UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

DISEÑO E IMPLEMENTACION DE UN SISTEMA QUE OPERE VIA WEB PARA LA ORGANIZACION, MANIPULACION Y CONSULTA DE LA INFORMACION ACADEMICO-ADMINISTRATIVA EN EL DEPARTAMENTO DE COMPUTACION DE LA FACULTAD DE INGENIERIA.

T E S I S

QUE PARA OBTENER EL GRADO DE:

INGENIERO EN COMPUTACION

P R E S E N T A

RAYMUNDO GONZALEZ TELLEZ

DIRECTOR DE TESIS: M. en I. JORGE VALERIANO ASSEM

MEXICO, D.F. ABRIL DE 2004

Dedicatoria

A mi Madre:

Agradezco a Dios el darme un tesoro como lo eres tú. Me has dado mucho de lo que hoy soy. Gracias por todo tu sacrificio incondicional, confianza y ejemplo. Te quiero mucho.

A mis Hermanos:

Por su grandioso apoyo que me brindaron y para quienes pienso he sido un ejemplo.

A mi Padre:

Ejemplo de mi niñez. Tu ausencia me dió la fuerza que hoy vive dentro de mí para seguir adelante y enfrentarme a la vida con más ganas.

A mis Tíos:

Por creer en mi y por su valioso apoyo en los momentos mas difíciles. Mi más sincero respeto y admiración.

A mis amigos:

Por convivir parte de su tiempo conmigo y para quienes les agradezco sus consejos y ayuda en momento de desesperación. Espero que nuestra amistad nunca se termine.

A la Universidad:

Alma Matter, forjadora de hombres y mujeres conscientes por el compromiso con la sociedad. Me siento orgulloso de ser puma.

A la Facultad de Ingeniería:

Lugar donde se forjan los grandes Ingenieros del país. Agradezco el espacio que me fué brindado para cumplir una meta más en mi vida y por el apoyo, conocimiento e inspiración obtenida a lo largo de mi carrera.

Índice

INTRODUCCIÓN

Toda aplicación en la computación trata de reflejar parte del funcionamiento del mundo real para automatizar tareas que de otro modo serían llevadas a cabo de forma más ineficiente o quizás no podrían realizarse.

Hoy en día, la excesiva desorganización en el almacenamiento y recuperación de la información en forma impresa o incluso en forma informática de una institución resulta laborioso en el momento de requerir de búsquedas inteligentes para la toma de decisiones, perdiendo de esta manera mucho tiempo valioso y obteniendo cada vez más un excesivo almacenamiento de información en forma impresa. Lo anterior, aunado a hechos que tienen que ver con la entrega inmediata de los documentos a las personas interesadas, hace más difícil que los documentos sean revisados a tiempo. Estos problemas se han ido creando en el Departamento de Ingeniería en Computación de la Facultad de Ingeniería el cual requiere de una solución inmediata de acuerdo a sus necesidades.

Por lo anterior, la tesis que se plantea consiste en la creación de un sistema capaz de organizar adecuadamente la información generada y recibida por el departamento de computación que permita actualizaciones y consultas eficientes para una toma de decisiones rápidamente, contando además con servicios, como Chat y correo electrónico, que permitan establecer contacto entre los diferentes académicos del departamento y así tener una mejor comunicación. Para ello, el sistema funcionará en un ambiente de trabajo cliente – servidor (arquitectura de tres niveles, ver Fig. I.1), utilizando herramientas de la compañía Microsoft disponibles en el Laboratorio Microsoft de la Facultad de Ingeniería, el cual fue donado por dicha compañía.

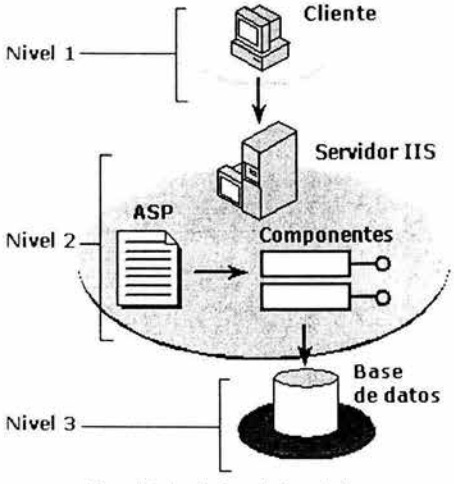


Figura I.1 Arquitectura de tres niveles

Para la creación del sistema se aplicarán diversas técnicas y tecnologías de la computación para el desarrollo de aplicaciones de Internet. Entre ellas, se pueden mencionar:

- UML y su extensión para aplicaciones Web (WAE) como lenguaje de modelado orientado a objetos utilizando como herramienta Microsoft Visio Profesional 2002.
- Microsoft SQL Server 2000 como la base de datos del sistema, en la cual residirá toda la información generada por el Departamento de Ingeniería en Computación.
- IIS 6.0 como el servidor de aplicaciones WWW, SMTP y FTP.
- ASP .NET como el lenguaje de programación basado en VB, utilizando como herramientas de desarrollo Visual Studio .NET y Macromedia Dreamweaver MX.

- Microsoft Windows 2003 Server como el sistema operativo sobre el cual se instalaron todas las aplicaciones necesarias para el desarrollo del sistema.

La base de datos contendrá documentos de distintas formas, pudiendo ser una imagen, un archivo en formato .doc o .pdf. Cada documento constituye una unidad de información que se recupera por separado, la cual tendrá que ser una búsqueda relevante a las necesidades de cada usuario de entre la gran cantidad de documentos disponibles. La eficiencia para la recuperación de información no solo dependerá de los mecanismos de acceso que se proporcione, sino que también dependerá en gran medida de la habilidad del usuario para encontrar la información.

Por otra parte la seguridad en el manejo del sistema en Internet es muy importante. Se tendrán diversos niveles de seguridad para el acceso de los usuarios al sistema (administrador, secretarias, académicos) que brinden un manejo de la información en forma segura, contemplando además la seguridad en etapas críticas del sistema como encriptación del login del usuario. De esta forma el usuario podrá acceder sin preocuparse por la integridad de su sesión.

La visión con que es desarrollado el sistema ha sido pensada en gran parte en las futuras modificaciones que se le puedan hacer al sistema. Por tal motivo, la documentación del sistema es de gran importancia, concentrándose en ésta las partes fundamentales del desarrollo y funcionamiento del sistema que hagan que el administrador del sistema sea capaz de entender en su totalidad el sistema y pueda llevar a cabo las modificaciones que crea conveniente y que requiera el departamento de computación para su buen funcionamiento.

La tesis está dividida en cuatro capítulos:

En el capítulo I "Antecedentes" se menciona la situación actual del Departamento de Ingeniería en Computación de la Facultad de Ingeniería, sus funciones principales, organización, requerimientos informáticos, así como una propuesta de solución del problema.

En el capítulo II "Marco Teórico" se describen los conceptos teóricos base que son necesarios para el desarrollo del sistema. Se explica en forma general lo referente a lenguajes de programación, bases de datos, redes de computadoras, modelos de desarrollo de software y metodologías de Ingeniería de Software.

El capítulo III "Análisis y Diseño del Sistema" es la parte medular del sistema, en donde se llevan a cabo el análisis y diseño tanto de la interfaz del sistema como de la base de datos aplicando para ello UML como lenguaje de modelado. Aquí se desarrollan diversos diagramas UML que nos permiten definir de manera concreta y precisa el funcionamiento que tendrá el sistema tales como diagramas de uso, diagramas de secuencias, diagramas de componentes y diagramas de clases. Por último se muestra el diagrama entidad – relación de la base de datos, base fundamental para la implementación de la base de datos.

El capítulo IV "Implementación y Pruebas del Sistema" trata de la construcción de la interfaz de usuario y de la base de datos, así como de la explicación de las herramientas de desarrollo utilizadas para su implementación. Para terminar este capítulo, se muestra la seguridad con que cuenta este sistema y también se llevan a cabo diversas pruebas en él para mostrar y comprobar su buen funcionamiento.

El capítulo V "Manteniminto del Sistema" explica de manera general la forma en que podrá ocurrir algún tipo de mantenimiento del sistema, así como su solución.

Posterior a estos capítulos se lleva a cabo una conclusión, mencionando los objetivos alcanzados y propuestas para el mejoramiento de este sistema.

Finalmente, esta tesis cuenta con un apartado para anexos, los cuales son los siguientes:

Anexo I "Manual de usuario". Se describe la forma de operar el sistema por parte de los diferentes tipos de usuarios que tienen acceso a dicho sistema.

Anexo II "Manual técnico". Se describe la integración del sistema para aquellos usuarios que cuenten con un mayor conocimiento de las tecnologías empleadas en el desarrollo del sistema. En este anexo se explica la forma en el almacenamiento de la base de datos, su configuración así como el catálogo full-text asociado a esta base. Además se muestra la configuración del servidor Web.

CAPÍTULO

I

ANTECEDENTES

CONTENIDO

1.1 La Facultad de Ingeniería

La Facultad de Ingeniería de la Universidad Nacional Autónoma de México es una institución educativa fundamental en la formación de los mejores recursos humanos del país. Por ello, responde al compromiso de ser una entidad creativa y efectiva para contribuir al desarrollo constante y amplio del país.

La Facultad de Ingeniería como parte de la Universidad realiza con transparencia sus tareas enfocadas principalmente, a la educación y formación de futuros ingenieros, así como el ofrecimiento de programas de estudio innovadores y actualizados que sean competitivos con los que se ofrecen en el país y en el extranjero, que ayuden a lograr el mejor aprovechamiento de los recursos y a resolver, en la forma más justa, los problemas de la nación.

Sus alumnos están comprometidos a realizar un mayor esfuerzo por aprovechar todo su potencial y capacidad para llegar a ser profesionales idóneos, pues es éste el compromiso que han adquirido con la sociedad al ingresar a la Facultad.

La Facultad de Ingeniería tiene como objetivos primordiales la formación de profesionales de alto nivel académico, la investigación, la docencia y la difusión de la cultura.

La planta académica de la Facultad de Ingeniería está actualmente integrada por 1,108 profesores de asignatura, 240 profesores de carrera, 106 técnicos académicos y 397 ayudantes de profesor. Cerca de la mitad de los profesores cuenta con estudios de especialización o de posgrado. La mayor parte de los profesores de la Facultad combina sus labores docentes con actividades profesionales propias de su especialidad. Por otra parte, es interesante mencionar que 42 profesores de la Facultad pertenecen al Sistema Nacional de Investigadores.

La misión de la Facultad de Ingeniería

" Formar integralmente recursos humanos en los niveles de licenciatura, especialidad y posgrado, para que sean competitivos en el ámbito nacional e internacional como ingenieros de la más alta calidad; con habilidades y actitudes que les permitan el mejor desempeño en el ejercicio profesional, la investigación y la docencia; con capacidad para aprender durante toda la vida y mantenerse actualizados en los conocimientos de vanguardia; con una formación humanista que sustente sus actos y sus compromisos con la Universidad y con México, para que coadyuven al mejoramiento social, económico, político y cultural de la nación. "

Calidad

La calidad es la resultante de la óptima operación de todos los recursos de la Facultad de Ingeniería: personal académico comprometido y actualizado, estudiantes responsables, planes y programas de estudio flexibles y puestos al día, procesos educativos modernos, sistemas de apoyo académico funcionales y actualizados, sistemas de administración modernos y eficientes, cooperación académica continua con otras instituciones y entidades educativas, investigación de alto nivel que sirva como apoyo a la docencia, vinculación con la sociedad y los sectores productivos para generar recursos y soluciones a problemas sociales.

Compromiso con la sociedad mexicana

La Facultad de Ingeniería tiene el compromiso social de formar recursos humanos que realicen sus actividades profesionales con un espíritu ético, crítico, creativo, solidario, eficiente, eficaz, honesto y responsable, para que contribuyan al desarrollo de la nación mexicana y hacerla competitiva en el ámbito internacional, como en la construcción de un país con mayor calidad de vida, justo y digno; dando solución a los principales problemas que lo aquejan. Asimismo, la comunidad de la Facultad

está obligada a utilizar adecuada y racionalmente los recursos que le han sido asignados por la misma sociedad.

Estructura básica

En la Figura I.1 se observa la estructura básica de la Facultad de Ingeniería.

Figura I.1 Estructura básica de la Facultad de Ingeniería

1.2 La División de Ingeniería Eléctrica

La División de Ingeniería Eléctrica (DIE) de la Facultad de Ingeniería imparte y coordina académica y administrativamente las carreras de Ingeniería Eléctrica - Electrónica, Ingeniería en Computación e Ingeniería en Telecomunicaciones.

La Figura I.2 muestra un esquema general de la DIE.

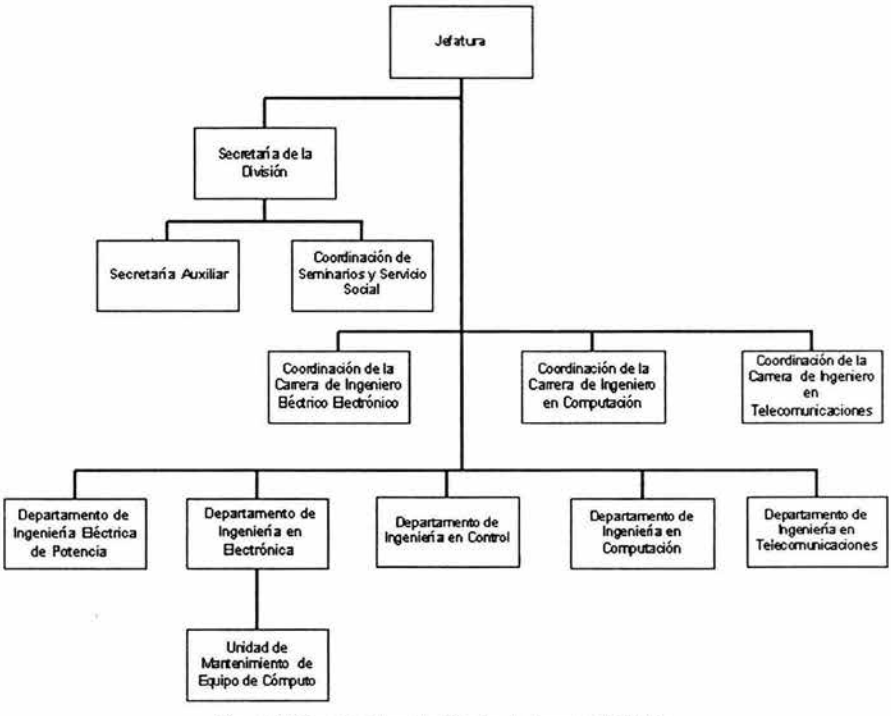


Figura I.2 Organización de la División de Ingeniería Eléctrica

1.3 El Departamento de Ingeniería en Computación

El Departamento de Ingeniería en Computación forma parte de la División de Ingeniería Eléctrica de la Facultad de Ingeniería, UNAM. Tiene como finalidad coordinar y administrar la carrera de Ingeniería en Computación, así como proponer las modificaciones pertinentes al plan de estudio de la carrera de Ingeniería en Computación, fomentar las relaciones de intercambio con dependencias universitarias tanto nacionales como extranjeras entre otras actividades.

1.3.1 Funciones

El departamento de ingeniería en computación de la facultad de ingeniería tiene a su cargo un gran número de funciones de carácter institucional de suma importancia entre las cuales se pueden mencionar las siguientes:

- Coordinar académica y administrativamente la carrera de Ingeniería en Computación.
- Impartir y actualizar los planes y programas de estudio de la carrera y, asimismo proponer las modificaciones pertinentes, si fuera necesario crear nuevas asignaturas y áreas afines.

- Desarrollar actividades de superación y actualización de su personal académico para mejorar el proceso enseñanza-aprendizaje.
- Mantener y fomentar las relaciones de intercambio con dependencias universitarias, instituciones de educación superior, asociaciones y colegios profesionales y de instituciones afines, tanto nacionales como extranjeras.
- Promover la realización de conferencias seminarios, exposiciones, cursos y demás actividades tendientes a la difusión científica y técnica en las disciplinas a su cargo.
- Realizar asesorías y actividades de investigación tecnológica en las líneas de investigación a cargo.
- Programar horarios.
- Supervisar y coordinar a los profesores del departamento así como el funcionamiento óptimo de los laboratorios y sus equipos.
- Asignar presupuesto según necesidades.

Para poder cumplir con los objetivos del Departamento de Ingeniería en Computación, la parte administrativa es una parte esencial para la creación y distribución de los diferentes documentos que son base para la toma de decisiones.

1.3.2 Organización

En la Figura I.3 se muestra la forma en que está organizado el Departamento de Ingeniería en Computación de la Facultad de Ingeniería.

El Departamento cuenta actualmente con la siguiente planta académica: aproximadamente 20 profesores de carrera y técnicos de tiempo completo.

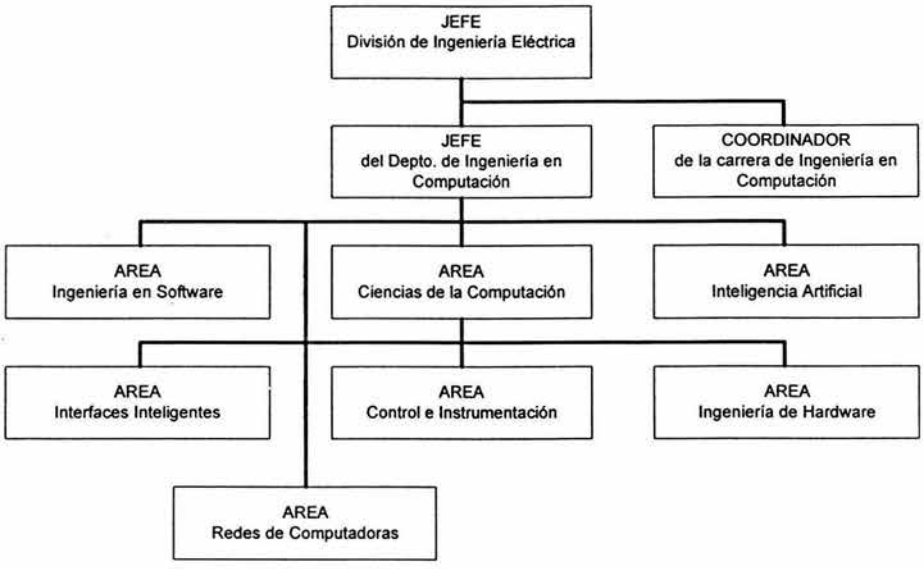


Figura I.3 Organización del Departamento de Ingeniería en Computación

1.3.3 Funciones del Jefe del Departamento de Ingeniería en Computación

- Coordinar las tareas del Personal Académico y administrativo adscrito al departamento.
- Definir, de común acuerdo con el Personal Académico adscrito al departamento, las actividades académicas del mismo.
- Participar en la elaboración de los planes de desarrollo del departamento a su cargo.
- Solicitar al personal del departamento, informes sobre el trabajo realizado.
- Elaborar los informes anuales de actividades del departamento y participar en la evaluación correspondiente, informando al personal del resultado de este último ejercicio.
- Opinar ante el Consejo Interno sobre las solicitudes de contratación, renovación de contrato y de promoción o definitividad del Personal Académico del departamento con base en sus informes de trabajo.
- Preparar el anteproyecto de presupuesto departamental correspondiente y proponerlo al Jefe de la DIE con base en los requerimientos solicitados por su personal.
- Promover la elevación del nivel académico del personal de su departamento y la colaboración intra e inter departamental y con otras dependencias.
- Convocar al Personal Académico adscrito al departamento a reuniones de trabajo con la frecuencia que asegure la buena marcha académica del mismo.

- Coordinar la distribución de recursos para el trabajo académico, mediante acuerdo con el Jefe de la DIE.
- Proponer al Consejo Interno la creación de nuevas líneas de investigación, la formación de laboratorios u otros cambios necesarios.
- Servir de enlace entre el Jefe de la DIE y el Personal Académico, sin perjuicio del derecho que este personal tiene para acudir directamente al Director.
- Elaborar los proyectos, comunicaciones o informes que le encomiende el Director sobre su Departamento.
- Autorizar las salidas de campo del personal a su cargo y del financiamiento necesario para tales fines. Las salidas de los Jefes de Departamento serán autorizadas por la dirección o la Secretaría Académica.
- Revisar, comentar y en su caso, dar visto bueno a las propuestas de seminario y/o tesis.

1.3.4 Funciones del Coordinador de la Carrera de Ingeniería en Computación

Las funciones principales del coordinador de la carrera de Ingeniería en Computación son las siguientes:

- Revisar y mantener actualizado el plan y los programas de estudio de la carrera de Ingeniero en Computación, incluyendo contenido, secuencia, métodos de enseñanza aprendizaje, créditos, bibliografías, asignaturas, prácticas, laboratorios, etc.
- Establecer los objetivos, metas y estrategias generales de la coordinación a corto y mediano plazo, con base en las políticas y lineamientos fijados por la dirección de la Facultad y las opiniones del Consejo Asesor Externo de la División de Ingeniería Eléctrica, así como la de profesores de la Facultad y gente externa relacionada con el área de la computación.
- Coordinar e impartir la orientación vocacional necesaria a los alumnos de bachillerato.
- Cuantificar las necesidades de recursos humanos, materiales y acciones para su eventual implantación de cada una de las modificaciones al plan y programas de estudio de la Carrera de Ingeniero en Computación.
- Interactuar con el Consejo Asesor Externo de la División, con Jefes de Departamento, profesores y gente externa relacionada con el área de la computación para la evaluación y en su caso, actualización del plan y programa de estudio de la carrera de Ingeniero en Computación.
- Revisar, solicitar y ser conducto, en su caso, para llevar al Consejo Asesor Externo y a los Departamentos correspondientes cualquier propuesta de modificación de los planes y programas de estudio vigentes.
- Supervisar que se cumpla el plan de estudios vigente.

- Realizar estudios referentes a la profesión (mercado de trabajo, demanda, tendencias, etc.)
- Efectuar el seguimiento del plan de estudios y del alumnado tomando en cuenta, entre otros, los siguientes factores: inscripciones, aprovechamiento de los alumnos, titulaciones, deserciones, etc.
- Opinar sobre casos individuales de cambio de carrera y otros problemas especiales de los alumnos.
- Proponer las acciones correctivas, en su caso, para asegurar el logro de las metas académicas establecidas.
- Solicitar los recursos que se requieran para el desarrollo de las actividades de la coordinación.
- Revisar, comentar y en su caso, dar visto bueno a las propuestas de seminario y/o tesis.

El coordinador le reporta al Jefe del Departamento de Ingeniería en Computación y al Jefe de la División de Ingeniería Eléctrica.

Por otra parte, las personas que le reportan al Coordinar son aquellos que están involucrados en el proceso que se esté llevando a cabo, como pueden ser: profesores, ayudantes, gente de servicio social, personal administrativo, etc.

1.3.5 Descripción actual del tratamiento de la información en el Departamento de Ingeniería en Computación

Debido a la complejidad de actividades que se realizan en el Departamento de Ingeniería en Computación, se maneja actualmente una gran cantidad de documentos la cual se lleva a cabo de forma impresa realizándose de la siguiente manera:

Cualquier documento que llega al Departamento de Computación (oficios, memorandums, minutas, convocatorias, boletines, informes) queda bajo resguardo de las secretarias del departamento, las cuales se encargan de proporcionar la información impresa al personal académico y de archivar el documento para una posterior consulta del mismo.

Los documentos son proporcionadas a los académicos por dos vías: de forma personal o enviado a su respectivo buzón ubicado en el segundo piso del edificio Valdez Vallejo, aun lado de la fotocopiadora, Anexo de la Facultad de Ingeniería. Esto trae consigo algunos problemas como son:

- Que el académico no sea localizado para la entrega oportuna del documento impreso.
- Que el académico se olvide por consultar los documentos impresos existentes en su buzón o espere para otra ocasión debido a la lejanía con la que se encuentra ubicado su buzón.
- Pérdida del documento.
- Acumulación excesiva de documentos impresos.
- Búsqueda ineficiente de los documentos impresos.

La forma en la que se lleva este manejo de los documentos resulta hoy en día ineficiente, pudiéndose automatizar muchas de las funciones que se llevan a cabo mediante herramientas de programación que actualmente existen.

La Figura I.4 muestra en forma general las situaciones anteriormente descritas.

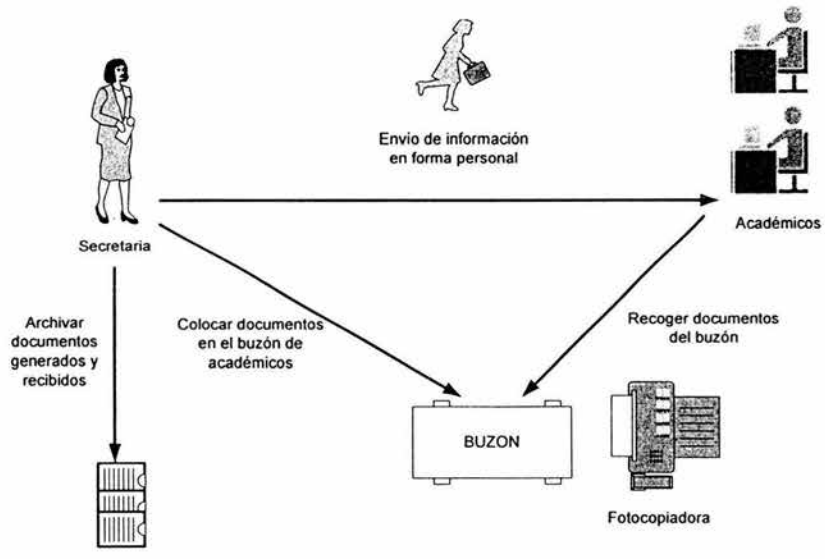


Figura I.4 Situación actual del tratamiento de la información en el Departamento de Ingeniería en Computación

1.3.6 Necesidad de un sistema para la organización, manipulación y consulta de la información en el Departamento de Ingeniería en Computación.

El gran avance tecnológico que existe no puede dejar atrás al Departamento sin un sistema que automatice algunas de las tareas del Departamento de Ingeniería en Computación, el cual brindará muchos beneficios.

La necesidad de un sistema es necesario porque:

- Brindará una mejor atención a la parte académico – administrativa del Departamento.
- Llevará un control sobre los documentos de interés para el Departamento.
- Brindará una consulta eficiente de los documentos, sin necesidad de asistir al archivo físico del Departamento de Ingeniería en Computación.
- Eliminará la necesidad de almacenar documentos impresos.
- Proporcionará una mejor comunicación entre los integrantes del departamento.
- Mantendrá un histórico permanente de los documentos.
- Almacenamiento eficiente de los documentos en forma electrónica
- Control de acceso de los usuarios que pueden contribuir y acceder a los documentos.
- La toma de decisiones en el departamento se llevará de una forma más rápida.

La Figura I.5 intenta mostrar el panorama que se tendrá en el Departamento con la automatización de algunas actividades que se llevan a cabo diariamente.

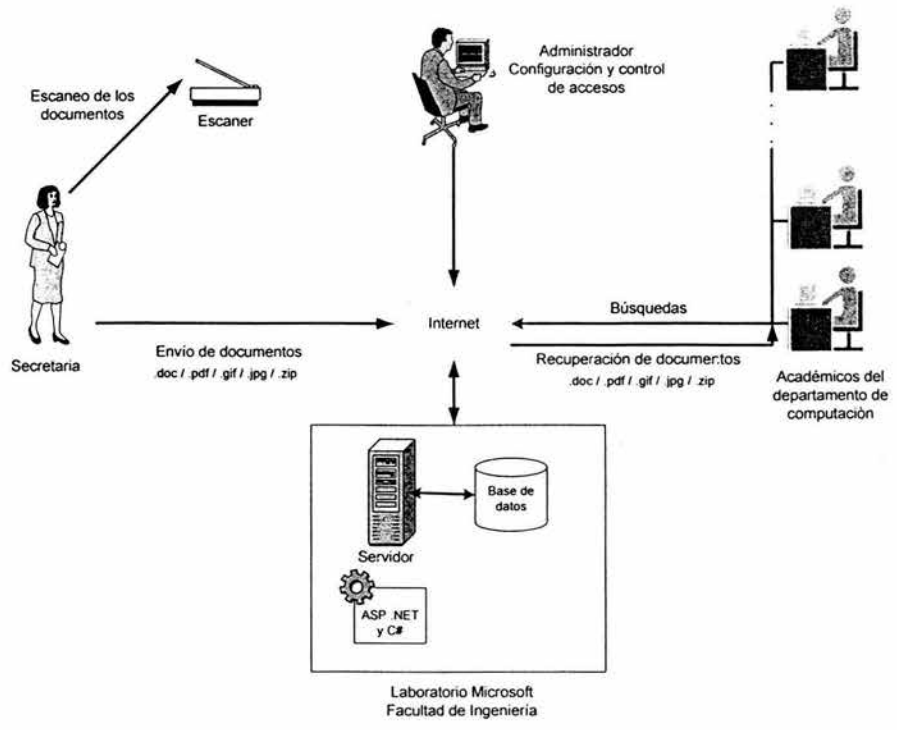


Figura I.5 Situación futura para el tratamiento de la información en el Departamento de Ingeniería en Computación

CAPÍTULO

II

MARCO TEÓRICO

CONTENIDO

2.1 Lenguajes de programación

Los lenguajes de programación han ido evolucionando en los últimos 50 años. A continuación se presentarán los principales hechos:

Década de los cuarenta: era prelingual
- La programación incluye el manejo de botones de consola, interruptores y enchufes.
- Konrad Zuse, 1944, Plankalkül (programa de cálculo). lenguajes de programación con variables, valores estructurados y procedimientos con parámetros. Más preocupado de la estructura que de la eficiencia.

Década de los cincuenta: aprovechamiento de la potencia de la máquina
- Programación en ensamblador.
- Aplicación principal: cálculo numérico. Mucho trabajo consiste en convertir fórmulas numéricas en instrucciones del ensamblador.
- Nacen los primeros lenguajes de programación de alto nivel: FORTRAN (IBM Mathematical FORmula TRANslating System). También incluye procedimientos y algunas estructuras de control de flujo.

Década de los sesenta: atención a la potencia de expresión de los lenguajes de programación
- Aparecen muchos lenguajes de programación: COBOL, LISP, ALGOL 60, PL/I.
- Proponen nuevas ideas: datos estructurados (COBOL, PL/I), recursión (LISP, ALGOL 60), interacción con el usuario (BASIC) y computación simbólica (LISP -> primer lenguaje funcional).

Década de los setenta: portabilidad (reducción de la dependencia de la máquina)
- Crisis del software y corrección de los programas.
- Programación estructurada: conjunto de directivas para mejorar la estructura de los programas.
- Objetivos interdependientes y un programa bien estructurado es más independiente de la máquina (las dependencias se encuentran más localizadas).
- Lenguajes: Pascal, ALGOL 68 y C. Considerable portabilidad y más abstracción del proceso de computación.
- Factores que permiten conseguir la portabilidad: decremento en la variedad del hardware y los SO, mejora en la tecnología de los compiladores y mejor estilo de programación (resultado de la programación estructurada).

Década de los ochenta: reducción de la complejidad en la programación y la administración de programas
- El crecimiento de los programas los convierte en "inmanejables".
- Para ayudar a manejar la complejidad aparecen: Ada y Modula-2.
- Surgen "otras formas de pensar" en la programación: lenguajes orientados a objetos (Smalltalk, C++), funcionales (SML, Miranda) y lógicos (PROLOG).

Década de los noventa: explosión del hardware paralelo y distribuido
- Tratamiento sistemático del hardware por parte de los lenguajes de programación tal como Java.
- Estos lenguajes permiten solucionar un problema de manera eficiente dividiéndolo en subproblemas que pueden ejecutarse en paralelo por procesadores diferentes.

2.1.1 Tipos y características fundamentales

En la Figura II.1 y II.2 se presentan los lenguajes que por su uso y comercialización, han resultado ser los más populares a lo largo de este medio siglo.

La existencia de tantos lenguajes obedece a que cada uno de ellos está encaminado a resolver ciertas tareas, dentro de la amplia problemática de la explotación de la información, o bien, a que su arquitectura, o su forma de llevar a cabo la programación, tiene un enfoque particular.

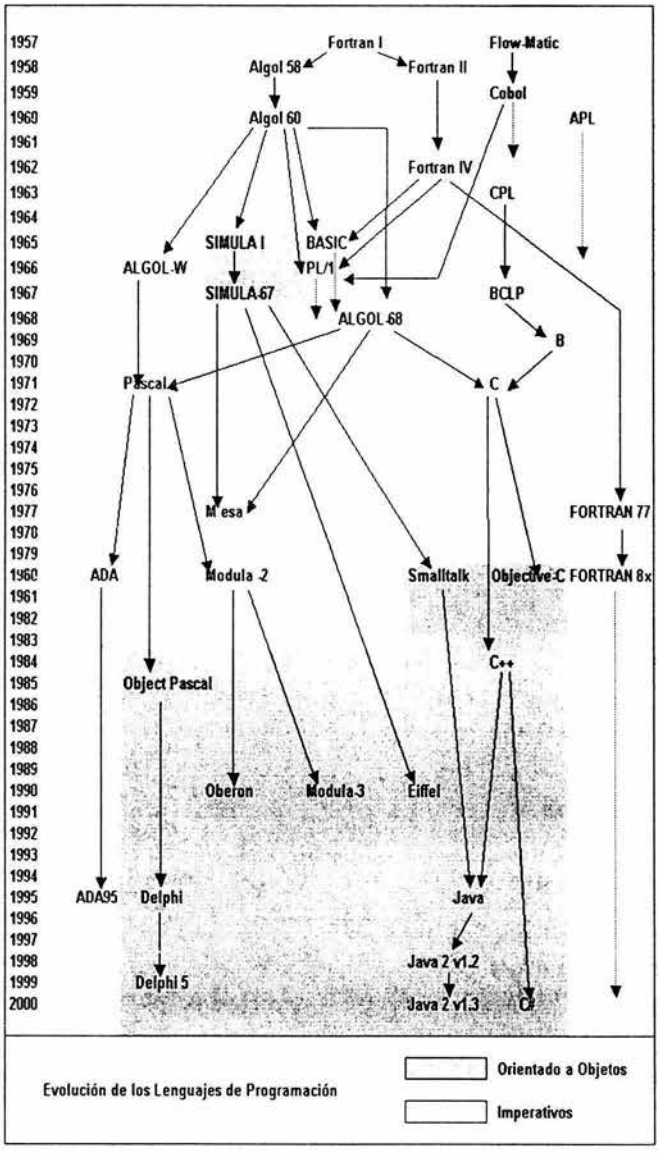


Figura II.1 Evolución de los Lenguajes Imperativos y Orientados a Objetos

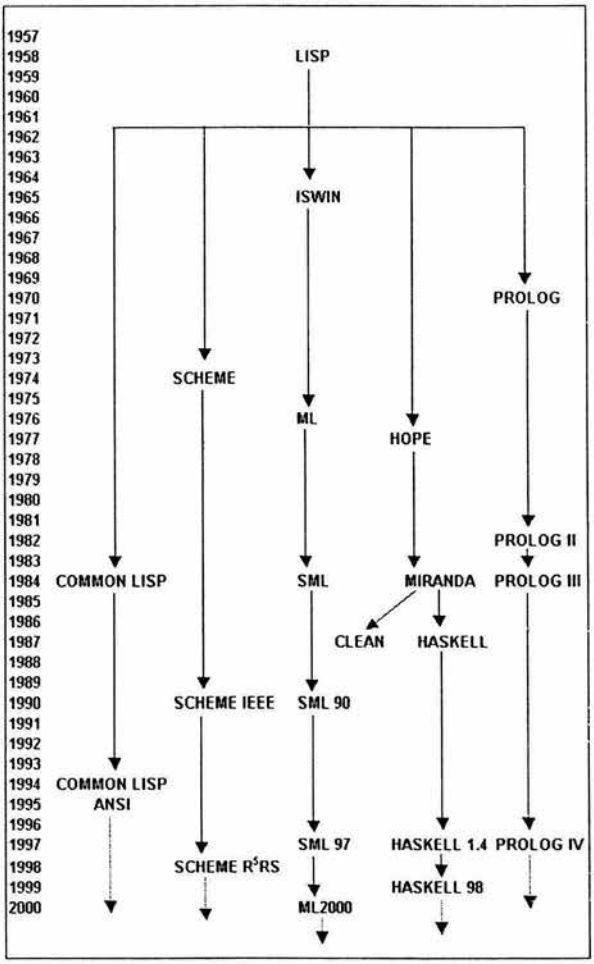


Figura II.2 Evolución de los lenguajes declarativos

De acuerdo con el estilo de programación, podemos clasificar los lenguajes en las siguientes categorías:

- **Imperativos**: Son aquellos lenguajes, que basan su funcionamiento en un conjunto de instrucciones secuenciales, las cuales, al ejecutarse, van alterando las regiones de memoria donde residen todos los valores de las variables involucradas en el problema que se plantea resolver. Es decir, se cambia progresivamente el estado del sistema, hasta alcanzar la solución del problema.

- **Declarativos**: En este paradigma, más que el ¿cómo? desarrollar paso a paso un proceso, nos interesa el ¿qué? deseamos obtener a través del programa. Quizás el lenguaje declarativo que nos sea más familiar, es SQL, el cual es utilizado para interactuar con la

información de bases de datos, concentrándose sólo en los resultados que van a ser obtenidos, dejándole al traductor la tarea de cómo llegar a ellos y presentárnoslos.

Dentro de este paradigma, se encuentran dos estilos distintos de programación, cada uno de los cuales posee su propia lógica.

- **Funcionales**: Son lenguajes basados en funciones, las cuales se representan mediante expresiones, que nos permiten obtener ciertos resultados a partir de una serie de argumentos. De hecho las expresiones están formadas por un conjunto de términos, que a su vez pueden encapsular otras expresiones, para con la evaluación de todas ellas, llegar a la solución deseada.

- **Lógicos**: Este tipo de lenguajes se basan en el cálculo de predicados, la cual es una teoría matemática que permite entre otras cosas, lograr que un ordenador basándose en un conjunto de hechos y de reglas lógicas, pueda derivar en soluciones inteligentes.

- **Orientados a Objetos**: Este último paradigma, como se puede observar en la Figura II.1, algunas veces se mezcla con alguno de los otros 2 modelos, sin embargo mantiene características propias, que lo diferencian claramente. Los programas de este tipo, se concentran en los objetos que van a manipular, y no en la lógica requerida para manipularlos. Ejemplos de objetos pueden ser: estudiantes, coches, casas etc. cada uno de los cuales tendrá ciertas funciones (métodos) y ciertos valores que los identifican, teniendo además, la facultad de comunicarse entre ellos a través del paso de mensajes. Cabe mencionar con más detalle los elementos fundamentales que deben de poseer este tipo de lenguajes:

  - **Abstracción**: Determinación de las características de los objetos, que sirven para identificarlos y hacerlos diferentes a los demás.

  - **Encapsulamiento**: Es el proceso que agrupa y almacena los elementos que definen la estructura y el comportamiento de una abstracción, en un mismo lugar.

  - **Modularidad**: Es la propiedad de agrupar las abstracciones que guardan cierta relación lógica, y a la vez minimizar la interdependencia entre las diversas agrupaciones.

  - **Jerarquía**: Consiste en establecer un orden o una clasificación de las abstracciones.

  - **Herencia:** Es un mecanismo que permite la definición de una clase a partir de la definición de otra ya existente.

  - **Polimorfismo:** Consiste en la referencia a objetos de diferentes clases. Para que esto sea posible, debe de haber una relación de herencia entre esas clases.

Ahora bien, si tomamos como referencia las herramientas usadas en el proceso de traducción y ejecución de los programas vamos a tener la siguiente clasificación de lenguajes:

- **Lenguajes Ensamblados**: Se refieren al lenguaje ensamblador, que viene a ser una representación simbólica de las instrucciones correspondientes al lenguaje ensamblador de alguna arquitectura específica, con lo que, casi siempre, la correspondencia entre las instrucciones de este lenguaje, y las del lenguaje máquina son de 1 a 1, si bien existen algunas excepciones, que dan lugar a lo que se conoce como lenguajes macro-ensambladores.

- **Lenguajes Compilados**: Son aquellos, que son traducidos de un lenguaje de alto nivel (como FORTRAN o PASCAL) a lenguaje máquina o bien a lenguaje ensamblador, produciendo un programa objeto permanente.

- **Lenguajes Interpretados**: Estos lenguajes, tienen la particularidad, de que no producen código objeto, sino que cada instrucción es analizada y ejecutada a la vez, lo que ofrece mucha interacción con los usuarios, pero a la vez resultan ineficientes, cuando se desea ejecutar repetitivamente un programa.

- **Lenguajes Preprocesados**: Son lenguajes que son traducidos primeramente a un lenguaje intermedio de más bajo nivel, para posteriormente volverlos a traducir y producir el programa objeto. Este tipo de lenguajes fueron creados, con la idea de proporcionar un lenguaje más potente que el lenguaje intermedio, mediante la implementación de algunas macroinstrucciones.

Finalmente, existen otros conceptos tomados en cuenta para agrupar los lenguajes, que dan origen a diversas clasificaciones, entre los que destacan las siguientes:

- **Lenguajes de cuarta generación 4GL**: Estos lenguajes se distinguen por formar parte de un entorno de desarrollo, que comprende el manejador de una base de datos, y todo lo que de esto se deriva, como la administración de un diccionario de datos, el control de accesos, el manejo de la consistencia de la información y otras características enfocadas a facilitar los programas de acceso y explotación de la información. Como ejemplos podemos citar a los 4 grandes: PROGRESS, SYSBASE, INFORMIX, y ORACLE.

- **Lenguajes Visuales**. Se les llama de esta manera a los lenguajes que forman parte de una aplicación dotada de una Interfase gráfica, la cual por medio de iconos y otras herramientas visuales y simbólicas, pretenden facilitar las tareas rutinarias de los programadores, como son el diseño y desarrollo de formularios e informes. Los ejemplos más comerciales de estos lenguajes son: VISUAL BASIC, VISUAL FOX PRO, etc.

- **Metalenguajes**: Son lenguajes como XML, SGML y HTML que sirven para definir otros lenguajes, cuyo objetivo es llevar a cabo la estructuración de textos mediante un conjunto de etiquetas, de manera tal, que puedan ser entendidos por los humanos y también procesado por los ordenadores. Estos lenguajes están teniendo un gran auge sobre la plataforma de Internet, en la cual son usados para la creación de documentos, y el intercambio o transferencia de información.

- **Lenguajes de propósito específico:** Son aquellos lenguajes desarrollados con la finalidad de resolver problemas de una naturaleza muy determinada, tal como SPSS para problemas estadísticos, MATLAB para cálculos científicos y de ingeniería, CAD/CAM para el diseño de piezas y programación de máquinas de control numérico, como tornos y fresadoras, GPSS para simulación de sistemas, CORBA para el manejo de interfaces en ambientes cliente-servidor, etc.

- **Lenguajes Script:** Son lenguajes como JAVASCRIPT, VBSCRIPT, PERLSCRIPT, que se utilizan en ambientes clientes servidor, mediante la incrustación de código en las páginas HTML, y así permitir la programación del lado del cliente, buscando fundamentalmente, hacer más atractivos los interfaces gráficos de las páginas.

Esta gran cantidad de lenguajes, señala de manera clara que existe un esfuerzo continuo en la creación, y mejora de los lenguajes de programación, en aras, de hacer más fácil la tarea del programador y/o hacer un uso más eficiente de los recursos computacionales.

2.2 Base de datos

Una Base de Datos es un conjunto de datos interrelacionados (tablas) con independencia física y lógica, consistentes, íntegros y con redundancia controlada que sostienen información relacionada, de forma que:

- Los datos son compartidos por diferentes usuarios y programas de aplicación; existe un mecanismo común para inserción, actualización, borrado y consulta de los datos.
- Tanto los usuarios finales como los programas de aplicación no necesitan conocer los detalles de las estructuras de almacenamiento.

Las bases de datos múltiples son comunes. Las bases de datos pueden abarcar varios discos. El Diccionario de Datos se almacena en cada base de datos.

2.2.1 Tipos de bases de datos

Existen diferentes gestores de bases de datos. Estos son:

- *Jerárquicos.-* Utiliza jerarquías o árboles para la representación lógica de los datos. Los archivos son organizados en jerarquías, y normalmente cada uno de ellos se corresponde con una de las entidades de la base de datos.
- *De red.-* Al igual que en la estructura jerárquica, cada nodo pude tener varios hijos pero, a diferencia de ésta, también puede tener varios padres. Esta basado en el concepto de conjunto.
- *Relacionales.-* Este modelo intenta representar la base de datos como un conjunto de tablas. Esta basado en el concepto de relación.
- *Orientados a objetos.-* Se basa en el concepto de encapsulamiento de datos y código que opera sobre estos en un objeto. Los objetos estructurados se agrupan en clases.

Actualmente, las bases de datos relacionales son las mas usadas, aunque las orientadas a objetos cada vez van siendo mas comunes.

2.2.2 DBMS

Entre la base de datos física (es decir, los datos tal y como están almacenados en la realidad) y los usuarios del sistema, existe un nivel de programas, denominado, manejador de bases de datos (MBD) o, en la mayoría de los casos, el sistema administrador de bases de datos DBMS (Data Base Management System). El DBMS maneja todas las solicitudes de acceso a las Bases de Datos formuladas por los usuarios, para la adición y eliminación de archivos (tablas), la obtención y puesta al día de los datos de esos archivos o tablas, etc. Todas estas posibilidades están incluidas en el DBMS. Así, una de las funciones generales del DBMS es distanciar a los usuarios de la base de datos de detalles al nivel del equipo (de manera muy similar a la forma como los sistemas de lenguajes de programación evitan a los programadores de aplicaciones la necesidad de ocuparse de detalles al nivel de la máquina), y hace posible sus operaciones (como por ejemplo las operaciones de SQL).Conceptualmente lo que sucede es lo siguiente:

- Un usuario solicita acceso, empleando algún sublenguaje de datos determinado
- El DBMS interpreta esa solicitud y la analiza
- El DBMS ejecuta las operaciones necesarias sobre la base de datos almacenada

2.2.3 Modelo de datos relacionales

Los principios del modelo relacional fueron planteados por primera vez por el Dr. E. F. Codd en Junio de 1970. Una base de datos relacional usa relaciones o tablas de dos dimensiones para almacenar información.

1. *Tabla:* Dentro del enfoque relacional una tabla es conocida como una Relación. Una relación es una tabla de dos dimensiones con las siguientes propiedades:

   - Cada columna contiene valores relativos al mismo atributo, y cada valor de una columna de la tabla debe ser simple (un solo valor).
   - Cada columna tiene un nombre distinto (nombre del atributo), y el orden de las columnas no es importante.
   - Cada renglón es distinto, esto es, un renglón no puede duplicarse en otro para un grupo de columnas seleccionadas como llave.
   - Cada atributo no llave debe depender sólo de la llave de la relación no de ningún otro no llave.
2. *Tupla:* Conjunto de valores que componen un renglón de la relación. Es equivalente a una instancia de un registro.
3. *Grado de una tupla:* Número de atributos que tiene una tupla (n de una n-tupla)
4. *Cardinalidad:* Número de tuplas de una relación.
5. *Dominio:* Conjunto de todos los valores posibles para un atributo.

Los componentes del modelo relacional son:

- Colecciones de objetos o relaciones que almacenan los datos.
- Conjunto de operadores que pueden actuar en las relaciones para producir otras relaciones.
- Integridad de datos por exactitud y consistencia.

2.2.4 SQL

EL SQL (Structured Query Lenguage) es un lenguaje para consultar y definir datos en un sistema manejador de bases de datos relacional (RDBMS). Entre sus principales características se encuentran las siguientes:

- Es un lenguaje semánticamente fácil de entender, ya que las sentencias (instrucciones) parecen sencillas frases en inglés, lo que lo hace fácil de aprender y utilizar. Estas sentencias hacen énfasis en el "qué" consultar o definir y no en el "cómo" el servidor realiza la consulta.
- En comparación con lenguajes no relacionales de base de datos, SQL permite en sus instrucciones manejar un conjunto de registros en lugar de un registro a la vez.
- No requiere especificar la estrategia de búsqueda de datos; SQL identifica y utiliza el método más eficiente de búsqueda de los datos solicitados.
- Utilizando interactivamente SQL, el usuario puede realizar consultas ad hoc a la base de datos en forma más sencilla que con otros lenguajes de acceso a datos.

Los componentes del SQL son:

- El Lenguaje de Definición de Datos / Data Definition Language (DDL), el cual permite crear, modificar y eliminar estructuras de datos como: tablas, bases de datos, índices, etc.,
  CREATE, ALTER, DROP, RENAME
- El Lenguaje de Manipulación de Datos / Data Manipulation Language (DML), el cual permite insertar, modificar y eliminar datos de las tablas de la base de datos.
  SELECT, INSERT, UPDATE, DELETE
- El Lenguaje de Control de Datos / Data Control Language (DCL), el cual permite establecer los privilegios de acceso a los datos; en otras palabras, establece la seguridad de la base de datos.
  GRANT, REVOKE

- El Diccionario de Datos / Data Dictionary (DD), en el cual se encuentran la definición de los objetos (tablas, vistas, procedimientos, triggers, tipos de datos, reglas, defaults) que componen la base de datos. Dentro del control de las transacciones encontramos las instrucciones:
  COMMIT, ROLLBACK, SAVEPOINT.

2.2.5 Definición de datos

Dentro de una base de datos se encuentran distintos tipos de objetos como: tablas, tipos de datos, campos, índices, reglas y defaults.

2.2.5.1 Tablas

Creación de tablas
Al crear una tabla se define el nombre de la misma así como los nombres de los campos, el tipo de datos y el tamaño de cada campo que va a contener dicha tabla. Al momento de crear la tabla no es necesario especificar el tamaño de la misma, automáticamente es asignado por el servidor.

Para crear las tablas se necesita ser usuario de la base en la cual se desea construir y tener permisos de creación (CREATE TABLE) por parte del DBA (Data Base Administrator). La sintaxis para crear una tabla es:

create table [usuario.]<nombre tabla> (sybase, sql server)
(<nombre campo> <tipo de dato> [<(tamaño)>] [acepta nulos],....)

Eliminación de tablas
Para eliminar una tabla y los datos que contiene, así como sus índices, triggers y permisos se utiliza la siguiente instrucción:

drop table <nombre tabla>

Alteración de tablas
Una vez creada una tabla, e incluso si ésta ya contiene información, se pueden agregar campos a la misma; para esto se utiliza el comando ALTER TABLE, con el cual podemos añadir columnas, añadir, cambiar, borrar constraints. En Oracle se puede modificar la definición de un campo. Su sintaxis es:

alter table [usuario.]<nombre tabla> add <nombre campo>
(sql estándar)
<tipo dato> [<tamaño>][<default>]

2.2.5.2 Reglas (Sybase, MS SQL Server)

Creación de reglas
Una regla especifica un rango de valores aceptables (condición) para una columna en particular o para cualquier columna con datos definidos por el usuario. Para definirla se realizan dos pasos: primero se define la regla y en segundo lugar se vincula esta regla al campo deseado. La sintaxis para crear una regla es:

create rule [usuario.]<nombre regla> as @<variable operador expresión>
[and | or] ....

Para vincular una regla a un campo de una tabla se utiliza el siguiente procedimiento almacenado.
sp_bindrule "<nombre regla>", "<tabla.campo>"

Para desvincular una regla a un campo se utiliza el siguiente procedimiento almacenado:
sp_unbindrule "<tabla.campo>"

Eliminación de reglas
Antes de eliminar una regla, ésta no debe estar vinculada a ningún campo. Para eliminar una regla se utiliza el comando:

drop rule [usuario.]rule_name [, [usuario.]rule_name]...

2.2.5.3 Defaults

Creación de defaults
Un DEFAULT se define cuando se desea que un campo tenga un valor aún cuando el usuario no lo introduzca. Al igual que las reglas, para la utilización de DEFAULTS se requieren dos pasos: el primero es definir el default y el segundo es vincular este default al campo deseado. La sintaxis para crear un DEFAULT es:

create default [usuario.]<nombre default> as <valor>

Para vincular un DEFAULT a un campo de una tabla se utiliza el siguiente procedimiento almacenado:
sp_bindefault "<nombre default>", "<tabla.campo>"

Para desvincular un DEFAULT de un campo se utiliza el siguiente procedimiento almacenado:
sp_unbindefault "<tabla.campo>"

Eliminación de defaults
Antes de eliminar un DEFAULT, éste no debe estar vinculado a ningún campo. Para eliminar un default se utiliza el comando:

drop default [owner.]default_name [, [owner.]default_name]...

2.2.5.4 Índices

Creación de índices
Un índice es una estructura de almacenamiento físico y que ocupa un espacio. Los índices ayudan al Servidor SQL a localizar datos y son transparentes para el usuario. El propósito principal de un índice es proporcionar un acceso más rápido a los datos, aunque en algunos casos su propósito es asegurar que el contenido de un campo sea único. Se necesitan permisos de sistema: CREATE INDEX.

El Servidor SQL decide si usar o no un índice desde una tabla. Las tablas pueden tener más de un índice. Así como los datos en una tabla cambian con el tiempo, el Servidor SQL puede cambiar los índices de las tablas para reflejar esas modificaciones.

Cuando una tabla es borrada sus índices correspondientes también son borrados. Los índices pueden ser creados de manera automática cuando se define una llave primaria o una definición de un constraint para valores únicos. El nombre del índice estará dado por el constraint.

El Servidor SQL soporta los siguientes tipos de índices:

- *Indices compuestos*: Estos índices involucran más de una columna. Este índice es usado cuando dos o más columnas son buscadas mejor como una unidad a causa de su relación lógica.

- *Indices únicos*: Estos índices no permiten más de dos líneas en la columna especificada con el mismo valor. El Servidor SQL checa por valores duplicados cuando el índice es creado (si los datos ya existen) y cada vez que los datos son añadidos.
- *clustered* y *nonclustered*. Los índices clustered obligan al Servidor SQL a ordenar continuamente y reordenar las líneas de una tabla, además de que su orden físico es el mismo que el orden lógico (indexado).

En los índices tipo *clustered* los datos son almacenados de acuerdo al orden especificado en el índice; por lo tanto, cada tabla sólo puede tener un índice de este tipo, ya que los datos sólo pueden estar almacenados físicamente en un orden.

En cambio, cada tabla puede tener hasta 249 índices *nonclustered*, debido a que éstos no ordenan físicamente los datos, sino que mantienen apuntadores o "ligas" a los registros.

Para crear índices se utiliza el siguiente comando:

create [unique][clustered | nonclustered] index <nombre índice>
on <nombre tabla>(<campo>,...) (sybase, sql server)

Eliminación de índices
Para eliminar un índice se utiliza la siguiente instrucción:
drop index <nombre índice>

Integridad
La integridad de una base de datos se refiere no solo a que los datos sean consistentes dentro de la base, sino además, que los valores que posean los datos sean válidos de acuerdo a las dependencias funcionales entre tablas y de acuerdo a las políticas de negocio.

La integridad de la base de datos se puede lograr mediante:

- Manteniendo una redundancia mínima y controlada
- El establecimiento de llaves primarias
- Estableciendo reglas de validación durante la creación y edición de los datos
- Estableciendo procedimientos que validen la dependencia funcional entre tablas relacionadas (integridad referencial)

Estos controles mantienen la consistencia dentro de la base de datos, lo cual se lleva acabo de forma automática al momento de insertar, borrar o actualizar datos, mediante los contraints, triggers, índices primarios y llaves primarias. Si se viola cualquiera de las restricciones establecidas la instrucción no será tomada en cuenta. Estas restricciones son llevadas a cabo a nivel de base de datos por lo que se llevan a cabo de forma independiente al cliente que se este utilizando.

Transacciones
Los cambios realizados en los datos de la base no son guardados hasta que el usuario o la aplicación deciden explícitamente que las instrucciones de insertar, actualizar y eliminar deben de hacerse permanentes. Una transacción puede comprender una o más instrucciones SQL, comienza desde el momento en que el usuario se conecta a la base de datos y termina cuando se emite una instrucción de confirmación: commit, o de anulación: rollback. Al momento de ser confirmados o descartados los cambios, no se puede dar marcha atrás.

2.2.6 Álgebra relacional

El Álgebra relacional se basa en la teoría de conjuntos y de relaciones y en el álgebra de conjuntos. Adicionalmente al conjunto básico de operadores como: unión, diferencia, producto cartesiano, intersección y join incorpora operadores específicos de base de datos tales como proyección y

selección. Dado que el resultado de una operación del álgebra relacional es una relación, ésta a su vez puede ser sujeto de posteriores operaciones algebraicas.

2.2.6.1 Operadores de Conjuntos.

Proyección
La proyección selecciona y genera un subconjunto con los atributos indicados de una tabla. También es conocida como operación vertical.

Selección
La selección selecciona y genera un subconjunto con los renglones indicados de una tabla. También es conocida como operación horizontal.

Unión
La operación unión realiza la misma acción que en el álgebra de conjuntos, es decir {1,4,5,10} U {1,4,3,9} = {1,3,4,5,9,10}. En términos de tablas, hay que considerar que la unión sea compatible, el número de atributos debe ser el mismo y del mismo tipo de datos.

Producto Cartesiano
El producto cartesiano es el producto cruz entre 2 tablas: {a,b} X {1,2} = {a1, a2, b1, b2}

El resultado es la unión de cada renglón de una tabla con cada renglón de la otra tabla. El producto es una yuxtaposición donde los elementos son combinados o concatenados.

Join
La operación Join es en esencia un producto cartesiano, donde se selecciona las columnas que satisfagan las condiciones indicadas. Es la operación más común en las bases de datos relacionales.

2.2.6.2 Selección de datos

Para seleccionar o consultar los registros y campos de una tabla se utiliza el comando SELECT. Este comando es quizá el más versátil y complejo del Transact-SQL (Sybase) y del PL/SQL (Oracle); con él podemos realizar desde una consulta simple a una tabla hasta consultas complejas donde se involucre más de una tabla; permite especificar condiciones de consulta y/o selección, así como generar consultas cuyo resultado se despliegue en un orden específico, agrupaciones de datos, entre otras características. A continuación se presenta la sintaxis del comando:

select * | [distinct] <campo>, <campo>
from <nombre tabla>, [<nombre tabla>]
[where <condición>]
[group by <campo>, <campo>,...]
[having <condición>]
[order by <campo>, <campo>,....]

La instrucción SELECT tiene dos componentes que son indispensables y deben estar en cualquier instrucción de este tipo; éstas son: las palabras SELECT y FROM. Los demás componentes son opcionales.

2.2.6.3 Inserción de datos

Para insertar datos en una tabla se utiliza el comando INSERT, el cual se puede utilizar en dos formas: la primera de ellas es insertando un solo registro a la vez y la segunda es insertando en la misma instrucción uno o más registros. En este último caso, se combina con la instrucción SELECT.

Cabe señalar que cuando no se inserta valor en algún campo, SQL Server inserta el valor que está definido en el DEFAULT vinculado; en caso de que no tenga vinculado ningún DEFAULT, entonces se insertará un valor nulo. Cuando se trata de insertar un valor nulo en un campo que no lo permite, SQL Server indica el error correspondiente.

2.2.6.4 Eliminación de registros

Existen dos comandos para borrar registros de una tabla, DELETE y TRUNCATE, que a continuación se explican.

Truncate
La instrucción TRUNCATE elimina *todos* los registros de la tabla indicada, con la particularidad de que sólo puede ser ejecutada por el dueño de la tabla. La sintaxis es:

Truncate table <nombre tabla>

Delete
La instrucción DELETE también elimina registros de la tabla indicada, sólo que en este caso y a diferencia de la instrucción TRUNCATE, el usuario puede especificar algún criterio o condición de eliminación. En caso de no hacerlo, se eliminan todos los registros de la tabla. La sintaxis es la siguiente:

Delete <nombre tabla>
[From <nombre tabla1>, <nombre tabla2>, ...]
[Where <condición>]

2.2.6.5 Actualización de datos

Para modificar o actualizar los valores de los campos de una tabla se utiliza el comando UPDATE, en el cual se indica la tabla, los campos a actualizar, los nuevos valores y en su caso, la condición de la actualización. La sintaxis es la siguiente:

Update <nombre tabla>
Set <campo1> = <valor1>, <campo2> = <valor2>, ...
[From <nombre tabla1>, ...]
[Where <condición>]

2.2.6.6 Vistas

Una vista es una manera lógica de ver los datos alojados en una o más tablas a través de una instrucción SELECT. Una vista no contiene los datos físicamente almacenados por ella misma, sino que es una ventana a los datos almacenados en sus respectivas tablas; por lo tanto, no ocupa espacio en la base de datos. De una vista sólo se almacena en el servidor la definición de la misma.

Creación de vistas
Una vista puede ser utilizada como cualquier otra tabla. La sintaxis para crear una vista es:
Create view <nombre_vista> as <instrucción SELECT>
Eliminación de Vistas
Para eliminar una vista se utiliza la siguiente instrucción:
Drop view <nombre_vista>

2.2.7 Definición de privilegios

El otorgamiento o revocación de permisos o privilegios de acceso o ejecución a los distintos objetos que componen la base de datos constituyen un aspecto muy importante en la seguridad de la misma. Comúnmente es el administrador del sistema el encargado de asignar o revocar permisos y/o crear usuarios en la base de datos.

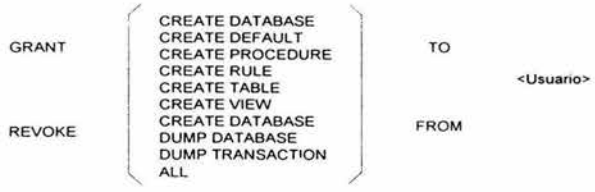
Antes de que un usuario pueda acceder a alguna base de datos, éste debe primero darse de alta como usuario en el servidor. Para esto se usa el siguiente procedimiento almacenado, teniendo activa la base de datos MASTER:

sp_addlogin <nombre_usuario> , [<password>] , [<base de datos default>]

Posteriormente debe darse de alta al usuario en la base de datos que se desea pueda acceder; para esto se activa la base de datos deseada y se utiliza el siguiente procedimiento almacenado para darlo de alta:

sp_adduser <nombre_usuario>

Dentro de Transact-SQL se pueden controlar los privilegios a los usuarios de creación de objetos en la base de datos, mediante la siguiente sintaxis:



Donde GRANT se utiliza para otorgar privilegios y REVOKE para eliminarlos.

Existe otro nivel de privilegios, el cual comprende la manipulación de datos, es decir la otorgación/revocación de permisos para actualizar, borrar, insertar, etc., datos. Esto se realiza con la siguiente sintaxis.

Donde GRANT se utiliza para otorgar privilegios y REVOKE para eliminarlos.

2.2.8 Modelo entidad-relación

Propuesto por P. Chen en 1976, modificado y ampliado por varios autores más, el modelo Entidad-Relación (E-R) está considerado como una forma de trabajar el modelo semántico de Bases de Datos y es muy utilizado para el modelado de datos. De hecho, una vez propuesto el modelo de

datos en E-R, existen métodos para convertirlo a cualquier modelo lógico de Bases de Datos existente (relacional, jerárquico, reticular). A pesar de la inclusión del modelado de bases de datos orientadas a objetos, el modelo semántico E-R no pierde vigencia y sigue utilizándose bajo este enfoque.

Las partes que conforman el modelo E-R se describen a continuación:

- Entidad.- Se expresa por medio de un rectángulo que en su interior lleva el nombre de un sustantivo, el cual puede representar persona, lugar, cosa o evento de interés en el sistema. Existen también las *entidades débiles*, las cuales se dibujan con un rectángulo de línea doble y no pueden existir por sí solas, esto es, dependen de una entidad fuerte. Ejemplo: si se tiene la Entidad vehículo, la entidad automóvil sería una entidad débil.

- Atributos.- Se utilizan para detallar el contenido de las entidades. Se representan por óvalos unidos al rectángulo de la entidad a la que pertenecen por medio de una línea recta. Existen dos tipos de atributos: los descriptivos y los identificadores. Éstos últimos solo los tienen las Entidades Fuertes y sirven para identificar un ejemplar de la entidad de otro. Los atributos identificadores van subrayados y son conocidos como llaves primarias o simplemente llaves.

- Relación.- Se representa por medio de un diamante que lleva dentro el nombre de una asociación entre dos o más entidades. Una relación puede o no contener atributos. Para establecer la relación semántica entre una entidad y otra se debe utilizar una relación (diamante) y la ocurrencia de esta conectividad (uno a uno, uno a muchos y muchos a muchos), se explícita poniendo números encima de las líneas de unión. Cabe aclarar que esta notación ha variado con el tiempo y una de las más utilizadas es la de poner punta de flecha a la línea que va del diamante a la entidad en lugar de ir un número uno sobre la raya.

En la Figura II.3 se muestran los elementos que conforman el modelo entidad – relación.

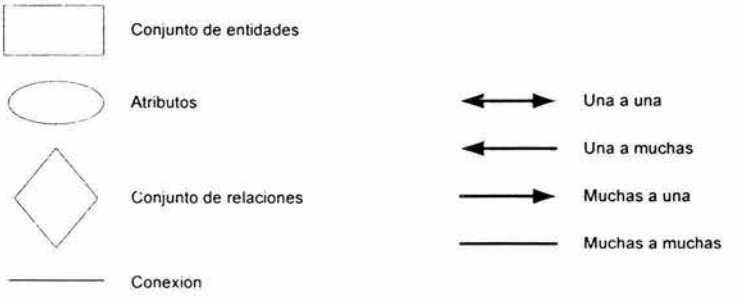


Figura II.3 Elementos del modelo entidad - relación

Dentro de las extensiones que tiene el modelo Entidad-Relación están los conceptos de Jerarquía de Generalización (generalización) y Jerarquía de subconjunto (especialización), las cuales se describen a continuación:

- Jerarquía de generalización.- Una entidad E es la generalización de las entidades E1, E2, ..., En si cada ocurrencia de E es también una ocurrencia de una y sólo una de las entidades E1, E2, .., En. Ejemplo: La entidad EMPLEADO es generalización de SECRETARIA, TÉCNICO e INGENIERO.

- Jerarquía de subconjunto.- Una entidad E1 es un subconjunto de otra entidad E2 si para toda ocurrencia de E1 hay también una ocurrencia de E2. Ejemplo: una entidad CUENTA puede incluir CUENTA_AHORRO, CUENTA_INVER como especializaciones que constituyen subconjuntos de la primera. Toda cuenta de ahorros o cuenta de inversión es también una cuenta.

2.2.9 Normalización de datos

El proceso de normalización es un estándar que consiste, básicamente, en un proceso de conversión de las relaciones entre las entidades, evitando:

- La redundancia de los datos: repetición de datos en un sistema.
- Anomalías de actualización: inconsistencias de los datos como resultado de datos redundantes y actualizaciones parciales.
- Anomalías de borrado: pérdidas no intencionadas de datos debido a que se han borrado otros datos.
- Anomalías de inserción: imposibilidad de adicionar datos en la base de datos debido a la ausencia de otros datos.

Tomando como referencia la tabla de la Figura II.4 siguiente:

AUTORES Y LIBROS

NOMBRE	NACION	CODLIBRO	TITULO	EDITOR
Date	USA	999	IBD	AW
Ad.Mig.	ESP	888	CyD	RM
Ma.Piat.	ITA	777	CyD	RM
Date	USA	666	BdD	AW

Figura II.4 Tabla no normalizada

Se plantean una serie de problemas:

- Redundancia: cuando un autor tiene varios libros, se repite la nacionalidad.
- Anomalías de modificación: Si Ad.Mig. y Ma.Piat. desean cambiar de editor, se modifica en los 2 lugares. A priori no podemos saber cuántos autores tiene un libro. Los errores son frecuentes al olvidar la modificación de un autor. Se pretende modificar en un sólo sitio.
- Anomalías de inserción: Se desea dar de alta un autor sin libros, en un principio. NOMBRE y CODLIBRO son campos clave, una clave no puede tomar valores nulos.

Asegurando:

- Integridad entre los datos: consistencia de la información.

El proceso de normalización nos conduce hasta el modelo físico de datos y consta de varias fases denominadas formas normales.

Definición de la clave

Antes de proceder a la normalización de la tabla lo primero que debemos de definir es una clave, esta clave deberá contener un valor único para cada registro (no podrán existir dos valores iguales en toda la tabla) y podrá estar formado por un único campo o por un grupo de campos.

En la tabla de alumnos de un centro de estudios no podemos definir como campo clave el nombre del alumno ya que pueden existir varios alumnos con el mismo nombre. Podríamos considerar la posibilidad de definir como clave los campos nombre y apellidos, pero estamos en la misma

situación: podría darse el caso de alumnos que tuvieran los mismos apellidos y el mismo nombre (Juan Fernández Martín).

La solución en este caso es asignar un código de alumno a cada uno, un número que identifique al alumno y que estemos seguros que es único. Una vez definida la clave podremos pasar a aplicar las formas normales.

Formas Normales

Forma Normal: para un conjunto de relaciones dado, una forma normal es un conjunto de restricciones que determinan el nivel de redundancia presente en las relaciones. Es una medida de la redundancia en un conjunto de relaciones. Existen varias formas normales, las cuales se muestran en la Figura II.5.

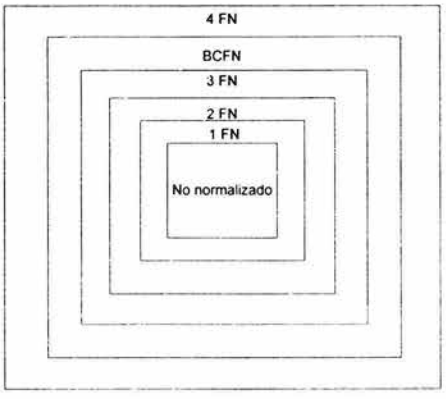


Figura II.5 Esquema de las formas normales

- *Primera Forma Normal (1FN)*

Un conjunto de relaciones esta en 1FN, si todos los atributos presentes en éstas, son atómicos, es decir que los atributos sean únicos. Ejemplo:

VENTA (n°factura, fecha_fact, ...{cod_producto, nombre, precio_unitario, cantidad},....)

- *Segunda Forma Normal (2FN)*

Un conjunto de relaciones esta en 2FN, si no contiene Dependencias Funcionales Parciales. Dependencias Funcionales Parciales, existe cuando atributos no clave, dependen funcionalmente de una parte de un conjunto de atributos clave. Ejemplo:

VENTA (n°factura, fecha_factura, ...)
DETALLE FACTURA (n°factura + cód.producto, nombre, precio_unitario, cantidad, total parcial)
cód.producto => nombre (esta es una dependencia funcional parcial)
Por lo tanto, si no hay clave compuesta, no existe dependencia funcional parcial.

- *Tercera Forma Normal (3FN)*

Un conjunto de relaciones esta en 3FN, si no presenta Dependencia Funcional Transitiva. Existe Dependencia Funcional Transitiva, si atributos no clave, dependen funcionalmente de otros atributos no clave. Ejemplo:

VENTA(n°factura, fecha_fact, monto_neto, IVA, total, rut_cliente, nombre_cli, direcc_cli, giro, rut_vendedor, nombre_vendedor)
CLIENTE (rut_cliente, nombre_cli, direcc_cli, giro)
VENDEDOR (rut_vendedor, nombre_vendedor)
VENTA (n°factura, fecha_fact, monto_neto, IVA, total, rut_cliente, rut_vendedor)

- *Forma Normal de Boyce Codd (BCNF)*

Esta en BCNF si no presenta Dependencias de Boyce Codd. Existe Dependencia Boyce Codd, si atributos que forman parte de una clave, dependen funcionalmente de atributos no clave. Ejemplo:

CTAS.CTES (n°sucursal + n°cuenta, fecha apertura, saldo, rut cliente, cód ejecutivo...)
cód.ejecutivo $\longrightarrow$ n°sucursal (con el código de ejecutivo, puedo saber a qué sucursal pertenece)

El ejemplo anterior, demuestra el hecho que no existe un algoritmo o proceso que permita a partir de un conjunto de relaciones no normalizadas, obtener un conjunto equivalente en la forma BCFN, sin pérdida de información y conservando las dependencias.

- *Cuarta Forma Normal (4FN)*

Una tabla está en cuarta forma normal si y solo si está en BCNF y para cualquier combinación clave (campo) no existen valores duplicados

2.3 Redes

Una red de computadoras es un conjunto de terminales, nodos, servidores y elementos de propósito especial que interaccionan entre sí con la finalidad de intercambiar información y compartir recursos. Anteriormente (aunque existen todavía) la información se almacenaba en los llamados "Mainframes". De este elemento, diferentes terminales eran conectadas con la finalidad de recibir información y compartir los recursos. El problema con este tipo de "red" es que se basaba en un sistema centralizado, es decir, del mainframe se extraía toda la información, teniendo de esta forma la limitante en cuanto a la capacidad de almacenamiento de datos.

Con la introducción de las diferentes clases de redes de computadoras y tecnologías, surgió la posibilidad de utilizar diferentes "servidores" que, como su nombre lo indica, proveen servicios a un conjunto de nodos denominados "clientes". De esta forma, no existe limitante en cuanto almacenamiento de información, ya que nuevos servidores pueden ser instalados, dando así, la facilidad en la expansión de las redes. De esta forma, una "red de computadoras" se define como un sistema distribuido. En un sistema distribuido, la existencia de múltiples computadoras autónomas es transparente para el usuario. El usuario puede teclear una orden para ejecutar un programa y éste se ejecutará. La tarea de seleccionar al mejor o el correspondiente procesador y colocar los resultados en el lugar apropiado, corresponde al sistema operativo y a la red en sí.

En un sistema distribuido, el usuario no está consciente de que existen múltiples procesadores. El sistema se ve como un monoprocesador virtual. El sistema distribuido es un sistema de software construido encima de una red.

2.3.1 Topologías de Red

Una topología de red va relacionada con el tipo de conexión entre los diferentes dispositivos que forman la red de computadoras, es decir, la forma como se interconectan todos los dispositivos para formar la red. Existen tres topologías básicas que son utilizadas para formar redes: Star (estrella), Ring (Anillo) y Bus. De estas tres topologías principales, es posible generar diferentes topologías "híbridas", logrando así, una integración entre las topologías básicas, expandiendo las redes de computadoras hacia redes de cobertura global.

Topología Estrella

La topología estrella, mostrada en la Figura II.6, consta de una unidad central que controla el flujo de información a través de la red. La topología estrella tiene limitaciones en cuanto a rendimiento y confiabilidad, ya que el tamaño de la red depende directamente de la capacidad del controlador central (número de conexiones que puede soportar) y en caso de fallar éste, todo el sistema deja de funcionar. Por otro lado, tiene la ventaja de poderse administrar únicamente administrando el dispositivo central. En la topología estrella se tiene un control de transmisión Centralizado y una forma de transferencia de Conmutación.

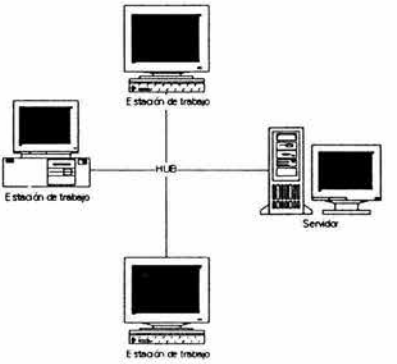


Figura II.6 Topología Estrella

Topología Anillo

Una de sus características importantes es que está formado por un conjunto de enlaces punto a punto, lo cual es una topología bien entendida y probada, en donde la información es pasada a través de los nodos de uno a uno en una comunicación peer to peer. La ventaja que tiene esta topología es que no se requiere un cuarto de control central, aunque la desventaja es que si uno de los enlaces peer to peer que la forman se rompe (o se desconecta debido a errores en la transmisión, etc.), la red deja de funcionar. La Figura II.7 muestra este tipo de topología.

El control de transmisión que usa esta topología es Distribuido y su modo de transferencia es de Conmutación. La tecnología común que utiliza dicha topología es denominada Token Ring.

Token Ring es una tecnología desarrollada por IBM, corresponde al estándar IEEE 802.5. El diseño básico es un anillo de nodos que no superan los 256, operando a 4 ó 16 Mbps. En Token Ring se utiliza un código de autorización llamado Token que actúa como método de acceso al medio denominado Token Passing.

El método de acceso al medio Token Passing trabaja de la siguiente forma. Si no hay mensaje, el token (tres bytes) es enviado a través del anillo. Cuando un nodo A con un mensaje a enviar recibe el token, retiene éste y envía el mensaje, el cual incluye un código de identificación del destinatario. Los nodos ignoran el mensaje si no es para sí mismo, en caso contrario, obtienen la información. La información sigue viajando hasta que se completa su trayectoria alrededor del anillo hasta que llega al nodo A. Dicho nodo suelta el token para que pase nuevamente alrededor del anillo para futuros envíos de información.

Topología Bus

En esta topología, mostrada en la Figura II.8, no existe un CPU o similar que controle la comunicación entre los nodos. Cada nodo está conectado a un bus, donde cada nodo actúa como si fuera parte de una red anillo, pero ninguno depende del nodo siguiente para que el flujo de información continúe, ni tampoco depende del nodo anterior para que la información llegue a él. La tecnología común que trabaja bajo una topología Bus es denominada Ethernet.

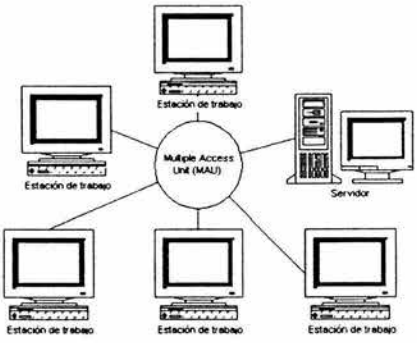


Figura II.7 Topología Anillo

Ethernet fue desarrollada por Digital, Intel y Xerox, normalizada con IEEE 802.3. Ethernet distribuye paquetes de datos de 1500 bytes y un relleno de 46 bytes con una velocidad de 10 Mbps a los diferentes nodos dispersos a lo largo de un bus que comúnmente es cable coaxial. Los nodos pueden estar separados hasta 50m de largo unidos también por cable par trenzado. Una red Ethernet puede estar formada hasta por 1024 nodos.

Así como Token Ring utiliza un Token como acceso al medio, Ethernet se basa en el acceso al medio denominado CSMA/CD (Carrier Sense Multiple Access with Collision Detect). Es denominada Carrier Sense porque cada nodo es capaz de saber si la información que viaja en el Bus es para sí mismo o no. Multiple Access porque como se ha mencionado, un bus es compartido por todos los nodos que forman la red. Collision Detect porque cada nodo sabe si existe información que viaja en la red y es posible detectar y eliminar colisiones.

Figura II.8 Topología Bus

2.3.2 Modelo OSI

Uno de los estándares más utilizados, y del cual prácticamente se basan los posteriores estándares, así como las diferentes tecnologías de red es el modelo OSI (Open System Interconnection), desarrollado por ISO (International Standard Organization) en 1984.
El modelo OSI está formado por siete capas (layers) como lo muestra la Figura II.9:

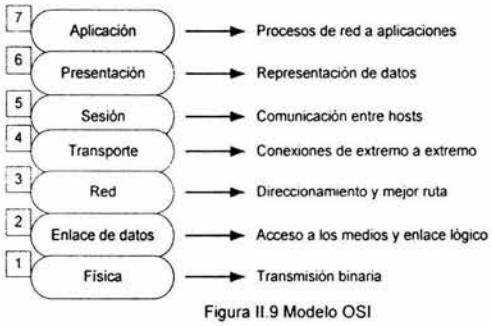


Figura II.9 Modelo OSI

Cada capa individual del modelo OSI tiene un conjunto de funciones que debe realizar para que los paquetes de datos puedan viajar en la red desde el origen hasta el destino. A continuación, presento una breve descripción de cada capa del modelo de referencia OSI.

Capa 1: La capa física
La capa física define las especificaciones eléctricas, mecánicas, de procedimiento y funcionales para activar, mantener y desactivar el enlace físico entre sistemas finales. Las características tales como niveles de voltaje, temporización de cambios de voltaje, velocidad de datos físicos, distancias de transmisión máximas, conectores físicos y otros atributos similares se definen a través de las especificaciones de la capa física.

Capa 2: La capa de enlace de datos
La capa de enlace de datos proporciona un tránsito de datos confiable a través de un enlace físico. Al hacerlo, la capa de enlace de datos se ocupa del direccionamiento físico (comparado con el lógico), la topología de red, el acceso a la red, la notificación de errores, entrega ordenada de tramas y control de flujo.

Capa 3: La capa de red:
La capa de red es una capa compleja que proporciona conectividad y selección de ruta entre dos sistemas de host que pueden estar ubicados en redes geográficamente distintas.

Capa 4: La capa de transporte
La capa de transporte segmenta los datos originados en el host emisor y los reensambla en una corriente de datos dentro del sistema del host receptor. El límite entre la capa de sesión y la capa de transporte puede imaginarse como el límite entre los protocolos de capa de medios y los protocolos de capa de host. Mientras que las capas de aplicación, presentación y sesión están relacionadas con aspectos de las aplicaciones, las tres capas inferiores se encargan del transporte de datos.

La capa de transporte intenta suministrar un servicio de transporte de datos que aísla las capas superiores de los detalles de implementación del transporte. Específicamente, temas como la confiabilidad del transporte entre dos hosts es responsabilidad de la capa de transporte. Al proporcionar un servicio de comunicaciones, la capa de transporte establece, mantiene y termina adecuadamente los circuitos virtuales. Al proporcionar un servicio confiable, se utilizan dispositivos de detección y recuperación de errores de transporte.

Capa 5: La capa de sesión
Como su nombre lo implica, la capa de sesión establece, administra y finaliza las sesiones entre dos hosts que se están comunicando. La capa de sesión proporciona sus servicios a la capa de presentación. También sincroniza el diálogo entre las capas de presentación de los dos hosts y administra su intercambio de datos. Además de regular la sesión, la capa de sesión ofrece

disposiciones para una eficiente transferencia de datos, clase de servicio y un registro de excepciones acerca de los problemas de la capa de sesión, presentación y aplicación.

Capa 6: La capa de presentación
La capa de presentación garantiza que la información que envía la capa de aplicación de un sistema pueda ser leída por la capa de aplicación de otro. De ser necesario, la capa de presentación traduce entre varios formatos de datos utilizando un formato común.

Capa 7: La capa de aplicación
La capa de aplicación es la capa del modelo OSI más cercana al usuario; suministra servicios de red a las aplicaciones del usuario. Difiere de las demás capas debido a que no proporciona servicios a ninguna otra capa OSI, sino solamente a aplicaciones que se encuentran fuera del modelo OSI. Algunos ejemplos de dichos procesos de aplicación son los programas de hojas de cálculo, de procesamiento de texto y los de las terminales bancarias. La capa de aplicación establece la disponibilidad de los potenciales socios de comunicación, sincroniza y establece acuerdos sobre los procedimientos de recuperación de errores y control de la integridad de los datos.

Encapsulamiento
Si una computadora (host A) desea enviar datos a otra (host B), en primer término los datos deben empaquetarse a través de un proceso denominado encapsulamiento. El encapsulamiento rodea los datos con la información de protocolo necesaria antes de que se una al tránsito de la red. Por lo tanto, a medida que los datos se desplazan a través de las capas del modelo OSI, reciben encabezados (información correspondiente a la dirección), información final y otros tipos de información.

El empaquetamiento y el flujo de los datos que se intercambian experimentan cambios a medida que las redes ofrecen sus servicios a los usuarios finales. La Figura II.10 muestra el encapsulamiento de los datos.

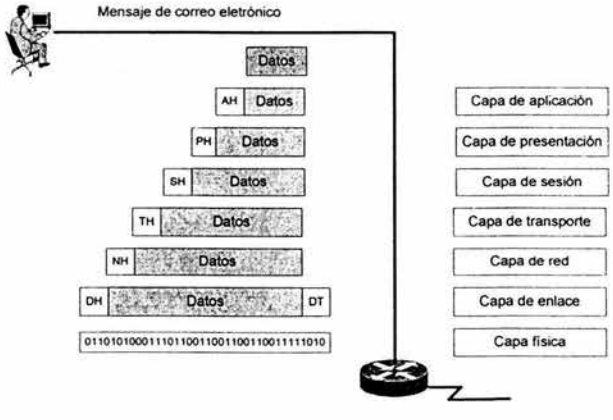


Figura II.10 Encapsulamiento

Los terminos AH, PH, etc. denotan las cabeceras añadidas por cada uno de los niveles (Aplication Header, Presentation Header, etc.)

2.3.3 Clasificación de Redes

Para facilitar su estudio, la mayoría de las redes de datos se han clasificado en redes de área local (LAN) o redes de área amplia (WAN). Las LAN generalmente se encuentran en su totalidad dentro del mismo edificio o grupo de edificios y manejan las comunicaciones entre las oficinas. Las WAN cubren un área geográfica más extensa y conectan ciudades y países. Las LAN y/o las WAN también se pueden conectar entre sí mediante internetworking.

2.3.3.1 Redes de Área Local (LAN)

Las redes de área local (LAN) se componen de computadores, tarjetas de interfaz de red, medios del networking, dispositivos de control del tráfico de red y dispositivos periféricos. Las LAN hacen posible que las empresas que utilizan tecnología informática compartan de forma eficiente elementos tales como archivos e impresoras, y permiten la comunicación, por ejemplo, a través del correo electrónico. Unen entre sí: datos, comunicaciones, servidores de computadoras y de archivo.

Las LAN están diseñadas para realizar lo siguiente:

- operar dentro de un área geográfica limitada
- permitir que varios usuarios accedan a medios de ancho de banda alto
- proporcionar conectividad continua con los servicios locales
- conectar dispositivos físicamente adyacentes

Dispositivos de LAN básicos

Repetidor
El propósito de un repetidor es regenerar y retemporizar las señales de red a nivel de los bits para permitir que los bits viajen a mayor distancia a través de los medios. Tenga en cuenta la Norma de cinco repetidores, también denominada Norma 5-4-3, cuando extienda los segmentos LAN. Esta norma establece que se pueden conectar cinco segmentos de red de extremo a extremo utilizando cuatro repetidores pero sólo tres segmentos pueden tener host (computadores) en ellos.

Los repetidores son dispositivos con un solo puerto "de entrada" y un solo puerto "de salida". En el modelo OSI, los repetidores se clasifican como dispositivos de Capa 1, dado que actúan sólo a nivel de los bits y no tienen en cuenta ningún otro tipo de información.

Hubs
El propósito de un hub es regenerar y retemporizar las señales de red. Esto se realiza a nivel de los bits para un gran número de hosts (por ej., 4, 8 o incluso 24) utilizando un proceso denominado concentración. Podrá observar que esta definición es muy similar a la del repetidor, es por ello que el hub también se denomina repetidor multipuerto. La diferencia es la cantidad de cables que se conectan al dispositivo. Las razones por las que se usan los hubs son crear un punto de conexión central para los medios de cableado y aumentar la confiabilidad de la red. La confiabilidad de la red se ve aumentada al permitir que cualquier cable falle sin provocar una interrupción en toda la red. Esta es la diferencia con la topología de bus, en la que si un cable falla, esto causa una interrupción en toda la red. Los hubs se consideran dispositivos de la Capa 1 dado que sólo regeneran la señal y la envían por medio de un broadcast de ella a todos los puertos (conexiones de red).

Puentes
Un puente es un dispositivo de la capa 2 diseñado para conectar dos segmentos de LAN. El propósito de un puente es filtrar el tráfico de una LAN, para que el tráfico local siga siendo local, pero permitiendo que el tráfico que se ha dirigido hacia allí pueda ser conectado con otras partes (segmentos) de la LAN. Cada dispositivo de networking tiene una dirección MAC exclusiva en la NIC, el puente rastrea cuáles son las direcciones MAC que están ubicadas a cada lado del puente y toma sus decisiones basándose en esta lista de direcciones MAC.

El aspecto de los puentes varía enormemente según el tipo de puente. Aunque los routers y los switches han adoptado muchas de las funciones del puente, estos siguen teniendo importancia en muchas redes. Es importante tener en cuenta que, al igual que un repetidor, el puente conecta solamente dos segmentos a la vez. Como sucede en el caso de la combinación repetidor/hub, hay otro dispositivo que se utiliza para conectar múltiples puentes.

Switches
Un switch, al igual que un puente, es un dispositivo de la capa 2. De hecho, el switch se denomina puente multipuerto, así como el hub se denomina repetidor multipuerto. La diferencia entre el hub y el switch es que los switches toman decisiones basándose en las direcciones MAC y los hubs no toman ninguna decisión. Como los switches son capaces de tomar decisiones, hacen que la LAN sea mucho más eficiente. Los switches hacen esto "conmutando" datos sólo desde el puerto al cual está conectado el host correspondiente. A diferencia de esto, el hub envía datos a través de todos los puertos de modo que todos los hosts deban ver y procesar (aceptar o rechazar) todos los datos.

A primera vista los switches parecen a menudo similares a los hubs. Tanto los hubs como los switches tienen varios puertos de conexión, dado que una de sus funciones es la concentración de conectividad (permitir que varios dispositivos se conecten a un punto de la red). La diferencia entre un hub y un switch está dada por lo que sucede dentro del dispositivo.

El propósito del switch es concentrar la conectividad, haciendo que la transmisión de datos sea más eficiente. Por el momento, piense en el switch como un elemento que puede combinar la conectividad de un hub con la regulación de tráfico de un puente en cada puerto. El switch conmuta paquetes desde los puertos (las interfaces) de entrada hacia los puertos de salida, suministrando a cada puerto el ancho de banda total (la velocidad de transmisión de datos en el backbone de la red).

Routers
El router está ubicado en la capa de red del modelo OSI, o capa 3. Al trabajar en la capa 3, esto permite que el router tome decisiones basándose en grupos de direcciones de red (clases) a diferencia de las direcciones MAC individuales, que es lo que se hace en la capa 2. Los routers también pueden conectar distintas tecnologías de la capa 2 como, por ejemplo, Ethernet, Token-ring y FDDI. Sin embargo, dada su aptitud para enrutar paquetes basándose en la información de la Capa 3, los routers se han transformado en el backbone de Internet, ejecutando el protocolo IP.

El propósito de un router es examinar los paquetes entrantes (datos de la capa 3), elegir cuál es la mejor ruta para ellos a través de la red y luego conmutarlos hacia el puerto de salida adecuado. Los routers son los dispositivos de regulación de tráfico más importantes en las redes de gran envergadura. Permiten que prácticamente cualquier tipo de computadora se pueda comunicar con otra computadora en cualquier parte del mundo. Aunque ejecutan estas funciones básicas, los routers también pueden ejecutar muchas otras tareas.

2.3.3.2 Redes de Área Amplia (WAN)

Las WAN interconectaban las LAN, que a su vez proporcionaban acceso a las computadoras o a los servidores de archivos ubicados en otros lugares. Como las WAN conectaban redes de usuarios dentro de un área geográfica extensa, permitieron que las empresas se comunicaran entre sí a través de grandes distancias. Como resultado de la interconexión de los computadores, impresoras y otros dispositivos en una WAN, las empresas pudieron comunicarse entre sí, compartir información y recursos, y tener acceso a Internet.
Dispositivos de WAN básicos

Las WAN utilizan varios tipos de dispositivos, incluyendo los siguientes:

- Routers, que ofrecen varios servicios, entre ellos puertos de interfaz LAN y WAN.
- Switches WAN, que se conectan al ancho de banda de las WAN para la comunicación de voz, datos y vídeo.

- Módems, que hacen interfaz con los servicios de grado de voz. Los módems incluyen los dispositivos CSU/ DSU y TA/NT1 que hacen interfaz con los servicios RDSI.
- Servidores de comunicación, que concentran las comunicaciones de usuarios de discado entrante y discado saliente.

Tecnologías WAN

Servicios de conmutación de circuitos

- *POTS (Servicio telefónico analógico):* No es un servicio informático de datos, pero se incluye por dos motivos: (1) muchas de sus tecnologías forman parte de la creciente infraestructura de datos, (2) es un modelo sumamente confiable, de fácil uso para una red de comunicaciones de área amplia; el medio típico es el cable de cobre de par trenzado
- *RDSI (Red Digital de Servicios Integrados) de banda angosta*: Una tecnología versátil, de amplio uso e históricamente importante. Fue el primer servicio con marcación totalmente digital. Es de uso bastante generalizado, aunque varía considerablemente de un país a otro. El costo es moderado. El ancho de banda máximo es de 128 kbps para la BRI (Interfaz de Acceso Básico) de menor costo y de aproximadamente 3 Mbps para la PRI (Interfaz de Acceso Principal). El medio típico es el cable de cobre de par trenzado

Servicios de conmutación por paquetes

- *X.25*: Tecnología más antigua pero todavía ampliamente utilizada, que posee amplias capacidades de verificación de errores desde la época en que los enlaces de las WAN eran más susceptibles a los errores, lo que hace que su confiabilidad sea muy grande, pero al mismo tiempo limita su ancho de banda. El ancho de banda puede ser de 2 Mbps como máximo. Es ampliamente utilizada, y su costo es moderado. El medio típico es el cable de cobre de par trenzado
- *Frame Relay*: Versión conmutada por paquetes del RDSI de banda angosta. Se ha transformado en una tecnología de WAN sumamente popular por derecho propio. Es más eficiente que X.25, con servicios similares. El ancho de banda máximo es de 44,736 Mbps. En los EE.UU. son muy populares los anchos de banda de 56kbps y 384kbps. Es de uso generalizado, el costo es de moderado a bajo. Entre los medios típicos se incluyen el cable de cobre de par trenzado y el cable de fibra óptica

Servicios de conmutación por celdas

- *ATM (Modo de Transferencia Asíncrona):* Tiene una cercana relación con el RDSI de banda ancha. Es una tecnología de WAN (e inclusive de LAN) cuya importancia va en aumento. Utiliza tramas pequeñas, de longitud fija (53 bytes) para transportar los datos. El ancho de banda máximo es actualmente de 622 Mbps, aunque se están desarrollando velocidades mayores. Los medios típicos son el cable de cobre de par trenzado y el cable de fibra óptica. Su uso es generalizado y está en aumento; el costo es elevado.
- *SMDS (Servicio de datos multimegabit conmutado):* Relacionado con ATM y utilizado normalmente en las MAN. El ancho de banda máximo es de 44,736 Mbps. Los medios típicos son el cable de cobre de par trenzado y el cable de fibra óptica. No es de uso común: el costo es relativamente alto.

Servicios digitales dedicados

- *T1, T3, E1, E3:* La serie T de servicios en los EE.UU. y la serie E de servicios en Latinoamérica y Europa son tecnologías de WAN sumamente importantes. Usan la multiplexación por división de tiempo para "dividir" y asignar ranuras de tiempo para la transmisión de datos; el ancho de banda es:
    - T1: 1,544 Mbps
    - T3: 44,736 Mbps
    - E1: 2,048 Mbps
    - E3: 34,368 Mbps
    - Hay otros anchos de banda disponibles

Los medios utilizados son normalmente el cable de cobre de par trenzado y el cable de fibra óptica. Su uso es muy generalizado; el costo es moderado.

- *xDSL (DSL por Digital Subscriber Line (Línea Digital del Suscriptor) y x por una familia de tecnologías)*: Tecnología de WAN nueva y en desarrollo para uso doméstico. Su ancho de banda disminuye a medida que aumenta la distancia desde el equipo de las compañías telefónicas. Las velocidades máximas de 51,84 Mbps son posibles en las cercanías de una central telefónica; son más comunes los anchos de banda mucho menores (desde 100 kbps hasta varios Mbps). Su uso es limitado pero en rápido aumento; el costo es moderado y se reduce cada vez más. x indica toda la familia de tecnologías DSL, entre ellas:
  - *HDSL*: DSL de alta velocidad de bits
  - *SDSL*: DSL de línea única
  - *ADSL*: DSL asimétrica
  - *VDSL*: DSL de muy alta velocidad de bits
  - *RADSL*: DSL adaptable a la velocidad
- *SONET (Red Óptica Síncrona)*: Conjunto de tecnologías de capa física de muy alta velocidad, diseñadas para cables de fibra óptica, pero que también pueden funcionar con cables de cobre. Tiene una serie de velocidades de datos disponibles con designaciones especiales. Implementadas a diferentes niveles de OC (portadora óptica) desde los 51,84 Mbps (OC-1) hasta los 9,952 Mbps (OC-192). Puede alcanzar estas impresionantes velocidades de datos mediante el uso de multiplexación por división de longitud de onda (WDM), en la que láseres configurados para colores ligeramente diferentes (longitudes de onda) envían enormes cantidades de datos ópticamente; su uso es generalizado entre las entidades de backbone de Internet. El costo es elevado: no es una tecnología que se pueda usar a nivel doméstico.

Otros servicios de WAN

- *Módems de discado (conmutación analógica)*: Su velocidad es limitada, pero son muy versátiles. Funcionan con la red telefónica existente. El ancho de banda máximo aproximado es de 56 kbps. El costo es bajo. Su uso es muy generalizado. El medio típico es la línea telefónica de par trenzado
- *Módems por cable (analógico compartido)*: Colocan señales de datos en el mismo cable que las señales de televisión. Es cada vez más popular en regiones donde hay gran cantidad de cable coaxial de TV instalado (90% de los hogares en los EE.UU.). El ancho de banda máximo disponible puede ser de 10 Mbps, aunque esto se degrada a medida que más usuarios se conectan a un segmento determinado de la red (comportándose como LAN no conmutadas). El costo es relativamente bajo. Su uso es limitado pero está en aumento. El medio es cable coaxial.
- *Inalámbrico*: No se necesita un medio porque las señales son ondas electromagnéticas. Existen varios enlaces de WAN inalámbricos, dos de los cuales son:
  - Terrestre: Anchos de banda normalmente dentro del intervalo de Mbps (por ej., microondas). El costo es relativamente bajo. Normalmente se requiere línea de vista. El uso es moderado.
  - *Satélite*: Puede servir a los usuarios móviles (por ej., red telefónica celular) y usuarios remotos (demasiado alejados de las instalaciones de cables). Su uso es generalizado. El costo es elevado.

2.3.4 Internet

Internet es, desde el punto de vista técnico y por encima de todas las demás posibles definiciones, una "red de redes". Su origen fue realmente la conexión de diversas redes, públicas y privadas, con el objetivo de potenciar la conectividad a nivel mundial y facilitar la transmisión de datos.

En un principio los usuarios de estas redes, que terminaron convirtiéndose en una, eran universidades, laboratorios tecnológicos implicados en el desarrollo de la propia red, centros gubernamentales y, por supuesto, el ejército (todo ello en un principio en territorio norteamericano)

Dos son los hechos más relevantes en la historia de Internet, en primer lugar la intención del gobierno norteamericano, y como consecuencia de la situación de guerra fría que se vivía en el año 1957 (los soviéticos acababan de poner en órbita el Sputnik), de crear una red de comunicaciones que no pudiese ser inutilizada en su totalidad en caso de destrucción de uno de sus nodos principales, esto llevó a la formulación de la "teoría de conmutación de paquetes de información" de Kleinrock, gracias a la cual se envía hoy en día la información a través de Internet.

La primera red diseñada con este propósito en 1969 por Bolt, Beranek y Newman para la recién creada Agencia de Proyectos de Investigación Avanzados (ARPA) se denominó Arpanet.

El segundo hecho importante fue el desarrollo de lo que conocemos como WWW (World Wide Web) por parte de Tim Berners-Lee en el laboratorio de física de alta energía del CERN (Centro Europeo de Investigación Nuclear) en el año 1989. Su intención era la de encontrar la forma de poder compartir la información de forma global utilizando una red, su primer intento se denominó "proyecto de hipertexto" y en el verano de 1991 la web tal como la conocemos hoy en día ya era una realidad.

Este concepto de hipertexto, enunciado ya en los años 60 por Ted Nelson, hace referencia a la conexión entre textos que permite pulsar sobre una palabra enlazada para obtener información adicional (y en general entre documentos como textos, imágenes, sonido o vídeo, que conocemos como "hipermedia")

Las entidades típicas que controlan Internet son:

- *empresas* (por ej., MCI Worldcom, Sprint, AT&T, Qwest, UUNet, France Telecom)
- *universidades* (por ej., Universidad de Illinois, Universidad Stanford)
- *institutos de investigación* (por ej., CERN en Suiza)
- *Proveedores de servicios de Internet (ISP)*

La red de Internet es del tipo Cliente-Servidor, esto significa que todos los datos están almacenados en ordenadores denominados servidores y que los sirven, mediante la correspondiente conexión, a los equipos cliente.

2.3.4.1 Protocolo TCP/IP de Internet y el modelo OSI

El conjunto de protocolos Protocolo de Control de Transmisión/Protocolo Internet (TCP/IP) se desarrolló como parte de la investigación realizada por la Agencia de Proyectos de Investigación Avanzada para la Defensa (DARPA). Originalmente, se desarrolló para suministrar comunicaciones a través de DARPA. Posteriormente, TCP/IP se incluyó en la Distribución del Software Berkeley de UNIX. TCP/IP es hoy el estándar de facto para las comunicaciones de internetwork y sirve como el protocolo de transporte para Internet, permitiendo que millones de computadores se comuniquen a nivel mundial.

TCP/IP es muy importante porque:

- TCP/IP es un protocolo disponible a nivel mundial.
- TCP/IP es una referencia útil para comprender otros protocolos porque incluye elementos que son representativos de otros protocolos.
- TCP/IP es importante porque el router lo utiliza como una herramienta de configuración.

La Figura II.11 muestra la diferencia entre los modelos TCP/IP y OSI.

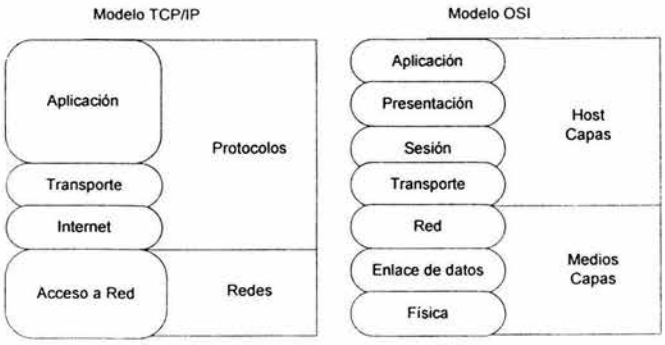


Figura II.11 Modelo TCP vs Modelo OSI

Capa de Aplicación
La capa de aplicación soporta los protocolos de direccionamiento y la administración de red. Adicionalmente tiene protocolos para transferencia de archivos, correo electrónico y conexión remota.

DNS (Sistema de denominación de dominio) es un sistema utilizado en Internet para convertir los nombres de los dominios y de sus nodos de red publicados abiertamente en direcciones.

WINS (Servicio de denominación para Internet de Windows) es un estándar desarrollado para Windows NT de Microsoft que asocia las estaciones de trabajo NT con los nombres de dominio de Internet de forma automática.

HOSTS es un archivo creado por los administradores de red que se mantiene en los servidores. Se utiliza para suministrar asignación estática entre direcciones IP y nombres de computadores.

POP3 (Protocolo de la oficina de correos) es un estándar de Internet para almacenar correo electrónico en un servidor de correo hasta que se pueda acceder a él y descargarlo a la computadora. Permite que los usuarios reciban correo desde sus buzones de entrada utilizando varios niveles de seguridad.

SMTP (Protocolo de transferencia de correo simple) maneja la transmisión de correo electrónico a través de las redes informáticas. No suministra otro soporte para la transmisión de datos más que texto simple.

SNMP (Protocolo de administración de red simple) es un protocolo que suministra un medio para monitorear y controlar dispositivos de red, y para administrar configuraciones, recolección de estadísticas, desempeño y seguridad.

FTP (Protocolo de transferencia de archivos) es un servicio confiable orientado a conexión que utiliza TCP para transferir archivos entre sistemas que soportan FTP. Soporta transferencias bidireccionales de archivos binarios y archivos ASCII.

TFTP (Protocolo de transferencia de archivos trivial) es un servicio no confiable no orientado a conexión que utiliza UDP para transferir archivos entre sistemas que soportan el Protocolo TFTP. Es útil en algunas LAN porque opera más rápidamente que FTP en un entorno estable.

HTTP (Protocolo de transferencia de hipertexto) es el estándar Internet que soporta el intercambio de información en la World Wide Web, así como también en redes internas. Soporta muchos tipos de archivos distintos, incluyendo texto, gráfico, sonido y video. Define el proceso a través del cual los navegadores de la Web originan solicitudes de información para enviar a los servidores de Web.

Capa de transporte
La capa de transporte ejecuta dos funciones: control de flujo, que se suministra a través de las ventanas deslizantes, y confiabilidad, que se suministra a través de los números de secuencia y los acuses de recibo.

La capa de transporte también proporciona dos protocolos:

- *TCP*: un protocolo confiable, orientado a conexión; suministra control de flujo a través de ventanas deslizantes, y confiabilidad a través de los números de secuencia y acuses de recibo. TCP vuelve a enviar cualquier mensaje que no se reciba y suministra un circuito virtual entre las aplicaciones del usuario final. La ventaja de TCP es que proporciona una entrega garantizada de los segmentos.
- *UDP*: protocolo no orientado a conexión y no confiable; aunque tiene la responsabilidad de transmitir mensajes, en esta capa no se suministra ninguna verificación de software para la entrega de segmentos. La ventaja de UDP es la velocidad. Como UDP no suministra acuses de recibo, se envía menos cantidad de tráfico a través de la red, lo que agiliza la transferencia.

Capa de Internet
La capa de Internet de la pila de TCP/IP corresponde a la capa de red del modelo OSI. Cada una de las capas tiene la responsabilidad de transportar paquetes a través de una red utilizando el direccionamiento por software.

Los protocolos que operan en la capa Internet de TCP/IP son:

- *IP* : suministra enrutamiento de datagramas no orientado a conexión, de máximo esfuerzo de entrega; no se ocupa del contenido de los datagramas; busca la forma de desplazar los datagramas al destino
- *ICMP*: aporta capacidad de control y mensajería
- *ARP* : determina direcciones a nivel de capa de enlace de datos para las direcciones IP conocidas
- *RARP* : determina las direcciones de red cuando se conocen las direcciones a nivel de la capa de enlace de datos

2.3.4.2 Herramientas de Internet

World Wide Web (WWW)
La característica diferencial que ha supuesto el éxito rotundo de las páginas web es su carácter no secuencial, de ahí su nombre de lenguaje de hipertexto, que permite al usuario de un web programado en HTML (HyperText Markup Language) "saltar" de un contenido a otro, utilizando un link (enlace), sin necesidad de pasar por la información contenida entre ambos puntos. La posibilidad, no sólo de enlazar contenidos de un web entre sí, sino, sobre todo, de enlazar distintos webs entre sí es lo que ha supuesto el éxito de Internet como medio de comunicación y consulta de datos. El usuario puede interactuar con la información, que a su vez está enlazada a más información.

Correo electrónico (E-mail)
Sistema de comunicación más conocido como e-mail (electronic mail), permite el intercambio de información entre usuarios.

Listas de correo
Se trata de una aplicación del propio correo electrónico, a través de este medio y mediante suscripción voluntaria del usuario se reciben en el buzón de correo una serie de mensajes de un grupo de personas con intereses comunes.

News
Conocidos como grupos de noticias o newsgroup, en este caso ya no se trata de una tertulia cibernética, como en las listas, sino que es mas bien un tablón de anuncios virtual en el que cada

cual (sin suscripción previa) lee o inserta mensajes, bien como comentario o contestación a otro leído o como una idea lanzada al aire.

Transmisión de ficheros
Internet es una red ("la Red") y como tal debe de poder soportar una actividad fundamental en cualquier tipo de red como es la de compartir e intercambiar ficheros. Para los de pequeño tamaño nos servimos de un sencillo e-mail al que adjuntamos el archivo correspondiente, pero en general podríamos querer transferir ficheros de tamaño considerable, para los cuales el protocolo HTTP no sirve, y usaremos entonces el FTP (File Transfer Protocol), más rápido y fiable.

Chat
En este caso se trata de un tipo de comunicación en tiempo real (síncrona) y en modo texto que permite mantener una conversación entre uno o varios usuarios. Para conectarse es necesario un programa IRC (Internet Relay Chat) cliente (en la máquina del usuario) para conectarse al servidor que gestiona el sistema.

Transmisión de voz
También en tiempo real es posible el envío de voz y fax gracias a la telefonía a través de Internet. Su coste es siempre el de una llamada local.

Videoconferencia
Como ampliación del servicio anterior se puede añadir la imagen en la comunicación telefónica, la mayor dificultad es la calidad de los accesos a Internet actuales para poder soportar el necesario flujo de información que requieren estos dos servicios.

2.3.5 Seguridad

La seguridad en redes es mantener bajo protección los recursos y la información con que se cuenta en la red, a través de procedimientos basados en una política de seguridad tales que permitan el control de lo actuado.

2.3.5.1 Características

- Confidencialidad.- Consiste en proteger la información contra la lectura no autorizada explícitamente. Incluye no sólo la protección de la información en su totalidad, sino también las piezas individuales que pueden ser utilizadas para inferir otros elementos de información confidencial.

- Integridad.- Es necesario proteger la información contra la modificación sin el permiso del dueño. La información a ser protegida incluye no sólo la que está almacenada directamente en los sistemas de cómputo sino que también se deben considerar elementos menos obvios como respaldos, documentación, registros de contabilidad del sistema, tránsito en una red, etc. Esto comprende cualquier tipo de modificaciones:

  - Causadas por errores de hardware y/o software.
  - Causadas de forma intencional.
  - Causadas de forma accidental.

  Cuando se trabaja con una red, se debe comprobar que los datos no fueron modificados durante su transferencia.

- Autenticidad.- En cuanto a telecomunicaciones se refiere, la autenticidad garantiza que quien dice ser "X" es realmente "X". Es decir, se deben implementar mecanismos para verificar quién está enviando la información así como identificar quién la accede, la visualiza, quién ingresa al sistema, ejecuta, procesa, etc.

- No – repudio.- Ni el origen ni el destino en un mensaje deben poder negar la transmisión. Quien envía el mensaje puede probar que, en efecto, el mensaje fue enviado y viceversa.

- Disponibilidad de los recursos y de la información.- De nada sirve la información si se encuentra intacta en el sistema pero los usuarios no pueden acceder a ella. Por tanto, se deben proteger los servicios de cómputo de manera que no se degraden o dejen de estar disponibles a los usuarios de forma no autorizada. La disponibilidad también se entiende como la capacidad de un sistema para recuperarse rápidamente en caso de algún problema.

- Consistencia.- Se trata de asegurar que el sistema siempre se comporte de la forma esperada, de tal manera que los usuarios no encuentren variantes inesperadas.

- Control de acceso a los recursos.- Consiste en controlar quién utiliza el sistema o cualquiera de los recursos que ofrece y cómo lo hace.

- Bitácora.- Consiste en contar con los mecanismos para poder determinar qué es lo que sucede en el sistema, qué es lo que hace cada uno de los usuarios y los tiempos y fechas de dichas acciones.

- Auditoría.- Consiste en verificar que las políticas de seguridad implementadas en un sistema se lleven a cabo.

En cuanto a los dos últimos puntos resulta de extrema importancia, cuando se trata de los derechos de los usuarios, diferenciar entre "espiar" y "monitorear" a los mismos. La ética es algo que todo buen administrador debe conocer y poseer.

Finalmente, todos estos servicios de seguridad deben ser tomados en cuenta en el momento de elaborar las políticas y procedimientos de una organización para evitar pasar por alto cuestiones importantes como las que señalan dichos servicios. De esta manera, es posible sentar de forma concreta y clara los derechos y límites de usuarios y administradores. Sin embargo antes de realizar cualquier acción para lograr garantizar estos servicios, es necesario asegurarnos que los usuarios conozcan sus derechos y obligaciones (es decir, las políticas), de tal forma que no se sientan agredidos por los procedimientos organizacionales.

2.3.5.2 Acceso de Red

Hay dos tipos de método de seguridad para proteger la red:

- Seguridad en el nivel compartido.
- Seguridad en el nivel de usuario.

La seguridad en el nivel compartido, considerada débil y difícil de administrar, permite que los usuarios tengan acceso a determinada información si el administrador de red les asigna una contraseña. Para que una persona tenga acceso a información en la red, debe suministrar una contraseña, que le es asignada de forma específica por el administrador de red.

La seguridad en el nivel del usuario especifica los derechos y privilegios de cada usuario. El administrador de red asigna al usuario una cuenta para acceder a una computadora o una red específica. Cuando alguien intenta conectarse a la red, la computadora compara el identificador de cuenta del usuario y la(s) contraseña(s) con la base de datos de seguridad antes de permitir el acceso al usuario.

Tanto en el modelo de nivel compartido como en el modelo de nivel de usuario, se suministran contraseñas al usuario para que acceda a la red o a datos específicos. Las contraseñas siempre deben ser confidenciales y nunca se deben anotar en lugares donde los usuarios no habilitados puedan encontrarlas.

2.3.5.3 Recuperación de datos

La recuperación de datos, que constituye la segunda parte de la seguridad de red, implica proteger los datos ante pérdidas. Hay varios métodos para evitar la pérdida de datos. Por lo general, hay más de un método en uso al mismo tiempo para proteger los datos. Tres de los métodos populares para la protección de datos son la copia de seguridad de la cinta que contiene los datos, las configuraciones de disco a prueba de fallas y el uso de sistemas de alimentación ininterrumpida (UPS) para evitar que el equipo deje de funcionar cuando se producen interrupciones del suministro eléctrico.

La copia de seguridad de la cinta es el proceso de duplicación de todos los datos almacenados en una cinta magnética. El motivo del uso de cinta es el costo y la capacidad. Los cartuchos de cinta son mucho más baratos y tienen una capacidad de almacenamiento mucho mayor que la de los discos duros extraíbles. La desventaja de la cinta para uso general es que almacena datos de forma secuencial, del mismo modo en que se graba música en un cassette. Esto significa que, así como resulta difícil intentar ubicar una canción en particular en un cassette de forma eficiente, lo mismo sucede cuando se intenta ubicar un archivo determinado en una cinta de datos. Pero, como los datos para una copia de seguridad se graban de forma secuencial y se recuperan del mismo modo, esto no constituye un problema para este tipo de uso.

Es importante realizar una copia de seguridad de la cinta lo más completa y rápidamente que sea posible, dado que puede constituir un desgaste importante sobre los recursos del sistema (ancho de banda de la red y alimentación del procesador del servidor). Para permitir que toda la copia de seguridad se realice del modo más eficiente posible, se han desarrollado distintos tipos de copias de seguridad. La mayoría de los tipos de copia de seguridad trabajan con un señalador o switch denominado bit de Archivo. El bit de archivo se guarda con un archivo y se activa siempre que ese archivo se crea o se modifica. Este señalador le indica al proceso de copia de seguridad si se debe realizar una copia de seguridad del archivo o no. Si se guarda un archivo en cinta durante el proceso de copia de seguridad, por lo general, el señalador se desactiva, indicando que el archivo actual está en la cinta.

La mayoría de las empresas recomiendan que las cintas y las copias de seguridad se guarden en una caja de seguridad ignífuga, o que se retiren del edificio en caso de daños debidos a incendios o inundaciones

El siguiente método para proteger los datos es con dispositivos de almacenamiento con tolerancia a las fallas. Este tipo de conjunto redundante de dispositivos es categorizada por los niveles 0-5 de RAID (Matrices redundantes de discos económicos). Estos niveles se describen a continuación:

1. **RAID 0** Obtiene datos de múltiples discos, sin paridad, de modo que no hay redundancia.

2. **RAID 1** Copia exacta de disco (dúplex de disco) escribe los datos en dos particiones idénticas en discos duros individuales, creando de este modo una copia de seguridad automática. El dúplex de disco utiliza dos tarjetas controladoras de disco duro así como también dos discos duros para evitar que la tarjeta controladora sea el único punto de falla para el sistema como ocurre en el caso de la copia exacta de disco.

3. **RAID 2** Escribe datos a través de múltiples discos duros, con verificación de errores. Este sistema se ha dejado de usar porque requiere modificaciones sumamente costosas del disco para que funcione.

4. **RAID 3** Obtiene datos de un byte a la vez y tiene una unidad de paridad dedicada. Una elección de redundancia buena pero costosa. Dado que es sumamente cara, esta solución tampoco se utiliza demasiado a menudo.

5. **RAID 4** Obtiene datos de un sector a la vez y tiene una unidad de paridad dedicada. Una elección de redundancia costosa que es muy lenta en lo que se refiere a la escritura de datos en el disco. Dado que es sumamente cara y es muy lenta en lo que se refiere a la escritura, esta solución tampoco se utiliza demasiado a menudo.

6. **RAID 5** Obtiene datos y paridad a través de múltiples discos (por lo menos tres para RAID 5). Al mezclar la paridad en todos los discos, no se requiere un disco de paridad individual y aún así se obtiene redundancia de datos total. La escritura de datos en el disco sigue siendo lenta, pero el costo no es tan elevado. Otro de los factores importantes acerca de RAID 5 es que en un sistema Windows NT las particiones *de arranque* y *de sistema* no se pueden ubicar en una matriz de disco RAID 5.

2.3.5.4 Virus Informáticos

Un gusano es un programa que se propaga a sí mismo de una computadora a otra, generalmente creando copias de sí mismo en la memoria de cada computadora. Un gusano se puede duplicar a sí mismo en una computadora con tanta frecuencia que hace que la computadora entre en colapso. A veces escrito en "segmentos" individuales, un gusano se introduce inadvertidamente en un sistema host o de red, ya sea por "diversión" o con la intención de dañar o destruir información. El término proviene de una novela de ciencia ficción y generalmente se ha reemplazado por el término virus.

Un Virus es un programa que "infecta" los archivos de la computadora (generalmente otros programas ejecutables) al introducir en esos archivos copias de sí mismo. Esto generalmente se hace de forma tal que las copias se ejecutan cuando el archivo se carga en la memoria, permitiendo que infecten otros archivos, y así sucesivamente. Los virus a menudo tienen efectos colaterales nocivos, a veces intencionalmente, otras veces sin intención. La variante más reciente es enviar esos virus a través de Internet en forma de mensajes adjuntos del correo electrónico.

Un Troyano es un programa destructivo que simula ser un juego, una utilidad o una aplicación. Cuando se ejecuta, el Troyano provoca algún tipo de daño en el sistema informático, aparentando, sin embargo, hacer algo útil.

Los tres párrafos anteriores describen ciertos tipos de software que pueden dañar la red o la computadora. Todos tratamos de evitar que estas cosas ocurran con nuestros computadores, ya sea un daño físico o un daño del software.

2.3.5.5 Firewall

El firewall se utiliza para proteger la red interna de Internet, que es pública y poco segura. Los firewall se pueden implementar utilizando hardware o software. Un firewall de software es un conjunto de programas en el gateway, que monitorea todo el tráfico que fluye hacia y desde una red y a menudo se implementa utilizando routers configurados de forma específica. Toda la información debe atravesar el firewall y se debe verificar comparándola con un conjunto de normas específico. Si la información no cumple con las normas especificadas, los datos se devuelven y no pueden continuar su camino hasta que cumplan con los estándares establecidos. Un ejemplo de firewall de hardware consiste en utilizar routers configurados de forma específica para controlar el tráfico entrante y saliente.

Cifrado de datos

El cifrado de datos permite la transferencia segura de la información enviada a través de la internetwork. Toma la información escrita en texto sin cifrar y la codifica en un texto denominado texto cifrado, que no se asemeja a ninguna otra cosa, haciéndola ilegible. Cuando se reciben los datos, el texto cifrado se descifra y se vuelve a convertir en el texto original siempre y cuando se trate de un receptor legítimo.

El cifrado de datos se emplea para proporcionar los servicios de seguridad mencionados en la página 49.

2.4 Modelos de Procesos

Para resolver un proyecto de software real, el ingeniero del software debe implementar una estrategia de desarrollo la cual guíe al proceso, métodos y herramientas del producto de software. A dicha estrategia se le conoce como Modelo del proceso o Paradigma de Ingeniería de Software.

Un modelo se utiliza para distinguir un punto de partida, una gestación, un nacimiento, un crecimiento, una obsolescencia y una muerte. Generalmente se selecciona un modelo de proceso para la ingeniería de software según sea la complejidad del proyecto y la aplicación, es decir de acuerdo a los métodos, herramientas y tiempo a utilizarse.

En la actualidad existen algunos modelos de proceso como los que describiré a continuación.

2.4.1 Modelo Lineal Secuencial

Este modelo de proceso, mostrado en la Figura II.12, sugiere un enfoque sistemático, secuencial para el desarrollo de software, donde principia con un modulo de sistemas (Ingeniería de sistemas/ información) y progresa con el análisis, diseño, codificación, pruebas y mantenimiento.

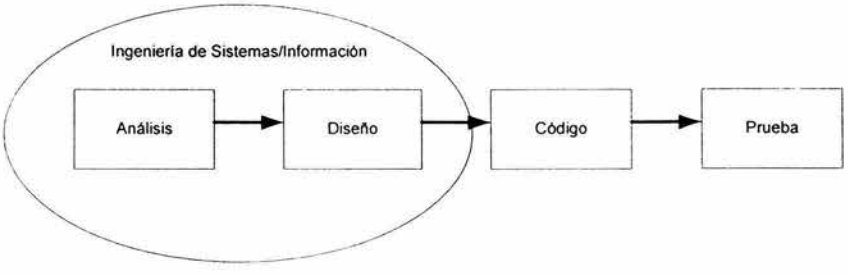


Figura II.12 Modelo lineal secuencial

Análisis de los Requisitos de Software
Tanto el desarrollador como el cliente tienen un papel activo en la ingeniería de requisitos del software, es aquí en esta iteración cliente-desarrollador donde nacen un conjunto de actividades llamadas análisis. El cliente intentará plantear un sistema confuso a nivel de descripción de datos, funciones y comportamiento. El desarrollador debe actuar como interrogador, consultor, es decir como un ente que resuelve problemas y como negociador.

El análisis de requisitos permite al ingeniero de sistemas especificar las características operacionales del software (función, datos y rendimientos), indica la interfaz del software con otros elementos del sistema y establece las restricciones que debe de cumplir el software.

El análisis de requisitos del software se enfoca en cinco áreas de trabajo.

- Reconocimiento del problema
- Evaluación del problema y Síntesis de la solución
- Modelado
- Especificación
- Revisión

Diseño
El diseño del software es un proceso de varios pasos a seguir y se centra en cuatro atributos distintos de programa: estructura de datos, arquitectura de software, representaciones de interfaz y detalle procedimental (algoritmos). El proceso de diseño traduce requisitos en una representación

del software donde se pueda evaluar su calidad antes de que comience la codificación, en la generación de código el diseño se debe traducir en una forma legible por la máquina, si se lleva a cabo el diseño en una forma detallada, la generación de código se realiza mecánicamente.

Código
Se trata de representar en algún lenguaje de programación, el diseño generado para un sistema.

Pruebas
Ya generado el código se comienza con las pruebas del programa, el proceso de prueba se centra en los procesos lógicos internos del software, asegurando que todas las sentencias se han sido verificadas y comprobadas y también en los procesos externos funcionales, es decir realizar pruebas para la detección de errores y asegurar que la entrada definida produce resultados reales con los resultados requeridos.

Desventajas en la Aplicación de un Modelo de Proceso Lineal
Suele ser muy difícil que un proyecto real siga el modelo lineal ya que como es un modelo secuencial no permite mucha interacción ya que para el buen funcionamiento de este modelo se tienen que tener bien definidos los requerimientos del sistema ya que si no es así los posibles cambios al sistema de software pueden causar confusión al desarrollador o al equipo de proyecto de software lo que generaría perdidas en costos y tiempo al proyecto, y lo que es una realidad tangible es que a menudo el cliente no expone explícitamente todos los requerimientos y suele haber siempre algunos cambios para mejorar el sistema aún cuando este ha sido terminado, hay una etapa de prueba donde suelen destacar nuevas necesidades del cliente, y si el ingeniero de software no selecciono bien el modelo de proceso a implementar en el proyecto sería muy complicado rediseñar sistemas con modelos que no permitieran evolucionar el sistema realizado.

2.4.2 Modelo de Construcción de Prototipos

Generalmente el cliente define un conjunto de objetivos generales para el software, pero no identifica los requisitos detallados de entrada, proceso o salida, en estos casos el desarrollador de software suele no estar seguro de la funcionalidad en el desarrollo de un sistema informático y en estas situaciones un paradigma de construcción de prototipos puede ofrecer el mejor enfoque.El paradigma de construcción de prototipos, Figura II.13, comienza con la recolección de requisitos, el desarrollador y el cliente encuentran y definen los objetivos globales para el proyecto de software.

Como siguiente paso se realiza un diseño rápido en este se realizará una representación de los aspectos del sistema de software que serán visibles para el usuario/cliente. El diseño rápido lleva a la construcción de un prototipo y este será evaluado por el cliente/usuario y se utiliza para refinar los requisitos del software a desarrollar, de tal manera que dicho prototipo le ayudará al desarrollador a entender mejor lo que necesita hacer.

La construcción de prototipos también puede ser problemática por las siguientes razones:

El cliente ve lo que parece ser una versión de trabajo del proyecto de software, sin tener conocimiento de que el prototipo es un producto que guía y puede servir como primer sistema, pero que se recomienda tirar ya que el cliente no sabe que con la prisa de hacer que funcione dicho prototipo no se tiene en cuenta la calidad global del software o la facilidad de mantenimiento a largo plazo. Generalmente cuando se le informa al cliente de que el proyecto se debe construir nuevamente para que se puedan mantener los niveles altos de calidad, el cliente no lo entiende y pide ajustes nuevos o nuevas necesidades, y frecuentemente estas nuevos requerimientos hacen de la gestión de desarrollo del software muy lento.

El desarrollador, a menudo, hace compromisos de implementación para hacer que el prototipo funcione rápidamente esto podría llevar al ingeniero de software a utilizar herramientas de software inadecuadas para el producto.

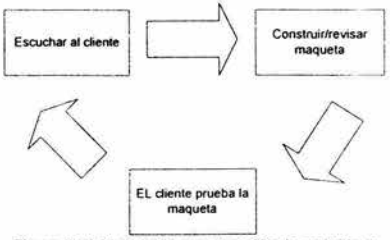


Figura II.13 Modelo de construcción de prototipos

2.4.3 Modelo DRA

El Desarrollo Rápido de Aplicaciones (DRA) es un modelo de proceso del desarrollo del software lineal secuencial que enfatiza un ciclo de desarrollo extremadamente corto. Si se definen y se comprenden bien los requisitos y se limita el ámbito del proyecto de software, el proceso DRA permite al desarrollador o al equipo de desarrollo crear un sistema completamente funcional dentro de un periodo de 60 a 90 días. La Figura II.14 muestra este modelo.

El enfoque del modelo de desarrollo rápido de aplicaciones comprende las siguientes fases:

Modelado de Gestión: El flujo de información entre las funciones de gestión se modela de forma que se responda las siguientes preguntas; ¿Qué información conduce el proceso de gestión?, ¿Quién la genera?. ¿ A dónde va la información?, ¿Quién la procesa?.

Modelado de Datos: El flujo de información definido en la fase de modelado de gestión se refina como un conjunto de datos necesarios para apoyar al proyecto en donde se definen las características llamadas atributos de cada uno de los objetos o entidades y se establecen las relaciones entre estas entidades u objetos.

Modelado del Proceso: Las entidades u objetos de datos definidos en la fase de modelado de datos quedan transformados para lograr el flujo de información necesario par implementar una función de gestión, las descripciones se crean para añadir, modificar, suprimir, o recuperar datos.

Generación de Aplicaciones: El DRA asume la utilización de técnicas de cuarta generación.

Inconvenientes del modelo DRA:
Para proyectos grandes aunque por escalas o módulos, DRA requiere recursos humanos suficientes como para crear el número correcto de equipos DRA.

El modelo DRA requiere clientes y desarrolladores comprometidos en las rápidas actividades necesarias para completar un sistema de software en un marco de tiempo abreviado, de lo contrario el DRA fracasará. No todos los tipos de aplicaciones son apropiados para el modelo DRA, si un sistema no se puede modularizar adecuadamente, la construcción de los componentes necesarios para DRA será problemático.

DRA no es adecuado cuando los riesgos técnicos son altos, esto ocurre cuando una nueva aplicación hace uso de tecnologías nuevas, o cuando el software nuevo requiere un alto grado de interoperatividad con programas de computadora ya existentes.

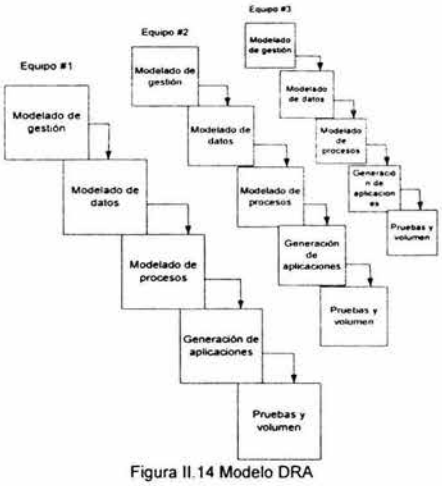


Figura II.14 Modelo DRA

2.4.4 Modelo Incremental

El modelo incremental combina elementos del modelo lineal secuencial (aplicados repetidamente) con la filosofía interactiva de construcción de prototipos. Cada secuencia lineal produce un incremento, teniendo en cuenta que el flujo del proceso de cualquier incremento puede incorporar el paradigma de construcción de prototipos. En la Figura II.15 se muestra este modelo. Cuando se utiliza un modelo incremental, el primer incremento, a menudo, es un producto esencial, es decir se afrontan requisitos básicos, pero muchas funciones suplementarias algunas conocidas, otras no, quedan sin extraer. El cliente utiliza el producto central o base, ya que se trabaja con un modelo incremental el desarrollador desarrolla un plan para el incremento siguiente, el plan afronta la modificación del producto central a fin de cumplir mejor las necesidades del cliente y la entrega de las funciones. Este proceso se repite siguiendo la entrega de cada incremento, hasta que se elabore el producto final de software.

El modelo incremental se centra en la entrega de un producto operacional con cada incremento, los primeros incrementos son versiones incompletas del producto final, pero proporciona al usuario la funcionabilidad que precisa y también una plataforma para la evaluación.

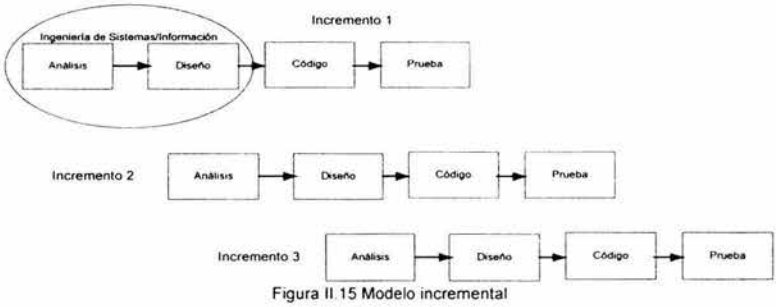


Figura II.15 Modelo incremental

2.4.5 Modelo Espiral

El modelo en espiral, el cual se puede apreciar en la Figura II.16, es un modelo de proceso de software evolutivo que conjuga la naturaleza iterativa de construcción de prototipos con los aspectos controlados y sistemáticos del modelo lineal secuencial, proporciona el potencial para el desarrollo rápido de versiones incrementales del software.

En el modelo espiral, el software se desarrolla en una serie de versiones incrementales, durante las primeras iteraciones, la versión incremental podría ser un modelo en papel o un prototipo y durante las últimas iteraciones se producen versiones cada vez más completas del sistema diseñado.

El modelo en espiral se divide en un número de actividades de marco de trabajo, también llamadas regiones de tareas, a continuación se mencionará cada una de ellas:

- Comunicación con el cliente: las tareas requeridas para establecer comunicación entre el desarrollador y el cliente.
- Planificación: las tareas requeridas para definir recursos, el tiempo y otra información relacionadas con el proyecto.
- Análisis de riesgos: las tareas requeridas para evaluar riesgos técnicos y de gestión.
- Ingeniería: las tareas requeridas para construir una o más representaciones de la aplicación.
- Construcción y acción: las tareas requeridas para construir, probar, instalar y proporcionar soporte al usuario (documentación y práctica).
- Evaluación del cliente: las tareas requeridas para obtener la reacción del cliente según la evaluación de las representaciones del software creadas durante la etapa de ingeniería e implementada durante la etapa de instalación.

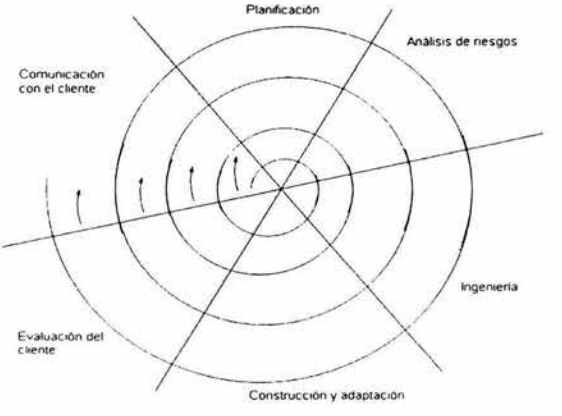


Figura II.16 Modelo espiral

Estas regiones de tareas se adaptan a las características del proyecto del software que va a emprenderse, para productos pequeños, el número de tareas de trabajo son bajas y en lo contrario para productos grandes cada región de tareas de trabajo se definen para lograr un nivel más alto de formalidad, en ambos casos se aplican los requisitos de protección y seguridad de software, por ejemplo; la gestión de configuración del software y garantía de calidad del software.

Cuando el ingeniero de software elige implementar para su desarrollo del producto de software un modelo de proceso espiral debe tener en cuenta que este proceso evolutivo comienza en el centro de la espiral y las regiones de trabajo, en la espiral giran en la dirección de las agujas del reloj.

El modelo en espiral puede adaptarse y aplicarse a lo largo de la vida del software de una computadora.

2.5 Metodologías de Ingeniería del software

2.5.1 Definición

Una metodología de Ingeniería de software es un conjunto integrado de Métodos, Técnicas y Herramientas que cubren más de una etapa del ciclo de vida de los sistemas de información. Lo anterior se muestra en la Figura II.17.

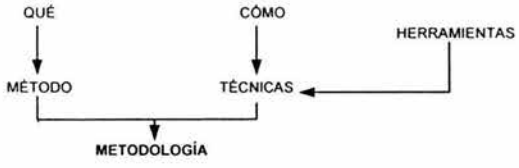


Figura II.17 Metodología de Ingeniería de Software

Una metodología:

- Ofrece un marco y un vocabulario común para el equipo de trabajo
- Sirve de guía en la utilización de las distintas técnicas y herramientas
- Ayuda a comprobar la calidad del producto final y el seguimiento de los proyectos
- Resuelve mucho de los problemas y necesidades actuales existentes en el desarrollo de aplicaciones
- Para solucionar los problemas de los sistemas de información de una empresa no es suficiente aplicar soluciones parciales (B.D., 4GL,...) si no que se precisan enfoques globales

2.5.2 Introducción.

Desarrollo convencional

- Los resultados finales son impredecibles
- No hay forma de controlar lo que está sucediendo en el Proyecto
- Los cambios organizativos afectan negativamente al proceso de desarrollo

Desarrollo estructurado

- Programación estructurada
- Diseño estructurado
- Análisis estructurado
- Especificaciones funcionales:
- Gráficas
- Particionadas
- Minimamente redundantes

Desarrollo orientado a objetos

- Trata procesos y datos de forma conjunta.
- Abstracción, ocultación de información y modularidad.
- Las técnicas estructuradas han influido en estas metodologías.

En la Tabla II.1 se aprecia las principales metodologías de Ingeniería de Software.

2.5.3 Características principales.

Impacto de la metodología en el entorno de desarrollo

En la Figura II.18 se muestran las características que se deben tomar en cuenta en una metodología de desarrollo de software.

Características deseables de una metodología

- Existencia de reglas predefinidas
- Cobertura total del ciclo de desarrollo
- Verificaciones intermedias
- Planificación y control
- Comunicación efectiva
- Utilización sobre un abanico amplio de proyectos
- Fácil formación
- Herramientas CASE
- Actividades que mejoren el proceso de desarrollo
- Soporte al mantenimiento
- Soporte de la reutilización de software

AÑO	METODOLOGIA
1968	Conceptos sobre la programación estructurada de DIJKSTRA
1974	Técnicas de programación estructurada de WARNIER y JACKSON
1975	Primeros conceptos sobre diseño estructurado de MYERS y YOURDON
1977	Primeros conceptos sobre análisis estructurado GANE y SARSON
1978	Análisis estructurado: DEMARCO y WEINBERG Nace MERISE
1981	SSADM (versión inicial) Information Engineering (versión inicial)
1985	Análisis y diseño estructurado para sistemas de tiempo real de WARD y MELLOR Tecnologías CASE integradas
1986	SSADM versión 3
1987	Análisis y diseño estructurado para sistemas de tiempo real de HATLEY y PIRHRAY
1989	METRICA (versión inicial)
1990	SSADM versión 4 Diseño orientado a objetos Metodologías de desarrollo orientadas a objetos
1993	METRICA versión 2
1995	METRICA versión 2.1 Primer intento de estandarización de los modelos orientados a objetos UML
1998	METRICA versión 3
2000	Consolidación de UML

Tabla II.1 Metodologías de Ingeniería de Software

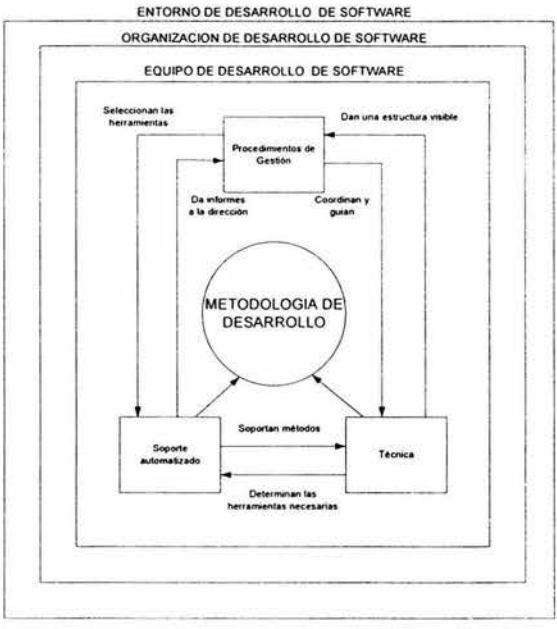


Figura II.18 Características de la Metodología de Ingeniería de Software

2.5.4 Clasificación de las metodologías.

En la Tabla II.2 se muestra los dos tipos de metodologías de ingeniería de software: Estructurada y Orientada a objetos.

ENFOQUE	TIPO DE SISTEMA	FORMALIDAD
Estructuradas • Orientadas a Procesos • Orientadas a datos - Jerárquicos - No jerárquicos • Mixtas	Gestión	No formal
Orientadas a objetos	Tiempo real	Formal

Tabla II.2 Clasificación de las Metodología de Ingeniería de Software

Metodologías orientadas a procesos

Especificación estructurada:

- Diagramas de Flujo de Datos
- Diccionario de Datos
- Especificaciones de procesos

La Tabla II.3 muestra la comparación entre el Método de DeMarco y el de Gane y Sarson, ambos orientados a procesos.

FASES DEL ANALISIS ESTRUCTURADO	
Método de DeMarco	**Método de Gane y Sarson**
1. Construir el modelo físico actual (DFT físico actual) 2. Construir el modelo lógico actual (DFD lógico actual) 3. Crear un conjunto de modelos físicos alternativos 4. Estimar los costes y tiempos de cada opción 5. Seleccionar un modelo 6. Empaquetar la especificación	1. Construir el modelo lógico actual (DFT lógico actual) 2. Construir el modelo del nuevo sistema: elaborar una especificación estructurada y construir un modelo lógico de datos de tercera forma normal que exprese el contenido de los almacenes de datos 3. Seleccionar un modelo lógico 4. Crear el nuevo modelo físico del sistema 5. Empaquetar la especificación

Tabla II.3 Metodologías Orientadas a Procesos

Metodología de Yourdon/Constantine

- Realizar los DFD del sistema
- Realizar el diagrama de estructuras
- Evaluar el diseño
- Preparar el diseño para la implantación

Metodologías orientadas a datos jerárquicos

- La estructura de control del programa debe ser jerárquica y se debe derivar de la estructura de datos del programa
- El proceso de diseño consiste en definir primero las estructuras de los datos de entrada y salida, mezclarlas todas en una estructura jerárquica de programa y después ordenar detalladamente la lógica procedimental para que se ajuste a esta estructura
- El diseño lógico debe preceder y estar separado del diseño físico

Metodologías orientadas a datos no jerárquicos

Metodología Ingeniería de la Información

- Planificación: construir una arquitectura de la Información y una estrategia que soporte los objetivos de la organización
- Análisis: comprender las áreas del negocio y determinar los requisitos del sistema
- Diseño: establecer el comportamiento del sistema deseado por el usuario y que sea alcanzable por la tecnología
- Construcción: construir sistemas que cumplan los tres niveles anteriores

Metodologías orientadas a objetos

- "Revolucionarios" o "puros"
- "Sintetistas" o "evolutivos"

Metodologías para sistemas de tiempo real

- Manejo de interrupciones
- Comunicación y sincronización entre tareas
- Gestión de procesos concurrentes
- Respuesta oportuna ante eventos externos
- Datos continuos o discretos

2.5.5 METODOLOGÍA ORIENTADA A OBJETOS: UML

UML (Lenguaje Unificado de Modelado) es una especificación de notación orientada a objetos. Se basa en las anteriores especificaciones BOOCH, RUMBAUGH y COAD-YOURDON. Divide cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto. Estos diagramas juntos son los que representa la arquitectura del proyecto.

Con UML nos debemos olvidar del protagonismo excesivo que se le da al diagrama de clases, este representa una parte importante del sistema, pero solo representa una vista estática, es decir muestra al sistema parado. Sabemos su estructura pero no sabemos que le sucede a sus diferentes partes cuando el sistema empieza a funcionar. UML introduce nuevos diagramas que representa una visión dinámica del sistema. Es decir, gracias al diseño de la parte dinámica del sistema podemos darnos cuenta en la fase de diseño de problemas de la estructura al propagar errores o de las partes que necesitan ser sincronizadas, así como del estado de cada una de las instancias en cada momento. El diagrama de clases continua siendo muy importante, pero se debe tener en cuenta que su representación es limitada, y que ayuda a diseñar un sistema robusto con partes reutilizables, pero no a solucionar problemas de propagación de mensajes ni de sincronización o recuperación ante estados de error.

UML también intenta solucionar el problema de propiedad de código que se da con los desarrolladores, al implementar un lenguaje de modelado común para todos los desarrollos se crea una documentación también común, que cualquier desarrollador con conocimientos de UML será capaz de entender, independientemente del lenguaje utilizado para el desarrollo.

UML es ahora un estándar, no existe otra especificación de diseño orientado a objetos, ya que es el resultado de las tres opciones existentes en el mercado. Su utilización es independiente del lenguaje de programación y de las características de los proyectos, ya que UML ha sido diseñado para modelar cualquier tipo de proyectos, tanto informáticos como de arquitectura, o de cualquier otro ramo.

UML permite la modificación de todos sus miembros mediante estereotipos y restricciones. Un estereotipo nos permite indicar especificaciones del lenguaje al que se refiere el diagrama de UML. Una restricción identifica un comportamiento forzado de una clase o relación, es decir mediante la restricción estamos forzando el comportamiento que debe tener el objeto al que se le aplica.

2.5.5.1 Tipos de diagramas

Se dispone de dos tipos diferentes de diagramas los que dan una vista estática del sistema y los que dan una visión dinámica. Los diagramas estáticos son:

- Diagrama de clases: muestra las clases, interfaces, colaboraciones y sus relaciones. Son los más comunes y dan una vista estática del proyecto.
- Diagrama de objetos: Es un diagrama de instancias de las clases mostradas en el diagrama de clases. Muestra las instancias y como se relacionan entre ellas. Se da una visión de casos reales.
- Diagrama de componentes: Muestran la organización de los componentes del sistema. Un componente se corresponde con una o varias clases, interfaces o colaboraciones.
- Diagrama de despliegue.: Muestra los nodos y sus relaciones. Un nodo es un conjunto de componentes. Se utiliza para reducir la complejidad de los diagramas de clases y componentes de un gran sistema. Sirve como resumen e índice.
- Diagrama de casos de uso: Muestran los casos de uso, actores y sus relaciones. Muestra quien puede hacer que y relaciones existen entre acciones (casos de uso). Son muy importantes para modelar y organizar el comportamiento del sistema.

Lo diagramas dinámicos son:

- Diagrama de secuencia, Diagrama de colaboración: Muestran a los diferentes objetos y las relaciones que pueden tener entre ellos, los mensajes que se envían entre ellos. Son dos diagramas diferentes, que se puede pasar de uno a otro sin perdida de información, pero que nos dan puntos de vista diferentes del sistema. En resumen, cualquiera de los dos es un Diagrama de Interacción.
- Diagrama de estados: muestra los estados, eventos, transiciones y actividades de los diferentes objetos. Son útiles en sistemas que reaccionen a eventos.
- Diagrama de actividades: Es un caso especial del diagrama de estados. Muestra el flujo entre los objetos. Se utilizan para modelar el funcionamiento del sistema y el flujo de control entre objetos.

El número de diagramas es muy alto, en la mayoría de los casos excesivos, y UML permite definir solo los necesarios, ya que no todos son necesarios en todos los proyectos.

Diagrama de casos de uso
Se emplean para visualizar el comportamiento del sistema, una parte de el o de una sola clase. De forma que se pueda conocer como responde esa parte del sistema. El diagrama de uso es muy útil para definir como debería ser el comportamiento de una parte del sistema, ya que solo especifica como deben comportarse y no como están implementadas las partes que define. Por ello es un buen sistema de documentar partes del código que deban ser reutilizables por otros desarrolladores. El diagrama también puede ser utilizado para que los expertos de dominio se comuniquen con los informáticos sin llegar a niveles de complejidad. Un caso de uso especifica un requerimiento funcional, es decir indica esta parte debe hacer esto cuando pase esto.

En el diagrama nos encontramos con diferentes figuras que pueden mantener diversas relaciones entre ellas:

Casos de uso: representado por una elipse, cada caso de uso contiene un nombre, que indique su funcionalidad. Los casos de uso pueden tener relaciones con otros caso de uso. Sus relaciones son:

- Include: Representado por una flecha, en el diagrama de ejemplo podemos ver como un caso de uso, el de totalizar el coste incluye a dos casos de uso.
- Extends: Una relación de una caso de Uso A hacia un caso de uso B indica que el caso de uso B implementa la funcionalidad del caso de uso A.
- Generalization: Es la típica relación de herencia.

Actores: se representan por un muñeco. Sus relaciones son:

- Communicates: Comunica un actor con un caso de uso, o con otro actor.

Parte del sistema (System boundary): Representado por un cuadro, identifica las diferentes partes del sistema y contiene los casos de uso que la forman.

En la Figura II.19 encontramos tres casos de usos Crear producto utiliza Validar producto, y Crear pack productos es una especialización de Crear productos.

Podemos emplear el diagrama de dos formas diferentes, para modelar el contexto de un sistema, y para modelar los requisitos del sistema.

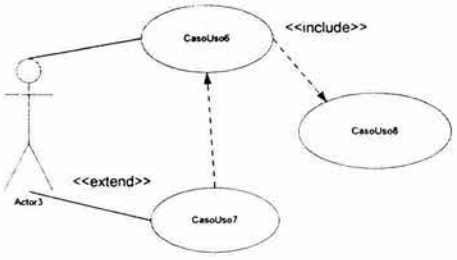


Figura II.19 Caso de uso

Modelado del contexto
Se debe modelar la relación del sistema con los elementos externos, ya que son estos elementos los que forman el contexto del sistema. Los pasos a seguir son:

- Identificar los actores que interactúan con el sistema.
- Organizar a los actores.
- Especificar sus vías de comunicación con el sistema.

La Figura II.20 muestra un ejemplo del modelado de contexto.

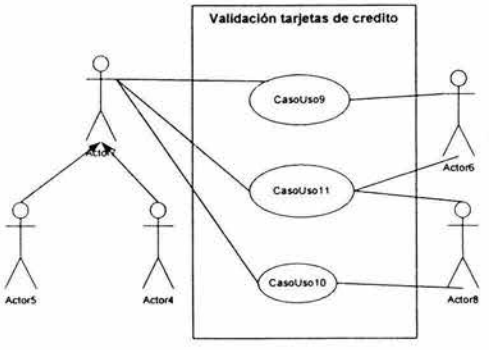


Figura II.20 Modelado del contexto

Modelado de requisitos
La función principal, o la mas conocida del diagrama de casos de uso es documentar los requisitos del sistema, o de una parte de el. Los requisitos establecen un contrato entre el sistema y su exterior, definen lo que se espera que realice el sistema, sin definir su funcionamiento interno. Es el paso siguiente al modelado del contexto, no indica relaciones entre autores, tan solo indica cuales deben ser las funcionalidades (requisitos) del sistema. Se incorporan los casos de uso necesarios que no son visibles desde los usuarios del sistema.

Para modelar los requisitos es recomendable:

- Establecer su contexto, para lo que también podemos usar un diagrama de casos de uso.
- Identificar las necesidades de los elementos del contexto (Actores).
- Nombrar esas necesidades, y darles forma de caso de uso.
- Identificar que casos de uso pueden ser especializaciones de otros, o buscar especializaciones comunes para los casos de uso ya encontrados.

Como podemos ver en la Figura II.21 se incluyen nuevos casos de uso que no son visibles por ninguno de los actores del sistema, pero que son necesarios para el correcto funcionamiento.

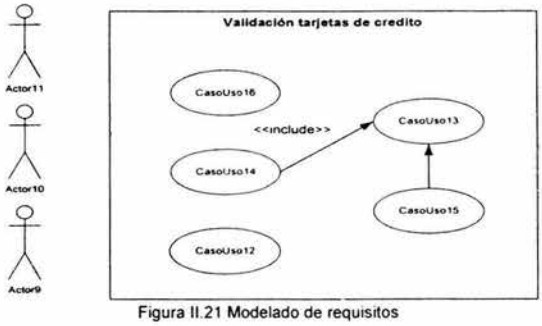


Figura II.21 Modelado de requisitos

Diagrama de clases

Forma parte de la vista estática del sistema. En el diagrama de clases como ya hemos comentado será donde definiremos las características de cada una de las clases, interfaces, colaboraciones y relaciones de dependencia y generalización. Es decir, es donde daremos rienda suelta a nuestros conocimientos de diseño orientado a objetos, definiendo las clases e implementando las ya típicas relaciones de herencia y agregación.

En el diagrama de clases debemos definir a estas y a sus relaciones.

La Clase

Una clase esta representada por un rectángulo que dispone de tres apartados, el primero para indicar el nombre, el segundo para los atributos y el tercero para los métodos. Cada clase debe tener un nombre único, que las diferencie de las otras.

Un atributo representa alguna propiedad de la clase que se encuentra en todas las instancias de la clase. Los atributos pueden representarse solo mostrando su nombre, mostrando su nombre y su tipo, e incluso su valor por defecto.

Un método u operación es la implementación de un servicio de la clase, que muestra un comportamiento común a todos los objetos. En resumen es una función que le indica a las instancias de la clase que hagan algo. Para separar las grandes listas de atributos y de métodos se pueden utilizar estereotipos.

Relaciones entre clases

Existen tres relaciones diferentes entre clases, Dependencias, Generalización y Asociación. En las relaciones se habla de una clase destino y de una clase origen. La clase origen es desde la que se realiza la acción de relacionar. Es decir desde la que parte la flecha, el destino es la que recibe la flecha. Las relaciones se pueden modificar con estereotipos o con restricciones.

Dependencias

Es una relación de uso, es decir una clase usa a otra, que la necesita para su cometido. Se representa con una flecha discontinua va desde la clase utilizadora a la clase utilizada. Con la dependencia mostramos que un cambio en la clase utilizada puede afectar al funcionamiento de la clase utilizadora, pero no al contrario. Aunque las dependencias se pueden crear tal cual, es decir sin ningún estereotipo (palabreja que aparece al lado de la línea que representa la dependencia) UML permite dar mas significado a las dependencias, es decir concretar mas, mediante el uso de estereotipos.

Asociación

Especifica que los objetos de una clase están relacionados con los elementos de otra clase. Se representa mediante una línea continua, que une las dos clases. Podemos indicar el nombre, multiplicidad en los extremos, su rol, y agregación.

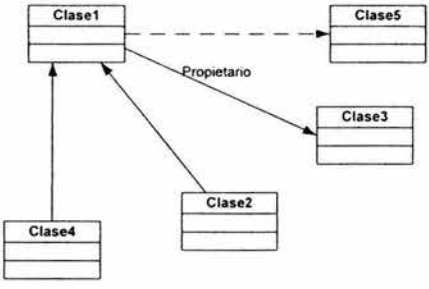


Figura II.22 Diagrama de clase

En la Figura II.22 se han creado cuatro clases. La clase principal es Usuario, que tiene dos clases hijas UsuarioADM y UsuarioINF. El usuario mantiene una relación de asociación con la clase Clave, se indica que es propietario de una clave, o de un número indeterminado de ellas. Se le crea también una relación de dependencia con la clase Perfil, es decir las instancias de usuario contendrán como miembro una instancia de Perfil.

Diagramas de objetos
Forma parte de la vista estática del sistema. En este diagrama se modelan las instancias de las clases del diagrama de clases. Muestra a los objetos y sus relaciones, pero en un momento concreto del sistema. Estos diagramas contienen objetos y enlaces. En los diagramas de objetos también se pueden incorporar clases, para mostrar la clase de la que es un objeto representado.

En este tipo de diagrama se muestra un estado del diagrama de eventos. Para realizar el diagrama de objetos primero se debe decidir que situación queremos representar del sistema. Es decir si disponemos de un sistema de mensajería, deberemos decidir que representaremos el sistema con dos mensajes entrantes, los dos para diferentes departamentos, dejando un departamento inactivo.

En un diseño nos podemos encontrar con multitud de diagramas de objetos, cada uno de ellos representando diferentes estados del sistema.

Diagrama de componentes
Se utilizan para modelar la vista estática de un sistema. Muestra la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema.

En el situaremos librerías, tablas archivos, ejecutables y documentos que formen parte del sistema. Uno de los usos principales es que puede servir para ver que componentes pueden compartirse entre sistemas o entre diferentes partes de un sistema.

En la Figura II.23 tenemos un componente del sistema de Windows. En el diagrama de componentes de Windows debe salir este componente, ya que sin el sistema no funcionaría.

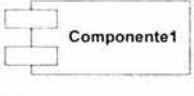


Figura II.23 Componente del sistema Windows

Como ya hemos indicado antes todo objeto UML puede ser modificado mediante estereotipos, los Standard que define UML son:

- Executable

- Library
- Table
- File
- Document

Aunque por suerte no estamos limitados a estas especificaciones. Si queremos modelar un proyecto de Internet donde nuestros componentes son ASP, HTML, y Scripts, y queremos marcarlo en el modelo, utilizamos un estereotipo. Existen ya unos definidos WAE (Web Applications Extensión) explicado más adelante.

Diagramas secuencia
El diagrama de secuencia forma parte del modelado dinámico del sistema. Se modelan las llamadas entre clases desde un punto concreto del sistema. Es útil para observar la vida de los objetos en sistema, identificar llamadas a realizar o posibles errores del modelado estático, que imposibiliten el flujo de información o de llamadas entre los componentes del sistema.
En el diagrama de secuencia se muestra el orden de las llamadas en el sistema. Se utiliza un diagrama para cada llamada a representar. Es imposible representar en un solo diagrama de secuencia todas las secuencias posibles del sistema, por ello se escoge un punto de partida. El diagrama se forma con los objetos que forman parte de la secuencia, estos se sitúan en la parte superior de la pantalla, normalmente en la izquierda se sitúa al que inicia la acción. De estos objetos sale una línea que indica su vida en el sistema. Esta línea simple se convierte en una línea gruesa cuando representa que el objeto tiene el foco del sistema, es decir cuando el esta activo.

En la Figura II.24 tenemos tres objetos, y un actor, situado a la izquierda que es el que inicia la acción. Como podemos ver el objeto de más a la derecha aparece mas abajo que los otros existentes. Esto se debe a que recibe una llamada de creación. Es decir el objeto no existe en el sistema hasta que recibe la primera petición.

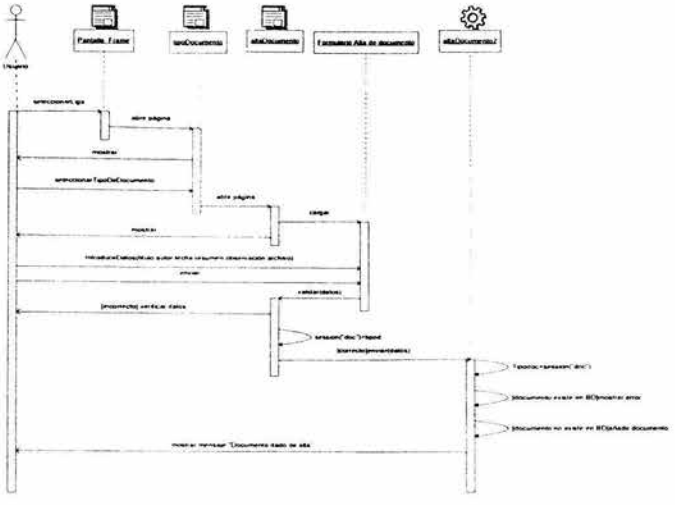


Figura II.24 Diagrama de secuencia

2.5.5.2 Extensión para Aplicaciones Web (WAE) con UML

Jim Conallen propone una extensión al UML para diseñar aplicaciones Web. Esto se muestra en la Figura II.25.

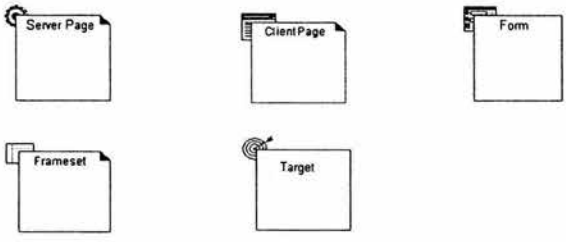


Figura II.25 Extensiones WAE

Dado que las páginas Web son los principales componentes de la arquitectura Web, hay que poder modelarlas. Usando UML podemos ver una página Web como un objeto.

Es conveniente hacer la distinción entre páginas del servidor y páginas del cliente. Los scripts de las páginas del servidor representan los métodos de esta clase. Las páginas del cliente tienen métodos que se ejecutan solamente del lado del cliente, como por ejemplo, Java Applets y controles ActiveX.

Hay una relación fundamental entre las páginas del servidor y las páginas del cliente, y es que las páginas del servidor crean las páginas del cliente. Esta relación es en una sola dirección, y para modelarla se usa el estereotipo <<builds>>. De este modo, se indica cuál página del servidor es encargada de crear la página del cliente. En la Figura II.26 se muestra un ejemplo.

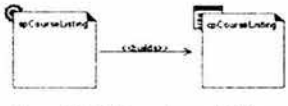


Figura II.26 Estereotipo <<builds>>

Algunas páginas del servidor podrían redireccionar ciertas solicitudes de procesamiento a otras páginas servidoras (una especie de IF). Permitir modelar estas situaciones es útil para la reutilización. Para esto se utiliza el estereotipo <<redirects>> tal como se muestra en la Figura II.27.

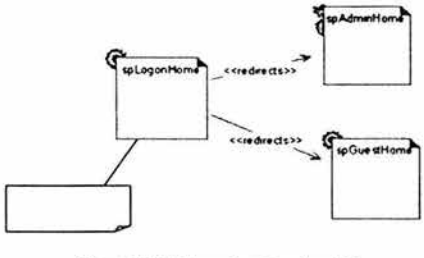


Figura II.27 Estereotipo <<redirects>>

Otra relación importante en el diseño de aplicaciones Web es el vínculo (*link*, o *anchor*) entre páginas. Las páginas vinculadas podrían ser páginas de cliente o del servidor. El estereotipo <<links>> define relaciones entre páginas cliente y otras páginas (cliente o servidoras). Un ejemplo de este tipo de estereotipo se observa en la Figura II.28.

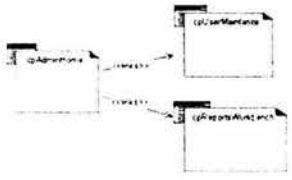


Figura II.28 Estereotipo <<links>>

Si un vínculo (*hyper link*) incluye parámetros, éstos son modelados como atributos del link fuera de la asociación.

Dado que una página podría tener varios formularios (forms) es posible que desde esta página se acceda a diferentes páginas. Los formularios se modelan con el estereotipo <<form>> (un estereotipo por cada formulario). Las páginas cliente contienen formularios tal como se describe en la Figura II.29

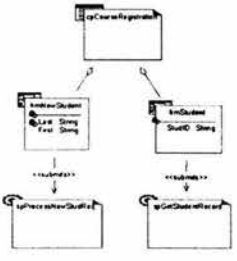


Figura II.29 Estereotipo <<forms>>

Usando *frames*, una página cliente podría estar compuesta por múltiples páginas al mismo tiempo. Los frames se implementan en HTML usando un *frameset*. Un frameset podría a su vez estar contenido en otro frameset. Las páginas Web contenidas en un frame se llaman *targets*. El estereotipo <<targeted link>> hace referencia a páginas que van ser cargadas en un frame distinto del que contiene la página que tiene el link. Un ejemplo de este tipo de estereotipo se muestra en la Figura II.30.

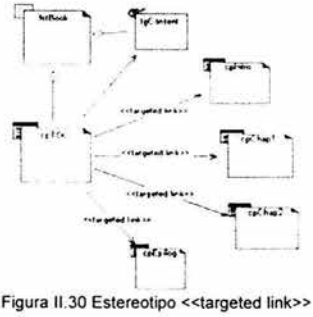


Figura II.30 Estereotipo <<targeted link>>

CAPÍTULO

III

ANÁLISIS Y DISEÑO DEL SISTEMA

CONTENIDO

3.1 Introducción

Para abordar el desarrollo de este sistema, se utilizará un método para modelado orientado a objetos (OO) según el planteamiento conceptual de UML.

Primeramente identificaremos los requerimientos del sistema los cuales serán una parte fundamental para poder modelar los requisitos del sistema utilizando los casos de uso y luego los describiremos utilizando plantillas que muestran las características del caso de uso en cuestión. Posteriormente, debemos describir los escenarios de los casos de uso. Para ello utilizaremos los diagramas de secuencia de UML, pero sustituiremos las clases controlador y las clases interfaz por páginas servidor y cliente respectivamente. Para esto se especificarán los estereotipos de UML para aplicaciones Web propuestas por Conallen[1]. Una vez descritos los casos de uso con sus correspondientes diagramas de secuencia, pasamos a realizar la parte del diseño.

Partiendo de la realización de los casos de uso debemos llegar al diseño de la página Web. En ella no tenemos clases, de modo que habrá que identificar los elementos y relaciones Web correspondientes. Para ello, inicialmente, crearemos un diagrama de componentes (una visión general del diagrama de clases) por cada caso de uso, donde se mostrarán los elementos y relaciones de la página Web (links, formularios, frames, etc.) correspondientes a la realización del caso de uso. Básicamente conseguimos describir, para cada diagrama de secuencia que muestra el funcionamiento de un caso de uso, los objetos de la página Web y las relaciones que existen entre ellos.

Una vez realizados los diagramas de componentes ya tenemos identificados todos los elementos y relaciones que intervienen en el Web para el proceso de negocio que estamos tratando. Ahora debemos refinar está visión general que conseguimos con los diagramas de componentes y plasmar el diseño en un diagrama de clases UML. Los elementos y relaciones son los mismos que aparecen en los diagramas de componentes, pero se muestran los elementos de la página Web como si fueran objetos de una clase software, incluyendo los atributos (variables de aplicación y de sesión) y los métodos (funciones javascript, asp, asp. net). Muchas de las clases y relaciones que aparecen en este diagrama serán estereotipadas, haciendo referencia a elementos propios y comunes de las páginas Web, siguiendo la notación descrita en las extensiones UML propuestas por Conallen.

Como soporte a este proceso se ha empleado la herramienta Microsoft Visio Profesional 2002 para la creación de los diferentes diagramas.

3.2 Requerimientos informáticos del Departamento de Ingeniería en Computación

En esta parte del capítulo se analizará detalladamente los requerimientos necesarios para el sistema de acuerdo a las necesidades del Departamento de Ingeniería en computación.

El Departamento de Ingeniería en Computación requiere del desarrollo de un sistema que opere vía Web. Dicho sistema contará con las herramientas necesarias que permita al Jefe del Departamento, Coordinador de Carrera y otros Académicos del Departamento de Ingeniería en Computación de la Facultad de Ingeniería disponer de la información en forma rápida, eficaz y de manera confiable desde cualquier sitio en cualquier momento; lo cual da como resultado una toma de decisiones precisa, exacta y fundamentada en la información.

[1] Conallen Jim, "Building Web Applications UIT UML", Addison-Wesley, 1999.

3.2.1 Identificación de la información

Se llevó a cabo una serie de entrevista con el coordinador de la carrera de Ingeniería en Computación y con las encargadas del área administrativa para recabar la información necesaria. El sistema almacenará información de documentos así como información de usuarios que tengan acceso al sistema.

Los documentos que podrán ser almacenados son: oficios, memorandums, minutas, convocatorias, boletines, informes y reportes entre otros que puedan añadirse en un futuro sin que altere el funcionamiento del sistema. La información que se necesita para almacenar estos documentos es:

- Titulo del documento
- Autor o autores
- Fecha de publicación
- Resumen
- Observación
- Archivo del documento en formato doc, pdf o imagen.

Además de los documentos anteriores, se necesita que se almacene la información de las tesis que se llevan a cabo en el Departamento. La información necesaria que se necesita es:

- Titulo de la tesis
- Autor o autores
- Fecha de terminación
- Director o directores
- Clave asignada por la Facultad
- Resumen
- Observación

La información requerida para el almacenamiento de los usuarios es la siguiente:

- Nombre completo (Nombre, apellido paterno y materno)
- Tipo de permiso (Administrador, académico, secretaria)
- Grado de estudios
- Domicilio (calle, número, colonia, delegación)
- Teléfono particular
- Teléfono de oficina
- Correo electrónico
- Login
- Contraseña

Un factor importante ha considerar en el sistema es la seguridad, la cual es importante al momento de consultar la información.

El sistema deberá aceptar a usuarios como administrador, secretarias y académicos del Departamento de Ingeniería en Computación, los cuales tendrán ciertos privilegios dentro del sistema. Una vez dentro del sistema, se podrá añadir, consultar, modificar y eliminar un documento, así como para un usuario. Además se contará con servicios de chat, envío de e-mail así como consultar información confidencial que solo podrá ser llevada a cabo por académicos del departamento a través de un buzón particular.

Las Figuras III.1 y III.2 muestran la situación actual del departamento y su situación futura respectivamente.

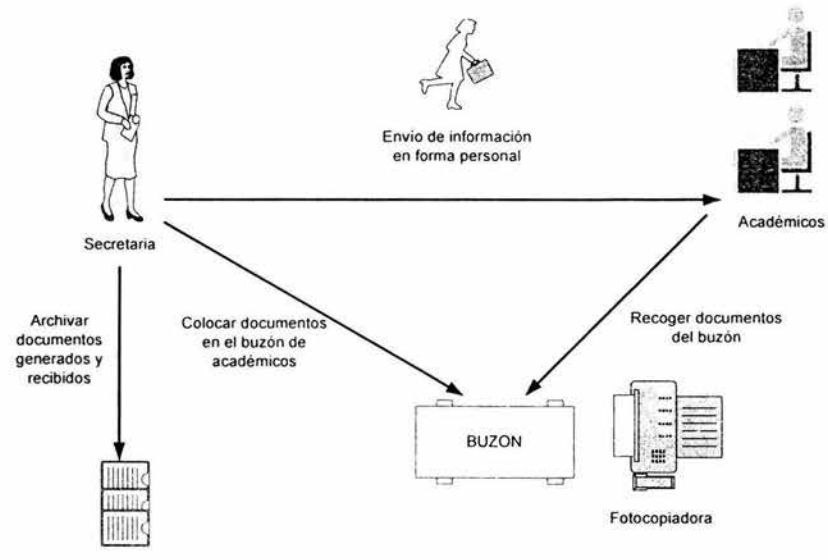


Figura III.1 Situación actual

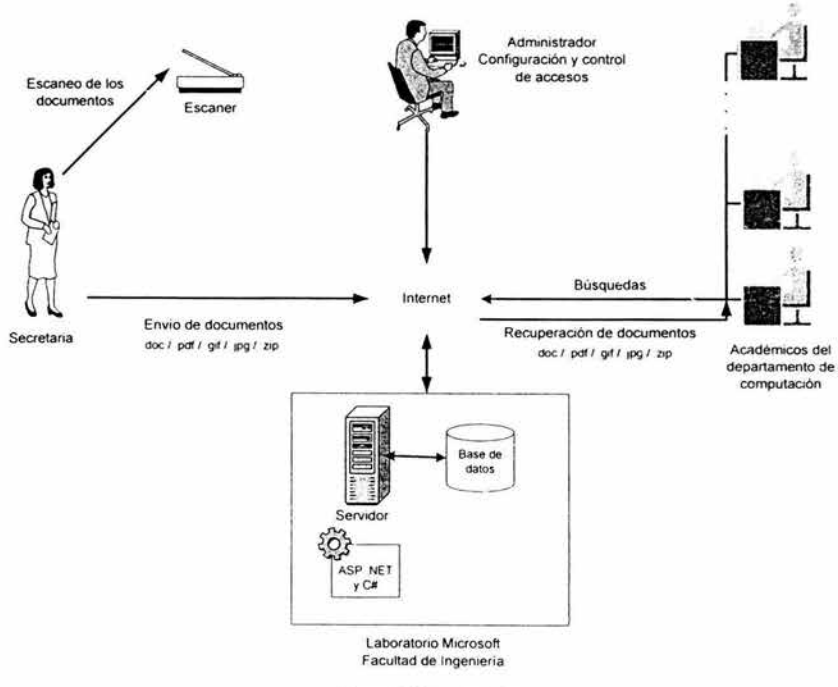


Figura III.2 Situación Futura

3.2.2 Requerimientos de hardware

El sistema para el Departamento de Ingeniería en Computación sigue una arquitectura cliente – servidor. Por tal motivo, la parte robusta de la aplicación esta en una maquina "principal" o "central" denominado servidor al cual se conectarán los ordenadores de los usuarios denominados clientes.

Para lograr lo anterior, se utilizarán las herramientas existentes en el laboratorio Microsoft de la Facultad de Ingeniería el cual cuenta con el siguiente hardware:

- Servidor Compact Proliant ML530. Las características de este servidor son:
  - Procesador Pentium III a 1 GHz
  - Memoria de 1152 MB
  - Tres discos SCSI con una capacidad total de 150 GB
- Equipos clientes.
  - Procesador Pentium III a 1 GHz
  - Memoria de 512 MB
  - Disco duro de 80 GB

Estos equipos se encuentran conectados a Internet, con lo cual podrán acceder sin problemas al sistema que se alojará en el servidor del laboratorio Microsoft.

3.2.3 Requerimientos de software

El software es la parte esencial para la construcción del sistema. El software debe cumplir con las siguientes características:

- Efectividad. Esto significa que si el software elegido cumple con el funcionamiento requerido.
- Fácil uso. Tiene que ver con el aprendizaje para el manejo del software.
- Portabilidad. Esto es muy importante si se desea poder migrar a una plataforma diferente en el futuro.
- Costo. Esta parte es muy importante ya que se puede elegir entre software gratuito o con licencia, el cual puede tener un alto costo.
- Ayuda en línea. Si el software a manejar esta respaldado por documentación en un sitio web.
- Recursos de hardware. Esto tiene que ver con el funcionamiento estable entre el software y el hardware.
- Soporte técnico. Se debe de tomar en cuenta que las herramientas seleccionadas cuenten con un respaldo técnico.

El sistema se desarrollara en el laboratorio Microsoft con las herramientas donadas con licencia por dicha compañía. Por este motivo, el uso de estas herramientas es factible para el desarrollo del sistema, el cual cumple con las características antes mencionadas. Entre las herramientas disponibles se encuentran:

- Sistema operativo Windows 2003 Server
- Servicios de Internet Information Server 6.0 (IIS).
- Microsoft SQL Server 2000
- Microsoft Active Server Page .NET (ASP .NET) con C#
- Microsoft Visual Studio .NET
- Macromedia Dreamweaver XP

El servidor de la base de datos es la parte fundamental del sistema, ya que ahí radicaran todos los datos necesarios para el buen funcionamiento. El paradigma para este sistema de acceso es Internet. La mayoría de los servidores de base de datos actuales soportan protocolos de red. El más difundido es TCP/IP usado en Internet.
Por tanto, un servidor de base de datos accesible a través de Internet debe permitir poder realizar:

- Búsquedas en la base de datos en función a criterios.
- Recoger, ordenar y formatear datos.
- Añadir, modificar y borrar casos.
- Crear nuevas bases de datos o estructuras de información.
- Mantener una absoluta integridad.
- Ejecutar ciertos procedimientos mediante código programado.
- Tener un alto nivel de seguridad, no sólo ante las intrusiones de usuarios malintencionados, sino también de fallos del hardware asociado.

La Tabla III.1 muestra a grandes rasgos las diferencias entre Microsoft SQL Server 2000, Sybase Adaptive Server Anywhere 6.0 y Oracle.

Compatibilidad con Windows	Muy bueno	Bueno	Regular
SQL Server 2000	x		
Sybase		x	
Oracle			x

Facilidad de instalación	Muy bueno	Bueno	Regular
SQL Server 2000	x		
Sybase		x	
Oracle			x

Costo	Muy bueno	Bueno	Regular
SQL Server 2000	x		
Sybase		x	
Oracle		x	

Interfaz para administración	Muy bueno	Bueno	Regular
SQL Server 2000	x		
Sybase	x		
Oracle			x

Integridad de la base de datos	Muy bueno	Bueno	Regular
SQL Server 2000	x		
Sybase	x		
Oracle	x		

Seguridad integrada	Muy bueno	Bueno	Regular
SQL Server 2000	x		
Sybase	x		
Oracle	x		

Usuarios concurrentes	Muy bueno	Bueno	Regular
SQL Server 2000	x		
Sybase	x		
Oracle	x		

Almacenaje de la base de datos	Muy bueno	Bueno	Regular
SQL Server 2000	x		
Sybase	x		
Oracle	x		

Tabla III.1 Comparación entre SQL Server 2000, Sybase y Oracle

Estas comparaciones demuestran que la elección de SQL Server 2000 es una herramienta eficaz para la construcción del back – end del sistema.

3.3 Modelado de requisitos

Los diagramas de casos de uso son uno de los cinco tipos de diagrama de UML que se utilizan para el modelado de los aspectos estáticos de un sistema (los otros 4 diagramas estáticos son los de clases, de objetos, de componentes y de despliegue) y así poder obtener los requisitos del sistema. Estos diagramas son explicados en el tema II "Marco teórico".

Los diagramas de casos de uso emplean para modelar la vista de casos de uso del sistema. La mayoría de la veces, esto implica modelar el contexto del sistema, subsistema o clase, o el modelado de los requisitos del comportamiento de estos elementos.

En base a los requerimientos establecidos, podemos identificar tres actores principales: administrador, secretarias y académicos. Sus respectivos permisos se muestran en la Tabla III.2

Permiso	Administrador	Secretarias	Académicos
Alta de documento	X	X	
Consulta de documento	X	X	X
Modificación de documento	X	X	
Eliminación de documento	X		
Alta de usuarios	X	X	
Consulta de usuarios	X	X	X
Modificación de usuarios	X	X	
Eliminación de usuarios	X		
Reportes	X		
Servicio de Email	X	X	X
Servicio de Chat	X	X	X
Buzón			X

Tabla III.2 Usuarios del sistema

A continuación se exponen los diagramas de casos de uso, implementados para el sistema.

3.3.1 Diagrama de casos de uso

Diagrama "comportamiento general del sistema"
En este diagrama, mostrado en la Figura III.3, se pretende dar un panorama general del funcionamiento global del sistema, observando la interacción que tienen los actores principales con su entorno.

Los tres actores, administrador, secretaria y académicos, tendrán que ser autentificados para ingresar al sistema mediante la validación de su login y password. Una vez que se haya ingresado al sistema, cada actor tendrá ciertos derechos autorizados en este, tal como se muestra en la Tabla III.2

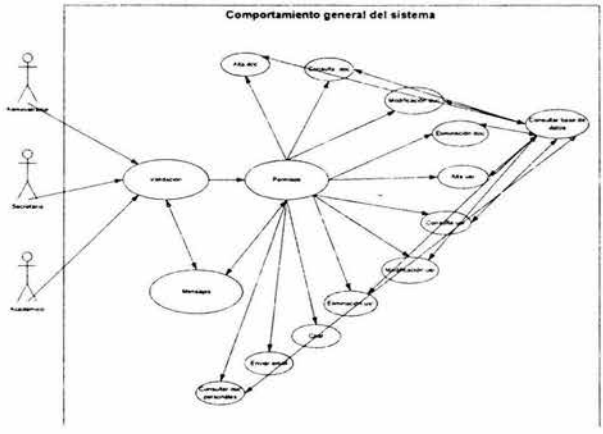


Figura III.3 Comportamiento general del sistema

En los siguientes casos de uso se obvia la literatura de estos, mostrando en su lugar una tabla que muestra en resumen el funcionamiento del caso de uso en cuestión.

***Caso de uso* Alta de documento**
Objetivo Guardar un documento para ser utilizado posteriormente
Actores Administrador, Secretaria
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Seleccionar el tipo de documento a guardar y dar sus datos correspondientes. 2. Servidor: Añadir nuevo registro a la base de datos.
Variaciones Si el documento existe en la base de datos, volver a pedir otro documento
Cuestiones - ¿Qué tipos de archivos pueden ser almacenados en la base de datos?

Tabla III.3 Resumen para el caso de uso de la Figura III.4

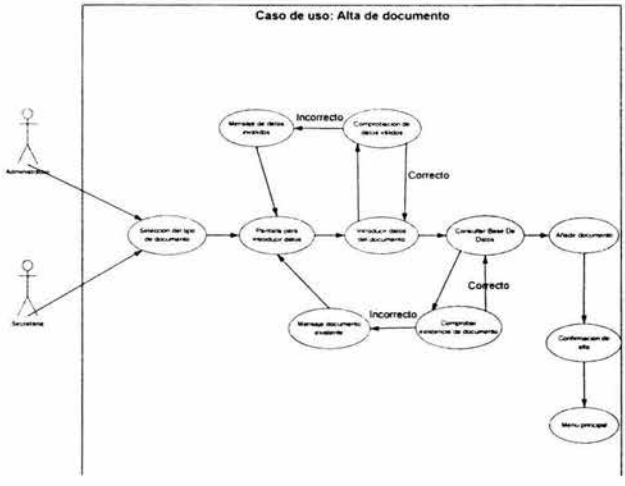


Figura III.4 Caso de uso para el alta de documento

Caso de uso **Alta de un nuevo tipo de documento**
Objetivo Guardar una nueva clasificación para almacenar documentos
Actores Administrador, Secretaria
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Seleccionar la liga correspondiente para guardar un nuevo tipo de documento y dar un nombre apropiado. 2. Servidor: Añadir el nuevo tipo de documento a la base de datos
Variaciones Si el tipo de documento existe en la base de datos, volver a pedir otro tipo
Cuestiones

Tabla III.4 Resumen para el caso de uso de la Figura III.5

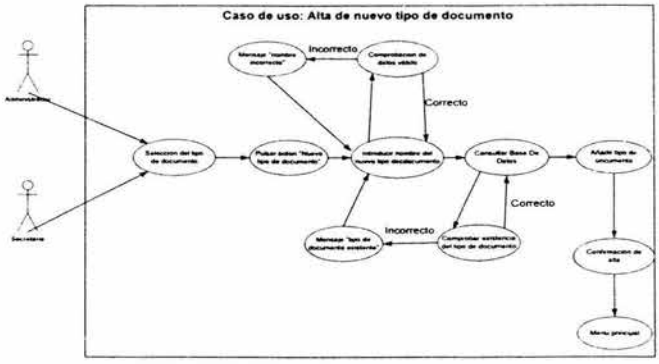


Figura III.5 Caso de uso para el alta de nuevo tipo de documento

Caso de uso **Alta de usuario**
Objetivo Añadir un nuevo usuario al sistema
Actores Administrador, Secretaria
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Seleccionar alta de usuario y llenar los campos correspondientes. 2. Servidor: Añadir el usuario a la base de datos.
Variaciones Si el usuario existe en la base de datos, volver a pedir otro usuario
Cuestiones

Tabla III.5 Resumen para el caso de uso de la Figura III.6

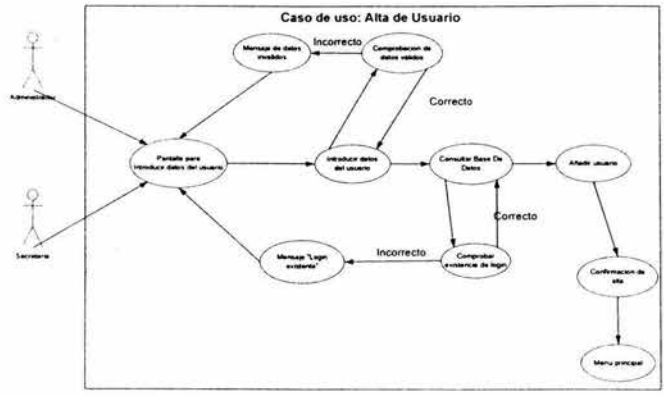


Figura III.6 Caso de uso para el alta de usuario

Caso de uso **Alta de un nuevo tipo de usuario**
Objetivo Añadir una nueva clasificación para almacenar usuarios
Actores Administrador, Secretaria
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Seleccionar nuevo tipo de usuario y dar un nombre apropiado. 2. Servidor: Añadir la nueva clasificación a la base de datos.
Variaciones Si la nueva clasificación existe en la base de datos, volver a pedir otro tipo
Cuestiones

Tabla III.6 Resumen para el caso de uso de la Figura III.7

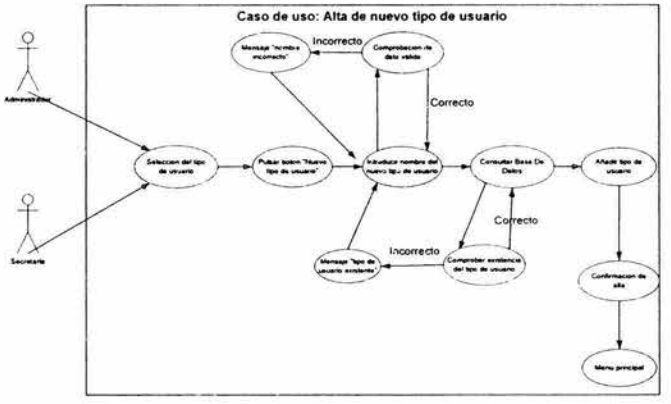


Figura III.7 Caso de uso para el alta de nuevo tipo de usuario

Caso de uso **Consulta de documento**
Objetivo Llevar a cabo una consulta eficiente de los diversos documentos almacenados
Actores Administrador, Secretaria, Académico
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Seleccionar el filtro para la búsqueda. 2. Servidor: Desplegar los documentos encontrados. 3. Cliente: Seleccionar el documento deseado. 4. Servidor: Desplegar las características del documento seleccionado
Variaciones Si no existen documentos en la base de datos con el criterio seleccionado, desplegar un mensaje al usuario.
Cuestiones

Tabla III.7 Resumen para el caso de uso de la Figura III.8

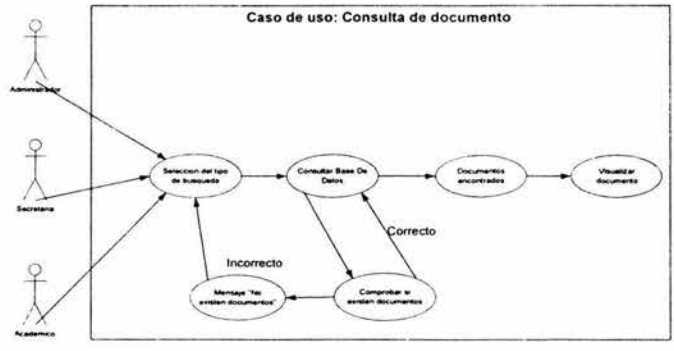


Figura III.8 Caso de uso para la consulta de documento

Caso de uso **Consulta de usuarios**
Objetivo Desplegar los usuarios registrados en el sistema
Actores Administrador, Secretaria
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Seleccionar la liga correspondiente a la consulta de usuarios. 2. Servidor: Desplegar los usuarios registrados.
Variaciones
Cuestiones

Tabla III.8 Resumen para el caso de uso de la Figura III.9

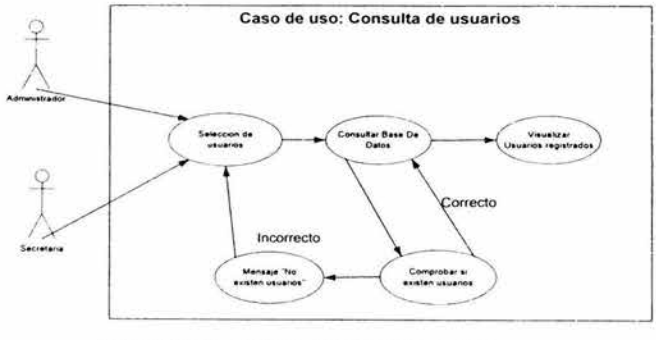


Figura III.9 Caso de uso para la consulta de usuarios

Caso de uso **Consultar buzón**
Objetivo Consultar, por parte de los académicos, los documentos contenidos en su buzón
Actores Académico
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Seleccionar el documento deseado. 2. Servidor: Mostrar características del documento.
Variaciones Si no existe documento en el buzón, mandar aviso.
Cuestiones -¿Cómo diferenciar la consulta de los documentos del buzón (particulares) con los documentos que todos pueden consultar?

Tabla III.9 Resumen para el caso de uso de la Figura III.10

Figura III.10 Caso de uso para consultar el buzón

Caso de uso **Modificación de documento**
Objetivo Poder modificar los campos que componen a un documento
Actores Administrador, Secretaria
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Seleccionar el documento a modificar. 2. Servidor. Mostrar los campos del documento. 3. Cliente: Añadir nuevos datos 4. Servidor: Modificar el registro en la base de datos.
Variaciones
Cuestiones - ¿Existirá un limite de tiempo para modicar el registro por parte de la secretaria?

Tabla III.10 Resumen para el caso de uso de la Figura III.11

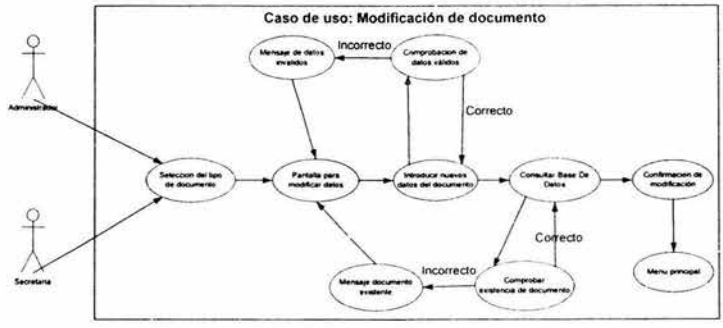


Figura III.11 Caso de uso para la modificación de documento

Caso de uso **Modificación de usuario**
Objetivo Modificar los campos de un usuario.
Actores Administrador, Secretaria
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Seleccionar el usuario a modificar. 2. Servidor: Mostrar los datos del usuario. 3. Cliente: Añadir nuevos datos 4. Servidor: Modificar el registro en la base de datos.
Variaciones
Cuestiones - ¿Existirá un limite de tiempo para modicar el registro por parte de la secretaria?

Tabla III.11 Resumen para el caso de uso de la Figura III.12

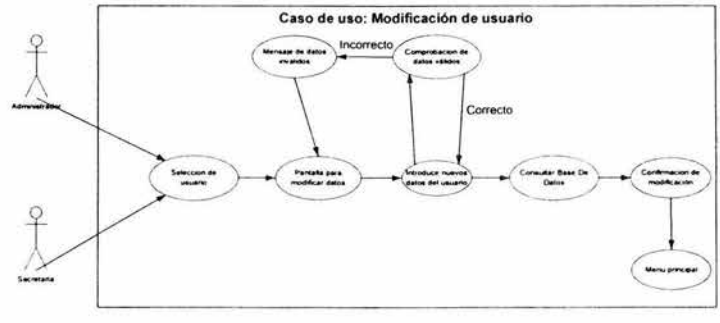


Figura III.12 Caso de uso para la modificación de usuario

Caso de uso **Eliminación de documento**
Objetivo Eliminar documentos de la base de datos
Actores Administrador
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Seleccionar el documento a eliminar. 2. Servidor: Eliminar el registro en la base de datos.
Variaciones
Cuestiones - ¿Habrá alguna forma de recuperar el archivo?

Tabla III.12 Resumen para el caso de uso de la Figura III.13

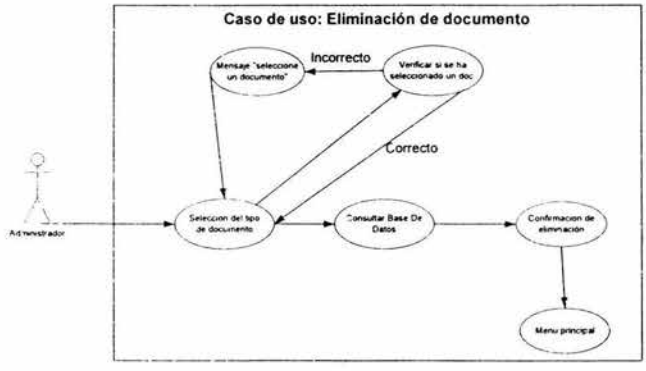


Figura III.13 Caso de uso para la eliminación de documento

Caso de uso **Eliminación de usuarios**
Objetivo Eliminar usuarios de la base de datos
Actores Administrador
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Seleccionar el usuario a eliminar. 2. Servidor: Eliminar el registro en la base de datos.
Variaciones
Cuestiones - ¿Habrá alguna forma de recuperar el archivo?

Tabla III.13 Resumen para el caso de uso de la Figura III.14

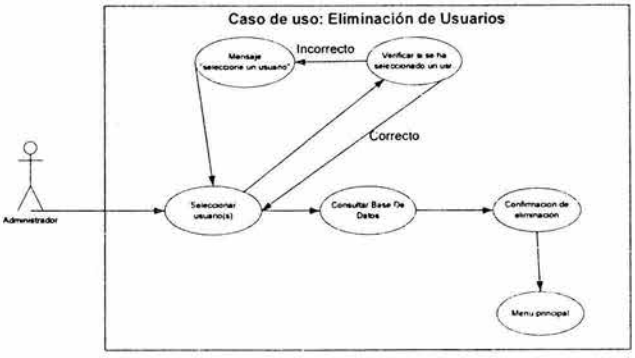


Figura III.14 Caso de uso para la eliminación de usuarios

Caso de uso **Enviar documento al buzón de académico**
Objetivo Enviar un documento a un académico para que este solo pueda consultarlo
Actores Administrador, Secretaria
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario 1. Cliente: Seleccionar el filtro para la búsqueda. 2. Servidor: Desplegar los documentos encontrados. 3. Cliente: Seleccionar el documento deseado. 4. Servidor: Desplegar las características del documento seleccionado. 5. Cliente: Seleccionar opción de enviar documento a buzón 6. Servidor: Mostrar lista de académicos 7. Cliente: Seleccionar académico 8. Cliente: enviar documento 9: Servidor: Almacena referencia del documento al buzón del usuario
Variaciones
Cuestiones

Tabla III.14 Resumen para el caso de uso de la Figura III.15

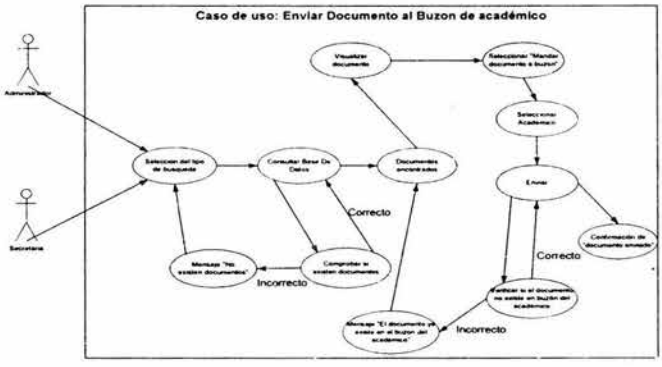


Figura III.15 Caso de uso para enviar documento al buzón de académico

Caso de uso **Enviar e-mail**
Objetivo Contar con un servicio de envió de correo, proporcionando la opción de anexar un archivo.
Actores Administrador, Secretaria, Académico
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Llenar los campos requeridos para el envio de correo 2. Servidor: Enviar correo.
Variaciones
Cuestiones

Tabla III.15 Resumen para el caso de uso de la Figura III.16

Figura III.16 Caso de uso para el envio de e-mail

Caso de uso **Chat**
Objetivo Contar con un servicio de chat para uso académico
Actores Administrador, Secretaria, Académico
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Introducir un nombre para entrar al chat 2. Servidor: Verificar si el nombre existe en el chat. 3. Cliente: Mandar mensajes 4. Servidor: refrescar la pantalla de los mensajes y de los usuarios conectados
Variaciones Si el nombre existe en el chat , volver a pedir otro nombre
Cuestiones

Tabla III.16 Resumen para el caso de uso de la Figura III.17

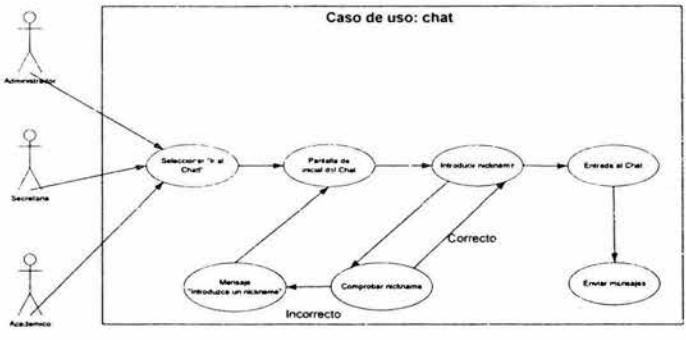


Figura III.17 Caso de uso para el chat

Caso de uso **Validación de usuarios**
Objetivo Permitir a un usuario la entrada al sistema con su respectivo permiso
Actores Administrador, Secretaria, Académico
Precondiciones
Pasos Realizar tantas veces como el usuario considere necesario: 1. Cliente: Introducir login y password. 2. Servidor: Validar datos. 3. Servidor: Si los datos son correctos, se permite la entrada al sistema
Variaciones Si los datos son incorrectos, acceso denegado
Cuestiones - ¿Qué tipo de seguridad se tendrá al enviar el password?

Tabla III.17 Resumen para el caso de uso de la Figura III.18

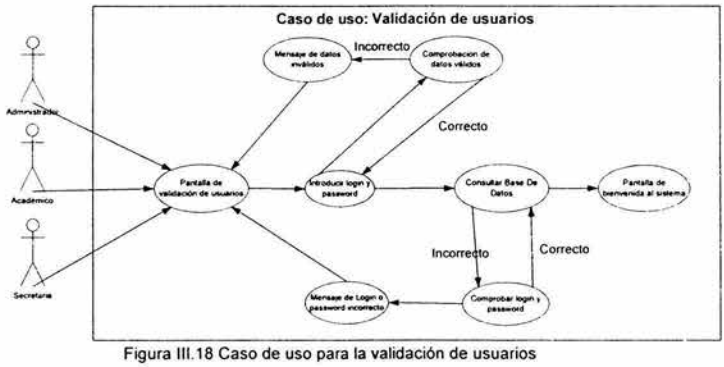


Figura III.18 Caso de uso para la validación de usuarios

Caso de uso **Salir del sistema**
Objetivo Salir del sistema, asegurándose con terminar el fin de sesión del usuario
Actores Administrador, Secretaria, Académico
Precondiciones
Pasos 1 Cliente: Seleccionar opción salir del sistema. 2. Servidor: Terminar con la sesión del usuario 3. Servidor: Redireccionar al usuario a la página de validación.
Variaciones
Cuestiones

Tabla III.18 Resumen para el caso de uso de la Figura III.19

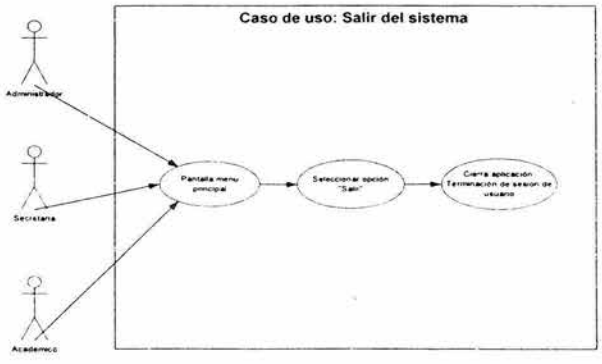


Figura III.19 Caso de uso para salir del sistema

3.4 Diagrama de secuencias

En esta parte del diseño del sistema describimos los escenarios de casos de uso mediante diagramas de secuencias UML, pero sustituyendo las clases controlador y las clases interfaz por páginas servidor y cliente. Para identificar que página es servidor y cual cliente, se describe con una pequeña figura por encima de la notación:

Página servidor

Página cliente

A continuación se presentan los diagramas de secuencia para los casos de uso anteriores.

Diagrama de secuencia "Alta de documento" (Figura III.20)

En este diagrama, Figura III.20, se muestra la secuencia de pasos que se deben seguir para dar de alta un documento en el sistema. Esto se inicia cuando el usuario selecciona la liga para abrir la página de "tipo de documento" contenida en un frame. Una vez mostrada esta página, el usuario selecciona el tipo de documento que desea dar de alta. Al hacer esto, se abre una nueva página correspondiente al "alta de documento" la cual carga un Formulario en donde el usuario tendrá que introducir los datos necesarios del documento como son: título, autor, fecha, resumen, observación, archivo.

Campos como título, autor, fecha, resumen y archivo son verificados para que no se encuentren vacíos. Si este es el caso, se le envía un mensaje al usuario indicándole que verifique los datos. Si los datos entrados son válidos, se guarda en una variable de sesión el tipo de documento (oficio, memorandum, reporte, etc.) para darle un seguimiento al documento.

Los datos son enviados a una página del servidor llamada "alta de documento 2", el cual en base a la variable de sesión del tipo de documento, se analizará primeramente si el documento ha sido ya almacenado en la base de datos. Si este es el caso, so mostrará un error al usuario. De lo contrario el documento es almacenado en la base de datos y se le muestra al usuario un mensaje indicándole que el documento ha sido dado de alta.

Los diagramas de secuencia subsecuentes son autodescriptivos.

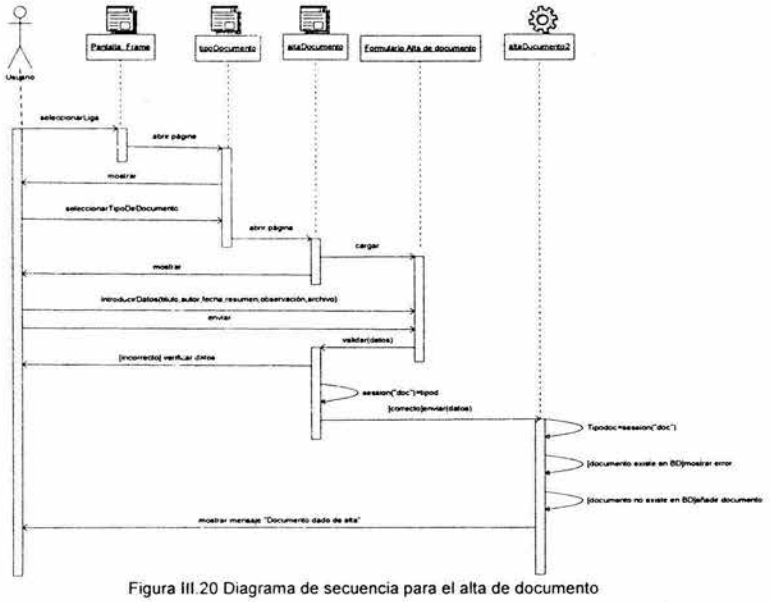


Figura III.20 Diagrama de secuencia para el alta de documento

Diagrama "Alta de un nuevo tipo de documento"

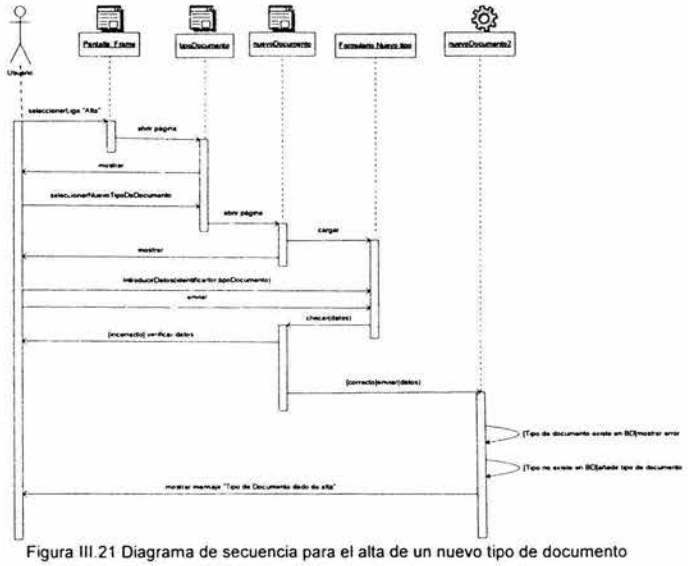


Figura III.21 Diagrama de secuencia para el alta de un nuevo tipo de documento

Diagrama "Alta de usuario"

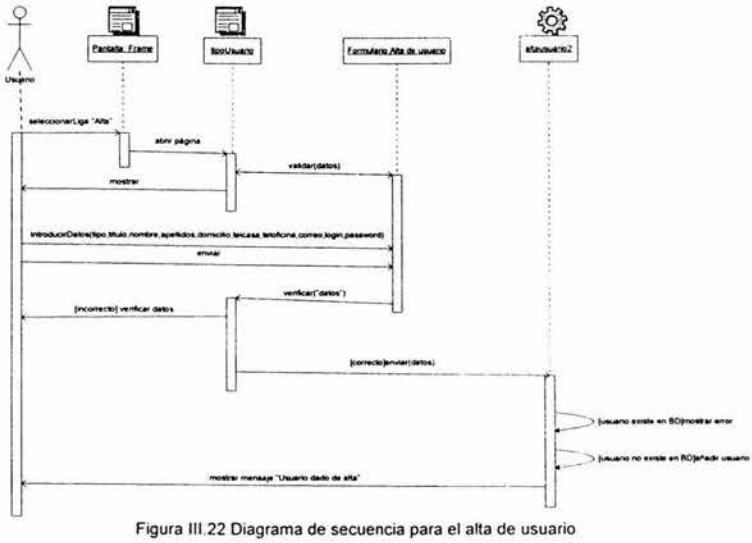


Figura III.22 Diagrama de secuencia para el alta de usuario

Diagrama "Alta de un nuevo tipo de usuario"

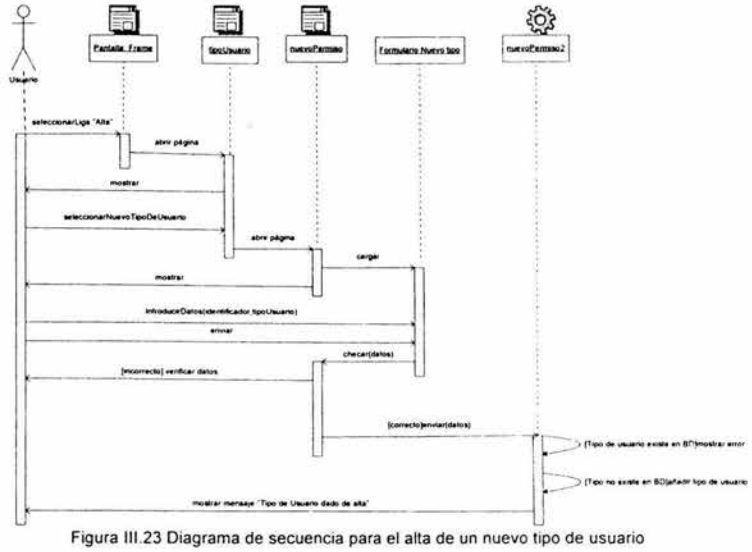


Figura III.23 Diagrama de secuencia para el alta de un nuevo tipo de usuario

Diagrama "Consulta de documento"

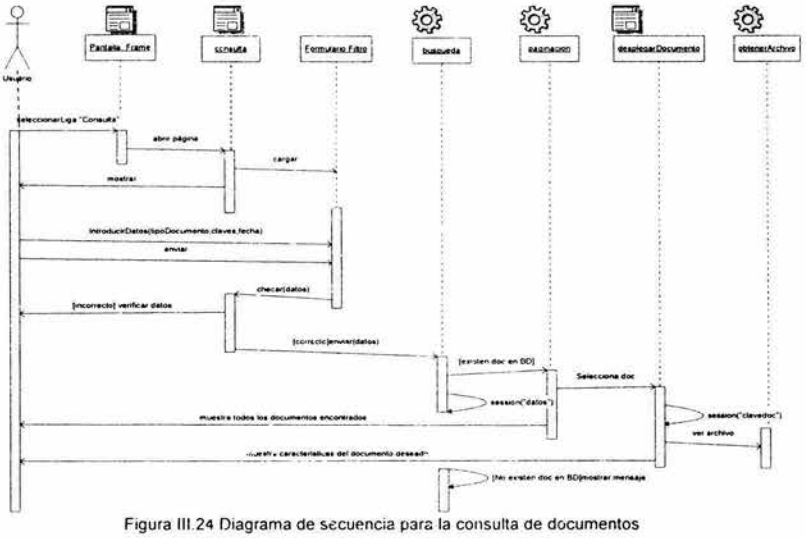


Figura III.24 Diagrama de secuencia para la consulta de documentos

Diagrama "Consulta de usuarios"

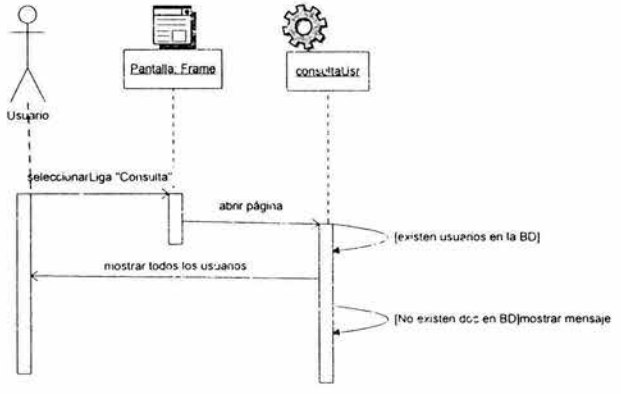


Figura III.25 Diagrama de secuencia para la consulta de usuarios

Diagrama "Consulta de buzón"

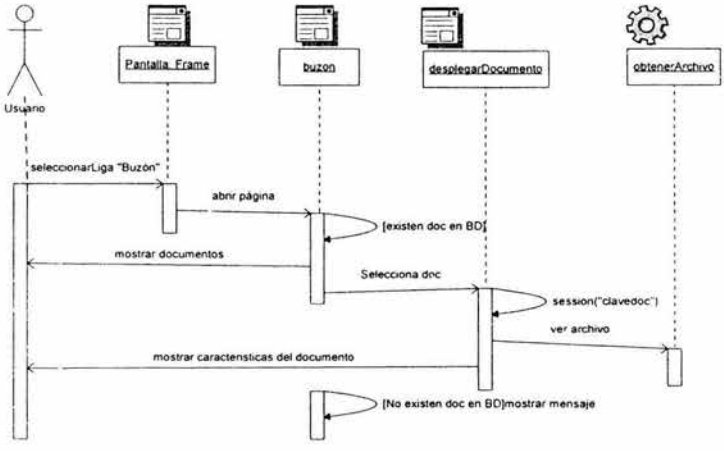


Figura III.26 Diagrama de secuencia para la consulta de buzón

Diagrama "Modificación de documento"

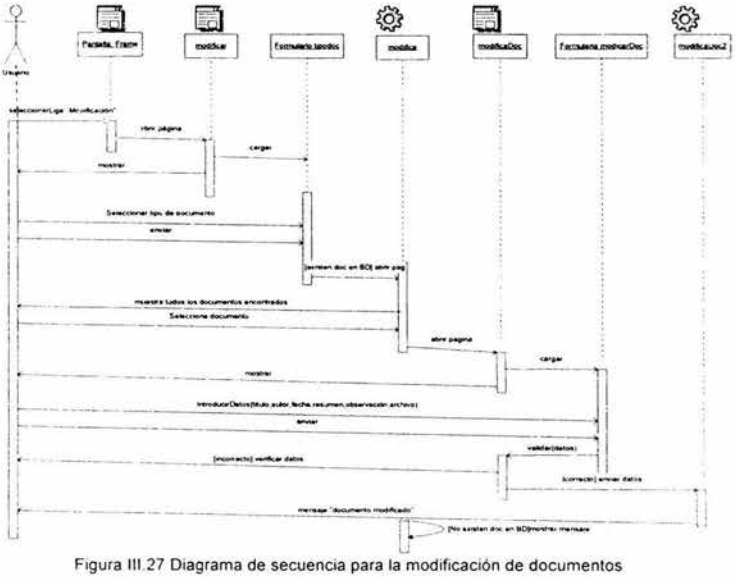


Figura III.27 Diagrama de secuencia para la modificación de documentos

Diagrama "Modificación de usuario"

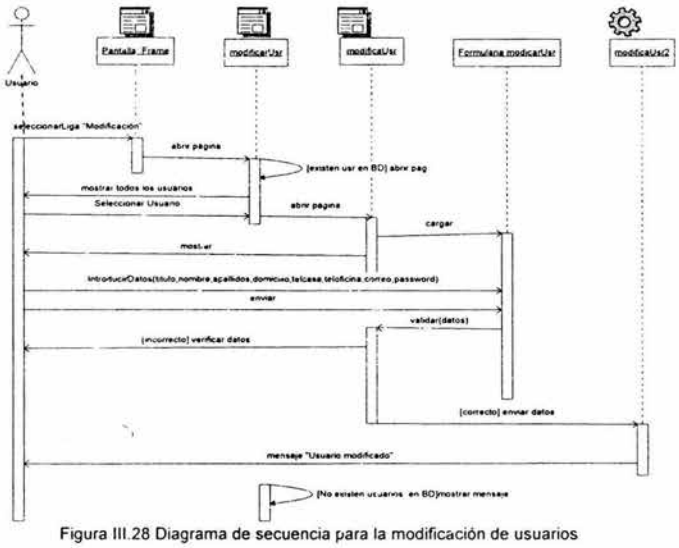


Figura III.28 Diagrama de secuencia para la modificación de usuarios

Diagrama "Eliminación de documento"

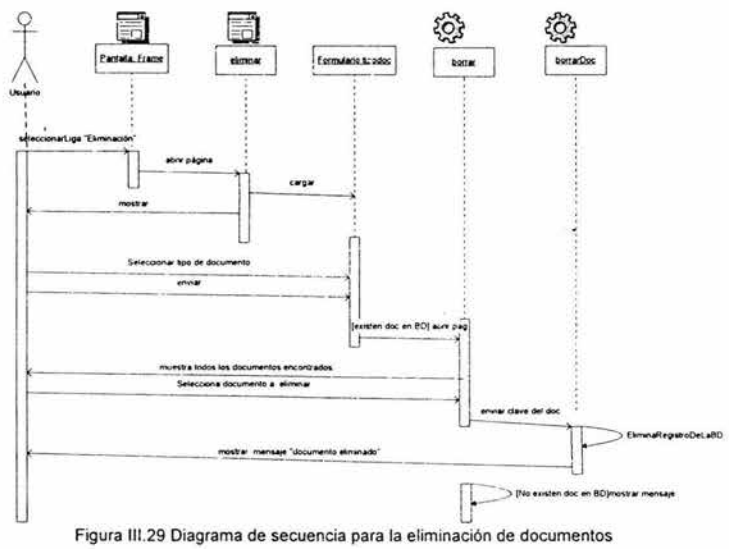


Figura III.29 Diagrama de secuencia para la eliminación de documentos

Diagrama "Eliminación de usuarios"

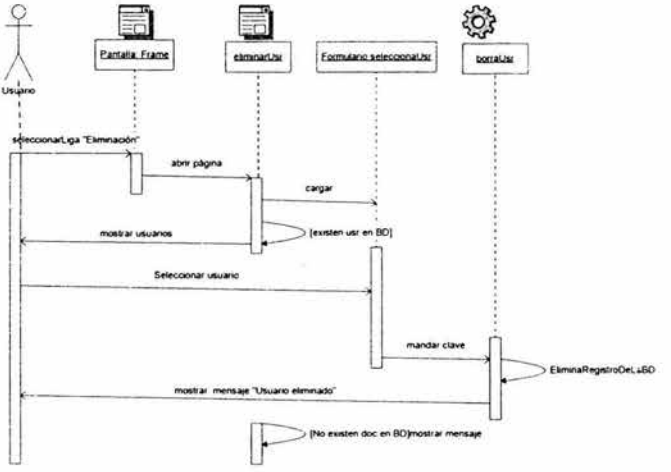


Figura III.30 Diagrama de secuencia para la eliminación de usuarios

Diagrama "Enviar documento al buzón de académico"

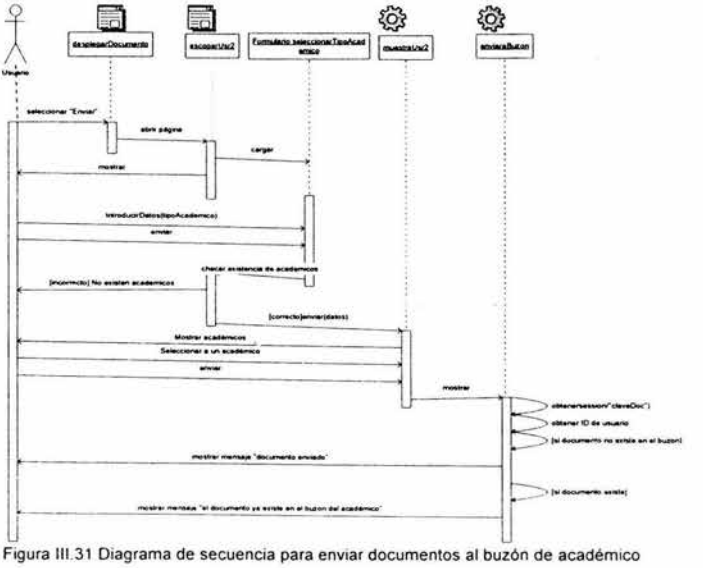


Figura III.31 Diagrama de secuencia para enviar documentos al buzón de académico

Diagrama "Enviar e-mail"

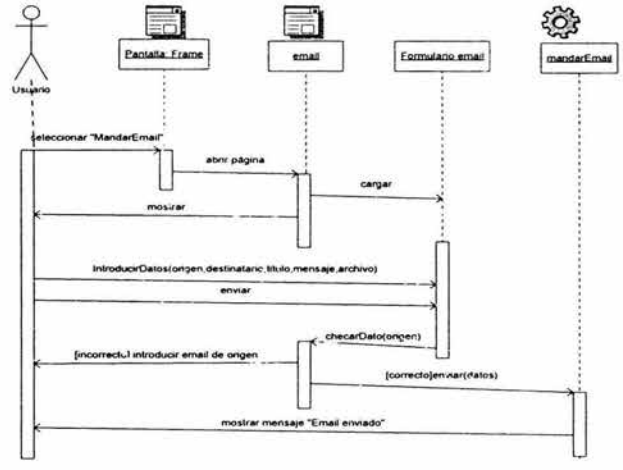


Figura III.32 Diagrama de secuencia para el envió de e-mail

Diagrama "Chat"

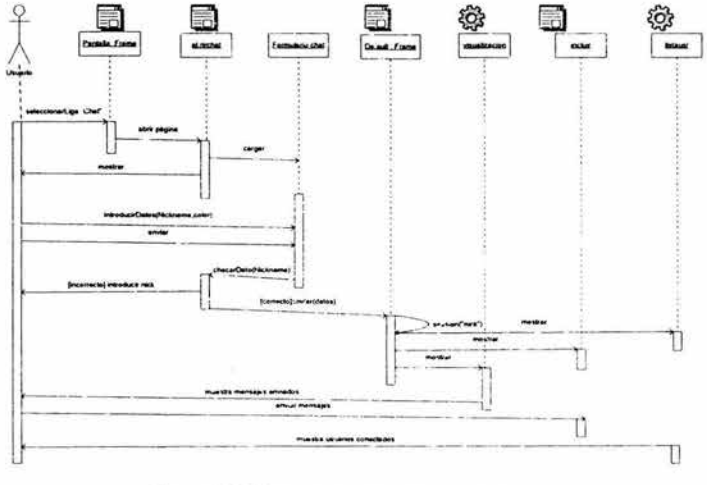


Figura III.33 Diagrama de secuencia para el chat

Diagrama "Validación de usuarios"

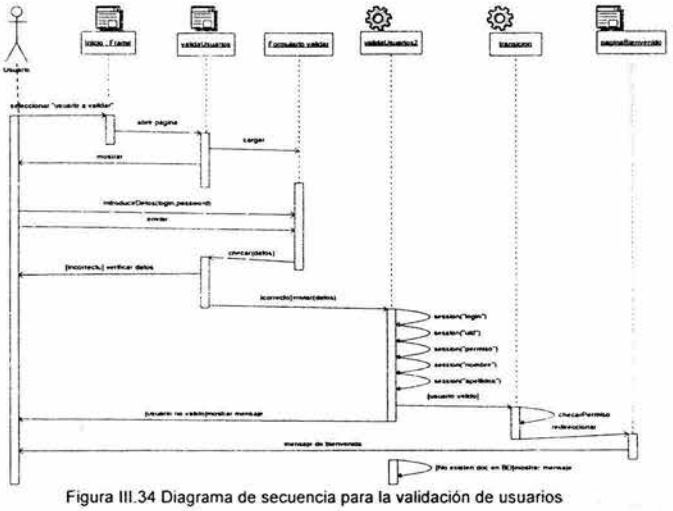


Figura III.34 Diagrama de secuencia para la validación de usuarios

Diagrama "Salir del sistema"

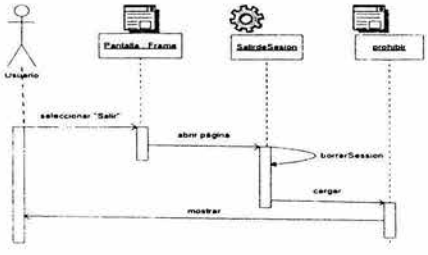


Figura III.35 Diagrama de secuencia para salir del sistema

3.5 Diagrama de componentes

Con los diagramas de componentes, en esta parte del diseño describiremos los elementos y relaciones que aparecerán en las páginas Web (páginas servidor, páginas cliente, formularios, frames, links, etc.) a partir de los diagramas anteriores (de caso de usos y de secuencias) utilizando la notación propuesta por Conallen para diseñar aplicaciones Web. La explicación completa se encuentra en el tema II "Marco teórico". A continuación se presentan los diagramas de componentes para los casos de uso.

Diagrama de componentes "Alta de documento"

En la Figura III.36 se muestra básicamente la estructura para añadir un documento en el sistema. El área de trabajo de la página se divide en dos marcos: El primer marco "Menú", que se encontrará

en la parte superior del frameset, cambia en sus opciones dependiendo si el usuario es administrador, secretaria ó académico. Una vez que el usuario abra la página de alta de documento, esta y sus páginas subsecuentes apuntarán al marco "Pantalla" (parte inferior del frameset) que es el marco donde se muestra la mayor parte de la información de la página.

Cuando el usuario haya seleccionado la página "alta de documento", se cargará en el un formulario para la captura de las características del documento dependiendo si el tipo de documento es Tesis o no, ya que el formulario cambiará. Esta información es suministrada a la página del servidor llamada "alta de documento 2" o "alta de tesis" según sea el caso, la cual se encargará de procesar los datos y finalmente enlazarse con la base de datos para dar de alta el documento.

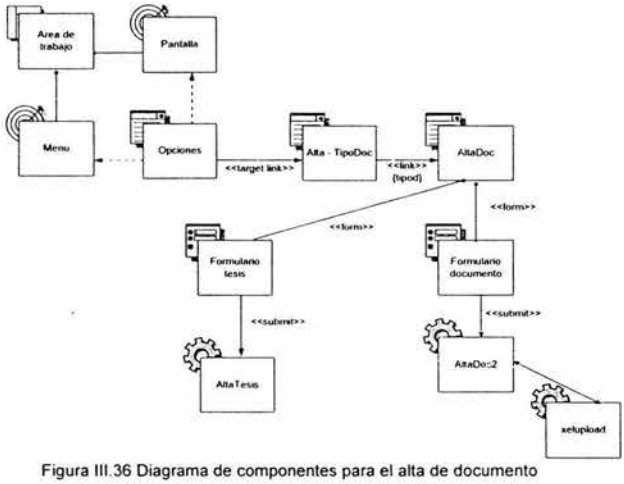


Figura III.36 Diagrama de componentes para el alta de documento

Los diagramas de componentes subsecuentes son autodescriptivos.

Diagrama "Alta de un nuevo tipo de documento"

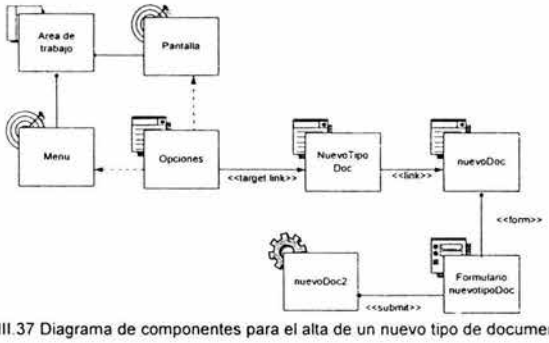


Figura III.37 Diagrama de componentes para el alta de un nuevo tipo de documento

Diagrama "Alta de usuario"

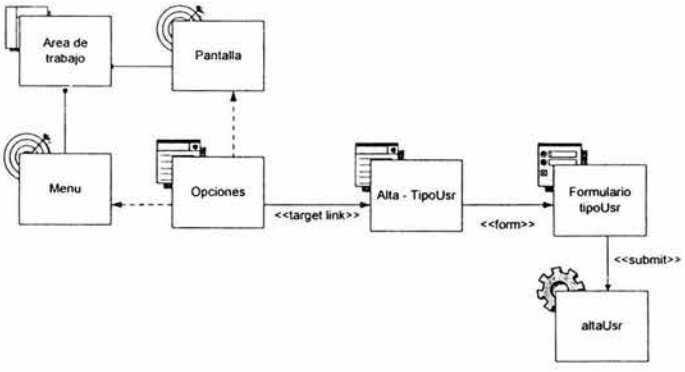


Figura III.38 Diagrama de componentes para el alta de usuarios

Diagrama "Alta de un nuevo tipo de usuario"

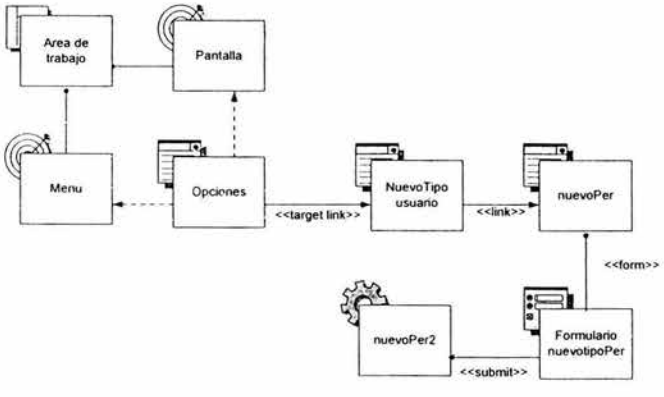


Figura III.39 Diagrama de componentes para el alta de un nuevo tipo de usuario

Diagrama "Consulta de documento"

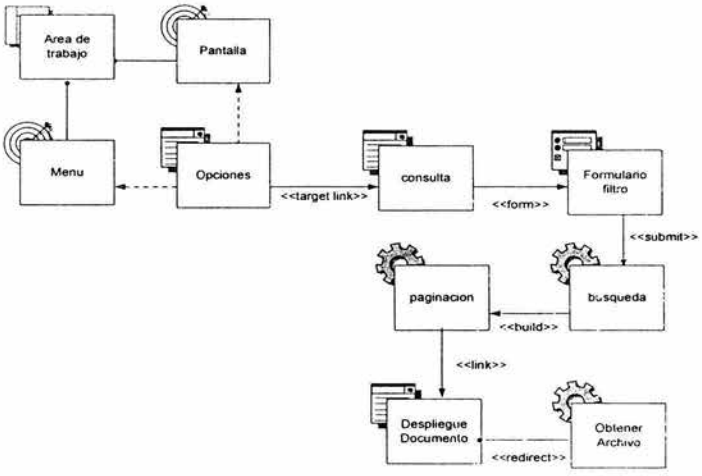


Figura III.40 Diagrama de componentes para la consulta de documentos

Diagrama "Consulta de usuarios"

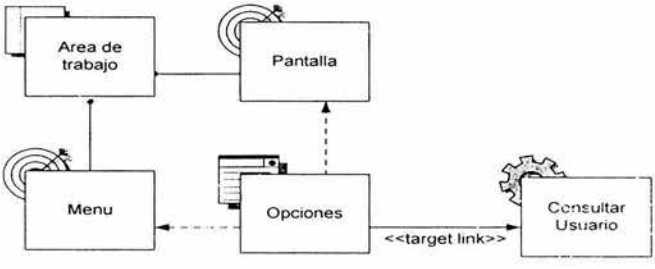


Figura III.41 Diagrama de componentes para la consulta de usuarios

Diagrama "Consulta de buzón"

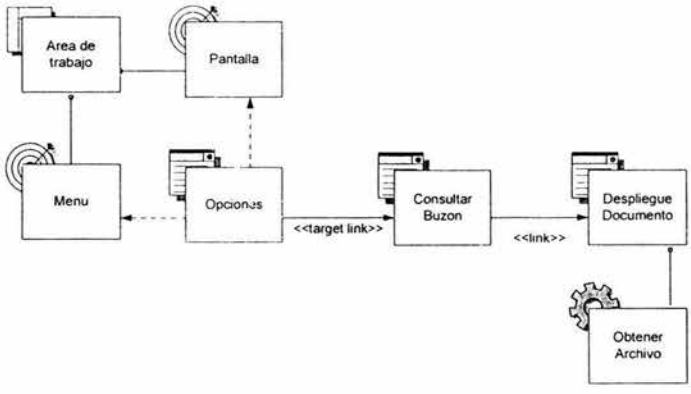


Figura III.42 Diagrama de componentes para la consulta de buzón

Diagrama "Modificación de documento"

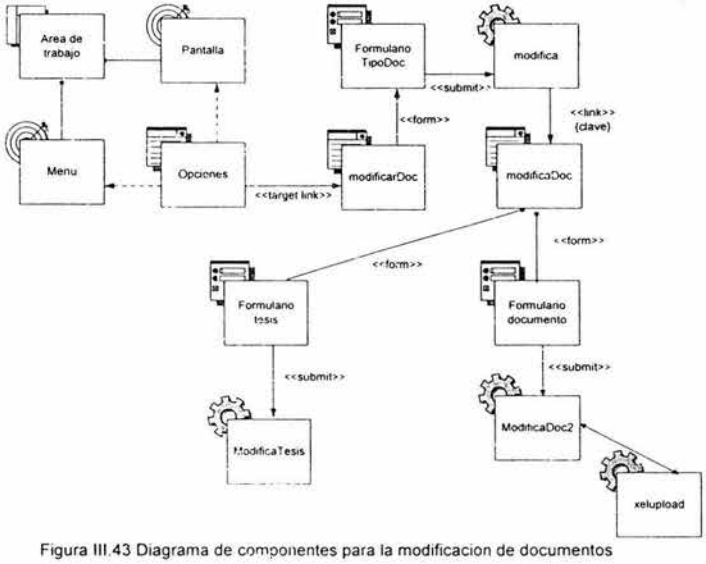


Figura III.43 Diagrama de componentes para la modificacion de documentos

Diagrama "Modificación de usuario"

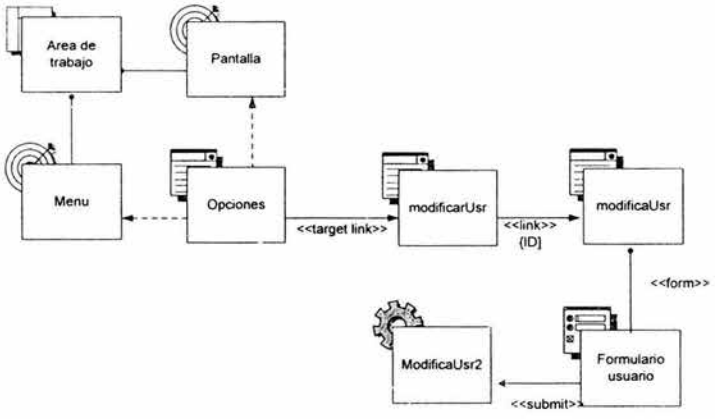


Figura III.44 Diagrama de componentes para la modificación de usuarios

Diagrama "Eliminación de documento"

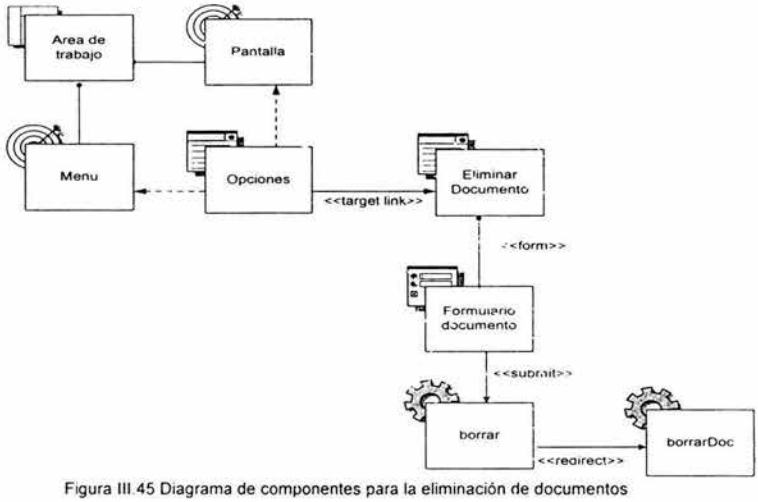


Figura III.45 Diagrama de componentes para la eliminación de documentos

Diagrama "Eliminación de usuarios"

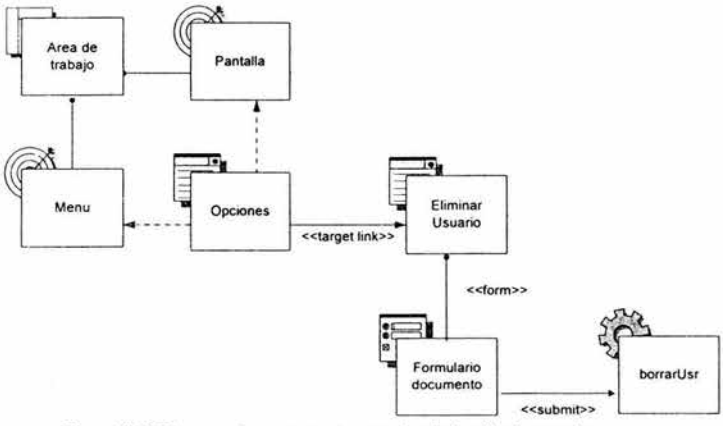


Figura III.46 Diagrama de componentes para la eliminación de usuarios

Diagrama "Enviar documento al buzón de académico"

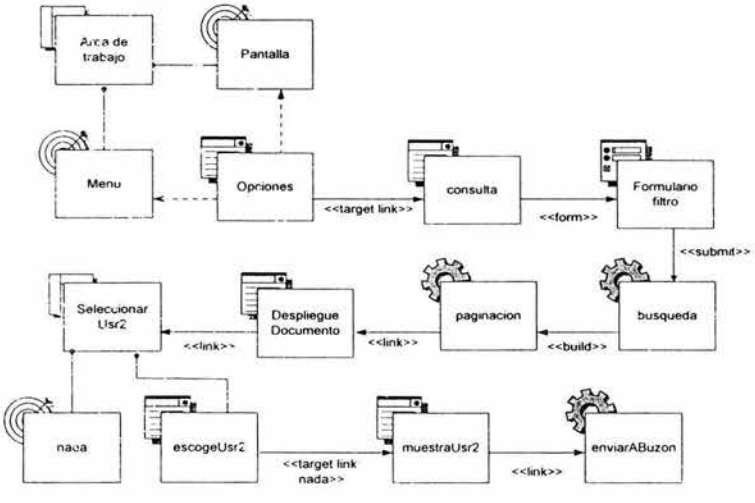


Figura III.47 Diagrama de componentes para el envió de documentos al buzón de académicos

Diagrama "Enviar e-mail"

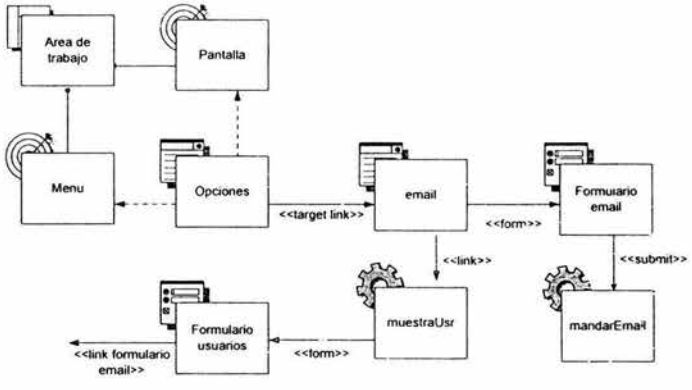


Figura III.48 Diagrama de componentes para el envío de e-mail

Diagrama "Chat"

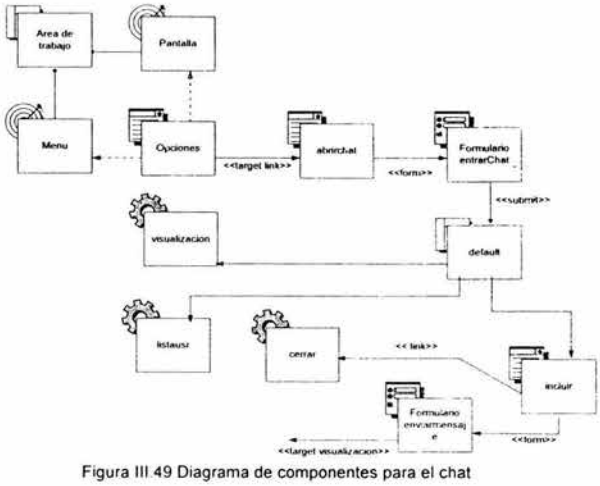


Figura III.49 Diagrama de componentes para el chat

Diagrama "Validación de usuarios"

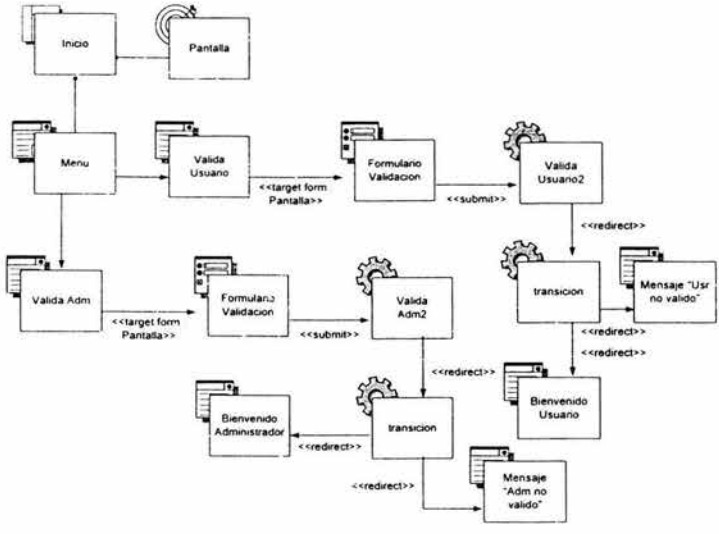


Figura III.50 Diagrama de componentes para la validación de usuarios

Diagrama "Salir del sistema"

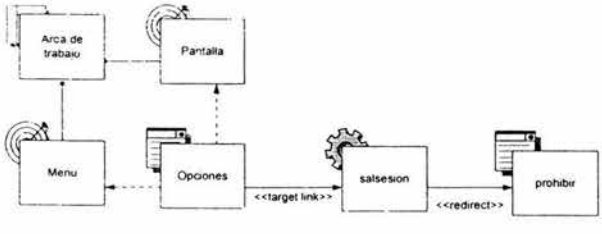


Figura III.51 Diagrama de componentes para salir del sistema

3.6 Diagrama de clases

Para finalizar con el modelado a través de los diagramas UML, se realizará el diagrama de clases donde se mostrará los elementos de la página Web como si fueran objetos de clases software, incluyendo los atributos (variables de aplicación y de sesión) y los métodos (funciones javascript, asp, asp.net)

A continuación se presentan los diagramas de clases para los casos de uso.

Diagrama "Alta de documento"

En este diagrama clases, como se muestra en la Figura III.52, se definen los atributos y métodos necesarios de las clases (páginas Web) relacionadas con la función de dar de alta un documento en el sistema. El funcionamiento de esta parte del sistema es el siguiente:

El usuario selecciona el tipo de documento para ser dado de alta. Introduce los campos necesarios y estos son validados por una página del lado del cliente "AltaDoc" a través de métodos como DateFormat() que se encarga de verificar que la fecha introducida sea válida y de ChecarDoc() que se encarga de verificar que campos como titulo, autor, entre otros no sean nulos. También se guarda el tipo de documento que se trata y el nombre para un seguimiento del mismo.

Una vez que se envía el documento para su almacenamiento en la base de datos, este es tratado por una página del lado del servidor "AltaDoc2" para generar su clave única a través del método GenerarClave() y posteriormente se almacena en la base de datos por medio del método bdAlmacenardatos().

Los diagramas de secuencias subsecuentes son autodescriptivos.

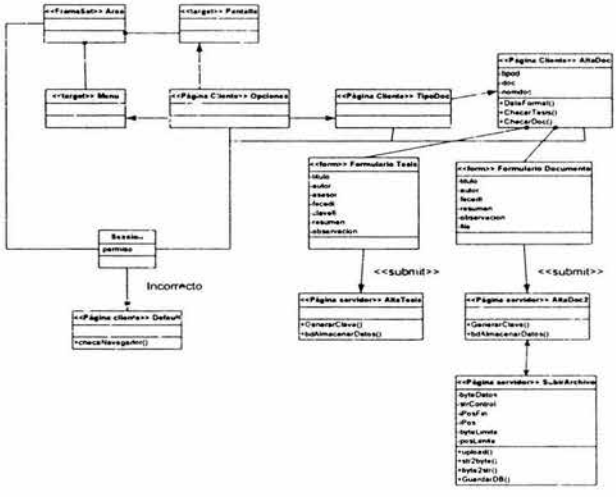


Figura III.52 Diagrama de clases para el alta de documentos

Diagrama "Alta de un nuevo tipo de documento"

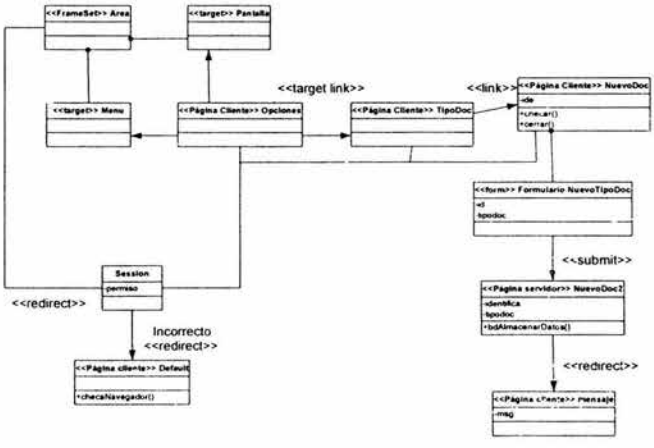


Figura III.53 Diagrama de clases para el alta de un nuevo tipo de documento

Diagrama "Alta de usuario"

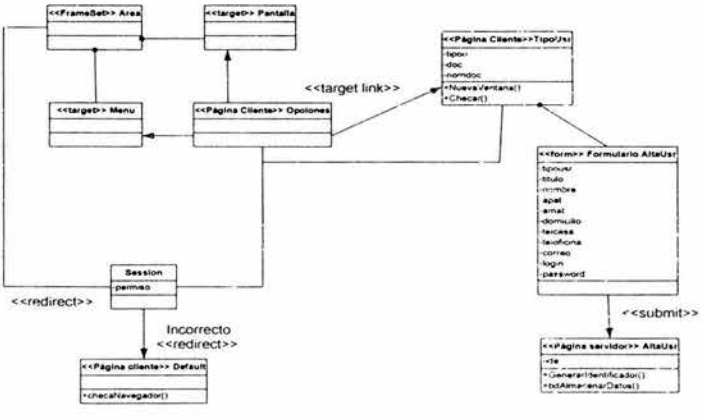


Figura III.54 Diagrama de clases para el alta de usuarios

Diagrama "Alta de un nuevo tipo de usuario"

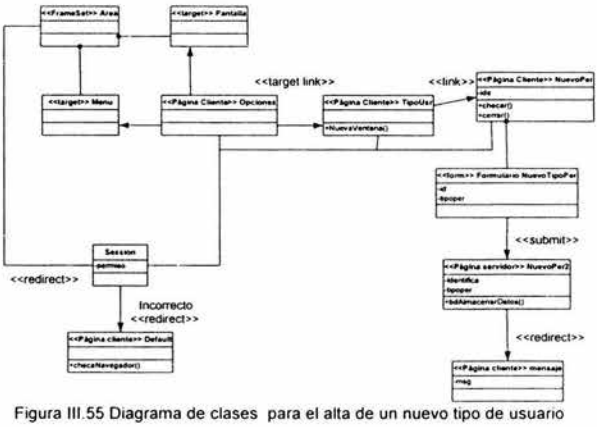


Figura III.55 Diagrama de clases para el alta de un nuevo tipo de usuario

Diagrama "Consulta de documento"

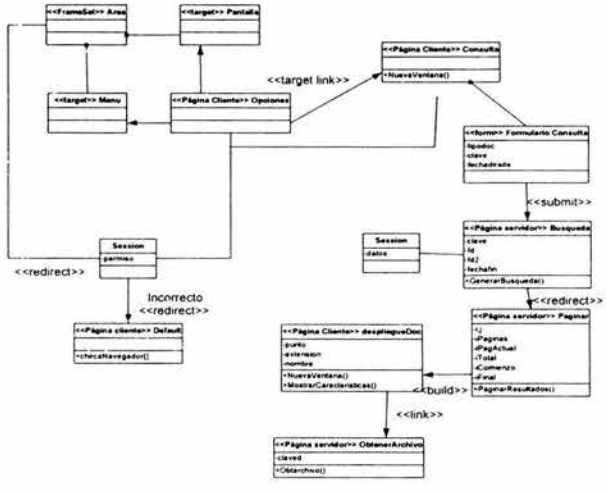


Figura III.56 Diagrama de clases para la consulta de documentos

Diagrama "Consulta de usuarios"

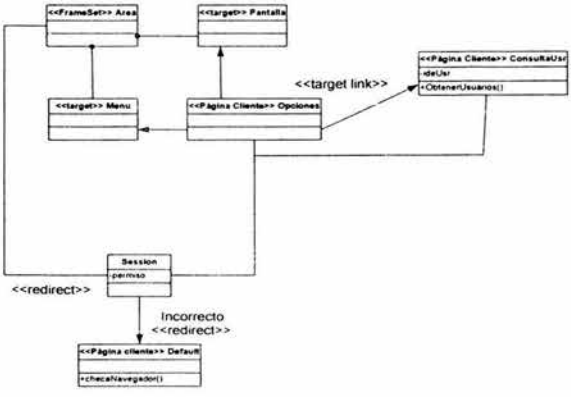


Figura III.57 Diagrama de clases para la consulta de usuarios

Diagrama "Consulta de buzón"

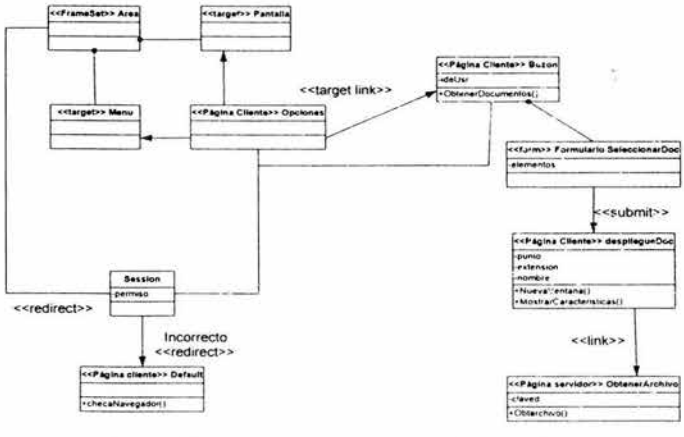


Figura III.58 Diagrama de clases para la consulta de buzón

Diagrama "Modificación de documento"

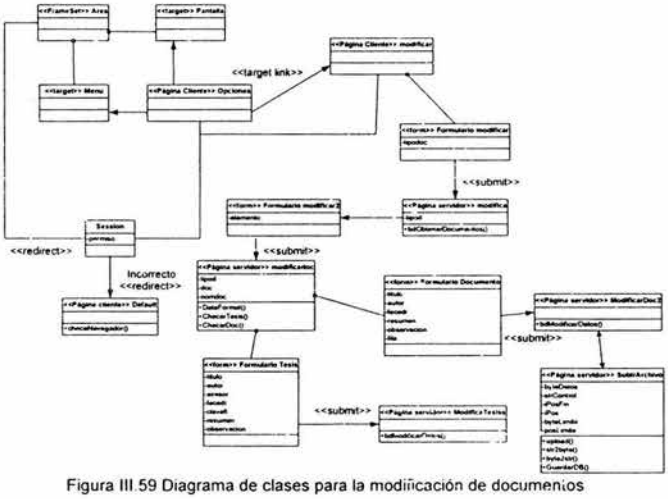


Figura III.59 Diagrama de clases para la modificación de documentos

Diagrama "Modificación de usuario"

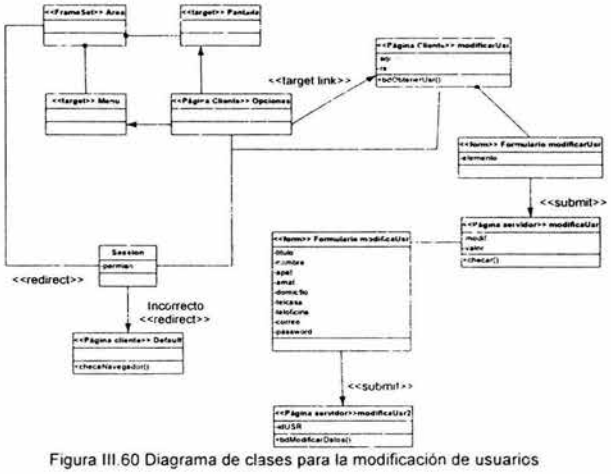


Figura III.60 Diagrama de clases para la modificación de usuarios

Diagrama "Eliminación de documento"

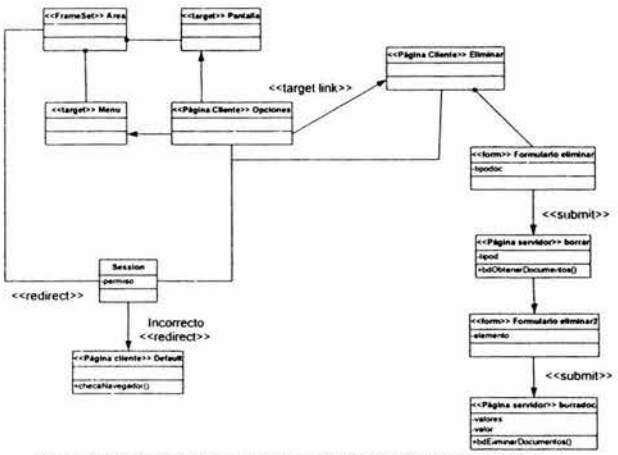


Figura III.61 Diagrama de clases para la eliminación de documentos

Diagrama "Eliminación de usuarios"

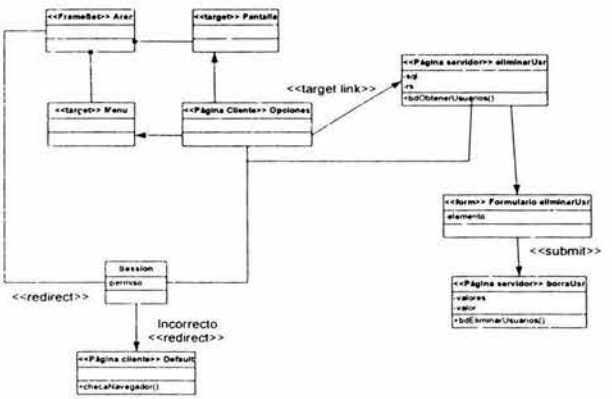


Figura III.62 Diagrama de clases para la eliminación de usuarios

Diagrama "Enviar documento al buzón de académico"

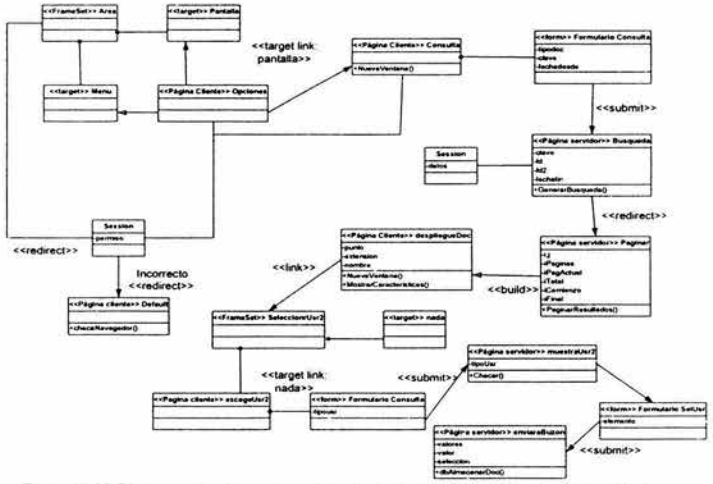


Figura III.63 Diagrama de clases para el envío de documentos al buzón de académicos

Diagrama "Enviar e-mail"

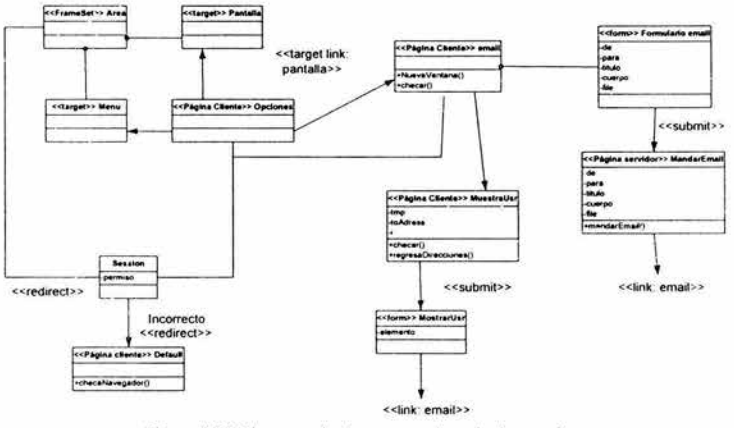


Figura III.64 Diagrama de clases para el envío de e-mail

Diagrama "Chat"

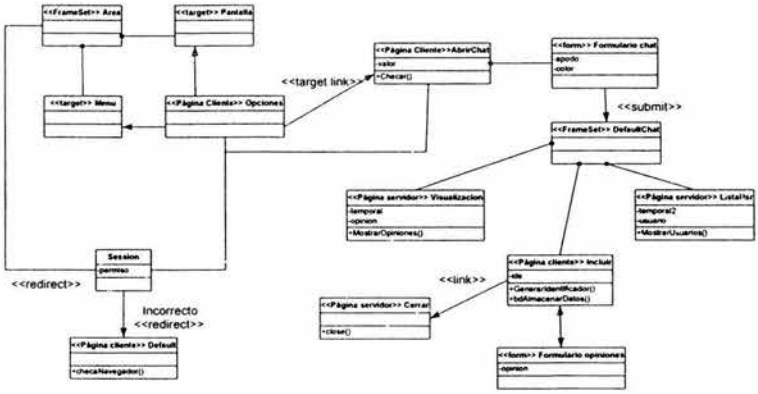


Figura III.65 Diagrama de clases para el chat

Diagrama "Validación de usuarios"

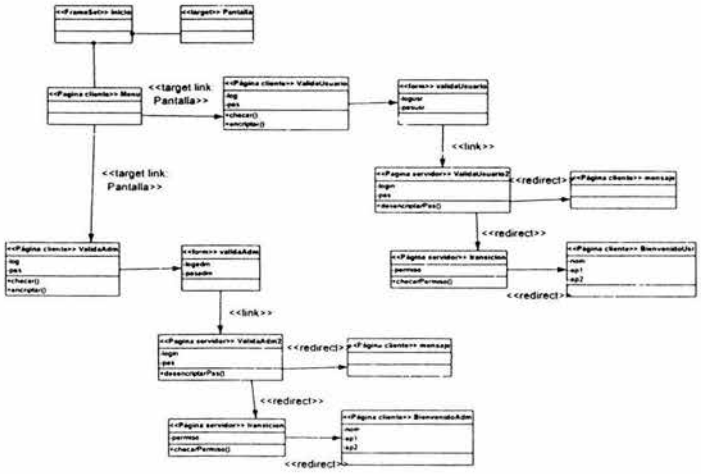


Figura III.66 Diagrama de clases para la validación de usuarios

Diagrama "Salir del sistema"

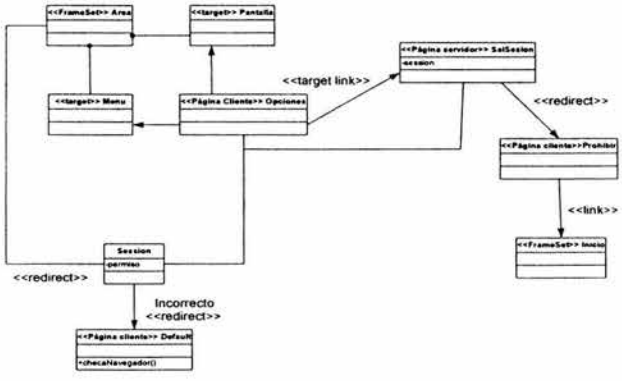


Figura III.67 Diagrama de clases para salir del sistema

3.7 Diseño de la base de datos

3.7.1 Introducción

Uno de los objetivos fundamentales de esta tesis es la implementación de una base de datos que contenga toda la información referente a la información generada y recibida por el Departamento de Computación utilizando las herramientas disponibles en el Laboratorio Microsoft. Por tal motivo, se tuvo que implementar y diseñar la base de Datos en SQL Server 2000.

En todo diseño e implementación de base de datos, se realiza tres fases o tres estudios. El primero, es el estudio conceptual de la base de datos, donde se definen las tablas y sus relaciones, el siguiente estudio es el lógico, que se definen los campos de las tablas y sus dependencias con otros campos. Por último se tiene el estudio físico de la base de datos, con el cual se implementa físicamente la base de datos, utilizando por ello el lenguaje SQL (en este caso una variante Transact – SQL).

Las primeras dos fases serán tratados con detenimiento en este capítulo. La fase tres "estudio físico de la base de datos" será tratado en el capítulo IV "Desarrollo, Implementación y Pruebas del Sistema".

3.7.2 Diseño Conceptual

Los modelos Conceptuales de datos permiten representar la realidad a un alto nivel de abstracción. Pero estos modelos no pueden representar todas las propiedades de la realidad, por lo que se hace necesario la utilización de esquemas que lo complementen. Por tal motivo el modelo debe ser claro y expresivo, y apoyarse en un entorno gráfico de representación y modelado.

En el diseño se ha intentado implementar tablas que se aproximasen lo más posible a la realidad del sistema, teniendo bastante cuidado en no construir tablas superfluas e innecesarias, que harían que la base de datos estuviera mal diseñada. En esta implementación, se ha tenido en cuenta todos los actores que intervienen en el sistema, debido a que en el modelado de la base de datos, uno de sus factores base de información, es el conocimiento exhaustivo del entorno que rodea al sistema.

La Figura III.68 representa el esquema conceptual de la base de datos, donde se esquematiza cada una de las tablas que formarán la base de datos del sistema a través de rectángulos, así como su posible relación de unas con otras a través de rombos.

De esta manera se pueden tener los siguientes escenarios:

- Un documento pertenece a un Tipo de documento único.
- Un documento tiene un archivo.
- Un documento tiene el registro de la fecha en el cual fue dado de alta.
- Un usuario tiene un permiso único (rol en el sistema).
- Un usuario tiene un buzón.
- Un usuario tiene el registro de la fecha en el cual fue dado de alta.

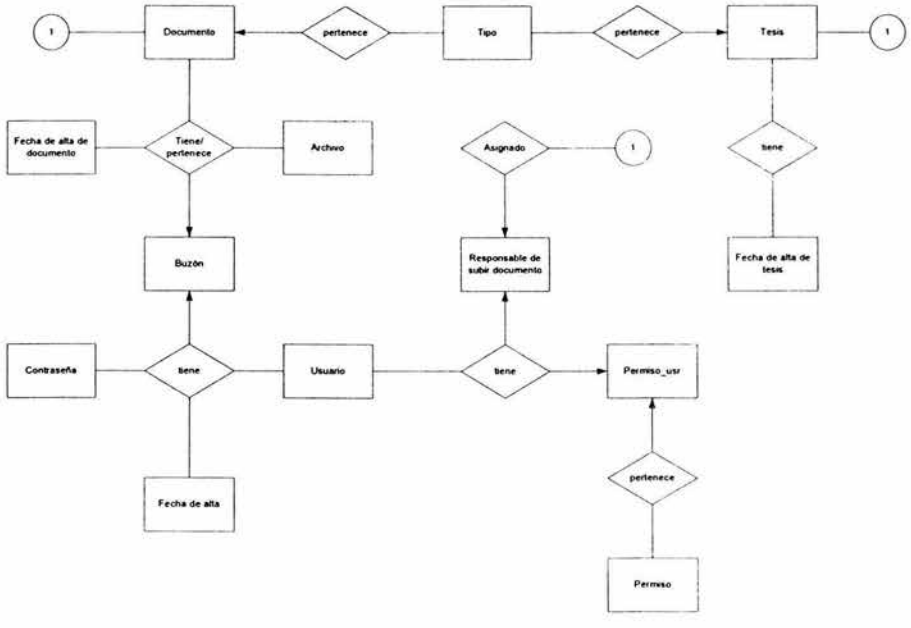


Figura III.68 Esquema conceptual de la base de datos del sistema

3.7.3 Diseño Lógico

El objetivo del diseño lógico es convertir un esquema conceptual en un esquema lógico que se ajuste al sistema de gestión de la base de datos a utilizar. Para ello se lleva a cabo el proceso de normalización (Ver capitulo II "Marco Teórico" en su parte "Dependencias funcionales") que es un proceso de conversión de las relaciones entre entidades.

Es en esta fase donde, se realiza el estudio profundo de cada una de las tablas, atribuyéndole campos y dependencias con otras tablas. Un buen diseño en está parte, forjaría una buena futura implementación, debido a que las tablas representan de forma factible la representación o modelado de un objeto o entidad de la vida real y como tal la definición de sus atributos, es un tema de importancia máxima.

Las tablas no son un objeto aislado de la base de datos, por lo que en toda base relacional, se deben de definir las posibles relaciones y dependencias que puedan existir entre ellas. Sólo así se puede modelar realmente el entorno del sistema o incluso el sistema. Los índices de las tablas, es otro tema que se tratará en este apartado. Existen dos tipos de índices en una tabla, las claves primarias, las cuales identifican a una instancia de una tabla y las claves ajenas, con las cuales que se pueden relacionar diferentes tablas con un nexo en común.

La Tabla III.19 muestra las tablas que formarán la base de datos del sistema:

Tabla	**Descripción**	**Llave primaria**	**Tablas relacionadas**
tipo_doc	Contiene los tipos de documentos válidos en el sistema.	tipo_id	car_doc car_tes
car_doc	Datos acerca de los documentos.	clave_doc	tipo_doc fechalta_doc archivo responsable_altadoc Buzon
fechalta_doc	Datos sobre la fecha de alta del documento en el sistema.		car_doc
archivo	Contiene las características del archivo del documento.		car_doc
car_tes	Datos acerca de las tesis de computación	clave_tes	tipo_doc fechalta_tes responsable_altadoc
fechalta_tes	Datos sobre la fecha de alta de la tesis en el sistema.		car_tes
usuario	Datos acerca de los usuarios del sistema.	usuario_id	fechalta_usr buzon contraseña permiso_usr responsable_altadoc
permiso	Tipos de permisos válidos en el sistema.	permiso_id	permiso_usr
permiso_usr	Asignación de permiso al usuario del sistema.		usuario permiso
contraseña	Datos para validar al usuario en el sistema.		usuario
fechalta_usr	Datos sobre la fecha de alta del usuario en el sistema.		usuario

buzon	Referencia a los documentos personales de cada académico.		car_doc usuario
Responsable_altadoc	Datos sobre el usuario responsable de subir un documento.		car_tes car_doc usuario
Mensaje	Mensajes diversos para errores, advertencias y sugerencias.	mensaje_id	

Tabla III.19 Tablas que formarán la base de datos

3.7.4 Diccionario de datos

El diccionario de datos es una herramienta que sirve para identificar y clasificar los datos almacenados en la base de datos. Un diccionario de datos consiste en archivos, registros y campos que contienen información descriptiva de los datos contenidos en la base de datos, como por ejemplo, cuántas y cuales son las columnas de las tablas y los tipos de datos que son validos para cada columna.

En el diccionario de datos no se especifican los valores de los datos, sino que definen el tipo de valor que debe ir en cada campo.

La Tabla III.20 muestra el diccionario de datos de la base de datos del sistema:

Tabla	Campo	Tipo de dato	Tamaño	Descripción
tipo_doc	tipo_id	Numérico	2	Identificador para el tipo de documento.
	tipo_doc	Cadena	50	Nombre del tipo de documento.
car_doc	clave_doc	Alfanumérico	10	Clave que identifica al documento
	tipo_id	Numérico	2	Llave foránea que hace referencia al tipo de documento.
	titulo	Cadena	250	Contiene el titulo del documento.
	autor	Cadena	50	Contiene el autor del documento.
	fecha_edi	Date	4	Guarda la fecha en que se publicó el documento.
	resumen	Cadena	1000	Contiene un

				resumen del documento,
	observación	Cadena	250	Contiene observaciones del documento.
	esbuzon	Bit	1	Testifica si un documento pertenece al buzón de un académico
fechalta_doc	clave_doc	Alfanumérico	10	Llave foránea que hace referencia a la clave del documento.
	fecha	Date	8	Guarda la fecha en que se almacenó el documento en el sistema.
	hora	Cadena	15	Guarda la Hora en que se almacenó el documento en el sistema.
archivo	clave_doc	Alfanumérico	10	Llave foránea que hace referencia a la clave del documento.
	archivo	Binario	16	Guarda el archivo de un documento.
	nombre	Cadena	255	Contiene el nombre del archivo.
	extension	Cadena	5	Contiene la extensión del archivo.
	tipo	Cadena	50	Guarda el tipo MIME del archivo.
	tamano	Numérico	4	Guarda el tamaño del archivo.
car_tes	clave_tes	Alfanumérico	10	Clave que identifica a la tesis.
	tipo_id	Numérico	2	Llave foránea que hace referencia al tipo de documento.

	titulo	Cadena	250	Contiene el titulo de la tesis.
	autor	Cadena	50	Contiene el autor o autores de la tesis
	fecha_edi	Date	4	Guarda la fecha en que se termino la tesis.
	asesor	Cadena	50	Contiene el nombre del director de tesis.
	clave_fi	Alfanumérico	10	Guarda la clave que asigna el departamento de computación a una tesis.
	resumen	Cadena	1000	Contiene un resumen de la tesis.
	observacion	Cadena	250	Contiene observaciones de la tesis.
fechalta_tes	clave_tes	Alfanumérico	10	Llave foránea que hace referencia a la clave de la tesis.
	fecha	Date	8	Guarda la fecha en que se almacenó la tesis en el sistema.
	hora	Cadena	15	Guarda la hora en que se almacenó la tesis en el sistema.
usuario	usuario_id	Numérico	4	Clave que identifica a un usuario.
	titulo	Cadena	50	Grado de estudio del usuario.
	nombre	Cadena	30	Contiene el nombre del usuario.
	ap_pat	Cadena	30	Contiene el apellido paterno

				del usuario
	ap_mat	Cadena	30	Contiene el apellido materno del usuario
	domicilio	Cadena	50	Contiene el domicilio del usuario
	telcasa	Numérico	20	Contiene el número telefónico de la casa del usuario.
	teloficina	Numérico	20	Contiene el número telefónico de la oficina del usuario.
	email	Cadena	50	Contiene el correo electrónico del usuario.
permiso	permiso_id	Numérico	2	Identificador para el tipo de permiso.
	tipo_per	Cadena	50	Contiene el nombre del tipo de permiso.
permiso_usr	usuario_id	Numérico	2	Llave foránea que hace referencia al identificador de usuario.
	permiso_id	Numérico	2	Llave foránea que hace referencia al identificador de un tipo de permiso.
contrasena	usuario_id	Numérico	2	Llave foránea que hace referencia al identificador de usuario.
	login	Alfanumérico	50	Contiene el login del usuario.
	contrasena	Binario	50	Contiene el password del usuario
fechalta_usr	usuario_id	Numérico	2	Llave foránea que hace referencia al identificador de

				usuario.
	fecha	Date	8	Guarda la fecha en que se dio de alta al usuario.
	hora	Cadena	15	Guarda la hora en que se dio de alta al usuario.
buzon	usuario_id	Numérico	2	Llave foránea que hace referencia al identificador de usuario.
	clave_doc	Alfanumérico	10	Llave foránea que hace referencia a la clave del documento.
	revisado	Bit	1	Testifica si un documento que pertenece al buzón de un usuario ha sido revisado.
responsable_altadoc	usuario_id	Numéricc	2	Llave foránea que hace referencia al identificador de usuario.
	clave_doc	Alfanumérico	10	Llave foránea que hace referencia a la clave del documento.
	clave_tes	Alfanumérico	10	Llave foránea que hace referencia a la clave de la tesis.
mensaje	mensaje_id	Numérico	1	Identificador para un tipo de mensaje.
	descripción	Cadena	75	Contiene una descripción de un mensaje en particular

Tabla III.20 Diccionario de datos

3.8 Estimación del costo para el desarrollo del sistema

En la estimación del costo y del esfuerzo del software intervienen demasiadas variables - humanas, técnicas, de entorno, políticas - que pueden afectar al coste final del software y al esfuerzo aplicado para desarrollarlo.

Algunos métodos, como el Bottom-up, Top-down y el modelo COCOMO (COnstructive COst MOdel) básico, están pensados y diseñados para lenguajes de tercera generación, donde la unidad de productividad es la línea de código. Así en este proyecto orientado a objetos, dicha medida no es adecuada. Por ejemplo, si se utiliza un lenguaje visual, se puede implementar una pequeña aplicación sin escribir ni una línea de código, al proporcionar estos entornos de programación, ayudas a la programación y componentes estándares que facilitan la generación semiautomática del código del programa.

En el modelo COCOMO, hay algunas variantes que consideran otros factores adicionales a las líneas de código, como puede ser la evaluación del producto, del hardware, del personal, etc. Estas variantes son el modelo intermedio COCOMO y el avanzado COCOMO.

Por otra parte, las métricas orientadas a la función son métricas indirectas del software, centrándose en la funcionalidad o la utilidad de la aplicación o el programa, utilizando por ello los conceptos de punto de función (PF) y otra variante para proyectos de gestión, el punto característico (PC). Estas medidas subjetivas no son del todo fiables para realizar mediciones concretas de un determinado software, pero nos sirven de indicador. Existe una variante de estas métricas 3D para proyectos de análisis orientados a objetos, denominada métrica de punto de función con dimensiones orientadas a objetos (PF DOO). Pero estas métricas son demasiado subjetivas para tratarse en un proyecto donde falta la implementación del código y sólo posee el modelado del mismo.

En el paradigma de orientado a objeto se deben de realizar o aplicar diferentes cambios en la técnica de COCOMO, debido a que la naturaleza del problema es diferente. Se debe de considerar las siguientes ecuaciones como fundamentales para extraer información del método.

$$EMOM = a_i (UPOO)^{b_i} \text{ (Esfuerzo Nominal)}$$
$$E = EMOM * FAE \text{ (Esfuerzo Real)}$$

UPOO representa a la unidad de producción de software orientado a objetos. Esta unidad no se mide en líneas de código (LDC), a no ser que se tome como líneas de código las del código de la máquina de los módulos ejecutables obtenidos finalmente. En los entornos de programación orientados a objetos, se hace uso intensivo y extensivo de componentes y controladores. Por tanto, se enmascaran las LDC puesto que, parte de los componentes se añaden en tiempo de compilación en la etapa de generación del lenguaje objeto, otra parte se añade en la etapa de enlace y el resto se añade en tiempo de ejecución. Así pues, es necesario determinar los factores finales que determinan la unidad de producción orientado a objeto.

El concepto fundamental de los proyectos orientados a objetos, es por excelencia el objeto. Por lo tanto el número de objetos generados en el proyecto es uno de los factores finales para la medición. Pero entre los objetos se deben de distinguir entre objetos de negocio y los objetos de infraestructura. Los objetos de negocio son los que realmente se construyen porque representan a las entidades del problema. En cambio los objetos de infraestructura forman el conjunto de objetos de interfaz y de base de datos, siendo los objetos estándar que proporciona los entorno de programación utilizados, siendo estos configurados y no construidos por el desarrollador de la herramienta. Después de observar la diferencia entre los tipos de objeto, se puede decir que el tiempo dedicado ha ambos es distinto.

Otro aspecto a tener en cuenta en los objetos es la estructura de los mismos, sus operaciones y atributos. Se tiene que tener en cuenta que el esfuerzo para obtener un atributo es diferente al esfuerzo obtenido por una operación.

Las interfaces, son elementos estándares en los proyectos orientados a objetos, al ser la gran mayoría de estos, visuales. En estas interfaces podemos distinguir tres elementos: formularios, que representarían las pantallas para la introducción y modificación de los datos, los informes es la otra clase, éstos representarían el volcado de información en la pantalla y por último están las consultas, las cuales son procedimientos para mandar información al sistema.

La última medida a tener en cuenta es la Unidad de Producción Orientado a Objetos UPOO, que representa el número de elementos ejecutables que pueda poseer el sistema. Estos módulos contienen objetos, formularios, informes, archivos, etc. que forman todo un conjunto que desempeña una o varias funciones relacionadas. En la Tabla III.21 se muestra la Tabla de Asignación de Coeficientes según el Tipo de Objeto.

Tipo	Elemento	N(ai)	Peso
Objeto	1: Objetos de negocio	2	15
	2: Objetos de infraestructura configurables	15	8
	3: Objetos de infraestructura no configurables	0	3
Partes estructurales	1: Atributos	10	3
	2: Operaciones	3	10
Interfaz	1: Formularios	8	15
	2: Informes	1	14
	3: Consultas	2	7
Ejecutables	1: Módulos ejecutables	0	10
	2: Módulos de interfaz	1	10
	3: Módulos de ayuda	1	5
Total		43	100

Tabla III.21 Tabla de Asignación de Coeficientes según el Tipo de Objetos

La unidad de producción se calcula de la forma siguiente:

$$UP_{OO} = \Sigma_i[P(a_i)^* N(a_i)]$$

Donde $P(a_i)$ es el peso del factor y $N(a_i)$ es la cantidad de ocurrencias del factor identificadas en el análisis inicial. Entonces tendremos el siguiente resultado.

$$UPOO = 373$$

Ahora ya podemos aplicar el resto de formulas de COCOMO.

Las siguientes formulas a aplicar serán las siguientes:

$$EMOM = a\ (UPOO)b$$
$$E = EMOM * FAE$$

Donde si consideramos un Modelo COCOMO básico de tipo orgánico el valor de a = 2.40 y el de b = 1.05, de acuerdo a la Tabla III.22, entonces el resultado será de:

$$EMOM = 2{,}4\ (373)1{,}05 = 940$$

El FAE (Factor de Aplicación de Esfuerzo), es otro de los factores que intervienen en la obtención del esfuerzo. Utilizaremos un FAE de 0,75 (que es el factor más bajo para el software, debido a que no esta implementado el mismo) entonces el resultado será de:

$$E = 6688 * 0,75 = 705 \text{ hombres-mes}$$

Luego la productividad será:

$$P = UPOO / E = 373/ 705 = 0.53 \text{ UP/Hombres-mes}$$

La duración será para un COCOMO orgánico (c = 2,50 y d =0,38). D= c * (E)d entonces la duración será:

$$D = 2,50 (705)0,38 = 30 \text{ meses} = 2.5 \text{ años}$$

Según este estudio se necesitaría 2.5 años, para implementar, diseñar y modelar el sistema. Si el costo mensual de un programador es de 8000.00 pesos, por lo tanto el costo final del sistema será de: 30 x 8000 = 240 000.00 pesos.

Proyecto de software	a_b	b_b	c_b	d_b
Orgánico	2.4	1.05	2.5	0.38
Semiacoplado	3.0	1.12	2.5	0.35
Empotrado	3.6	1.20	2.5	0.32

Tabla III.22 Modelo COCOMO básico

CAPÍTULO

IV

IMPLEMENTACIÓN Y PRUEBAS DEL SISTEMA

CONTENIDO

En el capítulo III se tienen identificadas las clases del sistema que han sido parte del análisis y diseño del sistema. En este capitulo se procederá a su implementación. Pero antes de llevarlo a cabo, se explicarán las opciones tecnológicas que exige el sistema.

4.1 Arquitectura cliente-servidor

Normalmente, las aplicaciones de cooperación pueden clasificarse como clientes o como servidores La aplicación cliente pide servicios y datos al servidor, que responde a las peticiones del cliente. Las primeras aplicaciones de dos niveles (cliente-servidor) se desarrollaron para tener acceso a grandes bases de datos e incorporaban las reglas que se utilizaban para manipular los datos junto con la interfaz de usuario en la aplicación cliente. La tarea del servidor se limitaba a procesar tantas peticiones de almacenamiento y recuperación de datos como fuera posible.

Las aplicaciones de dos niveles realizan muchas de las funciones de los sistemas autónomos: presentan una interfaz de usuario, reúnen y procesan los datos proporcionados por el usuario, realizan el proceso pedido e informan del estado de la petición. Esta secuencia de comandos puede repetirse tantas veces como sea necesario. Como los servidores se limitan a proporcionar acceso a los datos, el cliente utiliza sus recursos locales para realizar la mayoría de los procesos. La aplicación cliente debe contener información acerca de dónde residen los datos y de cómo está organizada la base de datos. Una vez obtenidos los datos, el cliente debe darles formato y mostrarlos al usuario.

Una de las principales ventajas del modelo cliente-servidor era que, al permitir el acceso simultáneo de varios usuarios a los mismos datos, las actualizaciones realizadas desde un equipo estaban disponibles al instante para todos los demás equipos que tenían acceso al servidor. Sin embargo, al aumentar el número de clientes, el servidor se veía desbordado rápidamente por las peticiones de los clientes. Además, como gran parte de la lógica de proceso se basaba en un conjunto monolítico de aplicaciones, los cambios en las reglas empresariales producían caras y largas modificaciones en el código fuente. Aunque la facilidad y la flexibilidad de los productos de dos niveles aún es la base de muchas aplicaciones empresariales a pequeña escala, la necesidad de un acceso a los datos más rápido y de reducir el plazo de desarrollo hizo que los desarrolladores de sistemas buscaran un nuevo método para crear aplicaciones distribuidas.

4.1.1 Arquitectura de tres niveles

Las aplicaciones cliente - servidor actuales se parecen tan poco a las anteriores que tienen un nuevo nombre, aplicaciones multinivel también conocidas como arquitectura en n niveles. En este modelo, el procesamiento se distribuye entre el cliente y el servidor, y la lógica empresarial se encuentra en un nivel intermedio. La mayoría de los sistemas realizarán las tres siguientes tareas principales, que corresponden a tres niveles, o capas, del modelo de múltiples niveles:

- *Interfaz de usuario y desplazamiento.* Corresponde al nivel 1 de la Figura IV.1 que abarca todos los aspectos de la interacción con el usuario. No sólo proporciona una interfaz gráfica para que los usuarios interactúen con la aplicación, proporcionen datos y vean los resultados de las peticiones, sino que también administra la manipulación de la información y el formato de los datos que el cliente recibe. En las aplicaciones Web, el explorador realiza las tareas de esta capa.

- *Lógica empresarial.* Es el nivel 2, situado entre el nivel de la interfaz y el de datos, es el dominio del desarrollador de aplicaciones distribuidas. La lógica empresarial, que captura las reglas que controlan los procesos de la aplicación, conecta al usuario que se encuentra en un extremo con los datos que están en el otro. Las funciones que estas reglas controlan se asemejan mucho a las tareas empresariales del día a día y pueden constar de una o más tareas.

- *Servicios de datos.* Como se ve en el nivel 3 de la Figura IV.1, los servicios de datos los proporciona un almacén de datos estructurado (una base de datos de SQL Server u Oracle) o no estructurado (Microsoft Exchange, Servicios de Microsoft Message Queue Server), que administra los datos de la aplicación y proporciona acceso a ellos. Una única aplicación puede utilizar los servicios de uno o más almacenes de datos.

La arquitectura de tres niveles aísla cada uno de los principales elementos de funcionamiento, de forma que la presentación es independiente de las reglas de proceso y de la lógica empresarial que, a su vez, es independiente de los datos. Este modelo requiere muchas más tareas previas de análisis y diseño, pero reduce considerablemente los costos de mantenimiento y aumenta la flexibilidad funcional a largo plazo. La Figura IV.1 muestra las tecnologías de Microsoft que dan servicio a los diferentes niveles del nuevo diseño de sistemas.

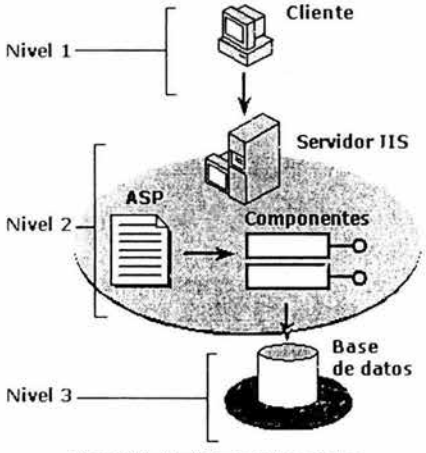


Figura IV.1 Arquitectura de tres niveles

4.1.2 Arquitectura Windows DNA

Microsoft desarrolló la Arquitectura de aplicaciones de Internet distribuidas de Windows (Windows DNA) como medio para integrar el Web con el modelo de desarrollo de múltiples niveles. Windows DNA define un marco para proporcionar soluciones que cumplan los exigentes requisitos de la informática empresarial, Internet, las redes intranet y el comercio electrónico global, al tiempo que reduce los costos generales de desarrollo y distribución.

La arquitectura Windows DNA utiliza servicios estándar basados en Windows para satisfacer los requisitos de cada nivel en una solución con múltiples niveles: la interfaz de usuario y el desplazamiento, la lógica empresarial y el almacenamiento de datos. Entre los servicios que se utilizan en Windows DNA, que se integran mediante el Modelo de objetos componentes (COM), cabe citar los siguientes:

- HTML dinámico (DHTML)
- Páginas Active Server (ASP)
- Componentes COM
- Servicios de componentes
- Servicios de Active Directory
- Servicios de seguridad de Windows®
- Servicios de Microsoft® Message Queuing
- Microsoft Data Access Components

Microsoft creó Windows DNA con protocolos abiertos e interfaces públicas, lo que permite a las organizaciones integrar fácilmente productos de terceros. Además, puesto que admite los estándares definidos por la industria para la informática en Internet, Windows DNA facilitará la respuesta de los desarrolladores ante los cambios tecnológicos. Algunas de las tecnologías que se agregaron recientemente a Windows DNA se muestran en la Figura IV.2.

Figura IV.2 Arquitectura Windows DNA

4.1.3 Plataforma .NET

La infraestructura y plataforma que .NET proporciona es una arquitectura Internet distribuida Windows (Windows DNA). .NET es la plataforma de Microsoft para servicios Web XML, la siguiente generación de software que conecta nuestro mundo de información, dispositivos y personas de una manera unificada y personalizada.

La Plataforma .NET permite la creación y uso como servicios de aplicaciones, procesos y sitios Web basados en XML, que compartan y combinen información y funcionalidad entre ellos por diseño, en cualquier plataforma o dispositivo inteligente, para proveer soluciones a la medida para empresas e individuos.

Microsoft .NET permitirá crear programas que trascienden los límites de los dispositivos y fortalecen la conectividad de Internet en sus aplicaciones. Además, es viable para todas sus aplicaciones, ya que no necesita pensar en dos infraestructuras separadas—una para aplicaciones Web y otra para aplicaciones internas o de escritorio.

Principales Componentes .NET

Desde un punto de vista tecnológico, .NET es la plataforma y las experiencias .NET construidas sobre la plataforma. La plataforma incluye:

- Herramientas – para construir aplicaciones y servicios Web XML (.NET Framework y Visual Studio.NET)
- Servers – sobre los que construir, proveer y desplegar esas aplicaciones y servicios (Windows 2000 Server y Windows 2003 Server)
- Servicios – un conjunto central de servicios .NET ensamblables (servicios "HailStorm")

- Software cliente – el software que provee dispositivos inteligentes, permitiendo a los usuarios interactuar y experimentar la plataforma .NET
- Experiencias – la combinación de los componentes de la plataforma .NET nombrados permiten experiencias de usuario más personales e integradas – experiencias .NET.

Estos componentes principales de .NET se ilustran en la Figura IV.3.

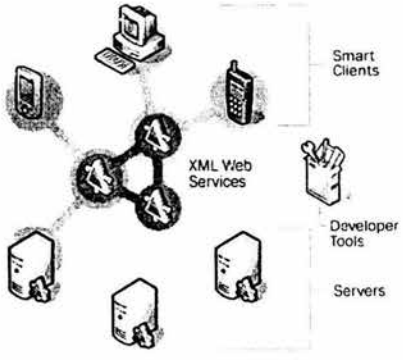


Figura IV.3 Componentes de la plataforma .NET

Arquitectura del entorno .NET
En la Figura IV.4 se observa la arquitectura básica del entorno .NET

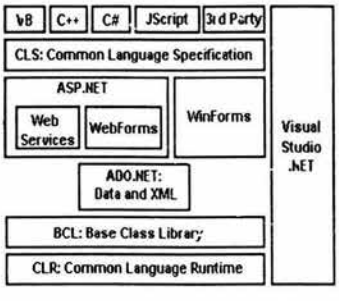


Figura IV.4 Arquitectura .NET

Estos componentes principales del Entorno .NET se describen a continuación:

- Common Language Runtime (CLR). CLR es el mecanismo de ejecución para las aplicaciones .NET. Proporciona varios servicios, incluyendo la carga y ejecución del código, aislamiento de la memoria de las aplicaciones, administración de memoria, manejo de excepciones, acceso a metadata (información mejorada de tipos), y la conversión de MSIL (Lenguaje Intermedio Microsoft) a código nativo.
- La Base Class Library (BCL). BCL proporciona un amplio conjunto de clases, lógicamente agrupadas en espacios de nombres jerárquicos que proporcionan acceso a las características más importantes del sistema operativo.
- ADO.NET. Esta es una actualización evolutiva para la tecnología de acceso a datos ActiveX® Data Objects (ADO) con mejoras importantes destinadas a la naturaleza desconectada del Web.

- ASP.NET. Esta es una versión avanzada de Active Server Pages (ASP) para el desarrollo de aplicaciones Web (utilizando Formas Web) y desarrollo de servicios Web.
- Common Language Specification (CLS). Esta es responsable de hacer que muchas de las tecnologías antes mencionadas estén disponibles para todos los lenguajes que soportan .NET Framework. CLS no es una tecnología, y no hay un código fuente para ella. Define un conjunto de reglas que proporcionan un contrato que rige la interoperabilidad entre los compiladores de lenguaje y las bibliotecas.
- Win Forms. Este modelo de programación y conjunto de controles proporciona una arquitectura sólida para el desarrollo de aplicaciones basada en Windows.
- Visual Studio.NET. Este proporciona las herramientas que le permiten explotar las características del Framework para crear aplicaciones concretas.

4.2 Herramientas de desarrollo

4.2.1 Microsoft SQL Server 2000

SQL Server es un sistema administrador para Bases de Datos relacionales basadas en la arquitectura Cliente / Servidor (RDBMS) que usa Transact-SQL para mandar peticiones entre un cliente y el SQL Server.

SQL Server usa la arquitectura Cliente / Servidor para separar la carga de trabajo en tareas que corran en computadoras tipo Servidor y tareas que corran en computadoras tipo Cliente:

- El Cliente es responsable de la parte lógica y de presentar la información al usuario. Generalmente, el cliente corre en una o más computadoras Cliente, aunque también puede correr en una computadora Servidor con SQL Server.
- SQL Server administra Bases de Datos y distribuye los recursos disponibles del servidor (tales como memoria, operaciones de disco, etc.) entre las múltiples peticiones.

La Figura IV.5 muestra la arquitectura cliente / servidor de SQL Server 2000.

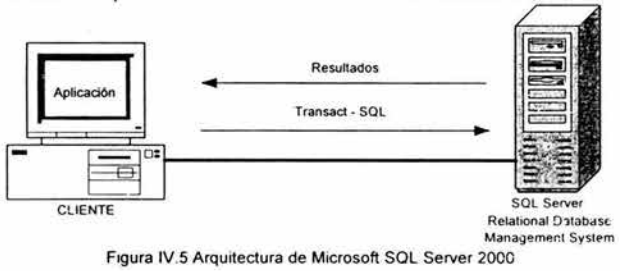


Figura IV.5 Arquitectura de Microsoft SQL Server 2000

TRANSACT - SQL
Éste es una versión de SQL (Structured Query Languaje) usado como lenguaje de programación para SQL Server. SQL es un conjunto de comandos que permite especificar la información que se desea restaurar o modificar. Con Transact – SQL se puede tener acceso a la información, realizar búsquedas, actualizar y administrar sistemas de Bases de Datos Relacionales

4.2.1.1 INTEGRACIÓN DE SQL SERVER CON WINDOWS 2003 SERVER

SQL se encuentra totalmente integrado con Windows 2003 Server y toma ventaja de muchas de sus características:

SEGURIDAD
SQL Server está integrado con el sistema de seguridad de Windows 2003 Server. Esta integración permite accesar tanto a Windows 2003 Server como a SQL Server con el mismo username y password. Además SQL Server una las características de encriptación que Windows 2003 Server para la seguridad en red. SQL Server está provisto de su propia seguridad para clientes no-Microsoft.

SOPORTE MULTIPROCESADOR
SQL Server soporta las capacidades de multiprocesamiento simétrico (SMP) de Windows 2003 Server. SQL Server automáticamente toma ventaja de cualquier procesador adicional que sea agregado al Servidor.

SERVICIOS DE WINDOWS 2003
SQL Server corre como un servicio dentro de Windows 2003 Server, permitiendo operarlo remotamente.

MICROSOFT CLUSTER SERVER
Es un componente de Windows 2003 Server.. Soporta la conexión de dos servidores, o nudos, en un cluster para aumentar las habilidades y tener un mejor manejo de la información y las aplicaciones. SQL Server trabaja en conjunto con el Cluster Server para intercambiar papeles automáticamente en caso de que el nodo primario falle.

4.2.1.2 SERVICIOS DE SQL SERVER

Los servicios de SQL Server, mostrados en la Figura IV.6, incluyen MSSQLServer, SQLServerAgent, Microsoft Distributed Transaction Coordinator (MSDTC), y Microsft Search. Aunque estos servicios de SQL generalmente corren en Windows 2003 Server, también pueden correr como aplicaciones.

MSSQLServer
Este servicio es el motor de la Base de Datos. Este es el componente que procesa todas las declaraciones de Transact-SQL y administra todos los archivos que definen a la Base de Datos dentro del Servidor. Sus características son:

- Asignar los recursos de la computadora a múltiples usuarios simultáneos.
- Previene problemas lógicos, tales como sincronización de peticiones de usuarios que desean actualizar la misma información al mismo tiempo.
- Garantiza la integridad y consistencia de datos.

SQLServerAgent
Este es un servicio que trabaja conjuntamente con SQL Server para crear y administrar tareas locales o externas; letras y operadores.

Microsoft Distributed Transaction Coordinator
MSDTC permite a los clientes incluir muchos tipos de datos en una transacción. Coordina la correcta realización de las transacciones distribuidas para asegurar que todas las actualizaciones en todos los servidores son permanentes; o en caso de errores que las modificaciones son canceladas.

Microsoft Search
Este servicio es un motor de full-text que corre como un servicio de Windows NT. El soporte Full Text involucra la habilidad de emitir queries hacia los datos y la creación y mantenimiento de índices que facilitan dichos queries.

Figura IV.6 Servicios de SQL Server

4.2.1.3 ARQUITECTURA DE SQL SERVER

COMUNICACIÓN

SQL Server usa una arquitectura de comunicación por capas para aislar aplicaciones internas de red y protocolos. Esta arquitectura permite desplegar la misma aplicación en diferentes ambientes de red. Los componentes en la arquitectura de comunicación incluyen:

- APLICACIÓN: Una aplicación es desarrollada usando una aplicación de interfaz de programación para Base de Datos (API). La aplicación no tiene conocimiento de los protocolos internos de red usados para la comunicación con SQL Server.

- INTERFAZ DE LA BASE DE DATOS: Esta es una interfaz usada por una aplicación para mandar peticiones a SQL Server y procesar los resultados devueltos por SQL Server.

- LIBRERÍA DE RED: Este es un componente de Software de comunicación que empaqueta las peticiones de la Base de Datos y los resultados para transmitirlos por medio del protocolo de red apropiado. Una librería de Red, también conocida como Net-Library, debe ser instalada tanto en el cliente como en el servidor. Tanto Clientes como Servidores pueden usar más de una Net-Library al mismo tiempo, pero deben usar una Librería de Red común para comunicarse satisfactoriamente. SQL Server soporta protocolos de red tales como TCP/IP, Novell, IPX/SPX, Banyan VINES/IP, Named Pipes,y Apple Talk ADSP.

- TABULAR DATA STREAM: (TDS) Es un protocolo por niveles de aplicación usado para la comunicación entre un Cliente y SQL Server. Los paquetes TDS son encapsulados en los paquetes de red hechos por la protocol stak usada por las Net-Libraries.

- SERVICIOS OPEN DATA: Este es un componente de SQL Server que se encarga de las conexiones de red, pasando las peticiones del cliente al SQL Server para procesar y regresar cualquier resultado a los Clientes. Open Data escucha automáticamente en todas las Net-Libraries que están instaladas en el servidor.

DESARROLLO DE APLICACIONES

Los usuarios acceden a SQL Server a través de una aplicación que está escrita con una interfaz de objetos de datos o con una API. SQL Server soporta interfaces comunes y APIs nativos de bajo nivel. Esto se muestra en la Figura IV.7.

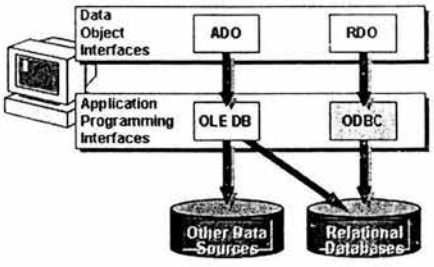


Figura IV.7 Desarrollo de aplicaciones

Interfaces de programación de aplicaciones
Una Base de Datos API define como escribir una aplicación para conectar una Base de Datos y pasar comandos a la Base de Datos. SQL Server provee soporte nativo para dos clases principales de Bases de Datos API, lo cual define la interfaz de objetos de datos que se puede usar. Las Bases de Datos API se usan para tener mayor control sobre el comportamiento y desarrollo de las aplicaciones.

- OLE DB: Esta es una interfaz de acceso a datos basada en el COM (Component Object Model). Soporta aplicaciones escritas usando OLE DB o Interfaces de Objetos de Datos basadas en OLE DB. Puede acceder a la información en SQL Server, otras Bases de Datos relacionales y otras fuentes de datos.

- OPEN DATABASE CONNECTIVITY: (ODBC) Es una interfaz por capas. Accesa directamente al protocolo SQL Server TDS y soporta aplicaciones o componentes que estén escritos usando ODBC o interfaces basadas en ODBC. Puede acceder a los datos en SQL Server, y otras Bases de Datos relacionales, pero generalmente no puede ser usado para accesar otras fuentes de datos.

Data object interfaces
En general, estas interfaces son más fáciles de usar que las Bases de Datos API pero pueden no tener tanta funcionalidad como un API.

- ACTIVE X DATA OBJECTS: (ADO) Encapsula la OLE DB API en un modelo simplificado de objetos que reduce el desarrollo de aplicaciones y los costos de mantenimiento. ADO puede ser usado a partir de Microsoft Visual Basic, Visual Basic para Aplicaciones, Active Server Pages (ASP) y el Scripting Object Model de Microsoft Internet Explorer.

- REMOTE DATA OBJECTS: (RDO) Mapea y encapsula al ODBC API. RDO puede ser usado desde Visual Basic y Visual Basic para aplicaciones.

ADMINISTRACIÓN
SQL Server provee una variedad de herramientas de administración para minimizar y automatizar las tareas administrativas rutinarias. Las declaraciones de Transact-SQL son el mecanismo interno usado para administrar SQL Server.

SQL Server puede ser administrado usando:
- Utilidades Batch incluidas en SQL Server, tales como OSQL o BCP.
- Herramientas de administración gráfica incluidas en SQL Server.
- Aplicaciones COM-compatibles: tal como Visual Basic.

La Figura IV.8 muestra un panorama general de la administración de SQL Server.

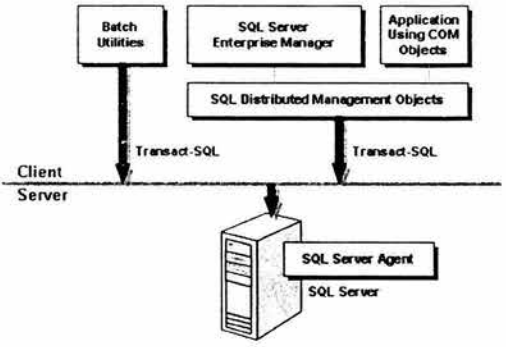


Figura IV.8 Administración SQL Server

SEGURIDAD EN SQL SERVER

SQL Server valida a los usuarios con 2 niveles de seguridad; autentificación del login y validación de permisos en la Base de Datos de cuentas de usuarios y de roles. La autentificación identifica al usuario que está usando una cuenta y verifica sólo la habilidad de conectarse con SQL Server. El usuario debe tener permiso para acceder a las Bases de Datos en el Servidor. Esto se cumple para asignar permisos específicos para la Base de Datos, para las cuentas de usuario y los roles. Los permisos controlan las actividades que el usuario tiene permitido realizar en la Base de Datos del SQL Server. La Figura IV.9 muestra un esquema de la autentificación del login de SQL Server.

AUTENTIFICACIÓN DEL LOGIN

Un usuario debe tener una cuenta para conectarse al SQL Server. Este reconoce 2 mecanismos de autentificación: Autentificación de SQL Server y de Windows. Cada uno tiene un diferente tipo de cuenta.

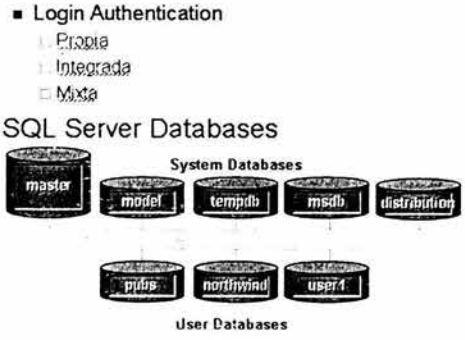


Figura IV.9 Autentificación del login

Cuando SQL Server está corriendo en Windows 2003 Server, un sistema administrador puede especificar que está corriendo en uno de 2 modos de autentificación:

- Modo de autentificación de Windows 2003 Server: Sólo está autorizada la autentificación de Windows. Los usuarios no pueden usar cuentas de SQL Server.
- Modo mixto: Cuando se usa este modo de autentificación, los usuarios se pueden conectar a SQL Server con la autentificación de Windows o con la de SQL Server.

CUENTAS DE USUARIO Y ROLES EN UNA BASE DE DATOS

Después de que los usuarios han sido autentificados, y se les ha permitido conectarse al SQL Server, deben tener cuentas en la Base de Datos. Las cuentas de usuario y los roles, identifican permisos para ejecutar tareas, tal como se muestra en la Figura IV.10.

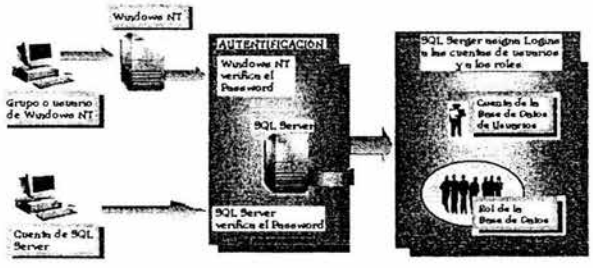


Figura IV.10 Cuentas de usuarios

SQL Server realiza los siguientes pasos cuando valida permisos:

1. Cuando el usuario realiza una acción, tal como ejecutar un comando de Transact-SQL o elegir una opción de un menú, los comandos de Transact SQL son enviadas al SQL Server.
2. Cuando SQL Server recibe un comando de Transact –SQL, checa que el usuario tenga permiso de ejecutar dicha instrucción.
3. Después, SQL realiza cualquiera de las siguientes acciones:
    a. Si el usuario no tiene los permisos adecuados, SQL Server devuelve un error.
    b. Si el usuario tiene los permisos adecuados, SQL Server realiza la acción.

4.2.2 Servidor IIS 6.0 (Internet Information Server)

IIS es una parte integrante de la arquitectura de Windows DNA. Una función importante de IIS es vincular los clientes que tienen acceso al sistema mediante el Protocolo de transferencia de hipertexto (HTTP) con los restantes servicios de Windows DNA, como DHTML, ASP, etc. Además, IIS incluye un conjunto básico de funciones que los desarrolladores de sistemas pueden ampliar para definir una arquitectura personalizada de aplicaciones. La Figura IV.11 muestra en forma general al Servidor IIS.

4.2.2.1 Funcionalidad de IIS

IIS define una funcionalidad básica que puede utilizar para crear aplicaciones Web. Páginas Active Server (ASP) y otras tecnologías de Microsoft amplían esta funcionalidad básica para crear un entorno rico para el desarrollo de aplicaciones. La funcionalidad básica del servidor se expone mediante la Interfaz de programación de aplicaciones de servidor de Internet (ISAPI). Estas son las funciones básicas proporcionadas por IIS:

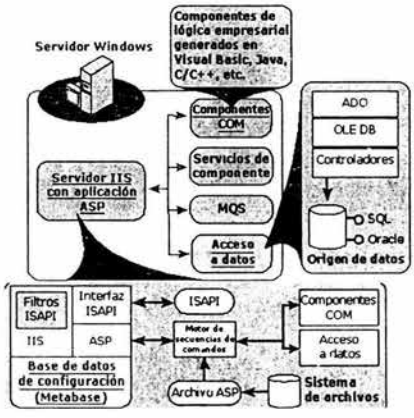


Figura IV.11 Servidor IIS

- Establecer y mantener conexiones HTTP.
- Leer peticiones HTTP y escribir respuestas HTTP.
- Modificar encabezados HTTP.
- Obtener información del certificado de cliente.
- Administrar conexiones asincrónicas.
- Asignar Localizadores de recursos universales (direcciones URL) a rutas físicas.
- Administrar y ejecutar aplicaciones.
- Transmitir archivos.

ASP amplía esta funcionalidad al proporcionar un vínculo con la arquitectura COM y, por tanto, a los demás participantes en Windows DNA. Del mismo modo, puede extender la arquitectura de IIS si define un conjunto personalizado de funciones mediante ISAPI. En la Figura IV.12 se muestra la relación existente entre la funcionalidad básica de IIS, ASP y las arquitecturas extendidas:

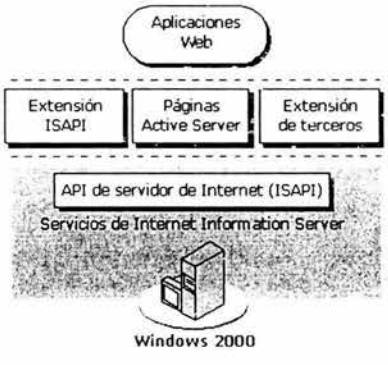


Figura IV.12 Funcionabilidad de IIS

4.2.2.2 IIS y Servicios de componentes

IIS y Servicios de componentes funcionan conjuntamente para formar la arquitectura básica para la creación de aplicaciones Web. IIS utiliza la funcionalidad proporcionada por Servicios de componentes para:

- Aislar las aplicaciones en procesos diferentes.
- Administrar la comunicación entre componentes COM (incluidos los objetos integrados de ASP).
- Coordinar el proceso de transacciones para las aplicaciones ASP transaccionales.

IIS define las aplicaciones Web como una colección de archivos de recursos agrupados en un espacio de nombres lógico. Al agrupar los recursos en aplicaciones, es posible compartir los datos en el espacio de nombres y ejecutar la aplicación en un proceso aislado.

Internamente, IIS coordina las aplicaciones aisladas mediante un objeto llamado Administrador de aplicaciones Web. Este objeto incluye una interfaz pública (IWAMAdmin) que puede utilizar para crear programas de administración de aplicaciones Web. Al ejecutar una aplicación Web en un proceso aislado, IIS utiliza los Servicios de componentes para coordinar el acceso simultáneo a los recursos y pasar información de contexto entre los componentes COM.

IIS utiliza el objeto ObjectContext de Servicios de componentes para proporcionar acceso a los objetos integrados de ASP a los componentes COM a los que ASP llama.

4.2.2.3 Procesar peticiones de IIS

Cuando IIS recibe una petición HTTP, evalúa la dirección URL para determinar si se trata de una petición de contenido estático (HTML) o dinámico (ASP o ISAPI). La Tabla IV.1 muestras los tipos de peticiones de IIS.

Request	Acción
Página HTML	IIS devuelve la página inmediatamente en formato HTML.
Extensión ISAPI	IIS carga la DLL ISAPI (si no está en ejecución) y se envía la petición a la extensión mediante la estructura de datos Extension_Control_Block.
Extensión de nombre de archivo asignada a una extensión ISAPI determinada	IIS carga el archivo DLL adecuado y presenta la petición mediante la estructura de datos Extension_Control_Block. Por ejemplo, la extensión .asp se asigna a Asp.dll, por lo que todas las peticiones de archivos con la extensión .asp se dirigirán a Asp.dll.
Aplicación CGI	IIS crea un nuevo proceso. Después, IIS proporciona la cadena de consulta y los parámetros incluidos en la petición mediante el identificador de entorno y entrada estándar (STDIN) para el proceso.

Tabla IV.1 Acciones de procesamiento de peticiones

4.2.3 Programación ASP .NET (Active Server Pages .NET) con C#

Antes de abordar el tema de la programación de ASP .NET, es conveniente hablar un poco sobre la programación ASP.

Programación ASP

Páginas Active Server (ASP) de Microsoft es un entorno de secuencias de comandos del servidor que se puede utilizar para crear páginas Web interactivas y para generar eficaces aplicaciones Web. Cuando un servidor recibe una petición de un archivo ASP, procesa las secuencias de comandos del servidor contenidas en el archivo para generar la página Web que se envía al explorador. Además de las secuencias de comandos del servidor, los archivos ASP pueden contener HTML (incluidas las secuencias de comandos del cliente relacionadas) así como llamadas

a componentes COM que realizan diversas tareas como conectarse a una base de datos o procesar la lógica empresarial.

El modelo de Páginas Active Server

Una secuencia de comandos del servidor comienza a ejecutarse cuando un explorador solicita un archivo .asp al servidor Web. El servidor Web llama a ASP, que procesa el archivo solicitado desde el principio hasta el final, ejecuta los comandos que encuentre y envía una página Web al explorador.

Puesto que las secuencias de comandos se ejecutan en el servidor y no en el cliente, el servidor Web hace todo el trabajo necesario para generar las páginas HTML que envía a los exploradores. Las secuencias de comandos del servidor no se pueden copiar, ya que sólo se devuelve al explorador el resultado de la secuencia de comandos. Los usuarios no pueden ver los comandos de la secuencia de comandos que crearon la página que están viendo. La Figura IV.13 muestra este proceso.

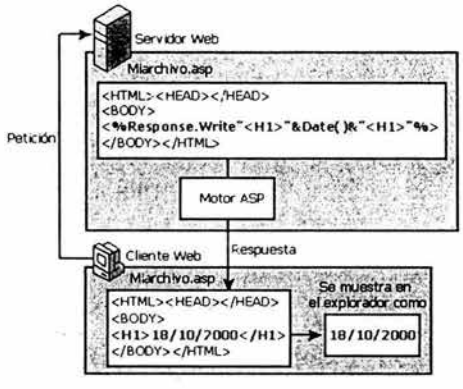


Figura IV.13 Funcionamiento de ASP

A diferencia de las aplicaciones CGI, que son difíciles de crear, ASP está diseñado para simplificar mucho el proceso de desarrollo de aplicaciones Web. Con una secuencia de comandos de unas pocas líneas se puede agregar a las páginas Web conexión con bases de datos o características de personalización avanzadas. En el pasado, era necesario saber PERL o C para agregar este tipo de funciones, pero con ASP se puede utilizar lenguajes normales de secuencias de comandos Web, como Microsoft JScript (la implementación de Microsoft de la especificación ECMA 262 del lenguaje), Microsoft Visual Basic Scripting Edition (VBScript) o cualquier lenguaje de secuencias de comandos compatible con COM, incluidos JavaScript, PERL y otros.

Se puede ampliar en gran manera la eficacia de ASP si se utiliza componentes COM (Modelo de objetos componentes) en las páginas ASP. Los componentes COM son partes de código compilado que se pueden llamar desde páginas ASP. Los componentes COM son objetos seguros, compactos y reutilizables que se compilan como archivos DLL. Pueden escribirse en Visual C++, Visual Basic u otros lenguajes que admitan COM.

Objetos integrados de ASP

Páginas Active Server (ASP) implementa clases que permiten a los componentes tener acceso a las propiedades y los métodos de los objetos integrados de ASP. En la Tabla IV.2 se resumen los objetos integrados de ASP.

Objeto	Función
ObjectContext	Devolver los objetos integrados y proporcionar los métodos que se utilizan en el procesamiento de transacciones.
Request	Llamar a los métodos y las propiedades del Objeto Request.
Response	Llamar a los métodos y las propiedades del Objeto Response.
ScriptingContext	Devolver los objetos integrados: Objeto Application, Objeto Request, Objeto Response, Objeto Server, or Objeto Session. Se trata de una opción obsoleta. En su lugar, utilice ObjectContext.
Server	Llamar a los métodos y las propiedades del Objeto Server.
Session	Llamar a los métodos y las propiedades del Objeto Session.

Figura IV.2 Objetos integrados de ASP

Programación ASP .NET

ASP.NET es la más reciente evolución de la tecnología Active Server Page (ASP) actual. Sin embargo, ASP.NET ha sido creada desde cero, de manera que se tienen nuevas características, incluyendo:

- Lenguajes compilados orientados a objetos
- Separación más limpia entre código y contenido
- Mayor desempeño y escalabilidad
- WebForms para rápido desarrollo de páginas Web, incluyendo un modelo de eventos y controles Web
- Varios niveles de soporte de memoria caché
- Controles de servidor que pueden generar diferentes salidas, tales como HTML 3.2, dependiendo del cliente. Clientes ricos, incluyendo Internet Explorer, pueden explotar más funcionalidad.
- Desarrollo y acceso a servicios Web simplificado
- Acceso a una plataforma rica a través del entorno .NET

Las páginas ASP.NET tienen una variedad de nuevas extensiones. Una página ASP.NET básica utiliza .aspx como la extensión del nombre de archivo. Un Servicio Web utiliza .asmx, y una nueva construcción denominada controles de usuario utiliza .aspc. Estas nuevas extensiones de nombre de archivo existen por una razón. Las aplicaciones ASP.NET se ejecutan de lado a lado con las aplicaciones ASP existentes. No comparten el estado de sesión o el estado de la aplicación, y coexisten tranquilamente en el mismo servidor. Estas nuevas extensiones del nombre de archivo son necesarias de manera que Internet Information Services (IIS) pueda llamar al filtro ISAPI adecuado para procesarlos.

ASP.NET aprovecha el CLR y el entorno de servicios para proporcionar un ambiente hosting confiable, robusto y escalable para las aplicaciones Web. Asimismo, ASP.NET se beneficia del modelo de ensambles de CLR para simplificar la implementación de aplicaciones. Además, proporciona servicios para simplificar el desarrollo de aplicaciones (como los servicios de administración de estado) y modelos de programación de más alto nivel (tales como Web Forms y servicios Web de ASP.NET). La Figura IV.14 muestra la arquitectura del sistema ASP.NET.

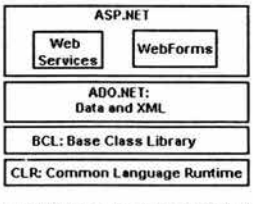


Figura IV.14 Arquitectura de ASP .NET

ASP.NET mejora los servicios de administración de estado presentados por ASP para proporcionar tres tipos de estado para las aplicaciones Web: aplicación, sesión y estado de Vista.

- Estado de aplicación. Al igual que ASP, el estado de aplicación es específico para una instancia de aplicación y no se persiste.
- Estado de sesión. Esta es específica para una sesión de usuario con la aplicación. A diferencia del estado de sesión ASP, el estado de sesión ASP.NET se puede almacenar en un proceso separado, y hasta puede configurarse para ser almacenado en una máquina separada. Esto hace que el estado de sesión sea utilizable cuando se implementa la aplicación en un Web Form.
- Estado de Vista. Esto se refiere al estado de una página y sus propiedades actuales, junto con el estado de cualquiera de los controles de la página. Se asemeja al estado de sesión pero no tiene vencimiento y se guarda automáticamente con cada ronda del cliente al servidor. El estado de Vista también es útil para almacenar preferencias del usuario y otra información de personalización.

ASP.NET encapsula gran cantidad de funcionalidad en común a través de los controles del servidor. También se proporcionan controles para administrar el estado de un formulario, y mostrar calendarios y tablas. Para casi todos los elementos HTML existe un control ASP.NET del servidor que lo produce y le permite interactuar con él programáticamente.

Los controles de servidor son nuevos, pero no son difíciles de aprender debido a que se correlacionan con sus equivalentes HTML. Los controles de servidor son identificados por el espacio de nombre 'asp:', y utilizan el método XML de una barra al final para cerrar el elemento. Al tiempo que no tiene la restricción de utilizar este estilo XML, y pueda utilizar el estilo HTML con una etiqueta de cerrado separada (</asp:TextBox>), éste es más rápido de escribir.

ASP.NET contiene cinco familias de controles de servidor:

- Controles intrínsecos que correlacionan sus equivalentes HTML,
- Controles de lista para proporcionar flujo de datos a través de una página,
- Controles ricos para proporcionar interfaz de usuario más rica de contenido y funcionalidad,
- Controles de validación para realizar una variedad de revalidaciones en formularios,
- Controles móviles para encapsular WML para dispositivos WAP.

Modelo orientado a objetos

ASP.NET ha sido reescrito con orientación a objetos en mente. ASP.NET proporciona un modelo de objetos para la página. En la parte superior de la jerarquía de objetos se encuentra el objeto Page. Cada control, aplicación y página en ASP.NET hereda de Page. Esto significa que cada página es una instancia del objeto Page. El evento Load de la página es uno de los eventos más importantes a los que puede responder.

Adicionalmente, cada objeto en una página puede tener su propio modelo de eventos y exponer y ejecutar eventos de servidor manejados por sus métodos de evento del lado del servidor. Rutinas como Button_Click or Listbox_Change pueden hacer relativamente simples el procesamiento estándar de formularios y varias otras tareas comunes. La legibilidad del código también se mejora ampliamente.

C#

C# es un lenguaje orientado a objetos y es el nuevo miembro de la familia de Visual Studio. C# es un descendiente de el lenguaje C y es muy parecido a C++ y a Java, pero diseñado con mucho mas simplicidad y uso en mente.

C# no requiere que uno se preocupe por la administración de asignación y desasignación de memoria. Debido a esto, C# es un lenguaje administrado por el CLR. Esto es una importante

ventaja porque la administración de la memoria es un de los mayores problemas de C++ y es responsable de las caídas de las aplicaciones.

C# lo he utilizado en las aplicaciones ASP .NET como el Code-Behind. El Code-Behind es una nueva característica de ASP .NET que permite separar el código HTML y los elementos UI de el código que proporciona la interacción con el usuario, validación, entre otras funcionalidades. Los módulos del Code –Behind ofrecen un número de ventajas, entre estas se encuentran:

- Separación limpia de HTML y código, lo cual hace una buena independencia y con ello un mejor entendimiento del código. La parte de HTML (Web Form) tendrá la extensión .aspx, mientras que la parte de módulos code-behind con C# tendrá el mismo nombre que la página Web Form, terminando con .cs
- Código reusable.
- Simplicidad en el mantenimiento.
- Desarrollo sin código fuente. Los proyectos en Visual Studio .NET usan módulos code-behind que pueden ser desarrollados como código compilado, permitiendo proteger de esta manera el código fuente.

4.3 MODELO DE OBJETOS COMPONENTES (COM)

Un componente COM es un bloque de código reutilizable que permite realizar una tarea o un conjunto de tareas. Es posible combinar unos componentes con otros (incluso a través de redes) para crear una aplicación Web. Los componentes COM ejecutan tareas comunes, de forma que no necesite crear su propio código para realizarlas. Por ejemplo, puede utilizar un componente de un ticker de bolsa para mostrar en una página Web las cotizaciones más recientes.

Para el desarrollo de aplicaciones Web, los componentes son el mejor método para encapsular la lógica empresarial en módulos seguros y reutilizables. Por ejemplo, podría utilizar un componente para comprobar números de tarjeta de crédito y llamarlo desde una secuencia de comandos que procese órdenes de venta. Como el proceso de comprobación está aislado del de pedido, puede actualizar el componente si cambia el proceso de comprobación de la tarjeta de crédito, sin necesidad de cambiar el proceso de pedido. Además, puesto que los componentes COM son reutilizables, se pueden utilizar de nuevo en otras secuencias de comandos y aplicaciones. Una vez instalado un componente en un servidor Web, puede llamarlo desde una secuencia de comandos ASP del servidor, desde una extensión ISAPI, desde otro componente del servidor o desde un programa escrito en cualquier lenguaje compatible con COM.

Es posible crear componentes en cualquier lenguaje de programación que sea compatible con el Modelo de objetos componentes (COM), como C, C++, Java, Visual Basic o muchos lenguajes de secuencias de comandos. Para ejecutarse en el servidor Web, los componentes COM no deben incluir elementos de la interfaz gráfica de usuario, como la función MsgBox de Visual Basic, ya que sólo se verían en el servidor y no en el explorador.

Creación de una instancia del objeto de un componente
Un componente es código ejecutable contenido en una biblioteca de vínculos dinámicos (.dll) o en un archivo ejecutable (.exe). Los componentes proporcionan uno o varios *objetos*, unidades de código autocontenidas que realizan funciones específicas dentro del componente. Cada objeto tiene métodos (procedimientos programados) y propiedades (atributos de comportamiento). Para utilizar un objeto proporcionado por un componente se crea una instancia del objeto y se le asigna un nombre de variable. Para crear la instancia del objeto se utiliza el método ASP Server.CreateObject o la etiqueta HTML <OBJECT>.

Interoperabilidad de .NET con COM
Para trabajar con un objeto COM, debe trabajar con un envoltorio .NET de manera que los clientes .NET crean que están llamando código .NET; el CLR maneja los detalles. Una utilidad analiza la

biblioteca de tipos COM y produce un DLL administrado cuya metadata describe los métodos que envuelve la interfaz del objeto COM. Ahora, una aplicación administrada puede crear instancias de un objeto y utilizarlas como si fuera un tipo nativo administrado. Para cada objeto COM, CLR genera un envoltorio invocable en tiempo de ejecución (RCW, Runtime Callable Wrapper). Este envoltorio actúa como un intermediario entre el código administrado y no administrado, manejando todas las tareas administrativas tales como ordenamiento de parámetros (marshaling), la administración de ciclos de vida, y demás. En la figura IV.15 se muestra un RCW generado en tiempo de ejecución para un objeto COM.

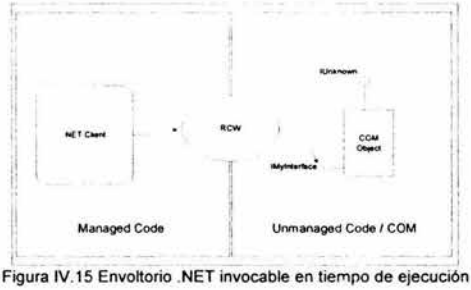


Figura IV.15 Envoltorio .NET invocable en tiempo de ejecución

4.4 Construcción de la base de datos (back-end)

Teniendo el diseño de la base de datos tanto conceptual como lógico, así como la documentación de la herramienta de desarrollo SQL Server 2000 se procede a la fase donde se construye la base de datos.

Creación de la base de datos

Para la creación de la base de datos del sistema se uso el SQL server enterprise manager. Una vez estando dentro de esta herramienta, damos clic con el botón derecho del raton y seleccionamos "nueva base de datos", como se muestra en la Figura IV.16.

Figura IV.16 Selección para la creación de la base de datos

A continuación aparecerá una pantalla que se muestra en la figura IV.17 en la cual se lleva a cabo las configuraciones necesarias para la base de datos.

Figura IV.17 Propiedades de la base de datos

Dos tipos de archivos son creados para almacenar la base de datos del sistema: Archivos de datos y archivos de transacciones. A estos se les configura su tamaño, si su crecimiento será automático, el porcentaje de crecimiento. Todo se lleva a cabo en la misma pantalla mostrada en la figura IV.17.

Creación de las tablas de la base de datos

Para la creación de las tablas de la base de datos del sistema se uso SQL server enterprise manager. Una vez estando dentro de esta herramienta, expandemos la base de datos y damos clic con el botón derecho del ratón en la opción Tablas y seleccionamos "nueva tabla", tal como se muestra en la Figura IV.18.

Figura IV.18 Creación de una nueva tabla

De esta forma accedemos a la vista de diseño de la tabla mostrada en la figura IV.17, en donde podemos empezar a meter registros a través de su nombre, tipo de dato, longitud, si será un campo nulo y otras características.

Una parte importante del diseño de la tabla es la elección de la llave primaria, la cual identifica de manera única a un registro. En la Figura IV.19 y Figura IV.20 se observa que el nombre de la columna clave_doc ha sido elegida como llave primaria, únicamente marcando la opción "poner como llave primaria".

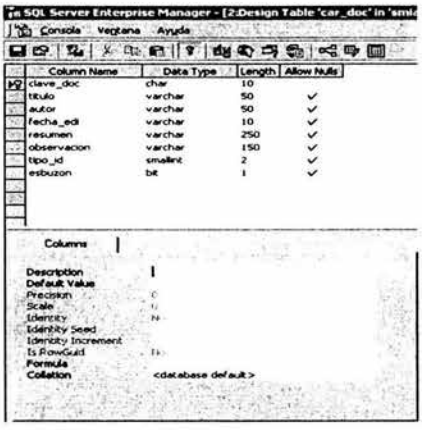


Figura IV.19 Vista de diseño de una tabla

Figura IV.20 Selección de la llave primaria

Una vez que tenemos las tablas construidas, se procede a llevar a cabo las relaciones entre ellas. Lo anterior constituirá el esquema físico de la base de datos.

4.4.1 Esquema físico de la base de datos del sistema

En esta fase, se suele utilizar el lenguaje propio de la Base de Datos (en la mayoría de los casos SQL). En este caso y tratándose de un producto de Microsoft, se ha utilizado Transact – SQL.

A continuación se muestran el diagrama de relaciones entra las tablas de la base de datos (Figura IV.21) y su script, generados con SQL server enterprise manager.

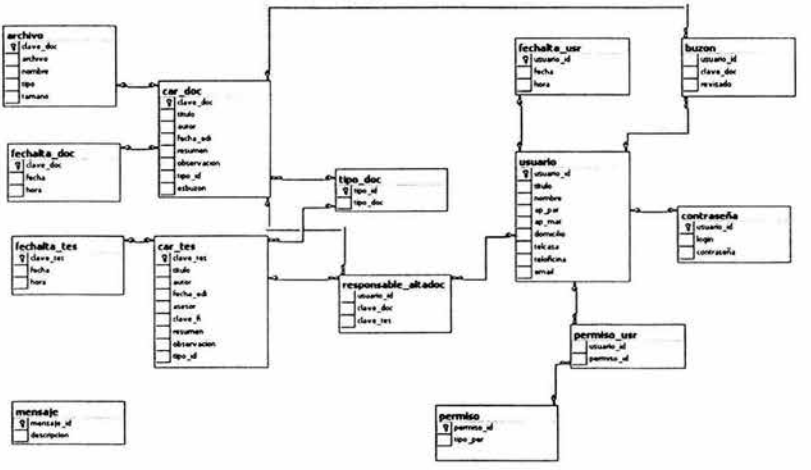


Figura IV.21 Esquema físico de la base de datos del sistema

```
CREATE TABLE [dbo].[archivo] (
	[clave_doc] [char] (10) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[archivo] [image] NULL ,
	[nombre] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
	[extension] [varchar] (5) COLLATE Traditional_Spanish_CI_AS NULL ,
	[tipo] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
	[tamano] [bigint] NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO

CREATE TABLE [dbo].[buzon] (
	[usuario_id] [int] NOT NULL ,
	[clave_doc] [char] (10) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[revisado] [bit] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[car_doc] (
	[clave_doc] [char] (10) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[titulo] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
	[autor] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
	[fecha_edi] [varchar] (10) COLLATE Traditional_Spanish_CI_AS NULL ,
	[resumen] [varchar] (250) COLLATE Traditional_Spanish_CI_AS NULL ,
	[observacion] [varchar] (150) COLLATE Traditional_Spanish_CI_AS NULL ,
	[tipo_id] [smallint] NULL ,
	[esbuzon] [bit] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[car_tes] (
	[clave_tes] [char] (10) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[titulo] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
```

```
	[autor] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
	[fecha_edi] [varchar] (10) COLLATE Traditional_Spanish_CI_AS NULL ,
	[asesor] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
	[clave_fi] [char] (10) COLLATE Traditional_Spanish_CI_AS NULL ,
	[resumen] [varchar] (250) COLLATE Traditional_Spanish_CI_AS NULL ,
	[observacion] [varchar] (150) COLLATE Traditional_Spanish_CI_AS NULL ,
	[tipo_id] [smallint] NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[contraseña] (
	[usuario_id] [int] NOT NULL ,
	[login] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL ,
	[contraseña] [varbinary] (50) NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[fechalta_doc] (
	[clave_doc] [char] (10) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[fecha] [datetime] NULL ,
	[hora] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[fechalta_tes] (
	[clave_tes] [char] (10) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[fecha] [datetime] NULL ,
	[hora] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[fechalta_usr] (
	[usuario_id] [int] NOT NULL ,
	[fecha] [datetime] NULL ,
	[hora] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[mensaje] (
	[mensaje_id] [int] NOT NULL ,
	[descripcion] [varchar] (75) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[permiso] (
	[permiso_id] [smallint] NOT NULL ,
	[tipo_per] [varchar] (35) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[permiso_usr] (
	[usuario_id] [int] NULL ,
	[permiso_id] [smallint] NULL
) ON [PRIMARY]
GO
```

```
CREATE TABLE [dbo].[responsable_altadoc] (
        [usuario_id] [int] NULL ,
        [clave_doc] [char] (10) COLLATE Traditional_Spanish_CI_AS NULL ,
        [clave_tes] [char] (10) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[tipo_doc] (
        [tipo_id] [smallint] NOT NULL ,
        [tipo_doc] [varchar] (25) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[usuario] (
        [usuario_id] [int] NOT NULL ,
        [titulo] [char] (10) COLLATE Traditional_Spanish_CI_AS NULL ,
        [nombre] [varchar] (30) COLLATE Traditional_Spanish_CI_AS NULL ,
        [ap_pat] [varchar] (30) COLLATE Traditional_Spanish_CI_AS NULL ,
        [ap_mat] [varchar] (30) COLLATE Traditional_Spanish_CI_AS NULL ,
        [domicilio] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
        [telcasa] [varchar] (20) COLLATE Traditional_Spanish_CI_AS NULL ,
        [teloficina] [varchar] (20) COLLATE Traditional_Spanish_CI_AS NULL ,
        [email] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL
) ON [PRIMARY] GO
```

4.4.2 Búsqueda de texto completo

Para implementar el sistema de búsqueda, se utilizó full-text search service (servicio de búsqueda de texto completo) incluido en SQL Server 2000, el cual no esta disponible en una instalación típica de SQL Server.

La búsqueda de texto completo se lleva a acabo actualmente mediante el uso de índices externos. Esto es porque tipo de datos como Text, nText e image no pueden ser indexados en SQL Server.

El concepto del sistema de búsqueda de texto completo es el mismo que se usa en los motores de búsqueda en la Web. Palabras "ruidosas" como 'el', 'o', 'un' son ignoradas. La Figura IV.22 ilustra la forma en que se lleva a cabo este servicio.

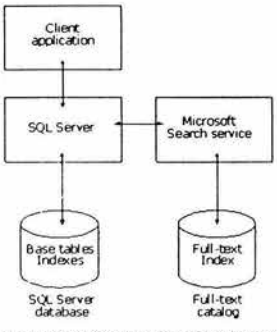


Figura IV.22 Servicio Full-Text Search

Para el desarrollo de este sistema, este servicio se implementó en tres tablas (car_tes, car_doc y archivo) mediante la creación de un catalogo de servicio de búsqueda de texto completo. A continuación se muestran las pantallas de la instalación del servicio de búsqueda de texto completo, así como para la creación del catálogo que contiene a las tablas mencionadas.

Creación del servicio "full-text search" en sql server

Para la instalación del servicio "full-text search" se necesitan aproximadamente 25 MB de espacio. Además, se necesitará una partición NTFS para almacenar el catálogo "full-text".

Para empezar, en la ventana de dialogo "seleccionar componentes", checar la opción asociada a "Full-Text Search" tal como se muestra en la Figura IV.23.

Una vez que se ha instalado este servicio, se puede iniciar usando uno de los siguientes métodos:

- Seleccionado 'Service manager' bajo el directorio Microsoft SQL Server.
- Usando la aplicación de servicios en el panel de control.
- Usando el comando net start mssearch.

El servicio "full-text search" consta de dos objetos: el índice full-text y el catálogo full-text. Para la base de datos del sistema, los índices full-text son los índices únicos (llave primaria) de las tablas car_doc , car_tes y archivo.

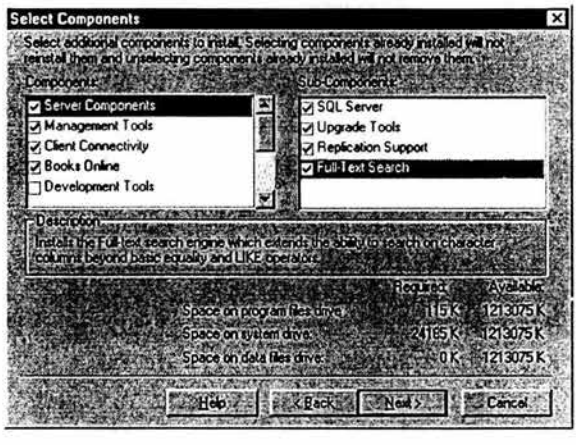


Figura IV.23 Selección de componentes

Creación del catalogo full-text

Usando el SQL Server Enterprise Manager:

1. Seleccionamos la base de datos donde se quiere crear el catálogo.
2. Desde el menú "Tools", seleccionar Full-Text Indexing

Lo anterior se muestra en la Figura IV.24.

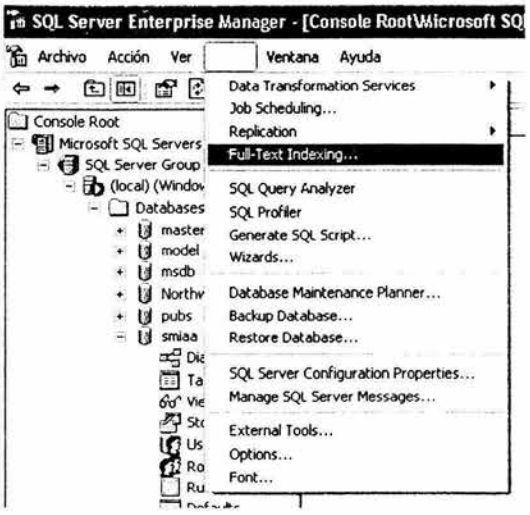


Figura IV.24 Selección del servicio de "Full-text Indexing"

De esta manera comienza un asistente. Damos click en siguiente en la pantalla de bienvenida. Seleccionamos la tabla, en este caso car_doc y damos siguiente, tal como se muestra en la Figura IV.25.

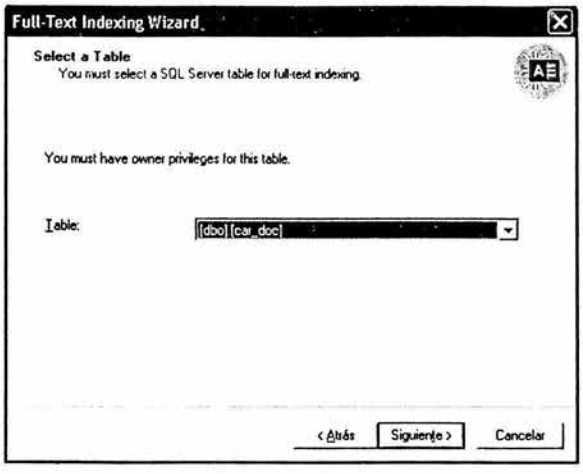


Figura IV.25 Selección de la tabla

Posteriormente seleccionamos el índice único para la tabla car_doc, como se indica en la Figura IV.26.

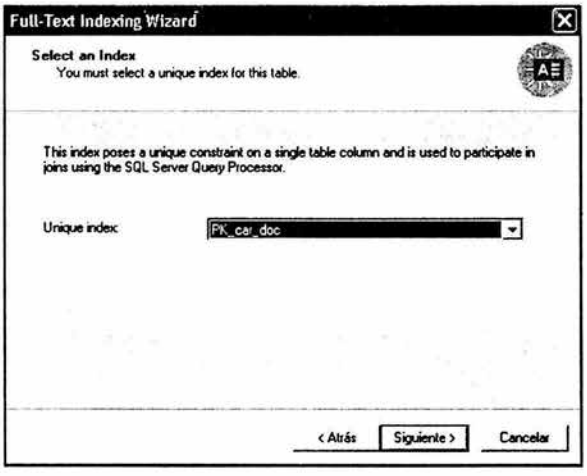


Figura IV.26 Selección del índice único

Ahora, seleccionamos las columnas en donde queremos habilitar la búsqueda de texto completo en la tabla car_doc. Para este caso, seleccionamos las columnas titulo, autor, resumen y observación, tal como se ilustra en la Figura IV.27.

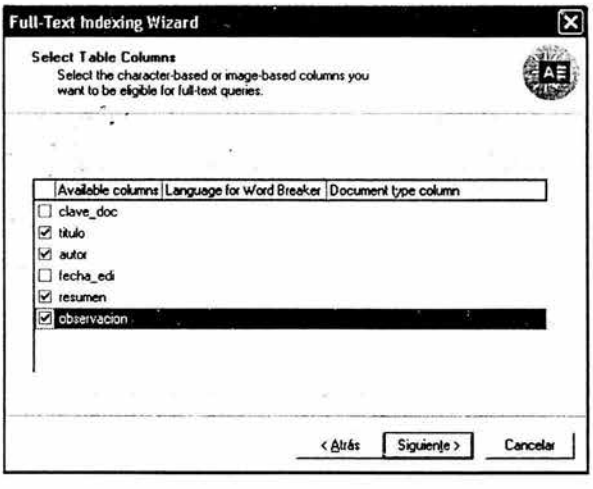


Figura IV.27 Selección de los campos de la tabla

A continuación damos un nombre para el catálogo o seleccionamos una ya creado. En nuestro caso seleccionamos el catálogo FTsmiia el cual se almacenará en: C:\Archivos de programa\Microsoft SQL Server\MSSQL\FTDATA. Lo anterior se observa en la Figura IV.28.

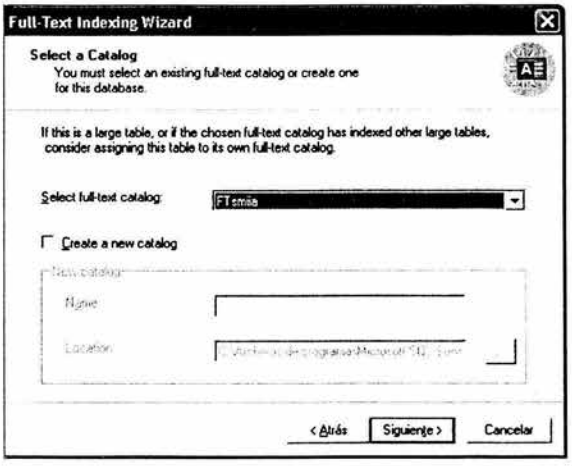


Figura IV.28 Selección del catálogo

Finalmente, podemos configurar un horario para refrescar el indexamiento full-text ó configurarlo para que los índices se actualicen en segundo plano cada vez que haya una modificación en la tabla. Una vez terminado lo anterior, finalizamos la creación del catálogo tal como se muestra en la Figura IV.29.

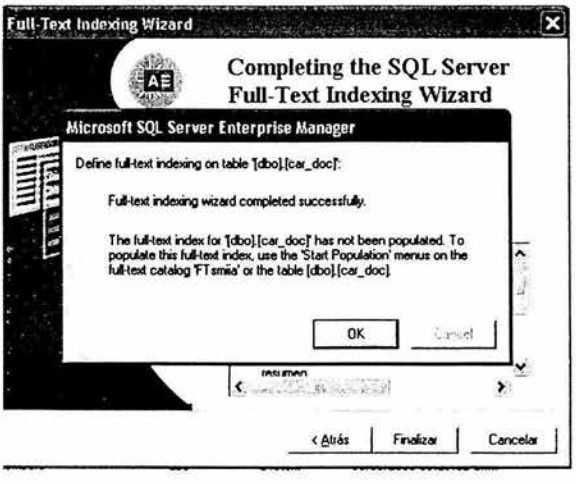


Figura IV.29 Finalización de la creación del catálogo

Una vez que se ha construido el catalogo para este servicio, se puede usar palabras claves en un query trabajando sobre las columnas de las tablas que contienen un índice de texto completo. Las palabras claves a usar son:

- CONTAINS. Determina si una palabra o frase está contenida en la columna

- FREETEXT. Busca un conjunto de palabras o frases dentro de una columna. SQL Server obtiene el significado de las palabras o frases que se especifiquen y busca aquellas palabras en la columna.

Al llevar a cabo la búsqueda mediante estas palabras claves, se obtienen mejores resultados en comparación a usar la palabra clave 'like', ya que se pueden realizar búsquedas del significado de un query más que solo especificar palabras o frases. Una desventaja que se tiene al usar el servicio de búsqueda de texto completo es que al llevar a cabo un respaldo de la base de datos, se tendrá que hacer por separado el respaldo del catálogo de servicio de búsqueda de texto completo.

4.4.3 Acceso a la base de datos con ADO .NET

ADO.NET es una tecnología de acceso a datos que se basa en los objetos ADO (Objetos de Datos ActiveX) anteriores. Es una manera nueva de acceder a los datos construida sobre ADO. ADO.NET puede coexistir con ADO. ADO.NET utiliza un modelo de acceso pensado para entornos desconectados. Esto quiere decir que la aplicación se conecta al origen de datos, hace lo que tiene que hacer, por ejemplo seleccionar registros, los carga en memoria y se desconecta del origen de datos.

ADO.NET es un conjunto de clases que usted utiliza para acceder y manipular orígenes de datos como por ejemplo, una base de datos en SQL Server o una planilla Excel. ADO.NET utiliza XML como el formato para transmitir datos desde y hacia su base de datos y su aplicación Web.

Hay 3 espacios de nombres disponibles para importar en un formulario Web si se esta usando ADO.NET: System.Data, System.Data.SqlClient y System.Data.OleDb.

El modelo de objetos ADO.NET provee una estructura de acceso a distintos orígenes de datos. Tiene 2 componentes principales: El Dataset y el proveedor de Datos .NET

- El Dataset: Esta formado por uno o más objetos de tipo DataTables. fue pensado para acceder a datos independientemente del origen. Por ejemplo, un DataSet puede obtener datos de SQL Server, Oracle o de un archivo XML. Puede utilizar un objeto llamada DataView para ver los datos de distintas maneras.
- El proveedor de Datos .NET: Provee del enlace entre el Origen de Datos y el DataSet.

La tabla IV.3 muestra un ejemplo de Objetos provistos por distintos proveedores de datos .NET

Objetos de Prov. de Datos .NET	Propósito	Objeto SQL Server 7.0 o 2000	Objeto para un origen OLEDB
Connection	Provee conectividad a un Origen de Datos	SqlConnection	OleDBConnection
Command	Provee acceso a comandos de Base de Datos como Select, Delete, Insert y Update	SqlCommand	OleDBCommand
DataReader	Provee acceso a datos de solo lectura	SqlDataReader	OleDBDataReader
DataAdapter	Utiliza el objeto Connection para enlazar un objeto DataSet con un Proveedor de Datos. También permite actualizar los Datos en el origen a partir de las modificaciones hechas en el DataSet.	SqlDataAdapter	

Tabla IV.3 Objetos de proveedores de datos .NET

Para el desarrollo del sistema se importaron 2 espacios de nombre (conjunto de clases):

```
using System.Data;
using System.Data.SqlClient;
```

Se uso SqlClient y no OleDb ya que el primero fue construido para conexiones con la base de datos SQL Server, tomando ventaja de esta. La conexión fue hecha de la siguiente manera:

```
string sql="SELECT *FROM usuario";
SqlConnection ConnBD = new SqlConnection("server='(local)'; trusted_connection=true;
database='smiaa'");
SqlCommand cmdTipoID = new SqlCommand(sql,ConnBD);
ConnBD.Open();
```

Se tiene al inicio la declaración de una sentencia sql. Posteriormente declaro una instancia para tener conectividad a un origen de datos, en este caso, a la base de datos del sistema "smiaa" con un tipo de conexión verdadera. En seguida obtengo acceso a los datos con la clase SqlCommand, el cual toma como argumento tanto la sentencia sql como la instancia del origen de datos.

A continuación se establece la conexión y se puede empezar a manipular los datos. Para esto, como se trata de una lectura de datos, uso una instancia de SqlDataReader de la siguiente manera:

```
SqlDataReader dbRead = cmdTipoID.ExecuteReader();
```

En seguida se lee un renglón de la tabla: `dbRead.Read();`
y extraigo un campo en particular, el cual asigno a una etiqueta del formulario:

```
Label1.Text=dbRead["usuario_id "].ToString().ToUpper();
```

Finalmente, cierro la conexión de la siguiente manera: `ConnBD.Close();`

4.4.4 Manejo de Procedimientos Almacenados

Un procedimiento almacenado es una colección de sentencias Transact-SQL que es almacenado en la base de datos. Los procedimientos almacenados son un método para encapsular tareas repetitivas. Soportan variables definidas por el usuario, ejecuciones condicionales y muchas otras características de programación poderosas.

Existen cinco tipos de procedimientos almacenados:

- Procedimientos almacenados del sistema
- Procedimientos almacenados locales
- Procedimientos almacenados temporales
- Procedimientos almacenados remotos
- Procedimientos almacenados extendidos

Las ventajas del uso de procedimientos almacenados son:

- Permite compartir la lógica de la aplicacion con otras aplicaciones.
- Proveen mecanismos de seguridad
- Se oculta a los usuarios de los detalles de las tablas de la base de datos.
- Mejoran el rendimiento
- Reducen el tráfico en la red.

En el desarrollo del sistema se emplearon solo cuatro procedimientos almacenados locales, para llevar a cabo tareas muy especificas y de esta manera obtener un mayor rendimiento de la aplicación.

Una de las tareas en la cual se aplico el uso de procedimientos almacenados en el sistema fue en el alta de un documento, ya que el usuario puede almacenar un archivo en la base de datos, lo cual se requiere que se haga en un tiempo mínimo.

El código que conforma al procedimiento almacenado para el *alta de un documento* es el siguiente:

```
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_NULLS ON
GO

CREATE PROCEDURE SP_AltaDoc
        @clave_doc varchar(10),
        @titulo varchar(250),
        @autor varchar(50),
        @fecha_edi smalldatetime,
        @resumen varchar(1000),
        @observacion varchar(250),
        @tipo_id smallint,
        @hora varchar(15),
        @usuario_id smallint,
        @archivo image,
        @nombre varchar(255),
        @extension varchar(5),
        @tipo varchar(50),
        @tamano int
 AS
        INSERT car_doc(clave_dcc,titulo,autor,fecha_edi,resumen,observacion,tipo_id)
        VALUES(@clave_doc,@titulo,@autor,@fecha_edi,@resumen,@observacion,@tipo_id)
        INSERT fechalta_doc(clave_doc,hora)  VALUES(@clave_doc,@hora)
        INSERT responsable_altadoc(usuario_id,clave_doc)  VALUES(@usuario_id,@clave_doc)
        INSERT archivo(clave_doc,archivo,nombre,extension,tipo,tamano)
        VALUES(@clave_doc,@archivo, @nombre, @extensión, @tipo,@tamano)
GO
SET QUOTED_IDENTIFIER OFF
GO
SET ANSI_NULLS ON
GO
```

La forma con la cual se interactúa el anterior procedimiento almacenado desde el code-behind con C# es el siguiente:

```
SqlConnection ConnBD;
SqlCommand cmdDoc;
ConnBD = new SqlConnection("server='(local)'; trusted_connection=true; database='smiaa'");

//Creo las configuraciones necesarias del objeto SqlCommand
cmdDoc = new SqlCommand("SP_AltaDoc",ConnBD);
cmdDoc.CommandType = CommandType.StoredProcedure;
cmdDoc.Parameters.Add("@clave_doc",SqlDbType.VarChar,10);
cmdDoc.Parameters.Add("@titulo",SqlDbType.VarChar,250);
```

```
cmdDoc.Parameters.Add("@autor",SqlDbType.VarChar,50);
cmdDoc.Parameters.Add("@fecha_edi",SqlDbType.SmallDateTime,4);
cmdDoc.Parameters.Add("@resumen",SqlDbType.VarChar,1000);
cmdDoc.Parameters.Add("@observacion",SqlDbType.VarChar,250);
cmdDoc.Parameters.Add("@tipo_id",SqlDbType.SmallInt,2);
cmdDoc.Parameters.Add("@hora",SqlDbType.VarChar,15);
cmdDoc.Parameters.Add("@usuario_id ",SqlDbType.SmallInt,2);
cmdDoc.Parameters.Add("@archivo",SqlDbType.Image);
cmdDoc.Parameters.Add("@nombre",SqlDbType.VarChar,255);
cmdDoc.Parameters.Add("@extension",SqlDbType.VarChar,5);
cmdDoc.Parameters.Add("@tipo",SqlDbType.VarChar,50);
cmdDoc.Parameters.Add("@tamano",SqlDbType.Int,4);
cmdDoc.Parameters[0].Value = claveDoc;
cmdDoc.Parameters[1].Value = titulo.Text;
cmdDoc.Parameters[2].Value = autor.Text;
cmdDoc.Parameters[3].Value = fecedi.Text;
cmdDoc.Parameters[4].Value = resumen.Text;
cmdDoc.Parameters[5].Value = observacion.Text;
cmdDoc.Parameters[6].Value = tipoIDDoc;
cmdDoc.Parameters[7].Value = horaDoc;
cmdDoc.Parameters[8].Value = usuarioID;
cmdDoc.Parameters[9].Value = Docbuffer;
cmdDoc.Parameters[10].Value = nombreDoc;
cmdDoc.Parameters[11].Value = tipoextension;
cmdDoc.Parameters[12].Value = tipoDoc;
cmdDoc.Parameters[13].Value = tamañoDoc;
ConnBD.Open();
cmdDoc.ExecuteNonQuery();
ConnBD.Close()
```

En el código anterior paso como argumentos en el SqlCommand el nombre del procedimiento almacenado y la instancia de la conexión a la base de datos. En seguida defino cada uno de los tipos de datos de las variables que recibe el procedimiento almacenado. Esto me garantiza la consistencia de los tipos de datos.

Finalmente paso cado uno de los datos del usuario al procedimiento almacenado, abriendo y ejecutando el query. Se termina cerrando la conexión de la base de datos.

4.5 Construcción de la interfaz de usuario (front-end)

En cualquier sitio Web intervienen diferentes tecnologías. Es fundamental el uso de páginas HTML, lenguajes de script para realizar algunas operaciones en el lado del cliente, acceso a bases de datos, servidor Web, etc. A continuación comentamos las decisiones más relevantes a la hora de elegir entre varias alternativas respecto a estas tecnologías:

- *Servidor Web*: es el punto de acceso principal para los browsers de los clientes. El servidor Web que hemos utilizado es IIS 6.0 (Internet Information Server). En el mercado hay varios servidores Web, pero he elegido este, principalmente, porque viene integrado en Windows 2003 Server que se tiene con licencia en el laboratorio Microsoft.
- *Páginas cliente*: para las páginas que no requieren procesamiento en el servidor se utiliza, como es lógico, *HTML*
- *Script cliente*: uso JavaScript embebido dentro de algunas páginas HTML para conseguir una interfaz mejorada, añadiendo funcionalidad como, por ejemplo, la validación de algunos formularios Existen otras alternativas, pero ésta es la más apropiada.

- *Página servidor*: son todas las páginas web que de alguna forma realizan procesamiento en el servidor. En este punto tenía tres opciones válidas: *Java Server Pages (JSP)*, PHP y ASP .NET (active server page .NET). Me he decidido a utilizar ASP .NET con C# debido a que presenta como ventajas sobre los JSP y PHP que permite un manejo mas fácil y limpio del código, así como un menor aprendizaje (teniendo como previo conocimiento de ASP y lenguaje C), facilitando de esta manera el mantenimiento.

Por otra parte, en lo referente a la arquitectura del sistema, el protocolo utilizado entre los browser de los clientes y el servidor Web es el HTTP. Este elemento de la arquitectura representa un tipo de comunicación no orientado a la conexión entre clientes y servidor. Una alternativa de la conexión HTTP es usar "HTTP seguro", que es HTTP utilizando como protocolo de transporte SSL (Secure Sockets Layer).

Para realizar la interfaz de cada una de las aplicaciones del sistema se ha utilizado las herramientas Dreamweaver XP que nos permite realizar de manera sencilla el diseño de interfaces con todos los elementos convencionales (cuadros de texto, botones, combo box, frames, etc.) y Visual Studio .NET para el desarrollo de la lógica del sistema, programando con C#. Estas herramientas están, además, perfectamente preparadas para interactuar con bases de datos SQL Server.

4.5.1 Creación de la aplicación Web en Visual Studio .NET

En la creación de la aplicación Web con Visual Studio .NET, se crea:

- Un directorio virtual en IIS, configurado como una aplicación root, el cual mantendrá los archivos que conformarán la aplicación y controlará el acceso a los archivos.
- Unos o mas archivos con extensión .aspx
- Un archivo global llamado Global.asax el cual tiene que ver con las configuraciones de variables de sesión y de aplicación.
- Un archivo de configuración llamado Web.config.

Para empezar la creación de la aplicación del sistema SMIAA, se sigue los siguientes pasos:

1 Ejecutar Visual Studio .NET.
2 Abrir la caja de dialogo "New Project" , dando Clic en el boton "New Proyect" mostrado en la Figura IV.30

Figura IV.30 Creación de un nuevo proyecto

3 En la caja de dialogo "New Proyect", tal como se muestra en la Figura IV.31, bajo "Project Types" seleccionamos "Visual C# Proyects" y posteriormente seleccionamos el template apropiado "ASP.NET Web Application". Escribimos el nombre del proyecto en la parte "Location", que para el sistema se nombra "smiaa". Terminado lo anterior, damos clic en el botón OK.

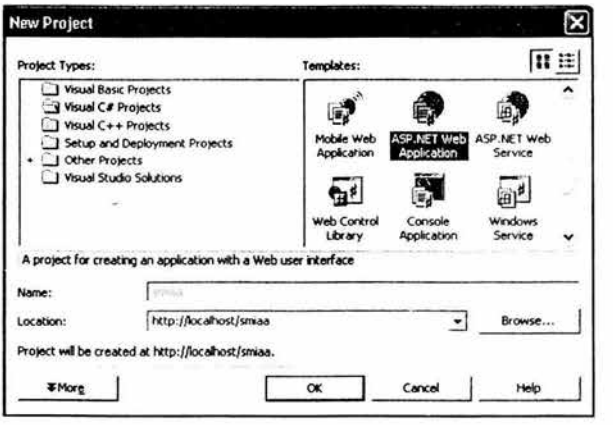


Figura IV.31 Opciones de configuración del nuevo proyecto

4.5.2 Creación de páginas ASP .NET en Visual Studio .NET

Una vez creado la aplicación Web, se empiezan a añadir nuevos "Web Form" a la aplicación de la siguiente forma:

1 En la ventana del explorador de soluciones, damos clic con derecho sobre el nombre de la aplicación, después seleccionamos "Add" y posteriormente "Add Web Form...", tal como se muestra en la Figura IV.32

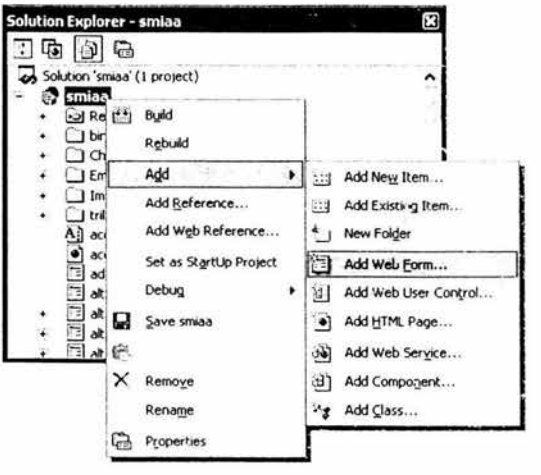


Figura IV.32 Añadir una nueva página Web

2 En la caja de dialogo "Add New Item", como la que se obsewrva en la Figura !V.33, verificamos que el template seleccionado haya sido "Web Form" y nombramos la nueva página default.aspx. Damos clic en Open.

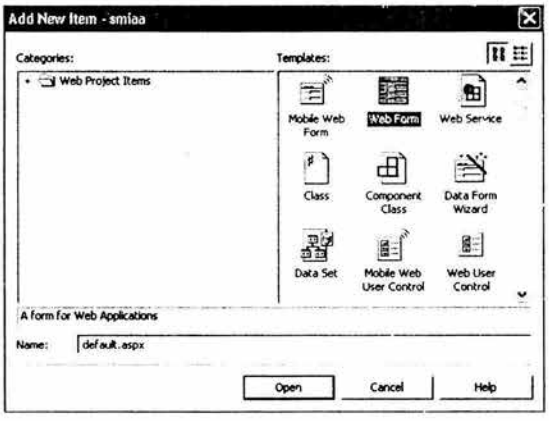


Figura IV.33 Selección de un Web Form

3 Una vez que se ha definido la página Web "default.aspx", se procede a añadirle funcionalidad a través de controles Web forms. En la Figura IV.34 se observa en la parte central el area de trabajo para esta página Web, a la izquierda se tiene los controles "Web Forms", en la parte superior derecha se tiene las propiedades del control seleccionado correspondiente al botón para el usuario y en la parte inferior se tiene el explorador de soluciones.

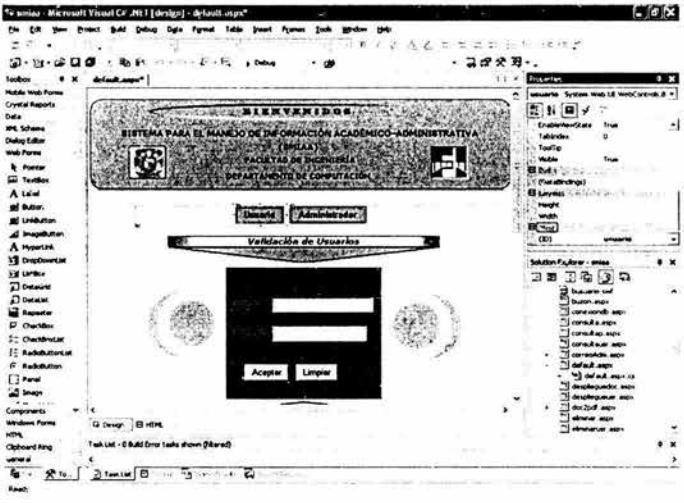


Figura IV.34 Ambiente de programación de Visual Studio .NET

4 Para hacer que la página haga algo útil, es necesario agregarle código. En nuestro caso, damos doble clic en el botón seleccionado que es el botón de usuario.

La página que aparece ahora pertenece a la página del code-behind en C# tal como se muestra en la Figura IV.35. Dentro de esta aparece la función para el botón de usuario. El propósito de la página de inicio "default.aspx" es que para cada evento del botón ya se de usuario o de administrador, las opciones de validación cambien, con lo cual se tendrá visibilidad para el panel de usuario y mientras se oculta el panel del administrador. De esta forma, el código para el botón de usuario es el siguiente:

```
PanelUsr.Visible=true;
PanelAdm.Visible=false
```

Cabe hacer mención la funcionalidad de la función "Page_Load". Esta función contiene código que se ejecutara inmediatamente que la página se cargue.

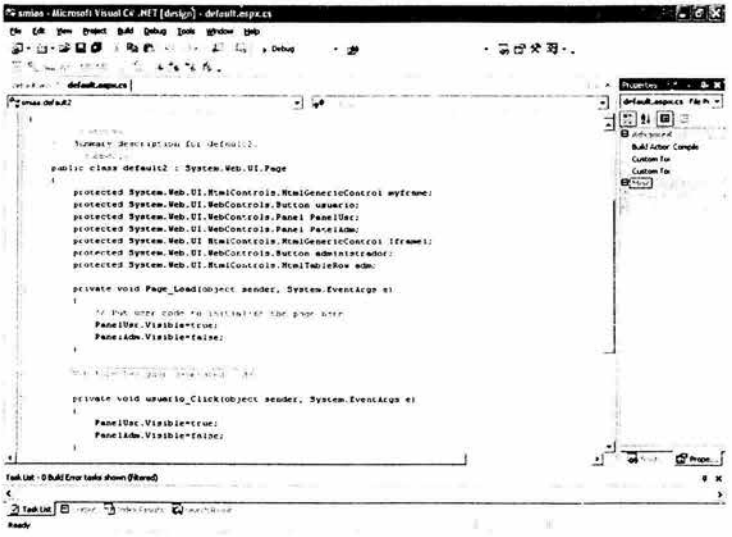


Figura IV.35 Code-Behind de un formulario Web

5 Una vez que se ha construido la página, se procede a probarla. La página "default.aspx" será la primera página que el usuario verá cuando ingrese al sistema. La página completa se muestra en la Figura IV.36

En dicha pantalla el usuario tendrá que validarse a través de un login y password para poder ingresar al sistema.

Figura IV.36 Pantalla inicial del sistema

Una vez que el usuario ha sido validado, ingresará al área de trabajo. En esta área, el usuario podrá hacer uso de las opciones alta, baja, consulta y eliminación de documentos así como de usuarios entre otras opciones de acuerdo al tipo de permiso que tenga

Se ha omitido poner las líneas de código que integran a cada una de las páginas Web. En lugar de esto se ha detallado en el Anexo II "Manual Técnico" cada página Web en tablas, explicando su funcionamiento y sus características.

4.6 Seguridad del sistema

Para la seguridad del sistema se persiguen los siguientes servicios de seguridad: autentificación, confidencialidad, integridad, no repudio y disponibilidad. Para esto, el usuario contará con un login y password, el cual será encriptado. Existirá seguridad tanto en el servidor como en Internet.

Las políticas de seguridad que se deben hacer cumplir son las siguientes:

- Únicamente podrán entrar al sistema aquellos usuarios que esten dados de alta.
- La autentificación del usuario se llevará a cabo con un login y password.
- Solo el administrador y secretarias del Departamento de Ingeniería en Computación pueden dar de alta a un usuario.
- Solo el administrador podrá hacer las modificaciones que se han necesarias para el buen funcionamiento del sistema.
- El servidor contará con un sistema ininterrumpible de potencia por 20 minutos.
- El servidor será accesado solo por el administrador, para evitar que alguien malintencionado lleve a cabo alguna desconfiguración.

4.6.1 Seguridad en el servidor

En el centro de la seguridad de Windows 2003 Server está la cuenta de usuario y su extensión lógica, el grupo del usuario. Cuando se instala IIS, crea dos cuentas de usuario, les asigna derechos de usuario específicos y los acomoda en grupos de usuarios específicos. Estas dos

cuentas son IUSR_computername y IWAN_computername. La cuenta IUSR_computername es usada por IIS para asegurar acceso anónimo a los recursos del Web. IWAN_computername es la cuenta usada por el Servidor de transacciones de Microsoft (Microsoft Transaction Serv<er, MTS) y varias entidades de IIS para proporcionar funciones programáticas y transaccionales. Para la configuración del servidor del presente sistema es importante la primera cuenta, IUSR_computername.

La cuenta IUSR_computername requiere tener el acceso de inicio de sesión local (Logon Locally) configurado correctamente. Esto se debe a que actúa dentro de IIS, que es un servicio que actúa localmente, tal como si fuera un usuario que físicamente se conecta al servidor. SI usa cualquier otra cuenta que no sea IUSR_computername para acceso anónimo, seleccione con cuidado los derechos que asigna. Cada visitante anónimo a su sitio obtendrá los derechos de la cuenta IUSR_computername. IIS también depende de las cuentas de usuarios al proporcionar autentificación Básica y Windows Challenge/Resource. Con el fin de finalizar la operación con éxito, ambos métodos requieren que cuentas de usuarios válidas estén asignadas. Esto es necesario porque aunque IIS crea y configura la cuenta IUSR_computername, que se usa para autentificación Anónima, no crea ninguna cuenta para autentificación Básica.

El sistema de archivos de Windows NT
Aunque IIS puede desempeñarse correctamente en la unidad de disco duro de una tabla de asignación de archivos (File Allocation Table, FAT), se debe considerar cambiar el formato al sistema de archivos Windows NT File System (NTFS). Aquí está el por qué:

- A diferencia de FAT, NTFS no es visible para DOS. Esto vuelve más seguros los recursos del sistema en contra de atentados de intrusión al usar los comandos de DOS.
- NTFS permite configurar la Lista de control de acceso (Access Control List, ACL) para garantizar o denegar varias formas de acceso a cuentas de usuario o de grupos.
- El sistema de archivos NTFS es más eficiente con suficiente espacio de disco duro.

Un ACL es una lista de cuentas de usuarios, grupos de usuarios y sus privilegios asociados con un recurso en particular, como un directorio o un archivo. Por ejemplo, el archivo MyISAPI.dll tendría una lista de cuentas de usuario y grupos de usuario que puedan accesarlo y qué nivel de acceso para el archivo van a obtener, como Lectura, Escritura o Ejecución. Los ACLs son otra función básica del modelo de seguridad a partir de Windows NT y permite un control de acceso flexible y preciso a los recursos en el disco duro. Cada directorio y cada archivo tienen su propio ACL que define quién puede hacer qué y dónde. Aún cada ACL tiene un ACL que especifica quién puede ver y cambiar el ACL por sí mismo. Los NTFS y los ACLs proporcionan una base para asegurar los recursos del servidor.

La seguridad con ASP .NET
ASP .NET permite una mejor integración en la seguridad con el servidor IIS. La Figura IV.37 muestra esta seguridad:

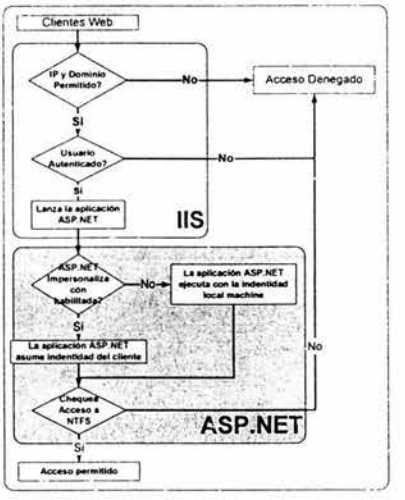


Figura IV.37 Esquema de seguridad de ASP .NET

De esta forma, podemos restringir el acceso de usuarios anónimos en el archivo Web.config:

```
<authorization>
        <deny users="?"/>
</ authorization >
```

4.6.1 Seguridad en Internet

La parte fundamental en la seguridad por Internet de este sistema es la transmisión de datos importantes como la validación para entrar al sistema y el envió de correo con lo cual se hace necesario una encriptación de los datos.

Para la encriptación de los datos se uso un componente COM, el cual usa RSA como tipo de encriptación. El algoritmo RSA de llave pública es el más usado. Fue inventado en 1978 por Ron Rivest, Adi Shamir y Leonard Adleman. Puede ser usado para encriptar y para firmas digitales. La seguridad de RSA esta considerada como equivalente a factorizar, aunque no ha sido probado. Esta encriptación es lenta. Usa números primos muy grandes para generar la llave pública y la llave privada. Aunque podría ser posible factorizar la llave pública para obtener la privada los números son tan grandes que resultaría impractico.

EL cálculo de RSA sucede con el modulo de enteros n= p*q para dos primos (p, q) largos secretos. Para encriptar el mensaje, es elevado con un exponente público pequeño. Para desencriptarlo, quien recibe el texto cifrado $c = m^e \pmod n$ calcula $d = e^{-1} \pmod{(p-1)*(q-1)}$ y se obtiene $c^d = m^{e*d} = m \pmod n$. La llave privada consiste de n, p, q, e, d; la llave publica solo consiste de n y e. Un intruso puede calcular el inverso de d de e pero no es más fácil que factorizar n.

La medida de los módulos de la llave debe ser mayor que 1024 bits, y una de 2048 debe proporcionar seguridad por décadas

El componente requerido para crear las llaves validas es RSADLL. Este componente tiene dos métodos:

RSA.DeCrypt(Inp As String,d As Long,n As Long) Regresa una cadena con el valor desencriptado

RSA.Encrypt(Inp As String,e As Long,n As Long) Regresa una cadena con el valor encriptado

El componente RSADLL no tiene propiedades. De este modo se muestra la forma en como se encriptaría el password:

```
Dim RSA,RSAKeyGen,PasEncriptado,e,d,n
Set RSA = Server.CreateObject("RSADLL.Crypt")
Set RSAKeyGen = Server.CreateObject("RSADLL.RSAKeyGen")
RSAKeyGen.GenerateKey
PasEncriptado = RSA.Encrypt(password,RSAKeyGen.e,RSAKeyGen.n)
```

Como se observa se crean dos objetos: el primero de ellos nos servirá para encriptar el password. El segundo objeto generará tanto la llave pública como la llave privada.

Se puede ver hasta aquí que en el proceso de envío y recuperación de los documentos no existe encriptación alguna que garantice la seguridad de los documentos. Para esto es necesario el uso de un protocolo de seguridad construido en browsers llamados Secure Sockets Layer o SSL. Este protocolo requiere que el servidor tenga de un certificado y no el usuario.

La obtención de un certificado se lleva acabo mediante una autoridad de certificados (Certificate Authoritym, CA) el cual es una entidad o servicio que emite certificados.
Un certificado es un archivo que contiene información sobre una entidad firmado por una CA

Algunas autoridades de certificación son:

- Thawte Consulting
- Verisign

Dado que un certificado requiere un costo económico, por el momento el presente sistema no implementará el protocolo SSL.

4.7 Pruebas y depuración del sistema

El presente sistema debe garantizar la seguridad, transparencia, confiabilidad e integridad de la información en todas sus fases de desarrollo. Para ello, es necesario llevar a cabo una fase de prueba que garantice el buen funcionamiento del sistema.

El llevar a cabo pruebas en el sistema tienen como objetivo:

- Detectar posibles fallas para su inmediata corrección.
- Realizar pruebas de funcionalidad de todos sus componentes.
- Estimar tiempos de captura, transmisión, procesamiento y difusión de los datos.
- Identificar posibles mejoras al modelo.
- Evaluar la prueba.

Los puntos que principalmente se evaluaron fueron:

1. Captura de documentos.
2. Consulta de documentos.
3. Captura de usuarios.
4. Modificación de usuarios.

5. Chat del departamento.
6. Envió de mensajes por medio del correo electrónico.
7. Consulta del buzón de académicos.
8. Aspecto de la interfaz del sistema.
9. Cumplimiento del sistema con los objetivos establecidos.
10. Mejoras propuestas para el sistema

A través de estas pruebas se llevaron a cabo algunas modificaciones necesarias al sistema para obtener un mejor desempeño.

CAPÍTULO

V

MANTENIMIENTO DEL SISTEMA

CONTENIDO

Después de que el sistema ha sido verificado, probado e implantado, se les debe seguir dando mantenimiento para asegurar que continúen operando en el nivel mostrado durante la etapa de prueba. Las rutinas de mantenimiento variarán de acuerdo con el problema detectado en el sistema o por la necesidad de una mejora del mismo.

5.1 Tipos de mantenimiento

Si un problema es detectado por el usuario, inmediatamente puede notificarlo al administrador del sistema a través del envió de un correo electrónico, opción que siempre estará disponible en el sistema en la parte inferior de su pantalla. Esta petición es registrada y se procede a diagnosticar de qué tipo de mantenimiento se trata. Atendiendo a los fines, podemos establecer los siguientes tipos de mantenimiento:

- Correctivo: son aquellos cambios precisos para corregir errores del producto software.
- Evolutivo: son las incorporaciones, modificaciones y eliminaciones necesarias en un producto de software para cubrir la expansión o cambio en las necesidades del usuario.
- Adaptativo: son las modificaciones que afectan a los entornos en los que el sistema opera, por ejemplo, cambio en las configuraciones del hardware, software de base, gestores de bases de datos, comunicaciones, etc.
- Perfectivo: son las acciones llevadas a cabo para mejorar la calidad interna de los sistemas en cualquiera de sus aspectos: reestructuración de código, definición más clara del sistema y optimización del rendimiento y eficiencia.

Una vez identificado el tipo de mantenimiento y su origen, se determina a fijar un tiempo razonable de tiempo para su modificación y prueba y se le notifica al usuario.

Si se trate de un mantenimiento correctivo o evolutivo, los cuales son más comunes, se verifica y reproduce el problema, o se estudia la viabilidad del cambio propuesto por el usuario. En ambos casos se estudia el alcance de la modificación. Hay que analizar las alternativas de solución identificando, según el tipo de mantenimiento de que se trate, cuál es la más adecuada. El plazo y urgencia de la solución a la petición se establece de acuerdo con el estudio anterior.

Las tareas de los procesos de desarrollo que va a ser necesario realizar son determinadas en función de los componentes del sistema actual afectados por la modificación. Estas tareas pertenecen a actividades de los procesos Análisis, Diseño, e Implementación.

Por último, y antes de la aceptación del usuario, es preciso establecer un plan de pruebas de regresión que asegure la integridad del sistema de información afectado.

5.2 Revisiones periódicas

El monitoreo permanente del sistema asegura que las necesidades de mantenimiento sean identificadas y satisfechas cuando resulte necesario. Cuando el sistema es de uso prolongado, se puede establecer un mecanismo para recibir retroalimentación de los usuarios como una forma efectiva para determinar las necesidades de mantenimiento y modificación.

A los sistemas se les debe dar mantenimiento para asegurar que continúen operando en el nivel mostrado durante la etapa de prueba. Si los sistemas se deterioran, existe el riesgo de que no se desempeñen conforme a los estándares requeridos.

5.2.1 Respaldos de la base de datos del sistema

El presenta sistema almacena información importante del Departamento de Ingeniería en Computación. Por este motivo, se llevarán a cabo respaldos cada quince días por parte del administrador, asegurando de esta forma que la Base de Datos no llegue a su máxima capacidad.

De lo anterior, se menciona que la Base de Datos del sistema ha sido configurada para que crezca en capacidad de almacenamiento automáticamente. La única limitante es la capacidad del Disco SCSI del servidor.

Otra forma de mantener prevenido al administrador sobre el estado de la base de datos del sistema es a través del envío de alertas usando el correo electrónico. Esta opción ha sido configurada en la base de datos.

5.2.2 Cambios en la lógica de programación del sistema

Si un cambio se presentará en la lógica de programación del sistema, el uso de la herramienta de Visual Studio .NET será de gran ayuda para el programador, ya que separa la parte de diseño del sistema de la lógica de programación, dando como resultado que la tarea de mantenimiento no se vuelva un gran problema.

CONCLUSIONES

Las herramientas de computación que hoy en día existen hacen que sea más fácil el desarrollo de sistemas que trabajan en un ambiente cliente – servidor. ASP .NET con C# ha demostrado ser una herramienta de programación eficiente y funcional para el control de los procesos de la aplicación. Lo anterior permite que el código sea más fácil de crear y de entender con lo cual las modificaciones futuras podrán hacerse sin dificultad, teniendo como respaldo una buena documentación de los diferentes diagramas UML en la fase de análisis y diseño, así como un buen manual técnico.

El uso íntegro de la plataforma .NET elimina en gran medida el uso de herramientas de terceros como por ejemplo: la encriptación de datos, uso del correo electrónico, etc.

El sistema cumple con los requerimientos establecidos al inicio del proyecto. El contar con un sistema que permita al departamento organizar de manera adecuada la información recibida y generada, así como permitir actualizaciones y consultas eficientes por parte de los integrantes del departamento da como resultado que las decisiones que se lleven a cabo se hagan rápida y oportunamente. Además los servicios tanto del Chat como del correo electrónico proporcionan a los integrantes del departamento herramientas adicionales para contar con una mayor comunicación.

El sistema permite además, entre otras cosas, agregar tipos de usuarios con su respectivo nivel de seguridad que no están contemplados hasta este momento y tipos de documentos para una nueva clasificación, sin perder su funcionalidad el sistema. Las pruebas del tamaño de un archivo a añadir se han hecho hasta el momento con 4 MB, consumiendo un tiempo aceptable de transferencia hacia el servidor.

El presente sistema abre las puertas para ir agregando más opciones de trabajo para que de esta forma se vaya enriqueciendo el sistema. Entre estas se puede mencionar:

- Conversión a formato PDF de los archivos DOC, IMG, etc. en el momento de añadir el archivo a la base de datos, con lo cual se tendrá una homogeneización de los documentos y darles mayor seguridad a través de este formato, además de que su almacenamiento sería un poco más reducido.
- Agregar un apartado para el almacenamiento total de la tesis que se llevan a cabo en el departamento de computación y que pueda ser consultada por la comunidad estudiantil.
- Ampliar el sistema para funcionar con los demás Departamentos que integran la Facultad de Ingeniería

Se espera que este sistema sea aprovechado al máximo y de gran utilidad para el Departamento de Ingeniería en Computación de la Facultad de Ingeniería.

ANEXO I

MANUAL DE USUARIO

CONTENIDO

AI.1 Entrada al Sistema

El presente manual tiene la finalidad de proporcionar la información necesaria para los usuarios que hagan uso del sistema. Para utilizar el sistema, lo primero será ingresar a la siguiente dirección electrónica:

http://microsoft.fi-b.unam.mx/smiaafi

Se desplegará entonces la página de inicio del sistema, tal como se muestra en la Figura AI.1 donde se encuentran los botones de usuarios (para académicos y secretarias) y de administrador. Por default se muestra la parte de validación de datos para los usuarios, donde se pide que se ingrese el Login y Password:

Figura AI.1 Página de inicio del sistema

Si los datos enviados por el usuario son correctos, se le desplegará la página de bienvenida como se ve en la Figura AI.2. En caso contrario se indicará *Login y/o Password incorrecto*.

Figura AI.2 Página de bienvenida

Inicio del Área de Trabajo del Sistema

Esta página, tal como se muestra en la Figura AI.3, es la encargada de proporcionar las distintas opciones que tienen los usuarios. Dependiendo del tipo de usuario, las opciones cambiarán.

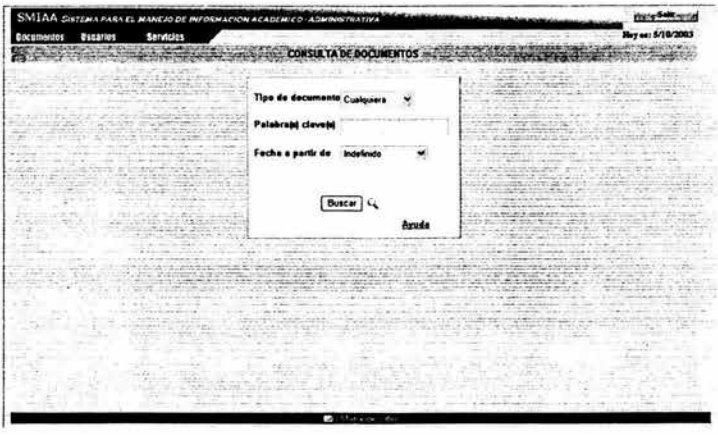


Figura AI.3 Área de trabajo del sistema

AI.2 Control de Documentos

Alta de documentos

Para dar de alta un documento, coloque el cursor del ratón sobre el área *Documentos*. A continuación aparecerá un menú horizontal, en la cual debe seleccionar *Alta*. Se pedirá que seleccione el tipo de documento que será dado de alta. Lo anterior se muestra en la Figura AI.4.

Si en la lista desplegable no está el tipo de documento que quiere dar de alta, entonces puede dar de alta una nueva clasificación de *tipo de documento*. Para esto, presione el botón *¿Registrar un nuevo tipo?*. Aparecerá entonces una pantalla, como se observa en la figura AI.5.

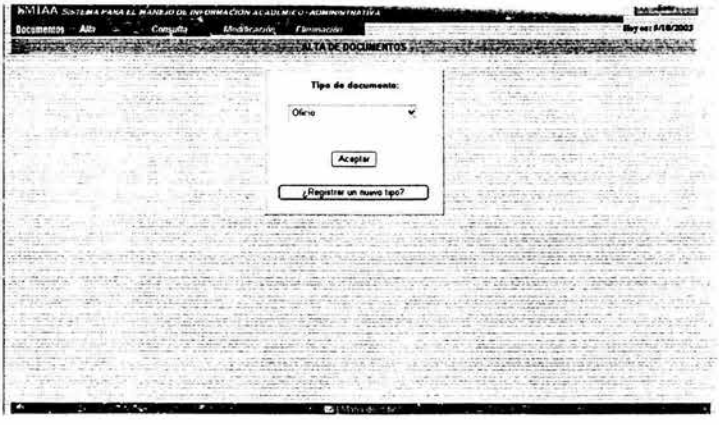


Figura AI.4 Selección para el alta de documentos

Figura AI.5 Alta de un nuevo tipo de documento

En la Figura AI.5, se pide que ingrese la nueva clasificación de tipo de documento. Es necesario que esta clasificación sea muy bien pensado para no redundar con aquellos tipos que ya han sido dados de alta. Por notación, es conveniente que el nombre para el nuevo tipo de documento sea una sola palabra significativa y que la primera letra empiece con mayúscula

Si el tipo de documento ya existe, aparecerá un error. De lo contrario, aparecerá el mensaje *Nueva clasificación de tipo de documento*.

El identificador para esta nueva clasificación es generado automáticamente y no puede ser modificado por el usuario.

Si en la Figura AI.4 se ha seleccionado un documento para ser dado de alta, por ejemplo *Oficio*, aparecerá una nueva ventana como la que se aprecia en la Figura AI.6. Necesitará llenar los campos de forma adecuada para que a la hora de consultar los documentos se haga de forma eficiente.

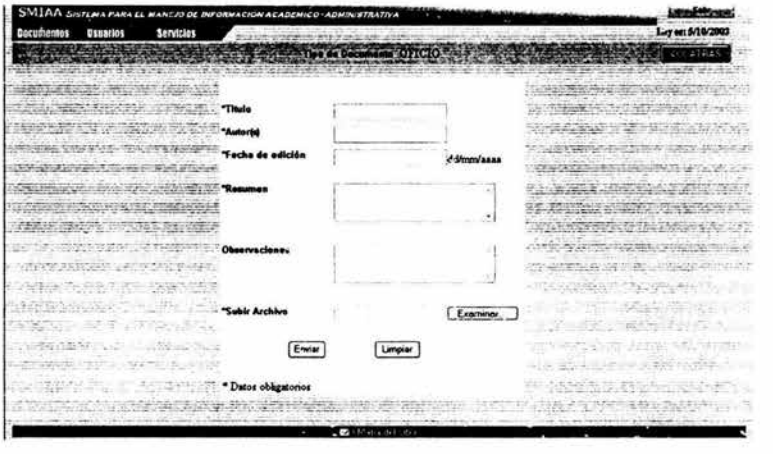


Figura AI.6 Alta de un documento tipo Oficio

Los archivos permitidos para se dados de alta son aquellos que extensión: .doc, .pps, .ppt, .xls, .txt, .pdf, .jpg, .gif, .bmp y .zip

Si el documento que vaya a subir excede de 4 Megabites, es conveniente que comprima el archivo con Winzip. Una vez que los campos hayan sido llenados adecuadamente, se mostrara una pantalla indicando que el documento ha sido dado de alta, tal como se muestra en la Figura AI.7. En esta parte, se tienen dos opciones:

1.- Mandar un aviso por correo electrónico a los académicos sobre el alta de este nuevo documento, presionando el primer botón con lo cual aparece una nueva pantalla (Figura AI.8). En esta pantalla tiene que seleccionar un tipo de usuario y presionar aceptar. Posteriormente aparecerán los académicos que estén dentro esta selección, en donde se podrá seleccionar ya se individualmente los académicos o la opción de *Seleccionar todos*. Una vez realizado lo anterior presionar el botón *enviar*.

Si por alguna razón no se puede enviar el correo, aparecerá un mensaje indicando el error. En caso contrario, aparecerá *correo enviado satisfactoriamente*. La estructura del correo enviado a los académicos es la siguiente:

Titulo: Nuevo(s) Documento(s)
Cuerpo: Para consultar los documentos, por favor visite el sitio Web http://132.248.59.84/smiaa

2.- Mandar el nuevo documento al buzón de algún académico, presionando el segundo botón. Si se lleva a cabo esta acción, aparecerá una ventana como la que se muestra en la Figura AI.9. Aquí se tendrá que escoger un tipo de usuario y presionar aceptar. Posteriormente aparecerán los académicos que estén dentro esta selección, en donde se podrá seleccionar solo un académico. Una vez realizado lo anterior presionar el botón *enviar*.

Si este documento se quiere enviar al mismo buzón, se mostrara un error como el siguiente: *El archivo no pudo ser entregado: el archivo ya existe en su buzón.*

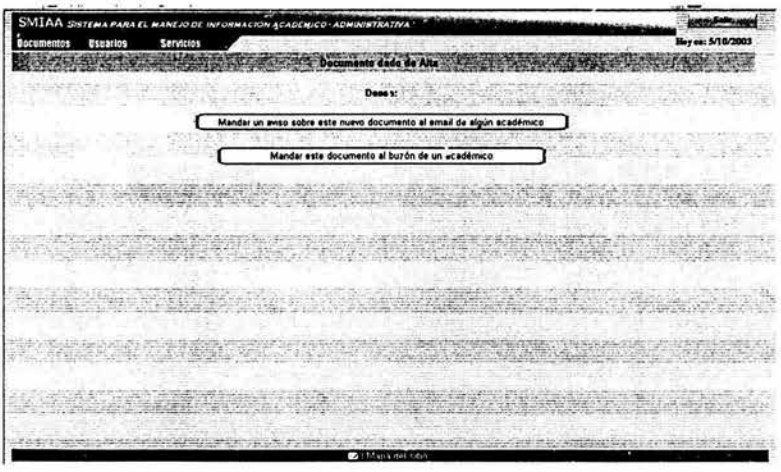


Figura AI.7 Documento dado de alta

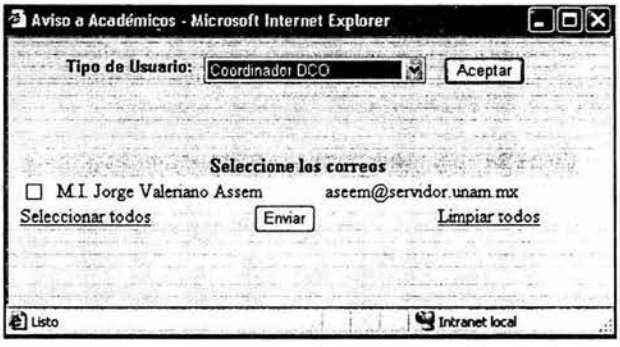


Figura AI.8 Aviso a académicos por email.

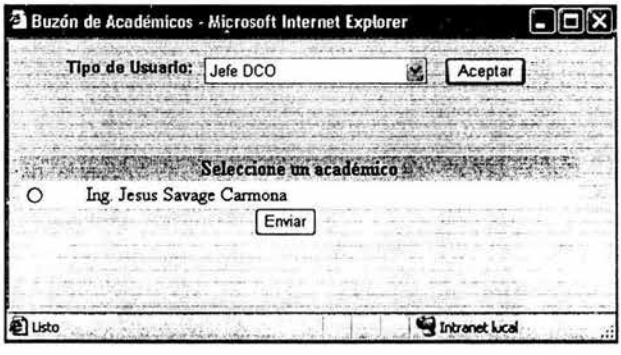


Figura AI.9 Mandar documento al buzón de académicos

Consulta de documentos

Por default, al iniciar el área de trabajo aparecerá esta opción. Si no es así, se puede seleccionar colocando el cursor del ratón sobre el área *Documentos*. A continuación aparecerá un menú horizontal, en la cual debe seleccionar *Consulta*. Esta opción se puede observar en la Figura AI.10.

La búsqueda de un documento se puede llevar a cabo mediante tres opciones:

- Búsqueda por tipo de documento
- Búsqueda por palabra clave
- Búsqueda por fecha

La búsqueda por palabra clave se lleva a cabo en los campos de *título, autor, resumen y observación*. Estos se pueden hacer en 6 formas:

- *Escribiendo una palabra o frase*

De esta forma, el sistema buscará los documentos que contengan la palabra (cadena de caracteres sin espacios ni signos de puntuación) ó frase (una o más palabras con espacios entre cada palabra).
Ejm: concurso de robots

- *Encerrando una frase entre comillas (" ")*

De esta forma, el sistema buscará la frase exacta.
Ejm: "concurso de robots"

- *Agregando el signo |*

Este operador lógico (|) especifica la palabra que puede aparecer en el resultado. Si requiere que aparezca una frase, póngala entre comillas. El operador | corresponde al caracter ASCII 124 (Alt+124)
Ejm: concurso |robot

- *Agregando el signo +*

Este operador lógico (+) especifica la palabra que debe aparecer en el resultado. Si requiere que aparezca una frase, póngala entre comillas.
Ejm: concurso +robot

- *Agregando el signo -*

Este operador lógico (-) especifica la palabra que no debe aparecer en el resultado. Si requiere que aparezca una frase, póngala entre comillas.
Ejm: concurso –robot

- *Agregando el signo ~*

Este operador (~) especifica una palabra o frase que debe estar cerca de otra palabra. Si requiere que aparezca una frase, póngala entre comillas. El operador ~ corresponde al caracter ASCII 126 (Alt+126).
Ejm: concurso ~robot

Para una búsqueda más efectiva, combine los tres tipos de búsqueda (por tipo de documento, por palabra clave, y por fecha)

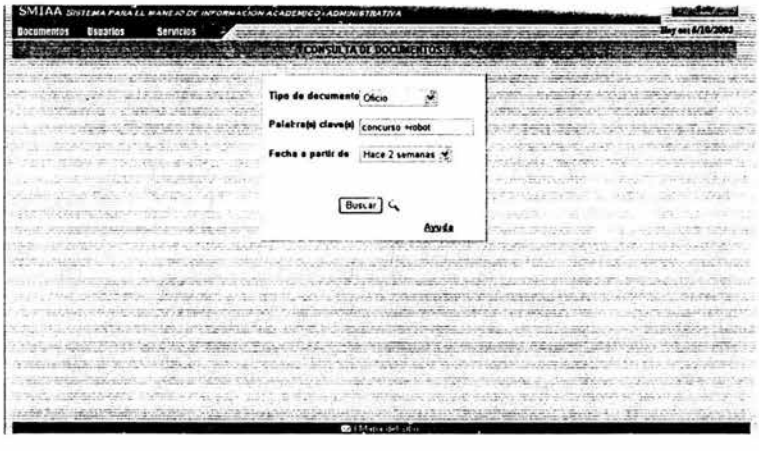


Figura AI.10 Consulta de documentos

Una vez que haya puesto los filtros necesarios o dejar los de default, presione el botón Buscar. Los documentos encontrados serán desplegados por páginas de diez documentos, tal y como se muestra en la Figura AI.11.

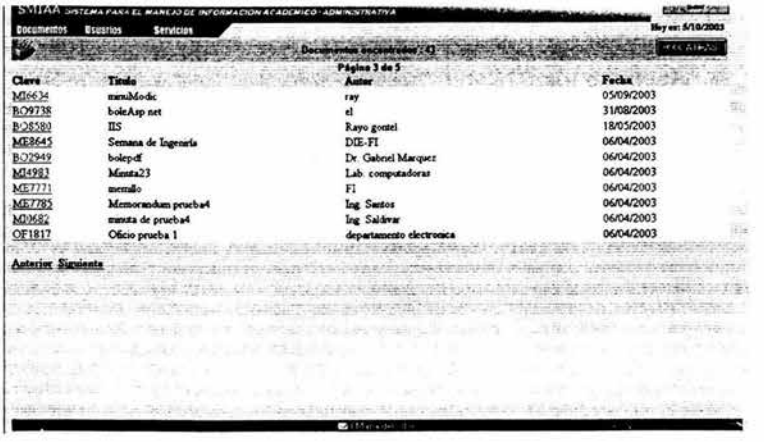


Figura AI.11 Paginación de los documentos encontrados

Para visualizar un documento en particular, presione en la clave del documento. Esto mostrará la información del documento tal y como se puede observar en la Figura AI.12.

Usted verá como aparece además de la información del documento, una liga que permite consultar el archivo del documento en el explorador. Se informa del espacio que ocupa el archivo.

Existe además la opción de mandar este documento a un buzón de algún académico, por si en la parte de *Alta de documento* fue olvidado enviarlo a algún buzón.

Figura AI.12 Consulta de un documento en particular

Modificación de documentos

Para llevar a cabo una modificación a algún documento, coloque el cursor del ratón sobre el área *Documentos*. A continuación aparecerá un menú horizontal, en la cual debe seleccionar *Modificación*. El área de modificación de documentos aparecerá frente a usted, tal como se muestra en la Figura AI.13.

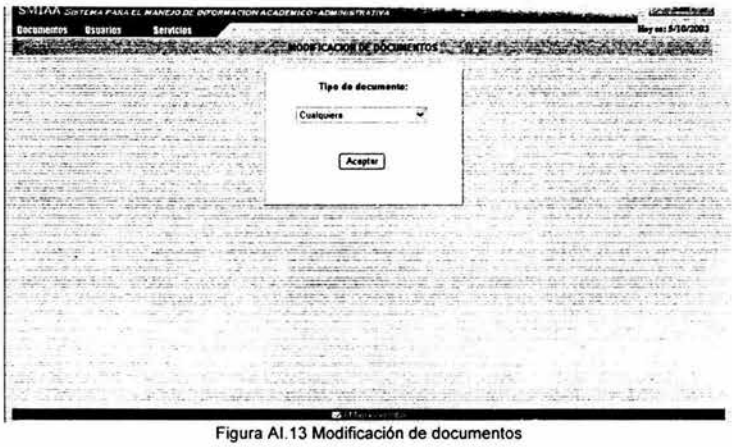


Figura AI.13 Modificación de documentos

En la lista desplegable, seleccione el tipo de documento a modificar. Los documentos encontrados serán devueltos en páginas de diez documentos, como se observa en la Figura AI.14.

Puede elegir solo un documento a modificar marcando el óvalo del documento y presionando el botón *Continuar* >>. A continuación se mostrarán los campos con información del documento, en donde es posible modificar cualquiera de estos. Lo anterior se muestra en la Figura AI.15.

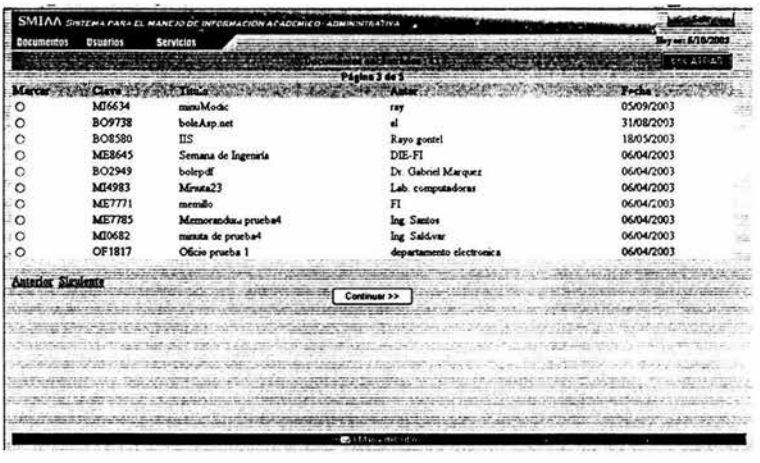


Figura AI.14 Documentos encontrados para su modificación

Cuando llegue a ocurrir un error en la modificación, se indicará un mensaje. De lo contrario, se mostrara un mensaje *Documento modificado.* Durante esta parte, al igual que en la parte de *Alta de documentos,* se puede mandar un aviso a los emails de los académicos, sobre la modificación del documento. Esto se muestra en la Figura AI.16.

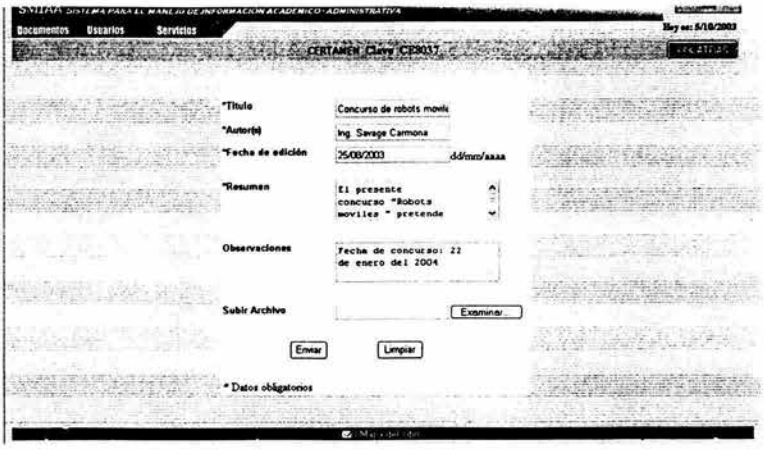


Figura AI.15 Modificación de un documento en particular

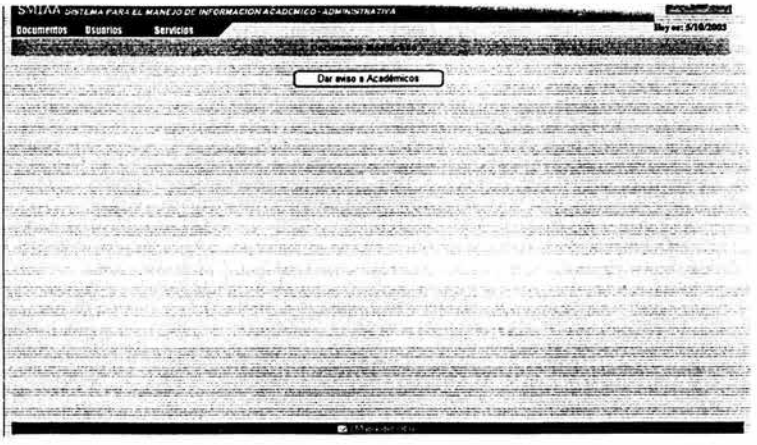


Figura AI.16 Confirmación del documento modificado

Eliminación de documentos

La eliminación de documentos es una opción del sistema exclusiva del administrador. Esto es una medida de seguridad para evitar perdidas de documentos. Cuando un usuario necesite llevar a cabo una eliminación de documento (en este caso las secretarias), necesitarán mandar un correo al administrador. Esta opción está disponible en cualquier página del sistema en la parte inferior.

Para llevar a cabo una eliminación de algún documento, coloque el cursor del ratón sobre el área *Documentos*. A continuación aparecerá un menú horizontal, en la cual debe seleccionar *Eliminación*. La Figura AI.13 es similar a la que se muestra para la eliminación de documentos, donde se debe seleccionar el tipo de documento a eliminar.

Una vez realizado lo anterior, los documentos encontrados se muestran en páginas de diez documentos. Puede elegir uno o más documentos a eliminar marcando el recuadro del documento y presionando el botón *Continuar* >>. Lo anterior se muestra en la Figura AI.17.

Figura AI.17 Eliminación de documentos

La Figura AI.18 muestra el número de documentos eliminados, así como sus claves correspondientes.

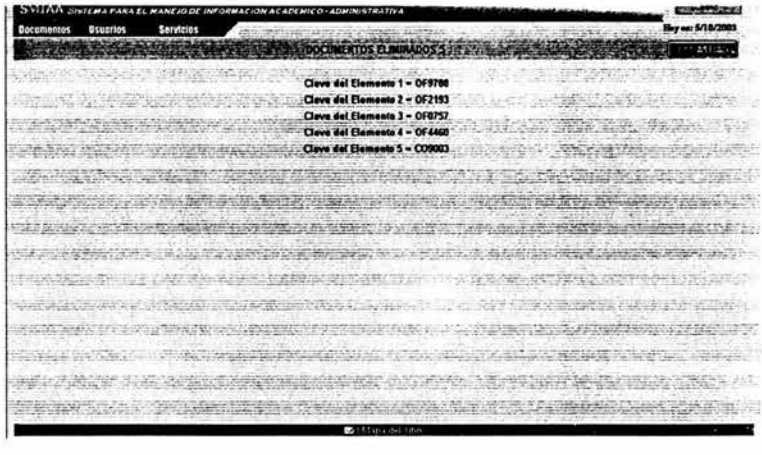


Figura AI.18 Confirmación de los documentos eliminados

AI.3 Control de Usuarios

Alta de usuarios

Para dar de alta un usuario, coloque el cursor del ratón sobre el área *Usuarios*. A continuación aparecerá un menú horizontal, en la cual debe seleccionar *Alta*. Se desplegará una pantalla como se muestra en la Figura AI.19.

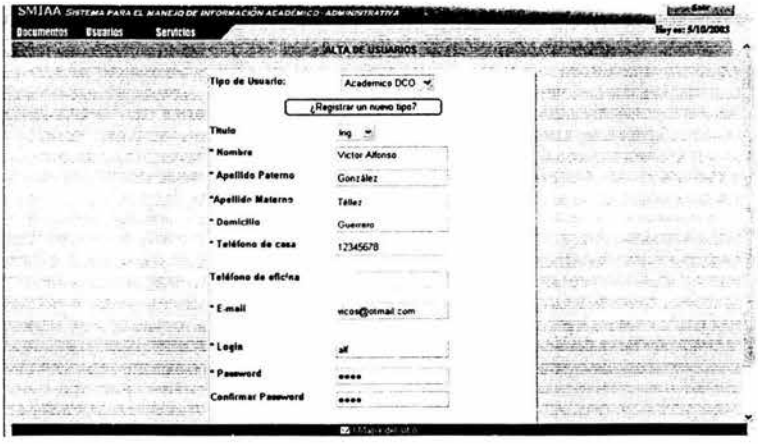


Figura AI.19 Alta de usuarios

Si en la lista desplegable no está definido un *Tipo de usuario* que quiere dar de alta, entonces puede dar de alta una nueva clasificación de *tipo de usuario*. Para esto, presione el botón *¿Registrar un nuevo tipo?*. Aparecerá entonces una pantalla, tal como se muestra en la figura AI.20.

Figura AI.20 Alta de un nuevo tipo de usuario

En la Figura AI.20 se pide que ingrese la nueva clasificación de tipo de usuario. Es necesario que esta clasificación sea muy bien pensado para no redundar con aquellos tipos que ya han sido dados de alta. Por notación, es conveniente que el nombre para el nuevo tipo de documento sea una o máximo tres palabras significativas (usando abreviaturas según sea el caso) y que la primera letra empiece con mayúscula Por ejemplo:

Académico DCO = Académico del Departamento de Computación

Si el tipo de usuario ya existe, aparecerá un error. De lo contrario, aparecerá el mensaje *Nueva clasificación de tipo de usuario*.

El identificador para esta nueva clasificación es generado automáticamente y no puede ser modificado por el usuario.

Regresando a la Figura AI.19, una vez que se han llenado los campos del nuevo usuario satisfactoriamente, por ejemplo para un tipo de usuario *Académico DCO*, aparecerá una nueva ventana al presionar el botón *aceptar* como la que se aprecia en la Figura AI.21, indicando que el usuario ha sido dado de alta.

Cabe hacer mención que el administrador es el único que puede dar de alta usuarios del tipo *Administrador*, el cual tiene todos los privilegios dentro del sistema.

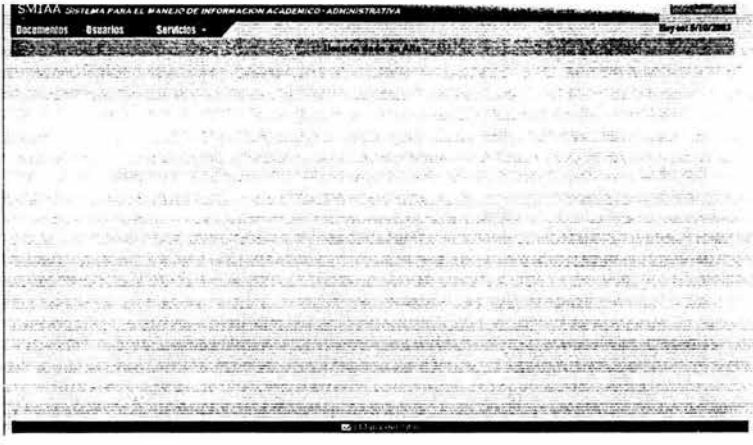


Figura AI.21 Usuario dado de alta

Consulta de usuarios

Para consultar un usuario, coloque el cursor del ratón sobre el área *Usuarios*. A continuación aparecerá un menú horizontal, en la cual debe seleccionar *Consulta*. Se desplegará una pantalla como se muestra en la Figura AI.22.

Figura AI.22 Consulta de usuarios registrados en el sistema

Para ver la información total de un usuario en particular, seleccione su login, el cual mostrará una página como se ve en la Figura AI.23.

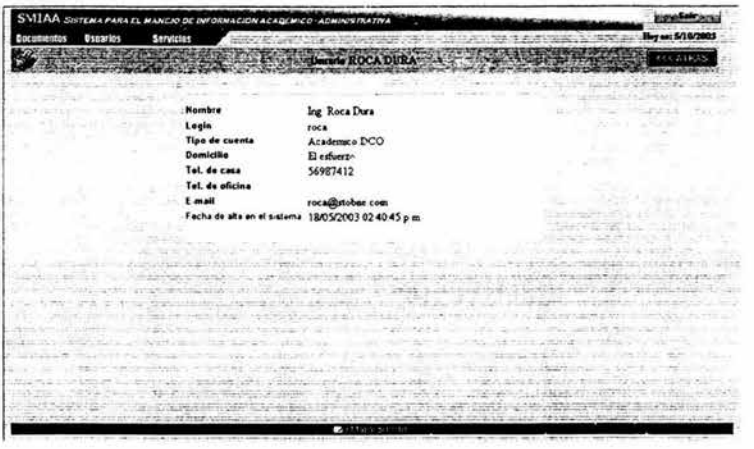


Figura AI.23 Consulta particular de un usuario registrado en el sistema

Modificación de usuarios

Para llevar a cabo una modificación de algún usuario, coloque el cursor del ratón sobre el área *Usuarios*. A continuación aparecerá un menú horizontal, en la cual debe seleccionar *Modificación*. Los usuarios encontrados serán devueltos en una página, como se observa en la Figura AI.24.

Puede elegir solo un usuario a modificar marcando el óvalo del usuario y presionando el botón *Continuar*. A continuación se mostrarán los campos con información del usuario, en donde es posible modificar cualquiera de estos. Lo anterior se muestra en la Figura AI.25.

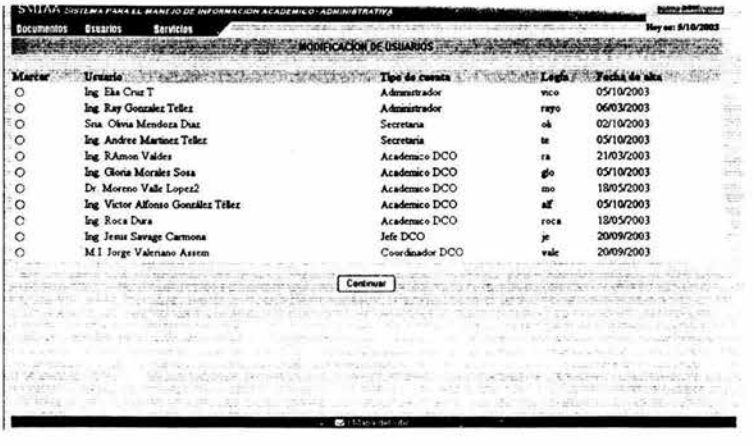


Figura AI.24 Modificación de usuarios

Cuando llegue a ocurrir un error en la modificación, se indicará un mensaje. De lo contrario, se mostrara un mensaje de *Usuario modificado*. Esto se muestra en la Figura AI.26.

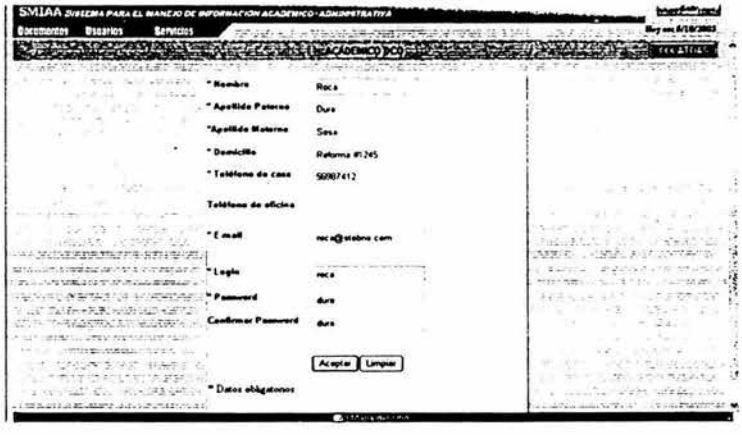


Figura AI.25 Modificación de un usuario en particular

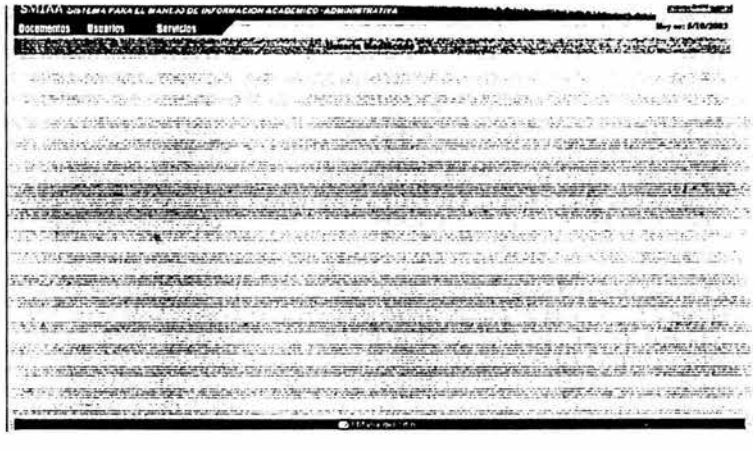


Figura AI.26 Confirmación de la modificación del usuario

Eliminación de usuarios

La eliminación de usuarios es una opción del sistema exclusiva del administrador. Esto es una medida de seguridad para evitar eliminar un usuario accidentalmente. Cuando un usuario necesite llevar a cabo una eliminación de documento (en este caso las secretarias), necesitarán mandar un correo al administrador. Esta opción está disponible en cualquier página del sistema en la parte inferior.

Para llevar a cabo la eliminación de algún usuario, coloque el cursor del ratón sobre el área *Usuarios*. A continuación aparecerá un menú horizontal, en la cual debe seleccionar *Eliminación*. Una vez realizado lo anterior, los usuarios encontrados son mostrados en la misma página. Puede elegir uno o más usuarios a eliminar marcando el recuadro del usuario y presionando el botón *Eliminar*. Lo anterior se muestra en la Figura AI.27.

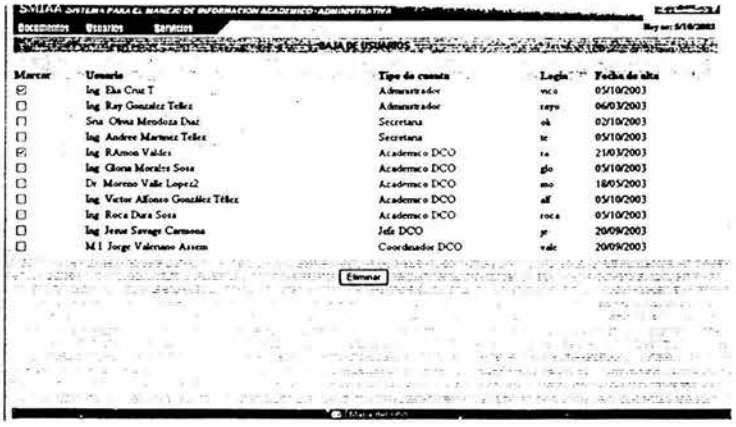


Figura AI.26 Eliminación de usuarios

La Figura AI.27 muestra el número de usuarios eliminados, así como su login correspondiente.

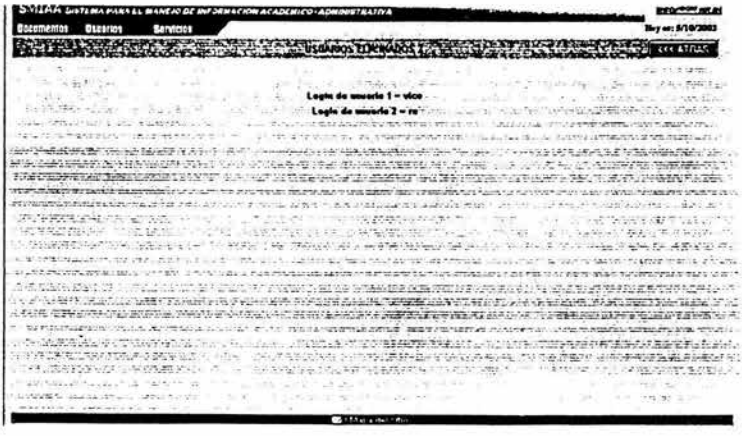


Figura AI.27 Confirmación de la eliminación de usuarios

Responsable del alta de un usuario

Esta opción muestra los primeros veinte documentos que han sido dados de alta recientemente, indicando el usuario que los registró.

Para llevar a cabo esta acción, coloque el cursor del ratón sobre el área *Usuarios*. A continuación aparecerá un menú horizontal, en la cual debe seleccionar *Responsable*. Una vez realizado lo anterior, se muestran este tipo de bitácora. La Figura AI.28 muestra esta pantalla.

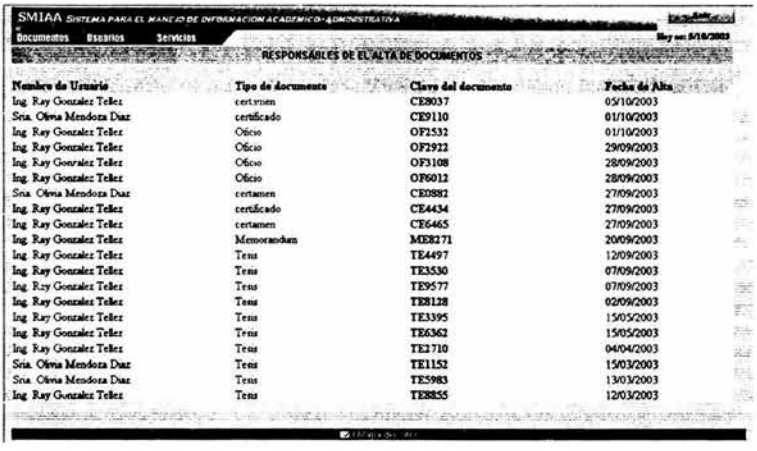


Figura AI.28 Responsables del alta de documentos

AI.4 Servicios

Consultar buzón

Esta opción es única de los académicos. Para consultar su buzón, seleccione el área que dice *Consultar Buzón*. Se desplegará una pantalla como la que se muestra en la Figura AI.29.

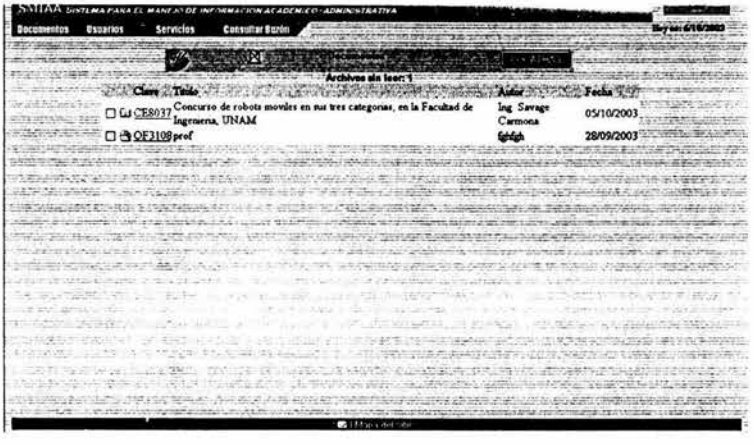


Figura AI.29 Consulta del buzón

El académico puede observar en su buzón, los documentos que le han sido enviados así como el número de documentos sin leer los cuales muestran una figura de una carpeta cerrada.

Para consultar un documento, basta con seleccionar su clave y aparecerá la información del documento, muy similar a como se muestra en la Figura AI.12, con excepción de que no muestra la opción *mandar este documento al buzón de académicos...*

El académico puede eliminar un documento de su buzón, marcando el recuadro y seleccionando la figura que muestra un documento con un X. Lo anterior muestra una pantalla similar a la Figura AI.18.

Una parte importante de la eliminación, es que en realidad no se borrar el documento físicamente de la base de daros, si no únicamente la referencia hacia ese documento. Por lo tanto si el académico necesita de nuevo el documento, tendrá que hacer uso del servicio de correo electrónico para enviar una petición ya sea a la secretaria o al administrador del sistema.

También se puede imprimir la lista de los documentos que aparecen en su buzón, seleccionando la figura de la impresora.

Servicio de correo electrónico

El servicio de correo electrónico es una parte fundamental del sistema con la cual se puede enviar un mensaje, con la opción de anexar un archivo, a cualquiera de los usuarios registrados en el sistema.

Para poder utilizar este servicio, coloque el cursor del ratón sobre el área *Servicios*. A continuación aparecerá un menú horizontal, en la cual debe seleccionar *Email*. Una vez realizado lo anterior, se muestran una pantalla como la Figura AI.30.

Al momento de abrir este servicio, automáticamente es llenado el campo del remitente con el correo del usuario actual del sistema.
Como se observa en la Figura AI.30, existe una liga nombrada *Incluir usuarios* con lo cual puede incluir uno o mas destinatarios. Estos usuarios son los que están registrados en el sistema. La Figura 31 muestra la pantalla que se obtiene al presionar en esta liga.

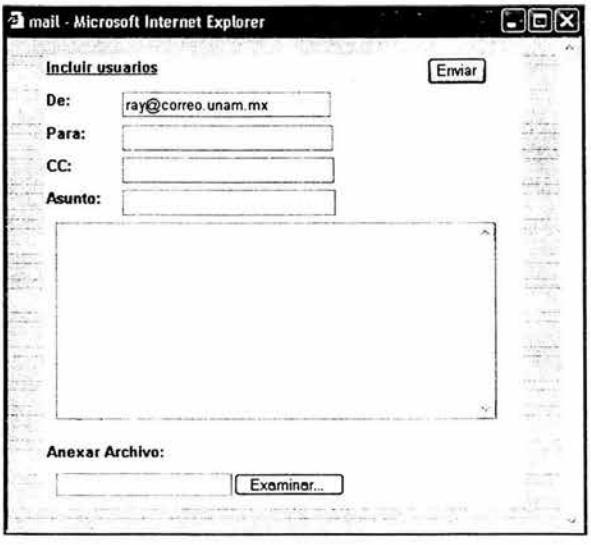


Figura AI.30 Servicio de correo electrónico

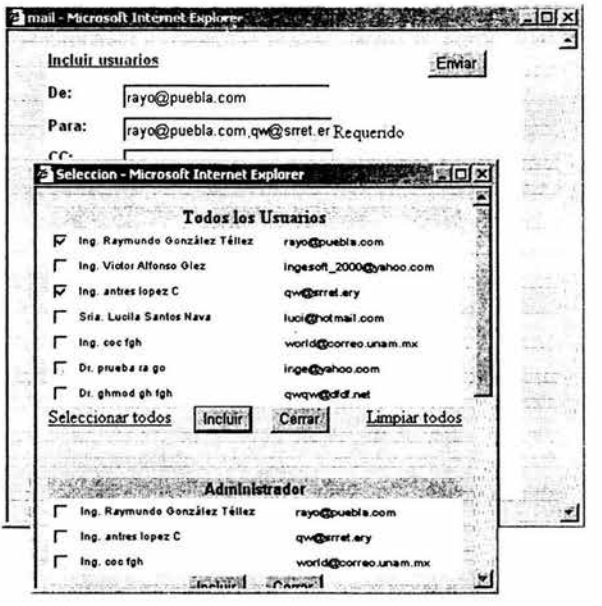


Figura AI.31 Incluir usuarios en el campo *Para*

Si requiere anexar un archivo muy grande, es recomendable que lo comprima.

Servicio de Chat

El servicio de Chat es una herramienta en el sistema que permitirá tener reuniones entre académicos sin que estos tengan que desplazarse a un lugar fijo.

Para poder utilizar este servicio, coloque el cursor del ratón sobre el área *Servicios*. A continuación aparecerá un menú horizontal, en la cual debe seleccionar *Chat*. Una vez realizado lo anterior, aparecerá una pantalla como la que se muestra en la Figura AI.31.

El usuario tendrá que utilizar un nick para ingresar al Chat. Si el nick esta siendo utilizando por otro usuario, se le mostrará un mensaje de que el *nick ya existe en el sistema*. El color con el que se mostrará este nick en el Chat es opcional en su elección.

Una vez que el nick se aceptado, se mostrará la pantalla de entrada al chat. La Figura AI.32 muestra esta pantalla. Aquí se pueden observar tres areas: Visualización de los mensajes enviados, los usuarios en línea y la parte de envío de mensajes.

Figura AI.31 Servicio de Chat del sistema

Presionando la liga *Salir* que se muestra en la Figura AI.32, se cierra la sesión del usuario en al Chat.

Figura AI.32 Entrada el Chat

ANEXO II

MANUAL TÉCNICO

CONTENIDO

El manual técnico es importante como parte de la documentación del sistema para una futura modificación del mismo. Esta manual debe ser capaz de ser entendido por usuarios con un cierto grado de conocimiento en las tecnologías que se usaron para el desarrollo de este sistema.

AII.1 Instalación del Servidor IIS

Como cualquier otro software de Windows, la instalación de IIS 6.0 es tan sencilla como hacer un doble clic de ratón. Normalmente es uno de los componentes de Windows 2003 Server que viene seleccionado por defecto. Si no fuera así, en la propia instalación de Windows 2003 se puede seleccionar bajo el epígrafe Componentes de Windows.

Para instalar manualmente habría que ir al Panel de Control->Añadir o Quitar Programas y hacer clic sobre el icono Agregar o Quitar componentes de Windows. Una vez lanzado el Asistente para Componentes de Windows, seleccionar Servidor de aplicaciones y en esta se encontrara la opción de Servicios de Internet Information Server. Esto se muestra en la Figura AII.1.

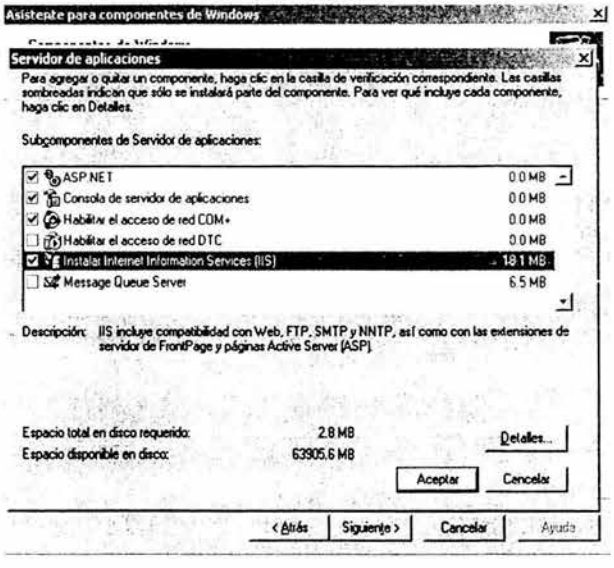


Figura AII.1 Asistente para componentes de Windows

Configuración del servidor IIS

Cada sitio web tiene asociadas una serie de propiedades que definen su comportamiento. Por tanto, el administrador es libre de cambiar este comportamiento modificando sus propiedades. Estas propiedades se pueden modificar a través de las páginas de propiedades, y pueden referirse al sitio, al directorio o a un fichero en cuestión.

La página de propiedades del sitio Web de este sistema se obtiene en la MMC (Microsoft Management Console) o Administrador de servicios de Internet pulsando el botón derecho sobre el sitio Web predeterminado y eligiendo el menú propiedades, tal como se muestra en la Figura AII.2.

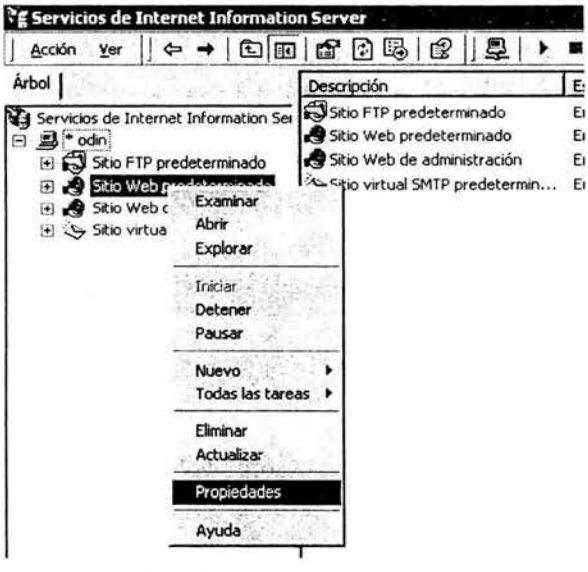


Figura AII.2 Selección de las propiedades del sitio web

Una vez elegida esta opción, se nos presentará una pantalla como la mostrada en la Figura AII.3, en donde podemos llevar a cabo las configuraciones necesarias para el funcionamiento del sitio Web.

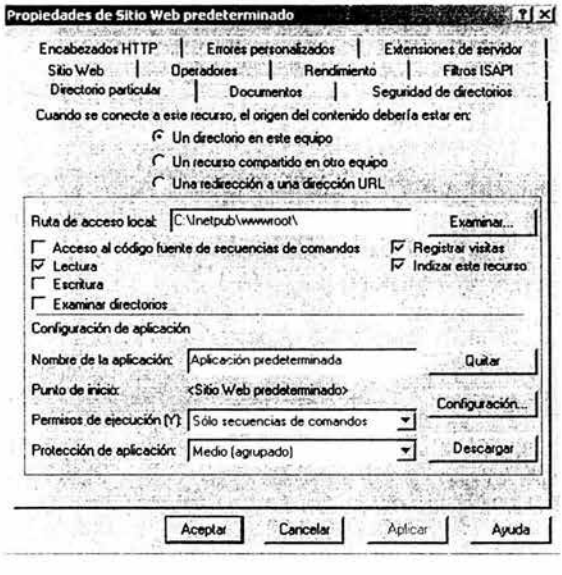


Figura AII.3 Ventana de configuración del sitio web

Entre las configuraciones que se pueden llevar a cabo están:

- Sitio Web. En esta lengüeta, además de definir la identificación del sitio Web, podemos definir el número de conexiones que aceptará nuestro servidor Web. También podremos habilitar un registro o log de los accesos y errores del sitio Web.
- Operadores. Los operadores del sitio Web son usuarios definidos en Windows 2003 que poseen permisos para alterar la configuración y el funcionamiento del servidor Web. Aquí añadiremos aquellos usuarios que deseamos administren el sitio Web.
- Rendimiento. En esta pestaña podremos ajustar una serie de parámetros que influirán en el rendimiento del sitio Web. Los parámetros que se configuran para cada sitio, prevalecen sobre los definidos en el servidor
- Documentos. Aquí definiremos el documento predeterminado que se mostrará si se invoca este sitio directamente sin indicar un página concreta.
- Encabezados HTTP. Utilizaremos esta pestaña para configurar los valores que se enviarán al navegador en el encabezado de la página HTML.

AII.2 Archivos del sistema

La implementación del sistema esta organizado en folders jerarquerizados, tal como se muestra en la Figura AII.4. La estructura es la siguiente:

- La raíz principal del proyecto se muestra en la figura XXX en negritas correspondiente al sistema "smiaa" la cual contiene los archivos y carpetas esenciales para el funcionamiento del sistema.

  Los archivos de la aplicación están integrados en tres módulos:
  - Módulo de Validación
  - Módulo de administración
  - Módulo de usuarios

  Por otra parte, las carpetas que integran la aplicación son:

  - References: esta carpeta hace referencia a los componentes necesarios que la aplicación necesita para su funcionamiento.
  - bin: esta carpeta contiene los assemblys de la aplicación.
  - Chat: como su nombre lo indica, esta carpeta contiene los archivos necesarios que conforman el Chat de la aplicación.
  - Email: contiene los archivos encargados de manejar el envió de correo electrónico.
  - Imágenes: contiene los archivos gráficos de la aplicación.

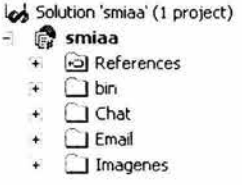


Figura AII.4 Composición de la aplicación Web

Los diferentes tipos de archivos que integran la aplicación son:

- .aspx Son los archivos ASP .NET que contiene los controles de servidor y código HTML.
- .cs Contienen el código que proporciona la interacción con el usuario, validación entre otras funcionalidades. Este código es llamado "code-behind"

- .dll Tipos de archivos que representan a los assemblys de la aplicación.
- Global.asax Usado para definir variables de sesión y de aplicación.
- Web.config Usado para la configuración de la aplicación Web.
- .htm Archivo html que representa a algunas páginas estáticas de la aplicación.
- .css Hojas de estilo usados para dar formato a algunos elementos del código HTML.
- .swf archivos flash que representan a algunos botones de la aplicación.
- .jpg y .gif son los tipos de archivos gráficos.

Instalación de los archivos del sistema SMIAA

La instalación de los archivos que componen a este sistema bastará únicamente con colocar la carpeta "smiaa" que contiene los archivos en la ruta C:\Inetpub\wwwroot, tal como se muestra en la Figura AII.5.

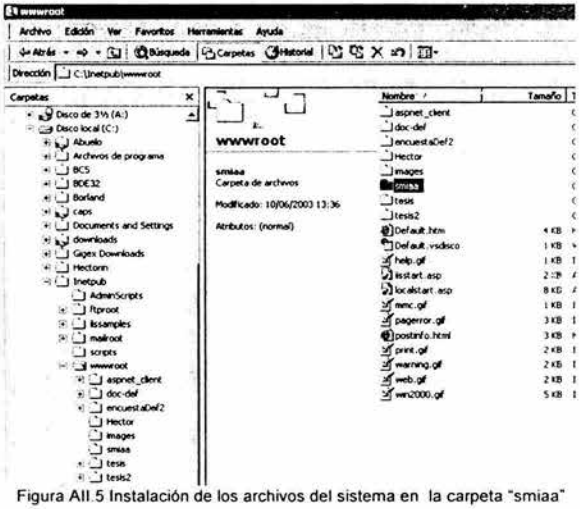


Figura AII.5 Instalación de los archivos del sistema en la carpeta "smiaa"

Composición de los archivos del sistema

A continuación se mencionan cada uno de los atributos (variables de aplicación, sesión, identificadores de formularios) y métodos (funciones en asp .net, java script) de las páginas asp.net que conforman el sistema, así como una breve descripción.

En todas las páginas, si la sesión del usuario es incorrecta, se redirecciona a la página default.aspx

default.aspx		
Atributos	**Métodos**	**Descripción**
PanelUsr PanelAdm	Page_Load() usuario_Click() administrador_Click()	Página de inicio del sistema. Se encarga de visualizar el panel de usuario o de administración, según sea el caso.

valida_usuario.aspx		
Atributos	**Métodos**	**Descripción**
logusr Pasusr Log1 Pas1 PasEncriptado Session[valord] Session[valorn]	Checar()	Página que muestra el formulario de validación para el usuario, ya sea secretaria o académico. Los datos entrados son redireccionados a la página valida_usuario2.aspx

valida_adm.aspx		
Atributos	**Métodos**	**Descripción**
logadm Pasadm Log1 Pas1 PasEncriptado Session[valord] Session[valorn]	Checar()	Página que muestra el formulario de validación para el administrador. Los datos entrados son redireccionados a la página valida_adm2.aspx

valida_usuario2.aspx		
Atributos	**Métodos**	**Descripción**
enc Login pass RSA PasDesencripado sql rs bytebuf Session[login] Session[password] Session[nom] Session[apat] Session[amat] Session[permiso]	Conexión()	Página que contiene el código para validar la entrada del usuario. Si usuario es aceptado, se redirecciona a la página transición.aspx. De otro modo, se redirecciona a la página de error mensaje.aspx

valida_adm2.aspx		
Atributos	**Métodos**	**Descripción**
Enc Loginadm Passadm RSA PasDesencripado sql rs bytebuf Session[login] Session[password] Session[nom] Session[apat] Session[amat] Session[permiso]	Conexión()	Página que contiene el código para validar la entrada del administrador. Si es aceptado, se redirecciona a la página transición.aspx. De otro modo, se redirecciona a la página de error mensaje.aspx

transicion.aspx		
Atributos	**Métodos**	**Descripción**
permiso	replace()	Se redirecciona al tipo de usuario, dependiendo del permiso, a la página de bienvenida.

mensaje.aspx		
Atributos	**Métodos**	**Descripción**
Permiso sql rs mensaje msg	Conexión()	Muestra en pantalla el tipo de error ocurrido durante una violación de algunas páginas de l sistema.

bienvenidoadm.aspx		
Atributos	**Métodos**	**Descripción**
Permiso nom ap1 ap2		Pantalla de bienvenida para el administrador.

bienvenidousr.aspx		
Atributos	**Métodos**	**Descripción**
Permiso nom ap1 ap2		Pantalla de bienvenida para la secretaria.

bienvenidoacad.aspx		
Atributos	**Métodos**	**Descripción**
Permiso nom ap1 ap2		Pantalla de bienvenida para el académico.

area_adm.aspx		
Atributos	**Métodos**	**Descripción**
Permiso		Frameset para el área de trabajo del administrador. El Frameset está compuesto por las páginas opciones.aspx, main.aspx y pie.aspx.

opciones.aspx		
Atributos	**Métodos**	**Descripción**
Fecha	now() mmLoadmenus()	Frame que contiene el menú de opciones para el área de administración del sistema.

opcionesusr.aspx		
Atributos	**Métodos**	**Descripción**
Fecha	now() mmLoadmenus()	Frame que contiene el menú de opciones para el área de trabajo de las secretarias.

opcionesusr2.aspx		
Atributos	**Métodos**	**Descripción**
Fecha	now() mmLoadmenus()	Frame que contiene el menú de opciones para el área de trabajo de los académicos.

main.aspx		
Atributos	**Métodos**	**Descripción**
		Frame que sirve de contenedor para las múltiples páginas disponibles en las áreas de trabajo de los usuarios.

pie.aspx		
Atributos	**Métodos**	**Descripción**
permiso	NuevaVentana()	Frame residente como pie de página de las áreas de trabajo de los usuarios.

tipodoc.aspx		
Atributos	**Métodos**	**Descripción**
permiso sql rs tipodoc	NuevaVentana() Conexion()	Esta página muestra al cliente los tipos de documentos que puede dar de alta. El ID del documento seleccionado es enviado a la página altadoc.aspx

altadoc.aspx		
Atributos	**Métodos**	**Descripción**
permiso tipodoc Session[doc]		Esta página tiene la función de redireccionarse dependiendo de si el documento es o no una tesis. Si es tesis, se redirecciona a la página altatesis.aspx. De lo contrario, se redirecciona a altadoc2.aspx

altadoc2.aspx		
Atributos	**Métodos**	**Descripción**
titulo autor fecedi resumen observación	Page_Load() Conexión() btnEnviar_Click() Clave() Extensión(nombreDoc, 4)	Lleva a cabo la validación y el formateo del documento. Si este es correcto, se almacena en la base de datos. De lo contrario, genera un tipo de

archivo permiso sesID tipoIDDoc claveDoc horaDoc usuarioID nombreDoc extensionDoc tipoDoc tamañoDoc Docbuffer clave lonCad sql	Documento(tipoIDDoc) NomDoc(nombreDoc)	error.

altatesis.aspx		
Atributos	**Métodos**	**Descripción**
titulo autor asesor fecedi clave resumen observación permiso tipoDoc claveDoc horaDoc responsableAlta nomDoc	Page_Load() Conexión() btnEnviar_Click() Clave()	Lleva a cabo la validación del documento Tesis. Si este es correcto, se almacena en la base de datos. De lo contrario, genera un tipo de error.

nuevodoc.aspx		
Atributos	**Métodos**	**Descripción**
id tipodoc permiso sql rs valor	Conexión() Checar()	Muestra un formulario para el alta de un nuevo tipo de documento. El Id del doc. Es generado automáticamente. Los datos son enviados a la página nuevodoc2.aspx.

nuevodoc2.aspx		
Atributos	**Métodos**	**Descripción**
identifica tipodoc sql	Conexión()	Almacena el nuevo tipo de documento en la base de datos.

seleccionarusr.aspx		
Atributos	**Métodos**	**Descripción**
permiso		Frameset para seleccionar a los usuarios, a quienes se les avisará por correo electrónico sobre al alta de un nuevo documento. El Frameset está

		compuesto por las páginas escogeusr.aspx, y nada.aspx.

escogeusr.aspx		
Atributos	**Métodos**	**Descripción**
sql rs tipousr	Conexión()	Frame que visualiza los usuarios almacenados por categoría en la base de datos.

muestrausr.aspx		
Atributos	**Métodos**	**Descripción**
permiso sql rs tipousr elemento	Conexión() Check() Clear()	Visualiza una categoría de usuarios en particular almacenados en la base de datos.

enviaremail.aspx		
Atributos	**Métodos**	**Descripción**
permiso tabla1 valores mail tipousr elemento	Conexión() Page_Load()	Se encarga de enviar el correo al o a los usuario(s).

consulta.aspx		
Atributos	**Métodos**	**Descripción**
permiso sql rs tipodoc clave fechadesde	Conexión() OpenWindow()	Muestra un formulario para la búsqueda de un documento en particular. La petición es enviada a la página búsqueda.aspx

busqueda.aspx		
Atributos	**Métodos**	**Descripción**
permiso pag iEstado itotal aDatos cdatos I, J Datos Ban, ban2 sql, sql2, sql3, sql4 rs, rs2, rs 3, rs4 cero Claves Clave Tipod Fechadesde	Conexión() Busqueda()	Se lleva a cabo la búsqueda de acuerdo a las opciones seleccionados por el usuario. Este tipo de búsqueda trata de encontrar el significado de lo que si quiere buscar y no solo las palabras claves coincidentes. Esta página manda los resultados encontrados a la página grpaginar.aspx

Fd, Fd2 punto referencia frases campos session[paginabuzon] session[Datos]		

grpaginar.aspx		
Atributos	**Métodos**	**Descripción**
permiso pag iEstado iTotal iComienzo iFin iPaginas iPagActual aDatos bDatos I, J Datos ban, ban2 sql rs, rs2 datos TotalDoc	printWindow() PaginarResultados(10, pag,Datos)	Se encarga de mostrar los resultados de la búsqueda en páginas.

despliegueedoc.aspx		
Atributos	**Métodos**	**Descripción**
permiso clavedoc PLDC sql, sql2.sql3 rs, rs2 IdeUsr nombre punto extension nombre2 diagonal extension2 Session[tip] Session[ClaveDoc] Session[paginabuzon]	Conexión() OpenWindow() printWindow()	Se encarga de mostrar las características del documento seleccionado.

obtenerarchivo.aspx		
Atributos	**Métodos**	**Descripción**
permiso claved sql Rs	Conexión()	Se encarga de obtener de la base de datos y mostrar en pantalla el archivo del documento seleccionado.

ayudaconsulta.aspx		
Atributos	**Métodos**	**Descripción**
permiso		Muestra la ayuda al usuario sobre la forma en que puede llevar a cabo la consulta de los documentos.

seleccionarusr2.aspx		
Atributos	**Métodos**	**Descripción**
permiso		Frameset para seleccionar a los académicos, a quienes se les enviará a su buzón respectivo algún documento particular. El Frameset está compuesto por las páginas escogeusr2.aspx, y nada.aspx.

escogeusr2.aspx		
Atributos	**Métodos**	**Descripción**
sql rs tipousr	Conexión()	Frame que visualiza los académicos almacenados por categoría en la base de datos.

muestrausr2.aspx		
Atributos	**Métodos**	**Descripción**
permiso sql rs tipousr elemento	Conexión() Check() Clear()	Visualiza una categoría de académicos en particular almacenados en la base de datos.

enviarabuzon.aspx		
Atributos	**Métodos**	**Descripción**
permiso documento valores valor selección poncero sql, sql2, sql3 rs, rs2 insertauno	Conexión()	Se encarga de enviar al buzón del académico un documento en particular.

eliminar.aspx		
Atributos	**Métodos**	**Descripción**
permiso sql rs tipodoc	Conexión()	Muestra un formulario para seleccionar un tipo de documento en particular para eliminar. La petición es enviada a la página borrar.aspx

borrar.aspx		
Atributos	**Métodos**	**Descripción**
permiso sql, sql2, sql3, sql4 rs, rs2, rs3, rs4 itotal aDatos cDatos I, J Datos ban, ban2 Tipod campos fin registrosdoc registrostes Session[datos]	Conexión()	Encuentra todos los documentos de acuerdo a su tipo seleccionado en la base de datos para ser eliminado. Los documentos encontrados son enviado a la página paginareli.aspx

paginareli.aspx		
Atributos	**Métodos**	**Descripción**
permiso pag iEstado iTotal iComienzo iFin iPaginas iPagActual aDatos bDatos I, J Datos ban, ban2 sql rs, rs2 datos TotalDoc	PaginarResultados(10, pag,Datos)	Se encarga de mostrar en pantalla los resultados de la búsqueda de los documentos para ser eliminados. Estos resultados son mostrados en páginas de diez documentos.

borradoc.aspx		
Atributos	**Métodos**	**Descripción**
permiso valores valor borra valordef sql, sql2, sql3, sql4, sql5 sql6, sql7 tipodoc	Conexión()	Borra de la base de datos el (o los) documento(s) seleccionado(s).

modificar.aspx		
Atributos	**Métodos**	**Descripción**
permiso sql	Conexión()	Muestra un formulario para seleccionar un tipo de

rs tipodoc		documento en particular para modificar. La petición es enviada a la página modifica.aspx

modifica.aspx		
Atributos	**Métodos**	**Descripción**
permiso sql, sql2, sql3, sql4 rs, rs2, rs3, rs4 itotal aDatos cDatos I, J Datos ban, ban2 Tipod campos fin registrosdoc registrostes Session[datos]	Conexión()	Encuentra todos los documentos de acuerdo a su tipo seleccionado en la base de datos para ser modificado. Los documentos encontrados son enviado a la página paginarmod.aspx

paginarmod.aspx		
Atributos	**Métodos**	**Descripción**
permiso pag iEstado iTotal iComienzo iFin iPaginas iPagActual aDatos bDatos I, J Datos ban, ban2 sql rs, rs2 datos TotalDoc	PaginarResultados(10, pag,Datos)	Se encarga de mostrar en pantalla los resultados de la búsqueda de los documentos para ser modificados. Estos resultados son mostrados en páginas de diez documentos.

modificadoc.aspx		
Atributos	**Métodos**	**Descripción**
permiso modif valor Session[clave]	Conexión()	Redirecciona a la página modificatesis.aspx y es del tipo Tesis. De lo contrario, redirecciona a la página modificadoc2.aspx

modificatesis.aspx		
Atributos	**Métodos**	**Descripción**
titulo autor asesor fecedi clave resumen observación permiso claveTes sql, sql2, sql3 horaDoc usuarioID	Conexión() Page_load() ValoresIniciales(clave) btnEnviar_Click()	Modifica en la base de datos el documento seleccionado.

modificadoc2.aspx		
Atributos	**Métodos**	**Descripción**
titulo autor fecedi resumen observación archivo permiso valorClave tipoDoc claveDoc horaDoc usuarioID nombreDoc extensionDoc tipoDoc tamañoDoc Docbuffer lonCad sql	Page_Load() Conexión() ValoresIniciales(clave) btnEnviar_Click() Extensión(nombreDoc, 4) NomDoc(nombreDoc)	Lleva a cabo la validación y el formateo del documento. Si este es correcto, se modifica en la base de datos. De lo contrario, genera un tipo de error.

altausr.aspx		
Atributos	**Métodos**	**Descripción**
nombre apat apmat domicilio telcasa teloficina email Login pass pass2 permiso Tipousr claveUsr horaDoc aceptar	Page_Load() Conexión() Clave() btnEnviar_Click() ValidarUsr(log,pass)	Permite ingresar y validar los datos para el alta de un nuevo usuario en el sistema.

sql, sql2, sql3, sql4 ID IDTemp		

nuevoper.aspx		
Atributos	**Métodos**	**Descripción**
id tipoper permiso sql rs ide valor	Conexión() Checar()	Muestra un formulario para el alta de un nuevo tipo de usuario. El Id del tipo de usuario es generado automáticamente. Los datos son enviados a la página nuevoper2.aspx.

nuevodoc2.aspx		
Atributos	**Métodos**	**Descripción**
identificador tipoper sql	Conexión()	Almacena el nuevo tipo de usuario en la base de datos.

consultausr.aspx		
Atributos	**Métodos**	**Descripción**
permiso sql rs ban	Conexión() printWindow()	Muestra los usuarios del sistema.

despliegueusr.aspx		
Atributos	**Métodos**	**Descripción**
permiso loginusr sql rs Ban	Conexión() openWindow() printWindow()	Muestra la información de un usuario en particular.

eliminarusr.aspx		
Atributos	**Métodos**	**Descripción**
permiso sql rs ban	Conexión()	Muestra los usuarios del sistema que pueden ser elegidos para su eliminación.

borrausr.aspx		
Atributos	**Métodos**	**Descripción**
permiso valores valor valordef borra sql, sql2, sql3, sql4, sql5 rs	Conexión()	Elimina de la base de datos del sistema al usuario seleccionado en la página eliminarusr.aspx.

modificarusr.aspx		
Atributos	**Métodos**	**Descripción**
permiso sql rs ban	Conexión()	Muestra los usuarios del sistema que pueden ser modificados.

modificausr.aspx		
Atributos	**Métodos**	**Descripción**
nombre apat apmat domicilio telcasa teloficina email Login pass pass2 permiso valorLogin horaDoc IDUSR aceptar sql, sql2, sql3 ID logViejo	Page_Load() Conexión() ValoresIniciales() btnEnviar_Click() ValidarUsr(log)	Modifica en la base de datos del sistema al usuario seleccionado en la página modificarusr.aspx.

responsablealta.aspx		
Atributos	**Métodos**	**Descripción**
Permiso sql, sql2 rs, rs2 ban	Conexión()	Muestra los usuarios responsables de cada uno de las altas de documentos en el sistema.

buzon.aspx		
Atributos	**Métodos**	**Descripción**
permiso usuario ideUsr sql, sql2 rs, rs2 ban sinrevisar Session[paginabuzon]	Conexión() printWindow()	Muestra los documentos particulares de cada académico. Dichos documentos pueden ser revisados o eliminados por el académico.

borrarefdoc.aspx		
Atributos	**Métodos**	**Descripción**
permiso usuarioID valores valor borra sql	Conexión()	Borra la referencia del documento particular del académico

rs sinrevisar Session[paginabuzon]		

mail.aspx		
Atributos	**Métodos**	**Descripción**
permiso de para cc asunto mensaje archivo nomArchivo path	Button1_click()	Permite enviar correo electrónico.

muestrausr.aspx		
Atributos	**Métodos**	**Descripción**
permiso sql rs valor ide TipoUsr toAddress	conexion() Check() Clear() regresaDirecciones() regresaDirecciones2()	Permite seleccionar el correo electrónico de un usuario, poniendo dicho correo en el campo de texto *para* de la página *mail.aspx*.

abrirchat.aspx		
Atributos	**Métodos**	**Descripción**
color apodo Tempo Valor Application[Usuarios] Session[color] Sesion[ID]	Checar()	Pantalla de validación para entrar al chat del sistema.

abrirchat2.aspx		
Atributos	**Métodos**	**Descripción**
mensaje color apodo Tempo Valor Application[Usuarios] Session[color] Sesion[ID]	Checar()	Si un nickname ya existe en el chat es redireccionado a esta página.

default.aspx		
Atributos	**Métodos**	**Descripción**
Application[Opciones]	Yaunclick()	Frameset del área del chat. Contiene a las páginas visualizacion.aspx, incluir.aspx y listausr.aspx.

incluir.aspx		
Atributos	**Métodos**	**Descripción**
color apodo Temporal Opinion Application[Opiniones]	Yaunclick()	Permite introducir los comentarios en el área del chat.

visualizacion.aspx		
Atributos	**Métodos**	**Descripción**
PaginaActual Auxiliar[] temporal opinion Application[Opiniones]	Yaunclick()	Visualiza los comentarios introducidos por los usuarios en el chat.

listausr.aspx		
Atributos	**Métodos**	**Descripción**
PaginaActual Estado[] temporal2 usuario Application[Usuarios]	Yaunclick()	Visualiza los usuarios conectados en el chat.

cerrar.aspx		
Atributos	**Métodos**	**Descripción**
Estado[] Temporal temporal2 i Session[ID] Application[Usuarios] Application[Opiniones]	Yaunclick()	Permite borrar las variables de aplicación del usuario cuando sale del chat

salsesion.aspx		
Atributos	**Métodos**	**Descripción**
session.Abandon		Elimina la sesión del usuario. El usuario es redireccionado a la página prohibir.htm

prohibir.htm		
Atributos	**Métodos**	**Descripción**
Session[permiso]		Muestra en pantalla "la sesion ha expirado" y una liga para la página default.aspx.

AII.3 Estructura de la Base de Datos del Sistema

La base de datos del sistema está compuesta por los objetos mostrados en la Tabla AII.1.

Objeto	Cantidad
Tablas	14
Llaves primarias	11
Llaves foraneas	14
Indices (Clustered)	11
Procedimientos almacenados	4
Defaults	4
Catálogos Full-Text	1

Tabla AII.1 Objetos que componen a la base de datos del sistema

La Tabla AII.2 muestra el nombre de cada uno de los objetos que integran la base de datos.

Nombre	Tipo de objeto
archivo	user table
buzon	user table
car_doc	user table
car_tes	user table
contraseña	user table
dtproperties	user table
fechalta_doc	user table
fechalta_tes	user table
fechalta_usr	user table
mensaje	user table
permiso	user table
permiso_usr	user table
responsable_altadoc	user table
tipo_doc	user table
usuario	user table
SP_AltaDoc	stored procedure
SP_AltaTes	stored procedure
SP_ModificaDoc	stored procedure
SP_ModificaTes	stored procedure
PK_archivo	primary key cns
PK_car_doc	primary key cns
PK_car_tes	primary key cns
PK_contraseña	primary key cns
PK_fechalta_doc	primary key cns
PK_fechalta_tes	primary key cns
PK_fechalta_usr	primary key cns
PK_mensajes	primary key cns
PK_permiso	primary key cns
PK_tipo_doc	primary key cns
PK_usuario	primary key cns
FK_archivo_car_doc	foreign key cns
FK_buzon_car_doc	foreign key cns
FK_buzon_usuario	foreign key cns
FK_car_doc_tipo_doc	foreign key cns
FK_car_tes_tipo_doc	foreign key cns

FK_contraseña_usuario	foreign key cns
FK_fechalta_doc_car_doc	foreign key cns
FK_fechalta_tes_car_tes	foreign key cns
FK_fechalta_usr_usuario	foreign key cns
FK_permiso_usr_permiso	foreign key cns
FK_permiso_usr_usuario	foreign key cns
FK_responsable_altadoc_car_doc	foreign key cns
FK_responsable_altadoc_car_tes	foreign key cns
FK_responsable_altadoc_usuario	foreign key cns
DF_car_doc_esbuzon	default (maybe cns)
DF_fechalta_doc_fecha	default (maybe cns)
DF_fechalta_tes_fecha	default (maybe cns)
DF_fechalta_usr_fecha	default (maybe cns)

Tabla AII.2 Nombres de los objetos de la base de daos

Almacenamiento de la base de datos y del catálogo full-text

La ruta física donde resíden los archivos de la base de datos, tanto el archivo de datos como el de transacciones, se encuentra en C:\Archivos de programa\Microsoft SQL Server\MSSQL\Data. Ambos archivos están configurados de tal forman que se incrementan en tamaño automáticamente a un 10%. En la Figura AII.6 se observa la configuración para el archivo de datos.

Figura AII.6 Configuración del archivo de datos

La pantalla de configuración para el archivo de transacciones es similar.

En la creación del catálogo full-text se crea una carpeta independiente de la base de datos. Dicha carpeta se encuentra en C:\Archivos de programa\Microsoft SQL Server\MSSQL\FTDATA. En la Figura AII.7 se muestra el contenido de esta carpeta:

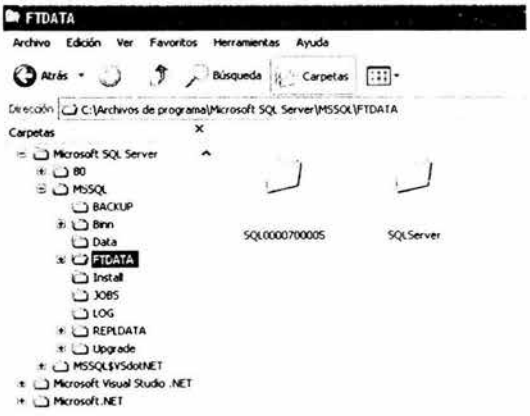


Figura AII.7 Contenido de a carpeta del catálogo full-text

Tablas del sistema

A continuación se muestra en la Tabla AII.3 las tablas del sistema, así como los campos que las integran con sus respectivas características.
PK = Llave primaria

Tabla	Campo	Tipo de dato	Tamaño	Descripción
tipo_doc	tipo_id (PK)	smallint	2	Identificador para el tipo de documento.
	tipo_doc	varchar	50	Nombre del tipo de documento.
car_doc	clave_doc (PK)	varchar	10	Clave que identifica al documento
	tipo_id	smallint	2	Llave foránea que hace referencia al tipo de documento.
	titulo	varchar	250	Contiene el titulo del documento.
	autor	varchar	50	Contiene el autor del documento.
	fecha_edi	smalldatetime	4	Guarda la fecha en que se publicó el documento.
	resumen	varchar	1000	Contiene un resumen del documento,

	observación	varchar	250	Contiene observaciones del documento.
	esbuzon	Bit	1	Testifica si un documento pertenece al buzón de un académico
fechalta_doc	clave_doc (PK)	varchar	10	Llave foránea que hace referencia a la clave del documento.
	fecha	datetime	8	Guarda la fecha en que se almacenó el documento en el sistema.
	hora	varchar	15	Guarda la Hora en que se almacenó el documento en el sistema.
archivo	clave_doc (PK)	varchar	10	Llave foránea que hace referencia a la clave del documento.
	archivo	Image	16	Guarda el archivo de un documento.
	nombre	varchar	255	Contiene el nombre del archivo.
	extension	varchar	5	Guarda la extensión del archivo
	tipo	varchar	50	Guarda el tipo MIME del archivo.
	tamano	int	4	Guarda el tamaño del archivo.
Car_tes	clave_tes (PK)	varchar	10	Clave que identifica a la tesis.
	tipo_id	smallint	2	Llave foránea que hace referencia al tipo de documento.
	titulo	varchar	250	Contiene el titulo

				de la tesis.
	autor	varchar	50	Contiene el autor o autores de la tesis
	fecha_edi	smalldatetime	4	Guarda la fecha en que se termino la tesis.
	asesor	varchar	50	Contiene el nombre del director de tesis.
	clave_fi	varchar	10	Guarda la clave que asigna el departamento de computación a una tesis.
	resumen	varchar	1000	Contiene un resumen de la tesis.
	observacion	varchar	250	Contiene observaciones de la tesis.
fechalta_tes	clave_tes (PK)	varchar	10	Llave foránea que hace referencia a la clave de la tesis.
	fecha	datetime	8	Guarda la fecha en que se almacenó la tesis en el sistema.
	hora	varchar	15	Guarda la hora en que se almacenó la tesis en el sistema.
usuario	usuario_id (PK)	smallint	2	Clave que identifica a un usuario.
	titulo	varchar	50	Grado de estudio del usuario.
	nombre	varchar	30	Contiene el nombre del usuario.
	ap_pat	varchar	30	Contiene el apellido paterno

				del usuario
	ap_mat	varchar	30	Contiene el apellido materno del usuario
	domicilio	varchar	50	Contiene el domicilio del usuario
	telcasa	varchar	20	Contiene el número telefónico de la casa del usuario.
	teloficina	varchar	20	Contiene el numero telefónico de la oficina del usuario.
	email	varchar	50	Contiene el correo electrónico del usuario.
permiso	permiso_id (PK)	smallint	2	Identificador para el tipo de permiso.
	tipo_per	varchar	50	Contiene el nombre del tipo de permiso.
permiso_usr	usuario_id	smallint	2	Llave foránea que hace referencia al identificador de usuario.
	permiso_id	smallint	2	Llave foránea que hace referencia al identificador de un tipo de permiso.
contraseña	usuario_id (PK)	smallint	2	Llave foránea que hace referencia al identificador de usuario.
	login	varchar	50	Contiene el login del usuario.
	contraseña	varbinary	50	Contiene el password del usuario
fechalta_usr	usuario_id (PK)	smallint	2	Llave foránea que hace referencia al

				identificador de usuario.
	fecha	datetime	8	Guarda la fecha en que se dio de alta al usuario.
	hora	varchar	15	Guarda la hora en que se dio de alta al usuario.
buzon	usuario_id	smallint	2	Llave foránea que hace referencia al identificador de usuario.
	clave_doc	varchar	10	Llave foránea que hace referencia a la clave del documento.
	revisado	Bit	1	Testifica si un documento que pertenece al buzón de un usuario ha sido revisado.
responsable_altadoc	usuario_id	smallint	2	Llave foránea que hace referencia al identificador de usuario.
	clave_doc	varchar	10	Llave foránea que hace referencia a la clave del documento.
	clave_tes	varchar	10	Llave foránea que hace referencia a la clave de la tesis.
mensaje	mensaje_id (PK)	tinyint	1	Identificador para un tipo de mensaje.
	descripción	varchar	75	Contiene una descripción de un mensaje en particular.

Figura AII 3 Tablas de la base de datos

Relación entre tablas

La Tabla AII.4 muestra la relación existente entre una tabla padre y una tabla hija, así como el nombre de la llave foránea que las relaciona.

Tabla padre	Tabla hija	Llave foránea
car_doc	archivo	FK_archivo_car_doc
car_doc	fechalta_doc	FK_fechalta_doc_car_doc
car_doc	Buzon	FK_buzon_car_doc
car_doc	responsable_altadoc	FK_responsable_altadoc_car_doc
car_tes	fechalta_tes	FK_fechalta_tes_car_tes
car_tes	responsable_altadoc	FK_responsable_altadoc_car_tes
tipo_doc	car_doc	FK_car_doc_tipo_doc
tipo_doc	car_tes	FK_car_tes_tipo_doc
usuario	Contraseña	FK_contraseña_usuario
usuario	fechalta_usr	FK_fechalta_usr_usuario
usuario	permiso_usr	FK_permiso_usr_usuario
usuario	Buzon	FK_buzon_usuario
usuario	responsable_altadoc	FK_responsable_altadoc_usuario
permiso	permiso_usr	FK_permiso_usr_permiso

Figura AII.4 Relación entre tablas

A continuación se muestra, en la Figura AII.8, el diagrama de relación de la base de datos:

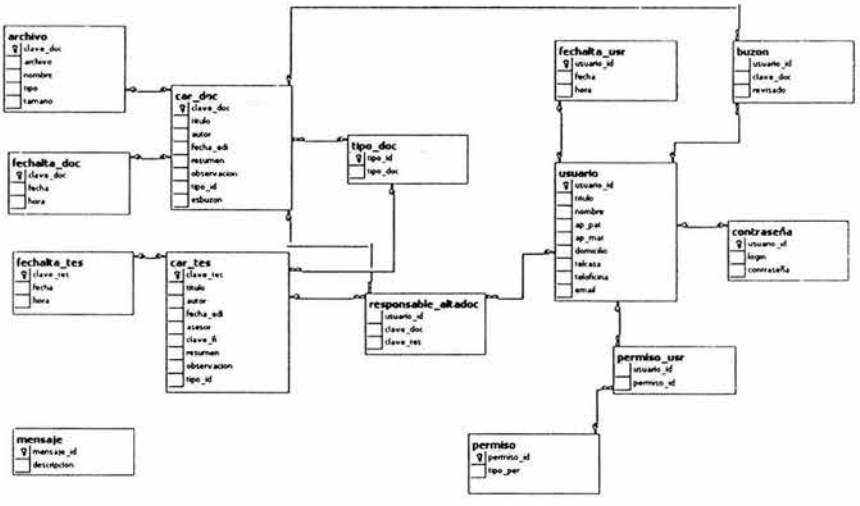


Figura AII.8 Diagrama de relación de la base de datos

Catálogo Full-Text del sistema

Este sistema cuenta con un catálogo Full-Text "FTsmiia" para el sistema de búsqueda que utiliza el servicio de SQL Server 2000 llamado "Microsoft Search"

Las tablas que han sido definidas para el indexamiento de Full-Text en el catálogo se muestran en la Tabla AII.5.

Tabla	Campos	Indice único Full-Text
car_doc	titulo autor resumen observación	PK_car_doc
car_tes	titulo autor resumen observación	PK_car_tes
Archivo	archivo Extension	PK_car_doc

Tabla AII.5 Tablas definidas para Full-Text Search

La actualización de los índices se lleva a cabo en segundo plano cada vez que un documento es almacenado en la base de datos.

BIBLIOGRAFÍA

Duthi G. Andrew "ASP .NET Programming With Microsoft Visual C# .NET". Microsoft Press. Versión 2003

"SQL Server y Servicios de datos". Microsoft Training and Certification 2002

"Plataforma .NET, lenguaje C# y Componentes". Microsoft Training and Certification 2002

R. Pressman "Ingeniería del software". McGraw Hill, 1997.

Solórzano Palomares, Fernando "Computadoras y Programación: Volumen 1". Facultad de Ingeniería, 1995.

SITIOS WEB

MSDN Online
http://www.microsoft.com/latam/vstudio/
http://msdn.microsoft.com/net/aspnet
http://www.microsoft.com/net/
http://msdn.microsoft.com/sqlserver

Tutorial ASP .NET
http://es.gotdotnet.com/quickstart/aspplus/doc/quickstart.aspx