP por:

- Fácil de aprender y programar.
- Total adaptabilidad con paquetería de Microsoft y herramientas como son los editores.

- Se pueden hacer grandes cosas con pocas líneas de código. Lo cual repercute en menor espacio de almacenamiento y disminución en tiempos de procesamiento.
- Viene acompañado por una excelente biblioteca de funciones que permite realizar cualquier labor (acceso a base de datos, encriptación, manipulación de gráficas a detalle, etc.)
- Al poderse encapsular dentro de código HTML se puede introducir el trabajo del diseñador gráfico e incrustar el código ASP posteriormente.
- Está siendo utilizado con éxito en varios millones de sitios Web.
- Hay multitud de aplicaciones ASP para resolver problemas concretos (weblogs, tiendas virtuales, periódicos, etc.) listas para usar.
- Interactúa perfectamente con la Base de Datos de SQL Server debido a que son de la familia Microsoft.
- Existe la posibilidad de contar en el momento preciso con la ayuda de personal especializado por vía telefónica, e-mail y foros de soporte.

4.4. BASES DE DATOS

A continuación presentamos un análisis de las posibles Base de Datos que podemos utilizar en conjunto con el lenguaje seleccionado y cual de ellas nos conviene más para el desarrollo de nuestro sistema:

- ACCESS
- ORACLE
- MySQL
- SQL Server

4.4.1. ACCESS

Una Base de Datos es un conjunto de información relacionada con un asunto, tema o actividad específicos. Así, se pueden utilizar Bases de Datos para cosas tan sencillas como mantener un registro de nuestra colección de discos de música, hasta llevar toda la gestión de una gran empresa u organización.

Microsoft Access es un potente sistema de administración de Bases de Datos relacionales. Las Bases de Datos de Access son documentos combinados donde se divide la información por parcelas de objetos especializados.

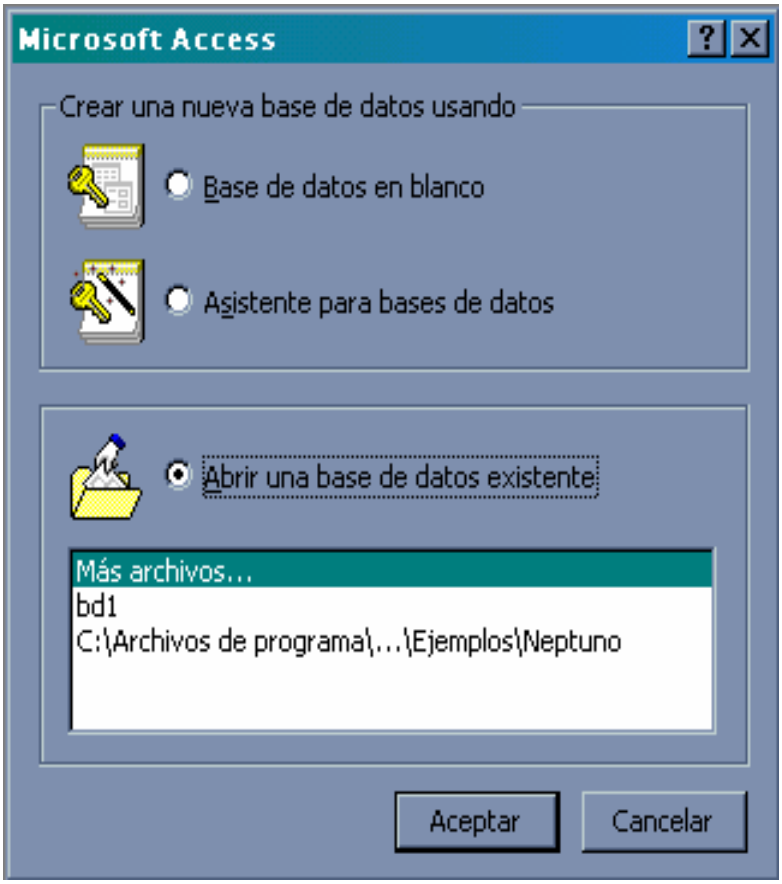
Normalmente, se crea una tabla para cada tipo de datos de los que componen la Base de Datos. Aunque las tablas se crean como elementos independientes, pueden crearse relaciones entre distintas tablas para recuperar datos de todas ellas mediante una consulta, formulario o informe.

Las consultas se utilizan para localizar y recuperar los datos específicos que cumple unas determinadas condiciones especificadas por el usuario. Las consultas permiten, además, actualizar varios registros al mismo tiempo, así como realizar operaciones de muy diversas índole con los datos almacenados en las tablas.

Los formularios permiten visualizar, introducir y modificar los datos de las tablas de una forma muy sencilla y amena. Al abrir un formulario, Access recupera en él los datos de una o varias tablas y les muestra en un diseño de ficha creado, bien de forma automática por el Asistente para formularios, o manualmente desde el principio por el propio usuario.

Los informes se utilizan primordialmente para presentar, resumir e imprimir los datos en la forma que resulte más apropiada para cada proyecto. Se pueden crear informes que incorporen cálculos basados en los datos de las tablas para mostrar resultados totales o promedios, o bien para generar e imprimir catálogos, listas de nombres y direcciones o etiquetas postales.

A continuación presentamos una imagen de la Base de Datos:



Access tiene las siguientes características:

- La interfaz es más amigable.
- Facilidad de uso, implementación y diseño de base de datos.
- Ideal para aplicaciones monousuario, o multiusuario que no sobrepase 5 conexiones en un servidor de archivos, en mas me ha dado muchos problemas.
- Los procedimientos almacenados son más sencillos, pero menos eficientes.
- Aunque puedes ligar con otras bases de datos con MSSQL, MySQL y Oracle, es extremadamente lento, así que no tiene ninguna ventaja sobre MySQL.
- No soporta tablas de más de 4GB, MySQL hasta de 1 Terabyte.

4.4.2. ORACLE

Oracle es básicamente una herramienta cliente/servidor para la gestión de Bases de Datos. Es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hacen que sólo se vea en empresas muy grandes y multinacionales, por norma general.

En el desarrollo de páginas Web pasa lo mismo: como es un sistema muy caro no está tan extendido como otras bases de datos, por ejemplo, Access, MySQL, SQL Server, etc.

En Oracle, una base de datos se refiere a todo el entorno RDBMS de Oracle e incluye los siguientes componentes:

- Procesos de base de datos de Oracle y búferes (instancias).
- Tabla SYSTEM que contiene un catálogo de sistema centralizado, creado a partir de uno o varios archivos de datos.
- Otras tablas, según haya definido el administrador de bases de datos (DBA) (opcional), cada una de ellas creada a partir de uno o varios archivos de datos.
- Dos o más registros de recuperación en conexión.
- Registros de recuperación archivados (opcional).
- Archivos varios (archivo de control, Init.ora, Config.ora, etc.)

Oracle para su utilización primero sería necesario la instalación de la herramienta servidor (Oracle 8i) y posteriormente podríamos atacar a la base de datos desde otros equipos con herramientas de desarrollo como Oracle Designer y Oracle Developer, que son las herramientas básicas de programación sobre Oracle.

Para desarrollar en Oracle utilizamos PL/SQL un lenguaje de 5ª generación, bastante potente para tratar y gestionar la base de datos, también por norma general se suele utilizar SQL al crear un formulario.

4.4.3. MySQL

MySQL es un Sistema de Gestión de Base de Datos.

Una Base de Datos es una colección estructurada de datos. Puede ser, desde una simple lista de artículos a las inmensas cantidades de información en una red corporativa.

Es un gestor de bases de datos SQL (Structured Query Language). Es una implementación Cliente-Servidor que consta de un servidor y diferentes clientes (programas/librerías).Podemos agregar, acceder, y procesar datos grabados en una base de datos.

Actualmente el gestor de base de datos juega un rol central en la informática, como única utilidad, o como parte de otra aplicación.

Es un Sistema de Gestión de Base de Datos Relacional. El modelo relacional se caracteriza a muy grandes rasgos por disponer que toda la información debe estar contenida en tablas, y las relaciones entre datos deben ser representadas explícitamente en esos mismos datos. Esto añade velocidad y flexibilidad.

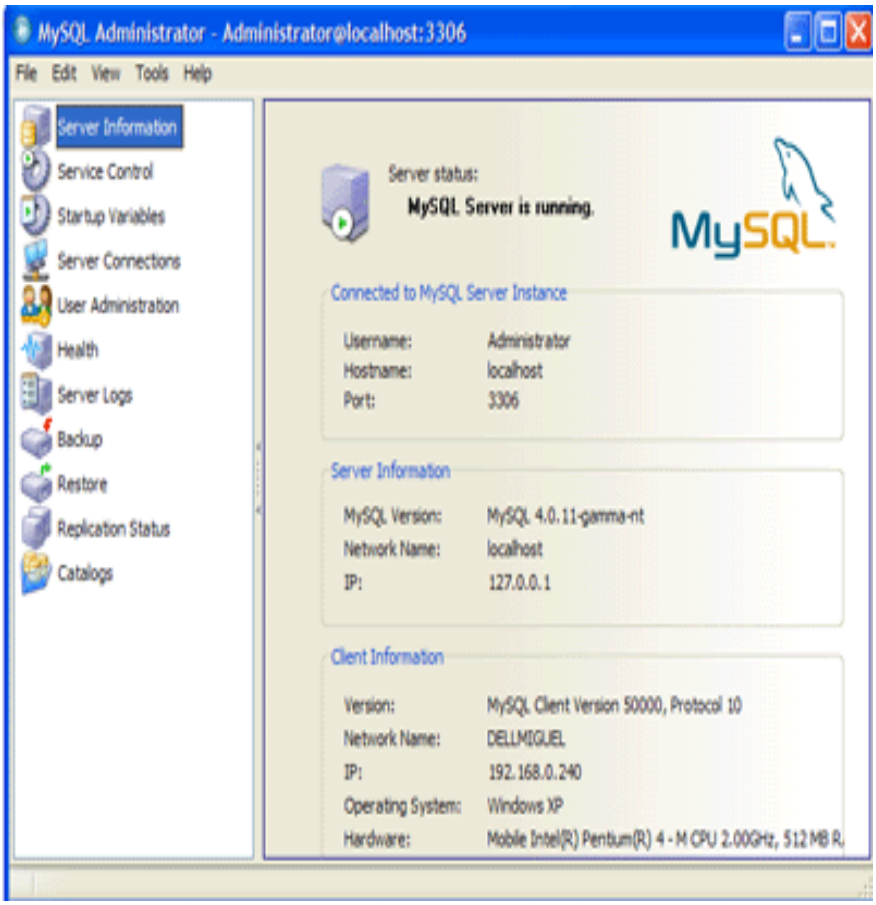
MySQL es un software de código abierto esto quiere decir que es accesible para cualquiera, para usarlo o modificarlo. Podemos descargar MySQL desde Internet y usarlo sin pagar nada, de esta manera cualquiera puede inclinarse a estudiar el código fuente y cambiarlo para adecuarlo a sus necesidades.

También, esta Base de Datos usa el GPL (GNU Licencia Publica General) para definir que podemos y no podemos hacer con el software en diferentes situaciones. Entre otras cuestiones esta licencia aclara que no cuesta dinero a menos que lo incluyamos en un software comercial y tenemos el código fuente.

MySQL es muy rápido, confiable, robusto y fácil de usar tanto para volúmenes de datos grandes como pequeños (siempre, claro está, comparada con las de su categoría). Además tiene un conjunto muy práctico de características desarrolladas en cooperación muy cercana con los usuarios.

Sin embargo bajo constante desarrollo, MySQL hoy en día ofrece un rico y muy útil conjunto de funciones. La conectividad, velocidad y seguridad hace de MySQL altamente conveniente para acceder a bases de datos en Internet.

A continuación presentamos una imagen de la Base de Datos:



MySQL tiene las siguientes características:

- Por ser cliente servidor puede crear respaldos sin tener que cerrar todos los objetos bloqueados por usuarios, por lo que si va a ser multiusuario es la mejor opción.
- Prácticamente gratuito.
- Es multiplataforma.
- Los procedimientos almacenados fueron implementos hasta la versión 5, la cual esta muy estable.

4.4.4. SQL Server

Una instalación (instancia) de SQL Server puede admitir varias bases de datos. Las aplicaciones creadas con SQL Server pueden utilizar bases de datos para dividir de forma lógica la funcionalidad de la empresa. En un único equipo, puede haber varias instancias de SQL Server. Cada instancia de SQL Server puede tener varias bases de datos.

Cada base de datos de SQL Server puede admitir grupos de archivos, que ofrecen la posibilidad de distribuir de forma física la ubicación de los datos. Un grupo de archivos de SQL Server clasifica los archivos del sistema operativo que contienen datos desde una única base de datos de SQL Server para simplificar tareas de administración de las bases de datos, como las copias de seguridad.

Un grupo de archivos es propiedad de una base de datos de SQL Server y no puede contener archivos de sistema operativo de más de una base de datos, aunque una única base de datos puede contener más de un grupo de archivos. Una vez creada una base de datos, pueden agregarse a ella grupos de archivos.

Microsoft SQL Server 2000 ofrece un valor a las empresas basado en un bajo costo total de propiedad, facilidad de uso (asistentes, herramientas de optimización y funciones de optimización automáticas) y capacidades integradas de almacenamiento de datos.

SQL Server 2000 ha recibido los mayores elogios y obtenido los mejores premios de manos de los críticos del sector; asimismo, ha demostrado su capacidad para satisfacer las necesidades de las empresas y sitios de comercio electrónico más importantes del mundo.

En Microsoft SQL Server 2000, las bases de datos conservan su propio catálogo de sistema, que contiene información acerca de:

- Objetos de la base de datos (tablas, índices, procedimientos almacenados, vistas, desencadenadores, etc.).
- Restricciones.
- Usuarios y permisos.
- Tipos de datos definidos por el usuario.
- Definiciones de réplicas.
- Archivos utilizados por la base de datos.

SQL Server también guarda un catálogo de sistema centralizado en la base de datos master, que, además de los catálogos de sistema, almacena parte de la información de las bases de datos individuales, como:

- Nombres de bases de datos y ubicación del archivo principal de cada base de datos.
- Cuentas de inicio de sesión de SQL Server.
- Mensajes del sistema.
- Valores de configuración de la base de datos.
- Servidores remotos o vinculados.
- Información de la actividad actual.
- Procedimientos almacenados de sistema.

SQL Server tiene las siguientes características:

- Escalabilidad: Se adapta a las necesidades de la empresa, soportando desde unos pocos usuarios a varios miles. Empresas centralizadas u oficinas distribuidas, replicando cientos de sites.
- Potencia: Microsoft SQL Server es la mejor base de datos para Windows NT Server. Posee los mejores registros de los benchmarks independientes (TCP) tanto en transacciones totales como en costo por transacción.
- Gestión: Con un completo interfaz gráfico que reduce la complejidad innecesaria de las tareas de administración y gestión de la base de datos.
- Orientada al desarrollo: ASP, Visual Basic, Visual C++, Visual J++, Visual Interdev, Microfocus Cobol y muchas otras herramientas son compatibles con Microsoft SQL Server.
- La mejor base de datos para INTERNET, INTRANET y EXTRANET.
- Diseñada desde su inicio para trabajar en entornos Internet e Intranet, Microsoft SQL Server es capaz de integrar los nuevos desarrollos para estos entornos específicos con los desarrollos heredados de aplicaciones "tradicionales". Cada aplicación que se desarrolle para ser empleada en entornos de red local puede ser utilizada de forma transparente -en parte o en su totalidad- desde entornos INTERNET, INTRANET O EXTRANET.
- Plataforma de desarrollo fácil y abierta: integrada con las mejores tecnologías de Internet como ActiveX, ADC y Microsoft Transaction Server y con las mejores herramientas de gestión y desarrollo para Internet como FrontPage97, Microsoft Office97 y Visual Interdev.
- Diseñada para INTERNET: Es el único gestor de base de datos que contiene de forma integrada la posibilidad de generar contenido HTML de forma automática.
- La Base de Soluciones Integradas: La Integración total con BaclOffice permite resolver toda las necesidades de infraestructura de la empresa con un sólo paquete.
- Potente y Escalable: Microsoft SQL Server es la única base de datos cuyo rendimiento sobre Internet está publicado, ofreciendo registros espectaculares.
- Mínimo costo de Propiedad: La sencillez de la instalación, y la potencia de sus herramientas de gestión y el menor costo de toda la industria para entornos Internet, hacen de Microsoft SQL Server la mejor opción con el menor costo.

4.4.5. Elección de la Base de Datos

Se eligió la Base de Datos SQL Server por:

- Estar diseñada para trabajar en entornos de Internet.
- Ser un software muy confiable ya que proporciona numerosas comprobaciones en compilación y tiempo de ejecución.

- Contar con una herramienta que nos ayuda a homogeneizar la información no importando su fuente llamada DTS.
- Potencia de sus herramientas de gestión.
- Ser capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea.
- Compatibilidad con ASP.
- Existir documentación disponible en Web así como soporte telefónico para resolver cualquier duda o problema.

También se utilizará un DTS que es una aplicación que provee SQL Server y que será muy utilizada semestre con semestre por el administrador del sistema debido a que actualizará la Base de Datos, dando de alta a los alumnos, profesores y personal que se requiera.

Adicional, es compatible con editores que es vital para implementar en el sistema diseño en la parte de creación de tutoriales y se presente amigablemente.

4.4.6. DTS (Data Transformation Services)

DTS tiene como objetivo principal importar y exportar datos al Servidor SQL, pero también soporta el cien por cien de las fuentes de datos originales y finales. Como resultado, se puede usar para transferir datos entre otras fuentes externas de datos sin comprometer el Servidor SQL.

DTS tiene una arquitectura OLE DB por lo que puede copiar y transformar múltiples orígenes de datos, entre ellos:

- SQL Server
- Oracle
- Access, Excel, Outlook, archivos de texto
- Exchange Server
- Y orígenes de datos proporcionados por terceros

DTS es un conjunto de herramientas que permiten importar, exportar y transformar datos heterogéneos entre uno o más orígenes de datos, como Microsoft SQL Server, Microsoft Excel o Microsoft Access.

La conectividad la proporciona OLE DB, un estándar abierto para el acceso a datos. Los orígenes de datos de ODBC (Open Database Connectivity) se admiten gracias al proveedor Microsoft OLE DB para ODBC.

Se puede crear una solución DTS como uno o varios paquetes, cada uno de los cuales puede contener un conjunto organizado de tareas que determinan el trabajo que se debe llevar a cabo, las transformaciones que se deben realizar en los datos y los objetos, las restricciones del flujo de trabajo que definen la ejecución de las tareas y las conexiones a los orígenes y destinos de datos.

Los paquetes DTS también proporcionan servicios como el registro de los detalles de la ejecución de los paquetes, el control de las transacciones y la administración de las variables globales.

Las siguientes son las herramientas disponibles para la creación y ejecución de los paquetes:

- El Asistente para la importación o exportación, que permite crear paquetes DTS relativamente simples y admite la migración de datos y las transformaciones simples.
- El Diseñador DTS, que implementa de forma gráfica el modelo de objetos DTS, permitiendo al usuario crear paquetes DTS con una variada funcionalidad.
- DTSRun, utilidad de solicitud de comandos empleada para ejecutar paquetes DTS existentes.
- DTSRunUI, una interfaz gráfica a DTSRun, que también permite el paso de variables globales y la generación de líneas de comando.
- SQLAgent, que no constituye una aplicación DTS, pero que éste emplea para programar la ejecución de los paquetes.

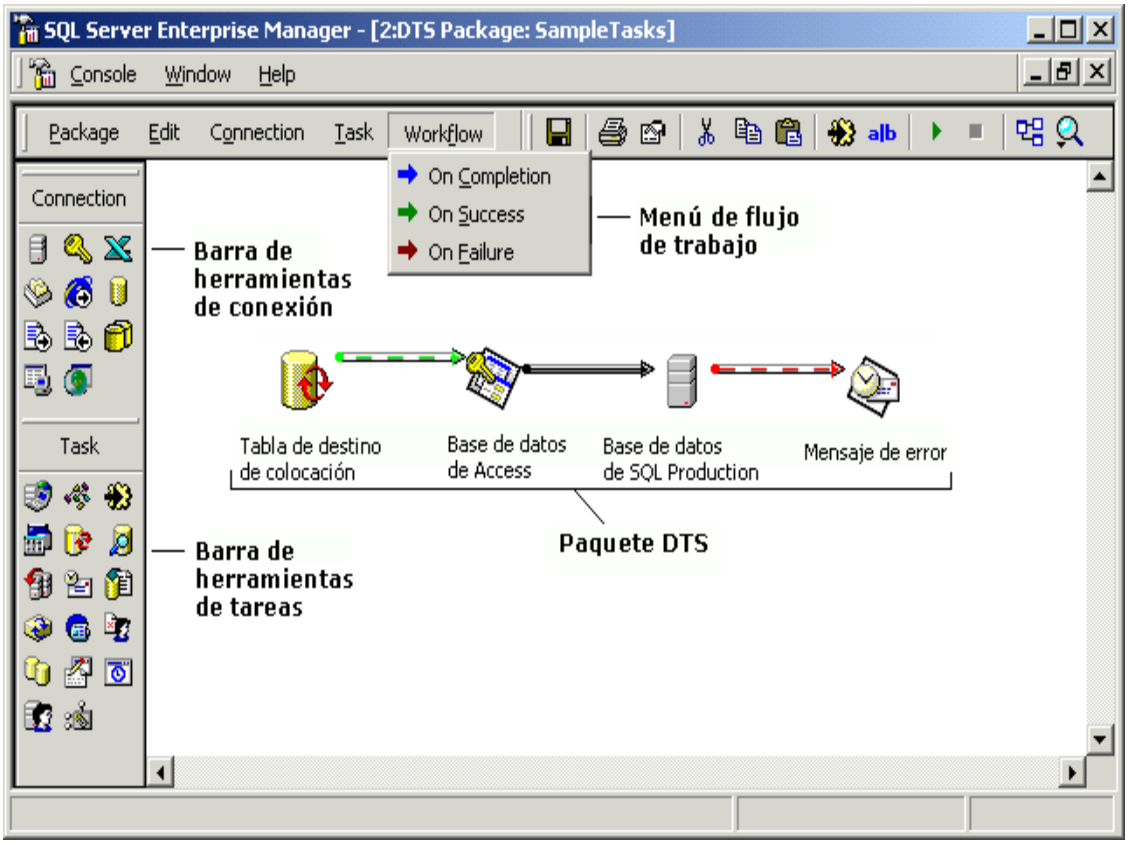
La utilización del modelo de objetos DTS permite crear y ejecutar paquetes por medio de la programación, crear tareas y desarrollar transformaciones personalizadas.

El Diseñador DTS implementa de forma gráfica el modelo de objetos DTS, posibilitando la creación gráfica de paquetes DTS. Puede utilizar el Diseñador DTS para:

- Crear un paquete simple que contenga uno o más pasos.
- Crear un paquete que incluya flujos de trabajo complejos que comprendan varios pasos utilizando la lógica condicional, el código controlado por eventos o las conexiones múltiples a los orígenes de datos.
- Editar paquetes existentes.

La interfaz del Diseñador DTS consta de un área de trabajo para la creación de paquetes, barras de herramientas que incluyen elementos del paquete y que se pueden arrastrar a la hoja de diseño y, por último, menús que contienen los flujos de trabajo y los comandos de administración de los paquetes.

A continuación se muestra la herramienta y sus características:



4.5. EDITOR

Para poder presentar un sistema lo más amigable posible y fácil de utilizar, se utilizará un editor llamado AreaHTML, debido a que nos proporciona la ventaja de poder asociarlo con el uso día a día de Windows.

Los editores más utilizados en programación son AreaHTML y Wisywig, por lo amigable que son para el usuario final y porque tienen una versión mejorada (3.1) que ha sido corregida de errores y es más confiable que las anteriores, además su programación no es tan complicada para modificarlos y adaptarlos para los fines que se requiera.

4.5.1. AreaHTML

Herramienta que permite crear elementos de formulario donde el usuario puede escribir texto con estilos.

Es una herramienta gratuita que sirve para crear elementos de formulario, parecidos a los <textarea>, pero con la particularidad de que permiten introducir

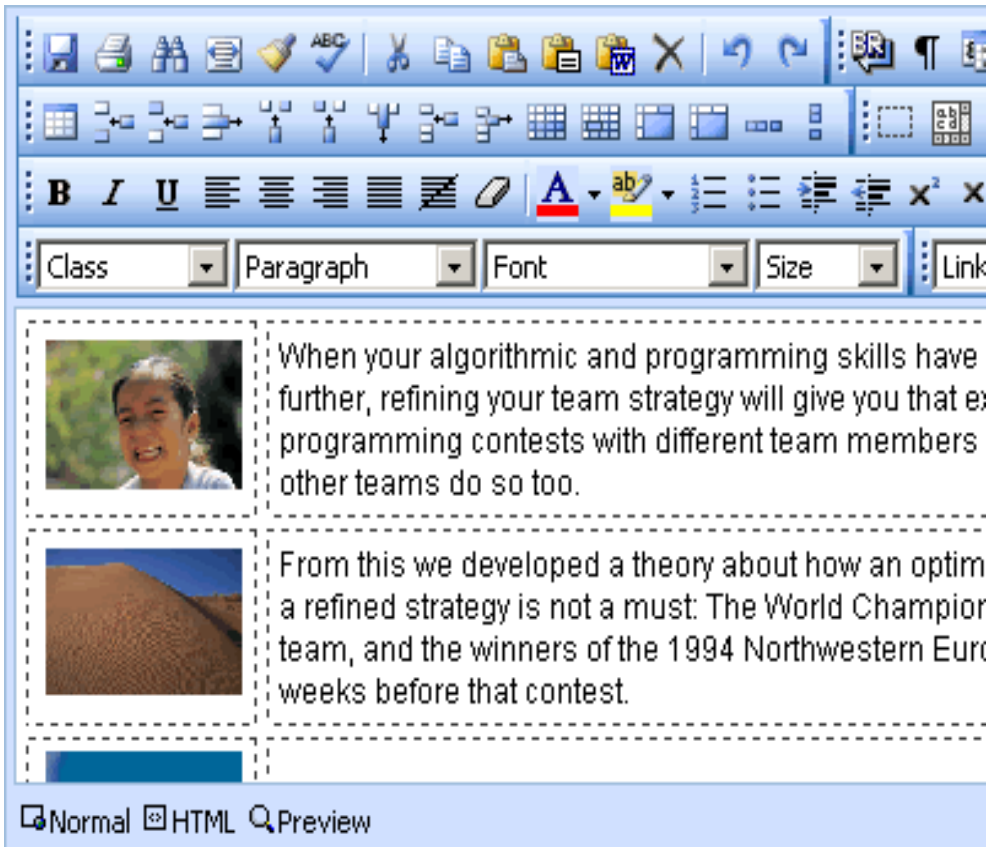
texto con estilos, como negritas, subrayados, distintos tipos de fuentes e incluso, tablas o imágenes.

En definitiva, provee de las funciones típicas de los editores HTML WYSIWYG, pero dentro de un campo de formulario de una página Web.

AreaHTML se incluye en la página con unas pocas líneas Javascript fáciles de escribir. Con ello obtenemos un editor que permite funcionalidades como:

- Formatear texto con negritas, cursivas y subrayados
- Cambiar la tipografía y el color
- Alinear los distintos párrafos
- Incluir listas, líneas horizontales, links, imágenes, etc.

A continuación se presenta la imagen del editor:



Como el programa está realizado en Javascript y trabaja únicamente del lado del cliente, lo podremos utilizar en cualquier tipo de servidor.

La desventaja es que funciona únicamente en versiones de Internet Explorer 5.5 o superiores. Por lo menos, en otros navegadores, no dará errores, sino que simplemente veremos un campo <textarea> normal.

CAPÍTULO 5



DISEÑO Y DESARROLLO DEL SISTEMA

Se incluye el diagrama de flujo de datos por cada tipo de usuario (Administrador, Desarrollador, Coordinador, Profesor, Alumno Ordinario y Alumno Extraordinario), el diagrama de entidad-relación, el diccionario de datos, los catálogos, las tablas y vistas que contiene la Base de Datos.

Se desarrolló el sistema por módulos, desarrollando cada uno por separado, realizándole pruebas individuales y de integración para corregir posibles fallas.

Se presentan las fases de pruebas que se le aplicaron al sistema y el mantenimiento correctivo otorgado, finalizando con las imágenes de las páginas principales del sistema como son: las páginas para administrar usuarios, asignaturas y grupos, las páginas de autenticación, las páginas de consulta de tutoriales y cuestionarios, la página para la aplicación de exámenes y muestra de resultados, etc.

5.4. DIAGRAMA DE FLUJO

Se presenta el Diagrama de Flujo por cada tipo de usuario del sistema:

1. Administrador. El Administrador podrá dar de alta, baja y modificar usuarios además de dar de alta asignaturas y los grupos relacionados a dichas asignaturas.

2. Desarrollador. El Desarrollador únicamente podrá modificar los tutoriales y cuestionarios que haya creado, podrá ver todos los tutoriales y hacer cualquier consulta.

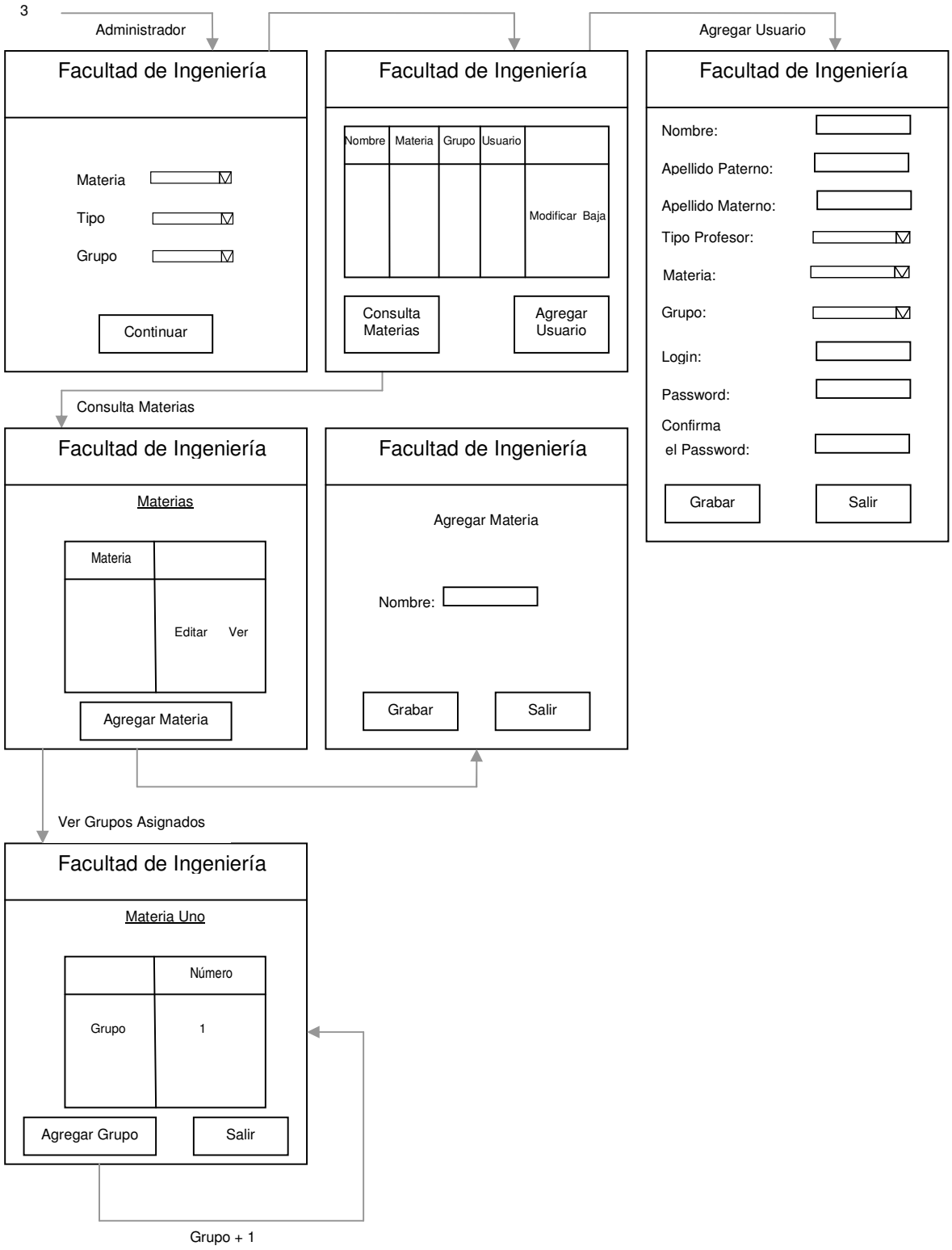
3. Profesor. El Profesor podrá ver los tutoriales y cuestionarios de las asignaturas que imparta, así mismo podrá hacer consultas a las mismas.

4. Coordinador. El coordinador de Área podrá ver los tutoriales, los cuestionarios y hacer cualquier consulta.

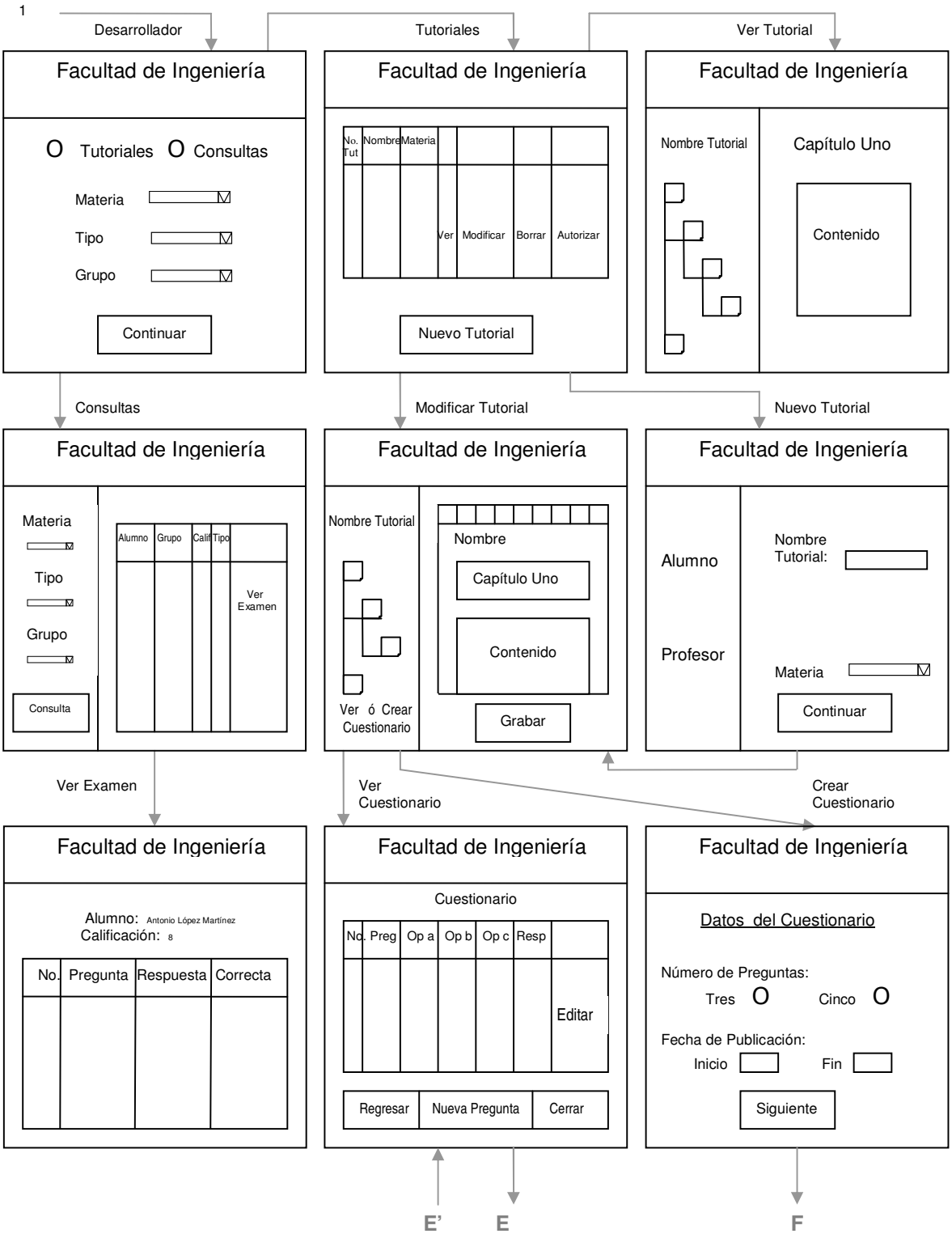
5. Alumno Ordinario. Los alumnos ordinarios podrán ver los tutoriales de las asignaturas a las que estén inscritos y resolver los exámenes departamentales de dichas asignaturas

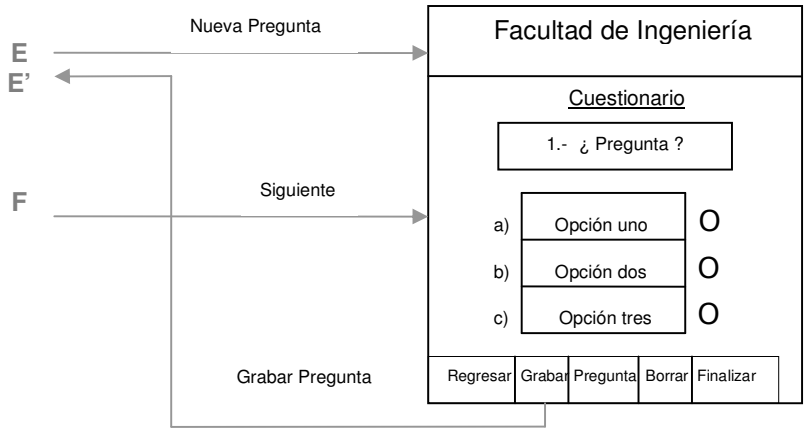
6. Alumno Extraordinario. Los alumnos extraordinarios podrán ver los tutoriales de las asignaturas a las cuales realizaron su trámite y se encuentran inscritos para presentar dicho examen.

5.4.1. Administrador

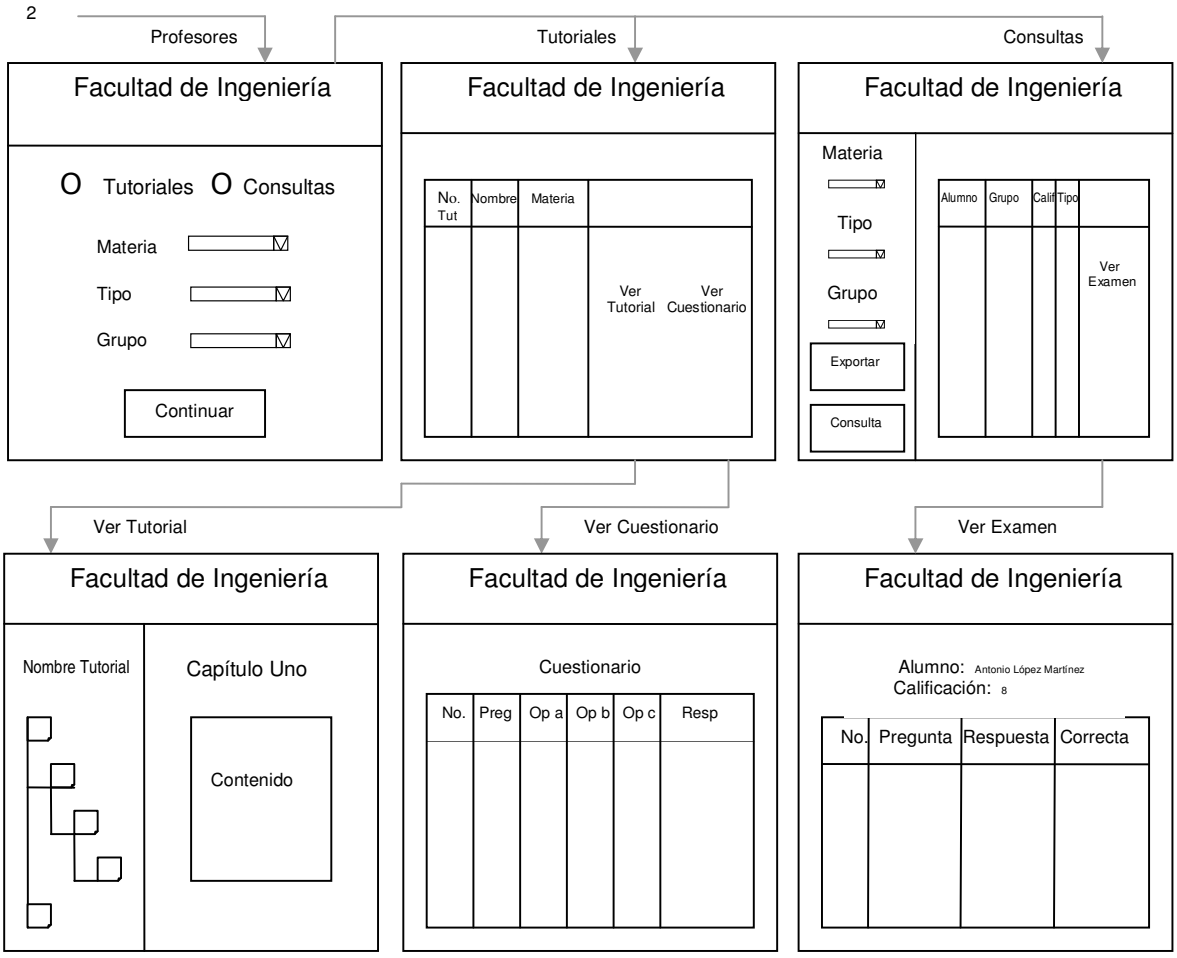


5.4.2. Desarrollador

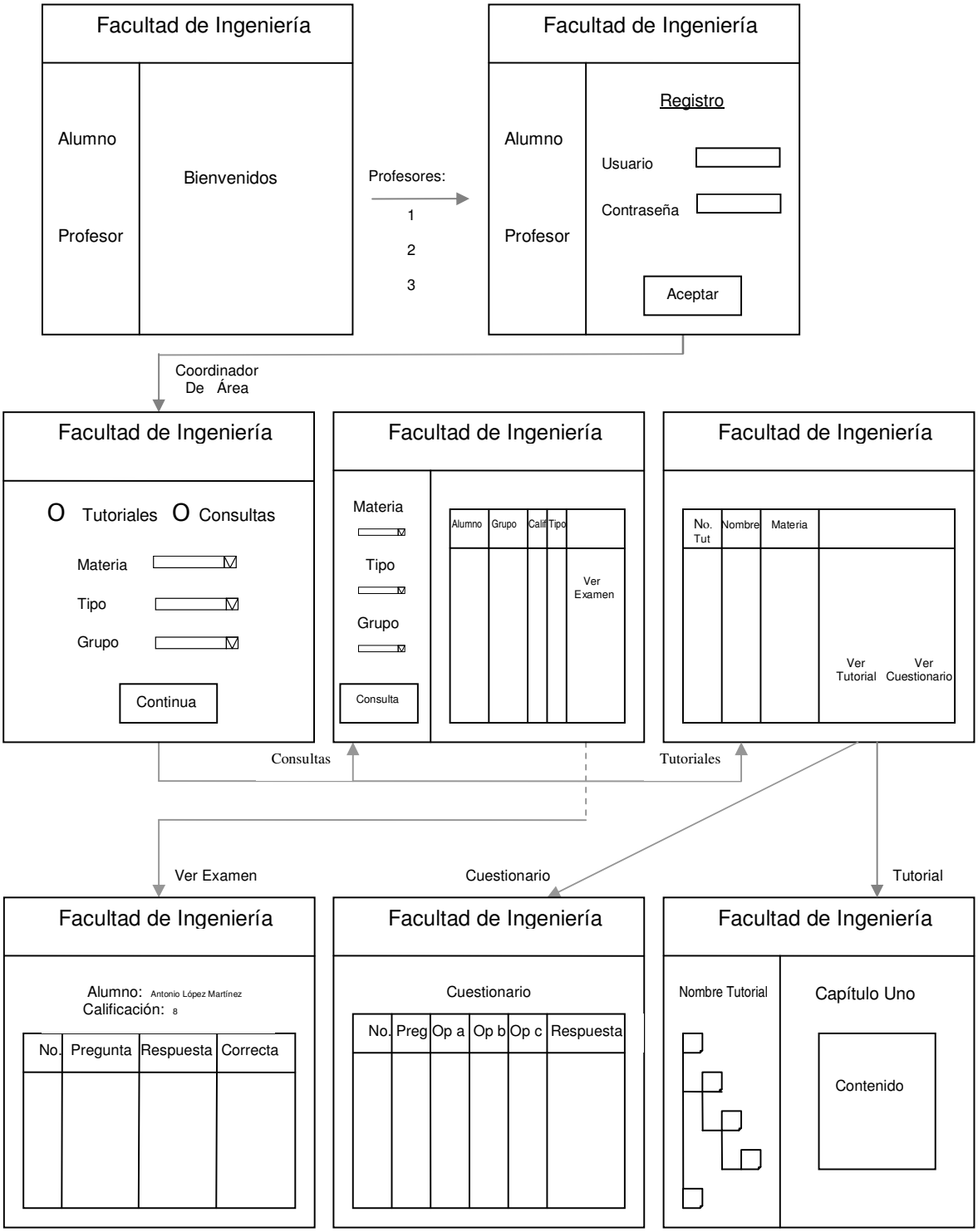




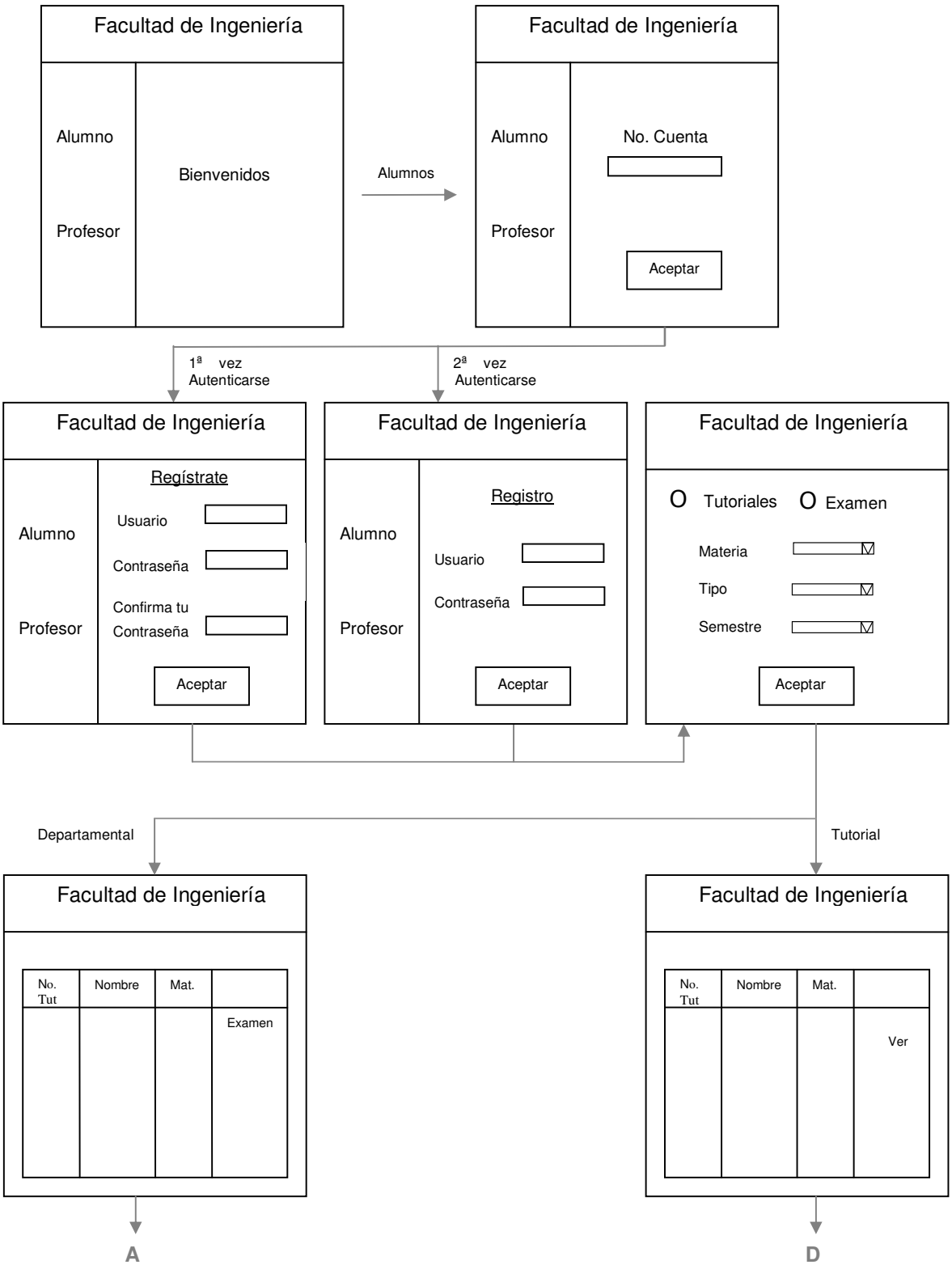
5.4.3. Profesor

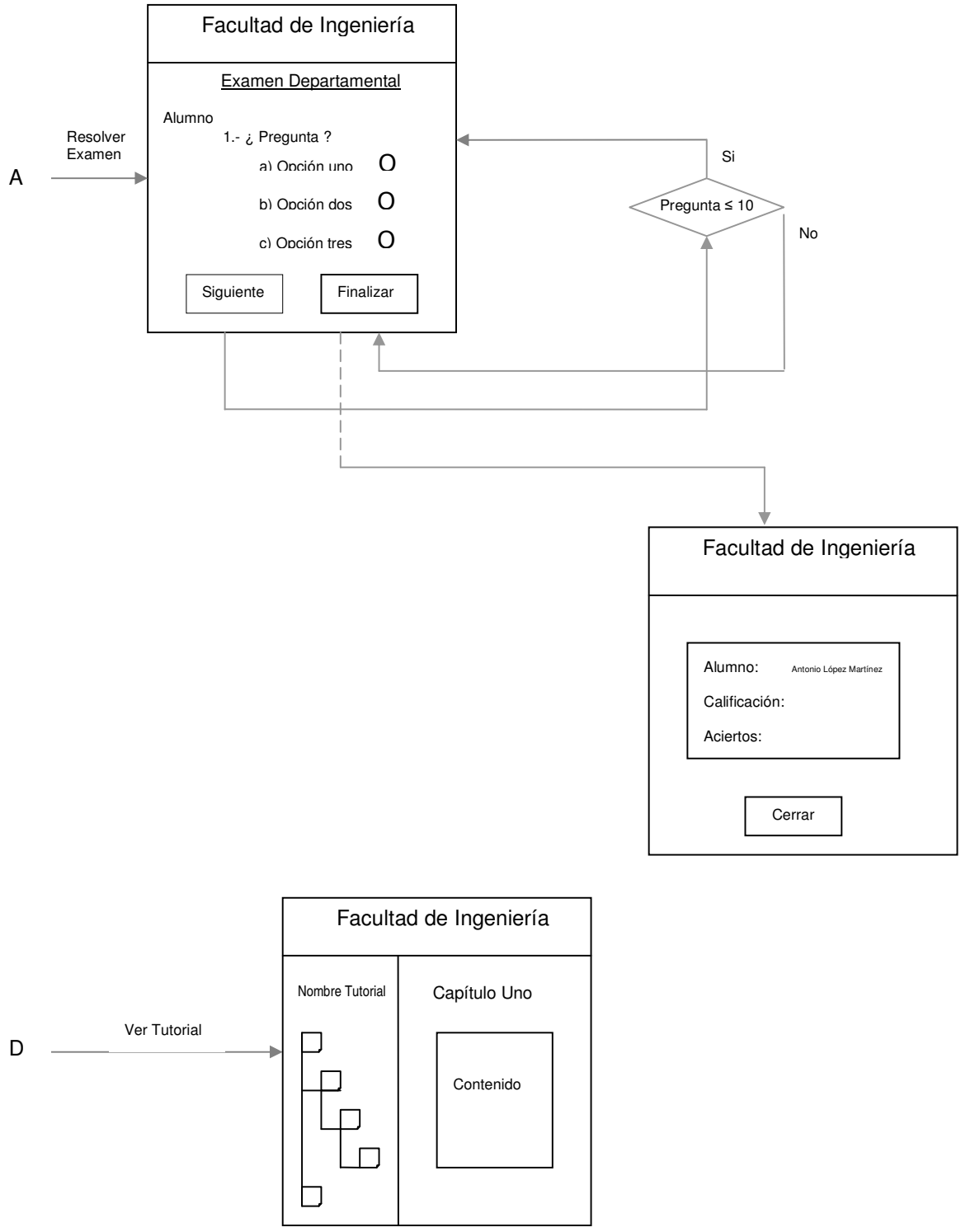


5.4.4. Coordinador

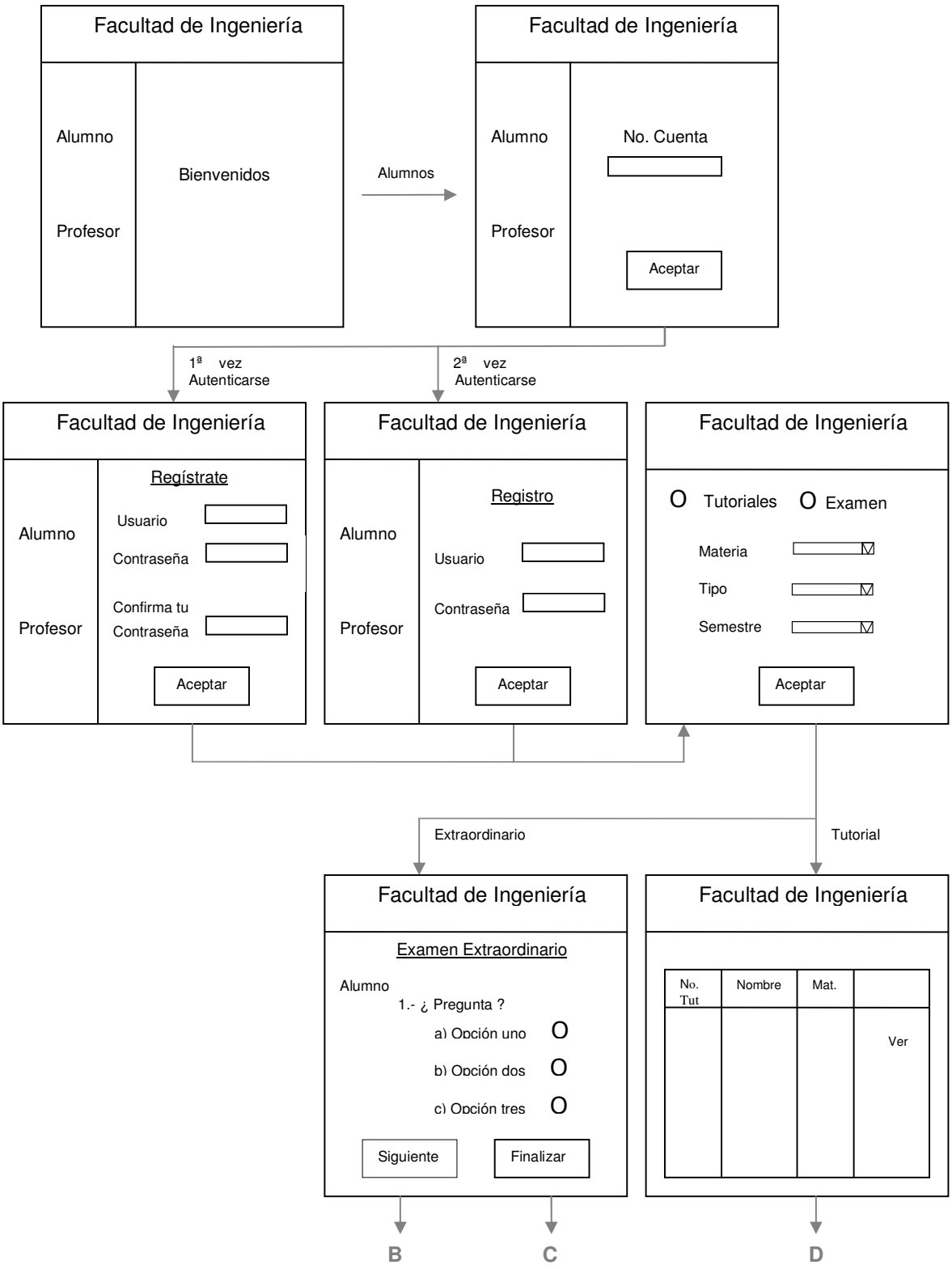


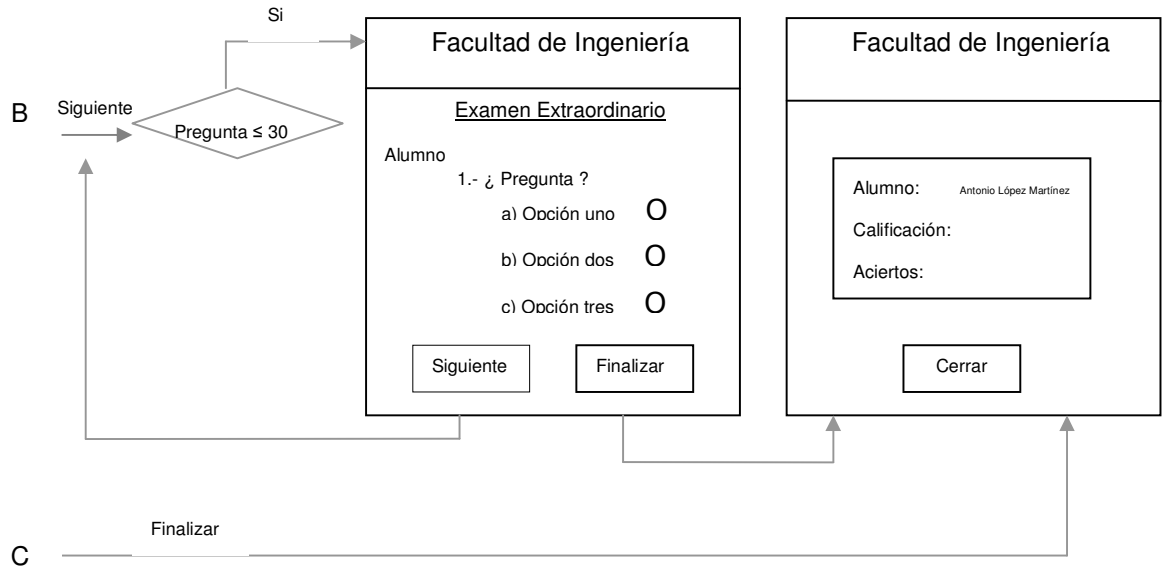
5.4.5. Alumno Ordinario

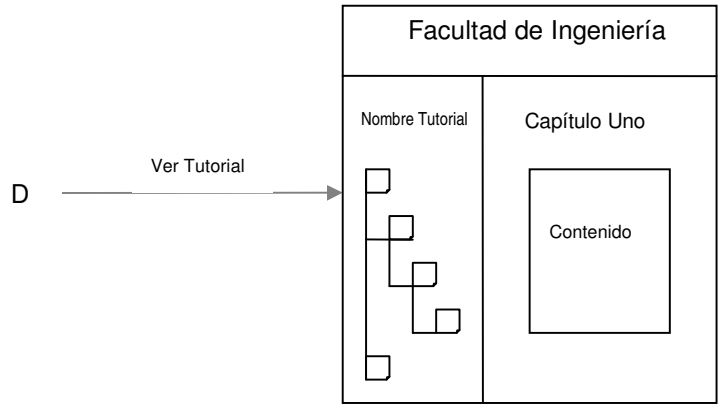



Facultad de Ingeniería
Examen Departamental
Alumno
1.- ¿ Pregunta ?
a) Opción uno
b) Opción dos
c) Opción tres
Siguiente
Finalizar
Resolver Examen
A
Si
Pregunta ≤ 10
No
Facultad de Ingeniería
Alumno: Antonio López Martínez
Calificación:
Aciertos:
Cerrar
Facultad de Ingeniería
Nombre Tutorial
Capítulo Uno
Contenido
Ver Tutorial
D

5.4.6. Alumno Extraordinario







5.5. MODELO DE DATOS

Es un conjunto de herramientas que permiten describir los datos, sus relaciones entre ellos y su significado. Éstos se dividen en tres grupos principalmente:

- Modelos lógicos basados en objetos
- Modelos lógicos basados en registros
- Modelo físico de datos

Utilizaremos el modelo lógico basado en objetos, para nuestros fines prácticos. Encontramos que se utiliza para describir los datos y sus relaciones de manera conceptual y de visión final.

Son una herramienta fundamental para el diseño conceptual al que se refieren administradores, analistas, programadores y algunos usuarios finales.

Dentro de estos modelos, se tiene, el Modelo Entidad-Relación, tiene su implementación gráfica en el Diagrama Entidad-Relación, que a continuación se mostrará.

5.5.1. Diagrama Entidad-Relación

Este diagrama es el resultado del análisis que se realizó para el sistema:

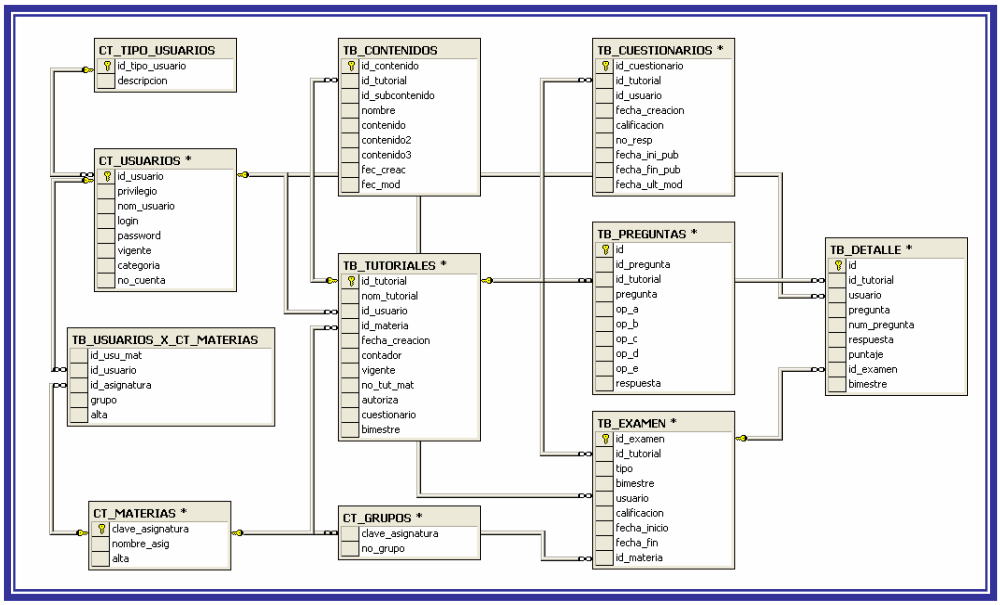


Figura 5.1. Diagrama Entidad-Relación

5.6. DICCIONARIO DE DATOS

La nomenclatura es la siguiente:

- Para los Catálogos, se utiliza: ct_nombre de la tabla
- Para las Tablas, se utiliza: tb_nombre de la tabla
- Para las Vistas, se utiliza: view_nombre de la vista

5.6.1. Catálogos

En los catálogos es donde se tiene la información que se utiliza para consultarla reiteradamente. Se tienen los siguientes catálogos:

- De grupos.
- De asignaturas.
- De tipo de usuario.
- De usuario.

Se presenta por medio de una tabla la descripción de cada catálogo:

CT_GRUPOS				
Campo	*Tipo de Dato*	*Tamaño*	*Identidad*	*Descripción*
clave_asignatura	Int	4	No	Identificador de la asignatura
no_grupo	Int	4	No	No. de grupo

Figura 5.2. Tabla de catalogo de grupos.

CT_MATERIAS				
Campo	*Tipo de Dato*	*Tamaño*	*Identidad*	*Descripción*
clave_asignatura	int	4	No	Identificador de la asignatura
nombre_asig	varchar	100	No	Nombre de la asignatura
alta	bit	1	No	Bandera que indica si esta dada de alta o baja la asignatura

Figura 5.3. Tabla de catalogo de materias.

CT_TIPO_USUARIO				
Campo	*Tipo de Dato*	*Tamaño*	*Identidad*	*Descripción*
id_tipo_usuario	Int	4	Si	Identificador del tipo de usuario
descripcion	Varchar	50	No	Descripción del tipo de usuarios

Figura 5.4. Tabla de catalogo por tipo de usuario.

CT_USUARIOS				
Campo	*Tipo de Dato*	*Tamaño*	*Identidad*	*Descripción*
id_usuario	int	4	Si	Identificador del usuario
privilegio	int	4	No	Tipo de usuario
nom_usuario	varchar	50	No	Nombre completo del usuario
login	varchar	50	No	Usuario de autenticación
password	varchar	50	No	Contraseña del usuario
vigente	bit	1	No	Bandera para saber si esta vigente o no el usuario
categoría	char	1	No	Bandera para saber si es alumno o profesor
no_cuenta	varchar	10	No	No. de cuenta de los usuarios alumnos

Figura 5.5. Tabla de catalogo por usuario.

A continuación se presenta un ejemplo de uno de los catálogos creados en la Base de datos de SQLServer:

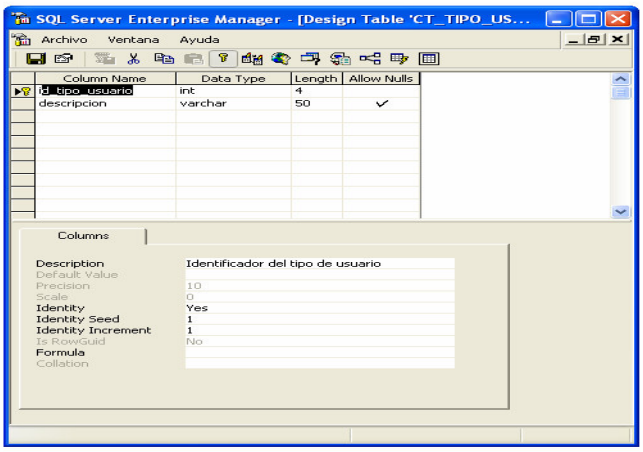


Figura 5.4.1. Catálogo por tipo de usuario en SQLServer.

5.6.2. Tablas

En las tablas es donde se tiene la información que se utiliza para guardar información que puede cambiar con frecuencia.

Se tienen las siguientes tablas:

- De cuestionarios.
- De detalle.
- De examen.
- De tutoriales.
- De usuarios por materias.

A continuación se muestra la descripción por cada tabla:

TB_CUESTIONARIOS				
Campo	*Tipo de Dato*	*Tamaño*	*Identidad*	*Descripción*
id_tb_cuestionarios	int	4	Si	Identificador de la tabla tb_cuestionarios
id_tutorial	int	4	No	Identificador de la tabla tb_tutoriales
id_usuario	int	4	No	Identificador de la tabla ct_usuarios
fecha_creacion	datetime	8	No	Fecha de creación del cuestionario
no_resp	int	4	No	No. de respuestas
fecha_ini_pub	datetime	8	No	Fecha de inicio de la publicación
fecha_fin_pub	datetime	8	No	Fecha de fin de la publicación
fecha_ult_mod	datetime	8	No	Fecha de ultima modificación

Figura 5.6. Tabla de catalogo por usuario.

TB_DETALLE				
Campo	*Tipo de Dato*	*Tamaño*	*Identidad*	*Descripción*
id	int	4	Si	Identificador de la tabla tb_detalle
id_tutorial	int	4	No	Identificador de la tabla tb_tutoriales
usuario	int	4	No	Identificador de la tabla ct_usuarios
pregunta	int	4	No	Identificador de la tabla tb_preguntas
num_pregunta	int	4	No	No. consecutivo de las preguntas de un cuestionario
respuesta	int	4	No	Respuesta del alumno
puntaje	int	4	No	Bandera para saber si la respuesta es correcta
id_examen	int	4	No	Identificador de la table tb_examen
bimestre	int	4	No	No. del bimestre

Figura 5.7. Tabla de catalogo por detalle.

TB_EXAMEN				
Campo	*Tipo de Dato*	*Tamaño*	*Identidad*	*Descripción*
id_examen	int	4	Si	Identificador de la tabla tb_examen
id_tutorial	int	4	No	Identificador de la tabla tb_tutorial
tipo	int	4	No	Bandera para saber si es un examen departamental o extraordinario
bimestre	int	4	No	No. de bimestre
usuario	int	4	No	Identificador de la tabla ct_usuarios
calificación	decimal	9	No	Calificación del alumno
fecha_inicio	datetime	8	No	Fecha de inicio de examen
fecha_fin	datetime	8	No	Fecha de termino del examen
id_materia	int	4	No	Identificador de la tabla ct_materias

Figura 5.8. Tabla de catalogo de exámenes.

TB_TUTORIALES				
Campo	*Tipo de Dato*	*Tamaño*	*Identidad*	*Descripción*
id_tutorial	int	4	Si	Identificador de la tabla tb_tutoriales
nom_tutorial	varchar	50	No	Titulo del tutorial
id_usuario	int	4	No	Identificador de la tabla ct_usuarios
id_materia	int	4	No	Identificador de la taba ct_materias
fecha_creacion	smalldatetime	4	No	Fecha de creación del tutorial

contador	bigint	8	No	Contador del numero de usuarios que consultan este tutorial
vigente	bit	1	No	Bandera que indica si el tutorial esta vigente
autoriza	bit	1	No	Bandera que indica si se puede publicar o no el tutorial
cuestionario	bit	1	No	Indica si existe un cuestionario para este tutorial
bimestre	int	4	No	No. de bimestre

Figura 5.9. Tabla de catalogo de tutoriales.

TB_USUARIOS_X_CT_MATERIAS				
Campo	*Tipo de Dato*	*Tamaño*	*Identidad*	*Descripción*
id_usu_mat	int	4	Si	Identificador de la tabla tb_usuarios_x_ct_materias
id_usuario	int	4	No	Identificador de la tabla ct_usuarios
id_asignatura	int	4	No	Identificador de la tabla ct_materias
grupo	int	4	No	No. de grupo
alta	int	4	No	Bandera que indica si esta dado de alta un usuario en alguna asignatura

Figura 5.10. Tabla de catalogo de usuarios por materias.

A continuación se presenta un ejemplo de una de las tablas creadas en la Base de datos de SQLServer:

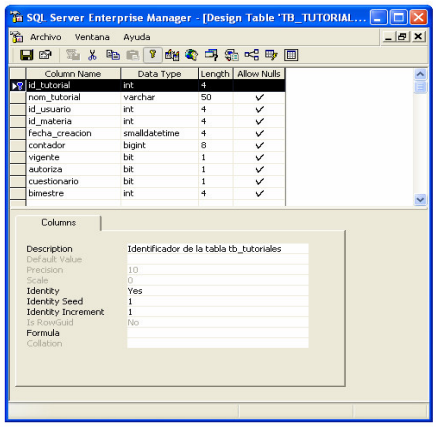


Figura 5.9.1. Tabla de Tutoriales.

5.6.3. Vistas

Las vistas muestran el diseño que se le aplica a cada tabla.

Se tienen las siguientes tablas:

- view_alumnos_examen
- view_examen_detalle
- view_preguntas
- view_tutoriales_exist
- view_usuarios_materias

A continuación se presenta un ejemplo de una de las vistas creadas en la Base de datos de SQLServer:

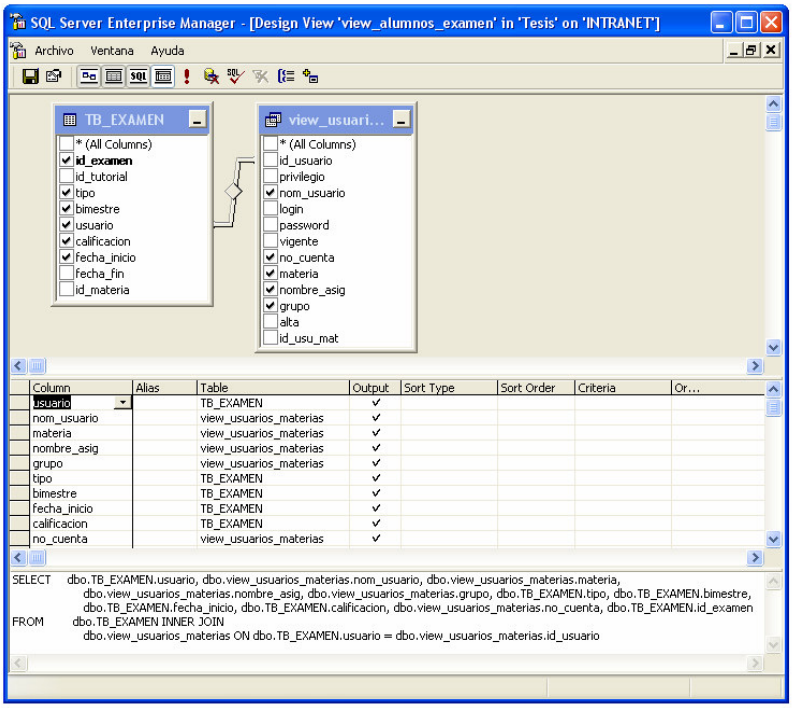


Figura 5.11. Vista que se aplica a la Tabla de exámenes.

5.7. PRUEBA DEL SISTEMA

Las pruebas que se le realizaron al sistema se dividieron en dos fases:

- Fase I: se realizó una presentación del Sistema Administrador y Creador de Tutoriales a un grupo de profesores en la Facultad de Ingeniería, mostrándoles el funcionamiento por tipo de usuario.
- Fase II: a un grupo de profesores, se les proporcionó claves y contraseñas para que pudieran acceder al Sistema Administrador y

Creador de Tutoriales y probar su funcionamiento por tipo de usuario durante tres semanas para corregir posibles desviaciones del mismo.

5.8. MANTENIMIENTO

Durante la explotación del software es necesario realizar cambios ocasionales, para corregir errores no detectados con anterioridad o para introducir mejoras en él, teniendo en este caso que rehacer parte de los trabajos anteriores.

Con los comentarios realizados por los profesores, se hicieron las adecuaciones que se consideraron pertinentes.

5.9. SISTEMA ADMINISTRADOR Y CREADOR DE TUTORIALES

Las pantallas principales del Sistema Administrador y Creador de Tutoriales son las siguientes:

- Pantalla de bienvenida.



- Pantalla de Autenticación para Administrador, Coordinadores, Desarrolladores y Profesores.



- Pantalla de consultas del Administrador.



- Tabla de resultados de la consulta.



- Pantalla para cambio de clave y contraseña de algún usuario.



- Pantalla para crear un nuevo usuario.



- Tabla de asignaturas que están dadas de alta en el sistema.



- Pantalla para editar el nombre de alguna Asignatura.



- Tabla de los grupos que tienen dados de alta las asignaturas.



- Pantalla para agregar una asignatura.



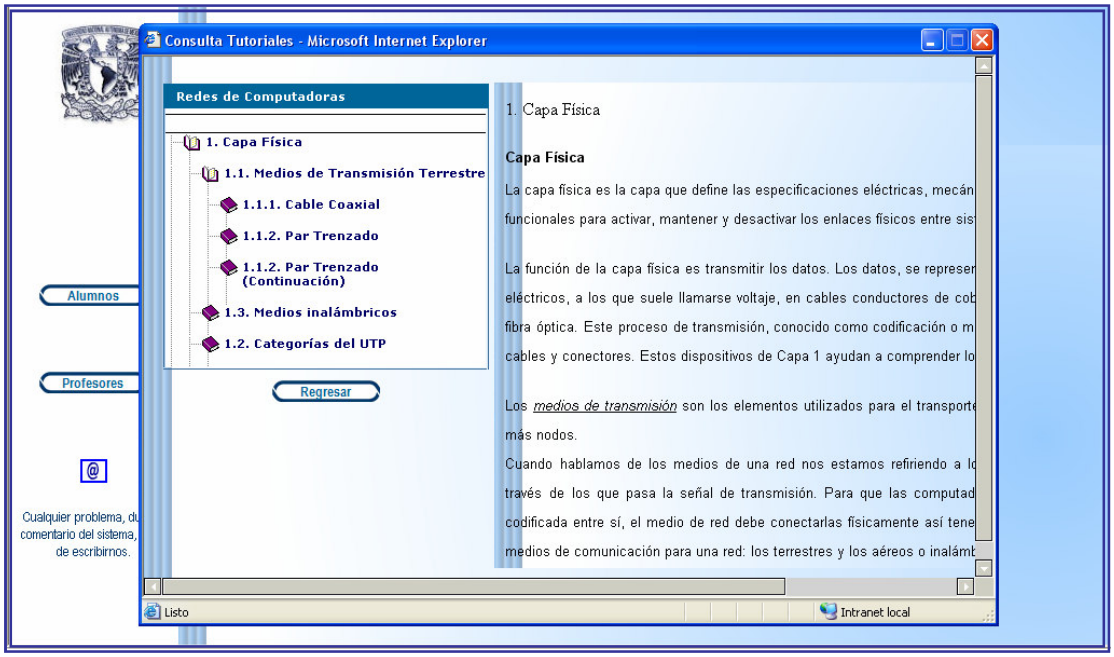
- Pantalla de consultas para coordinadores, profesores y desarrolladores.



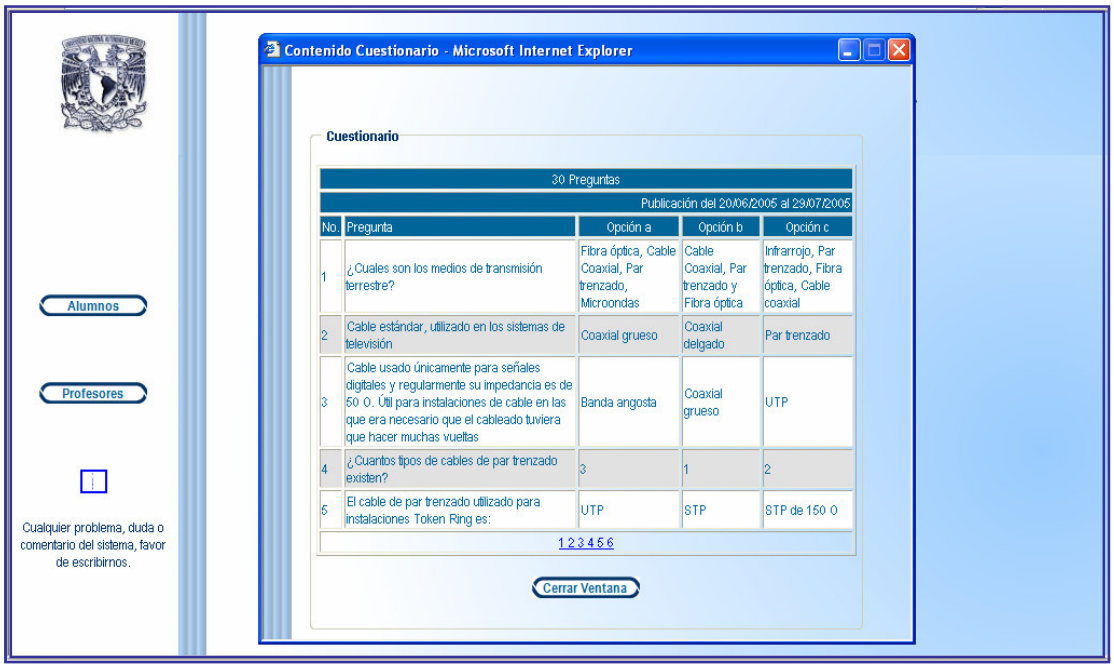
- Tabla para el coordinador de área, solo le muestra los tutoriales que están publicados, tiene la opción de ver el tutorial y el cuestionario.



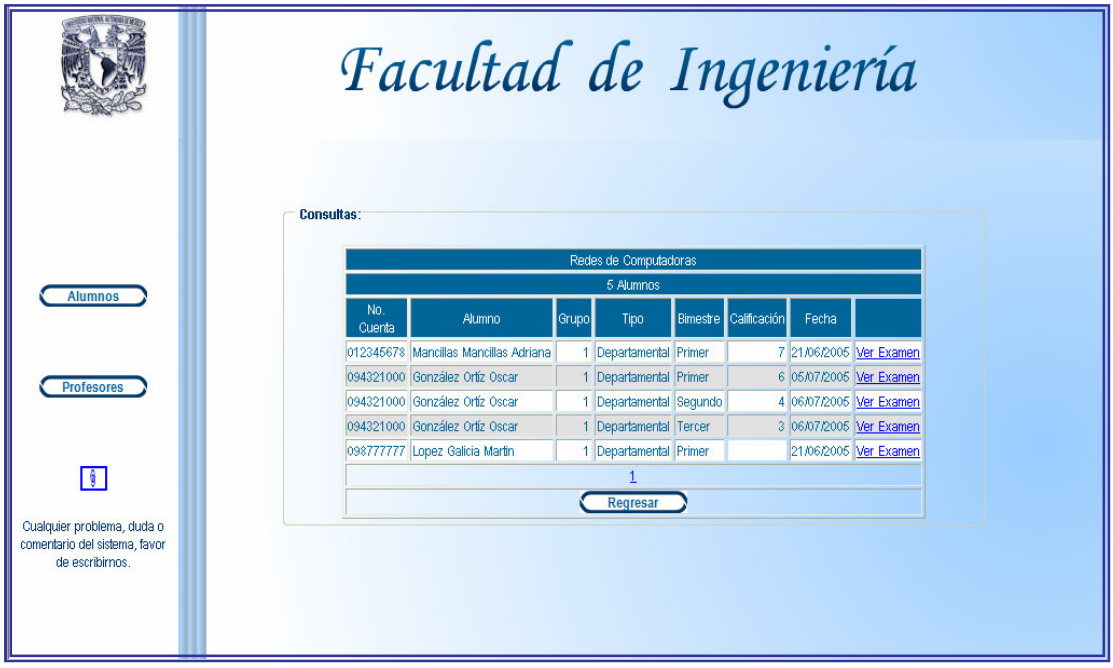
- Pantalla para consultar los tutoriales.



- Tabla para consultar el contenido de algún cuestionario.



- Tabla de alumnos que presentaron algún examen.



- Pantalla del examen realizado por un alumno.



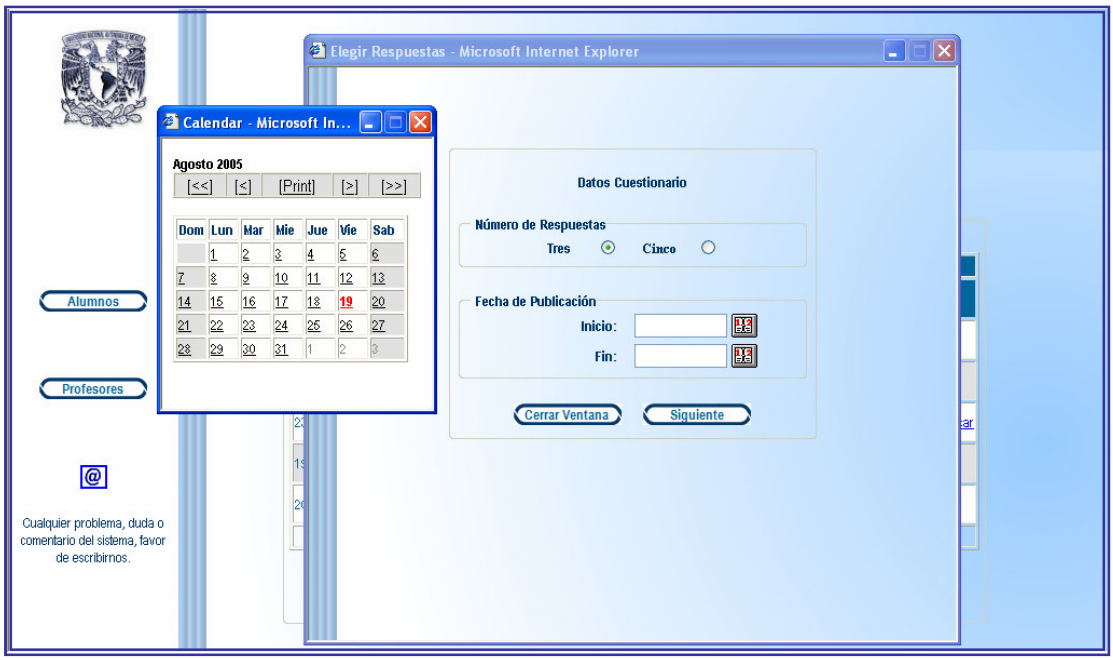
- Tabla para el desarrollador en donde tiene opción de ver ó modificar los tutoriales y cuestionarios que halla creado, así como decidir la publicación de los mismos, también podrá ver los otros tutoriales realizados por otros desarrolladores.



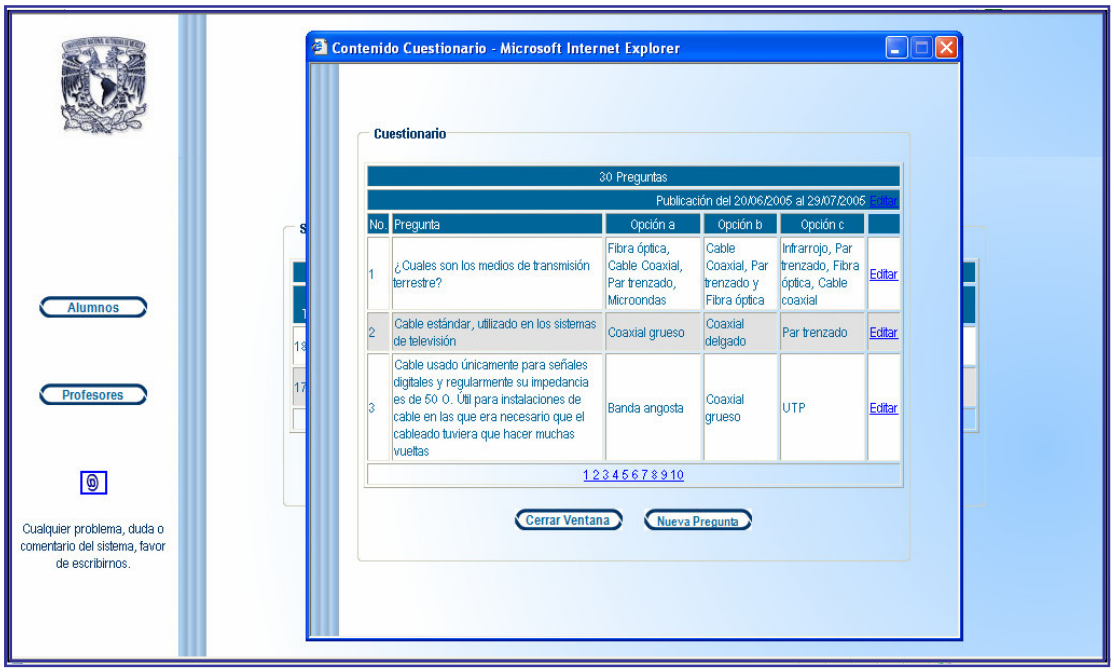
- Pantalla para asignar un nombre a un nuevo tutorial.



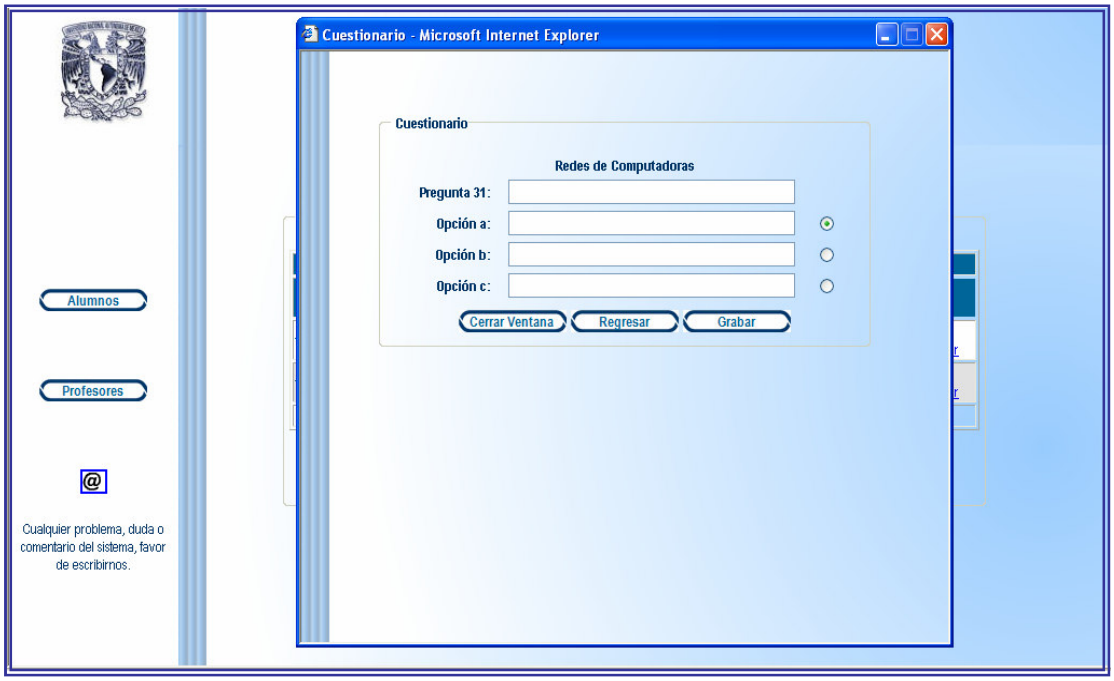
- Pantalla para crear un cuestionario.



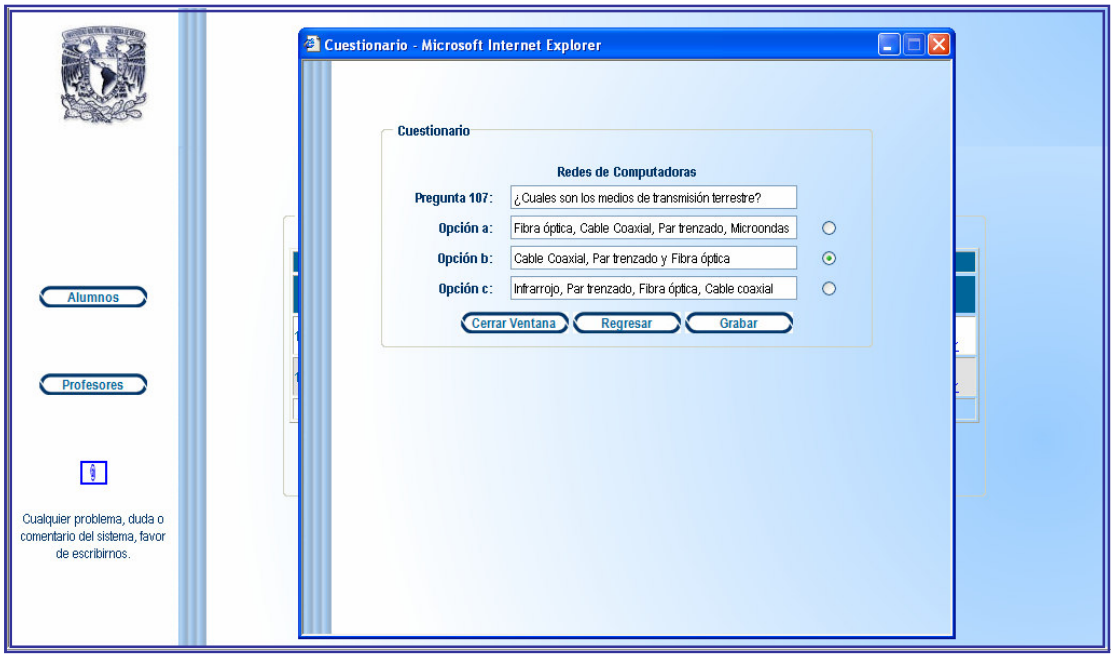
- Tabla del contenido del cuestionario para el desarrollador con opción a modificar las preguntas.



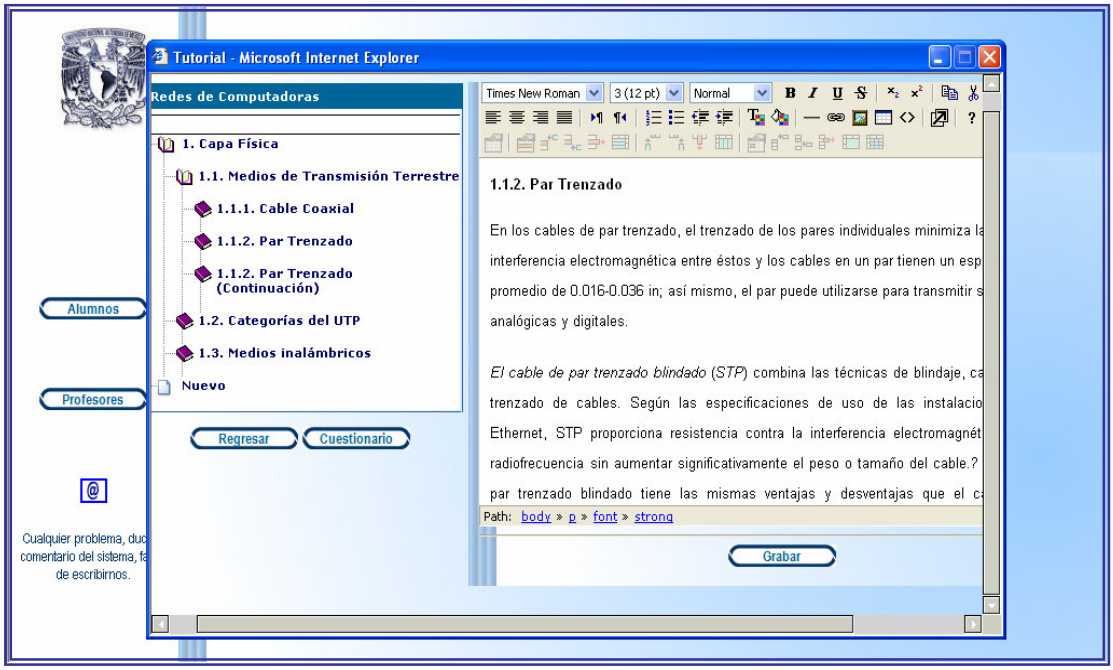
- Pantalla para generar una nueva pregunta.



- Pantalla para editar alguna pregunta.



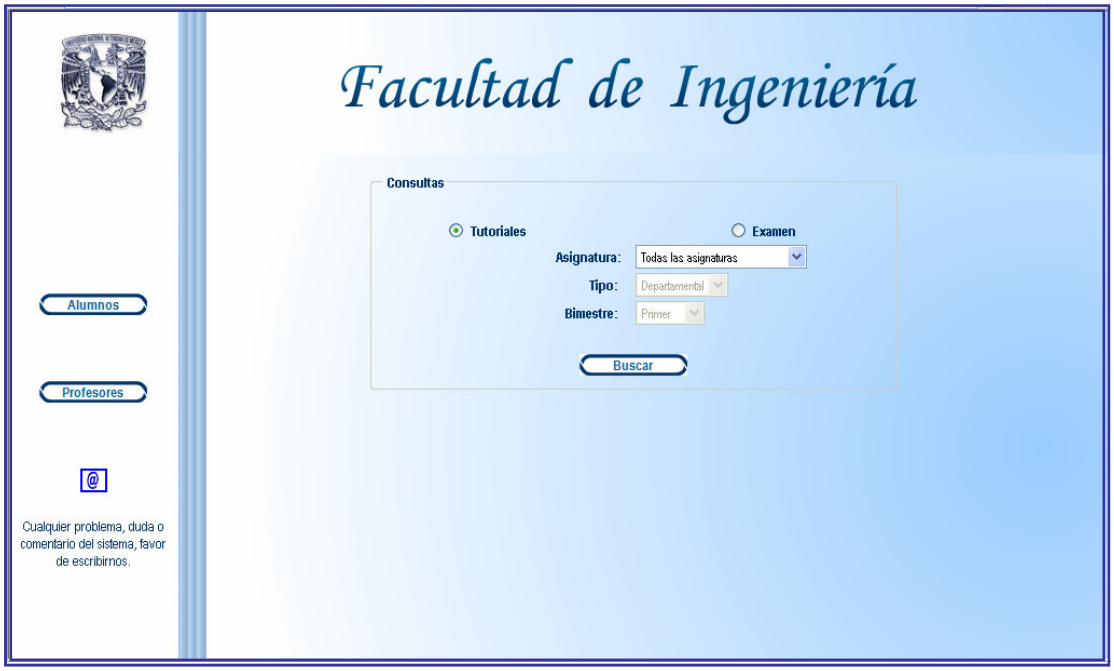
- Pantalla para modificar o editar algún tutorial.



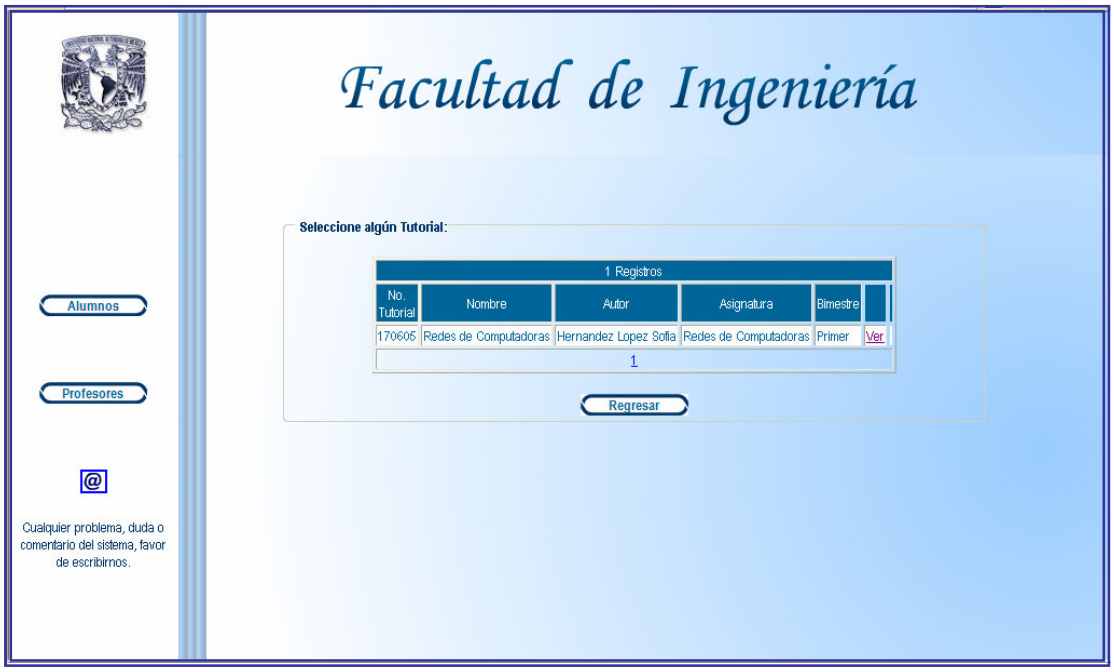
- Pantalla de autenticación para los alumnos ordinarios y extraordinarios. Autenticación por número de cuenta.



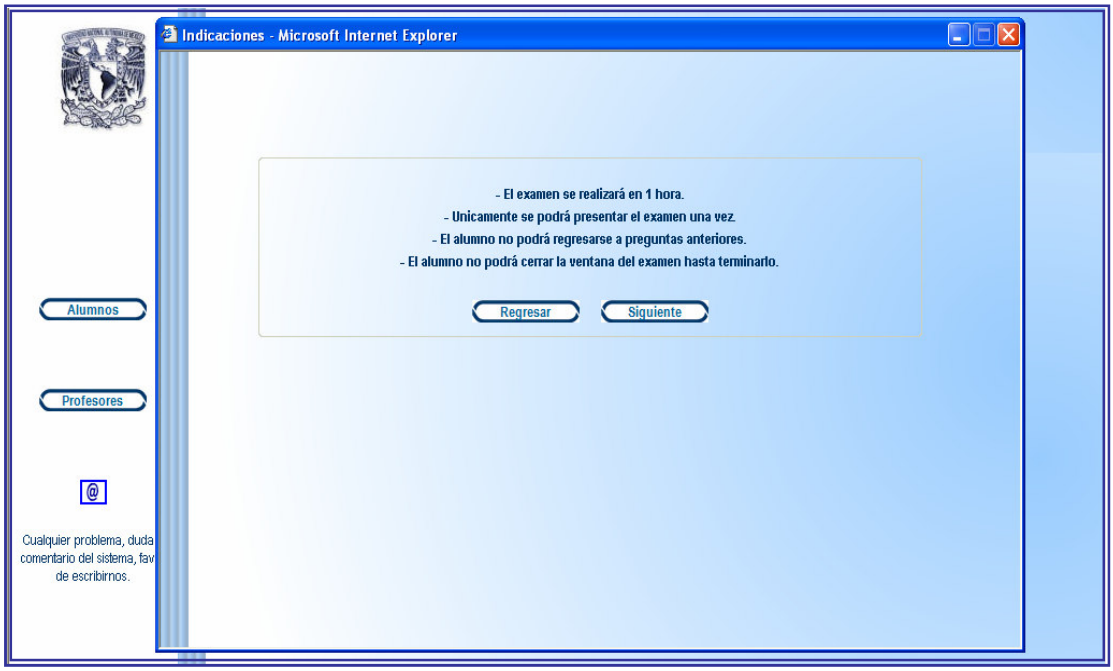
- Pantalla para consultar los tutoriales ó los exámenes realizados por los alumnos.



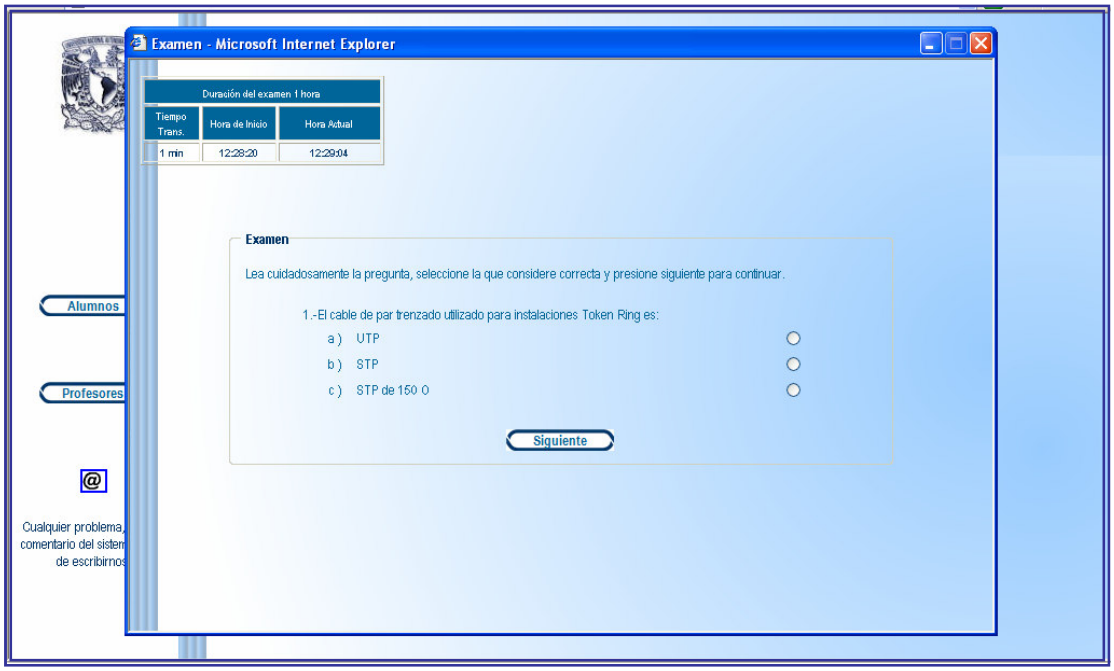
- Tabla de los tutoriales que están publicados.



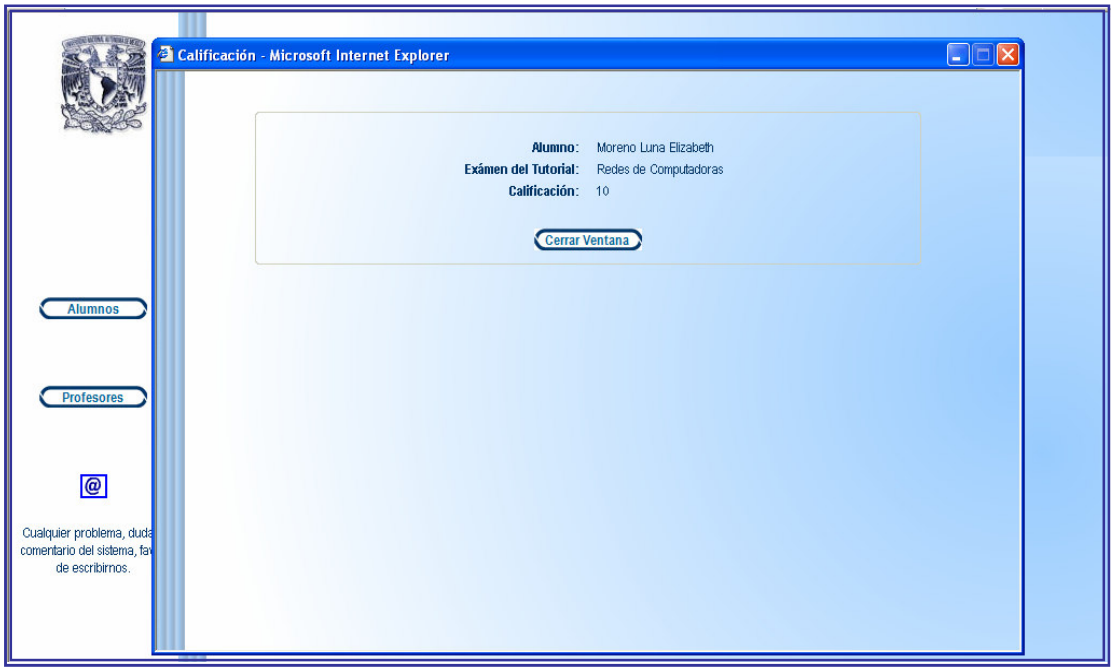
- Pantalla de indicaciones para la realización del examen.



- Pantalla para la presentación del examen.



- Calificación del examen.





CONCLUSIONES

Al tomar conciencia de la importancia de los enfoques psicopedagógicos en el proceso de aprendizaje y cómo estos nos pueden ayudar a la creación de software de alta calidad, se obtiene un producto que resulta mas atractivo a los educandos con los colores, tipo de letra y tamaño adecuados, con ello logramos mejores resultados en el proceso enseñanza – aprendizaje ya que se toman en cuenta aspectos que son importantes para que una página sea interesante para los alumnos, e incluso sea agradable a la vista para que se pueda leer y entender mejor el contenido de los temas propuestos.

La etapa inicial en el desarrollo de nuestro proyecto fue el conocer los diferentes tipos de paradigmas de programación que existen y elegir uno, utilizamos el ciclo de vida modificado ya que fue el que más se adaptó a nuestras necesidades. Este modelo nos permitió hacer un análisis y diseño del problema a resolver, así como una codificación por módulos que al final se integraron para dar como resultado un software muy intuitivo y fácil de utilizar, además de que hicimos programas que facilitan el mantenimiento del sistema al empezar un nuevo semestre.

Se hicieron pruebas individuales por módulo y de integración, además se hizo una presentación del sistema y tomamos en cuenta los comentarios que nos hicieron los profesores que asistieron a dicha presentación, a fin de mejorar la calidad del software y brindar funcionalidades no consideradas, por lo que la naturaleza del paradigma nos permitió hacer las correcciones necesarias ya que es posible regresar a cualquier etapa.

Con base en las pruebas realizadas al sistema por parte de los profesores, concluimos que nuestros objetivos respecto a lo que el sistema puede proporcionar se cumplieron. Y los objetivos orientados al uso del sistema por parte de los profesores y alumnos se cumplirán a medida que el sistema sea utilizado.

Con este lenguaje se hizo la extracción de información con pocas líneas de código lo que hace muy ligera la aplicación, además utilizamos la librería aspuploads para subir imágenes al servidor y darle mejor presentación a los tutoriales, la implementación de esta librería fue muy sencilla pues se puede obtener de INTERNET de forma gratuita.

Elegimos el servidor de INTERNET IIS por su óptima reutilización de objetos por parte de su propio administrador, con lo que obtenemos un entorno más optimizado en lo que concierne a la utilización de recursos, otros servidores de INTERNET como Apache no cuentan con esta optimización. La elección del lenguaje de programación fue ASP ya que tienen soporte nativo para servidores IIS, permiten el acceso a la base de datos a través de objetos que administran esas llamadas, por lo que no se desperdician recursos a ese nivel, además son compatibles debido a que son de la familia Microsoft.

La base de datos que se eligió fue SQL Server ya que está diseñada para trabajar en entornos de INTERNET, además es una base muy estable puesto que su nivel de fallos es bajo y tiene el soporte de Microsoft con esto aseguramos que la información está segura, además es muy confiable ya que al contar con un método commit y rollback se garantiza que una transacción fue exitosa al 100%, con esto se asegura que la integridad los datos almacenados son confiables.

Al desarrollar un árbol como el de un explorador de Windows para la presentación del contenido de los tutoriales, se cumplió el objetivo de crear una visualización amigable y sencilla para el usuario final, ya que solo se presenta información específica de un capítulo, tema o subtema, por lo que se hace más fácil la navegación y consulta de información de algún determinado tema.

Al implementar una herramienta de edición de texto se le da al desarrollador la libertad de poder elegir como quiere que su información se visualice en la Web, con ciertas restricciones como el tipo de letra y fondo utilizados ya que no se pueden dejar de lado las recomendaciones de tipografía y colorimetría hechas por los psicólogos, con el objetivo de hacer más atractiva una página.

Para tener un mejor control de información en el sistema, se diseñaron siete módulos, en el primero se lleva a cabo el control de los usuarios, asignaturas y grupos, en el segundo se diseñan y alimentan los tutoriales que se crean, en el tercero se generan las preguntas que contendrán los exámenes, en el cuarto se visualizan los tutoriales creados y autorizados por su desarrollador, en el quinto se aplican los exámenes y las preguntas presentadas se generan de forma aleatoria y de opción múltiple, en el sexto se califica el examen y dicha calificación se le muestra al alumno y en el séptimo los profesores pueden ver la calificación de sus alumnos exportando esta información a un formato en Excel.

Con este sistema pretendemos dar una herramienta más a los profesores y alumnos para que puedan reforzar el conocimiento adquirido en las aulas, además, dichos apuntes van a estar evaluados por un grupo de profesores, con lo que se garantiza que el contenido de los tutoriales se basara en los temarios de las asignaturas correspondientes al nuevo plan de estudios, por lo que estarán

actualizados y ajustados a los temas aprobados por el consejo académico, esto contribuirá a formar ingenieros de primer nivel.

Puesto que los exámenes tienen un periodo de publicación, los profesores contarán con un elemento más de evaluación al darle a esta calificación distintos porcentajes según considere.

Este sistema se puede integrar a otros proyectos que sean diseñados dentro de un esquema de educación a distancia o E-learning ya que por si algún motivo un estudiante no puede presentarse a clase, esta herramienta le proporciona todos los elementos necesarios para poder estudiar los temas de las asignaturas que requiera desde el lugar en donde se encuentre, siempre y cuando tenga acceso a INTERNET así como su usuario y contraseña para ingresar al sistema, posteriormente podrá realizar sus exámenes, lo que permite a los profesores contar con material suficiente para la evaluación del alumno.

Es importante mencionar que este sistema presenta la información por semestre con el fin de optimizar el espacio que ocupara la información en el servidor en donde resida.

Además se debe tener en cuenta que existen factores que no se pueden controlar y que afectan el buen funcionamiento del sistema, como son: desastres naturales, hackers, fallas en el suministro de energía eléctrica, etc.

Por ejemplo si existe una falla en la energía eléctrica, y afecta al servidor donde reside la aplicación o en el equipo en donde se esta realizando un examen, el usuario pierde su derecho a presentarlo nuevamente, ya que desde el momento en que decide hacerlo y empezar a contestar la primera pregunta, el sistema registra que ese examen ya fue realizado. Esta restricción se hizo con el objetivo de eliminar la posibilidad de repetir el examen buscando aumentar su calificación.

Lo que no se debe esperar que el sistema proporcione, debido que no forma parte del propósito del mismo, son: estadísticos, gráficas, comparativos, etc.

Es muy grato para nosotras poder poner en práctica los conocimientos adquiridos en la facultad y así darle a ésta un sistema de calidad que apoye a la comunidad académica para seguir teniendo una formación de excelencia.



APÉNDICE

A

Abigarrado: De varios colores, mal combinados.

Active X: Componente que se puede insertar en una página Web para proporcionar una funcionalidad que no está directamente disponible en HTML, como secuencias de animación. Los controles ActiveX se pueden implementar en diferentes lenguajes de programación.

Son pequeños programas que permiten mostrar páginas web dinámicas en la PC y que suplen las limitaciones que, al respecto, tiene el lenguaje HTML. Los controles ActiveX tienen que descargarse al disco duro de la computadora para que los documentos que los utilizan puedan visualizarse.

ADC: (Analogue to Digital Converter). Conversor analógico a digital. Se utiliza para convertir el sonido analógico en información digital para que los aparatos de hoy en día la puedan manejar con fiabilidad suficiente para los usuarios.

ADO: (Data Access Object) Objeto de acceso a datos. Interface de acceso a datos usado para comunicar OLEDB data sources, como MS SQL Server. Es una Interface a nivel aplicación que usa OLEDB, una librería de Objetos COM que permite el acceso a diversas fuentes de datos.

Algoritmo: Conjunto ordenado y finito de operaciones que permite hallar la solución de un problema. Suelen expresarse a través de letras, cifras y símbolos, que forman un algoritmo determinado.

Dícese del procedimiento para resolver problemas en términos de las acciones a ejecutar o el orden en que se ejecutarán dichas acciones en un problema dado.

API: (Application Program Interface). Conjunto de convenciones internacionales que definen cómo debe invocarse una determinada función de un programa desde una aplicación. Cuando se intenta estandarizar una plataforma, se estipulan unos APIs comunes a los que deben ajustarse todos los desarrolladores de aplicaciones. Es una herramienta de programación para rutinas, protocolos y software.

Apodo: Nombre que suele darse a una persona, tomado de sus defectos corporales o de alguna otra circunstancia.

Aplicación: Programa preparado para una utilización específica, como el pago de nóminas, formación de un banco de términos léxicos, etc, son cada uno de los programas que, una vez ejecutados, permiten trabajar con la computadora. Son aplicaciones los procesadores de textos, hojas de cálculo, bases de datos, programas de dibujo, paquetes estadísticos, etc.

ASCII: (American Standard Code of Information Interchange) Código Normalizado Americano para el Intercambio de Información. Estándar aceptado casi mundialmente que recoge 128 caracteres, letras, números y símbolos utilizados en procesadores de textos y algunos programas de comunicaciones. Su principal ventaja es su amplia difusión y aceptación.

La mayoría de los procesadores de textos presentes en el mercado pueden importar y exportar ficheros a formato ASCII, lo que facilita el intercambio de información entre personas o empresas que no trabajan con la misma aplicación. El más utilizado es al ASCII extendido (de 8 bits) que permite representar 256 caracteres, como la ñ, vocales acentuadas, etc., frente al ASCII de 7 bits que solo permite representar 127 caracteres.

Awt: Abstract Window Toolkit. Es un paquete de clases para crear componentes como botones, menus y scrollbars de applets para aplicaciones únicas.

C

C: El lenguaje C es una herramienta de programación de tipo general, utilizada para el desarrollo del sistema operativo Unix. Fue realizado a principios de la década de los setenta por Dennis Ritchie, como evolución del lenguaje B que creara Ken Thompson.

C++ : Versión de C orientada a objetos creada por Bjarne Stroustrup. C++ se ha popularizado porque combina la programación tradicional en C con programación orientada a objetos. Smalltalk y otros lenguajes originales de programación orientada a objetos no suministraban las estructuras familiares de lenguajes convencionales como C y Pascal.

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

Cuadratín: Es una medida tipográfica que equivale a un espacio de los mismos puntos que tenga el cuerpo de la letra donde está insertado.

D

DBMS: (Data Base Management System). Sistema de administración de bases de datos. Software que controla la organización, almacenamiento, recuperación, seguridad e integridad de los datos en una base de datos. Acepta solicitudes de la aplicación y ordena al sistema operativo transferir los datos apropiados.

Los DBMS pueden trabajar con lenguajes de programación tradicionales (COBOL, C, etc.) o pueden incluir su propio lenguaje de programación. Por ejemplo, dBASE y Paradox son programas de base de datos con un DBMS, un lenguaje completo de programación y un lenguaje de cuarta generación, haciendo de ellos sistemas completos de desarrollo de aplicaciones.

Los comandos de los lenguajes de cuarta generación permiten a los usuarios crear en forma interactiva archivos de bases de datos, editarlos, formular preguntas e imprimir informes sin necesidad de programación. Miles de aplicaciones han sido desarrolladas en ambientes como éstos.

Desarrollo cognoscitivo: La adquisición de destrezas complejas e inteligencia a través de la madurez y experiencias en el ambiente del individuo.

DHTML: Dinamic HTML, son aplicaciones que contienen objetos y eventos y se procesan en el lado del cliente dentro del navegador Web.

Diagrama de flujo: Representación gráfica, mediante la utilización de signos convencionales, del proceso que sigue la información en un programa determinado. Se utilizan habitualmente en la fase de desarrollo de aplicaciones por los programadores.

El diagrama de flujo o Flowchart es una de las técnicas de representación de algoritmos más antigua, y consiste en representar mediante símbolos las operaciones a realizar. Por ejemplo: el inicio y el fin del algoritmo se representan con un símbolo elíptico, las entradas y salidas con un paralelogramo, las decisiones con un rombo, los procesos con un rectángulo, etc.

Driver: Los drivers o controladores, son los encargados de actuar como interfaz entre el sistema operativo y los dispositivos que componen una computadora. De esta forma todos los componentes del PC se entienden y trabajan conjuntamente.

E

Excel: Hoja de cálculo con todas las facilidades para computadora personales y Macintosh de Microsoft. Puede enlazar varias hojas de cálculo para su consolidación y provee una amplia variedad de gráficos y diagramas comerciales, para crear materiales de presentación.

Extranet: Es una red de colaboración que utiliza la tecnología Internet. Esta interconecta a una empresa con sus proveedores, clientes u otros socios. El término ha sido acuñado por Jim Barksdale y Mark Andreesen para describir el software que facilita la relación entre diferentes compañías.

La tecnología Extranet puede ser concebida como parte de una Intranet que es accesible para otras empresas o como una herramienta que permite la colaboración entre empresas, la información compartida puede ser sólo para aquellos miembros colaboradores de la empresa que poseen Intranet y en algunos casos éstos podrían ser públicos.

H

HTML: (HyperText Markup Language). Lenguaje de marcado de Hipertexto. Es el lenguaje estándar para describir el contenido y la apariencia de las páginas, es decir, lenguaje que se usa para crear documentos World Wide Web.

HTTP: (HyperText Transport Protocol). Protocolo para la transferencia de hipertexto. Es el protocolo de comunicaciones que permite recuperar con rapidez los documentos de hipertexto World Wide Web.

Hypertext: Hipertexto. Texto que contiene elementos a partir de los cuales se puede acceder a otra información, o un documento que contiene palabras o gráficos que contienen vínculos a otros documentos. Por lo regular, la selección del área de vínculo en pantalla (con un ratón o comando del teclado) activa estos vínculos.

I

IIS: (Internet Information Server). Servidor de Información de Internet. Proporciona la forma más fácil de compartir información, crear y distribuir aplicaciones de negocio, así como alojar y administrar sitios Web.

Incidental: Que sobreviene en algún asunto y tiene alguna relación con él.

Inherente: Que por su naturaleza está de tal manera unido a algo, que no se puede separar de ello.

Inferencia: Es la operación intelectual por medio de la cual se llega a una conclusión a partir de una verdad ya conocida. Se puede realizar a través de deducciones o inducciones y se puede aplicar a textos orales y escritos.

Intranet: Red propia de una organización, diseñada y desarrollada siguiendo los protocolos propios de Internet, en particular el protocolo TCP/IP. Puede tratarse de una red aislada, es decir no conectada a Internet.

Una red de equipos que es interna a una organización y es compatible con aplicaciones de Internet, especialmente el WWW. La mayoría de las Intranet están configuradas de forma que sus usuarios puedan tener acceso a Internet sin permitir que los usuarios de Internet tengan acceso a los equipos de la Intranet.

Interactivo: Un sistema es interactivo cuando permite un diálogo continuo entre el usuario y la aplicación, respondiendo ésta a las órdenes de aquél

Interacción: Acción que se ejerce recíprocamente entre dos o más grupos, personas u otros agentes

Internet: Conjunto de redes de computadoras creada a partir de redes de menos tamaño, cuyo origen reside en la cooperación de dos universidades estadounidenses.

Es la red global compuesta de limes de redes de área local (LAN) y de redes de área extensa (WAN) que utiliza TCP/IP para proporcionar comunicaciones de ámbito mundial a hogares, negocios, escuelas y gobiernos.

Iteración: Cada una de las repeticiones de las acciones contenidas en un bucle de programa.

IP Address: Dirección IP. Es un número único asignado para identificar a un host en Internet (también llamado dirección IP o dirección de punto). Por lo regular esta dirección se representa como cuatro números entre 1 y 254 separados por puntos

J

J2SE: (Java 2 Platform, Standard Edition). Es una colección del API de Java utilizada sólo en programas con plataforma java. La java EE incluye todas las clases de la java SE, la diferencia es que son mas utilizados en los programas que corren en servidores o estaciones de trabajo.

JAVA: Un idioma de la programación de alto nivel desarrollado por la compañía Sun Microsystems.

L

Ladillo: Composición breve que suele colocarse en el margen de la plana, generalmente para indicar el contenido del texto.

LAN: (Local Area Network) Red de area local. Es una red de computadoras que está limitada a un área física reducida, como un edificio.

M

MAC: Macintosh. Serie de computadores personales de Apple. El Mac original con su gabinete vertical se introdujo en 1984. El sistema operativo del Mac con su interfaz gráfica de usuario ha proporcionado una medida de consistencia y facilidad de uso que es sin igual.

La familia Macintosh es la serie de computadores personales más grande, en uso, y no compatible con IBM. Hasta 1994, los Mac recibían potencia exclusivamente de la familia de CPU 680x0 de Motorola. En 1994, Apple introdujo los Power Macintosh (PowerMac), que usan la PowerPC CPU y proporcionan un desempeño mejorado.

Los PowerMac utilizan una versión PowerPC del Mac OS, que ejecuta aplicaciones PowerMac nativas y emula aplicaciones Mac 680x0. También ejecuta aplicaciones DOS y Windows a través de emulación de Insignia Solutions.

Método: Procedimiento para alcanzar un determinado fin.

N

Neologismo: Palabra, vocablo, acepción o giro nuevo en una lengua.

Netscape Navigator: Navegador WWW creado por la empresa norteamericana Netscape. Es uno de los navegadores Internet más difundidos que, como todos los programas navegadores hoy en día, integra en uno solo las aplicaciones de correo electrónico, chat y lector de noticias.

NET: Red

O

Obsoleto: Viejo, caduco, pasado de moda, anticuado, inadecuado a las circunstancias actuales.

ODBC: (Open Data Base Connection). Conexión abierta a bases de datos. Es una forma de conexión a bases de datos, independientemente del lenguaje o programa que estemos utilizando. Cada fabricante provee su propia librería (DLL) con las características de conexión a las bases de datos

Oracle 8i: Oracle es básicamente una herramienta cliente/servidor para la gestión de Bases de Datos. Es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hacen que sólo se vea en empresas muy grandes y multinacionales, por norma general.

OS: (Operating System) Sistema operativo. Programa o conjunto de programas que efectúan la gestión de los procesos básicos de un sistema informático, y permite la normal ejecución del resto de las operaciones.

El sistema operativo tiene tres grandes funciones: coordina y manipula el hardware de la computadora, como la memoria, las impresoras, las unidades de disco, el teclado o el mouse; organiza los archivos en diversos dispositivos de almacenamiento, como discos flexibles, discos duros, discos compactos o cintas magnéticas, y gestiona los errores de hardware y la pérdida de datos.

OS/2: OS/2 son las siglas de "Sistema operativo de segunda generación". La idea de OS/2 surgió entre IBM y Microsoft a mediados de los 80, en un intento de hacer un sucesor de MS-DOS.

OS/2 1.0 salió en abril de 1987 y era un sistema operativo de 16 bits, pues estaba pensado para trabajar sobre el microprocesador 286. Sin embargo, aprovechaba plenamente el modo protegido de la computadora, haciendo uso de sus capacidades para protección de memoria, gestión de multitarea, etc. El resultado fue un sistema operativo estable, rápido y muy potente.

P

Páginas dinámicas: Páginas Web que contienen, además de texto y gráficos, archivos de sonido, de vídeo, animaciones o elementos interactivos. Para su correcta reproducción, suele requerir la instalación e integración en el sistema de pequeñas aplicaciones especiales.

Paradigma: Un paradigma es una técnica, un modelo o un conjunto de herramientas para representar la solución de problemas específicos.

Proceso: Término general con que se designa toda prolongación, extensión o proyección.

Protocolo: Descripción técnica de un estándar, incluyendo las reglas de diseño y funcionamiento.

R

RAM: (Random Access Memory). Memoria de acceso aleatorio. Almacenamiento de trabajo de la computadora, que físicamente es una colección de chips RAM. Es un recurso importante de la computadora, ya que determina el tamaño y el número de programas que pueden ejecutarse al mismo tiempo, como también la cantidad de datos que pueden ser procesados instantáneamente.

Robusto: Diseñado para crear software altamente fiable. Para ello proporciona numerosas comprobaciones en compilación y en tiempo de ejecución. Sus características de memoria liberan a los programadores de una familia entera de errores.

ROM: (Read Only Memory). Memoria de sólo lectura. Chip de memoria que almacena permanentemente instrucciones y datos. Sus contenidos se crean en el momento de la fabricación y no se pueden alterar.

S

Script: Programa o macro que ejecuta un conjunto unitario de instrucciones que permite a una computadora realizar funciones diversas.

Secuencial: Referido a una sucesión regulada de procesos para conseguir un resultado; del latín, sequor, seguir.

Server: (servidor) Proveedor de un servicio. También se refiere con frecuencia a la pieza de hardware que proporciona acceso a la información que se le solicita.

Servidor: Genéricamente, dispositivo de un sistema que resuelve las peticiones de otros elementos del sistema, denominados clientes.

Computadora conectada a una red que pone sus recursos a disposición del resto de los integrantes de la red. Suele utilizarse para mantener datos centralizados o para gestionar recursos compartidos. Internet es en último término un conjunto de servidores que proporcionan servicios de transferencia de ficheros, correo electrónico o páginas WEB, entre otros.

Sistema: Es un programa de computadora que tiene capacidad para dar respuestas semejantes a las que daría un experto en la materia

Sistema operativo: Conjunto de programas fundamentales sin los cuales no sería posible hacer funcionar la computadora con los programas de aplicación que se desee utilizar. Sin el sistema operativo, la computadora no es más que un elemento físico inerte. (Véase OS)

SQL: Structured Query Language, una sintaxis usada por muchos sistemas del bases de datos recuperar y modificar información.

Software: Se refiere a todas las aplicaciones o programas que se encuentran funcionando en cualquier equipo computacional o de comunicación. Las aplicaciones o Software más comunes son desarrollados con fines administrativos (Procesadores de palabras, hojas de cálculo, contabilidad, entre otros).

Sosiego: Quietud, tranquilidad, serenidad.

Swing: es una Biblioteca gráfica para java que forma parte de las Java Foundation Classes (JFC). Incluye widgets para interfaz gráfica de usuario tales como cajas de texto, botones, desplegables y tablas.

T

Terabyte: Múltiplo del byte: un terabyte son 1.024 GigaBytes, cerca de un billón (un millón de millones) de bytes.

Test: Prueba destinada a evaluar conocimientos o aptitudes, en la cual hay que elegir la respuesta correcta entre varias opciones previamente fijadas.

U

UNIX: Es una familia de sistemas operativos tanto para computadoras personales como para mainframes. Soporta gran número de usuarios y posibilita la ejecución de distintas tareas de forma simultánea (multiusuario y multitarea). Su facilidad de adaptación a distintas plataformas y la portabilidad de las aplicaciones (está escrito en lenguaje C) que ofrece hacen que se extienda rápidamente.

Unix fue desarrollado a finales de los sesenta en los laboratorios Bell, y hasta principios de los ochenta su uso estuvo restringido fundamentalmente al entorno académico. La firma AT&T fue la primera en comercializarlo en 1983. Posteriormente, AT&T se salió del negocio UNIX en 1993, y Novell adquirió su versión System V de UNIX.

Unix puede compilarse en muchos y diferentes lenguajes de máquina, lo que le permite ejecutarse en una variedad más amplia de hardware en relación con cualquier otro sistema operativo. De esta forma, Unix se ha convertido en sinónimo de "sistemas abiertos".

URL: (Universal Resources Locator) Dirección de una cierta página de información dentro de Internet.

El URL es la cadena de caracteres con la cual se asigna dirección única a cada uno de los recursos de información disponibles en Internet. Existe un URL único para cada página de cada uno de los documentos de la World Wide Web.

V

VISUAL BASIC: Es un lenguaje de programación visual, también llamado lenguaje de 4ª generación. Esto quiere decir que un gran número de tareas se realizan sin escribir código, simplemente con operaciones gráficas realizadas con el ratón sobre la pantalla.

Es un programa basado en objetos, aunque no orientado a objetos como C++ o Java. La diferencia está en que visual Basic 6.0 utiliza objetos con propiedades y métodos, pero carece de los mecanismos de herencia y polimorfismo propios de los verdaderos lenguajes orientados a objetos como Java y C++.

VBScript: es el "Lenguaje Script" de Visual Basic. Hay lenguajes que tienen scripts, los cuáles son una especie de programación derivada de determinado lenguaje con el fin de economizar espacio, tiempo o recursos al realizar cierta actividad.

Una diferencia en cuanto a recursos por parte de VBScript es que no necesita los mismos archivos que el programa en sí, puesto que los Scripts son lenguajes de interpretación directa por la computadora, esto significa que no es compilado ni transformado a código binario para ser ejecutado por la máquina, únicamente son necesarias las librerías del Script y un linker para que el programa haga una compilación temporal del archivo.

W

WEB: Por éste término se suele conocer a WWW (World Wide Web), creado por el Centro Europeo de Investigación Nuclear como un sistema de intercambio de información y que Internet ha estandarizado. Supone un medio cómodo y elegante, basado en multimedia e hipertexto, para publicar información en la red. Inicial y básicamente se compone del protocolo http y del lenguaje html.

Win32: Interfaz de programación (API) para el modo de 32 bits del 386 que puede funcionar completamente en Windows NT. Muchas funciones también funcionan en Windows 3.1 y las aplicaciones pueden escribirse para el subconjunto Win32 para lograr un rendimiento mejorado en un 386 o superiores que ejecuta Windows 3.1 o superiores.

Windows: Es el nombre del popular entorno (no es un sistema operativo y no es una aplicación) software creado por Microsoft. Su novedad es el uso de diferentes pantallas que se superponen, denominadas ventanas, para mostrar distintos tipos

de información. Su implantación ha representado un gran avance en la facilidad de operación para los usuarios de computadoras personales.

Las características principales de Windows serían pues, precisamente, que es un entorno esto es, un software que permite compartir a las aplicaciones los recursos del sistema y que se manifiesta gráficamente por medio de ¡conos que hacen innecesarias las tradicionales pantallas llenas de texto. Windows permite, además, el intercambio de datos entre programas ejecutados en este entorno y la operación simultánea de varias aplicaciones.

Windows 95: Este sistema operativo, desarrollado por la compañía norteamericana Microsoft, ha sido calificado por muchos como «la gran revolución de la informática personal». Windows 95 supone un importante salto cualitativo respecto a los anteriores sistemas operativos para computadoras personales. De hecho, se trata de un entorno de 32 bits que integra DOS y Windows, y que resulta sumamente intuitivo y amigable. Según sus creadores, cualquier usuario sin conocimientos de informática puede utilizar una computadora con Windows 95, dada su facilidad de manejo.

Según sus detractores, Windows 95 no es más que una burda copia del sistema operativo que Apple desarrolló hace once años para sus computadoras Macintosh. Entre las características principales de Windows 95 se encuentran su capacidad multitarea, plug & play, gestión de entornos de 32 bits, programas directamente al escritorio, capacidades multimedia, etc.

Windows NT: (Windows New Technology). Nueva tecnología de Windows. Sistema operativo avanzado de 32 bits para redes, de Microsoft, que ejecuta aplicaciones DOS y Windows. Introducido en 1993, se ejecuta en 386 y superiores, MIPS, Alpha y plataformas PowerPC. Entre las características se incluyen redes par a par, multitarea preferente, multilectura, multiprocesamiento, tolerancia a fallas y soporte para el conjunto de caracteres Unicode. NT provee características amplias de seguridad e incluye protocolos de comunicaciones NetWare, TCP/IP y NetBEUI.

WWW: (World Wide Web). Telaraña o malla mundial. Sistema de información con mecanismos de hipertexto creado por investigadores del CERN. Los usuarios pueden crear, editar y visualizar documentos de hipertexto.

Wysiwyg: Son las siglas de una frase en inglés: «What you see is what you get», es decir, lo que se ve es lo que se obtiene. Es la propiedad de ciertos programas de presentar en pantalla exactamente lo mismo y en igual formato que como aparecerá impreso.

X

XHTML: (eXtensible Hyper Text Markup Language). Lenguaje extensible de marcado de hipertexto, es el lenguaje de marcado pensado para sustituir a HTML

como estándar para las páginas web. XHTML es la versión XML de HTML, por lo que tiene, básicamente, las mismas funcionalidades, pero cumple las especificaciones, más estrictas, de XML. Su objetivo es avanzar en el proyecto del World Wide Web Consortium de lograr una web semántica, donde la información, y la forma de presentarla estén claramente separadas.

En este sentido, XHTML serviría únicamente para transmitir la información que contiene un documento, dejando para hojas de estilo (como las hojas de estilo en cascada) y JavaScript su aspecto y diseño en distintos medios (computadoras, PDAs, teléfonos móviles, impresoras, etc.).

XML: (Extensible Markup Language). Es un meta-lenguaje que permite definir lenguajes de marcado adecuados a usos determinados. En la práctica corresponde a un estándar que permite a diferentes aplicaciones interactuar con facilidad a través de La Red.

XML fue creado al amparo del World Wide Web Consortium (W3C) organismo que vela por el desarrollo de WWW partiendo de las amplias especificaciones de SGML. Su desarrollo se comenzó en 1996 y la primera versión salió a la luz el 10 de febrero de 1998. La primera definición que apareció fue: Sistema para definir, validar y compartir formatos de documentos en la Web.



BIBLIOGRAFÍA

Capítulo 1

- Enseñanza programada en la educación activa
  Norman Balabanian
  Editorial PAX-MÉXICO
  México D. F. 1985
- [1] Introducción a la psicología del aprendizaje
  Gordon R. Cross
  Narcea S. A. De Ediciones Madrid
  España 1984
- Nuevas tecnologías para el aprendizaje
  Carmen Vizcarro y José A. León
  Ediciones Pirámide
  España 1998
- Psicología del aprendizaje y de la activación del aprendizaje
  R. A. Champion
  Editorial Limusa,
  México, 1972

Capítulo 2

- http://www.ggili.com/nov_73950.htm
- http://www.juanval.net/fundamentos_tipometr%C3%ADa.htm
- http://www.psicología/El lenguaje del color (2da parte).htm
- http://dai1.galeon.com/apuntes/analisis/Analisis-t3.doc
- http://www.ldc.usb.ve/~vtheok/cursos/ci3711/apuntes/99-01-14/Info/Modelo%20Espiral.htm
- [2] http://www.monografias.com/trabajos6/audi/audi.shtml
- [2.1] Ingeniería de Software. Un enfoque práctico
  Pressman, Roger S
  Mc.Graw Hill
  5ta. Edición, 2002, México
- Software Engineering.
  Bauer, Friendrich L
  North Holland Publishing Co.,
  1993, Amsterdam

Capítulo 3

- [3.1] Cruz, Alejandro; Evolución de las Culturas Mesoamericanas, 2000, México, Pág. 67
- [3.2] Pressman, Roger S; Ingeniería de Software. Un enfoque práctico; McGrawHill, 5ta. Edición 2002, México, Pág. 332.
- http://www.irc-euro.org/pc/programacion/asp.htm
- http://www.desarrolloweb.com/articulos/758.php?manual=27
- http://www.efaber.net/formacion/fp/curso_php/intro.html
- http://www.htmlpoint.com/php/guida/php_03.htm
- http://www.desarrolloweb.com/articulos/936.php?manual=21

Capítulo 4

- http://www.desarrolloweb.com/articulos/758.php?manual=27
- http://www.gamarod.com.ar/articulos/tutorial_de_cgi.asp
- http://www.iec.csic.es/criptonomicon/java/quesjava.html
- http://paidoteca.dgsca.unam.mx/neopaidoteca/cursos/becas-java/Java2b.html
- http://www.people.virginia.edu/~am2zb/cursos/java/aplicaciones/intro.htm
- http://php.ciberaula.com/articulo/PHPoASP/
- http://www.ignside.net/man/php/functions.php
- http://ascii.eii.us.es/cursos/php/php4.html
- http://www.aulambra.com/ver.asp?id=20
- http://www.deltaasesores.com/esta/EST183.html
- http://www.hackingballz.com/php_manual/faq.languages.html
- http://www.webtaller.com/construccion/lenguajes/asp/programas/index.php
- http://www.desarrolloweb.com/articulos/840.php?manual=15
- http://www.microsoft.com/latam/technet/articulos/200202/art15/
- http://www.monografias.com/trabajos14/sqlserver/sqlserver.shtml
- http://www.elguruprogramador.com.ar/foros/mensaje.asp?id=9330
- http://www.sqlmax.com/dtc.asp
- http://www.microsoft.com/spanish/msdn/articulos/archivo/010201/voices/dts_overview.asp
- http://www.desarrolloweb.com/articulos/1695.php
- http://www.htmlarea.com/directory/Detailed/137.html

Apéndice

- http://www.telefonica.es/fat/lexf.html
- http://members.es.tripod.de/ncabanes/ndic.htm#D
- http://www.geocities.com/curso_virtual_2000/Glosarios.htm
- http://www.eu.microsoft.com/hwdev/GLOSSARY1.HTM#I
- http://www.webopedia.com/
- http://www.educarchile.cl/eduteca/nb3/sitio/glosario.asp.htm
- http://www.tsedi.com/siguienteestilo/estildistan1-3A.html
- http://espanol.geocities.com/andy_n_ve/pala.html
- http://usuarios.lycos.es/hv1102/glosario.html
- http://www.servitel.es/atv/AYU/INTERNET/DICCIO/diccio.htm
- http://controlinterno.udea.edu.co/ciup/glosario.htm
- http://www.xtec.es/~fchorda/civiweb/varia/glosari.htm
- http://www.videoedicion.org/manuales/dsi.htm
- http://java.sun.com
- http://www.red.es/glosario/glosariop.html
- http://www.mekate.com/glosario-s.html
- http://usuarios.lycos.es/Resve/diccioninform.htm
- http://es.wikipedia.org/wiki/URL
- http://buscon.rae.es/diccionario/drae.htm
- http://www.desarrolloweb.com/articulos/840.php?manual=15
- Biblioteca de Consulta Microsoft® Encarta® 2005. © 1993-2004 Microsoft Corporation. Reservados todos los derechos.
- Hernández Santos, Enrique. (1994). Diccionario Fundamental de la Educación. San Juan: Publicaciones Puertorriqueñas.