UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES
"ARAGÓN"

"PROGRAMACIÓN PARA INTERNET/INTRANET Y SERVIDORES WWW CON ENFOQUE UML"

T E S I S

QUE PARA OBTENER EL TÍTULO DE :
INGENIERO EN COMPUTACIÓN
P R E S E N T A :
R A Ú L S Á N C H E Z S Á N C H E Z

ASESOR:
MAT. LUÍS RAMÍREZ FLORES

SAN JUAN DE ARAGÓN, EDO. MÉX.
2006

Agradecimientos.

A la UNAM por haberme abierto sus puertas y su espíritu.

Agradezco a Luis Ramírez Flores su interés y empatía que se refleja en sus excelentes clases, sin duda el mejor profesor del área de matemáticas.

A Juán Mendez Moreno, profesor que con su partida deja un gran hueco que será difícil de cubrir, pero que generosamente se sembró en cada uno de nosotros.

A mis amigos que solamente siendo como son, hicieron sacudir mi mente, algunos de ellos excelentes y destacados profesionistas.

ÍNDICE

Introducción.

En este trabajo se pretende explicar los pasos para programar en ambiente internet/intranet con enfoque UML, para ir aclarando esto, primero se expone la historia y los conceptos; para después ir explicando la programación e instalación de los servidores de Internet. Con estos pasos tan sencillos es posible instalar cualquier Intranet; teniendo como base esto, el cómo hacer estos pasos y el enfoque, es lo que hace la diferencia y lo que se pretendió desarrollar en este trabajo. El trabajo está formado por nueve capítulos que tratan de buscar respuestas a estos problemas.

Programar en el entorno de Internet/Intranet es una tarea cada vez más común en el mundo de la computación, es por eso que se hace necesario contar con herramientas que permitan una organización adecuada de las ideas para comunicarlas con facilidad, cuando se realizan proyectos que involucran una gran cantidad de software es común realizar modelos intuitivos o bien modelos matemáticos o administrativos muy complejos que satisfacen hasta cierto punto las expectativas de los desarrolladores de software pero no así a los usuarios de los sistemas y los clientes, en todo caso la sinergia entre los desarrolladores y los usuarios finales, es una meta deseable. En este trabajo se trata de exponer el desarrollo de sistemas con tecnología de Internet utilizando el enfoque orientado a objetos que nos permite la reutilización de los sistemas, además de que se realiza una breve exposición de una nueva herramienta orientada a objetos, el Lenguaje Unificado de Modelado (UML). Teóricamente, el UML es un lenguaje simbólico que nos permite el modelado del cualquier sistema, aunque originalmente fue diseñado para el desarrollo de sistemas que involucran una gran cantidad de software, nos permite entender el sistema desde la perspectiva del usuario, de los diseñadores y de todas las partes involucradas en el mismo. En el primer capítulo realizo una reseña histórica del Internet con el propósito de ubicar las Intranets, posteriormente se exponen los elementos indispensables para el diseño de una Intranet, su funcionamiento, etc. En el capítulo tres se exponen los elementos de los sistemas de información, la forma en como se organizan, la organización de la información en los diferentes bloques etc. En cada uno de estos bloques hay un determinado tipo de información que puede ser procesada con la tecnología de cómputo, la mejor forma de modelar esta información es por medio del análisis y diseño orientado a objetos, representando cada uno de estos bloques como si fueran clases, vamos a dar una visión de la orientación a objetos para posteriormente representar en diagramas, nuestros bosquejos que vayamos desarrollando, realizado esto, vamos a juntarlo para representar y organizar la estructura de la información para clarificar de la mejor forma nuestro problema, en los capítulos 6, 7 y 8 vamos a realizar la programación y un pequeño prototipo de un sistema de información, en cada uno de estos capítulos vamos a exponer ejemplos de elementos indispensables en una Intranet y la creación de los contenidos. En el último capítulo vamos a instalar y configurar dos servidores Web donde se albergará nuestro prototipo.

Justificación

La programación Internet/Intranet, es una oportunidad para saber el alcance de modelar y pensar en Objetos. La programación ya es global, basta con encender una máquina, ponernos a trabajar sobre una aplicación que nos interese, y consultar por Internet que tanto se ha estudiado y desarrollado acerca de dicha aplicación y ponerse a trabajar. La programación ya no es algo aislado, para muestra basta con que reflexionemos sobre las aplicaciones que son relevantes en la actualidad, por ejemplo, el servidor Apache es una aplicación de código abierto, y contribuyen para su desarrollo decenas de programadores en todo el mundo, esta idea ya está en el ámbito doméstico cuando se programa.

Para desarrollar este trabajo, se consultó por Internet que tanto se ha investigado sobre este tema, y con base a esto se consideró importante desarrollar un trabajo sencillo que proporcione las nociones para crear aplicaciones de Internet/Intranet Orientadas a Objetos.

Ahora bién, el concepto Internet/Intranet es un concepto tan de moda y tan difundido que lamentablemente se ha pasado por alto, que buena parte de ellas son compradas y mantenidas por compañías externas a la organización, es difícil encontrar libros, páginas electrónicas que expliquen de forma sencilla la construcción de una Intranet, se trató de buscar manuales, bibliografía, y todo tipo de documentación que permitiera a una persona construir el prototipo de una Intranet de forma sencilla, y aunque hay bibliografía que menciona que una Intranet es algo tan sencillo y tan difícil como se quiera, es decir que hay intranets que se pueden armar en un día, hasta proyectos muy elaborados que pueden tardar algunos años. El intento de crear un trabajo sencillo que nos explique paso a paso, los elementos para la creación de intranets, la explicación de su desarrollo utilizando el enfoque de Orientación a Objetos, y programación para Internet/Intranets, es lo que nos va a ocupar en la mayor parte de esta investigación.

Objetivo General.

Adquirir los elementos para desarrollar aplicaciones sencillas para Internet/Intranet con Orientación a Objetos, entendiendo los elementos básicos y realizando aplicaciones sencillas para estar en posibilidades de desarrollar y entender las ventajas y desventajas de utilizar el enfoque Orientado a Objetos.

Tesis.

La Orientación a Objetos no es la panacea en la programación, programar con objetos tiene sus ventajas, entre ellas, los planos del software son entendibles y permite una mejor sinergia entre las partes. Las desventajas, es que todavía no hay un lenguaje cien por ciento orientado a objetos, en la orientación a objetos hay diversas corrientes y a veces resulta difícil juntar todas las corrientes en una sola. Para realizar el diseño de planos con orientación a objetos se requieren de grandes esfuerzos de abstracción y de visión para que un sistema sea realmente reutilizable, además de que los especialistas en objetos son muy escasos hasta este momento.

Capítulo I. Una reseña histórica.

1.1. Los Orígenes de Internet[1]

- ✓ **Un poco de historia**.

Internet nació en 1968 cuando el gobierno de los Estados Unidos desarrolló un plan para comunicar a cuatro de sus mejores Universidades mediante una red conocida como ARPANET, el origen de esta red es ARPA (Agencia de Proyectos de Investigación Avanzada), esta agencia fue creada por el gobierno que estaba comprensiblemente preocupado por el avance militar de la Unión Soviética al lanzar el Sputnik en 1957. El plan era intercambiar información valiosa y acelerar la investigación armamentista… Esta forma de intercambiar la información se realizaba a base de paquetes de información junto con su dirección de destino, se transferían por líneas telefónicas normales, debido a que los paquetes eran pequeños requerían poco ancho de banda. Las cuatro Universidades que formaban ARPANET eran: Standford Research Institute(SRI), University of California en Santa Barbara(UCSB), University of California en Los Angeles(UCLA) y la Universidad de Utah(UU). Pero había un gran problema, era completamente difícil comunicar a ARPANET, entonces surgió la necesidad de crear un conjunto de reglas que los paquetes de datos pudieran entender no importando la computadora destino, este conjunto de reglas se le denominó protocolo. *Un protocolo determina las reglas de intercambio de datos entre dos o más sistemas.* El primer protocolo de intercambio de datos fue diseñado por estudiantes graduados de estas cuatro universidades, las reglas de este protocolo son sencillas, y cualquiera podría contribuir a la especificación ya que no era algo oficial, pero cuando se volvió a retomar el proyecto por profesionales, el protocolo se volvió fijo, este protocolo se conocía como Petición de Comentarios (Request For Coments-RFC). Como las definiciones eran públicas, cualquier programador podía utilizarlos para programar una computadora que pudiera entenderlos. Esto fue la causa de que ARPANET se extendiera a tal grado que ya conectaba a la mayor parte de las universidades más importantes de los Estados Unidos en 1972.

Este propósito de dudosa ética resultó beneficioso para el nacimiento de Internet, que en estos momentos del ataque armado contra Irak es la herramienta tecnológica que utilizan las organizaciones de todo tipo para oponerse en todo el planeta a la invasión de un país pobre donde se están lanzando más bombas y misiles que en toda la segunda guerra mundial junta.

- ✓ **Protocolos.**

Un protocolo son aplicaciones que determinan las reglas de intercambio de datos que deben seguir dos o más sistemas. Se diseñaron con la finalidad de que la comunicación entre computadoras se realizara con mayor facilidad, el primer protocolo RFC, comunicó a la mayor parte de las universidades más importante de los Estados Unidos, que un año después dio lugar al protocolo de Internet de más uso en la actualidad, el protocolo TCP/IP (Protocolo de Control de Transferencia/Internet Protocolo). Como consecuencia, todas las máquinas que se comunicaban con Internet tenían que utilizar este protocolo. No todos los sistemas operativos cuentan con TCP/IP. Como es el caso de las redes que trabajan bajo Novell Netware, para este tipo de sistemas operativos se utiliza un proceso de conversión que se conoce como *pasarela* mediante un protocolo Novell llamado IPX (Intercambio de Paquetes de Internet) este protocolo se inventó para que las computadoras que trabajan bajo Novell puedan comunicarse con otras computadoras que usan protocolo TCP/IP de aquí salieron los protocolos e-mail y Telnet.

- ✓ **Protocolo e-mail.**

Es un conjunto de reglas de intercambio de datos donde se define cómo se deben entregar y recibir los mensajes.

- ✓ **Protocolo Telnet.**

Este protocolo permite la interacción entre una computadora cliente y un servidor remoto. Este host es normalmente un sistema UNIX.

- ✓ **Protocolos de red.**

El protocolo de red utilizado para conectar páginas Web es el http, el protocolo ftp: permite descargar archivos desde un servidor de FTP. El protocolo malito: se utiliza para correo electrónico, el protocolo News establece comunicación de intercambio con servidores de grupos de noticias. Gopher, Verónica, jughead y WAIS son protocolos que localizan información por todo el mundo. Esto es, consiste en aplicaciones constantemente actualizadas que generan menús de la información disponible por Internet.

- ✓ **El Protocolo World Wide Web**.

El físico Timothy Berners Lee, físico del laboratorio europeo de física de las partículas en Ginebra Suiza, propuso en 1989 un sistema de hiperenlaces, un sistema de enlaces que permitiera desplazarse de una computadora a otra, para poder realizar esto, se creó un lenguaje de programación sencillo llamado HTML(Lenguaje de Marcas de Hipertexto, *Hipertext Markup Language*), con este lenguaje se crean documentos sencillos que contienen información que es parecida a la que contienen los documentos que se realizan en los procesadores de palabras. Estos documentos contienen información sobre un tema concreto que nos interesa, estos documentos están organizados de tal forma que uno observa frases o palabras resaltadas, este es el hiperenlace que representa un tema o una frase que habla de un tema en específico, cuando se le da un clic con el ratón nos lleva a una página con las mismas características que la anterior con el tema que se activo en el hiperenlace en cuestión, este tipo de documentos se llaman páginas Web. Desplazarse de una página Web a otra a través de hiperenlaces es a lo que se le llama navegar. Ahora bien

Cuando se multiplicó la comunicación de las redes por medio de páginas Web a través de estos hiperenlaces, Berners Lee llamó a este sistema World Wide Web, el problema es que antes estas páginas se veían en modo de texto. Marc Andreeson, estudiante de la Universidad de Illinois diseño el primer visualizador gráfico de Internet, que hacía más sencillo, este salto es equivalente al que se dio del Sistema Operativo en modo texto a Sistema Operativo en modo gráfico como UNIX o Windows. Este primer visualizador se llamó mosaic. La empresa Netscape Corporation con el mosaic de Andreeson desarrolló la primera versión de su primer visualizador de Internet llamado Netscape Corporation, que está disponible en Internet desde 1994. La visualización de la información en modo gráfico es la principal y revolucionaria ventaja de los visualizadores, pero ¿Cómo buscar lo que uno necesita en un océano inmenso de datos? La respuesta está en los motores de búsqueda.

La enorme facilidad para encontrar la información es la ventaja principal, esto se realiza a través de motores de búsqueda que localizan los registros en una base de datos inmensa ubicada en Servidores de Sistemas de Nombres de Dominio (DNS). Estos

motores son localizadores de listados de registros que contienen direcciones IP. Para localizar estas direcciones se debe introducir en el motor de búsqueda una o más palabras claves que identifiquen el tema que nos interesa.

¿Es posible el anonimato por Internet?

Entre este océano de datos que se manejan en Internet se perdían en el anonimato millones de usuarios en todo el mundo, acceder e intercambiar información por la red mundial era garantía de anonimato, y de hecho era uno de los atractivos de Internet y de los servicios que proporciona. Gracias al avance tecnológico en telecomunicaciones e Informática, se han desarrollado redes que han capturado más de dos mil millones de identidades y tienen perfectamente identificados los hábitos, los gustos, los consumos, los lugares de preferencia, los *hobbies*, entre otros aspectos de cada uno de sus usuarios, *además a esta información se le da seguimiento*. Actualmente Internet rompe la barrera de la privacidad, esta se rompió cuando Internet dejó de ser académico, y entró a todos los ámbitos, entre ellos el comercial. Esto se acentuó por los acontecimientos del 11 de Septiembre en Nueva York. El gobierno de Estados Unidos vio la necesidad de controlar una arquitectura desde la que sea posible identificar a un usuario sin importar su origen o los sitios a los que accede. La Sun lanzó desde mediados de 2002 productos Sun ONE, que dan a los usuarios la libertad de seleccionar un operador de identidad, y para mediados de 2003 se calcula que estará disponible la versión 1.2 de Liberty que tratará de proporcionar la "Federación de Identidad". En la actualidad se estima que hay doce mil millones de identidades en el mundo.

Para más detalles consultar la página 34 del periódico El Financiero del lunes 21 de abril de 2003.

1.2. Los orígenes de las Intranets.[1]

Al principio los Sistemas de Información que soportaban gigantescos volúmenes de datos que eran administrados a través de muy diversas aplicaciones de software por usuarios y administradores de red que tenían que aprender a manejar un buen número de paquetes y aplicaciones de software, además del trabajo que tenían que desempeñar en la organización o institución, esto representa mucha pérdida de tiempo y frustración pues cuando ya empezaban a familiarizarse con el uso de una o de otra aplicación sucede que ya había salido la nueva versión y tenían que volverse a capacitar y esto en el mejor de los casos por que a veces la aplicación ya se había vuelto obsoleta. Había usuarios que estaban tan familiarizados con una aplicación de software que difícilmente estaban dispuestos a abandonarlas por otra más eficiente. A principio de la década de los noventa Internet ya era de uso muy común por que era fácil de usar y maneja a diario una cantidad inmensa de información. Siendo esto así, la tecnología de Internet, de los Sistemas de Información y de redes LAN son los elementos fundamentales para crear la Innovación de la segunda mitad de los noventas que actualmente conocemos como Intranet. Las Intranets son redes LAN que soportan Sistemas de Información (Local Area Network, *Red de área Local*) que son gestionados comúnmente con tecnología de Internet. Las ventajas de usar este tipo de tecnología es que es de bajo costo, usar una Intranet es sumamente fácil y divertido, basta con que sepamos usar lo básico de la tecnología de Internet, y software de oficina. Los usuarios no necesitan

aprender tantas aplicaciones de software para manejar parte de un Sistema de Información. El correo electrónico es en la primera herramienta a implantar en la Intranet esta es la forma básica de intercambio de documentación y la más usual para comunicarse tanto en Internet como en la Intranet. Por medio de esta herramienta se pueden discutir temas por e-mail y todos pueden ver la discusión, y dar seguimiento. En la aplicación basta con pulsar Message, Reply All para enviar la respuesta a todos los que reciben la respuesta sobre un tema específico. De esta forma responderán cuando puedan, se puede tener un registro de toda la discusión guardando los mensajes en una carpeta o imprimiéndolos en papel. El resultado implica una toma de decisiones más rápida y a un menor costo. En una Intranet la comunicación entre todos los usuarios es más fluida, las ideas se intercambian, se discuten a nivel local y a niveles que trascienden al Sistema Intranet, esto se hace por lo general por medio de Servidores de grupos de noticias, para acceder a esta información es necesario contar con lectores de noticias. Un grupo de noticias es un conjunto de mensajes colocados en un servidor por una o varias personas, mantenido por una organización o los mismos usuarios, algunos grupos de noticias se supervisan, algunos no. Los mensajes pueden ser leídos por cualquier usuario que tenga acceso al grupo.

Siempre la discusión de las ideas ha sido la mejor forma de retroalimentar y mejorar lo que ya se sabe o bien generar conocimiento nuevo, compartir las nuevas inquietudes sobre innovaciones,… etc. Esto se realiza a través de *grupos de discusión*, en tales grupos los debates son casi similares a las discusiones que se daban en los diálogos de Platón donde se discutía sobre un tema, se profundizaba y los participantes aportaban ideas sobre el tema hasta llegar a una conclusión parcial. La información se vuelve más dinámica, las discusiones normalmente se presentan en diálogos de manera que se puede seguir no solo el mensaje original, sino todas las respuestas al mismo.

- ✓ **Participación activa dentro de la Intranet.**

Todos los usuarios participan activamente dentro de este sistema Intranet, y por lo tanto necesitamos documentar de una u otra forma las actividades que se llevan a cabo en el transcurso del día, esta documentación se realiza en la misma forma que en los Sistemas de Información, la diferencia es que todo se documenta en páginas Web. Esta participación puede ser del siguiente modo:

- ✓ Los departamentos pueden aportar ideas en los grupos de discusión internos.
- ✓ La facilidad de acceso a los datos Web permite espacio a la creatividad.
- ✓ Todos los usuarios pueden crear documentos Web relevantes para la organización mediante herramientas WYSIWYG (<<What You See Is What You Get>>, Lo que se ve es lo que se obtiene). [2]

La mayor de las veces se utiliza el Internet para buscar información, sin embargo también se pueden crear aplicaciones del tal forma que el Internet se haga interactivo. Por ejemplo se pueden crear aplicaciones en JAVA que se pueden ejecutar desde el navegador y realizar operaciones más o menos complejas, como el cálculo de amortizaciones o mejoras visuales en las páginas HTML.

- **El concepto de la Intranet**

El lenguaje HTML , Junto a los Scripts y el acceso a bases de datos no son únicamente posibilidades existentes en Internet, sino que pueden aplicarse también a las redes locales de las organizaciones creando un tipo de red de funcionamiento similar a Internet. Creando un tipo de red de funcionamiento similar a Internet, pero limitada al ámbito local. Este tipo de redes recibe el nombre de Intranets.[3]

Una Intranet es una red local organizada gracias al empleo de la tecnología Web. El lenguaje HTML se emplea para crear páginas Web que permiten compartir datos con los usuarios de la red. Una Intranet es un Sistema de Información con tecnología de Internet, en este sistema circula toda la información que se va generando en una organización a lo largo del día. La información que se va generando en una organización es basta y compleja por lo que es necesario que se distribuyan adecuadamente las responsabilidades para su manejo pues una única persona no puede dominar todos los temas técnicos y administrativos que involucran una Intranet.

El empleo de Intranets favorece el trabajo en equipo, y permite a los usuarios tanto compartir la información, como modificar sus contenidos, lo que implica una actualización permanente de los mismos, evitando recurrir a complejos sistemas informáticos de gestión de información. De esta forma los trabajadores pueden consultar y modificar las bases de datos de la organización, compartiendo de esta forma todos los datos que precisen para desarrollar sus tareas. Por ejemplo puede mantener una base de datos con los clientes mediante el empleo de la Intranet. Los directivos podrán acceder a los datos personales de los clientes y ponerse en contacto con ellos, no hay nada que impida que los clientes, mediante claves y teléfonos de acceso conectados a módems, accedan a la Intranet para consultar precios, hacer pedidos o incluso, para ponerse en contacto, con el servicio técnico ó comercial.

- **Ventajas de las Intranets.**

Facilitan el entendimiento y el empleo de los Sistemas de Información y su sinergia, por ejemplo:

- ✓ Los grupos de discusión, son un puente de comunicación importante para el análisis de los temas de importancia.
- ✓ Bajo costo y fáciles de usar.
- ✓ Se usan las propiedades de Internet para la búsqueda e interacción de datos e información en forma rápida y de fácil acceso.
- ✓ Ahorro en infraestructura de cómputo: Ahorros en hardware y software, reducción de mano de obra, reducciones de cuentas telefónicas, reducción de espacio físico lo que implica ahorro en costos de renta y servicios de oficinas.
- ✓ Ahorros en envíos.
- ✓ Reducciones en gastos de viaje: Videoconferencias, audio en línea.
- ✓ Entrenamiento interactivo. El aprendizaje puede acoplarse al ritmo individual. Y éste beneficio es especialmente atractivo, pues no sólo ahorran los costos del viaje, sino que el entrenamiento suele ser más eficaz.
- ✓ Los documentos necesarios están disponibles en línea.
- ✓ Mejoras es la coordinación y en el manejo de tiempos.
- ✓ Mejoras en soporte de ventas y servicios al cliente.
- ✓ Mejor retroalimentación para los ejecutivos
- ✓ No se necesita una profunda cultura informática para su uso.

- **Documentación que comúnmente circulan por las Intranets.**

  - ✓ Programas para la instalación de discos de todo el software aprobado.
  - ✓ Formación basada en las computadoras.
  - ✓ El manual de políticas corporativas.
  - ✓ Estado de la producción para todos los productos actuales y los nuevos.
  - ✓ Previsiones de ausencias vacacionales o por enfermedades.
  - ✓ Informaciones generales de la plantilla o de la empresa.
  - ✓ Oferta de nuevos puestos de trabajos.
  - ✓ Informes sobre la calidad en el trabajo.
  - ✓ Manuales de orientación a nuevos empleados.
  - ✓ Estados financieros de la empresa, especialmente si esta cotiza en la bolsa.
  - ✓ Boletines internos de la empresa o de los departamentos.
  - ✓ Información sobre fusiones empresariales
  - ✓ Inventarios.
  - ✓ Material de Marketing.
  - ✓ Políticas de precios.
  - ✓ Lideres de ventas.
  - ✓ Programación de proyectos y fechas clave.
  - ✓ Asignación de personal.
  - ✓ Programa de reuniones de equipo e informes posteriores.
  - ✓ Informes sobre objetivos.
  - ✓ Informe sobre problemas y su seguimiento.
  - ✓ Accesibilidad al servicio de ayuda.
  - ✓ Informes históricos de los servicios.
  - ✓ Y un largo etcétera.

Como hemos visto una Intranet es el mejor medio del que se pueden valer los Sistemas de Información para una mejor administración y distribución de la misma, ahorra tiempos, permite una mejor cooperación entre los empleados para una mejor sinergia del sistema. Pero las Intranets permiten algo que no se había visto en la historia de los sistemas de Información: No es difícil y costosa su actualización y adaptación a los cambios tan acelerados que son tan característicos de esta época**.**

- **¿Qué es una base de datos para la Web y por qué es tan necesaria?**

Las bases de datos son tan imprescindibles, abarcan tanto y encierran una complejidad que incluso hay especialidades y postgrados que se dedican en forma exclusiva a su estudio. Debido a la facilidad de manejo y gestión de enormes volúmenes de información las bases de datos son de enorme utilidad en la gestión de la información por Internet y las Intranets, y esto sucede así que escuchar bases de datos en la Web, es imprescindible asociarlas con las Intranets. Esto que estoy mencionando es ya tan común en la cultura informática y sin embargo es tan poco difundido en nuestros centros de estudio que incluso va haber un déficit de especialistas en Bases de Datos, y yo agregaría que <<Especialistas en Bases de Datos en la Web>>. Empresas y gobiernos en Estados Unidos van a perder muchos miles de millones de dólares por el déficit que habrá de especialistas en computación en esta área y en nuestro país que tiene una enorme dependencia tecnológica, se verá arrastrado por este problema si desde

este momento no toma cartas en el asunto y prepara especialistas exclusivamente en el estudio e investigación en esta área.

- **Ventajas de utilizar bases de datos en las Intranets.**

  - ✓ Ahorro en cursos de actualización continua.
  - ✓ El equipo que se utiliza es comparativamente más barato y fácil de usar.
  - ✓ La impresionante flexibilidad a los cambios tecnológicos.
  - ✓ Mayor satisfacción laboral de los empleados, lo que reduce su rotación.
  - ✓ Mayor productividad en el personal de desarrollo.
  - ✓ La posibilidad de interactividad con más clientes potenciales a un costo menor por cliente que por otros medios.
  - ✓ La capacidad de ofrecer productos y servicios globalmente sin tener que soportar el costo de establecer representaciones por ultramar.
  - ✓ La oportunidad ampliada de desarrollar sistemas que cumplan mejor las necesidades de la compañía debido a la naturaleza de rapidez inherente a las herramientas de HTML y GUI.
  - ✓ Facilidad de mantenimiento de las bases de datos debido a la sencilla integración de los programas HTML y GUI.

- **Ventajas para los usuarios de bases de datos en Web.**

  - ✓ **Usuarios internos.**
    - Acceso de la interfaces gráficas de usuario a los datos de la Intranet.
    - Acceso a la información interna actualizada.
    - Mayores oportunidades de interacción con otros usuarios, departamentos y tecnologías dentro de la compañía.
    - Habilidad para personalizar los buscadores y así cumplir con las necesidades específicas.
    - Integración de la aplicación de las bases de datos con otras aplicaciones que funcionan en las computadoras.
  - ✓ **Usuarios externos.**
    - Una facilidad impresionante para acceder a cualquier hora del día a una inscripción semestral, trámite escolar, trámites de titulación, una suscripción a una revista, posibilidad de interacción con los datos de una compañía.
    - Acceso al sistema desde la comodidad del hogar o desde el puesto del trabajo.
    - Posibilidad con los datos del sistema sin necesidad de adquirir equipo o software costoso.

Capítulo II. Los elementos de una Intranet.

Si se dispone de una red LAN, podemos implementar nuestra Intranet si le agregamos además del servidor de datos, un servidor Web y software que se conoce comúnmente como kit de recursos para Intranet que contiene lo necesario para crear nuestro sistema en muy poco tiempo, y en ocasiones en un día. Visto así una Intranet básica quedaría del siguiente modo:

Red LAN + Sistema de Información + Tecnología de Internet = Intranet

Las redes LAN son las que más se utilizan para el implementar las Intranets, sin embargo, el surgimiento de las Intranets se debió gracias a la concepción de la informática distribuida que surgió en el mundo de la empresa en la década de los ochentas. Uno de los avances en este campo, la llegada de lo que se denominan Redes de Área Extensa o WAN, ha hecho de las Intranets un fenómeno posible en la realidad actual.

Cuando hablamos de informática distribuida, no referimos a las redes de computadoras (Las más conocidas son posiblemente las redes LAN), que han sustituido en parte a las famosas *mainframes* corporativas.

A la hora de construir cualquier red de computadoras, y en especial de una Intranet necesitaremos de los siguientes elementos básicos:

2.1. Hardware de Intranet:

- ✓ PCs o estaciones de trabajo que actúan como servidores Web bajo el modelo Cliente-Servidor.
- ✓ PCs de los trabajadores, que actúan como clientes y sus correspondientes periféricos.
- ✓ Un sistema de cableado que interconecta el servidor o los servidores con los equipos Cliente(cable coaxial, de par trenzado, o fibra óptica)
- ✓ Elementos de hardware que configuran el concepto tradicional de red: tarjetas de conexión o NIC(*Network Interface Card*), repetidores, concentradores o hubs, etc…
- ✓ Máquinas que actúan como firewalls (Cortafuegos), y su correspondiente software.

2.2. Software de Intranet:

- ✓ Un Sistema operativo de red que soporta el intercambio de información y, que, como tal, reside tanto en clientes como en servidores. Hoy en día, existen varios Sistemas operativos disponibles en el mercado: Unix, Linux, Windows NT, Novell Netware, y otros.
- ✓ Aplicaciones de red, que en este caso, se refiere a la utilización de browsers, residentes en los equipos servidor y clientes, así como de programas específicos de correo electrónico, FTP, etc.
- ✓ Un Sistema de Gestión de Red, que permite el control de prestaciones, problemas, seguridad o configuración.
- ✓ Protocolos de comunicación.

2.3 . Modelo Cliente-Servidor:

La tecnología cliente servidor se utiliza para todas las aplicaciones de Internet/Intranet.

- ✓ Un servidor es una computadora remota, en algún lugar de la red que proporciona información según petición.
- ✓ Un cliente funciona en su computadora local, se comunica con el servidor remoto, y pide a éste información.
- ✓ El servidor envía la información solicitada.

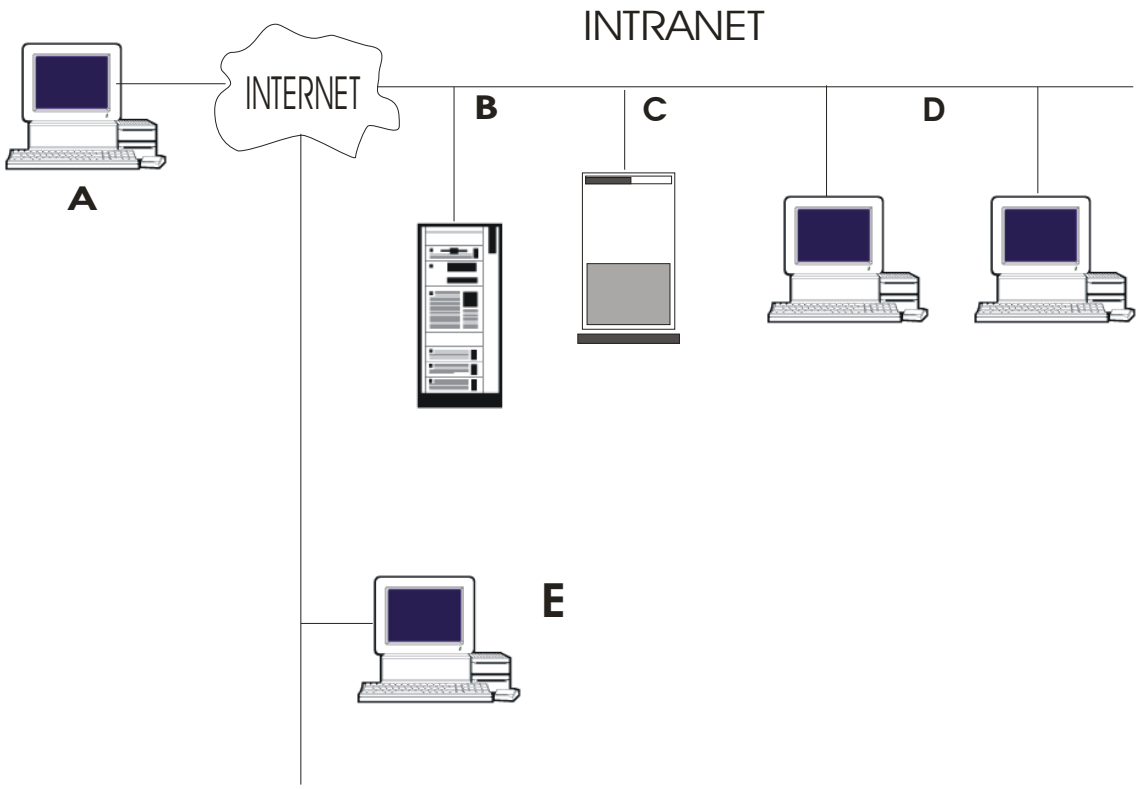
Los sistemas Cliente-Servidor pueden ser de muchos tipos, dependiendo de las aplicaciones que el servidor pone a disposición de los clientes. Entre otros, existen:

- ✓ Servidores de Impresión, mediante el cuál los usuarios comparten impresoras.
- ✓ Servidores de Archivos, con el cuál los clientes comparten discos duros.
- ✓ Servidores de Bases de Datos donde existe una única base de datos.
- ✓ Servidores de Sistema de nombres de dominio (DNS), hospedan la base de datos de todas las direcciones IP.
- ✓ Servidores Web, servidor donde se almacenan los documentos Web.
- ✓ Servidores Proxy.
- ✓ Servidores de correo electrónico.
- ✓ Servidor Web:

Los servidores Web son aquellos que permiten a los clientes compartir datos, documentos y multimedia en formato Web. Además de las ventajas que se obtienen al utilizar la tecnología Cliente-Servidor, el servidor Web aporta unas ventajas adicionales en aspectos importantes tales como:

- ✓ El Web se crea normalmente como un sistema abierto al que cualquiera puede contribuir y acceder desde cualquier punto de la red de la empresa: No se requieren logias ni passwords como los Sistemas tradicionales Cliente-Servidor.
- ✓ Los servidores Web vuelcan la información con un simple clic con el ratón. A través de un proceso de hiperenlaces.
- ✓ La información servida puede ser de cualquier tipo (datos, documentos, multimedia, etc,..), gracias a la utilización de los estándares de Internet. Esta información es de solo lectura pues, a diferencia de los sistemas normales Cliente-Servidor, el usuario no puede hacer cambios en el dispositivo original de los datos.
- ✓ Dado que el servidor de Web es de tan fácil acceso, ello hace posible publicar información de forma instantánea en toda la compañía mediante un simple almacenamiento de la misma en el servidor.
- ✓ Un servidor Web en la Intranet puede servir la misma copia de un archivo a toda la organización, de la misma forma que un único servidor Web en Internet puede servir al mundo entero. Así que ahora solo hay una única copia de archivo a actualizar, y cuando lo actualizas, la nueva versión es servida instantáneamente a toda la compañía.
- ✓ La amplitud de la red suele ser mayor que otros sistemas Cliente-Servidor

- ✓ Un Servidor Web en una Intranet



A) Por medio de un visualizador Web un usuario puede acceder al servidor Web. Puede ver las páginas Web, acceder a alguna aplicación. Puede querer permitir que cualquiera en Internet acceda al Web o impedir completamente el acceso a Internet.

B) El Servidor Web, puede tratarse de un servidor Web en Windows o en UNIX, o un Macintosh, una gran computadora con OS/2.

C) En una de las computadoras de la compañía puede tener, por ejemplo, una base de datos que quiere actualizar. Las personas que acceden al Servidor Web rellenan un formulario en un visualizador para proporcionar la información a la base de datos.

D) Los empleados en la Intranet pueden acceder al servidor Web para ver documentos y descargar archivos o ejecutar aplicaciones de respaldo. También podrían realizar tareas de administración si el software se lo permitiese.

E) Los empleados, desde cualquier punto de la Intranet, pueden acceder al servidor Web si tienen un acceso a la red. Pueden utilizar un visualizador para acceder a las páginas del servidor Web si tienen acceso a la red. Pueden utilizar un visualizador para acceder a las páginas del servidor o pueden utilizar las herramientas de administración remota del servidor Web, si el software dispone de esta posibilidad.

- ✓ **Servidor de Sistema de nombres de dominio(DNS)**

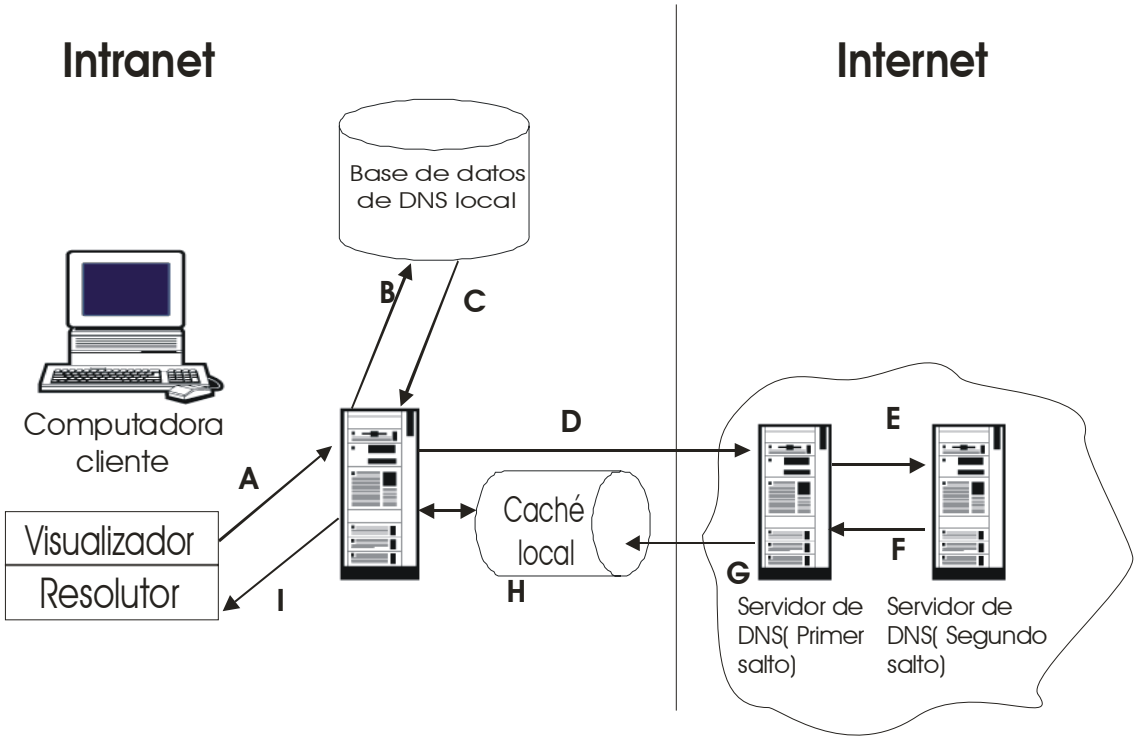
Los Servidores de Sistemas de Nombres de Dominio, son un tipo de Servidores que almacenan gigantescas bases de datos cuyos registros son direcciones IP de cada host y

su alias y la información del servidor de correo electrónico. El esquema utilizado en Internet o en una Intranet se encuentra en una base de datos distribuida llamada Sistema de Nombres de Dominio. En Estados Unidos la base de datos de dominios la administra interNIC, una organización responsable del registro de nombres de dominio que ofrece servicios.
La base de datos completa se distribuye, de forma que pequeñas partes de la misma puedan gestionarse en pequeñas partes de la misma puedan gestionarse en redes individuales por todo el mundo.

- ✓ **Uso de un Servidor DNS en una Intranet propia**.

El servidor DNS para Intranet hace las mismas funciones que un servidor DNS para Internet, la diferencia es que ahora lo hará internamente dentro del área de la red local. En una Intranet los servidores DNS reciben peticiones de las máquinas clientes del sistema, las respuestas que da a dichas peticiones son direcciones IP. *Los resolutor* son aplicaciones cliente que envían peticiones de direcciones IP a los servidores DNS. En muchas de las implementaciones de TCP/IP (Protocolo de control de transmisión / Protocolo de Internet), entre las que se encuentran las de Windows, el resolutor se encuentra integrado en la pila TCP/IP y el usuario no puede verlo.

- ✓ **Funcionamiento de un servidor DNS**.



A) El visualizador consulta a su resolutor interno. El resolutor envía una solicitud al servidor DNS de la misma red. La configuración de TCP/IP del cliente contiene la dirección de IP de este servidor DNS.

B) El servidor DNS busca en su base de datos local la dirección. También puede buscar este nombre en su memoria caché local.

C) Si se encuentra el registro en la base de datos se lo devuelve al cliente. El registro contiene la dirección de IP con la que el visualizador puede conseguir la página inicial.

D) Si el registro no se encuentra en el servidor de DNS local y está configurado como reenviador, o como computadora que traslada la solicitud a otra computadora, simplemente envía la solicitud a otro servidor de DNS y espera una respuesta.

E) Esta computadora (segundo salto) puede trasladar la solicitud de nuevo, hasta que se resuelve el nombre.

F) Cuando se encuentra el registro el segundo salto (o el salto en que se resuelve el nombre) envía la información hacía atrás al servidor de DNS anterior.

G) El primer servidor de DNS envía el nombre hacía atrás al servidor de DNS de la Intranet.

H) Se añade el registro a la caché local del servidor de DNS de la forma que no necesite solicitarla a otros la próxima vez que se recibe la misma solicitud de una computadora cliente.

I) El cliente recibe la respuesta. Algunos resolutor también usan una caché local de respuesta de manera que no necesitan repetir el proceso de búsqueda. El resolutor devuelve la dirección de IP al visualizador consigue la información que necesita para obtener la página principal de la compañía.

Otras opciones

Para una Intranet sin conexión a Internet, hay dos posibles soluciones para resolver el problema de las direcciones:

- ✓ Con una tabla de host. Todas las direcciones IP de las computadoras de una Intranet se organizan en una tabla de host. Sin embargo, cualquier cambio en los dominios necesitan actualizar todas las tablas de todas las computadoras.
- ✓ Con un servidor de DNS. El mantenimiento se simplifica manteniendo una única base de datos en un servidor DNS.

Servidor de correo electrónico.

El servidor de correo actúa como un cartero electrónico que envía y recibe correspondencia, debería de disponer de opciones flexibles de seguridad y posibilidades de administración remota. Las distintas aplicaciones varían pero deberían permitir el acceso directo basado en el nombre del dominio o del host a una computadora especificada. Una característica crítica es la multitarea que permite a un servidor atender a múltiples conexiones entrantes. Si el servidor de correo tiene muchas peticiones, se necesita que varias personas puedan conectarse al servidor. Dependiendo del número de usuarios a los que se dé servicio, puede elegirse entre dedicar una máquina para el correo o ejecutar el servidor de correo en la misma computadora que el servidor Web.

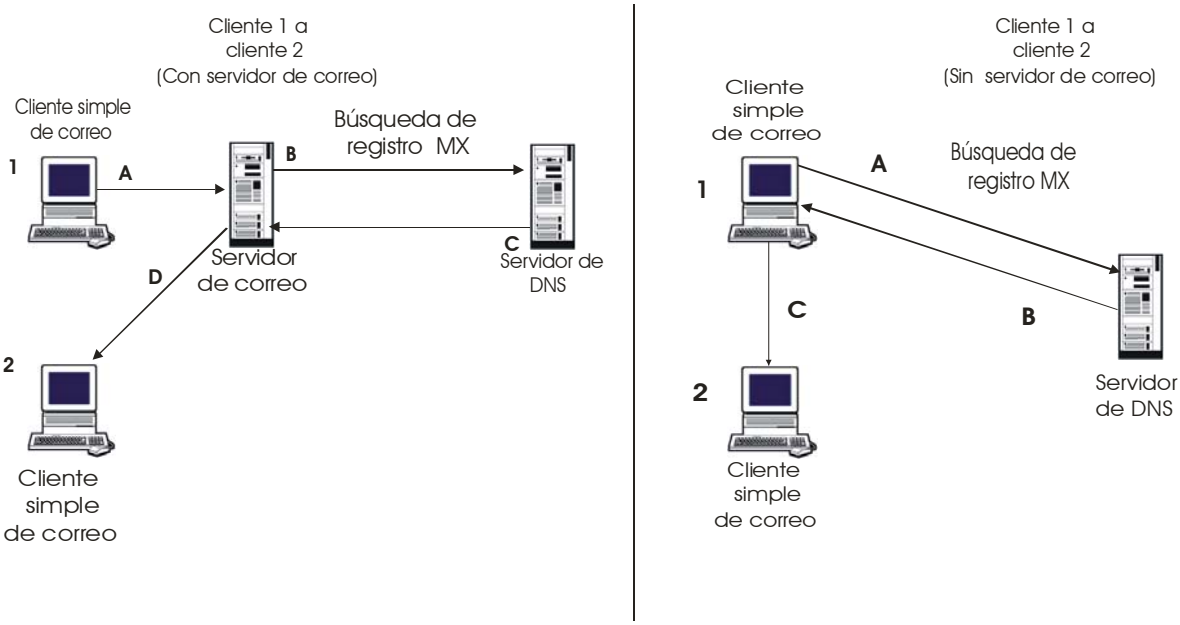
Protocolos más populares de envío de mensajes.

- ✓ **SMTP:** Protocolo de transferencia de mensajes simple. SMTP almacena los mensajes en la computadora del usuario no en el servidor remoto. Por lo tanto si se desconecta la PC de la red TCP/IP no se pueden entregar los mensajes en ese intervalo de tiempo.

- ✓ **POP3:** Protocolo de oficina de correos 3. Permite que los mensajes se almacenen en un servidor de red, de forma que los usuarios puedan descargar los mensajes del servidor cuando abran la aplicación de correo.

- ✓ **IMAP4:** Protocolo de acceso a mensajes versión 4. Permite que los mensajes queden en el servidor de forma que los clientes simplemente consultan los mensajes que se encuentren en el servidor.

- ✓ Envío de mensajes a clientes de SMTP.



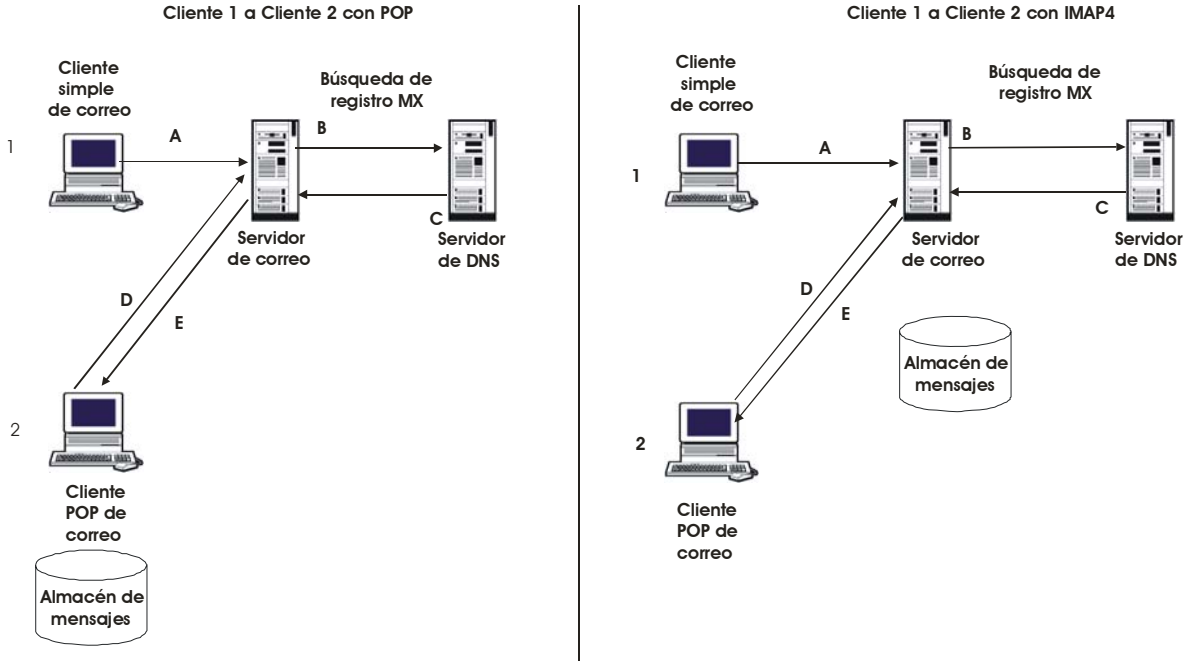
Con servidor de correo.

- **A)** Para enviar un mensaje la computadora cliente de SMTP envía el mensaje. De camino a su destinatario el mensaje se encuentra primero con el servidor de correo de la Intranet.
- **B)** El servidor de correo realiza una petición de búsqueda al servidor DNS que realice ***una búsqueda de intercambio de mensajes (MX),*** que traduce la dirección de correo electrónico de la persona que se envía a un nombre de máquina. El servidor DNS encuentra el nombre de máquina.
- **C)** Cuando el servidor DNS ha encontrado la dirección envía el nombre de la máquina de vuelta al servidor de correo.
- **D)** El servidor de correo envía el mensaje al nombre de máquina correcto.

Sin servidor de correo.

- **A)** El mensaje es enviado por el cliente de **SMTP**. En paralelo la computadora envía una solicitud al servidor de DNS de la Intranet para que se realice una búsqueda MX, para encontrar el nombre de la máquina de destino.
- **B)** Cuando el servidor DNS encuentra el nombre de la máquina (la puede haber encontrado en su propia base de datos o mediante una búsqueda en las bases de datos de otros servidores de DNS) envía el nombre de la máquina a la computadora cliente.
- **C)** El cliente de **SMTP** envía directamente el mensaje a otro cliente de correo de SMTP, que puede encontrar en la Intranet o en cualquier otro lugar de Internet.

- ✓ **Envío de mensajes a clientes de POP3 e IMAP4.**

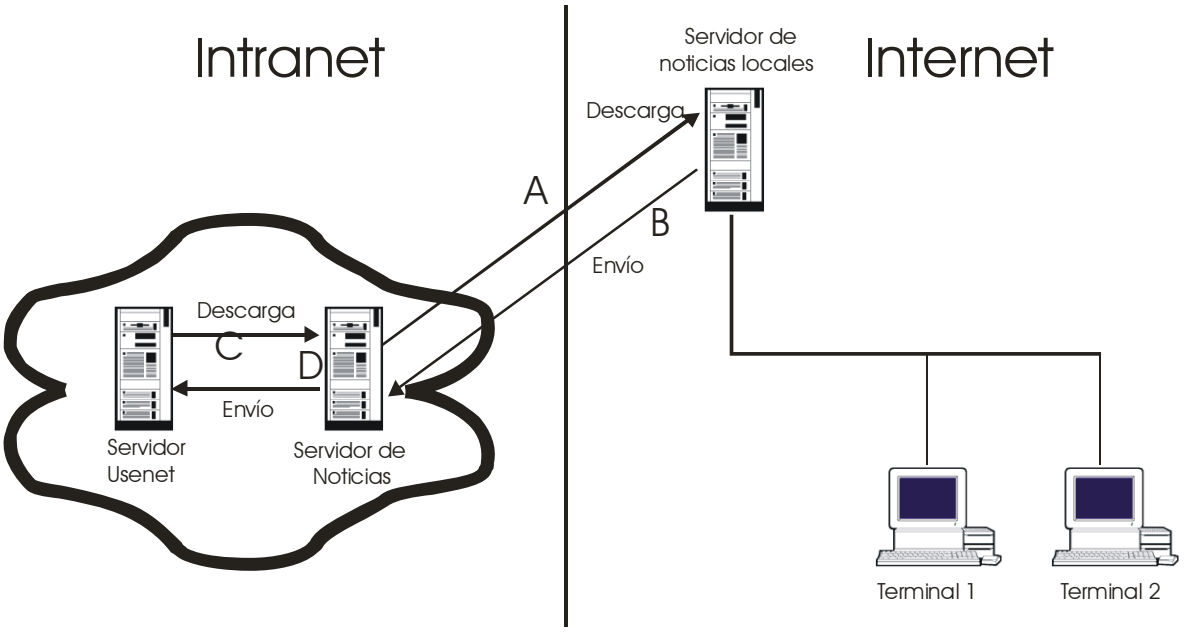


A) La computadora cliente envía un mensaje por correo electrónico a un servidor de la Intranet.
B) El servidor de correo pide al servidor DNS que se realice una búsqueda MX.
C) El servidor DNS envía el nombre de la máquina al servidor de correo. Si el destinatario se encuentra fuera de la LAN se envía el mensaje al servidor de correo del destinatario.
D) El cliente de POP o de IMAP4 piden mensajes al servidor de correo.
E) Por último se envía el mensaje al cliente. Si se usa una POP como se aprecia en la parte izquierda, el mensaje se copia físicamente en la computadora del destinatario. Si se usa IMAP4 el mensaje permanece en el servidor de correo.

Servidor de noticias (Foros de discusión)

Un servidor de noticias es un servidor de Internet, casi siempre bajo el nombre de **Usenet**. Su función es distribuir boletines, noticias sobre temas de interés, las noticias se dividen a su vez en categorías, es decir en grupos de noticias donde se analizan y discuten temas tales como las ventajas y desventajas de utilizar LINUX versus Windows. El protocolo que usan los servidores de noticias es el Protocolo de Transferencia de Noticias en Red (NNTP). Los grupos de noticias son muy populares por que son un excelente mecanismo para compartir ideas. En una Intranet los grupos de noticias facilitan la comunicación entre grupos independientemente de su localización. En la actualidad existen varias decenas de miles de usuarios en debate.

En la actualidad como una alternativa a los grupos de noticias, se han desarrollado los foros de discusión basados en páginas Web. En la actualidad todos los portales de Internet incluyen sus propios foros clasificados por temas para fomentar la participación de sus usuarios y la creación de comunidades virtuales en torno a los servicios del portal.

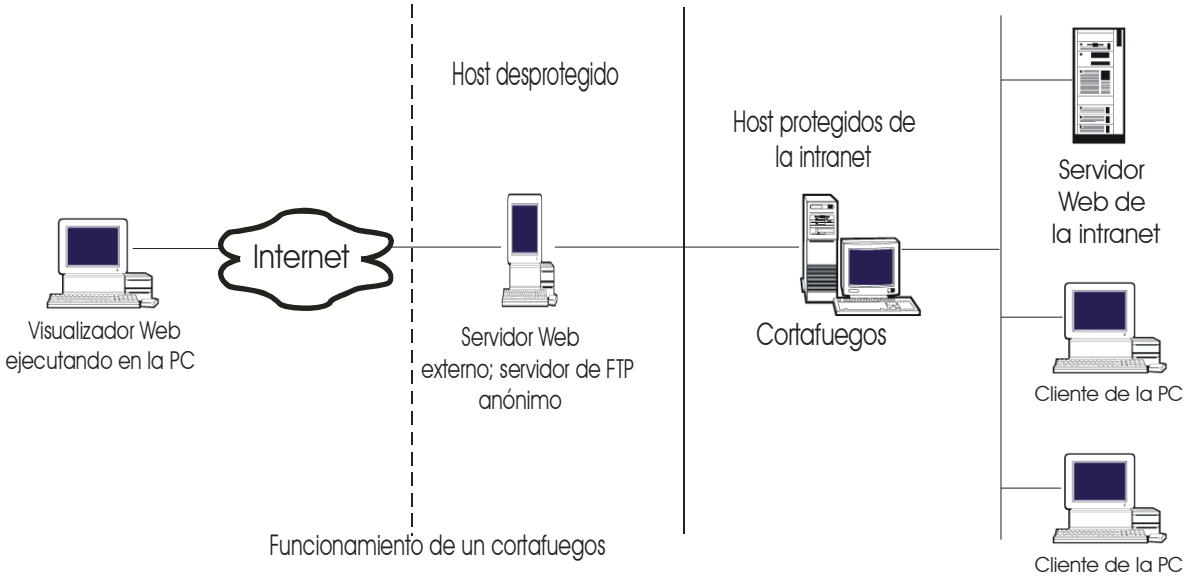


Funcionamiento de un servidor de noticias

A) El servidor de noticias de Intranet realiza una petición de descarga al servidor de noticias de Internet.
B) El servidor de noticias de Internet envía mensajes al servidor de Internet.
C) Un servidor de Usenet descarga nuevas noticias al servidor de noticias de la Intranet.
D) Uno de los servidores principales de Usenet captura un mensaje de un servidor de noticias.

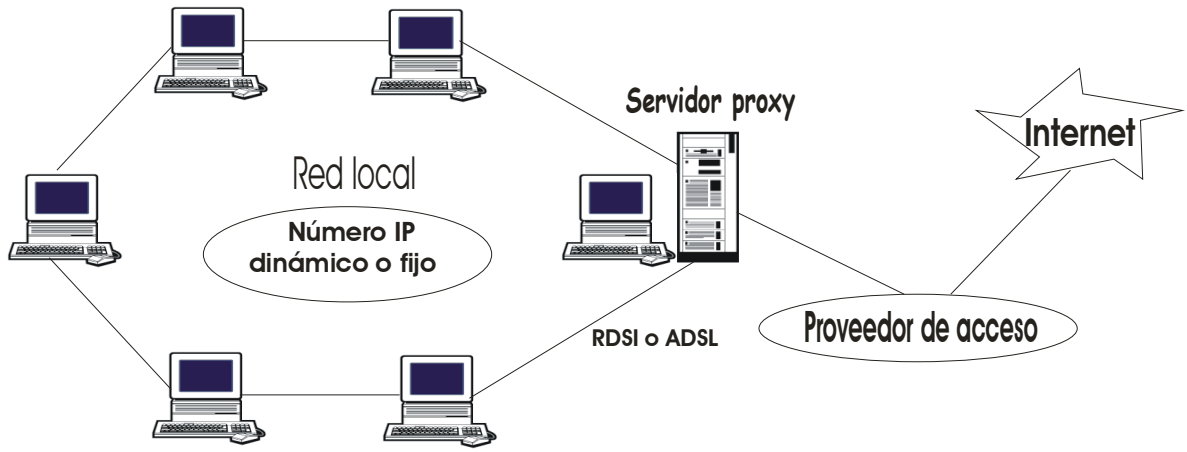
- ✓ **Software cortafuegos**

Es un conjunto de aplicaciones de software y hardware que se utilizan para estrategias de defensa perimetral dentro de una red, que consiste en auditar los intentos de conexión hacía la red y los intentos de conexión hacía el exterior de la misma. Permitiendo realizar este tipo de operaciones a personas autorizadas por los administradores de la red.



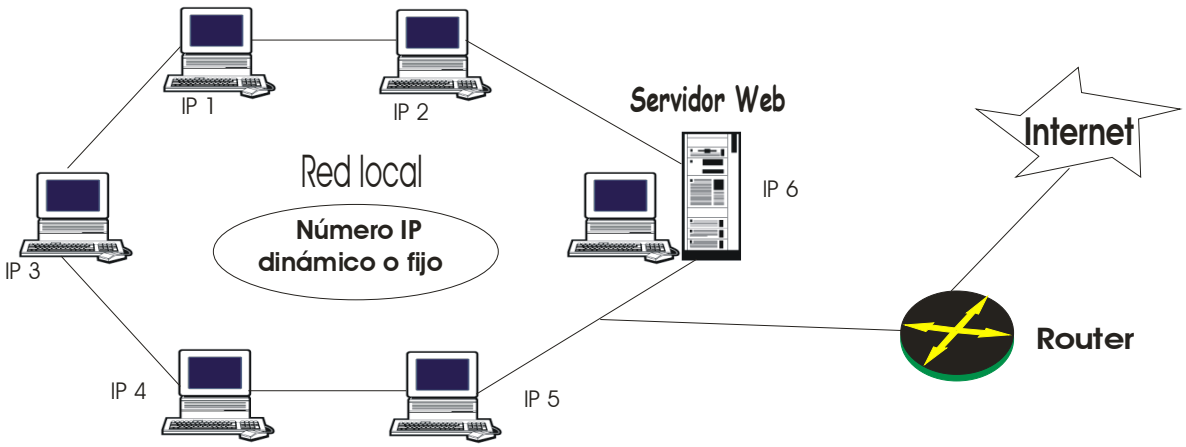
Funcionamiento de un cortafuegos

Servidor Proxy: Es un servidor que controla el acceso y salida de direcciones IP. Este tipo de servidores se usan como cortafuegos para controlar el uso de recursos de la red y limitar el acceso a Internet. Es el administrador de la red se encarga de controlar el servidor Proxy. De este modo, todas las conexiones pasan por un único equipo que se encarga de su supervisión y control, proporcionando además mayor seguridad a la red de la empresa frente a intentos de acceso desde el exterior. La seguridad de la conexión se podría mejorar empleando unos cortafuegos (firewall), dispositivo encargado de supervisar y limitar el tipo de conexiones permitidas en la red interna de la organización e Internet.



Conexión compartida a través de un servidor Proxy

La conexión puede organizarse de tal forma que cada terminal de la red tenga su propia dirección IP fija en Internet, por lo que en este caso se hace imprescindible el uso de routers para realizar la conexión de la red local con Internet.



Conexión directa a través de un router

Capítulo III. Los Sistemas de Información.

3.1. Sistema de Información.

Un Sistema de Información es la agrupación coherente y sistematizada de las diferentes clases de información que *se generan en una organización en particular*. La información nace de las decisiones, de los actos y de los hechos. El Sistema de Información debe ser un medio para obtener la información que sea relevante de entre el cúmulo de datos, informaciones, etc., que sabemos se producen día a día, tanto en el medio ambiente como en la propia empresa. De la dirección emana la información que da origen al desarrollo de la actividad. De aquí que los elementos del Sistema de Información deben estar perfectamente definidos.

3.2. Componentes de un Sistema de Información.

Un Sistema de Información consiste en un conjunto de componentes que generan una clase de información claramente definida de tal forma que sea funcional y permita la claridad en el desarrollo de las actividades una organización.

Con estos componentes deben quedar cubiertas todas las necesidades de información que permitan la viabilidad de la organización. Entre ellos podemos distinguir:

a) **La Información emanada de la Dirección:** La Dirección tiene la Información relevante de los componentes del Sistema que le permite tomar las decisiones que definen el rumbo de la organización. Una sinergia de todo el Sistema organizacional con la realidad, política, económica y social.
b) El conjunto de planes, objetivos, etc.: Forman la base de la actividad.
c) **La información de relación:** La información que relaciona a las personas de la organización entre sí y con el medio ambiente.
d) **Los resultados:** El conjunto de resultados del cumplimiento de la actividad.
e) **De comparación:** La información que se genera al comparar los planes objetivos, etc., con los resultados. Es decir, b) y d).
f) **La síntesis:** La síntesis de la Información de comparación a fin de que sea conocida por la dirección.
g) **Información de Investigación:** La información que resulta al aplicar modernas técnicas matemáticas con la computadora para la planificación y la prospectiva.

Estos incisos forman un componente del Sistema de Información. **Ver fig1**. Los cuáles en el mismo orden son:

- ✓ Información normativa
- ✓ Información de planificación.
- ✓ Información de relación.
- ✓ Información operacional.
- ✓ Información de control y de gestión.
- ✓ Información Integrada.
- ✓ Información de investigación.

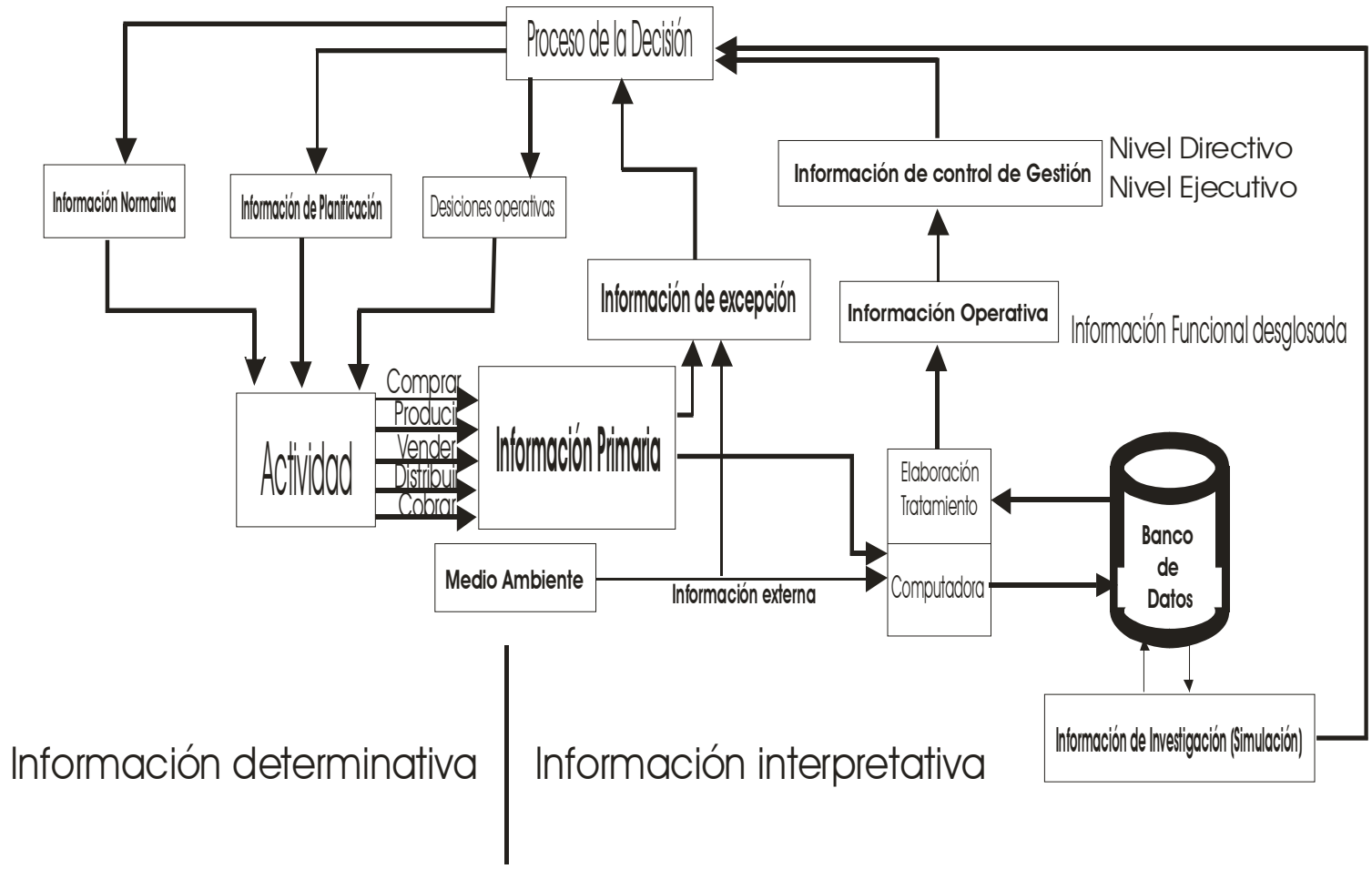


Fig1. Esquema del Sistema de Información

3.3. Desarrollo de una aplicación para la Web.

Entre las ventajas de utilizar una aplicación de un entorno de bases de datos en la Web se encuentran:

- ✓ Aumenta la posibilidad de compartir datos.
- ✓ Integra servicios.
- ✓ Comparte los resultados.
- ✓ Hace que los datos sean intercambiables.
- ✓ Enmascara el acceso físico a los datos.
- ✓ Independiza la posición de los datos y su procesado.
- ✓ Amplía los recursos de puestos de trabajo.

3.4. Características de una aplicación de bases de datos para la Web.

- ✓ **Servicio.** Ofrecen unos servicios que incluyen la aceptación de entradas, el procesado de los datos y finalmente, la exhibición en pantalla de la información resultante.
- ✓ **Recursos compartidos.** Comparten por lo menos algunos de los recursos en los que se ejecutan los componentes; puede que la manera más notoria de los componentes sea la propia base de datos.
- ✓ **Independencia de las plataformas**. Una característica muy importante de la Web es la independencia de los sistemas operativos y de las plataformas.
- ✓ **Interfaz basada en el mensaje**. El papel del servidor, es el de traducir los mensajes recibidos desde un buscador de Web en algo que sea inteligible y manejable.
- ✓ **Transparencia del servidor**. La garantía de una aplicación para la Web es que se la puede construir y desplegar para los usuarios sin develar donde residen físicamente los datos. ¿Podríamos saber cuál es el nombre del servidor que está asociado físicamente con la URL http://microsoft.com? Esto es la transparencia del servidor.
- ✓ **Ampliación**. Una aplicación de bases de datos para la Web es que es escalable, lo que significa que el proceso de añadir más espacio para mayores bases de datos es muy sencillo, o el de añadir más módems para incrementar el número de visitantes de la página Web.
- ✓ **Responsabilidades separadas**. En estas aplicaciones quedan claramente separadas de las funciones de la aplicación realizadas dentro del buscador, de las ejecutadas en el servidor de http o en el servidor de la base de datos.
- ✓ **Relación de muchos a uno.** Se puede considerar que hay muchos más buscadores de Web accediendo a la aplicación, vía el servidor de http, que servidores disponibles.

3.5. Claves para Implantar con éxito un sistema.

- ✓ **Compromiso de los ejecutivos**. Consiste en ofrecer las aplicaciones de bases de datos en Web para una mejor coordinación.
- ✓ **Integración de las herramientas.** En muchos productos de software ya vienen integradas las herramientas de construcción de una aplicación de bases de datos en Web. Esto queda evidenciado por la frecuencia con la que llegan al mercado nuevas herramientas integradas.
- ✓ **Soporte del entorno y de la arquitectura.** La planificación de una aplicación, tanto para bases de datos en la Web, como para cliente/servidor, debe incluir una seria consideración sobre la cantidad de soporte que requiere.
- ✓ **Formación.** Es común que en los altos niveles jerárquicos y por los usuarios generales se subestime la cantidad de entrenamiento que se necesita para construir una aplicación cliente/servidor.

El proceso.

El en el diagrama siguiente se exponen los pasos necesarios para desarrollar una aplicación de base de datos para la Web. Ver Fig2.

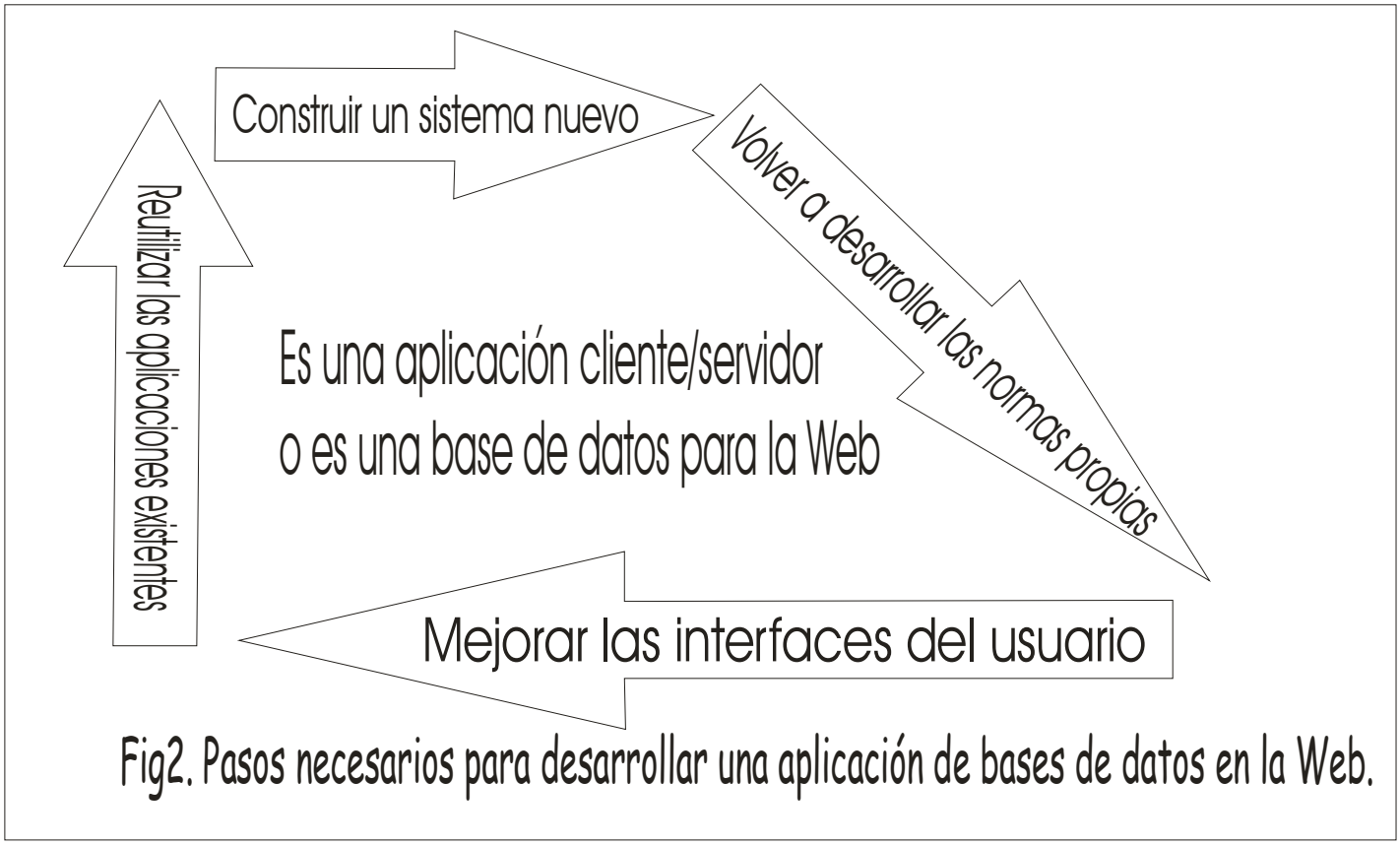


Fig2. Pasos necesarios para desarrollar una aplicación de bases de datos en la Web.

Pasos para desarrollar una aplicación de bases de datos en la Web.

- ✓ **Paso n:** Construir un sistema nuevo utilizando las ventajas de lo aprendido y de la información disponible de sistemas anteriores.
- ✓ **Paso n+1**: Reutilizar las aplicaciones existentes tanto como sea posible.
- ✓ **Paso n+2:** Mejorar las interfaces del usuario en el nuevo sistema diseñado**.**
- ✓ **Paso n+3:** Volver a desarrollar las propias normas para soportar el proceso de desarrollo.

- ✓ Peticiones para las herramientas de programación.
- ✓
- ✓ Mantener la funcionalidad en todos los sistemas operativos.
- ✓ Soportar el desarrollo en equipo.
- ✓ Un depurador altamente interactivo.
- ✓ Buenas posibilidades de orientación a objetos.
- ✓ Documentación de fácil entendimiento.
- ✓ Manejabilidad.
- ✓ Sistema de ayuda muy intuitivo y sensible.
- ✓ Compromiso del fabricante para futuras aplicaciones.
- ✓ Soporte de GUI populares**.**

- ✓ Soporte de las especificaciones SQL.
- ✓ Soporte de bibliotecas de clase.
- ✓ Integración con otras herramientas cuando y como sea necesario.
- ✓ Estabilidad financiera del fabricante.

Componentes de una aplicación de bases de datos para la Web.

- ✓ Servidor
- ✓ RDBMS.
- ✓ Servidor de aplicación.
- ✓ Cliente de la Web.
- ✓ Programa CGI.

El Servidor. Se refiere al servidor Web que puede leer una petición de información que le llegue desde un buscador, normalmente en forma de URL, puede localizar la página requerida y enviarla de nuevo al buscador.

RDBMS. Un Relational DataBase Management System (Sistema relacional de gestión de bases de datos). Es un conjunto de aplicaciones de software que almacena y presenta los datos en forma de tabla y permite la manipulación de los datos almacenados en paquetes, como alternativa a los registros individuales.

Servidor de aplicación. El servidor de aplicación es el responsable de mantener una conexión abierta entre el servidor y el RDBMS en cualquier ocasión.

Cliente de la Web. Es el software que funciona en una máquina cliente y realiza las funciones de comunicación necesarias. Las funciones principales realizadas por un cliente de la Web son:

- ✓ Establecer y mantener las comunicaciones con el servidor.
- ✓ Pasar las peticiones del usuario al servidor.
- ✓ Mostrar la información recibida desde el servidor.
- ✓ Visualizar los archivos no provenientes del servidor.

Los paquetes de clientes de la Web más populares son:

- ✓ Netscape Navigator.
- ✓ Microsoft Internet Explorer.
- ✓ Mosaic.

El programa CGI. Es un componente opcional y está pensado fundamentalmente para interactuar con el servidor utilizando varias normas diferentes. Su método principal de interacción en una aplicación de base de datos para la Web es conectar el servidor con programas externos.

Para el desarrollo de Sistemas de Información con tecnología de Internet hay muchas herramientas de software, desde las comerciales tales como las de Microsoft o bien el software libre por medio de plataformas en Linux.

Ciclo de vida de un Sistema de Información.

1.- **Estudio de Viabilidad:** Hacer un análisis costo-beneficio, económico, técnico y operativo.

2.- **Análisis:** Se recolecta la información detallada sobre las necesidades reales de los usuarios del sistema de Información actual. Para definir los problemas y necesidades del sistema.

3.- **Diseño:** Se realiza un diseño del sistema de la base de datos y de los Sistemas de aplicación o programas para manipular la base de datos.

4.- **Implementación:** Se implanta el Sistema de Información y se carga la base de datos así como las transacciones de esta para probarse.

5.- **Validación:** Se revisa que el Sistema sea aceptado por los usuarios.

6.- **Operación:** Se capacita a los usuarios para el uso del Sistema de Información y el mantenimiento correspondiente.

Ciclo de vida de una base de datos.

1.- Definir el Sistema: Aquí se define el alcance del Sistema de base de datos, sus aplicaciones y usuarios.

2.- **Diseño:** Se realiza el diseño físico y lógico del Sistema de base de datos en el sistema gestor de base de datos (SGBD).

3.- **Implementación:** Consiste en crear archivos de base de datos vacíos e implementar las aplicaciones de software.

4.- **Cargar datos:** Se introducen los datos a la base de datos directamente o a través de convertir la información de un archivo existente a la estructura que se tiene en la base de datos.

5.- **Conversión de aplicación:** Todas las aplicaciones de software que se utiliza con el sistema anterior se adecuan al nuevo sistema, en caso de que exista.

6.- **Prueba:** Se pone a prueba el Sistema para corregirlo, si es necesario o validarlo si es funcional.

7.- **Operación:** Implantar el sistema y ponerlo en uso.

8.- **Mantenimiento:** Se mantiene el sistema nuevo en observación para que se modifique y/o actualice a partir de surgir nuevos requerimientos del usuario.

Seis fases para realizar un buen diseño de la base de datos.

1.- Recolección y análisis de requerimientos.

- ✓ Identificar las áreas que utilizan el Sistema y seleccionar según su desarrollo, a un individuo que tenga los elementos necesarios para explicar la función del área dentro del sistema de información actual.
- ✓ Recabar toda la documentación existente para realizar el procesamiento de la información del sistema actual. Por documentación entendemos manuales, políticas, formatos, códigos fuente, entre otros.
- ✓ Realizar un diagrama de flujo de datos para identificar los datos que se requieren durante el ciclo de vida de la base de datos.
- ✓ Identificar los niveles de acceso o restricciones de la información para los usuarios que utilizarán el sistema de información.

2.- Diseño conceptual de la base de datos

El diseño conceptual de la base de datos debe ser independiente de cualquier SGBD.

- ✓ Expresivo, para distinguir los tipos de datos, restricciones y relaciones.
- ✓ Sencillo, ya que cualquier usuario lo puede entender, independientemente de tener o no elementos de diseño de base de datos.
- ✓ Diagramático, para que de forma visual se pueda interpretar el esquema conceptual del diseño de la base de datos. (Utilizando el diagrama de Entidad-Relación).
- ✓ Formal, ya que el modelo del diseño conceptual no debe de ser ambiguo, sino claro y confiable.

Modelo de datos.

- ✓ **Modelo de alto nivel o conceptual:** Los usuarios no expertos en el área de informática podrían interpretar o leer este tipo de modelo, ya que el conjunto de elementos que presenta (entidades, atributos y relaciones) son muy significativas para cualquier persona.
  Entenderemos por entidad a la representación real de un objeto o cosa, como por ejemplo una persona, un documento, un perro, etc., A un atributo como una propiedad o característica importante que da una información más completa de una entidad y una relación que existe entre dos o más entidades.
- ✓ **Modelo de bajo nivel o físicos:** Este tipo de modelo va dirigido a los usuarios expertos en el área de informática, ya que este modelo es el más abstracto, presenta la forma en como se almacena la base de datos en la computadora y las rutas de acceso de la misma.
- ✓ **Modelo de representación o de implementación:** Es el punto intermedio entre los dos modelos anteriores, de tal forma que un usuario no experto podría interpretarlo, pero podría ver ciertos detalles de la forma en que se encuentran organizados los datos. En este se encuentran los SGBD comerciales, de este modelo se desprende tres esquemas o modelos más: El relacional, el de red y el jerárquico.

Modelo relacional.

Este modelo es el que tiene un mayor sustento teórico y es el más utilizado actualmente en el mundo de las aplicaciones de bases de datos, implicando un mayor desarrollo de SGBD de tipo relacional. Este modelo presenta a la base de datos como un conjunto de tablas relacionadas entre sí.

Modelo de red.

Los datos de la base de datos se presentan como registros y conjuntos, cada conjunto define una relación de uno a n registros. Los conjuntos pueden tener a un mismo registro como participante o propietario.

Modelo jerárquico o arborescentes.

Presenta una estructura de tipo Padre-Hijo, un esquema de este tipo presenta a los registros ordenados de forma jerárquica, es decir, en forma de árbol.

Capítulo IV. Programación Orientada a Objetos.

4.1. Un poco de historia.

Al principio la programación se basaba en un conjunto de instrucciones que se ejecutaban una tras otra. En estos programas se podían efectuar cambios de secuencia basados en dos tipos de especiales de instrucciones de control. Las instrucciones de control se dividen en dos tipos: Instrucciones de transferencia e instrucciones de llamada a subrutina.

- ✓ Las instrucciones de transferencia, que dan lugar a un salto sin retorno en la ejecución. Estas sentencias a su vez pueden ser condicionales (Si el salto se produce únicamente cuando se cumple una condición determinada) o incondicionales.
- ✓ La instrucción llamada de subrutina, que guarda la dirección de retorno. De esta forma, la secuencia de ejecución original puede reanudarse cuando termina la ejecución de la subrutina, es decir, cuando se ejecuta una instrucción de retorno.

Programación simbólica.

En los años cincuenta ya era un problema la programación que cada vez era una tarea más compleja que hacía prácticamente inutilizable el lenguaje de máquina, esto dio lugar a que surgiera lo que se conoce como la programación simbólica. El código de operación y la dirección de memoria asociados a cada instrucción de la máquina se representan pos símbolos alfanuméricos.

Lenguajes de tercera generación.

A finales de los años cincuenta las aplicaciones iban creciendo en tamaño y complejidad, la programación en los lenguajes simbólicos resultaba aún demasiado compleja. Así comenzaron a aparecer los lenguajes de alto nivel o de tercera generación tales como: FORTRAN, LISP, COBOL, ALGOL, APL, PL/I, BASIC, Pascal, C, C++, entre otros. Los programas escritos en estos lenguajes se caracterizan por que las instrucciones están escritas en inglés y las expresiones tienen aspecto matemático.

El nivel de los lenguajes.

Según se aumenta el nivel de los lenguajes de programación, también se aumenta el número de instrucciones de código máquina asociadas a una determinada instrucción. Cuanto menor es el nivel del lenguaje, más cerca se encuentra del lenguaje de máquina y más lejos del lenguaje humano, pero lo que pierde en inteligibilidad se suele ganar en flexibilidad. En la actualidad, los lenguajes de bajo nivel se suelen utilizar para la programación de pequeñas rutinas que necesitan estar altamente optimizadas o bién para la creación de librerías de funciones especializadas que lo único que hacen es asociar varias instrucciones del lenguaje de bajo nivel, para lograr funciones más complejas. Hay lenguajes que combinan la flexibilidad y rapidez de los lenguajes de bajo nivel y la facilidad de programar de los lenguajes de bajo nivel a estos lenguajes se les conoce como lenguajes de nivel medio como por ejemplo el lenguaje C y el C++.

Programación procedimental.

En este tipo de programación los programas son un conjunto de bloques, donde cada bloque tiene una función o propósito específico, estas actividades está coordinadas por un bloque principal tal y como se muestra a continuación:

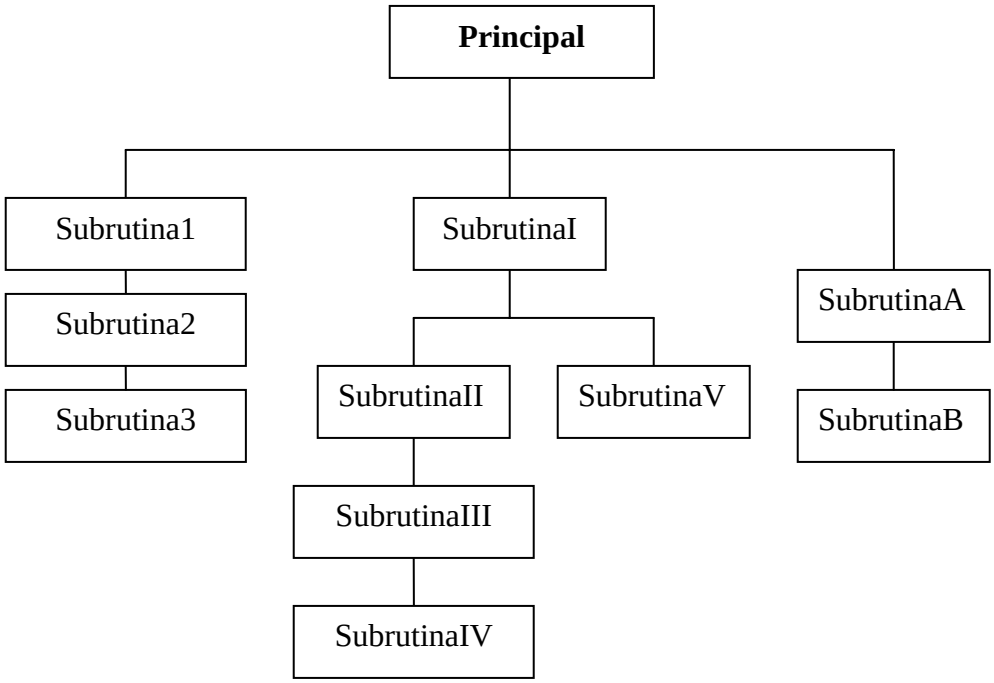


Figura 4.1

El diagrama anterior es un diagrama de TOP DOWN y puede representarse en lenguajes de alto nivel tales como C y Pascal, etc. Esta era una forma muy común de organizar los programas. Este diagrama es muy parecido a los árboles genealógicos y representan la estructura padre-hijo. Esto es, el bloque principal controla a los bloques Subrutina1, SubrutinaI y subrutinaA, es decir, principal tiene tres hijos, la SubrutinaA controla a la subrutinaB, es decir, SubrutinaA tiene un hijo y así con los demás… Si elaboramos la estructura principal de este programa en un lenguaje de alto nivel quedaría del siguiente modo:

```
Program principal;
Uses crt;
                    .
                    .
                    .
procedure Subrutina1;
procedure Subrutina2;
procedure Subrutina3;
begin
                                        .
   .      { Subrutina3}
```

```
                                                    .
    end;


    begin
                                                    .
.   {Subrutina2}
                                                    .
    end;
    begin
                                                    .
    .   {Subrutina1}
    end;
procedure SubrutinaI;
procedure SubrutinaV;
begin
                                                    .
                                                    .
                                                    .
    end;
    procedure SubrutinaII;
procedure SubrutinaIII;
procedure SubrutinaIV;
begin
                                                .
    .       {SubrutinaIV}
                                                .
    end;
    begin
                                                .
                                                .   {SubrutinaIII}
                                                .
    end;
    begin
                                            .
                                            .   {SubrutinaII}
                                            .
    end;
```

```
    begin
                        .
    .{SubrutinaI}
                        .
end;
procedure SubrutinaA;
    procedure SubrutinaB;
    begin
                        .
    .   {SubrutinaB}
                        .
    end;
    begin
                    .
    .   {SubrutinaA}
                    .
    end;
begin
    .
.   {Principal}
end.
```

En los niveles más bajos suelen situarse los programas que prestan servicios especiales a casi todos los demás. En los niveles más altos se suelen encontrar las rutinas que se encargan de la interfaz de usuario o de las comunicaciones del programa con el exterior. Si embargo a pesar de que esta forma de programar facilita mucho el análisis, este tipo de organización depende todavía más del programador que de la realidad en sí. El decide como se van a organizar los bloques y como se van a relacionar los datos, esto se vuelve todavía mucho más dramático cuando los programas son muy extensos y tienen fallas, la dependencia hacia el autor del programa es notable y le resta productividad al programa, estos son los problemas más importantes:

- ✓ El partir de un modelo mental equivocado.
- ✓ La dificultad para modificar y extender programas.
- ✓ La dificultad para mantener los programas.
- ✓ La dificultad a la hora de reutilizar los módulos.

Modelo mental equivocado.

En diferencia entre lo que pensamos y lo que es, se encuentran muchos de nuestros problemas y esto también se extiende en la programación. La programación procedimental se basa fundamentalmente en los verbos, que en nuestra estructura mental. Si un perro pasa corriendo por nuestro campo de visión, lo primero que pensamos no es << Algo corre>>, sino <<un perro corre>> Para comprobar que esta afirmación es cierta, basta recordar cuáles son los nombres más corriente en las funciones de nuestros programas: abrir, cerrar, leer, escribir, guardar, etcétera. La jerarquía de una aplicación procedimental está formada por subprogramas, es decir, por verbos, mientras que los datos desempeñan un papel secundario en este estilo de programación.

Modificación y extensión de los programas.

La introducción de componentes nuevos no suele poder hacerse de una manera simple y ordenada, sin que tenga que modificarse, en consecuencia gran parte de la versión anterior de la aplicación. Esto se debe a la existencia de datos globales o locales compartidos por varios subprogramas, que introducen interacciones ocultas entre ellos. Debido a esto, cualquier cambio en uno de los subprogramas que afecte a estos datos compartidos, puede provocar efectos secundarios, imprevistos e indeseables.

La solución al problema de los datos compartidos entre módulos o subrutinas de una aplicación, lo encontramos en la programación orientada a objetos. Mediante el encapsulamiento es posible asignar datos a un módulo que solamente se podrán utilizar por dicho módulo y por lo tanto, que no serán afectados por las modificaciones introducidas en otros módulos.

Mantenimiento de programas.

En casi todos los sistemas informáticos muy grandes se encuentran errores ocultos, que no surgen a la luz hasta después de muchas horas de funcionamiento, cuyo efecto es tanto peor cuanto mayor sea la envergadura del mismo. Sobretodo cuando esta tarea es realizada por programadores que no participaron inicialmente en el diseño del programa, este problema puede ser de muy difícil solución. En ocasiones por la tecnología utilizada, es mucho más difícil buscar y solucionar el problema, que recurrir a la realización de una nueva aplicación utilizando modernas técnicas y lenguajes de programación.

Reutilización de programas.

Con el tiempo las situaciones y los problemas cambian, y las aplicaciones de software que fueron útiles y oportunas en su momento actualmente ya no lo son y por lo general son reemplazadas por otras más funcionales. Y si a esto añadimos que un producto de software no resuelve por sí sola todos los problemas de una organización y que a medida que utilizamos nuestro sistema van surgiendo nuevos problemas, tenemos que hacer nuevas versiones de nuestro software pero estas se van complicando pues este tipo de programación que se fundamenta en los tres inconvenientes anteriores hace que conforme aumentan las modificaciones, el programa se va haciendo más incomprensible en el mejor de los casos, algo así como el juego de <<Tripas de gato”>>, un completo espagueti hecho software. Y con el tiempo el programa ya no se puede reutilizar. Es preciso cambiar nuestra forma de programar, aumentar la reutilización del código, disminuir el costo de diseño de

las aplicaciones, reducir los ciclos de construcción de los productos y facilitar su extensibilidad y mantenimiento.

4.2. Programación Orientada a Objetos.

La solución a los inconvenientes de la programación procedimental fue el desarrollo de una forma distinta de programar, que se apoyaba más en los sustantivos (Objetos) que en su funcionalidad (verbos), la programación orientada a objetos.[5]

4.2.1. ***Objeto*** es cualquier cosa del mundo real, por ejemplo un perro, es un objeto que puede tener las siguientes características:

Color, Raza, Edad, Peso, Tamaño, etc.… A estas características les llamaremos las propiedades, estas propiedades tienen valores, por ejemplo: Color negro, Raza Pastor

Alemán, cuatro años, pesa treinta kilos y mide metro y medio de cabeza a cola. Estos valores pueden variar incluso con el tiempo. Ahora bien este objeto realiza acciones tales como ladrar, mover la cola, comer, dormir, dar vueltas, a estas acciones se les conoce como el comportamiento del objeto. Ubiquemos esto en una tabla:

Objeto	Propiedades	Comportamiento
Sustantivo	Adjetivo	Verbo
Perro	Color Raza Edad Peso Tamaño Etc.	Ladra Mueve la cola Come Duerme Da vueltas. Etc.

Tabla 4.1

Ahora bien ***los eventos*** son acontecimientos externos que ocasionan una reacción en un objeto. Por ejemplo si le mostramos un jugoso bistec a nuestro perro, lo más seguro es que empiece a salivar y nos lo quiera quitar. Si le damos de palmaditas nuestro perro va a mover la cola etc.

Hagamos una analogía, con el objeto llamado computadora, la computadora recibe estímulos o señales externas, de acontecimientos externos que son los eventos, los receptores de eventos son: el mouse, el teclado, la impresora, el scanner, etc.

Los ***tres elementos fundamentales de la programación orientada a objetos*** que se consideran básicos para que pueda hablarse de esta forma de programar son:

- ✓ Los objetos: Entidades complejas provistas de datos (propiedades o atributos) y comportamiento (funcionalidad, programas o métodos).
- ✓ Las clases: conjuntos de objetos que comparten propiedades y funcionalidad.
- ✓ La herencia: Las clases no están aisladas, sino que se relacionan entre sí, formando una jerarquía de clasificación. Los objetos heredan las propiedades y el comportamiento de todas las clases a las que pertenecen.

Objetos, como su nombre lo indica, la programación orientada a objetos, tiene como elementos esenciales los objetos. De este modo es más sencillo trasladar nuestros modelos mentales a un lenguaje de programación. Dichos modelos se apoyan en objetos, que nombramos mediante **sustantivos**. Estos objetos que percibimos tienen a su vez una serie de **propiedades** (que representamos mediante adjetivos, como rojo, grande, estrecho, etc.) y comportamientos (que representamos mediante verbos como camina, detente, corre, da vuelta, etc.).

4.2.2. Clases. ***La clasificación de objetos es una de las actividades mentales básicas***. No solo percibimos los objetos, inmediatamente los clasificamos en clases. De esta forma cuando vemos un gato, sabemos que es un animal, mamífero y felino. De hecho la clasificación de objetos es una de las actividades mentales básicas. En efecto, cuando decimos <<veo un perro>>, en realidad estamos diciendo mucho más en forma implícita. Algo así como: <<veo un ser que pertenece a la clase o conjunto de los perros>>.

Con estos elementos estamos en posibilidad de definir una clase:

Una *clase* es una plantilla que se utiliza para crear múltiples objetos con características similares.

Las clases engloban todas las características de un conjunto particular de objetos. Cuando escribe un programa en un lenguaje orientado a objetos, u***sted no define objetos individuales, sino que define clases de objetos***. Pero no sólo conocemos muchísimas clases, y representamos cada una por nombre común. También distinguimos relaciones entre las clases diferentes; vemos que existe cierto parentesco más o menos lejano entre ellas. Como hemos visto, somos capaces de formar jerarquías, basadas ***en la relación subconjunto,*** que también podemos expresar mediante la frase <<está comprendida en>>. Ver figura 4.2

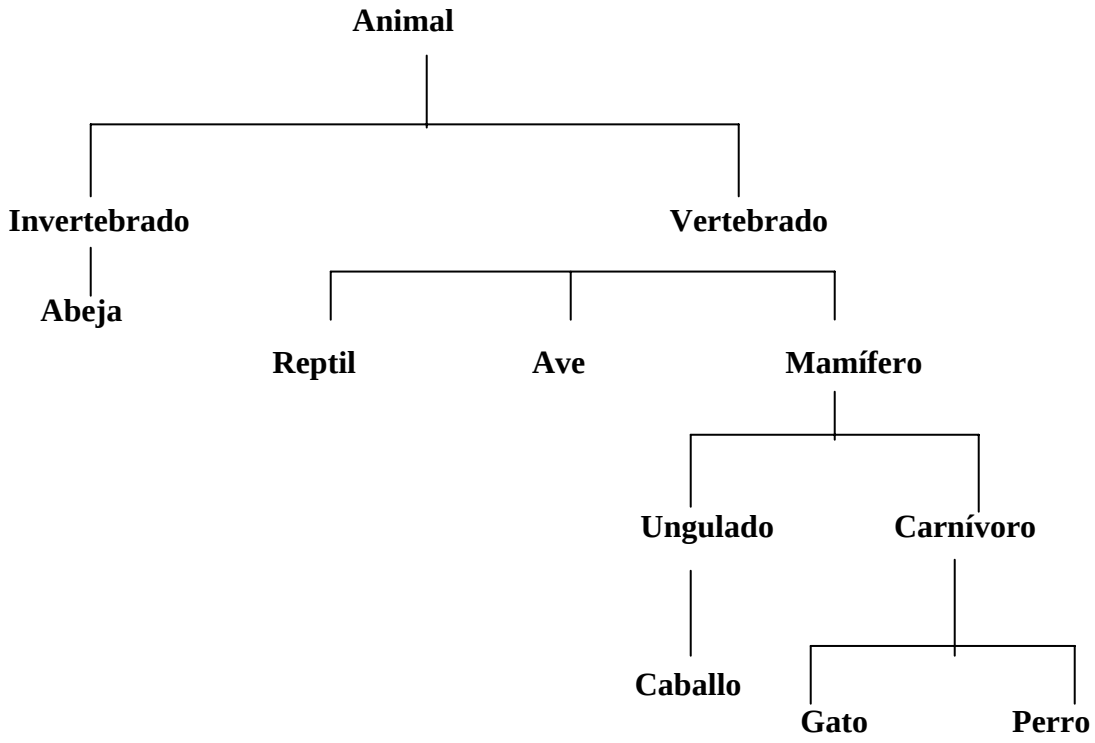


Figura 4.2

Como podemos apreciar, la jerarquía está organizada en forma de árbol que parte desde el animal y se ramifica en una clasificación entre vertebrados e invertebrados. Por ejemplo no solo decimos: <<este perro ladra >>, sino también <<los perros ladran >>, lo que equivale a decir que <<todos los perros ladran>> y lo consideramos como un propiedad característica de dicha clase. Esto es: Las clases de objetos también tienen propiedades y comportamientos asociados; los que son comunes a todos los objetos que pertenecen a la clase.

4.2.3. Herencia: Son los atributos y acciones que poseen elementos específicos de otros más generales. Por el mero hecho de que un objeto pertenece a una clase, sabemos ya muchas cosas de él. Por ejemplo: Si hablamos de un perro, aún cuando no lo hayamos visto, supondremos que ladra y que tiene cuatro patas. Esto es debido a que le aplicamos las propiedades y comportamientos de su clase. Pero este perro puede tener peculiaridades, por ejemplo: El perro puede no tener una pata, faltarle la cola o bien tener un color poco común, pero no por estas peculiaridades dejará de ser un perro. Otros ejemplos podrían ser: un tulipán negro, una perla negra o un garbanzo de a libra.

Por lo tanto, que los objetos pertenecen a una clase heredan las propiedades y el comportamiento de estas.

Tipos de herencia.

La herencia es una de las propiedades fundamentales de la programación orientada a objetos. Sin embargo, el concepto de herencia no es único, existen diversos tipos, y no todos son esenciales, en el sentido de que se sigue hablando de la programación orientada a objetos aunque falte alguno.

Dichos tipos, son desde el punto de vista de lo que se hereda:

- ✓ Herencia de métodos: Los objetos pertenecientes a una clase determinada heredan el comportamiento en forma de métodos (Programas) asociados a dicha clase y a todas sus superclases. Esta forma de herencia es esencial.
- ✓ Herencia de propiedades: los objetos pertenecientes a una clase determinada heredan las propiedades (declaraciones de datos) asociadas a dicha clase y a todas sus superclases. Esta forma de herencia también es esencial.
- ✓ Herencia de valores de las propiedades: Los objetos pertenecientes a una clase determinada heredan valores por omisión de las propiedades asociadas a dicha clase y a todas sus superclases. Esta forma de herencia no se considera esencial.

Según el número de clases a las que pertenece un objeto, la herencia se puede clasificar en:

- ✓ Herencia simple: Cada objeto puede pertenecer a una clase y heredar de ella. Esta forma de herencia es esencial.
- ✓ Herencia múltiple: Un objeto puede pertenecer a más de una clase simultáneamente y puede heredar de todas ellas. Esta forma de herencia no se considera esencial, ya que si no existe no es difícil simularla utilizando clases compuestas.

Aún hay más: los objetos no solo heredan las características de la clase a la que pertenecen, sino las que cualquier clase situada en una posición más alta en la jerarquía. Este sistema de herencia ahorra gran parte del trabajo a la hora de definir las propiedades de las clases ya que gran parte del proceso de definición es automático.

Este sistema de herencia ahorra gran parte del trabajo a la hora de definir las propiedades de las clases ya que gran parte del proceso de definición es automático.

Nota: Hay lenguajes basados en clases, que poseen objetos y permiten agruparlos en clases, pero no utilizan herencia. CLU, Sedl y Visual Basic son lenguajes de este tipo. Solo cuando se dan los tres elementos: ***objetos, clases y herencia***, podemos hablar de una verdadera programación orientada a objetos.

Métodos, mensajes y variables de objetos.

Los objetos vienen a corresponderse con los sustantivos en nuestra forma de ver el mundo. Además de estos *sustantivos*, hemos visto que aparecen de forma natural los *adjetivos*, que nos indican las propiedades de un objeto, y los verbos, que nos dan su comportamiento.

Ahora nos concentraremos en estos dos últimos aspectos, que vienen a definir un objeto: Su *comportamiento*, implementando en forma de *métodos*, y sus *propiedade*s, que se especifica mediante variables de objeto. Además del mecanismo de disparo de un objeto mediante *mensajes*.

Importante: En la programación orientada a objetos se utilizan *mensajes* en lugar de llamadas a *subrutinas.* Los mensajes son similares a los parámetros que se les pasaban a las funciones y que hacen que un objeto se comporte de una forma o de otra.

Métodos: Un método es un programa especial que está asociado a un objeto(o más bien, a una clase de objetos), y su ejecución no se desencadena mediante una llamada de subrutina, sino mediante un mensaje. Así los métodos son los que dan el comportamiento de un objeto. Son los que implementan los verbos o acciones relacionados con el objeto.

Mensajes: Un mensaje es la comunicación dirigida a un objeto, que le ordena a este que ejecute uno de sus métodos con ciertos parámetros, como por ejemplo el objeto perro le transmitimos el mensaje: Ladra, corre, camina, da vueltas que hará que estos métodos se activen. En general todo mensaje considerará las tres partes siguientes:

- ✓ El objeto que es destinatario del mensaje.
- ✓ El método de dicho objeto que se ha de disparar.
- ✓ Los parámetros que se han de pasar a dicho método.

Variables asociadas a objetos.

Otro de los términos propios de la programación orientada a objetos es el de propiedad o atributo que corresponde a los datos (las variables) de la programación clásica, procedimental. Por esta razón a veces se denominan las propiedades de un objeto con el nombre de la variable. En la programación orientada a objetos existen dos formas principales de variables asociadas a los objetos:

- ✓ Variables de objeto: si cada objeto que pertenece a esta clase tiene su propio valor para esta propiedad.
- ✓ Variables de clase: Variables de clase, si existe un solo valor de la propiedad para todos los objetos de la clase. Estas propiedades o atributos en forma de variables vienen a implementar los adjetivos o cualidades del objeto.

Propiedades fundamentales de la Programación Orientada a Objetos.

Además de los tres elementos esenciales de la Programación Orientada a Objetos (objetos, clases o herencia) y del paso de mensajes ***existen otras dos propiedades*** que también se consideran fundamentales, hasta el punto de que muchos autores se niegan a reconocerle a un sistema o lenguaje el carácter de Orientado a Objetos, si falta alguna de ellas. Estas propiedades son:

- ✓ El encapsulamiento y ocultación de datos y programas.
- ✓ El polimorfismo.

Encapsulamiento: El encapsulamiento consiste en el hecho de que cada objeto, entendido como un agregado de datos y de programas, esté aislado del exterior, como envuelto en una cápsula. De esta manera el objeto es un módulo natural, y la aplicación entera se reduce a un agregado o rompecabezas de objetos. Además, el aislamiento protege a los datos asociados a un objeto contra su modificación por quién no tenga derecho a acceder a ellos, eliminando los efectos secundarios e interacciones mencionados al hablar de la programación procedimental.

Ocultación de datos y programas: En principio todos los datos internos de un objeto deberían de estar ocultos dentro de este, accesibles tan solo a los métodos o programas asociados al mismo objeto. Esta consecuencia del encapsulamiento se denomina ocultación de los datos.

Relaciones entre objetos: Además de los datos y programas, un objeto tiene también una tercera componente: Sus relaciones con otros objetos o clases de objetos. Estas relaciones pueden ser de varios tipos:

- ✓ Jerárquicas: Que definen al objeto como miembro de una clase y lo engarzan en la organización en que se basa la herencia. Esta relación puede describirse puede describirse con las palabras "*es un caso particular de*".
- ✓ Semánticas: Relacionadas con el propio significado de los objetos que forman parte de la relación, sin restricción alguna. La estructura interna de los objetos presenta dos capas o niveles diferentes, que podemos comparar con la clara y la yema del huevo. En efecto, cada una de las tres secciones que componen el objeto le corresponde un contenido diferente:

- ✓ **A los datos:** Propiedades, atributos o variables del objeto, se les asignarán los valores.
- ✓ **Los métodos, programas o funciones:** Poseen un código ejecutable**.**

- ✓ **Las relaciones:** De nuestro objeto con otros diferentes quedarán especificadas por medio de punteros a dichos objetos**.**

Polimorfismo: El polimorfismo consiste en el hecho de que programas diferentes asociados a objetos distintos, puedan compartir el mismo nombre, aunque el significado del programa varíe ligeramente de objeto en objeto. En la programación clásica procedimental, dos programas están obligados a tener distinto nombre, si desea que coexistan en la misma aplicación. En la Programación Orientada a Objetos, los nombres de los métodos (programas) están asociados a la clase de objetos a la que pertenecen. Dos clases diferentes pueden tener programas asociados con nombres idénticos, pues la existencia de mensajes en su lugar de llamadas directas, hace posible distinguir distintas versiones de programas con el mismo nombre.

Ventajas de la Programación Orientada a Objetos:

Estas ventajas son:

- ✓ Un modelo mental natural, basado en objetos (nombres), más bien que en la funcionalidad.
- ✓ La modularidad, pues los objetos y sus clases constituyen los módulos naturales en que se descomponen las aplicaciones construidas de esta manera.
- ✓ Es fácil modificar y extender los programas: El encapsulamiento asegura que unos objetos estén aislados de otros, por lo que pueden añadirse clases y objetos nuevos sin afectar el funcionamiento de los antiguos y sin provocar efectos secundarios desconocidos o imprevistos.
- ✓ Es fácil reutilizar los programas: las clases de objetos, si están bien diseñadas, podrán transplantarse de una aplicación a otra con el mismo esfuerzo y sin realizar cambios.
- ✓ Es fácil mantener los programas: las clases reutilizadas ya habrán sido bien depuradas en sus aplicaciones precedentes, por lo que no es probable que introduzcan errores ocultos.

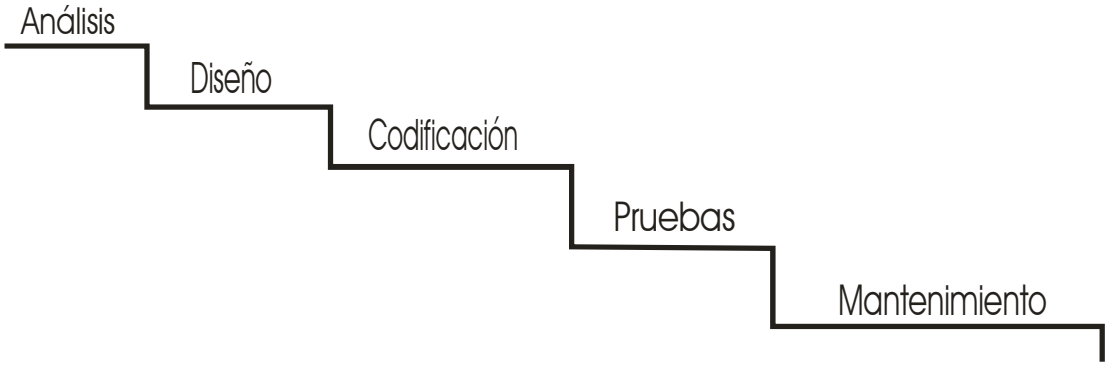
Inconvenientes de la Programación Orientada a Objetos:

Estos inconvenientes son:

- ✓ Un análisis y diseño más costoso, pues las clases deben ser cuidadosamente diseñadas para poder ser reutilizadas en el mayor número de aplicaciones posible. Es perfectamente posible diseñar mal las clases, de tal manera que sea prácticamente imposible reutilizarlas fuera de una aplicación concreta.
- ✓ El aprendizaje es más lento, especialmente cuando los programadores estaban ya acostumbrados a programar de manera clásica, pues deben aprender a pensar de otra manera.

Análisis y Diseño Orientado a Objetos.

En la programación procedimental clásica se acostumbra a utilizar una metodología establecida ya desde hace bastantes años, que se llama modelo en cascada. Ver figura 4.3:



Modelo en cascada para la creación de una aplicación clásica.

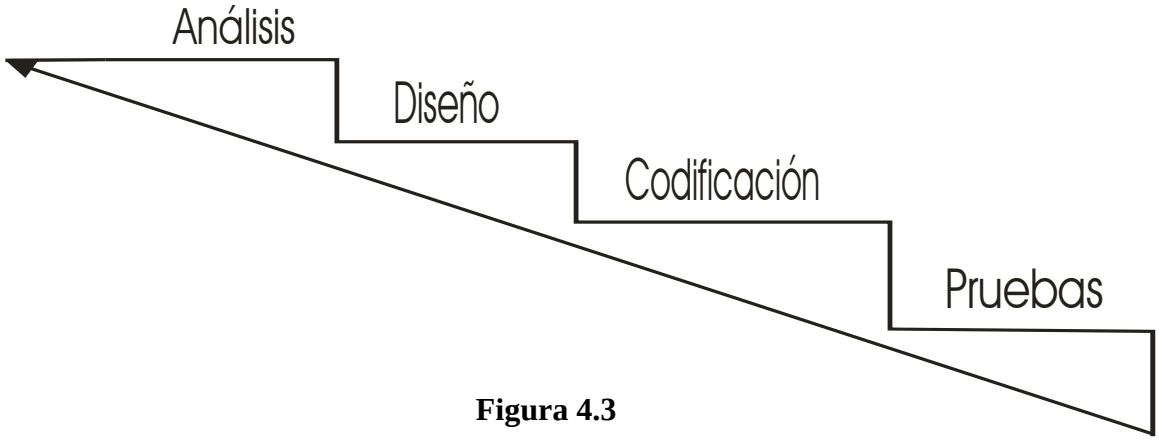


Figura 4.3

Modelo iterativo para la creación de una aplicación utilizando la Programación Orientada a Objetos.

- ✓ Análisis de la aplicación: Es decir, qué queremos hacer.
- ✓ Diseño de la aplicación: Cómo la vamos a hacer.
- ✓ Codificación o programación de la aplicación: Hagámoslo.
- ✓ Pruebas exhaustivas de la aplicación: En las que se comprueba que la aplicación, en efecto hace lo que debe hacer. En caso negativo, corrijámosla.
- ✓ Mantenimiento de la aplicación: Después de todo, puede que resulte que no hacía exactamente lo que debía hacer. Es preciso corregirla a posteriori. El modelo se llama en cascada por que se supone que una fase debe estar totalmente terminada antes de pasar a la siguiente. Es decir, no se comienza con el diseño hasta que no se da por definitivo el análisis, no se comienza a programar hasta que no se ha finalizado el diseño, y así sucesivamente. La programación orientada a objetos no utiliza más que parcialmente el modelo en cascada. Esto se debe a que esta forma de programar es mucho más modular, y además el encapsulamiento nos permite aislar cada clase de objetos de todas las demás, por lo que es posible realizar el proceso completo del análisis, el diseño, etc., de cada clase concreta como si no existieran las otras.

Algunas sugerencias:

No hay nada mejor que ver el resultado de nuestro trabajo plasmado en un programa. Seamos ingenieros, construyamos ideas y plasmémoslas en nuestro programa. Como norma general, si se quiere llegar a ser un experto programador, conviene pensar en distintas alternativas de resolución de problemas concretos, intentando optimizar en cada una de ellas algún parámetro esencial: Velocidad de ejecución y tamaño de código, son dos de las más típicas.

Programación Orientada a Eventos.

Un evento es un estimulo que provoca una respuesta en la computadora. **Ver tabla:**

Estímulo	Respuesta
Clic con el ratón	• Cerrar una aplicación. • Imprimir • Activar una herramienta de una aplicación de software. • Desactivar una herramienta de una aplicación de software. • Etc.
Oprimir una tecla	• Escribir un carácter. • Cerrar una ventana de diálogo. • Activar una ayuda. • Ejecutar un programa. • Compilar un programa. • Etc.

Tabla 4.2

La tarea de la programación orientada a eventos es darle inteligencia a los dispositivos que van a recibir eventos, es decir, escribir código para el ratón, para el teclado, para la impresora, para el escáner, y todos los medios receptores de eventos posibles. Un producto de software no sólo debe manejar eventos aleatorios, sino que necesita analizar y responder a los eventos. Un programa debe controlar todos y cada uno de los eventos apropiados en el momento en que ocurren e ignorar todos los demás.

La programación Orientada a Eventos consiste en que los objetos que maneja son controlados por los eventos. Teniendo en cuenta de que los objetos = sustantivos, adjetivos = propiedades = atributos, y verbos = métodos, son controlados por los eventos a través de sus receptores, mouse, teclado impresora, scanner, etc. Esta es la filosofía de la programación orientada a eventos que controlan los objetos.

Diseño, Análisis y Planificación de un Proyecto.

- ✓ Documentos necesarios para la creación de un programa.
- ✓ Plan de viabilidad.
- ✓ Documento de requerimiento del sistema.
- ✓ Modelo de datos.
- ✓ Análisis técnico.
- ✓ Desarrollo.

- ✓ Instalación.
- ✓ Formación de usuarios.

a) Plan de viabilidad.

- ✓ Reuniones con los usuarios.
- ✓ Presentación de un cuestionario a usuario.
- ✓ Documentos de requerimientos de usuarios.
- ✓ Planificación del proyecto.
- ✓ Plan de reuniones. Confección de la agenda de citas.
- ✓ Detección de problemas externos e internos.
- ✓ Documentos de soluciones que aporta el sistema si es viable o motivos de la no viabilidad.
- ✓ Tarificación (tarifa), por partes de cada módulo.
- ✓ Aprobación del usuario**.**

b) Documentos de requerimientos del sistema.

- ✓ Requerimientos globales de la aplicación.
- ✓ Reunión con usuarios.
- ✓ Requerimientos puntuales de la aplicación.
- ✓ Definición del entorno de trabajo.
- ✓ Definición del lenguaje del programa.
- ✓ Definición del equipo necesario (material informático, otro material, personal, etc.)
- ✓ **Aprobación y visto bueno del usuario.**

c) Modelo de datos.

- ✓ Estructura básica de la aplicación.
- ✓ Interrelación de los datos.
- ✓ Diccionario de datos.
- ✓ Diseño lógico de datos.
- ✓ Diseño físico de datos.
- ✓ Aprobación y visto bueno del departamento técnico informático del usuario(si existe)

d) Análisis técnico.

- ✓ Diseño de prototipo.
- ✓ Diccionario de funciones.
- ✓ Integración de las funciones. Análisis funcional**.**

e) Desarrollo(temas a realizar por cada uno de los módulos del programa)

- ✓ Diseño.
- ✓ Prototipo.

- ✓ Aceptación del prototipo.
- ✓ Desarrollo.
- ✓ Pruebas.
- ✓ Integración de módulos.
- ✓ Desarrollo de la integración.
- ✓ Pruebas integradas**.**

f) Instalación
g) Formación de usuarios.

- ✓ Manual de referencia rápida.
- ✓ Manual de usuario.

El Lenguaje Unificado de Modelado UML es un lenguaje gráfico, para visualizar, especificar, construir y documentar los artefactos de un sistema con gran cantidad de software[1].

El UML es una creación de Grady Booch, James Rumbaugh e Ivar Jacobson, este lenguaje gráfico surgió por la necesidad de programar sistemas complejos que puedan entender todas las personas involucradas en el diseño de un sistema de información.

Orientación a objetos. Los objetos, son todos los elementos que forma el universo. Ahora bien, para distinguir unos objetos de otros se les agrupa en clases tal y como se indica a continuación:

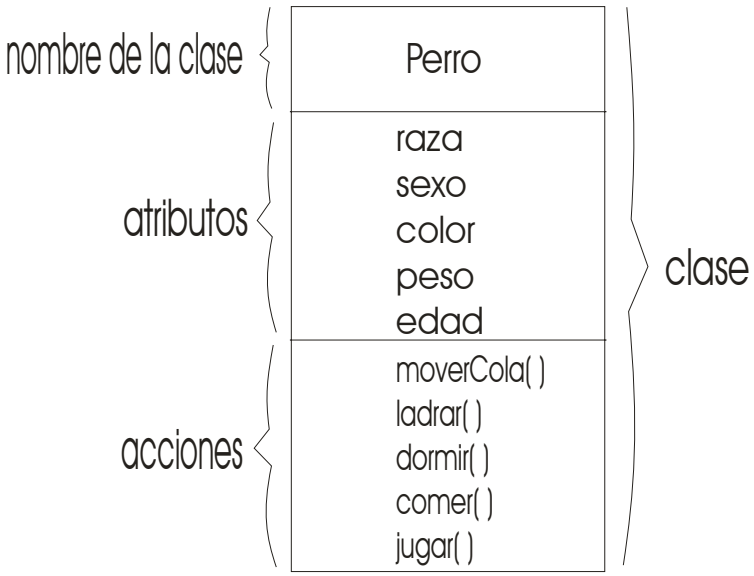


Figura 4.3

Entre más atributos y acciones tenga la clase, más similitud tendrá con la realidad.

Sobre la forma de escribir el nombre de las clases, atributos y acciones. El nombre de una clase empieza con una letra mayúscula y se coloca en la parte superior del rectángulo**,** cuando el nombre de una clase consta de dos o más palabras, estas empiezan con

mayúsculas por ejemplo las clases Perro, Computadora, LavadoraIndustrial, DiscoCompacto, etc.

El nombre de los atributos empiezan con minúsculas y cuando tiene dos o más palabras solamente la primera empieza con una letra minúscula, el nombre de los acciones es similar al de los atributos y va precedido por paréntesis Ver fig4.4.

Ahora centrémonos en la clase microcomputadora:

Figura 4.4

En la figura 4.5 se puede apreciar que se le pueden agregar más atributos y acciones a la clase Microcomputadora vista desde el usuario común.

Figura 4.5

En la figura 4.5 se muestra la clase Microcomputadora vista desde el especialista.

Abstracción: Es agregar o quitar atributos y acciones para que solo queden los que sean necesarios.Para el usuario son necesarios los atributos y acciones de la figura 4.5, para el especialista puede ser suficientes los atributos y acciones de la figura 4.6, aunque el atributo numeroSerie es irrelevante para el especialista y se puede omitir, auque es un atributo importante para el auditor.

La clase Canario hereda los atributos y acciones de la clase Aves: Su piel está cubierta de plumas, tiene pico y alas, es ovíparo y además realiza acciones tales como emitir sonidos con el pico para comunicarse, revolotear las alas etc.

Ahora lo vamos a ilustrar con la clase servidores.

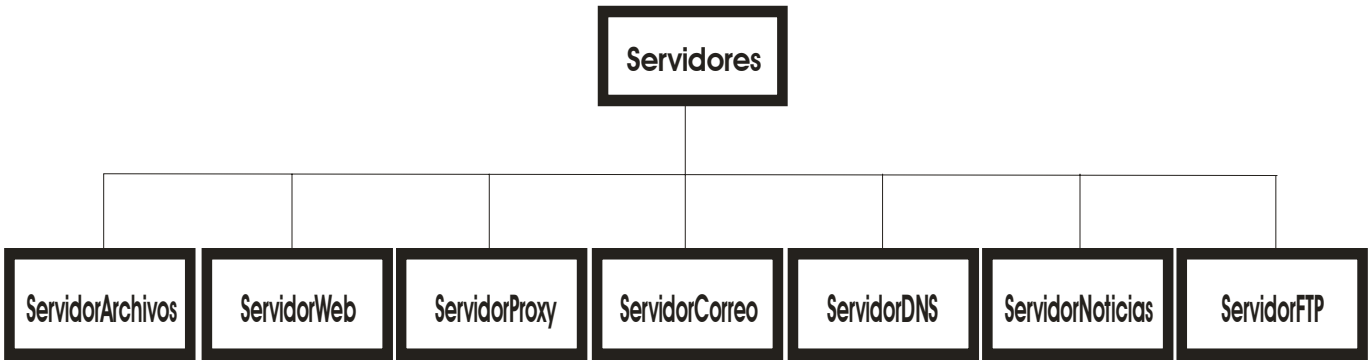


Figura 4.7

Polimorfismo: Se le llama a las acciones que tienen el mismo nombre en clases diferentes. Por ejemplo la operación abrir ( ), podemos abrir la llave del agua, abrir la puerta, abrir un libro, abrir el periódico, abrir el refrigerador, en la orientación a objetos se realiza una operación diferente con el mismo nombre, en este caso abrir ( ), cada clase sabe como realizar tal operación.

Encapsulamiento: Característica que tienen los objetos de ocultar su funcionalidad interna. Por ejemplo cuando se imprime un archivo, no es relevante saber como le hizo la computadora para comunicarse con la impresora y cómo imprime los archivos. Lo importante en ese momento es imprimir el documento.

Envío de mensajes: Es la forma de comunicación entre los objetos. Por ejemplo: Un censor detecta que un cliente se acerca y entonces manda una señal a un sistema electromecánico que hace que una puerta se abra automáticamente.

Asociaciones: Son todas las relaciones posibles que hay entre los objetos. Por ejemplo:

Juan e Irma estudian en el mismo grupo. Por lo tanto ambos son compañeros. Pero Irma da clases de aerobics y Juán está en ese grupo. Por lo tanto Juán es alumno de Irma.

Gráficamente se expresa del siguiente modo:

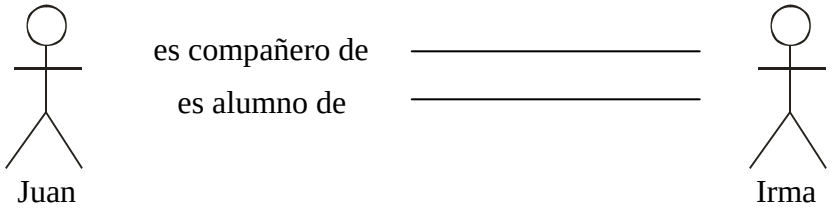


Figura 4.8

La primera asociación es en dos direcciones y la segunda es una asociación en una dirección.

Ahora una clase puede asociarse con más de una clase:

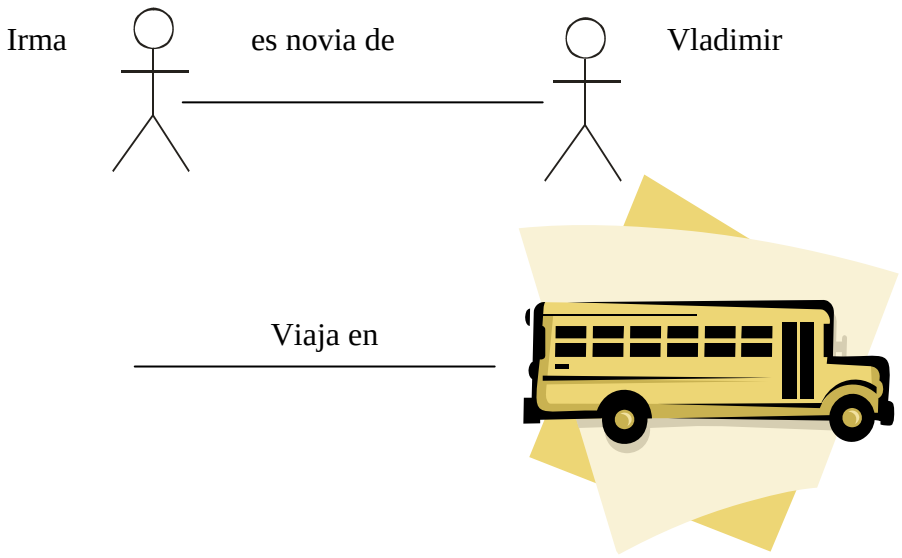


Figura 4.9

Agregación: Es la conformación de un objeto a partir de otros de diverso tipo. Por ejemplo una red LAN es una computadora que está formada por varias computadoras, impresoras, módems, etc. Ver figura 4.9

Figura 4.10

Paquete: Es un conjunto cuyos elementos son grupos. Figura 4.10

Paquete 

Figura 4.11

Ahora bien, los elementos de un paquete pueden organizarse en clases e identificar a la clase a través de su paquete:

Clase con nombre de ruta

Figura 4.12

La clase servidor Web es una subclase de la clase servidores, pero ahora la clase servidor Web un paquete y sus elementos lo componen las clases: Servidor Apache, Servidor Microsoft IIS, Servidor Zeus, Servidor IPlanet.

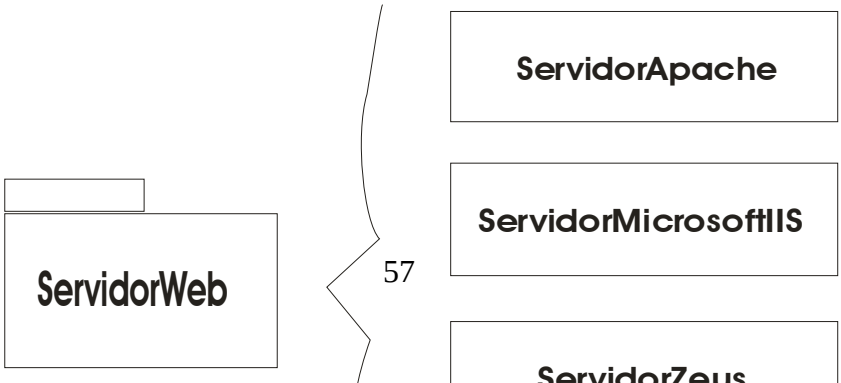


Figura 4.13

Como ya lo mencionamos, un servidor Apache, es un elemento de un paquete llamado ServidorWeb, podríamos darle el nombre de ServidorWeb : : ServidorApache, a este tipo de nombre se le conoce como nombre de ruta, la siguiente figura es otra forma de representar la clase ServidorApache.

Figura 4.14

A continuación mostramos la clase servidor con algunos de sus atributos:

Figura 4.15

UML da la opción de indicar la información adicional de los atributos. En el símbolo de la clase, podrá especificar un tipo para cada valor del atributo. Entre los posibles tipos se encuentran cadena (string), número de punto flotante (float), entero(integer) y booleano (boolean), así como otros tipos enumerados

Generación de objetos: Las clases generan objetos cuando los atributos de la clase tienen valores específicos. El nombre de un objeto está precedido por dos puntos que a su vez están precedidos por el nombre de su clase y todo el nombre va subrayado. Un objeto cuenta con un valor específico en cada uno de los atributos que lo comparan.Ver figura 4.16:

Figura 4.16

Importante: Si el atributo consta de una sola palabra, se escriben en minúsculas, por otro lado, si contiene más de una palabra, cada palabra será unida a la anterior y comenzará con una letra mayúscula, a excepción de la primera palabra que comenzará en minúscula.

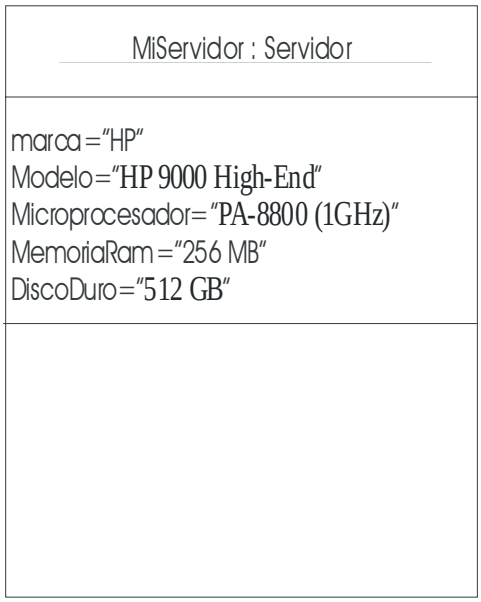


Figura 4.17

Así como es posible establecer información adicional de los atributos, también lo es concerniente a las operaciones. En la práctica no siempre es necesario mostrar todos los atributos y operaciones de una clase, pues esto crearía un diagrama muy saturado.

Se trata de representar todos los componentes del sistema desde diferentes perspectivas, cada perspectiva es un diagrama UML, de los cuales se incluyen nueve de estos diagramas:

1. Diagrama de clases.
2. Diagrama de objetos.
3. Diagrama de casos de uso.
4. Diagrama de secuencia.
5. Diagrama de colaboración.
6. Diagrama de estados.
7. Diagrama de actividades.
8. Diagrama de componentes.
9. Diagrama de despliegue.

Para que las clases tengan algún sentido tenemos que relacionarlas de algún modo. Tales relaciones son: Asociaciones, Multiplicidad, Asociaciones calificadas, Asociaciones reflexivas, Herencia, generalización, y dependencias.

Vamos a juntar dos clases para que esto tenga sentido:

Cada una de las clases tiene un papel determinante dentro de la asociación. Pueden aparecer dos asociaciones entre las clases.

Pueden asociarse diversas clases con una en particular:

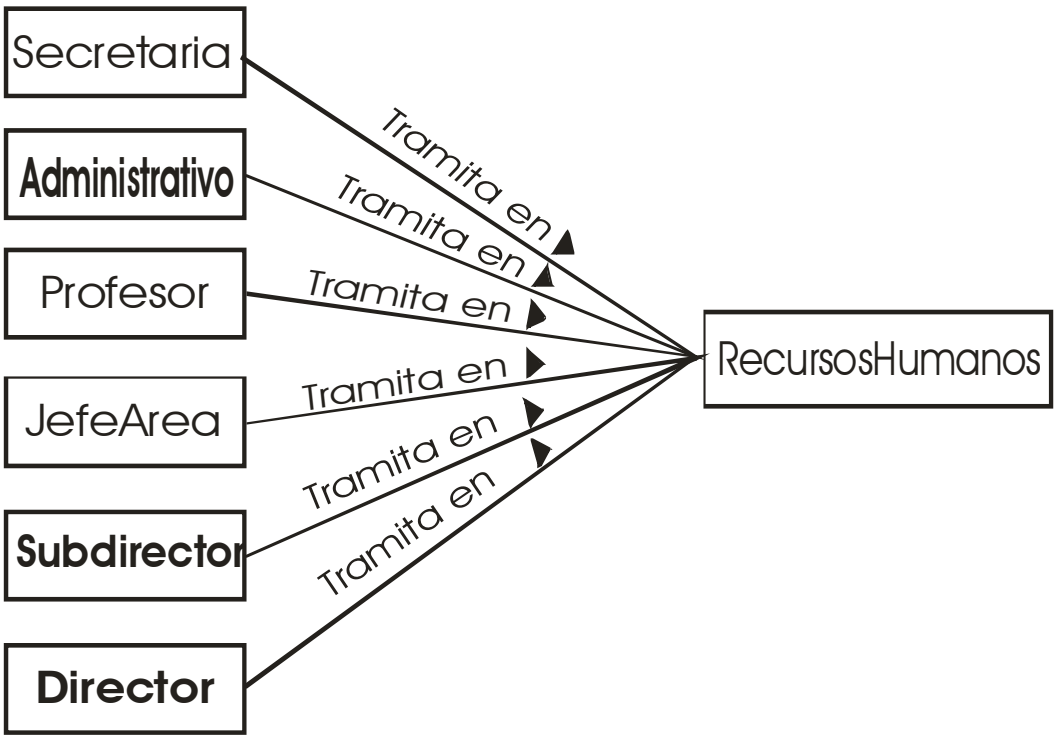


Figura 4.18

Restricciones en las asociaciones:

Ejemplo supongamos que un estudiante de nivel medio superior va a escoger una capacitación, primero que nada una restricción que va a estar encerrada entre llaves.

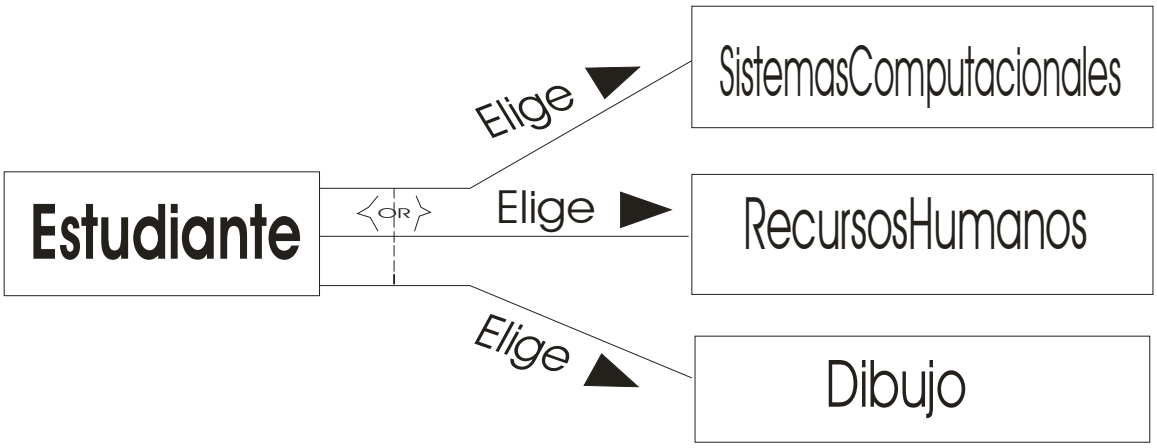


Fig6.15

Figura 4.19

Multiplicidad.

La multiplicidad es la especificación del rango de cardinalidades permisible que puede asumir un conjunto. (Grady Booch, James Rumbaugh e Ivar Jacobson).

La multiplicidad señala la cantidad de objetos de una clase que pueden relacionarse con un objeto de una clase asociada.

Hay varios tipos de multiplicidades. Una clase puede relacionarse con otra en un esquema de uno a uno, uno a muchos, uno a uno o más, uno a ninguno o uno, uno a un intervalo definido (Por ejemplo de uno a cinco hasta diez), uno a exactamente n , o uno a un conjunto de opciones. El UML utiliza un asterisco (*) para representar más y para representar a muchos. En un contexto O se representa por dos puntos, como en "I . . *" (Uno o más). En otro contexto, O se representa por una coma como 5, 10" ("5 o 10").

Búsquedas en UML.

En los sistemas con frecuencia se presenta un reto muy particular. La búsqueda. Su símbolo es un pequeño triángulo adjunto a la clase que hará la búsqueda. La idea es reducir, con eficiencia la multiplicidad de uno a muchos a una multiplicidad de uno a uno. Ver figura 4.20.

Figura 4.20

Dependencias. Es un tipo especial de relación donde una clase utiliza a la otra. Por ejemplo vamos a diseñar un sistema de inscripción para estudiantes, en este sistema vamos a utilizar como es lógico un formulario base. Ver figura 4.21:

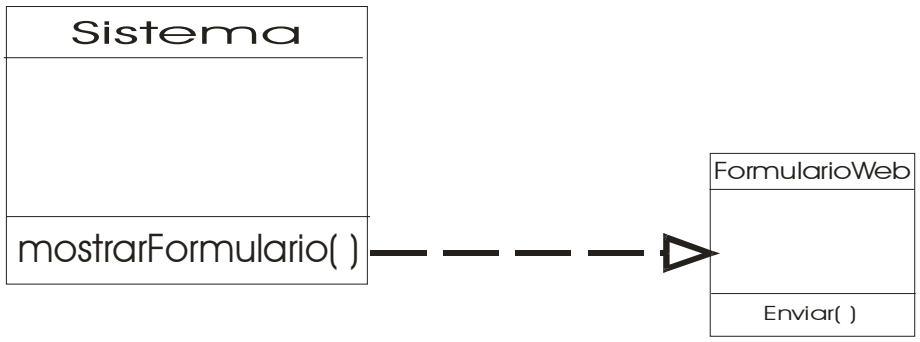


Figura 4.21

Las ideas estáticas de un modelo UML ayudan a que un analista se comunique con el cliente, y las ideas dinámicas ayudan al analista a que se comunique con el grupo de desarrolladores y ayudará a estos últimos a crear programas. La parte estática de un sistema

fue lo que tratamos en este capítulo y fue un puente entre los objetos que comúnmente estamos acostumbrados a manejar y el UML.

El reto es representar en un sistema UML la parte de la realidad que nos interesa estudiar, después de todo, de lo que se trata es de capturar nuestrar ideas, en un modelo que sea fácil de comprender, de tal forma que podamos comunicar nuestras ideas a otras personas, considerando que nuestro sistema es una combinación de hardware y de software, la comunicación de la idea es de suma importancia, después de todo ¿Por qué muchos de los sistemas en uso son ineficientes, engorrosos y difíciles de utilizar?

Pero sucede algo, los sistemas cambian con el tiempo y tienen que adaptarse a las nuevas circunstancias, hoy en día, es necesario contar con un plan bién analizado, en donde todos los actores coincidan, el objetivo de este trabajo no es profundizar en todas la vistas de un modelo, por que de lo que se trata es de que se compreda la importancia del UML en los sistemas de Internet/Intranet, los alcances y las limitaciones de estos conceptos en este caso en particular. Vamos a seguir explicando más conceptos del UML en los siguientes capítulos, y utilizar los objetos de varias herramientas de programación.

Capítulo V. Sistemas de Información con UML

5.1. Distribución de tareas en un Sistema de Información.

Vamos a plantear, el diseño de un prototipo de un sistema de información, utilizando UML. Comencemos con la organización de los datos. También deberemos adaptarlo a una red LAN, que es donde comúnmente se almacenan los sistemas de información.Una adecuada distribución las tareas determina el éxito o fracaso no solo de una Intranet sino de toda una organización, ahora bien, en este punto hay que ser muy cuidadosos ya que una sobrecarga de trabajo en una persona o departamento hace que un proyecto se venga abajo. Es por eso que este punto es importante, pues una inadecuada distribución de roles hace que este logro tecnológico sea ineficiente e irrelevante. Esta distribución de roles la puede plantear uno mismo haciendo un bosquejo aunque es recomendable la participación de los empleados o gente externa que ya esté usando una Intranet. Aunque se puede ser más específico en el diseño, este bosquejo presenta características frecuentes de la distribución de responsabilidades en una Intranet. Ver figura 5.1.

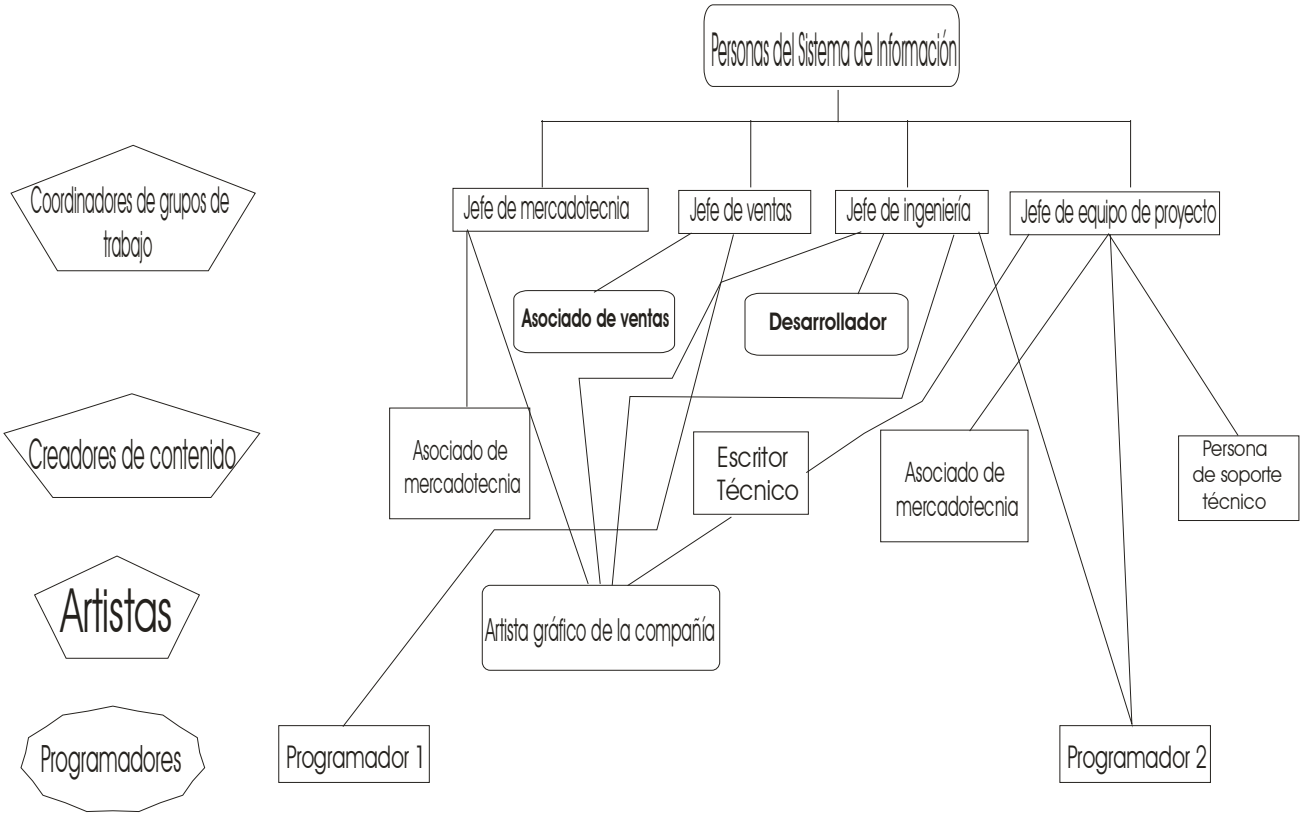


Figura 5.1. Distribución de tareas en un sistema de Información.

- ✓ El administrador Web

  El administrador Web se encarga de los problemas que se puedan presentar en los servidores de una Intranet, tales como los servidores de correo electrónico, los servidores de grupos de noticias, servidores de documentos Web entre otros. Tales personas pueden ser las responsables de la instalación de la Intranet.

- ✓ El administrador técnico de la Web.

Son técnicos en la organización que se encargan de la administración de los sistemas de Información. Esta persona se encarga de resolver los problemas de las contraseñas, correo electrónico, además de instalar y configurar cualquier tipo de software de la Intranet en las computadoras de los empleados y de los servidores. Hay que establecer los derechos de acceso y de crear cuentas en los servidores, configurar las cuentas de usuarios remotos y algo muy importante, crear copias de seguridad en la información.

- ✓ El administrador no técnico de la Web.

Es una persona con preparación administrativa que gestiona la Intranet. Es preferible que esta persona tenga un completo dominio de la gestión de los Sistemas de Información. A esta persona se le encargará realizar la administración necesaria del lugar Web de acuerdo con las peticiones de los empleados.

- ✓ Coordinadores de grupos de trabajo.

Un coordinador de grupos de trabajo puede tener distintos roles, pero el fundamental es el de fijar los objetivos y procedimientos al grupo. Los coordinadores de grupo deben ser jefes de departamento o haber sido nombrados jefes de proyecto. Coordina al grupo y mantiene a todos en línea respecto a sus objetivos y fechas límite. Los conocimientos de un coordinador se limitan a la parte operativa y explicarles a sus subordinados a utilizar las herramientas de comunicación. Hay que indicar el tipo de información que debería comunicarse con cada herramienta.

- ✓ Creadores de contenido.

Los creadores de contenido se dedican a crear la información útil, situadas en los lugares accesibles para el resto del grupo. Aunque todo el mundo de la compañía hasta cierto punto es un creador de contenido, los principales son quienes crean las páginas Web, envían y responden a las preguntas que llegan al servidor Web y diseñan los documentos, como los formularios financieros. También crean diagramas, tablas y otros elementos visuales que le ayuden a comprender un concepto o un hecho incrementando la productividad de la compañía. El creador de contenido está relacionado con las jefaturas departamentales, intercambiando la información que va a ser publicada en los servidores de la Intranet. El creador de contenidos también tiene que dedicarse a actualizar la información de la compañía. La información tiene su propio tiempo de vida, por lo que debe de actualizarse regularmente.

- ✓ Artistas de la Intranet.

Son los encargados de darle presentación gráfica a un sitio Web. Los artistas ayudan a que el Web sea virtualmente atractivo, divertido y fácil de usar. Si uno se fija en los lugares Web inmediatamente se puede ver que los que presentan un solo texto ya no atraen. Se necesitan gráficos atractivos, fuertes para que el mensaje llegue y se recuerde. Este profesional debe contar con el dominio de las herramientas para crear y gestionar gráficos y sus librerías.

- ✓ Programadores de la Intranet.

Los programadores no están involucrados con el servidor, su objetivo fundamental es incrementar la productividad de la Intranet. Los programadores crean software de apoyo de la Intranet. Así como también son los principales involucrados cuando esta cambia notablemente con el tiempo. Los programadores pueden ayudar a que un lugar en el Web sea más atrayente. Los programadores pueden, por ejemplo, crear applet de Java que hace girar gráficos y deslizar texto por la página Web. Pero Java sirve para mucho más que eso. Para el desarrollo de software para crear una Intranet son suficientes las siguientes herramientas:

Figura 5.2

- ✓ Experiencia del programador.

Estas son algunas de las habilidades exigibles para un programador:

- ✓ Dominio del lenguaje de programación.
- ✓ Experiencia en:
    1. Análisis de los sistemas o programas.
    2. Diseño de bases de datos.
    3. Ajuste y puesta en servicio de las bases de datos.
    4. Diseño de las interfaces gráficas de usuario (GUI).
    5. Open Database Connectivity (ODBC) o Java Database Connectivity (JDBC).
    6. Conocimiento de las aplicaciones para páginas Web y de HTML.

5.2. El diagrama de clase describe los tipos de objetos que hay en el sistema y las diversas clases de relaciones estáticas que existen entre ellos. Hay dos tipos principales de relaciones estáticas:

1. **Asociaciones.** Por ejemplo, un estudiante se llena un formulario de inscripción.
2. **Subtipos.** Un formulario es un objeto, un objeto es un elemento de una clase.

Vamos a desarrollar un diagrama de clase de <<inscripciones para estudiantes de preparatoria>>. Ver figura 5.3

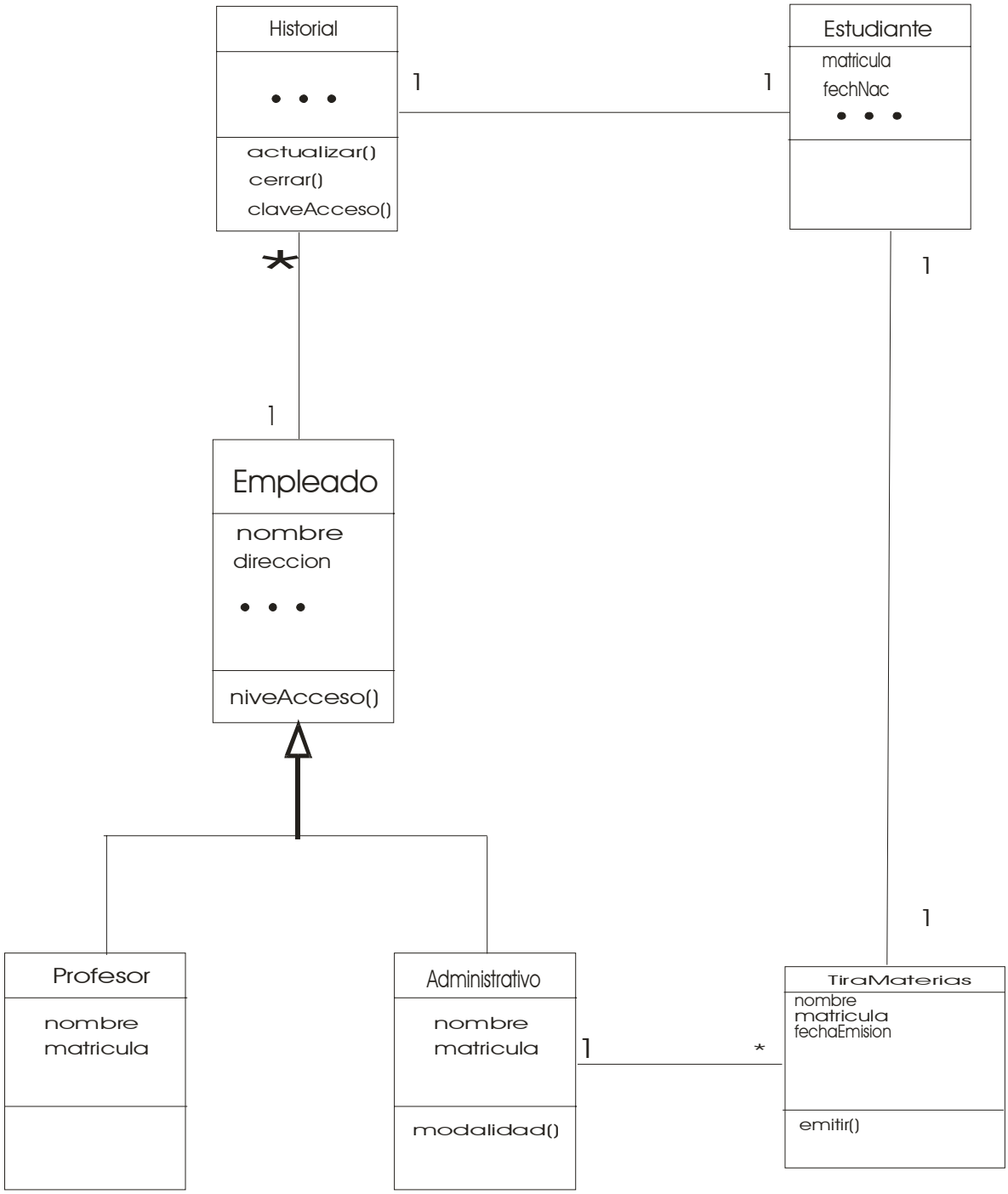


Figura 5.3

El diagrama anterior es ilustrativo y es más sencillo de entender para un programador promedio (ver pág5 cap2) pero no para el usuario común del sistema, que es con quien vamos a iniciar para desarrollar nuestro sistema de información, es decir, vamos a empezar por apreciar el sistema desde el punto de vista del usuario y para tal efecto vamos a desarrollar los diagramas de caso de uso.

- ✓ 5.3. Diagramas de casos de uso.

Con esta información podemos empezar a plantear nuestro **diagrama de casos de uso,** es decir vamos a empezar a modelar a partir de la información que nos proporcionen los usuarios del sistema, que ya han sido mencionados anteriormente en la distribución de los roles. El caso de uso es una estructura que ayuda a los analistas a trabajar con los usuarios para determinar la forma en que se usará el sistema.

A cada una de estas personas que les hemos distribuido alguna responsabilidad se les llama **actores.** El resultado de la secuencia debe ser algo utilizable ya sea por el actor que la inició, o por otro actor.

Importancia de los casos de uso.

La ventaja de diseñar diagramas de casos de uso es que son una herramienta que permite a los clientes comunicarnos desde su punto de vista aspectos importantes del sistema. Pues el diseño tradicional de un sistema era casi una ciencia oculta.

Ivar Jacobson, además de introducir los casos de uso como elementos primarios del desarrollo del software, también diseñó un diagrama para la presentación gráfica

Para empezar se desarrollará un caso de uso donde se expongan las distribuciones de cada uno de los compromisos de los actores en el sistema, donde el estudiante es uno de ellos.

Para esto visualizaremos la jerarquía de los usuarios que van a interactuar con la red LAN. Ver figura 5.4.

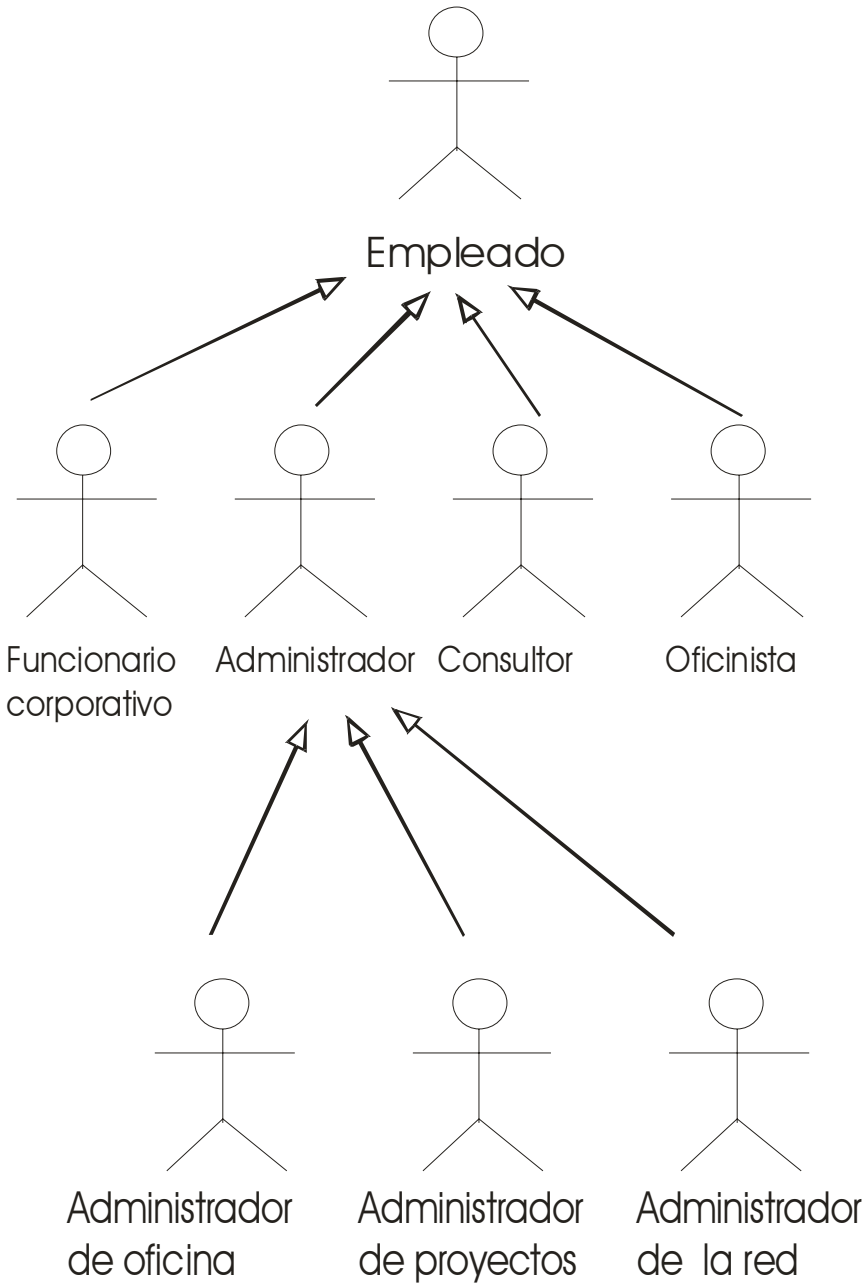


Figura 5.4. Distribución de responsabilidades

Como estamos hablando del funcionamiento interno del sistema no vamos a tomar por el momento en cuenta los actores, estudiantes, profesores o clientes, sino a las personas involucradas con el sistema.

¿Qué se entiende por incluir un caso de uso?

Cuando un conjunto de pasos de un caso de uso son los mismos que los de otro caso de uso.

¿Qué se entiende con extender un caso de uso?

Significa que se agregan pasos a un caso de uso existente.

- ✓ Alcance del proyecto.

Algunos de los factores que habría que ser considerados al evaluar el alcance de un proyecto son los siguientes:

- ✓ Interfaces con otras aplicaciones informáticas (por ejemplo: inventarios, contabilidad, etc.).
- ✓ Relación con las nuevas tecnologías (por ejemplo: sistemas sensibles a la voz, nuevos sistemas operativos, etc.).
- ✓ Marco de tiempo del proyecto (por ejemplo: comprimido o acelerado, o flexible u adaptado).
- ✓ Composición de los miembros del equipo (Por ejemplo: ¿Se necesitan muchas técnicas repartidas entre los diversos componentes del equipo, o se espera que cada miembro sea experto en varias técnicas?).
- ✓ Existencia de archivos de datos (Por ejemplo: ¿Existen datos o hay que crearlos?)

Departamentos y carpetas

Una de las tareas prioritarias en la administración de Intranets es la organización de la Información.

Estos pasos son recomendables:

1. Realizar un boceto general: Las divisiones de los departamentos, estos se van a convertir en nuestros directorios principales.
2. Crear subdirectorios en los directorios que contengan la información relacionada de cada uno de los directorios.

- ✓ **Opciones de almacenamiento.**

Los puntos más importantes a considerar son:

1. ¿Quién va a necesitar la información?
2. ¿Cuál es el lugar práctico e intuitivo para situarla? Cuando alguien la necesite, ¿podrá recuperarla fácilmente?
3. ¿Hay suficiente espacio en donde se encuentra la información?
4. Si la información es sensible o confidencial, ¿se puede restringir el acceso a la misma?

Entre las más comunes opciones de almacenamiento se encuentran los servidores de archivos y los servidores Web, siendo más recomendable esta última. Pues hay más respaldo de productos de software para gestionar la información con facilidad y que restringen el acceso a usuarios no autorizados.

- ✓ Uso de servidores de FTP.

Los servidores FTP se usan para descargar archivos, que son usados con frecuencia. Se pueden crear hiperenlaces con el mensaje, descargar archivos. El enlace está dirigido al servidor de FTP.

- ✓ Niveles de acceso.

El modo de controlar el acceso depende del tipo de servidores y el software que se use. Una vez seleccionado el método de acceso se puede construir un diagrama de acceso para definir los niveles. Algunos Sistemas Operativos, permiten diseñar algo que se conoce como sistema de archivos, que se hacen antes de ser instalado. Este sistema consiste en crear particiones y a cada una de estas particiones está diseñada para una tarea específica, tan específica como el sistema que nos interese. En la figura 5.5 se presenta un bosquejo de distribución de tareas, se puede diseñar un sistema de archivos en donde las distribuciones de las tareas queden en una partición o en un servidor diferente, y con las particiones restantes podemos instalar las aplicaciones vitales de tal forma que sea consistente con lo que necesitemos, lo cuál va adquiriendo sentido cuando nos enfrentamos a los problemas de seguridad. Es preferible dar al principio pocos permisos que demasiados. Ver figura 5.5

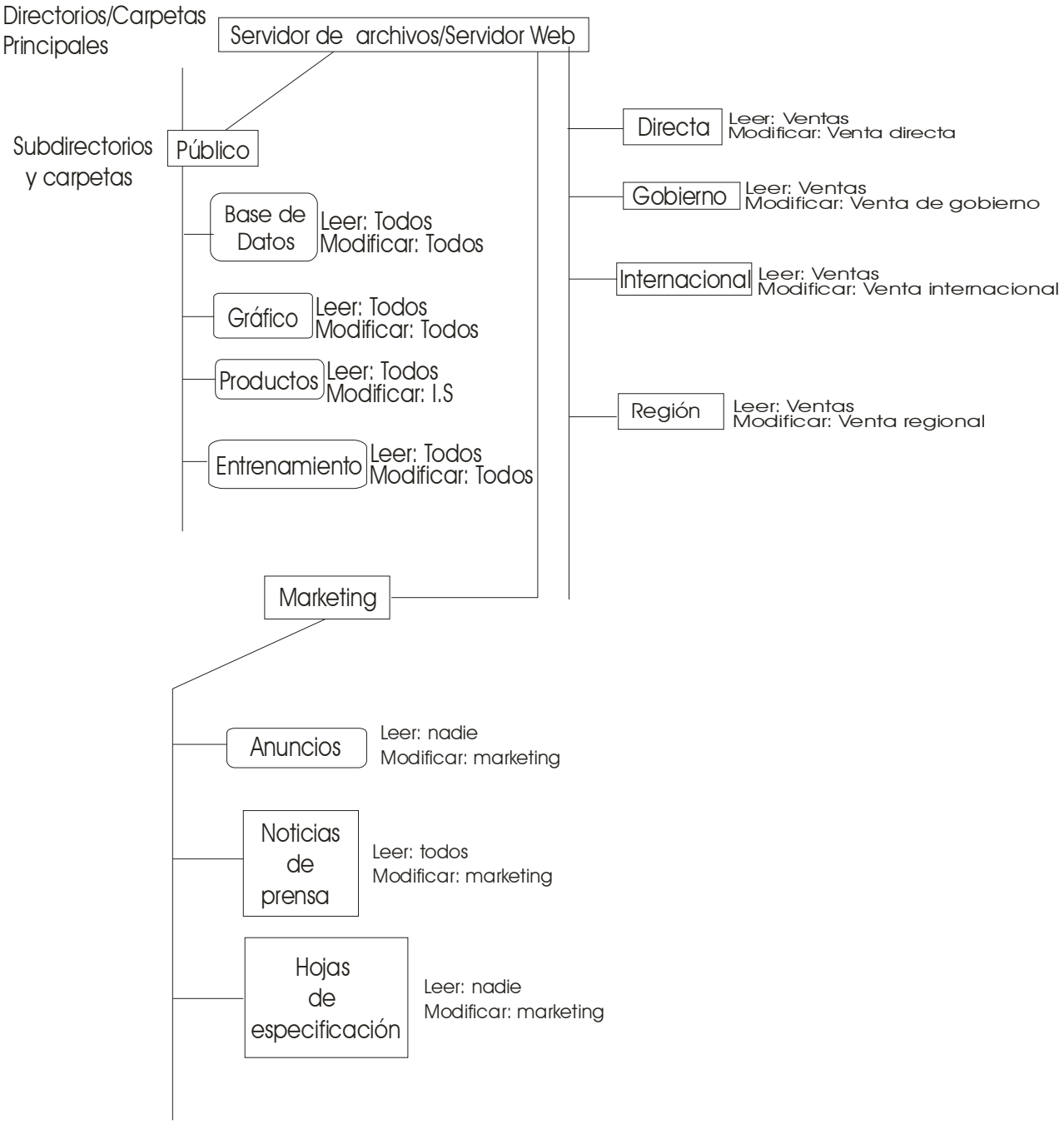


Figura 5.5 Modelo de niveles de acceso a un servidor.

- ✓ Accesos a niveles de grupo

Un grupo de trabajo puede ser un grupo de proyecto o departamento. Se puede dar a un grupo de trabajo su propia carpeta en un servidor Web, su propio disco en un servidor de archivos o su propio grupo en un servidor de noticias.

- ✓ Creación de un lugar central.

Tiene el propósito de almacenar la información relevante y actualizada que se va generando en la organización así como también ir eliminando la información que ya no es útil.

- ✓ El servidor Web.

El servidor Web hay que organizarlo de tal forma que haya niveles de acceso dependiendo de los privilegios otorgados en la organización, cuando tenemos un servidor Web es muy común que lo utilicemos para comunicarnos con Internet cuando hay empleados remotos que se comunican con su Lap-Top desde el avión a la compañía. Aquí es donde debemos poner especial cuidado en la seguridad pues hay personas externas al sistema que pudieran robarse información valiosa de la compañía, como por ejemplo los puntos débiles de un producto, un secreto industrial. Se puede prevenir este tipo de errores configurando cuidadosamente la estructura de carpetas y derechos de acceso. Una técnica efectiva es tener dos carpetas principales, una para uso interno y otra para uso externo.

En un artículo de la Jornada del día 10 de Junio de 2001. Titulado: ***La cultura de los soplones***. El Gobierno de los Estados Unidos a través del FBI está merodeando las computadoras y los servidores en todo el mundo con el argumento de que están vigilando a los criminales que utilizan Internet. Este sistema que antes espiaba milicias ahora espía a los globalifóbicos. Pero al igual que espía a opositores al régimen vigila a personas, a las grandes empresas e instituciones científicas para robarse secretos e incluso enterarse de las oportunidades de negocios multimillonarios que hacen otras empresas para ofrecérselas a las multinacionales estadounidenses. El truco consiste en conectar físicamente las computadoras espías en oficinas del servidor/proveedor de Internet y se conecta a la computadora del proveedor y hace un download (Copiado) de todo lo que está ahí guardado, a este sistema de espionaje se le llama *Carnívoro*. El otro es el sistema es *Echelon*, es global e incluye satélites, flotas de aviones militares, submarinos y otros artefactos.

- ✓ Servidor FTP.

La configuración de acceso al servidor FTP es similar al del servidor Web. Hay que asegurarse de que no se permite el acceso a personas externas a la información restringida.

- ✓ Planificación.

En la planificación es importante tomar en cuenta:

- ✓ La finalidad de la Intranet.
- ✓ Los usuarios.
- ✓ Su diseño virtual.
- ✓ La cultura de la organización.
- ✓ ¿Para qué la queremos?
- ✓ ¿Cómo vamos a utilizarla?
- ✓ ¿Quién la va a diseñar?
- ✓ ¿Quién la va a gestionar?

- ✓ ¿Quién va a proporcionar y actualizar el contenido?
- ✓ ¿Cuándo va a estar operativa?
- ✓ ¿Cuánto va a costar?

Una Intranet no tiene por que estar totalmente formada, básicamente por su naturaleza, es fácilmente escalable. La funcionalidad puede crecer a medida que se vayan presentando las necesidades. La gran ventaja radica en la adaptabilidad a las circunstancias específicas de la organización. El costo de la Intranet básicamente depende del alcance y de las características. Las Intranets son particularmente útiles para solucionar problemas tales como la velocidad de proceso, comunicación y el trabajo en quipo obstaculizado por distancias o métodos de comunicación engorrosos. Es muy útil contar con un equipo multidisciplinario con al menos un especialista de software, a una persona motivada por cada departamento o división de afectados y un comunicador que facilite la creación de un proceso electrónico. Debemos de pensar en la audiencia pues siempre habrá novatos y usuarios expertos, al tomar en cuenta a estos dos sectores hacemos de los usuarios que tengan el hábito de investigar y de involucrarse más en el sistema, hay veces que los usuarios son experimentados pero la disposición del sistema no cuenta con una organización universal que lo haga accesible por ejemplo íconos básicos con terminología poco común, procesos engorrosos de instalación, configuración y desarrollo de un proyecto. Un sistema como estos genera desorden, escasa organización y poco interés de los usuarios que no logran familiarizarse con el sistema.

Se debe considerar un entorno para usuarios discapacitados, tales como facilitar el software para letras grandes en la pantalla.

Conviene emplear grupos de enfoque que comprendan una amplia gama de usuarios para descubrir sus actitudes, habilidades y preferencias. Lo que puede parecer un diseño lógico para los desarrolladores de software puede dejar fríos a los usuarios. Las Intranets han de adaptarse a los usuarios.

Diagrama de casos de uso de alto nivel.

A continuación se mostrará el diagrama de casos de uso de alto nivel para una red LAN. Se trata de distribuir responsabilidades que nos faciliten el diseño del sistema ver Figura 5.6

Este caso de uso se utiliza para procesar las calificaciones que capturan los profesores de una escuela preparatoria.

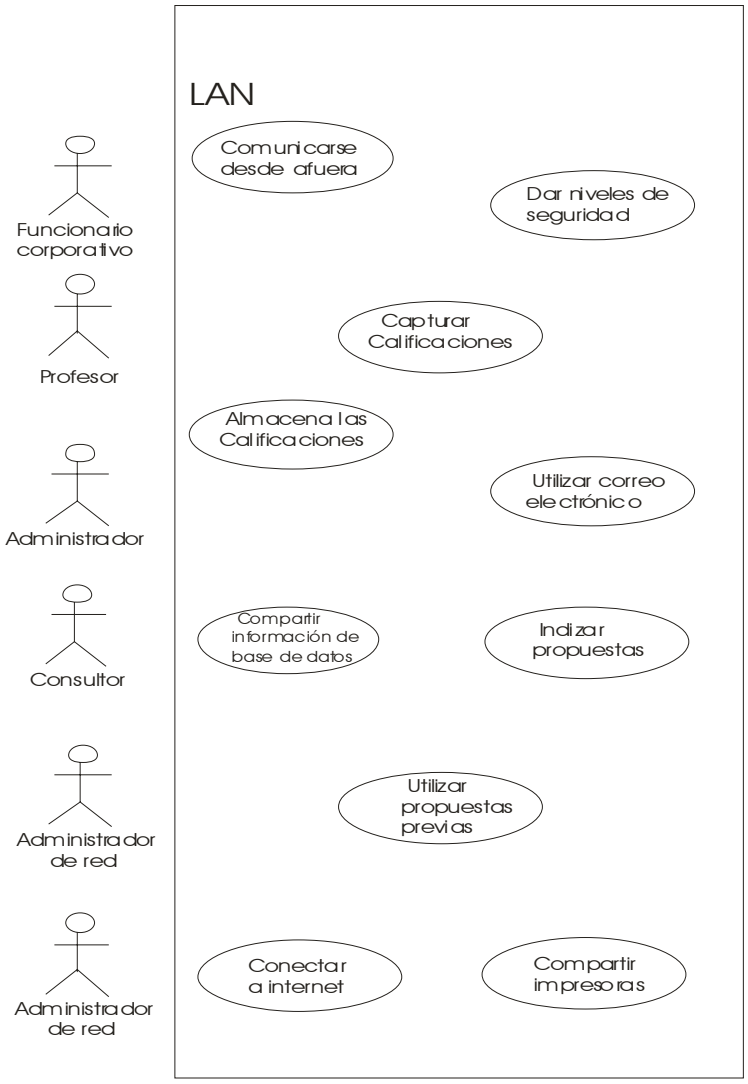


Figura 5.6. Procesamiento de las calificaciones de los alumnos de una preparatoria.

Diagrama de estados.

Los diagramas de estados, se utilizan para representar la situación que guarda un sistema en un tiempo determinado. Por ejemplo, cuando encendemos una computadora, se realiza la secuencia de arranque de la ROM y seguidamente se carga el sistema operativo en memoria, a este estado se le conoce como de Inicialización, realizado esto, el sistema pasa al siguiente estado que es el de Operación donde uno ya puede disponer del sistema y realizar alguna actividad de Interés, cuando nos disponemos a apagar el quipo pasa al estado de apagado. Ver figura 5.7

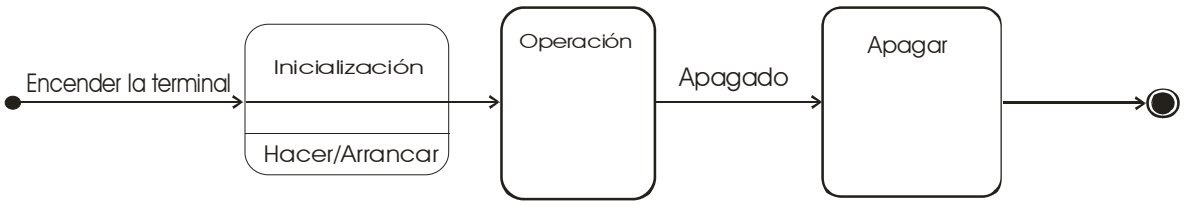


Figura 5.7. Diagrama de estados del proceso Encendido/Apagado

En un sistema de información son inevitables los sistemas concurrentes, es decir cuando un sistema modifica su estado, también suceden otras acciones en el mismo sistema que modifican su estado.

Los diagramas de estados, son una técnica para describir por etapas el comportamiento de un sistema. Se describen todos los estados posibles en los que puede entrar un objeto particular y la manera en que cambia el estado del objeto, como resultado de los eventos que llegan a él.

Diagrama de secuencias.

En UML un diagrama de secuencias muestra la forma en que se comunican los objetos entre sí al paso del tiempo, esto se muestra por lo común, paso a paso. Vamos a realizar un diagrama de secuencias enumerando las responsabilidades de un administrador de red.

Estas son las responsabilidades de un administrador de red:

Responsabilidades de hardware.

- ✓ Verificar la correcta instalación del hardware.
- ✓ Comprobar el estado de los periféricos y ser capaz de buscar el fallo en caso de error de la instalación.
- ✓ Instalar nuevos dispositivos de hardware (Memoria, discos, terminales, etc.).

Responsabilidades de software.

- ✓ Instalar el sistema operativo y configurarlo.
- ✓ Crear y mantener los sistemas de archivos, detectando y corrigiendo los posibles errores que puedan producirse.
- ✓ Controlar la utilización de este sistema de archivos y su crecimiento.
- ✓ Diseñar e implementar las rutinas para realizar copias de seguridad así como para su posterior recuperación.
- ✓ Configurar y mantener el software de cualquier dispositivo: impresoras, módem, tarjetas de red, etc.
- ✓ Actualizar el sistema operativo en caso en caso de poseer una versión más moderna.
- ✓ Instalar el software de cualquier aplicación (X Window, base de datos, procesadores de texto, etc.)

Responsabilidad sobre los usuarios

- ✓ Añadir nuevos usuarios y dar de baja a los que ya no se conectan al sistema.
- ✓ Permitir el acceso a los usuarios de forma controlada.
- ✓ Evaluar las necesidades en cuanto a equipos se refiere. Determinar si es necesario añadir nuevos discos , impresoras, memorias, etc. Con objeto de que los usuarios encuentren un entorno agradable de trabajo.
- ✓ Proporcionar asistencia a cada una de las personas.
- ✓ Tener a los usuarios informados en todo momento de los posibles nuevos servicios y sus características.

Seguridad

- ✓ El administrador del sistema tiene acceso sin restricciones a todos los recursos.
- ✓ No emplear privilegios de administrador en forma prolongada. Los errores pueden tener contraseñas fatídicas.
- ✓ La palabra clave del administrador debe de mantenerse estrictamente en secreto.
- ✓ Vigilar la cantidad de accesos erróneos producidos en el sistema.
- ✓ La política de seguridad debe estar perfectamente definida, siempre que los mecanismos de seguridad del sistema lo permitan.
- ✓ Vigilar estrechamente a los usuarios potencialmente peligrosos.

Vamos a revisar las responsabilidades de software de un administrador de red con UML a través de un diagrama de secuencias Ver 5.9

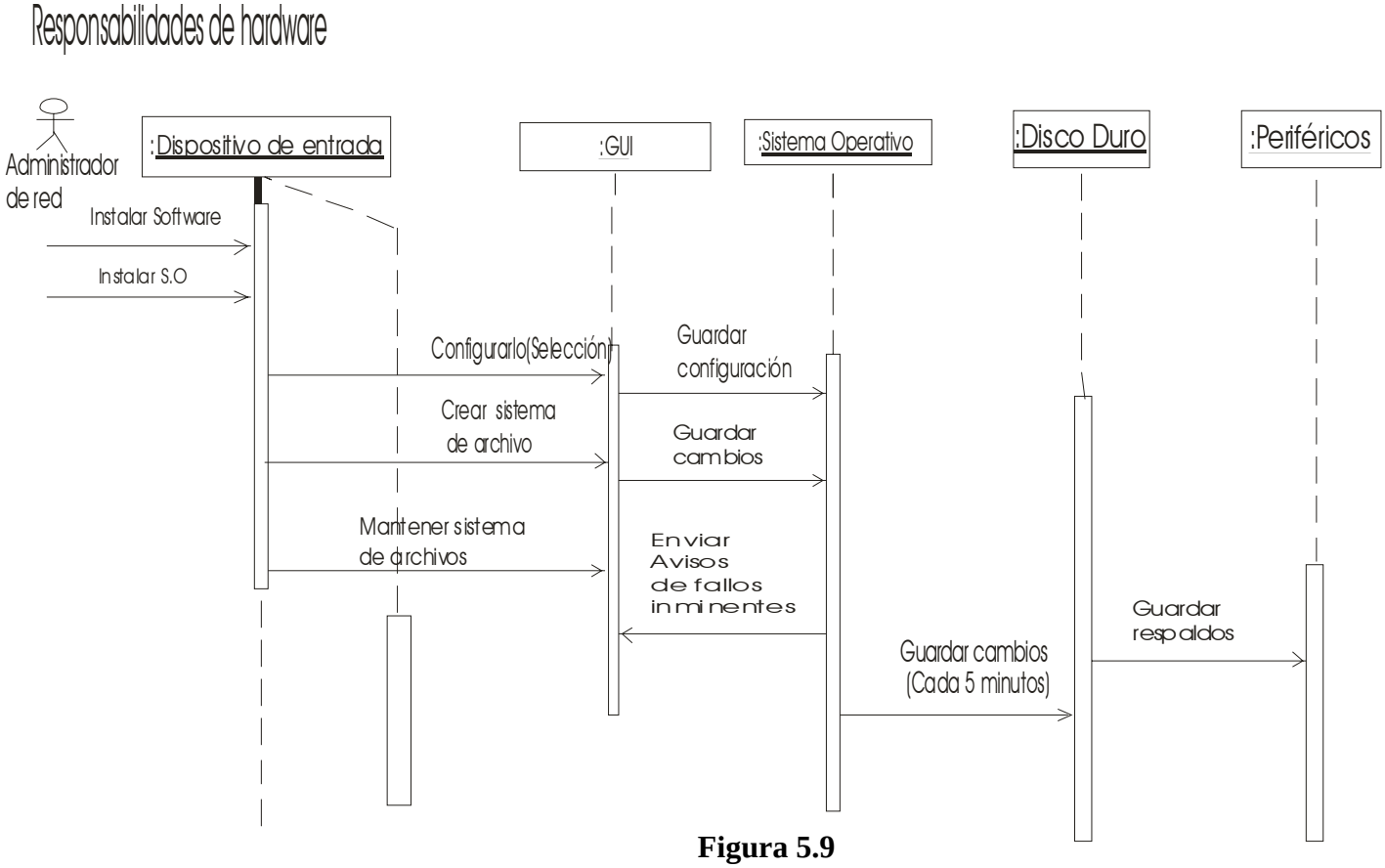


Figura 5.9

Diagrama de colaboraciones.

Los diagramas de colaboraciones muestran los mensajes que los objetos se envían entre sí, y a diferencia de los diagramas de secuencias que se organiza de acuerdo al tiempo, el diagrama de colaboraciones se organiza en función del contexto, es decir, en función del espacio. He aquí algunos ejemplos:

En nuestra base de datos, el profesor envía un mensaje a los estudiantes:

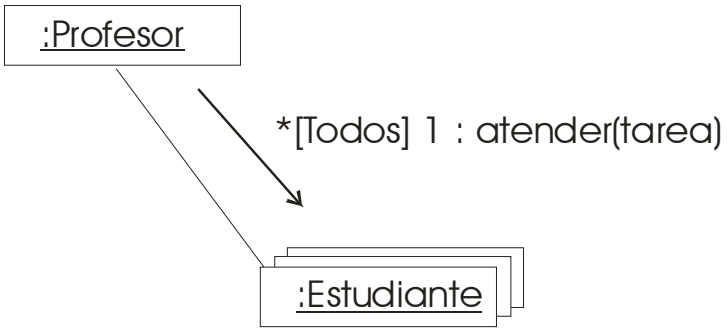


Figura 5.10

Vamos a observar otro ejemplo, en este caso observamos un objeto que controla a otros objetos específicos. Este objeto activo puede enviar mensajes a los objetos pasivos e interactuar con otros objetos activos, tales objetos activos se resaltan con un borde grueso. Ver figura 5.11.

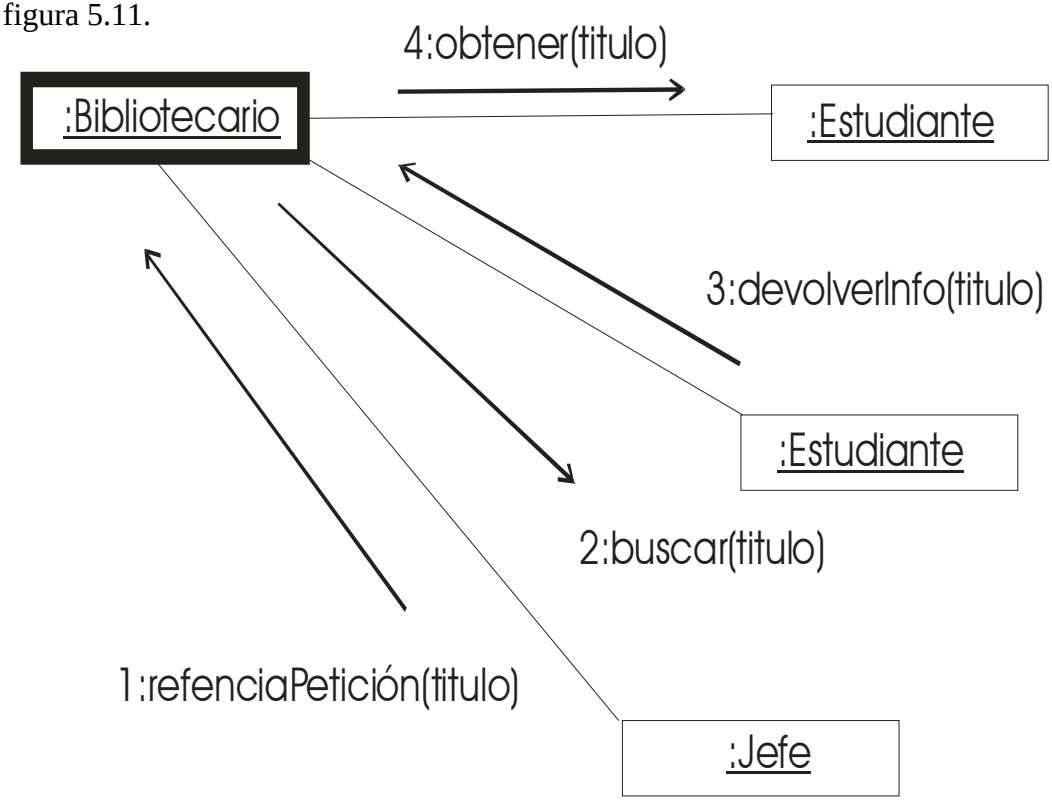


Figura 5.11

Diagrama de emplazamiento.

Es aquél que muestra las relaciones entre los componentes de hardware y de software en un sistema. Cada nodo de un diagrama de emplazamiento representa alguna clase de unidad de cómputo; en la mayoría de los casos se trata de una pieza de hardware. El hardware puede ser un dispositivo o un sensor simple.

Los componentes en un diagrama de emplazamiento representan módulos físicos de código.

La mayoría de los encargados de diseñar sistemas realizan bocetos informales en estas situaciones, en general no hay ningún problema en este tipo de bocetos, pero cuando se trata de representar sistemas distribuidos, se requiere un poco más de formalidad, este tipo de diagramas orientado a objetos es una muy buena solución.

A medida que se tiene que lidiar con sistemas distribuidos, se tiene que pensar en la posibilidad de utilizar diagramas de emplazamiento.

Así es como representamos un nodo de nuestro diagrama:

Figura 5.12

El siguiente ejemplo tenemos un nodo con componentes que están interrelacionados, hay un componente que realiza la búsqueda en función de la llave que se nos proporcione, ya sea que busquemos en la base de datos de los alumnos o de los trabajadores, hecho esto se despliega el resultado de la búsqueda desde el servidor. Ver figura 5.13:

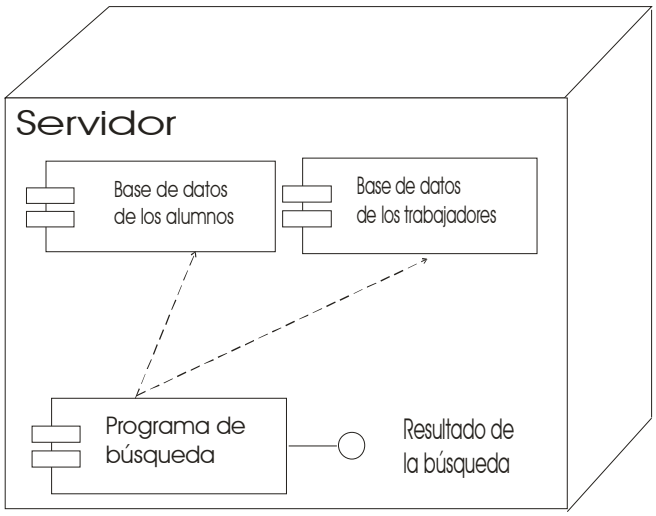


Figura 5.13

La siguiente figura muestra la relación entre dos nodos, es decir, entre un servidor y una terminal en un diagrama de emplazamiento y cuatro componentes que se relacionan del siguiente modo. El programa de presentación es el que va a interactuar con el usuario, cuando este le haga una solicitud. Cuando el usuario haya hecho la solicitud, la terminal realiza una petición al servidor de bases de datos, este le da la respuesta a través del resultado de la búsqueda. Cuando el resultado de la búsqueda esté cargado en el servidor, este se visualizar en el programa de presentación. Ver figura 5.14.

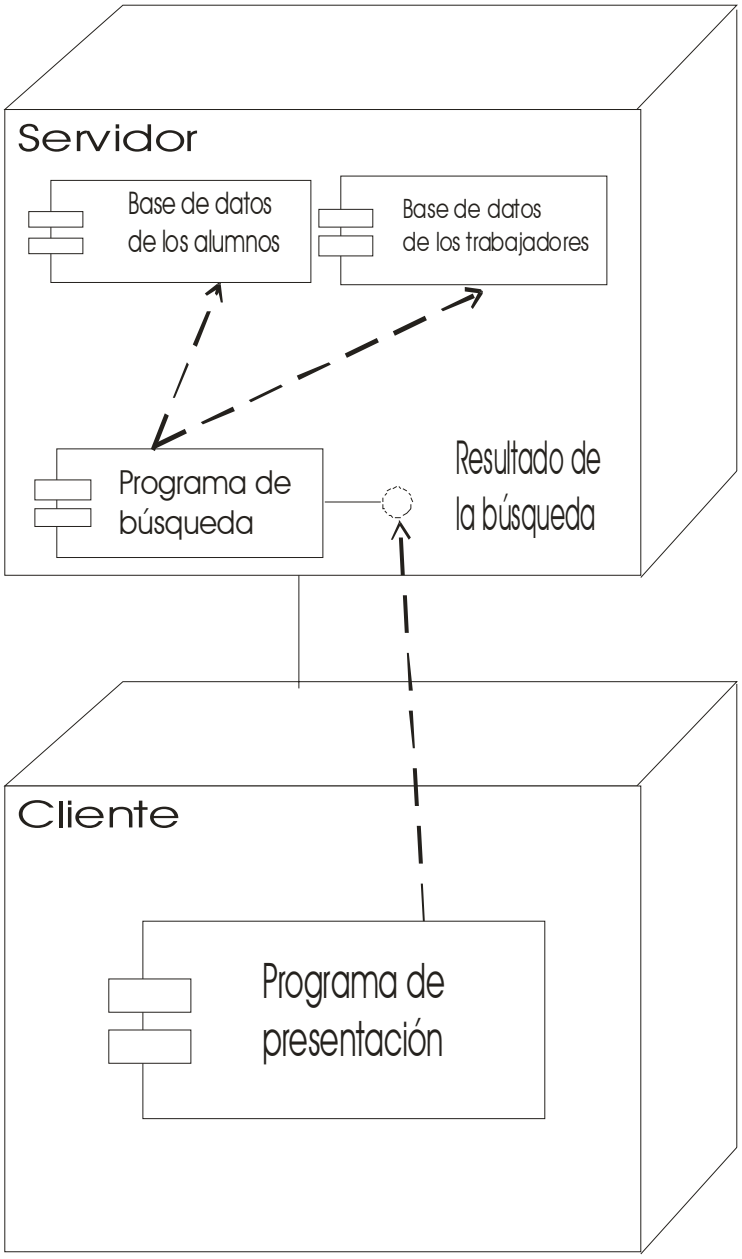


Figura 5.14. En el programa de presentación se muestran los resultados de la búsqueda de la base de datos de los alumnos o de la base de datos de los trabajadores, petición que realiza la terminal al servidor donde están alojadas las bases de datos de los alumnos y de los trabajadores.

Con esta forma tan sencilla se pueden representar procesos distribuidos, la siguiente figura es un diagrama de emplazamiento de una red ARCnet.Ver figura 5.15.

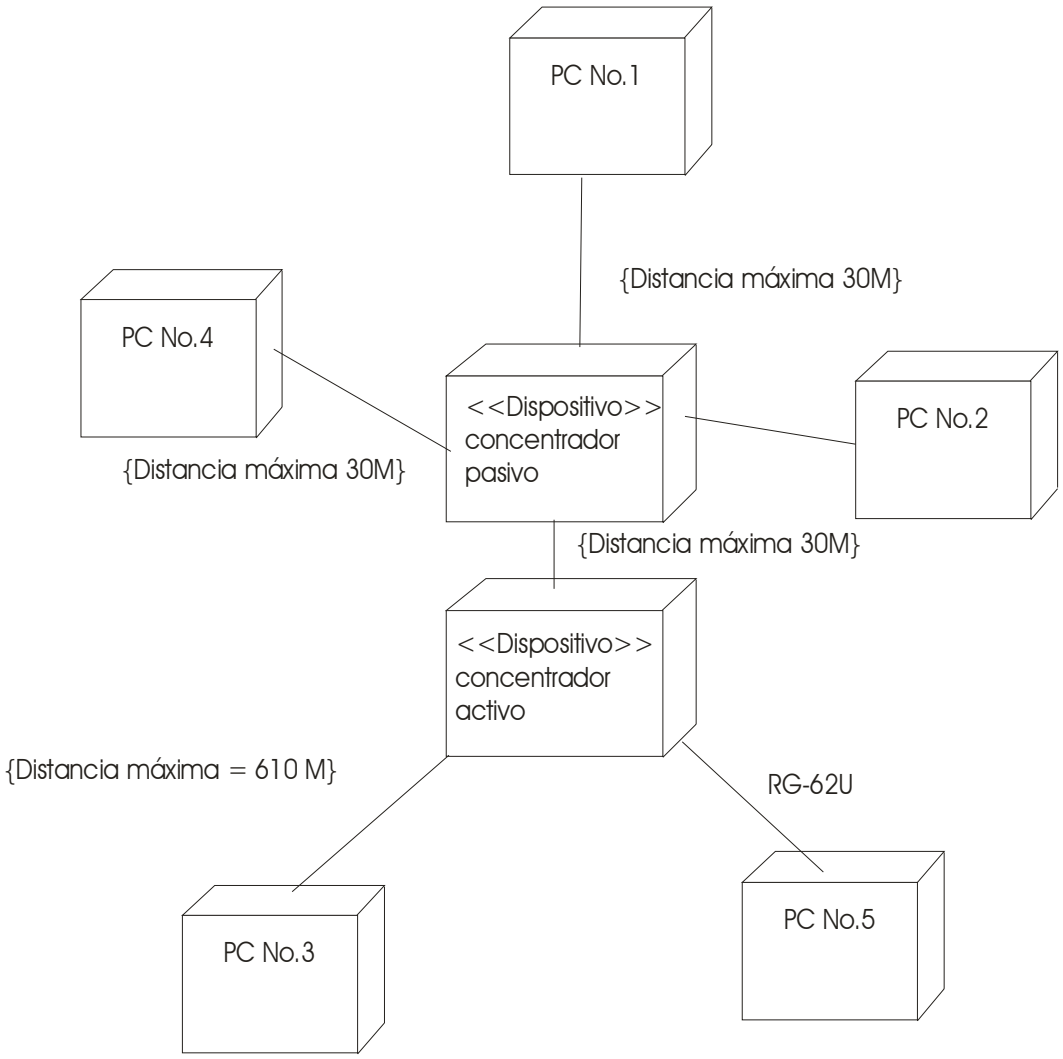


Figura.5.15. diagrama de emplazamiento de una red ARCnet.

El siguiente modelo es la representación de un sistema de información en una escuela que tiene contacto con el servidor de dirección general, a su vez esta entidad tiene su propio servidor con sus componentes. Figura 5.16

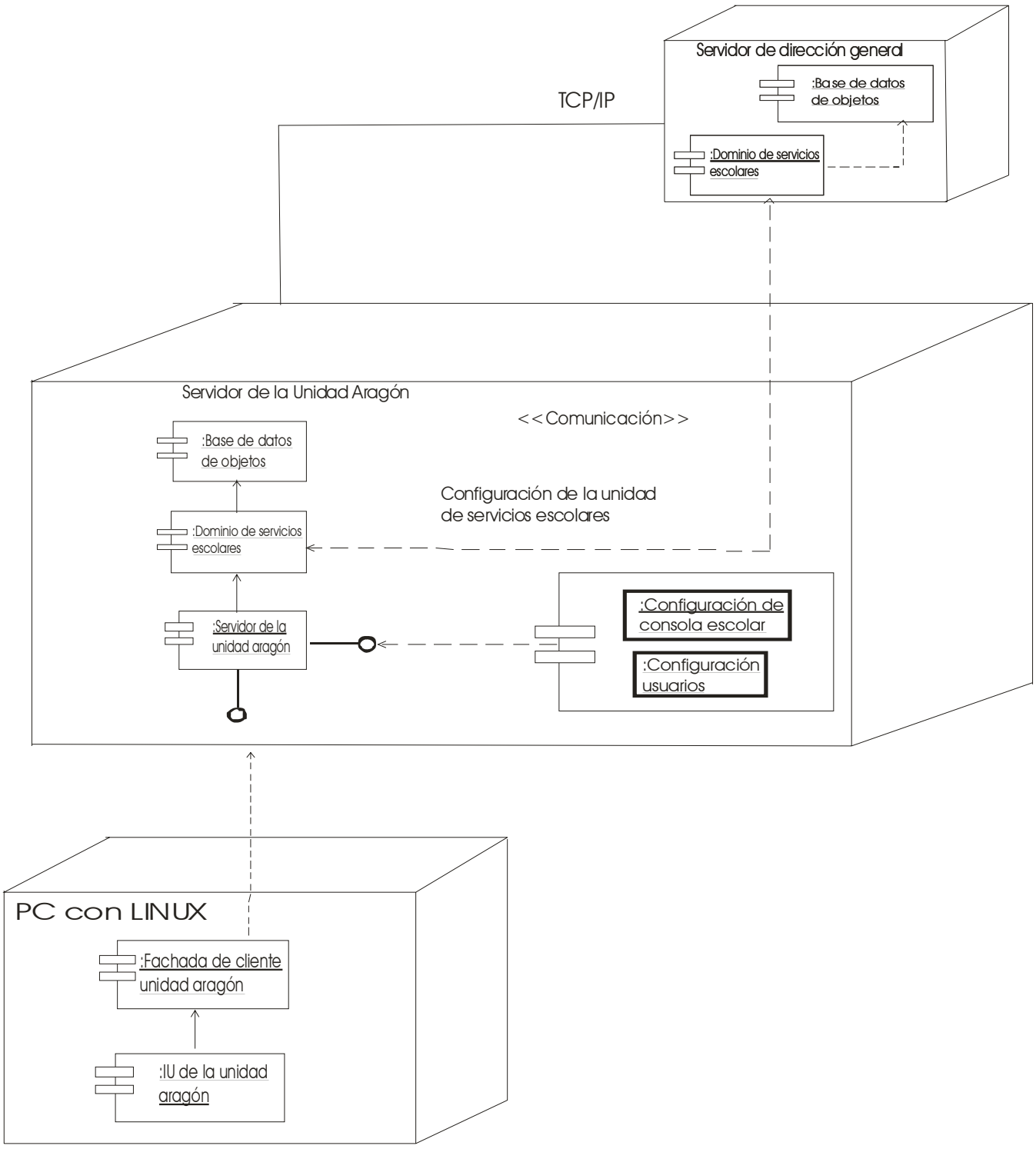


Figura 5.16

En la figura podemos notar que hay tres nodos, esto es, dos servidores de bases de datos y una terminal con sus componentes correspondientes.

En el primer nodo, los componentes tienen la finalidad de facilitar la administración de los datos, desde la terminal PC, la fachada del cliente y la UI de la unidad aragón interactúan, aunque en el diagrama se puede apreciar que la fachada de cliente de la unidad aragón depende directamente de la UI de la unidad aragón. Con estos dos componentes del nodo PC se pueden hacer peticiones al servidor de la Unidad Aragón.

En el servidor de la Unidad Aragón, se encuentran todas las bases de datos con todos los tipos de información distribuida, representado por el componente de las bases de datos de objetos, en el otro componente se encuentra la base de datos de los dominios de servicios escolares, por ,medio de estos se pueden realizar peticiones a servidores externos como por ejemplo, el servidor externo que pertenece a la Dirección General, por medio de una comunicación con protocolo TCP/IP podemos mandar toda la información que se requiera o que nosotros necesitemos, como por ejemplo, las calificaciones de los alumnos, trámites vía Internet para obtener historiales académicos, inscripciones, revisiones de estudios, y un gran etcétera.

Podemos representar cualquier sistema de información (Ver capítulo III página 28), utilizando diagramas de emplazamiento, ver figura 5.16, este último diagrama es más amigable para el programador, es sencillo pero con la investigación que se ha hecho en este trabajo, este tipo de diagramas no se utilizan frecuentemente para diseñar la implementación física de los sistemas, casi siempre se realizan bocetos informales.

Capítulo VI. Programación en XML.

6.1. Diseño de hojas de estilo en HTML.

La organización de los archivos de un sitio Web es similar a la forma en que los sistemas operativos organizan los archivos, es decir; en forma de árboles invertidos. Ahora bién cuando localizamos una página Web desde nuestro navegador, en la barra de direcciones de nuestro navegador nuestra terminal le hace una petición a un servidor Web, si nos da una respuesta afirmativa, visualizaremos la página solicitada y su correspondiente ruta absoluta tal como: http://mx.geocities.com/raul_xut_lcp/index.html a este tipo de rutas absolutas se les llama URL (Localizador Uniforme de Recursos). Ver figura6.1.

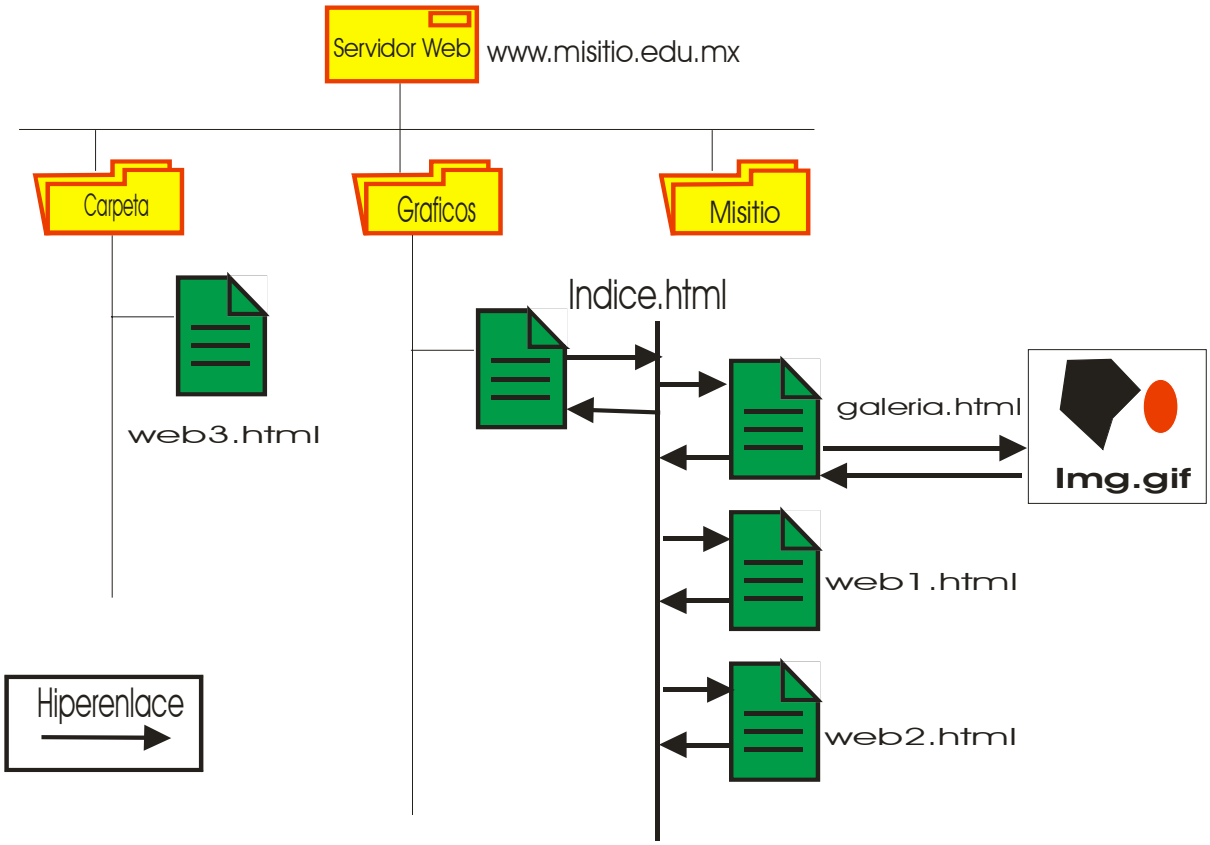


Figura 6.1. Organización de los archivos en un servidor Web.

Nombre del archivo	URL absoluto(desde raíz /)	URL(desde Indice.html)
Web3.html	www.misitio.edu.mx/Carpeta/web3.html	Carpeta/web3.html
Indice.html	www.misitio.edu.mx/Graficos/Indice.html	Indice.html

Tabla 6.1 tabla de rutas absolutas y relativas.

Como se puede apreciar, las rutas absolutas empiezan desde el directorio raíz hasta el archivo destino, y las rutas relativas son aquellas que empiezan desde un archivo origen hasta un archivo destino. En la tabla 6.1 hay dos ejemplos, uno de ruta absoluta y otro de

ruta relativa de un archivo: www.misitio.edu.mx/Graficos/Indice.html todos los navegadores visualizarán la dirección completa (URL) como:

http://www.misitio.edu.mx/Graficos/Indice.html

Donde:

http:	→ Sirve para realizar una petición de red a un servidor Web.
//	→ Indica que a continuación viene el nombre de una máquina.
www	→ identifica al servidor web.
misitio	→ Es el nombre del servidor.
edu	→ tipo de dominio.
mx	→ dominio del lugar en donde está ubicado el servidor Web.
/Graficos/	→ Ruta de acceso.
Indice.html	→ Nombre del archivo.

6.2. Creación de contenidos.

Para crear los contenidos de un sitio Web bién consolidado necesitamos contar con las herramientas office, las herramientas WYSIWYG (What You See Is What You Get), estas herramientas permiten crear aplicaciones Web en forma sencilla sin necesidad de ser un programador Web experto. Este tipo de aplicaciones se utilizan frecuentemente por los empleados de los departamentos que se encargan de hacer el registro de las actividades diarias y con esto realizar los reportes de actividades que van a la dirección para la toma de decisiones utilizando la tecnología de Internet donde se realiza de manera más rápida y efectiva la organización y el flujo de los datos, así como el nivel de acceso a la información por las personas involucradas en el **Sistema de Información,** Las herramientas gestoras de gráficos, gestores de bases de datos, herramientas de programación, herramientas de programación simbólica UML.

El Software libre.

También llamado código abierto, es un tipo de software desarrollado por una comunidad de programadores independientes, permite ser modificado pues viene con el código fuente, el software libre se está expandiendo en forma impresionante, un ejemplo de ello es LINUX, un sistema operativo que es muy potente, es robusto y seguro, es muy común ver a este software en los servidores de Internet. Entre las herramientas para **desarrollar contenidos** que se ubican en esta categoría se encuentran las siguientes:

- ✓ Herramientas de desarrollo de Bases de Datos: MySQL, PHP, Postgre entre otras.
- ✓ Herramientas de desarrollo de páginas Web: Quanta de LINUX.
- ✓ Office de LINUX.
- ✓ Servidores Web: El clásico y muy difundido servidor Apache.
- ✓ Herramientas gráficas gratuitas incluidas en LINUX.

A continuación vamos a desarrollar contenidos Web para una Intranet sencilla y repasaremos algunos conceptos.

- ✓ Hojas de estilo en cascada.

La etiqueta STYLE.

Sintaxis.

<style>

etiqueta1 {prop11: valor12; prop12:valor12; prop13:valor13;…}

etiqueta2 {prop21: valor21; prop22:valor22; prop23:valor23;…}

etiqueta3 {prop31: valor31; prop32:valor32; prop33:valor33;…}

.
.
.

etiquetan {propn1:valorn1;propn2:valorn2;propn3:valorn3; … }

</style>

Hay tres formas de aplicar hojas de estilo: Localmente, internamente y externamente. Se puede utilizar uno o cualquiera de los tres métodos.

Creando una hoja de estilo interna.

1.- Entre las etiquetas <head > y </head> abrir la etiqueta <style>

2.- A partir de este momento defina las propiedades de las etiquetas

etiqueta1 {prop11: valor12; prop12:valor12; prop13:valor13;…}

etiqueta2 {prop21: valor21; prop22:valor22; prop23:valor23;…}

etiqueta3 {prop31: valor31; prop32:valor32; prop33:valor33;…}

.
.
.

etiquetan {propn1:valorn1;propn2:valorn2;propn3:valorn3; … }

3.- Ya definidas las propiedades de la hoja de estilo, cierre la etiqueta de estilo </style>

4.- Empiece a crear el contenido de la hoja.

Cómo definir estilos para las clases.

1. En la sección de style escriba etiqueta.nombreclase {propiedad1:valor1, propiedad2:valor2,…}
2. Añadir en los elementos de Interés de la página HTML **class=nombreclase**. Ver listado 6.1

He aquí un listado de nuestra primera hoja de estilo$_1$:

```
<html><head><title>Crear una hoja de estilo
interna</title>
<style>
h1 {text-align:center;letter-spacing:.1cm;background:#B91A1A;
color:#ffffff;font:normal 20pt “Times New Roman”, “Calisto MT”}
p{text-align:justify;text-indent:8pt;font: 10pt/15pt “Arial”, “Verdana”}
p.intro {text-indent:0;font: italic bold}
p.intro EM {font-style:normal}
a:link {background:yellow;color:green;text-decoration:none}
a:visited {background:orange; color:yellow;text-decoration:none}
a:hover {background:red;color:white;text-decoration:none}
a:active {background:white;color:black;text-decoration:none}
</style>
</head><body>
<p class=”intro”> Variante troyana instala en la computadora dispositivo que permite robo
de datos
<h1>Nuevo virus se activa con solo visitar la red</h1>
<p> <em>Whashington, 25 de Junio</em>. Un misterioso virus inform&aacute;tico infecta
desde populares sitios web las computadoras de los visitantes, quienes,
sin darse cuenta, descargan programas que facilitan el robo de informaci&oacute;n por
<em>hackers</em>, alertaron expertos. Al contrario
de los virus que se propagan por correo electr&oacute;nico
esta variante llamada Scob, Download.Ject, Toofer o Webber.P. se transmite al visitar un

sitio afectado y opera mediante la instalación de un dispositivo llamado

troyano que permite a los piratas cibern&eacute;ticos acceder a las computadoras.

Los usuarios deben estar alertas de cualquier sitio web, inclusive aquellos de confianza
del internauta, ya que puede estar afectado
por este dispositivo y, por tanto, contener ese c&oacute;digo potencialmente maligno,
indic&oacute; este Jueves la Sociedad de Emergencia de Equipo
Inform&aacute;tico(<a href=”www.cert.com/”>CERT</a>, por sus siglas en
ingl&eacute;s),fundada por el gobierno estadounidense. Scob afecta a los sitios que tienen

programas de Microsoft IIS 5.0, agregaron los expertos.<br><br>
“Si los usuarios de Internet explorer visitan algunas p&aacute;ginas
Web afectadas por Scob, su computadora podr&iacute;a intentar bajar un archivo de
un sitio de internet ruso”,
indic&oacute; este Viernes
la firma de seguridad <a href=”www.sophos.com/”>Sophos.</a>La empresa de seguridad
LURHQ

indic&oacute; que el

programa troyano tiene por objetivo el robo de claves e informaci&oacute;n financiera.

El troyano

supuestamente fu&eacute; dise&ntilde;ado con el prop&oacute;sito de robar detalles

fiancieros y de otro tipo del
```

usuario infectado”, agregó.

Microsoft calificó el incidente de crítico, y urgió a los usuarios
a bajar filtros actualizados para proteger sus sistemas.

“Un gran número de sitios Web, algunos de ellos bastante populares, se vieron involucrados en la
distribuci´n de este código maligno a comienzos de la semana” indicó

el Centro de Internet

Storm, del SANS Institute, un emprendimiento conjunto de investigadores privados y universidades. SANS

no nombró los sitios web involucrados.

Panda software añadió que el peligro de este virus es que es difícil

de reconocer, ya que no despliega nigún

mensaje o advertencia que indique que está presente en la computadora”.
Patric Hinojosa, jefe de tecnología de Panda, dijo que aún no se sabe cuántas computadoras están infectadas, pero que se tendrá una mejor idea en los próximos días. Hinojosa y otros expertos dijeron que los usuarios pueden protegerse desactivando la
función « javascript » del internet explorer.
</body>
</html>

Listado 6.1

Al ejecutar el programa se visualiza del siguiente modo:

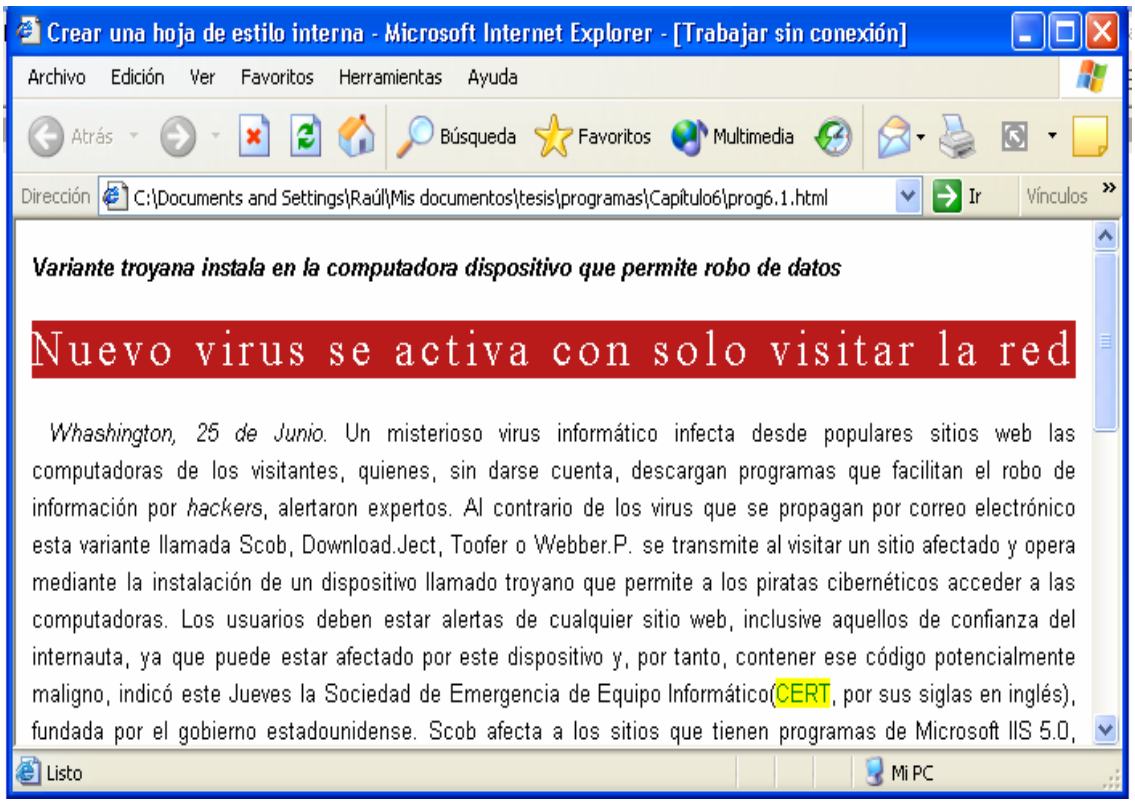


Figura 6.2

Cómo crear una hoja de estilo externa.

Las hojas de estilo externas son ideales para otorgar un aspecto común a todas las páginas de un sitio Web. En lugar de crear sitios a partir de hojas de estilos individuales, es posible indicar a cada página que consulte la hoja externa, asegurando así que todas tendrán los mismos ajustes.

A continuación se muestran los pasos:

1. Crear un nuevo documento.
2. Dentro de **<style>** escribir todas las etiquetas y sus propiedades a definir entre { … }
3. Defina tantas propiedades como desee.

etiqueta1 {propiedad1:valor1, propiedad2:valor2, propiedad3:valor3,…}

4. Guarde el documento en formato de sólo texto en el directorio deseado. Asigne al documento la extensión .css para indicar que es una hoja de estilo en cascada.

Cómo usar una hoja de estilo externa

Para utilizar una hoja de estilo externa:

1. En la sección head de todas y cada uno de las páginas Web en las cuales desee utilizar la hoja de estilo, escriba:

link rel = hojaestilo type=”text/css”

2. Escriba href = “url.css” donde url.css es el nombre del documento con todos los atributos del documento de estilo css.
3. Escriba > final.

Ejemplo:

a) Vamos a definir nuestra plantilla base en el bloc de notas tal y como se indica en la figura 6.3:

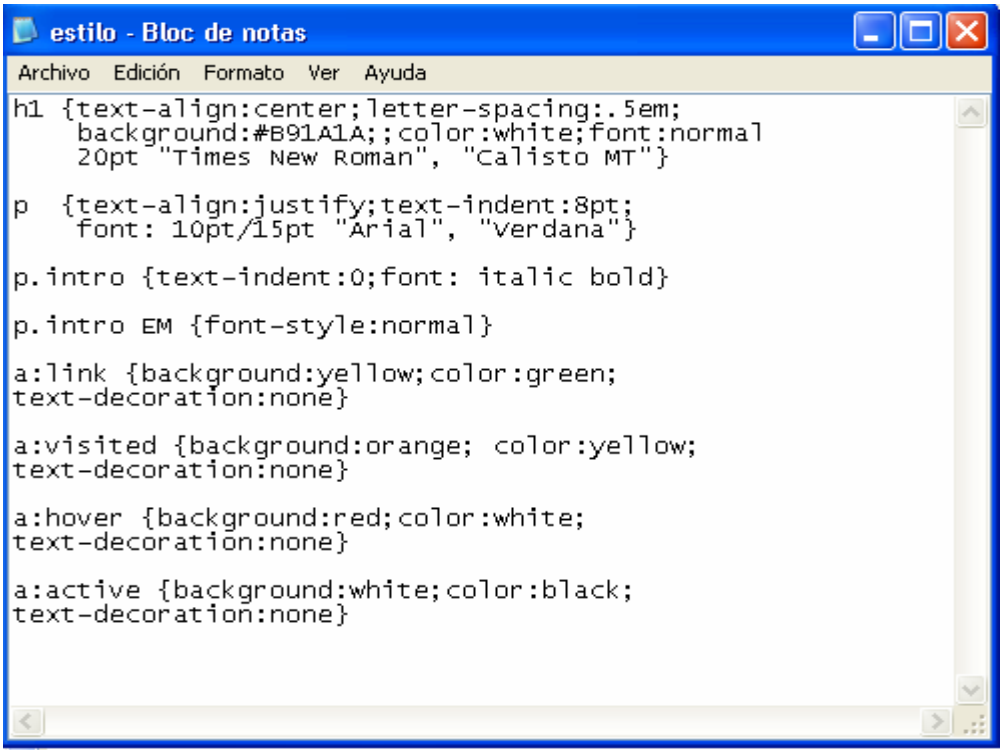


Figura 6.3

b) Guarde este documento con extensión .css

c) Ahora procedemos a realizar un documento en el bloc haciendo uso de la plantilla anterior.

Los estilos ofrecen muchas posibilidades que nunca podrán ofrecernos las etiquetas HTML y las extensiones.

6.3. Programación en XML.

El lenguaje de programación XML o Extensible Markup Languaje es un metalenguaje que permite diseñar un lenguaje con etiquetas propias, es decir, que permite diseñar etiquetas para múltiples clases de documentos. Es un lenguaje que no se basa en características

prefijadas como el HTML sino en habilitar el uso del SGML en la Web. SGML es el acrónimo de Standard Generalizad Markup Languaje, el estándar internacional para la definición de la estructura y el contenido de los diferentes tipos de documentos electrónicos.

Esto permite que las personas desarrollen sus propios lenguajes de etiquetas para intercambiar información específica sobre distintos campos como la música, electrónica, medicina, etc. Entre las ventajas podemos enumerar las siguientes:

- ✓ El lenguaje es de formato público, es decir, no es propiedad de ninguna compañía.
- ✓ La opción de diseñar etiquetas propias facilita la labor de diseño de tareas complejas para las que no existen etiquetas específicas.
- ✓ Las funciones de hipertexto son mejores que las del HTML actual.
- ✓ La información será más compatible ya que la flexibilidad de las etiquetas XML no tiene por que adaptarse a ningún navegador específico, bastará con que soporte XML.

A primera vista, una página escrita en XML es similar a una escrita en HTML, pero mientras HTML sólo utiliza 90 claves, las del XML son infinitas, ya que los autores de cada documento pueden inventarse las suyas. El problema es que los buscadores puedan interpretarlas, los documentos tienen que llevar un documento de definición (DTD), una especie de glosario. Esto no es necesario en HTML que tiene un DTD predefinido.

Para que un documento XML esté completo debe contar con:

- ✓ Consta de al menos un elemento.
- ✓ Todos los elementos están comprendidos dentro del elemento documento o raíz.
- ✓ Sus elementos están anidados correctamente.
- ✓ Todas las entidades que se utilizan han sido declaradas en el subconjunto interno de la DTD.

Las propiedades de nuestras etiquetas dependen de los recursos de hardware y software. Por ejemplo, si queremos crear unas etiquetas elementales en XML, podemos hacer uso de las propiedades de las hojas de estilo en cascada tal y como se indica a continuación:

Definiendo los atributos de las etiquetas XML.

```
acto{border:double medium green; text-align:center;
background:blue; font family:Tahoma;color:white;
font-size:16pt;width:100%}
```

Listado 6.2

Programa XML

```
<?xml version=”1.0” encoding=”ISO-8859-1”?>
<?xml-stylesheet type=”text/css” href=”uno.css”?>
```

```
<plantilla_inicial>
        <comunicado_1>
<acto> La libertad sin oportunidades es un regalo envenenado</acto>
        </comunicado_1>
</plantilla_inicial>
```

Listado 6.3

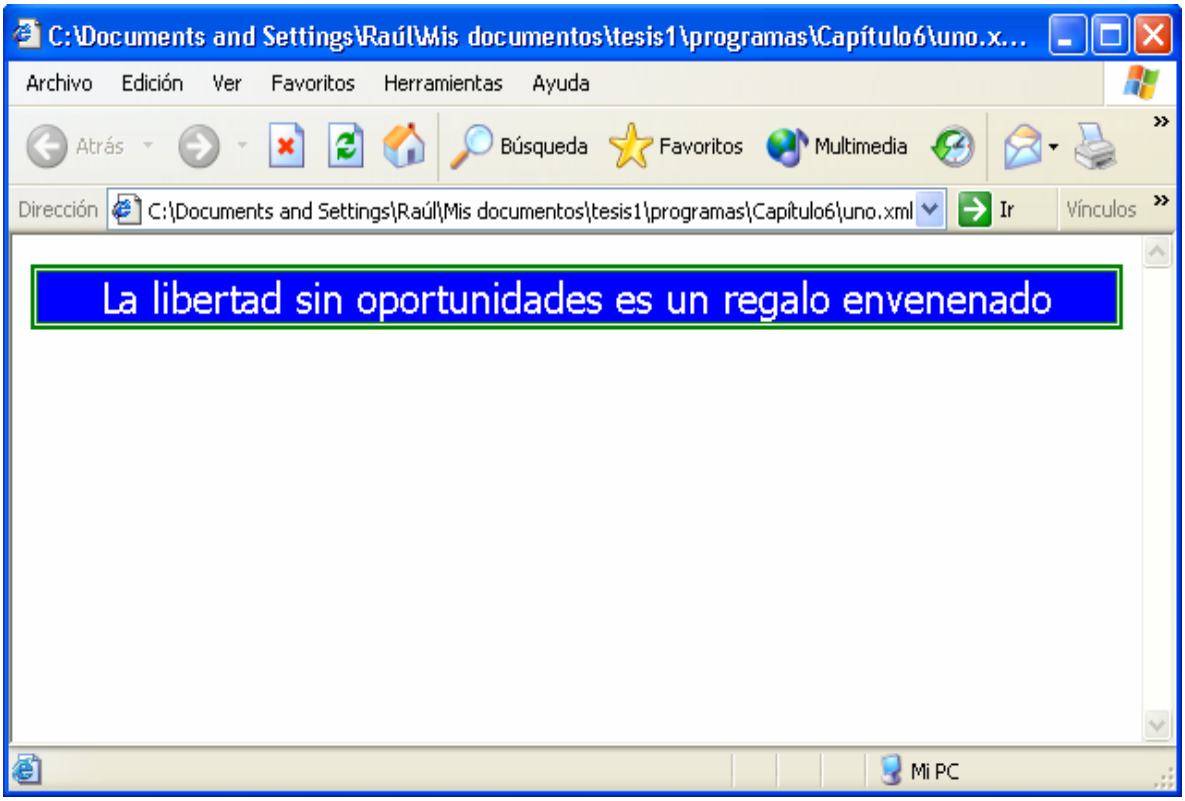


Figura 6.4

Las propiedades de las marcas que se determinaron en el documento DTD (Definición de tipo de documento) se utilizarán para precisar el tipo de documento. Todos los documentos de un mismo tipo compartirán una DTD común. Por ejemplo, todos los documentos de tipo informe se codificarán según la misma DTD, las cartas comerciales según otra, etc.

1. Los documentos que se escriben utilizando una DTD específica, se llaman instancias de esa DTD.
2. HTML es una aplicación del lenguaje SGML.
3. XML es un perfil de SGML y no una aplicación de SGML, como es el caso de HTML.
4. Al igual que sucede con SGML, en XML es posible crear DTD.

5. HTML sigue siendo el lenguaje idóneo para crear páginas web; xml se está utilizando para el intercambio de datos por internet. Se ha demostrado que ambos lenguajes cumplen con su función y que deben de utilizarse de manera conjunta.

Espacios de nombres. La especificación para espacios de nombres establece la posibilidad de incluir, en un mismo documento XML, marcas que proceden de distintos tipos de documentos (DTD o esquemas). Se encuentran en estado de recomendación desde 1999. Los espacios de nombres nos permiten combinar, en un mismo documento XML, marcas procedentes de dos DTD diferentes.

6.4. DOM (Document Object Model)

Es un modelo de documento orientado a objetos para documentos Web. El propósito de DOM es ofrecer una interfaz orientada a objetos para acceder y procesar documentos XML. DOM es un modelo de objetos independiente de cualquier lenguaje de programación. Entre las aplicaciones XML encontramos la UXF (UML eXchange Format), para el intercambio de modelos UML (Unified Modeling Language).

6.5. Granularidad.

Granularidad se refiere al nivel de detalle con el que destacamos o diferenciamos los contenidos informativos mediante la inclusión de marcas. Cuanto mayor sea el nivel de detalle con el que se intercalan las marcas, mayor será la ranularidad del marcado aplicado sobre el documento, y dispondremos de más posibilidades para procesar el documento y su contenido.

He aquí un ejemplo:

Aquí se definen los atributos de las etiquetas, en un documento con extensión .css

titulo1{top:60;left:40;width:50%;display:block;border:medium none silver thin dashed; background-color:yellow; padding:2px; margin-left:”25px”; text-indent:”20”; font-size:20;color:black;text-align:center}

barravertical { position:absolute; top:50; left:20; height:230; width:30;background-color:#00CCFF;}

barravertical2 { position:absolute; top:170; left:750; height:500; width:30;background-color:teal;}

posicion1 { position:absolute; top:50; left:45; color:Black; font:bold 24pt Verdana; }

posicion2 { position:absolute; top:62; left:38; color:red; font:bold 24pt Verdana; }

posicion3 {display:block; position:absolute; top:80; left:75; color:white; font: bold 20pt Courier New;}

posicion4 { position:absolute; top:146; left:23; color:white; font: bold 12pt verdana;right:130;text-align:justify}

posicion5 { position:absolute; top:144; left:170; color:Black; font: bold 12pt verdana; }

posicion6 { position:absolute; top:115; left:350; color:Black; font: bold 34pt verdana; }

```
posicion7 { position:absolute; top:118; left:353; color:red; font: bold 34pt verdana; }
   .efectobrillo {positon:relative; filter: glow(color=red, strength=6) }
texto {display:block; color: blue;
   margin-top: 85px;
   margin-left: 70px;
   margin-right: 70px;
   font-size: 16px;
   line-height: 150%;
   font-family: Arial;
        text-align:justify       }

H1      {font : 30pt/25pt « Arial »
        font-weight: bold;
        color: green}
```

Listado 6.4

He aquí las marcas:

```
<?xml version=”1.0” encoding=”ISO-8859-1”?>
<?xml-stylesheet type=”text/css” href=”tres.css”?>
<documento xmlns:HTML=”http://www.w3.org/Profiles/XHTML-transitional”>
<plantilla_inicial>
<barravertical></barravertical>
   <comunicado_1>

<posicion3 class=”efectobrillo”><HTML:A HREF=”eje1.xml”
ACCESSKEY=”w”>Software de Intranet</HTML:A></posicion3>
<titulo1>Programación para redes Internet/Intranet</titulo1>
   </comunicado_1>
<barravertical2></barravertical2>
<texto></texto>
<texto><H1>1. </H1>Un Sistema operativo de red que soporta el intercambio de
información y, que, como tal, reside tanto en clientes como en servidores. Hoy en día,
existen  varios Sistemas operativos disponibles en el mercado: Unix, Linux, Windows NT,
Novell Netware, y otros.</texto>
<texto><H1>2. </H1>Aplicaciones de red, que en este caso, se refiere a la utilización de
browsers, residentes en los equipos servidor y clientes, así como de programas específicos
de correo electrónico, FTP, etc.</texto>
<texto><H1>3. </H1>Un Sistema de Gestión de Red, que permite el control de
prestaciones, problemas, seguridad o configuración.</texto>
<texto><H1>4. </H1>Protocolos de comunicación.</texto>
</plantilla_inicial>
```

</documento>

Listado 6.5

El programa se visualiza así:

Figura 6.5

Tablas en XML.

Existen tres formas para realizar tablas en XML:

- ✓ Incluir tablas utilizando las marcas para tablas del lenguaje HTML.
- ✓ Disponer un documento XML de tal forma que sus elementos utilicen los parámetros de las hojas de estilo en cascada.
- ✓ Utilizando etiquetas de especificación CALS, que son los mecanismos más utilizados para incluir tablas en documentos SGML.

Inclusión de tablas HTML en documentos XML.

En un documento XML podemos incluir tablas codificadas en HTML. Como hemos señalado, para hacer esto bastará con declarar en el documento XML el espacio de nombres HTML y luego escribir los elementos HTML precedidos por el identificador del espacio de nombres.

Para crear una tabla XML incluyendo HTML hay que considerar lo siguiente:

- ✓ Crear un documento XML con un elemento *raíz* con u identificador, en este caso le llamaremos *documento* y un elemento *titulo*, tal y como se indica a continuación:

```
<?xml versión="1.0"?>
<documento>
  <titulo>
  </titulo>
</documento>
```

- ✓ Declarar el espacio de nombres para HTML. El documento se indica a continuación:

```
<?xml versión="1.0"?>
  <documento xmlns:HTML="http://www.w3.org/Profiles/XHTML-transitional">
  <titulo>
  </titulo>
  </documento>
```

- ✓ Incluir los elementos HTML de la tabla, dentro del elemento titulo. Ver listado 6.4:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/css" href="vacia.css"?>
<documento xmlns:HTML="http://www.w3.org/Profiles/XHTML-transitional">

<titulo>
<HTML:table border="1" bordercolorlight="yellow" bordercolordark="orange"
align="center">
  <HTML:tr>
<HTML:th bgcolor="red"><HTML:font color="white">Hola</HTML:font></HTML:th>
  </HTML:tr>
</HTML:table>
</titulo>
</documento>
```

Listado 6.6

Podemos añadir atributos a las celdas de una tabla, ver listado 6.7.

```
<?xml version=”1.0”?>
<?xml-stylesheet type=”text/css” href=”vacia.css”?>
<documento xmlns:HTML=”http://www.w3.org/Profiles/XHTML-transitional”>
<titulo>
<HTML:table border=”1” bordercolorlight=”yellow” bordercolordark=”orange”
align=”center” width=”30%”>
    <HTML:tr>
<HTML:th><HTML:marquee direction=”left” bgcolor=”black”><HTML:font size=”4”
color=”white” face=”Comic Sans MS”>Bienvenido al mundo de la programaci&#243;n en
XML</HTML:font></HTML:marquee></HTML:th>
    </HTML:tr>
<HTML:tr><HTML:th><HTML:img
src=”COMPUTR9.gif”></HTML:img></HTML:th></HTML:tr>
</HTML:table>
</titulo>
</documento>
```

Listado 6.7

Podemos añadir las propiedades colspan y rowspan tal y como se indica en los siguientes listados:

```
<?xml version=”1.0”?>
<?xml-stylesheet type=”text/css” href=”vacia.css”?>
<documento xmlns:HTML=”http://www.w3.org/Profiles/XHTML-transitional”>
<titulo>

<HTML:table border=”4” bordercolordark=”red” bordercolorlight=”yellow”
align=”center”>
<HTML:tr>
<HTML:th align=”center”> uno</HTML:th>
<HTML:th align=”center”> dos</HTML:th>
<HTML:th align=”center”> tres</HTML:th>

</HTML:tr>
<HTML:tr>


<HTML:td rowspan=”3” valign=”top”>Pizza</HTML:td>
<HTML:td align=”center”>Mexicana</HTML:td>
<HTML:td align=”right”>1200000</HTML:td>
</HTML:tr>
<HTML:tr>
<HTML:td align=”center”>Texana</HTML:td>
```

```
<HTML:td align=”right”>500000</HTML:td>
</HTML:tr>
<HTML:tr>
<HTML:td>Toluque&#241;a</HTML:td>
<HTML:td align=”right”>500000</HTML:td>
</HTML:tr>
</HTML:table>
</titulo>
</documento>
```

Listado 6.8

```
<?xml version=”1.0”?>
<?xml-stylesheet type=”text/css” href=”vacia.css”?>
   <documento xmlns:HTML=”http://www.w3.org/profiles/XHTML-transitional”>

<titulo>

<HTML:table border=”1” width=”90%” bordercolordark=”red” bordercolorlight=”yellow”
align=”center”>

   <HTML:tr>

   <HTML:th>Uno</HTML:th>
   <HTML:th>dos</HTML:th>
   <HTML:th>tres</HTML:th>
   </HTML:tr>

   <HTML:tr>

   <HTML:th rowspan=”2”>Los n&#250;meros</HTML:th>

   <HTML:th>cuatro</HTML:th>

   <HTML:th>cinco</HTML:th>

   </HTML:tr>

   <HTML:tr>

   <HTML:th>seis</HTML:th>
   <HTML:th>siete</HTML:th>


   </HTML:tr>


   <HTML:tr>

                                        <HTML:th colspan=”2”
                                        align=”right”>Ocho</HTML:th>
                                        <HTML:th>Nueve</HTML:th>
</HTML:tr>
   </HTML:table>
```

</titulo>
</documento>

Listado 6.9

En los visualizadores se muestran las corridas de los listado 6.8 y 6.9:

uno	dos	tres
Pizza	Mexicana	1200000
	Texana	500000
	Toluqueña	500000

Figura 6.6. Corrida del listado 6.8

Uno	dos	tres
Los números	cuatro	cinco
	seis	siete
Ocho		Nueve

Figura 6.7 Corrida del listado 6.9.

Presentar elementos XML en forma de tabla.

- ✓ La propiedad display: table

  Se crea una tabla con el contenido del elemento. La tabla se presenta como un bloque, con un salto de línea antes y otro después.

- ✓ La propiedad display: table-row

  El elemento constituirá una fila de la tabla. El elemento a cuya propiedad display se asigna este valor debe estar contenido en otro elemento cuya propiedad display recoja el valor table-row.

- ✓ Propiedad display:table-cell

  El elemento constituirá una celda en la tabla. El elemento a cuya propiedad display se asigna este valor debe estar contenido en otro elemento cuya propiedad display recoja el valor table-row.

- ✓ Propiedad display: in-line-table

  Se creará una tabla que no se mostrará como un bloque, sino como si se tratara de un objeto en línea. Es decir, el texto del elemento contenedor fluirá a su alrededor.

- ✓ Propiedad display:table-row-group

  Permite utilizar un elemento del documento XML para agrupar dos o más filas a las que se quiere aplicar una presentación común. Equivaldría al elemento tbody del lenguaje HTML.

- ✓ Propiedad display:table-footer-group

  Permite utilizar un elemento para agrupar filas que servirán como pie de la tabla. Equivale al elemento thead del lenguaje HTML.

- ✓ Propiedad display:table-caption

  Muestra el texto del elemento como título de la tabla. Cuando se utiliza esta declaración en la regla de estilo de un elemento XML, se puede añadir a su regla otra declaración con la propiedad caption-side.

En el siguiente ejemplo se muestra algunas aplicaciones de las propiedades anteriores:

```
<?xml version=”1.0”?>
<?xml-stylesheet type=”text/css” href=”tabla10.css”?>
<hojaControl>
<titulo1>Control de publicaciones de Intranets</titulo1>
<centrar>
<infoPublicacion>
<titulo>Bibliograf&#237;a</titulo>
<fechas>
<fechaInicio>15-Oct-2004</fechaInicio>
<fechaFin>22-Ago-2005</fechaFin>
</fechas>
<aplicacion>
<nombreAplicacion>Programa</nombreAplicacion>
<versionAplicacion>2.0</versionAplicacion>
```

```
</aplicacion>
</infoPublicacion>
</centrar>
<salto></salto>
<documentos>

<documento>
<docId><campo>ID documento</campo></docId>
<docTitulo><campo>T&#237;tulo</campo></docTitulo>
<responsable tarea=”redaccion”><campo>Autor</campo></responsable>
<fechaInicio tarea=”redaccion”><campo>Fecha de
Adquisici&#243;n</campo></fechaInicio>
<fechaFin tarea=”redaccion”><campo>Adqusiciones Extras</campo></fechaFin>
</documento>
<documento>
<docId>EFR-8989-ESD</docId>
<docTitulo>HTML vesi&#243;n 4</docTitulo>

<responsable tarea=”redaccion”>Elisabeth Castro</responsable>
<fechaInicio tarea=”redaccion”>20-Ago-2004</fechaInicio>
<fechaFin tarea=”redaccion”>14-Ene-2005</fechaFin>
</documento>

<documento>
<docId>EFG-7364-DFS</docId>
<docTitulo>UNIX y LINUX Gu&#237;a pr&#225;ctica</docTitulo>
<responsable tarea=”redaccion”>Sebasti&#225;n S&#225;nchez</responsable>
<fechaInicio tarea=”redaccion”>18-Ago-2004</fechaInicio>
<fechaFin tarea=”redaccion”>14-Jun-2005</fechaFin>
</documento>

<documento>
<docId>RSA-7346-JHG</docId>
<docTitulo>Perl, CGI y JavaScript</docTitulo>
<responsable tarea=”redaccion”>SYBEX</responsable>
<fechaInicio tarea=”redaccion”>13-Oct-2004</fechaInicio>
<fechaFin tarea=”redaccion”>16-Nov-2004</fechaFin>
</documento>

<documento>
<docId>GFH-YWE-ILA</docId>
<docTitulo>Programaci&#243;n con XML</docTitulo>
<responsable tarea=”redaccion”>Ricardo Eito Brun</responsable>
<fechaInicio tarea=”redaccion”>1-Oct-2004</fechaInicio>
<fechaFin tarea=”redaccion”>1-Dic-2004</fechaFin>
```

```
</documento>


<documento>
<docId>CVD-UYT-IUF</docId>
<docTitulo>El Libro de Linux</docTitulo>
<responsable tarea=”redaccion”>Syed M. Sarwar</responsable>
<fechaInicio tarea=”redaccion”>1-Oct-2004</fechaInicio>
<fechaFin tarea=”redaccion”>1-Dic-2004</fechaFin>
</documento>
</documentos></hojaControl>
```

Listado 6.10. Programa XML.

El programa anterior lo vamos a asociar con los siguientes estilos CSS.

```
titulo1 { display : block ;
    Text-align : center ;
    Font-family : Verdana ;
    Font-size : 14 pt }
titulo { display : block ;
    Text-align : center ;
    Font-family : Verdana ;color:white;



    Font-size : 14 pt }
salto{display:block}
infoPublicacion { display : table ;text-align:center;color:white;
        Border : thin solid yellow ;
Width : 40% ;
        Margin-left : auto ;
        Margin-right : auto;background-color:#3A056B }

centrar{Margin-left:30%}
infoPublication titulo { display : table-row }
infoPublicacion fechas { display : table-row }
infoPublicacion fechas fechaInicio { display : table-cell }
infoPublicacion fechas fechaFin { display : table-cell }
infoPublicacion aplicacion { display : table-row }
infoPublicacion aplicacion nombreAplicacion { display : table-cell }
infoPublicacion aplicación versionAplicacion { display : table-cell }
documentos { display : table; font-family : verdana ; font-size : 10pt}
infoDocs { display : table-header-group }
campo {display : table-cell ; font-weight : bold}
```

```
documento {display : table-row }
docId { display : table-cell ; width : 3cm; }
docTitulo { display : table-cell ; width : 6cm }
responsable { display : table-cell ; width : 3.5cm}
fechaInicio { display : table-cell ; width : 3.2cm}
fechaFin { display : table-cell ; width : 3.4cm}
```

Listado 6.11. Estilo en cascada del programa 6.10

El programa se visualiza del siguiente modo:

Figura 6.8

Multimedia.

Hay una enorme variedad de de sitios Web donde ofrecen gratuitamente objetos de multimedia, tales como sonido, video, animación y un gran etcétera.

Gif animado. Es una imagen vectorial con movimiento. Ver listado 6.11:

```
<?xml version=”1.0”?>
<?xml-stylesheet type=”text/css” href=”vacia.css”?>
<documento xmlns:HTML=”http://www.w3.org/Profiles/XHTML-transitional”>
<titulo>
<HTML:table border=”1” bordercolorlight=”yellow” bordercolordark=”orange”
align=”center” width=”30%”>
    <HTML:tr>
<HTML:th><HTML:marquee direction=”left” bgcolor=”black”><HTML:font size=”4”
color=”white” face=”Comic Sans
MS”>Bienvenido al mundo de la programaci&#243;n en
XML</HTML:font></HTML:marquee></HTML:th>
    </HTML:tr>
<HTML:tr><HTML:th><HTML:img
src=”AN_00807.gif”></HTML:img></HTML:th></HTML:tr>
</HTML:table>
</titulo>
</documento>
```

Listado 6.11

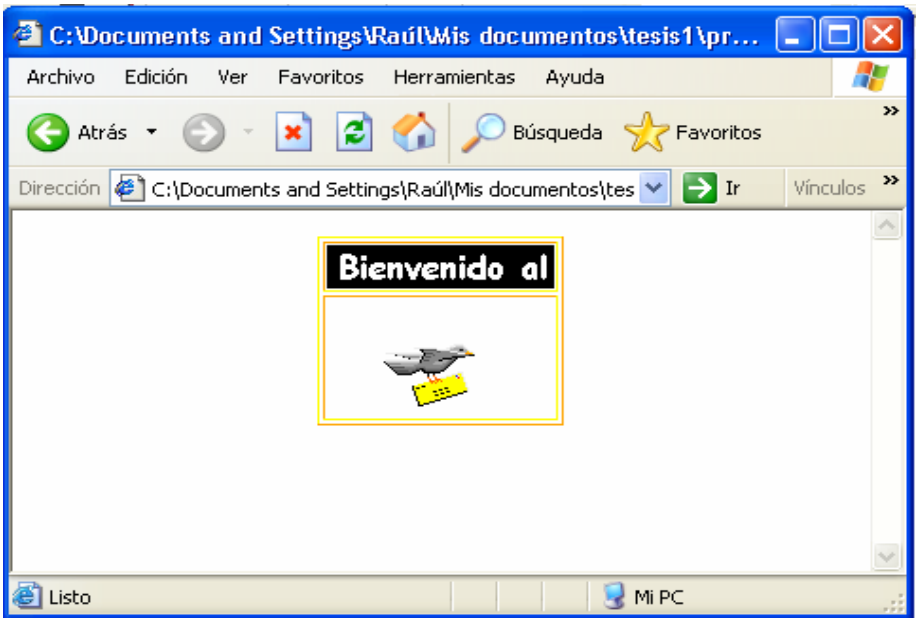


Figura 6.9

Multimedia en XML.

Multimedia en XML es un tipo especial de página Web integrada por elementos tales como: Texto, sonido, animación, video e interactividad gráfica.

Para el propósito de este trabajo utilizaremos los espacios de nombres, es decir, vamos a combinar diferentes DTD en nuestra página Web. Ver listado 6.12

- ✓ Definiendo los estilos:

```
*{background-image: url('B1_00090.gif')}
titulo2{position:absolute;top:80;font-family: Comic Sans MS;border:medium none silver thin dashed; background-color:yellow; padding:2px; margin-left:"25px"; text-indent:"20"; font-size:20;color:white}
barravertical { position:absolute; top:140; left:20; height:230;  width:30;background-color:#00CCFF;}
barravertical2 { position:absolute; top:220; left:720; height:600;  width:30;background-color:teal;}
posicion4 { position:absolute; top:146; left:73; color:white; font: bold 22pt verdana; }
posicion5 { position:absolute; top:144; left:70; color:black; font: bold 22pt verdana; }
    .efectobrillo {positon:relative; filter: glow(color=orange, strength=6) }

texto{position:absolute; left:50pt;right:40pt;text-align:justify;text-indent:8pt;
    font: 10pt/15pt "Arial", "Verdana";color:white}
texto.intro {text-indent:0;font: italic bold;color:white}
texto.intro EM {font-style:normal}

texto1{position:absolute; top:230; left:80pt;right:80pt;text-align:justify;text-indent:10pt;
    font: 10pt/15pt "Arial", "Verdana";color:white}
texto1.intro {text-indent:0;font: italic bold}
texto1.intro EM {font-style:normal}

HTML\:A {position:absolute; top:320; left:250; height:20;
width:30;color:red;background:white;
display:block;text-align:center;width:70%}
HTML\:A:actived { color : yellow }
HTML\:A:visited { color :black ;background:yellow }
HTML\:A:hover{background:white;color:red;text-decoration:none}
```

- ✓ Definiendo las marcas XML.
- ✓

✓

```
<?xml version=”1.0” encoding=”ISO-8859-1”?>
<?xml-stylesheet type=”text/css” href=”multi4.css”?>
<documento xmlns:HTML=”http://www.w3.org/Profiles/XHTML-transitional”>
<iniciar_el_documento>
<barravertical></barravertical>
<barravertical2></barravertical2>
<titulo2> Concierto para viol&#237;n y orquesta No. 2, en Vivace</titulo2>


<posicion4 class=”efectobrillo”> J. S. BACH </posicion4>

<texto class=”intro”>En este apartado aprenderemos

la forma de crear enlaces de multimedia para introducir m&#250;sica en nuestras
p&#225;ginas Web; hay distintas formas de hacerlo, una es introduciendo m&#250;sica de
fondo, la otra como hierenlaces, etc..

</texto>

<texto1>  Johann  Sebastian Bach, compuso sus conciertos para viol&#237;n alrrededor del
1720, cuando ten&#237;a aproximadamente treinta y cinco a&#241;os de edad. La
estructura de su discurso musical est&#225; basada en la m&#225;s meticulosa
l&#243;gica, y entre sus melod&#237;as encontramos algo de lo m&#225;s hermoso,
po&#233;tico y aflictivo de la historia de la m&#250;sica.

<HTML:A HREF=”multi5.xml” ACCESSKEY=”w”>Da un click y empieza a disfrutar o
bi&#233;n Alt + W,enter</HTML:A>

</texto1>
<HTML:br></HTML:br>
<HTML:br></HTML:br>
<HTML:br></HTML:br>
<HTML:br></HTML:br>
<HTML:br></HTML:br>
<HTML:br></HTML:br>
<HTML:br></HTML:br>
<HTML:br></HTML:br>
<HTML:br></HTML:br>
<HTML:table border=”3” bordercolordark=”white” bordercolorlight=”white”
align=”center”>

<HTML:tr>

<HTML:th align=”center”><HTML:font color=”white”> J. S.
BACH</HTML:font></HTML:th>

<HTML:th align=”center”><HTML:font color=”white”>
Movimiento</HTML:font></HTML:th>
```

```
<HTML:th align=”center”> <HTML:font color=”white”>
T&#237;tulo</HTML:font></HTML:th>

</HTML:tr>
<HTML:tr>
<HTML:th rowspan=”3” valign=”top”><HTML:font color=”white”>  Concierto para
viol&#237;n y orquesta No. 2, en Vivace</HTML:font></HTML:th>

<HTML:th align=”center”><HTML:font color=”white”>1</HTML:font></HTML:th>

<HTML:th align=”center”><HTML:font
color=”white”>Vivace</HTML:font></HTML:th>

</HTML:tr>
<HTML:tr>
<HTML:th align=”center”><HTML:font color=”white”> 2</HTML:font></HTML:th>
<HTML:th align=”center”><HTML:font color=”white”>Largo ma non
tanto</HTML:font></HTML:th>

</HTML:tr>
<HTML:tr>

<HTML:th align=”center”><HTML:font color=”white”> 3</HTML:font></HTML:th>

<HTML:th align=”center”><HTML:font
color=”white”>Allegro</HTML:font></HTML:th>

</HTML:tr>


</HTML:table><HTML:BGSOUND SRC=”08 Pista 8.wma”></HTML:BGSOUND>
</iniciar_el_documento>
</documento>
```

Listado 6.12

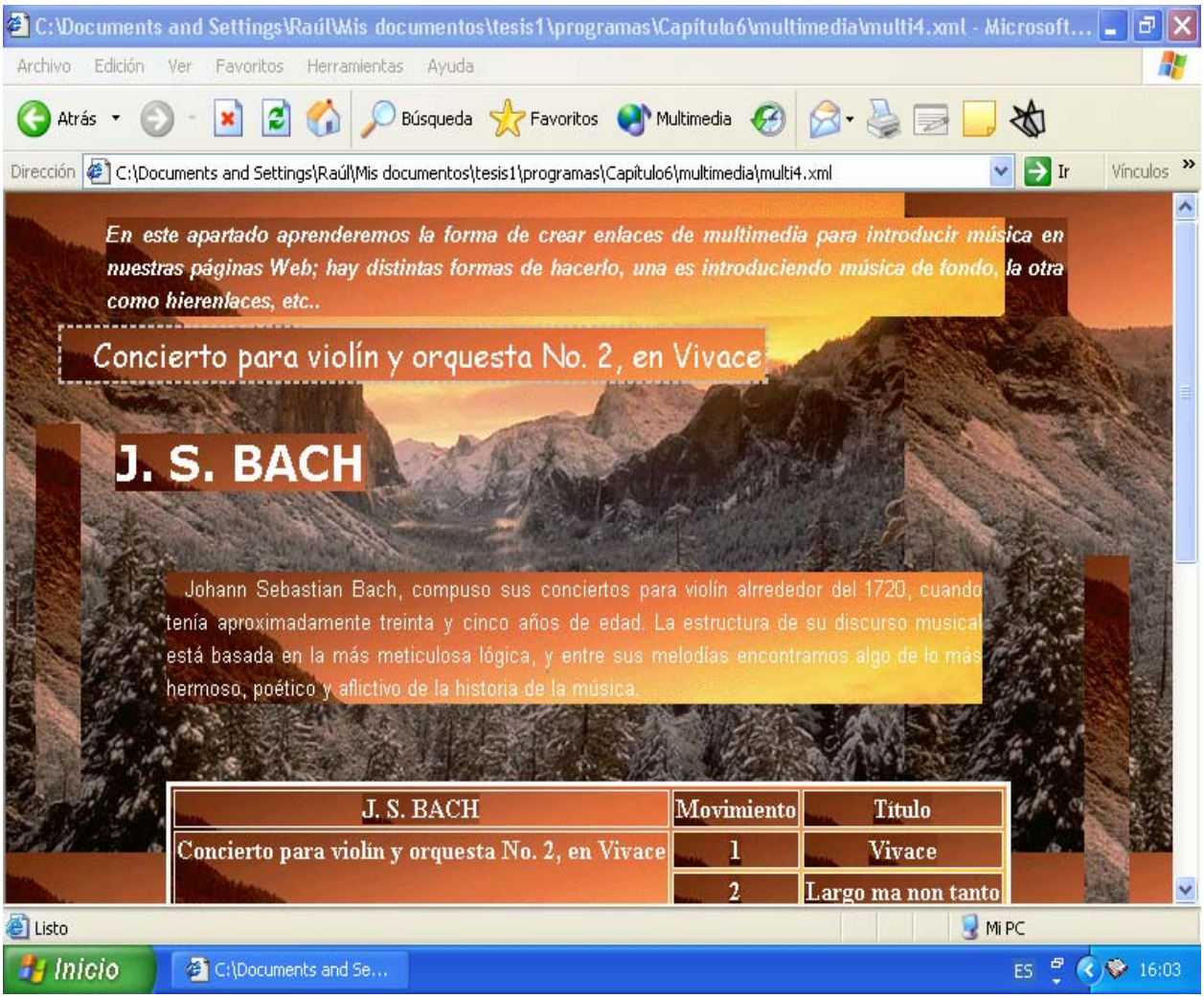


Figura 6.10

En la figura anterior se puede apreciar, una página con alto nivel de granularidad con música de fondo, tablas y estilos en cascada. Ahora bién, con esto ya estamos en posibilidad de establecer una serie de criterios para crear documento XML válidos y bién formados, un documento XML está bién formado si:

- ✓ Consta de al menos un elemento.
- ✓ Todos los elementos están comprendidos dentro de un elemento documento o raíz.
- ✓ Sus elementos están anidados correctamente.
- ✓ Todas las entidades que se utilizan han sido declaradas en el subconjunto interno de la DTD.

Las condiciones de válido y bién formado no son excluyentes. De esta forma un documento válido será siempre un documento bién formado, pero un documento bién formado no tiene

por que ser un documento válido, ya que la condición de bién formado, no implica que se tenga que comprobar si el documento cumple con las restricciones indicadas en la DTD.

Parsers

Un parser es un validador que se encarga de comprobar si el documento cumple con las restricciones indicadas en la DTD.

Islas de datos XML en HTML.

Las islas de datos nos permiten utilizar datos XML dentro de una página Web html. Hay que mencionar que las islas de datos es una tecnología desarrollada por Microsoft por lo cuál no pertenecen a los estándares desarrollados por W3C. Lo que hacemos es mostrar datos XML como si se tratasen de tablas de bases de datos en Internet.

Primero que nada para crear una isla de datos tenemos que declarar la etiqueta de apertura obligada:

```
<?xml version=”1.0”?>

.
.
.
</xml>
```

para después crear el registro datosPersonales de la base de datos estudiantes tal y como se muestran a continuación.:

```
        <estudiantes>
<datosPersonales>
   <estudianteID>REDT2004-2008</estudianteID>
   <nombre>Ju&#225;n</nombre>
   <paterno>Obrador</paterno>
   <materno>Landa</materno>
   </datosPersonales>

   </estudiantes>
```

Listado 6.13

Para mostrar la información procedente de la isla de datos, hay que asociar los elementos del documento XML con los elementos de un documento HTML

Para asociar elementos de HTML con un elemento XML de la isla de datos, se incluirán en la etiqueta de inicio del elemento HTML con los siguientes atributos:

- ✓ datasrc: Tomará como valor el identificador correspondiente a la isla de datos. Este atributo. Este atributo habrá sido señalado dentro del atributo id que acompaña al elemento xml que ha sido utilizado para la creación de la isla de datos.
- ✓ datafld: tomará como valor el nombre del elemento xml de la isla de datos, cuyo valor se quiere mostrar en el espacio correspondiente al elemento html.

- ✓ dataformatas: este atributo es de carácter opcional, recogerá los valores text o html, dependiendo de si se quieren interpretar las marcas que contenga el elemento xml como texto o como marcas.

Esta es la forma en como se presenta un elemento xml con si id.

```
<xml id=”etiqueta”>
<?xml version=”1.0”?>

<xml>
```

al añadirle datos a la isla y estableciendo una relación con html quedará del siguiente modo:

```
<html><head><title>Una isla sencilla</title></head>
<body>
        <xml id=”basededatos”>
<?xml version=”1.0”?>
                                                <base_datos>
    <estudiante>
    <nombre>Felipe</nombre>
    <apepat>Cruz</apepat>
    <apemat>Flores</apemat>
    </estudiante>
    </base_datos>
    </xml>
        <span datasrc=”#basededatos” datafld=”nombre”></span><br>
        <span datasrc=”#basededatos” datafld=”apepat”></span><br>
        <span datasrc=”#basededatos” datafld=”apemat”></span><br>
</body>
</html>
```

Listado 6.14

- ✓ span y div: estos dos elementos nos permiten delimitar un espacio dentro de un documento html. Si se le asocia a un elemento xml de una isla de datos, el valor de este se mostrará en el espacio que delimitan.

En el listado anterior tenemos una base de datos con un solo registro, vamos a crecer nuestra pequeña isla de datos y hagámosla un poco más presentable:

```
<html><head><title>Una isla sencilla</title></head>
<body>
        <xml id=”basededatos”>
```

```
<?xml version=”1.0”?>
                                        <base_datos>
    <estudiante>
    <nombre>Felipe</nombre>
    <apepat>Cruz</apepat>
    <apemat>Flores</apemat>
    <matricula>WER-873643</matricula>
    <telefono>56-78-90-12</telefono>
    </estudiante>
    <estudiante>
    <nombre>Rodolfo</nombre>
    <apepat>Rojas</apepat>
    <apemat>Aristeo</apemat>
    <matricula>DFR-654643</matricula>
    <telefono>87-90-11-34</telefono>
    </estudiante>
    <estudiante>
    <nombre>Rosa</nombre>
    <apepat>Robles</apepat>
    <apemat>Rivera</apemat>
    <matricula>ERT-753765</matricula>
    <telefono>45-90-67-11</telefono>
    </estudiante>

    <estudiante>
    <nombre>Gabriel</nombre>
    <apepat>Gordillo</apepat>
    <apemat>Amat</apemat>
    <matricula>YUT-784343</matricula>
    <telefono>78-90-87-11</telefono>
    </estudiante>
    </base_datos>
    </xml>

<table align=”center” width=”40%”>
<tr><td bgcolor=”skyblue” align=”center”>
<font size=”5” color=”brown” face=”Verdana”>agenda telef&oacute;nica</font>
</td></tr>


</table>
<br>
<table align=”center” width=”60%”>

<tr><td bgcolor=”brown”>
```

```
<font color=”white” size=”4” face=”Comic Sans MS”>Nombre:</font>
</td>
<td bgcolor=”gray”>
   <span datasrc=”#basededatos” datafld=”nombre”></span><br>
</td>
</tr>
<tr>
<td bgcolor=”brown”>
<font color=”white” size=”4” face=”Comic Sans MS”>ApePaterno:</font>
</td>
<td bgcolor=”gray”>
   <span datasrc=”#basededatos” datafld=”apepat”></span><br>
</td>
<tr>
<td bgcolor=”brown”>
<font color=”white” size=”4” face=”Comic Sans MS”>ApeMaterno:</font>
</td>
<td bgcolor=”gray”>
   <span datasrc=”#basededatos” datafld=”apemat”></span><br>
</td>
</tr>
<tr>
<td bgcolor=”brown”>
<font color=”white” size=”4” face=”Comic Sans MS”>ApeMaterno:</font>
</td>

<td bgcolor=”gray”>
   <span datasrc=”#basededatos” datafld=”matricula”></span><br>
</td>
</tr>
<tr>
<td bgcolor=”brown”>
<font color=”white” size=”4” face=”Comic Sans MS”>Tel&eacute;fono:</font>
</td>
<td bgcolor=”gray”>
   <span datasrc=”#basededatos” datafld=”telefono”></span><br>
</td>
</tr>
</table>
<hr border=”2” color=”red”>
<!------A continuación siguen los botones----------->
<table cellpadding=”4” cols=”6 width=”95%” align=”center”>
        <tr>
<td align=”center” valign=”middle”>
```

```
<input type=”button” value=”Primero” onClick=”basededatos.recordset.MoveFirst()”>
<td align=”center” valign=”middle”>
<input type=”button” value=”anterior” onclick=”basededatos.recordset.MovePrevious();
        if(basededatos.recordset.BOF)
basededatos.recordset.Movefirst();”>
        <td align=”center” valign=”middle”>
<input type=”button” value=”Siguiente” onClick=”basededatos.recordset.MoveNext();
        if(basededatos.recordset.EOF) basededatos.recordset.MoveLast();”>
<td align=”center” valign=”middle”>
        <input type=”button” value=”Último”
        onclick=”basededatos.recordset.MoveLast()”>
<tr>
</table>
</body>
</html>
```

Listado 6.15

Nuestra isla se visualizará del siguiente modo:

Figura 6.11

Métodos para desplazarse en los registros de datos.

moveFirst	Nos desplazamos al primer elemento de la isla de datos.
moveLast	Nos desplazamos al último elemento de la isla de datos.
moveNext	Nos desplazamos al siguiente elemento.
movePrevious	Nos desplazamos al elemento anterior.
move(n)	Permite desplazarnos a un elemento que ocupa una posición determinada. Indicaremos esta posición como argumento del método. Los registros de la isla de datos se comienzan a numerar con el número 0. Así al primer registro le corresponderá el índice 0, al segundo el 1, etc.

Nota importante: Se puede llamar a estos métodos desde scripts que estén escritos en JavaScript o VBScript.

A continuación mostramos otra forma de presentar las islas de datos en xml.

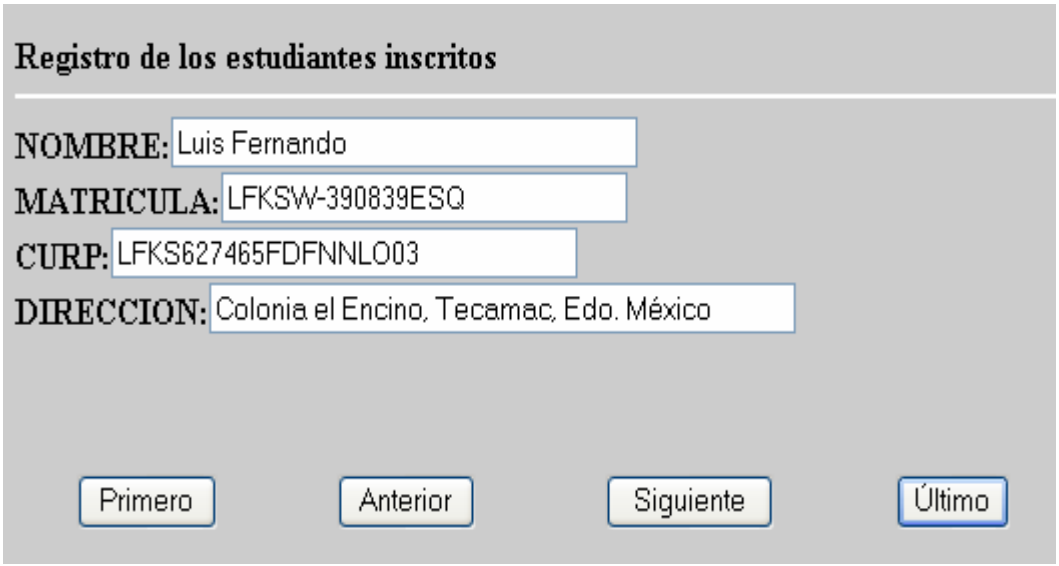


Figura 6.12

Llamadas a archivos xml con islas de datos. Consiste en crear archivos xml que contienen islas de datos y que sean referenciados en documentos html. Figura 6.12. Comenzaremos creando un documento xml que contenga una isla de datos

```
<?xml version=”1.0”?>
        <alta>
<estudiantesInfo>
   <nombre>Ana Laura</nombre>
   <matricula>ANLSW-87809QESQ</matricula>
   <curp>DERS457465HDFNNLO03</curp>
   <direccion>Col. Rinconada de Arag&#243;n 23, Edo. M&#233;xico</direccion>
   </estudiantesInfo>

   <estudiantesInfo>

   <nombre>Adam L&#243;pez</nombre>
   <matricula>ADDLS-87809QESQ</matricula>
   <curp>ADDLS857465HDFLNLO03</curp>
   <direccion>Colonia Las Palmeras, Acapulco Gro. 243, Lt-23</direccion>
   </estudiantesInfo>

   <estudiantesInfo>

   <nombre>Dulce Sugey</nombre>
   <matricula>DDS-870809QESQ</matricula>
   <curp>DSSRS457465FDFNNLO03</curp>
<direccion>Col. Benito Ju&#225;rez 345, Nezahualc&#243;yotl Edo.
M&#233;xico</direccion>

   </estudiantesInfo>

   <estudiantesInfo>

   <nombre>Ra&#250;l Manzanares</nombre>
   <matricula>RMLW-87809QESQ</matricula>
   <curp>RMF467465HDFNULO03</curp>
   <direccion>Dulce Oliva, Bosques de Pedregal, Calz Tlalpan 345 D,F</direccion>
   </estudiantesInfo>

   <estudiantesInfo>

   <nombre>Jos&#233; L&#243;pez </nombre>
   <matricula>JLNJW-77899QESQ</matricula>
   <curp>JLK457465HDFNNLO03</curp>
   <direccion>Col Ju&#225;rez 243, esq, Balderas, Calle Delicias, D,F</direccion>
   </estudiantesInfo>

   <estudiantesInfo>

   <nombre>Cristina Ahumada</nombre>
   <matricula>CALSW-77839QESQ</matricula>
   <curp>CAS427465FDFNNLO03</curp>
```

```
<direccion>Colonia San Agust&#237;n Ecatepec , Edo. M&#233;xico</direccion>
    </estudiantesInfo>
    <estudiantesInfo>
    <nombre>Luis Fernando</nombre>
    <matricula>LFKSW-390839ESQ</matricula>
    <curp>LFKS627465FDFNNLO03</curp>
    <direccion>Colonia el Encino, Tecamac, Edo. M&#233;xico</direccion>
    </estudiantesInfo>


</alta>
```

Listado 6.16

A continuación crearemos la plantilla de datos en html:

```
<html><head><title>Ejemplo de un formulario sencillo</title></head>
<body bgcolor="#CCCCCC">
</body>

<xml id="basedeDatos" src="estudiantes.xml"></xml>

<form name="DatosPersonales">
    <b>Registro de los estudiantes inscritos</b><br><hr border="4" color="white">
<b>NOMBRE:</b><input name="Nombre" size="35" datasrc="#basedeDatos"
datafld="nombre"><br>
<b>MATRICULA:</b><input name="Matricula" size="30" datasrc="#basedeDatos"
datafld="matricula"><br>
<b>CURP:</b><input name="CURP" size="35" datasrc="#basedeDatos"
datafld="curp"><br>
<b>DIRECCION:<b><input name="direccion" size="45" datasrc="#basedeDatos"
datafld="direccion"><br>
<br><br><br>
<table ceepadding=4 cols=6 width="95%">
<tr><td align="center" valign=middle>

<input type="button" value="Primero"  ACCESSKEY="f"
onClick="basedeDatos.recordset.MoveFirst( )">
<td align="center" valign=middle>
<input type="button" value="Anterior" ACCESSKEY="p"
    onClick="basedeDatos.recordset.MovePrevious( );
        if(basedeDatos.recordset.BOF)
        basedeDatos.recordset.MoveFirst;">
```

```
<td align=”center” valign=middle>
<input type=”button” value=”Siguiente”  ACCESSKEY=”n”
onClick=”basedeDatos.recordset.MoveNext( );

if(basedeDatos.recordset.EOF)
                                                basedeDatos.recordset.MOvelast( );”>
        <td align=”center” valign=middle>
<input type=”button” value=”Último” ACCESSKEY=”l”
onClick=”basedeDatos.recordset.MoveLast( )”>

</tr>
</table>
</form>
</html>
```

Listado 6.17

Diseño de tablas con islas de datos xml.

Se trata de crear tablas html como plantillas con datos de islas xml. Lo primero que debemos hacer es crear una tabla html y posteriormente desde esta hacer las llamadas a la isla de datos xml, hay que tener en cuenta que primero se declara la etiqueta de inicio xml con su identificador y la liga que llama a los datos de la isla. Ver listado 6.18

```
<html><head><title>Tablas con islas de datos</title></head>
<body>
<h1 align=”center”>Una base de datos sencilla</h1>

<xml id=”x” src=”estudiantes.xml”></xml>

   <table border=1 datasrc=”#x” datapagesize=1 id=”y”>

<thead>
<td>Nombre:</td>
<td>Matr&iacute;cula:</td>
<td>CURP:</td>
<td>Direcci&oacute;n:</td>
</thead>
<tr>
<td>
   <span datafld=”nombre”></span>

</td>

<td>
<span datafld=”matricula”></span>
</td>


<td>
<span datafld=”curp”></span>
</td>
```

```
<td>
<span datafld=”direccion”></span>
</td>
<table cellpadding=”4” cols=6 width=”95%”>
<tr>
<td align=”center” vaign=middle>
<input type=”button” value=”Primero”  onClick=”y.firstPage()”>
<td align=”center” valign=”middle”>
<input type=”button” value=”Anterior” onClick=”y.previousPage()”>
<input type=”button” value=”Siguiente” onClick=”y.nextPage()”>
<td align=”center” valign=”middle”>
<input type=”button” value=”Última” onClick=”y.lastPage()”>
</tr>
</table>
</body>
</html>
```

Listado 6.18

El programa anterior muestra se pueden hacer llamadas a otros programas, y también manejar, pero también, además de hacer llamadas, se pueden manejar objetos, pero esto se abordará en el capítulo VIII, por el momento, XML es:

- ✓ Un lenguaje que fue diseñado para hacer páginas Web, con la facilidad de que aquí la cantidad de etiquetas son infinitas, ya que es uno quién define sus propiedades.
- ✓ Las etiquetas pueden heredar de otras etiquetas.
- ✓ Uno determina el nivel de detalle, es decir, la Granularidad de los programas.
- ✓ Con XML se puede hacer uso de otros lenguajes de programación, sobretodo los que son orientados a objetos como el JAVA, entre otros.
- ✓ Se puede hacer uso de objetos de Microsoft para bases de datos, pero no solamente de eso, también puede utilizar la potencia de otras plataformas como LINUX y UNIX.
- ✓ Con XML se pueden también crear páginas dinámicas.

Una buena parte de los conceptos orientados a objetos se pueden aplicar en XML, no hay un lenguaje de programación que maneje completamente todos los conceptos de la orientación a objetos, ya que el estudio de análisis y diseño orientado a objetos es muy complicado y extenso, además de que hay corrientes y unas ofrecen ciertas ventajas con respecto a otras.

VII. Programación para Internet.

7.1. Programación Para Internet. Ahora vamos a crear los elementos lógicos que permitan la interactividad con un sitio Web utilizando los lenguajes Java Script y Visual Basic Script, para crear los elementos de desarrollo de las bases de datos en Web.

7.2. Asociando formularios con Scripts.

Un formulario es un conjunto de marcas receptoras de datos introducidos por el usuario de forma interactiva para ser enviados a una base de datos. Ver figura 7.1.

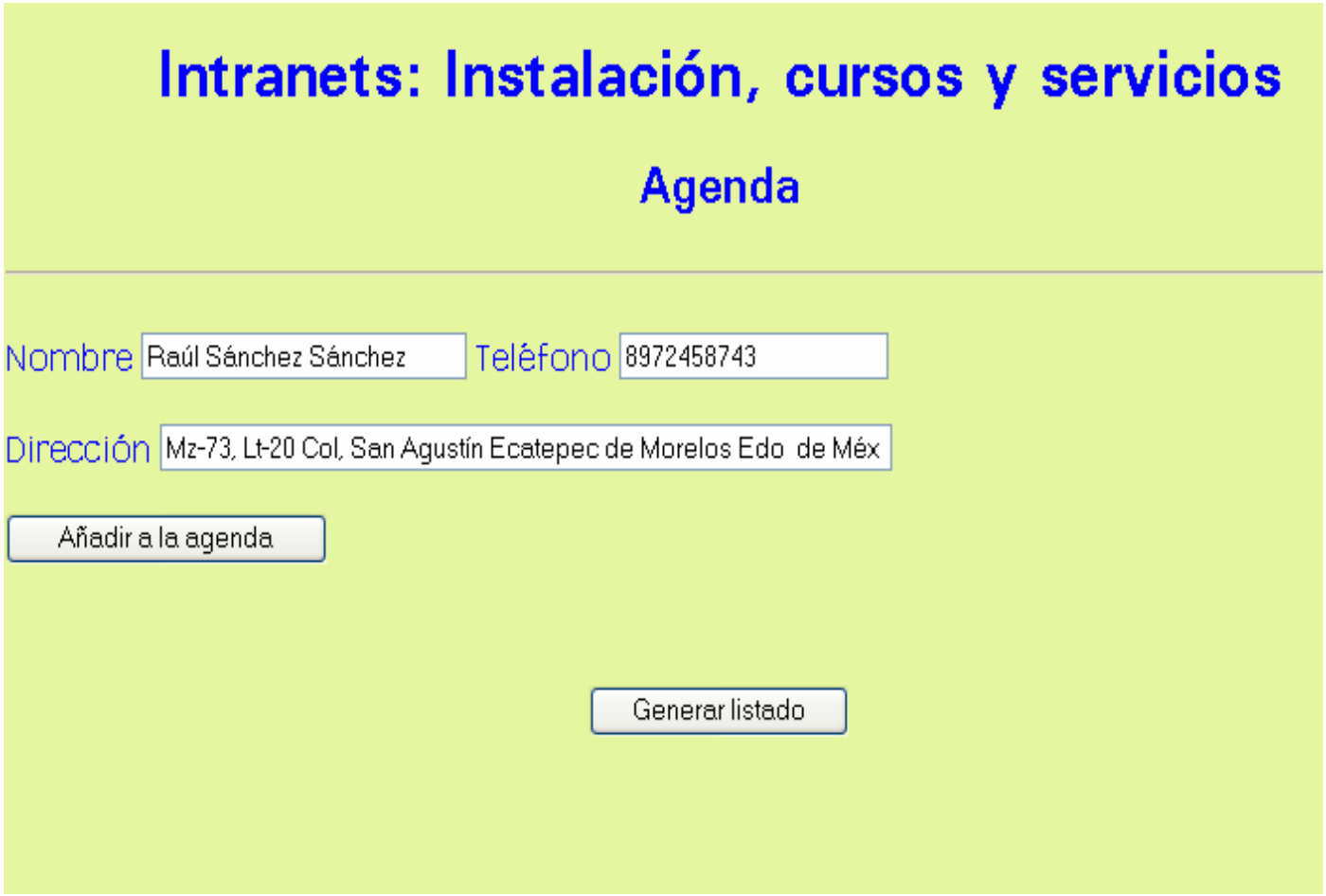


Figura 7.1

Ahora bién, un formulario por sí mismo no permite la interactividad con la computadora, pues como sabemos, el lenguaje de marcado es completamente estático, es decir, es un lenguaje que no se hizo para el procesamiento de datos de entrada, ni tampoco para el manejo de eventos tan elementales en la computadora tales como, un clic con el ratón, oprimir una tecla, la comunicación con los periféricos tales como mandar a imprimir un documento, o guardar archivos en un disco, para tal efecto los lenguajes de marcado permiten la incorporación de lenguajes que permiten la interactividad con la computadora tales como el Java Script, el Visual Basic Script, el PHP, MySQL, Java, C y C++, y un gran etcétera.

Para realizar páginas Web interactivas es necesario que se tenga el dominio de un lenguaje de cualquier nivel que permita la interactividad, para este propósito es suficiente con dominar el Java Script y Visual Basic Script.

7.3. Java Script.

El Java Script fue implementado por NetScape y su éxito se debe a que la firma no deja de implementar nuevas funciones al mismo. Es muy común que en casi todas las páginas en Internet[1].

El JAVA implementado por la empresa SUN en nada se parece a Java Script, Java Script fue creado exclusivamente para la implementación de documentos Web interactivos y JAVA es un lenguaje que va más allá, pues aparte de que se pueden desarrollar implementaciones para ser ejecutadas en Internet llamadas applets, es orientado a objetos, de alto nivel y de propósito general, es un lenguaje con alto nivel de seguridad, lo que no necesariamente sucede con Java Script.

El java Script es portable y multiplataforma fácil de aprender, y se presta para el fácil manejo de formularios. Ver listado 7.1.

```
<script language="JavaScript">
function esfera(form){
r=eval(form.r.value)
vol=(4/3)*3.1415927*r*r*r
form.vol.value=vol
    }
</script>
<form name="formulario">
<input type="text" name="r" size="5">
<input type="button" value="Volúmen" onClick="esfera(this.form)">
<input type="number" name="vol" size=10>
</form>
```

Listado 7.1

Para relacionar un atributo de un elemento de formulario del listado anterior se va a obtener el volumen de una esfera podemos explicar de forma simple este proceso: Primero que nada, vamos a dividir en dos bloques el programa anterior, uno en donde se realice propiamente el Script y la otra en donde se encuentra el formulario. En el formulario se reciben los datos y los Scripts se encargan de los procesos. Esto lo explicaremos a continuación.

Entrada: En el bloque de las marcas HTML debemos definir los elementos que van a recibir los datos que introduzcan los usuarios.

Proceso: Aquí se diseñan los Scripts que van a procesar los datos recibidos por el o los formularios.

Salida: Son los elementos del formulario que muestran los resultados de los procesos realizados en los Scripts.

Vamos a ilustrar lo anterior con un diagrama EPS.

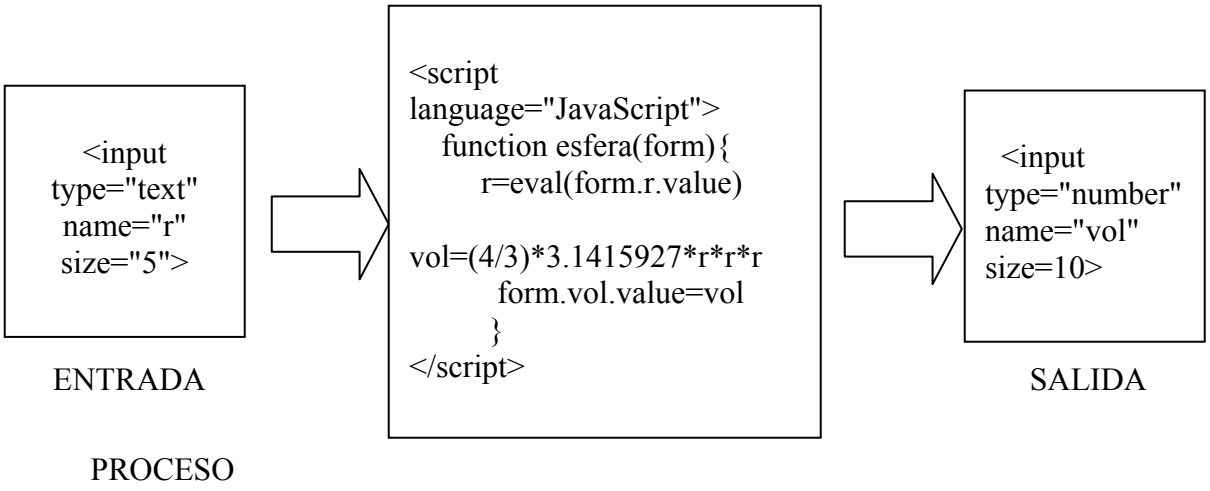


Figura 7.2

En el diagrama anterior ilustramos en forma sistémica la forma de recibir datos por parte de los formularios, aunque los formularios se utilizan comúnmente para recibir datos de los usuarios hay casos como el anterior en el que también dan respuestas, mientras los Scripts son los que se encargan de procesar los datos, aunque como veremos más adelante, no es su única tarea, cuando hablamos de interactividad con las páginas Web, esto supone que se llevan implícitas todas las posibilidades.

A continuación se muestra otro ejemplo donde el usuario introduce los datos, y son procesados por el Script para obtener una respuesta, por medio de los mismos elementos del formulario, en este caso el formulario la hace como receptor de datos y emisor de información.

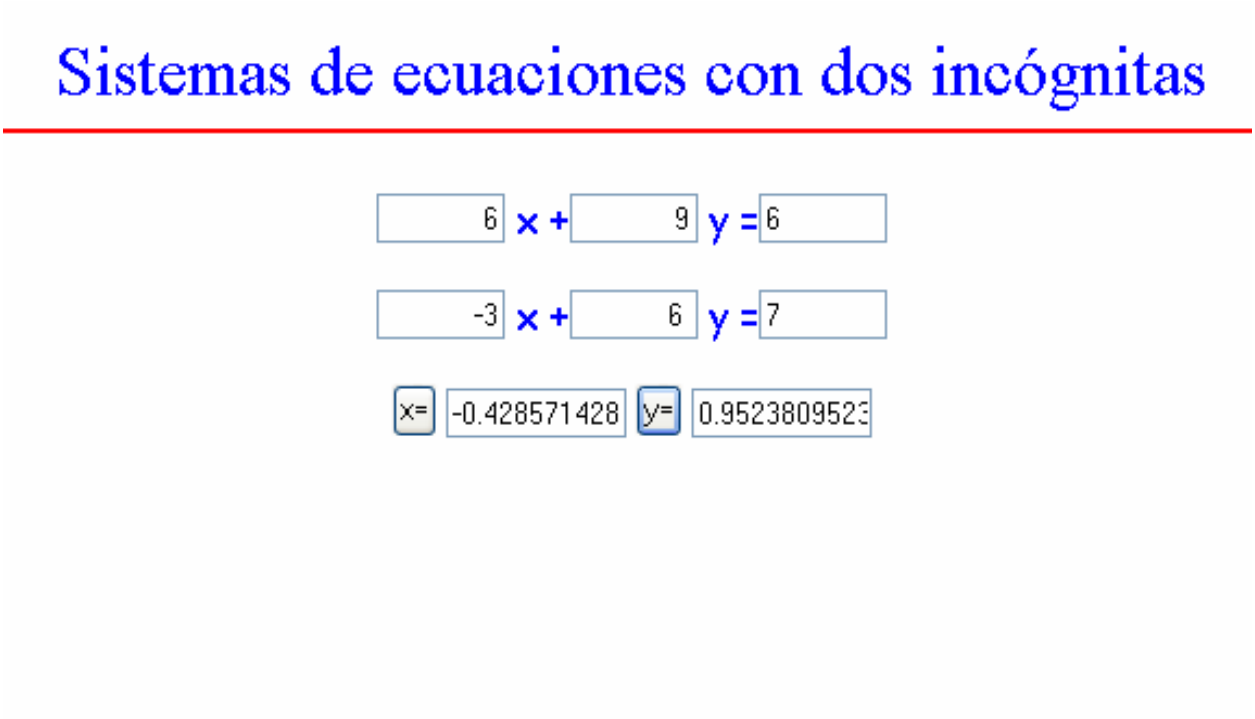


Figura 7.3

Aunque procesar los datos no es lo único que caracteriza a los Scripts, estos también tienen la posibilidad de acceder a algunos de los dispositivos disponibles, tales como las unidades de almacenamiento, las impresoras, las videoconferencias, las cámaras de video, etc.... A continuación se muestra el listado de la figura anterior:

```
<html><head><title>Sistemas de ecuaciones con dos inc&oacute;gnitas</head></title>
<body>
<center><font color="blue" size="6">Sistemas de ecuaciones con dos
inc&oacute;gnitas</font></center>
<hr color="red">
<script language="JavaScript">
    function equis(form){
    a11=eval(form.a11.value)
    a12=eval(form.a12.value)
    b1=eval(form.b1.value)
    a21=eval(form.a21.value)
    a22=eval(form.a22.value)
    b2=eval(form.b2.value)

x=(b1*a22-b2*a12)/(a11*a22-a21*a12)
form.x.value=x

 }

function ye(form){
    a11=eval(form.a11.value)
```

```
    a12=eval(form.a12.value)
    b1=eval(form.b1.value)
    a21=eval(form.a21.value)
    a22=eval(form.a22.value)
    b2=eval(form.b2.value)

y=(a11*b2-a21*b1)/(a11*a22-a21*a12)
form.y.value=y
 }

</script>
<center><form name=”formulario”>
<input type=”text” name=”a11” size=”6”><font face=”Comic Sans MS” color=”blue”
size=”4”> x +</font><input type=”text” name=”a12” size=”6”><font face=”Comic Sans
MS” color=”blue” size=”4”> y =</font><input type=”text” name=”b1”
size=”6”><br><br>

<input type=”text” name=”a21” size=”6”><font face=”Comic Sans MS” color=”blue”
size=”4”> x +</font><input type=”text” name=”a22” size=”6”><font face=”Comic Sans
MS” color=”blue” size=”4”> y =</font><input type=”text” name=”b2”
size=”6”><br><br>

<input type=”button” value=”x=” onClick=”equis(this.form)”>
<input type=”text” name=”x” size=”10”>
<input type=”button” value=”y=” onClick=”ye(this.form)”>
<input type=”text” name=”y” size=”10”>
</form></center>
</body>
</html>
```

Listado 7.2

El manejo de eventos es una de tantas posibilidades que ofrecen los Scripts, estos nos dan la posibilidad de interactuar con los dispositivos más importantes de la computadora y que son comunes al usuario común.

Un programa bién hecho

Hay que resaltar que los programas bién hechos son aquellos que están bién organizados, abarcan la menor cantidad de código posible, son reutilizables, son multiplataforma, fáciles de entender y son seguros en casi todos los aspectos.

Palabras reservadas.

Las palabras reservadas son aquellas que tienen un propósito predeterminado.

Estas son algunas palabras reservadas de JavaScript.

abstract	Else	if	new	static
try	Bolean	extends	implements	null
super	Typeof	break	false	import
package	Switch	var	byte	final
in	Private	synchronized	void	case
finally	Instanceof	protected	this	while
catch	Flota	int	public	throw
with	Char	for	interface	reset
throws	Do	function	long	return
trasient	Double	goto	native	short
true				

Tabla 7.1. Palabras reservadas de JavaScript.

Crear Objetos.

JavaScript nos permite crear nuestros propios objetos. Para ello habrá que utilizar las siguientes palabras reservadas:

var nombreVariable= new NombreClase

en el caso anterior estamos creando un nuevo caso de NombreClase y que le llamaremos nombreVariable.

En JavaScript los números, las cadenas y los arreglos son objetos. Por ejemplo vamos a crear un caso objeto en JavaScript:

var cadena_x = new string("Esto es un caso de objeto");

que equivale a:

var cadena_x= "Esto es un caso de objeto";

Métodos.

Un método es una función cuyas variables son objetos. La sintaxis es la siguiente:

```
function  etiqueta(obj1, obj2, obj3, … objn)

      {

   this.obj1=obj1 ;
   this.obj2=obj2 ;
```

```
    this.obj3=obj3 ;
            .
            .
            .
    this.objn=objn ;
}
```

He aquí un ejemplo donde se resalta la interacción de los casos de clase con los métodos:

```
<html><head><title>Base de Datos de los Estudiantes de Informática</title></head>
        <body>
<script languaje="JavaScript">
function estudiante(nombre,matricula,edad,direccion)
                                {
    this.nombre=nombre;
    this.matricula=matricula;
    this.edad=edad;
    this.direccion=direccion;
                                }
CB_estudiante1=new estudiante("Raúl Rodríguez Sánchez", "CB2002-LPJK-6","16","Ex Hacienda Espinoza de los Monteros A-34")
CB_estudiante2=new estudiante("Regina Rendón Durazo","CB2002-LPJK-7","15","Lomas de Chapultepec Colonia Condesa B90")
CB_estudiante3=new estudiante("Rosa Espinoza Sánchez","CB2002-LPJK-9","18","Av Lourdes Col San Agustín Ecatepec de Morelos")
var Base_Datos;
Base_Datos="Escuela de Programación de Computadoras\rEstudiante:"+CB_estudiante1.nombre+"\n"+"Matrícula: "
+CB_estudiante1.matricula+"\n"+"Edad: "+CB_estudiante1.edad+"\n"+"Dirección:"+CB_estudiante1.direccion;
alert(Base_Datos);
Base_Datos="Escuela de programación de Computadoras\rEstudiante:"+CB_estudiante2.nombre+"\n"+"Matrícula: "
+CB_estudiante2.matricula+"\n"+"Edad: "+CB_estudiante2.edad+"\n"+"Dirección:"+CB_estudiante2.direccion;
alert(Base_Datos);
Base_Datos="Escuela de Programación de Computadoras\rEstudiante:"+CB_estudiante3.nombre+"\n"+"Matrícula: "
```

```
+CB_estudiante3.matricula+"\n"+"Edad:"+CB_estudiante3.edad+"\n"+"Dirección:"+CB_e
studiante3.direccion;

alert(Base_Datos);
</script>
</body>
</html>
```

Listado 7.3

En el ejemplo anterior se creó una sencilla base de datos, el método se visualiza de la siguiente forma:

```
function estudiante(nombre,matricula,edad,direccion)
                                {
   this.nombre=nombre;
   this.matricula=matricula;
   this.edad=edad;
   this.direccion=direccion;
                                }
```

en este caso los objetos son: nombre, matricula, edad y direccion. Como se habrá notado, los casos del objeto nombre pueden ser:

nombre=" Raúl Pérez Sánchez, nombre="Julio Martínez Domínguez" ,… etc.

Ya lo demás es el manejo de las clases y los objetos que como se habrá notado es muy sencillo.

Breve repaso de métodos de control del flujo de los datos.

El control del flujo de los datos ofrecen la posibilidad de manejarlos a nuestras necesidades y pos eso son imprescindibles.

switch.

El método switch es utilizado comúnmente para el diseño de menús interactivos o bién cuando se realizan procesos que tienen distintas posibilidades de respuesta. La sintaxis es la siguiente:

```
   switch( opcion)
                                {

   case opc1:
   case opc2:
   case opc3:
```

.

.
.

case opcn:

}

El funcionamiento de esta estructura depende del resultado de una opción o bién de un proceso que se haya realizado interna o externamente en la computadora. Vamos a poner un ejemplo donde la estructura anterior funciona sin la intervención del usuario:

```
<script language=”JavaScript”>
var tiempo = new Date();
var mes = tiempo.getMonth();
switch (mes) {
case 0:
dateName = “Enero”;
alert(dateName);
break;
case 1:
dateName = “Febreo”;
alert(dateName);
break;
case 2:
dateName = “Marzo”;
alert(dateName);
break;
case 3:
dateName = “Abril”;
alert(dateName);
break;
case 4:
dateName = “Mayo”;
alert(dateName);
break;
case 5:
dateName = “Junio”;
alert(dateName);
break;
case 6:
dateName = “Julio”;
alert(dateName);
break;
case 7:
```

```
dateName = “Agosto”;
alert(dateName);
break;
case 8:
dateName = “Septiembre”;
alert(dateName);
break;
case 9:
dateName = “Octubre”;
alert(dateName);
break;
case 10:
dateName = “Noviembre”;
alert(dateName);
break;
case 11:
dateName = “Diciembre”;
alert(dateName);
break
}
</script>
```

Listado 7.4

El programa anterior depende del método getMonth( ) que a su vez está supeditado al objeto Date.

Un método switch lo podemos controlar por medio entadas de formulario. Ver figura 7.4.

```
<html><head><title>Diseño de un men&uacute; sencillo</title></head>

<body>

<SCRIPT LANGUAGE=”JavaScript”>

function juego(form)

   {
palabra_clave=eval(form.palabra_clave.value)

switch(palabra_clave) {
case 1:
dateName = “En un momento se activará la base de datos para dar de alta a los empleados”;
alert(dateName);
break;
case 2:
dateName = “En un momento se activará la base de datos para dar de baja a los empleados”;

alert(dateName);
break;
case 3:
```

```
dateName = “En un momento se activará la base de datos para realizar cambios”;
alert(dateName);
break;
}
}
</script>

<center><font size=”6” font face=”Verdana” color=”red”>Men&uacute; de
empleados</font></center><br><br>

<table border=”3” width=”50%” bordercolordark=”red” bordercolorlight=”yellow”
align=”center”>

<tr><td align=”center”>

<font size=”4” color=”blue” face=”Comic Sans MS”>1.- Altas</font><br>
<font size=”4” color=”blue” face=”Comic Sans MS”>2.- Bajas</font><br>
<font size=”4” color=”blue” face=”Comic Sans MS”>3.- Cambios</font><br>
</tr></td>
</table>
<center>
<form name=”formulario”>
<input type=”number” name=”palabra_clave” size=”9”>
<input type=”button” onClick=”juego(this.form)” size=”9” value=”Opci&oacute;n”>
</form>
</center>
</body>
</html>
```

Listado 7.5

Al interpretar este programa en un navegador de Internet, resulta lo siguiente:

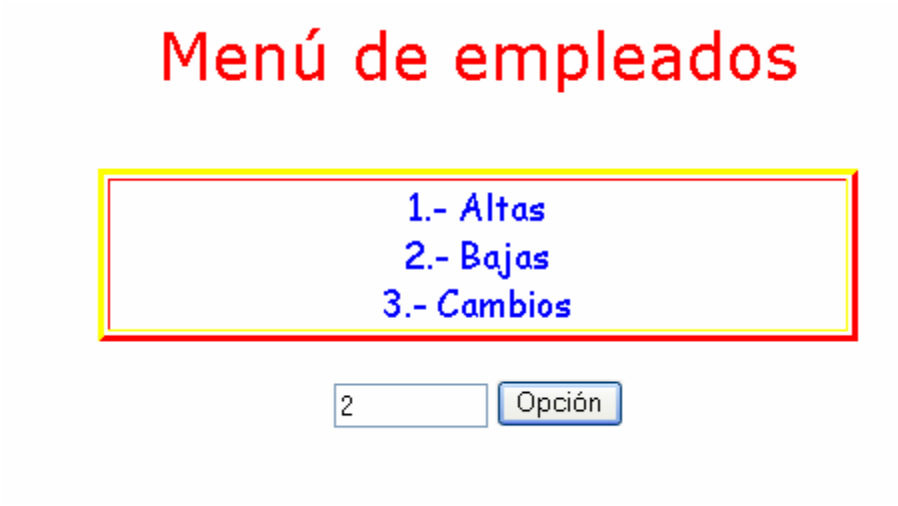


Figura 7.4

Como se puede notar, el ejemplo anterior solamente es el diseño de un sencillo menú para ilustrar el funcionamiento de switch. En el capítulo siguiente realizaremos una base de datos para la Web utilizando la herramienta anterior.

for

Es un bucle que se utiliza para realizar uno o más procesos. Todos los bucles tienen un inicio, un fin y un incremento, dicho incremento se realiza de uno en uno. El bucle for es muy útil en el diseño de bases de datos, por ejemplo para la ordenación y búsqueda de registros. Ahora bién, si queremos realizar procesos de dos en dos, de tres en tres u otro tipo de conteos determinados, se utilizan artificios de computación u otro tipo de algoritmos que sean viables con la estructura lógica del for. Su sintaxis es la siguiente:

```
for (inicio; fin; incremento)
        {
sentencia1;
sentencia2;
sentencia2;
                .
                .
                .
sentencia_n;

        }
```

Para ilustrar el método anterior vamos a discutir uno de los problemas actuales en México, según el INEGI desde la segunda mitad de los noventas, México está empezando a envejecer, dentro de 30 años la población de acianos subirá 400%[2] y en el 2050 la cuarta parte de los mexicanos rebasará los 60 años de edad[3]. La tasa de natalidad descenderá de 1.73% en 1995 a 0.20% a mediados del siglo XXI según la CONAPO. A esto hay que sumarle que los ancianos siendo aproximadamente el 5% de la población del planeta, consumen el 60% de las medicinas que se producen en el mundo. En México tenemos esa cifra similar que casi se aproxima al 6% de la población, a esto hay que sumarle, el problema de las pensiones, que ya resultan insuficientes además de que son precarias, de hecho el Instituto Mexicano del Seguro Social ya trabaja con números rojos, pues buena parte de sus ingresos se van en pagar las pensiones de los jubilados, y algo similar pasa con las pensiones del ISSSTE, las reformas recientes que se aprobaron, lamentablemente no van a solucionar los problemas de fondo, pues todavía pesan los intereses de grupo y la irresponsabilidad de algunos funcionarios y líderes sindicales… El análisis del comportamiento de las poblaciones son de vital importancia para tratar de adelantarse a los problemas futuros y darles la mejor solución posible, pues tenemos otros problemas igual de importantes como el desempleo, el abastecimiento de agua potable, la conservación de los bosques, el combate a la pobreza, etcétera. Con este panorama nos damos cuenta de la importancia que juega el comportamiento poblacional y los Censos que se realizan periódicamente.

2. La Jornada Domingo 10 de Septiembre de 2000. P. 29

Vamos a realizar un breve estudio de la ecuación logística de población: Pn = CPn-1*(1 – Pn-1), donde:

C → Condiciones ambientales cuyo rango está entre 0 y 4.
Pn-1 → Población correspondiente a la generación anterior.
Pn → Población Correspondiente a la generación actual.

Esta interesante ecuación se utiliza para realizar pronósticos poblacionales cuyo crecimiento depende de las condiciones ambientales C, esto es:

Si C Є [0, 1], entonces la población se extingue.

Si C Є [1, 2], el crecimiento poblacional tiende a la estabilidad.
Si C Є [2, 4], entonces el crecimiento es oscilatorio.
Si C Є (4, ∞), se genera un caos.

Al realizar el programa, queda lo siguiente:

```
<html><head><title>Comportamiento de la ecuación logística de poblaci&oacute;n
</title></head><body>
<Script Languaje="JavaScript">
    var poblacion1,poblacion=0.95,i=1;
for(i;i<=300;i++){
poblacion1=4*poblacion*(1-poblacion);
poblacion=4*poblacion1*(1-poblacion1);
document.write("Generaci&oacute;n"+i+":"+poblacion+"<br>");
        }
</script>
</body>
</html>
```

Listado 7.6

Esta ecuación, tan conocida en los libros de teoría del caos, nos permite conocer la importancia que tiene el bucle for no sólo para esta ecuación sino para un sinnúmero de aplicaciones científicas que utilizan procesos que se cuentan a veces por millones para verificar los límites de una aplicación real.

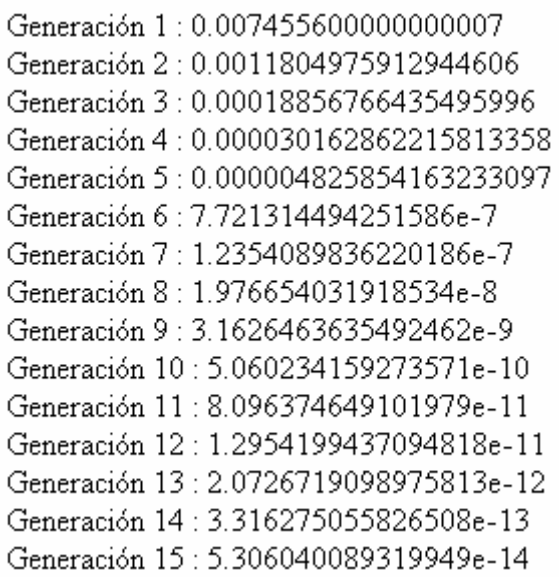


Figura 7.5. Comportamiento de la ecuación logística cuando C=0.4

El método if

Este método JavaScript es un si condicional, es decir es una pregunta, Augusta Ada Byron entendió que las preguntas sólo tenían dos posibles respuestas, Sí o No, algo sumamente trascendente en el campo de la computación, if tiene un condición lógica que puede ser falsa o verdadera, entonces su sintaxis, similar a una pregunta queda del siguiente modo:

Sintaxis:

```
if(  c )  {
Sentencia1;
Sentencia2;
Sentencia3;
                                .
                                .
                                .
                }
```

donde if se ejecutará si c es verdadero,

El método else

else es un complemento del sí condicional, que se ejecutará si C resulta falso. Vamos a ilustrar esto con un ejemplo:

```
<html><head><body>
        <script languaje=”JavaScript”>
var a,b,c,mayor;
   a=10;
```

```
    b=3;
    c=8;
if(a>b){
    mayor=a;
}
else
    {
    mayor=b;
          }

    if(mayor>c)
                    {
    mayor=mayor;
                              }
else
{
mayor=c;
}

document.write("El número mayor es:"+mayor+"<br>");
</script>
</body>
</html>
```

Listado 7.7

En el programa anterior se comparan tres números y de estos se determina cuál es el mayor. Se evalúa la condición en cada uno de los métodos y si esta es verdadera, se ejecuta, cuando la condición es falsa se ejecuta else que significa de lo contrario.

do while. Se ejecuta al menos una vez y significa ejecuta mientras la condición no se cumpla.

Sintaxis:

```
do  {



}
while ( condición )
```

array

Un array es una referencia a un región de la memoria de la computadora que almacena elementos que son del mismo tipo.

Cuando declaramos un arreglo en JavaScript estamos reservando un espacio en la memoria RAM de la computadora, en JavaScript un arreglo es manejado como un objeto, esto es todos los arrays serán un caso de la clase array.

Como habíamos dicho un array es un espacio de memoria con datos del mismo tipo, vamos a declarar un espacio en la memoria para las matrículas de una base de datos de estudiantes:

```
var  matriculas_estudiantes= new Array(4);

matriculas_estudiantes[0] =”87RE00AA”
matriculas_estudiantes[1] =”89ERS0ES”
matriculas_estudiantes[2] =”86ERCILA”
matriculas_estudiantes[3] =”88KJUYYI”
matriculas_estudiantes[4] =”87YTREBI”
```

También lo podemos declarar así:

```
var  matriculas_estudiantes= new Array( );

matriculas_estudiantes[0] =”87RE00AA”
matriculas_estudiantes[1] =”89ERS0ES”
matriculas_estudiantes[2] =”86ERCILA”
matriculas_estudiantes[3] =”88KJUYYI”
matriculas_estudiantes[4] =”87YTREBI”
```

esta también puede ser una forma de hacer explícitos los elementos de los arrays:

```
var  matriculas_estudiantes= new Array(”87RE00AA” , ”89ERS0ES”, ”86ERCILA”,
”88KJUYYI”, ”87YTREBI”);
```

las siguientes expresiones son equivalentes:

matriculas_estudiantes[1] y matriculas_estudiantes[“89ERS0ES”]

Métodos Array

El objeto array tiene tres métodos:

- ✓ join(separador) . Devuelve una cadena que contiene todos los elementos del array separados por el carácter especificado para este fin.
- ✓ reverse( ) Invierte al orden de los elementos de un array.
- ✓ sort( ) . Ordena los elementos de un array.

He aquí un resúmen del tema anterior:

```
<html><head><title>Matrices uno</title></head>

<body bgcolor=”#F8F368”>

<script language=”JavaScript”>
Openindice=new Array();
```

```
        Openindice[0]=”Clavecímbalo”;
        Openindice[1]=”Violín”;
        Openindice[2]=”Piano”;
        Openindice[3]=”Oboe”;
        Openindice[4]=”Violoncelo”;
        Openindice[5]=”Flautín”;
    for(i=0;i<=5;i++){
    document.write(“El índice  “+i+” es: “+Openindice[i]+”<br>”);
    }
</script>
</body>
</html>
```

Listado 7.8

Para añadirle el método por ejemplo sort( ) basta con que lo declaremos como una propiedad de un objeto. Ver **listado 7.9.**

```
<html><head><title>Utilizando m&eacute;todos en arreglos</title></head>
    <body bgcolor=”#F8F368”>
    <script language=”JavaScript”>
    Openorden=new Array();
    Openorden[0]=”Zapata”;
    Openorden[1]=”Gálvez”;
    Openorden[2]=”Lara”;
    Openorden[3]=”Peña”;
    Openorden[4]=”Altamirano”;
    Openorden[5]=”Salinas “;
    Openorden[6]=”Emil”;
        document.write(Openorden.sort().join()+”<br>”);
</script>
        </body>
</html>
```

Listado 7.9

Vamos a ver unos listados de arreglos y manejo de archivos muy comunes en la programación de Internet:

Vamos a realizar un diseño de página Web para dar de alta a estudiantes y que sean guardados en un arreglo:

Es recomendable realizar un bosquejo en papel de cómo va a quedar nuestra base de datos Web, o bién realizar un programa Script piloto de lo que queremos y con base a esto, involucrar a los usuarios y los programadores de nuestra idea. Ahora bién la programación no necesariamente es la parte más difícil, pienso que la parte más difícil, es la pregunta más sencilla, es decir, diseñar un software que sea amigable, eficiente y reutilizable. Ver figura 7.6.

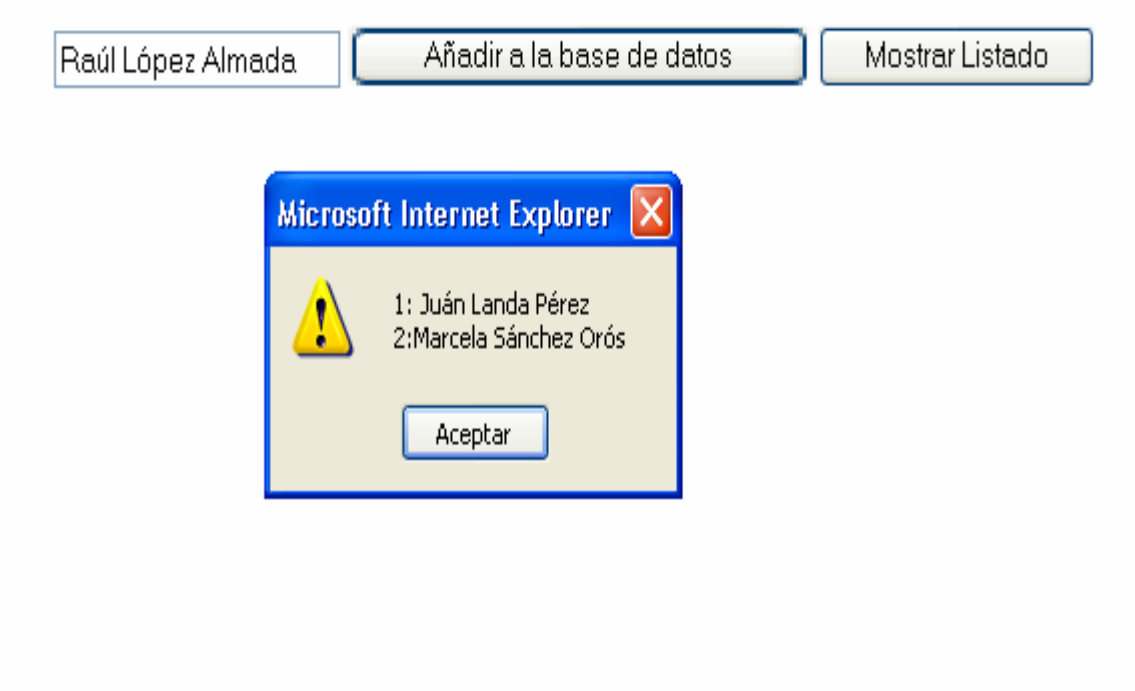


Figura 7.6

Como habíamos mencionado anteriormente, un formulario puede tener el papel de receptor y emisor de datos, estos receptores y emisores de datos los vamos a tomar como los eventos de la computadora, a cada evento le vamos a asociar una función, es decir, le vamos a dar inteligencia a los eventos, vamos a introducir en la entrada de texto de formulario los estudiantes y van a hacer almacenados en un arreglo y se van a desplegar los estudiantes al oprimir un botón. Ver listado 7.10.

```
<CENTER>
<FORM name=”historial”>
<INPUT name=”command” type=”text” value=” “>
<INPUT type=”button” value=”A&ntilde;adir a la base de datos”
onclick=”archivo(document.historial.command.value)”>
<INPUT name=”historial” type=”button” value=”Mostrar Listado” onclick=”imprimir()”>
</FORM>
<P>
<SCRIPT language=”JavaScript”>
```

```
function Generaarreglo( n ) {
if( n <= 0 ) {
this.length = 0;
return this;
}
this.length = n;
for( var i = 1; i <= n; i++ ) {
this[ i ] = 0;
}
return this;
}
var historial = new Generaarreglo( 15 );
var index = 0;
var cmmnd = 1;
function archivo( sTR ) {
var i;
if( index >= historial.length ) {
for( i = 1; i < historial.length; i++ )
historial[i-1] = historial[i];
index = historial.length - 1;
}
historial[ index ] = cmmnd + ":" + sTR;
++cmmnd;
++index;
document.historial.command.value= » »;
}

function imprimir() {
var allCmmnds, i;
allCmmnds = "";
for( i = 0; i < index; i++ )
allCmmnds += historial[i] + "\n";
alert( allCmmnds );
}

</SCRIPT>
```

Listado 7.10

Desarrollo de una agenda

Las agendas de trabajo, de empleados, o de cualquier tipo son importantes aplicaciones en las bases de datos, en las oficinas la agenda es un instrumento imprescindible, el objetivo de una agenda es tener un calendario personalizado de nuestras actividades cotidianas. En

casi todas las herramientas Office es posible encontrar este tipo de instrumentos. Otros prefieren programarlas para hacerlas más personalizadas. Aquí vamos a realizar una agenda de Intranet, nuestra agenda está pensada para personas que acostumbran a realizar viajes de negocios y sería ágil contar con una agenda electrónica que tenga acceso a las bases de datos de interés de los empleados de la compañía., esto es, que nuestra agenda electrónica pueda conectarse con la base de datos de los empleados de la compañía, y que nos dé la posibilidad de modificar los datos de los empleado que tenemos a nuestro cargo. Este podría ser un bosquejo:

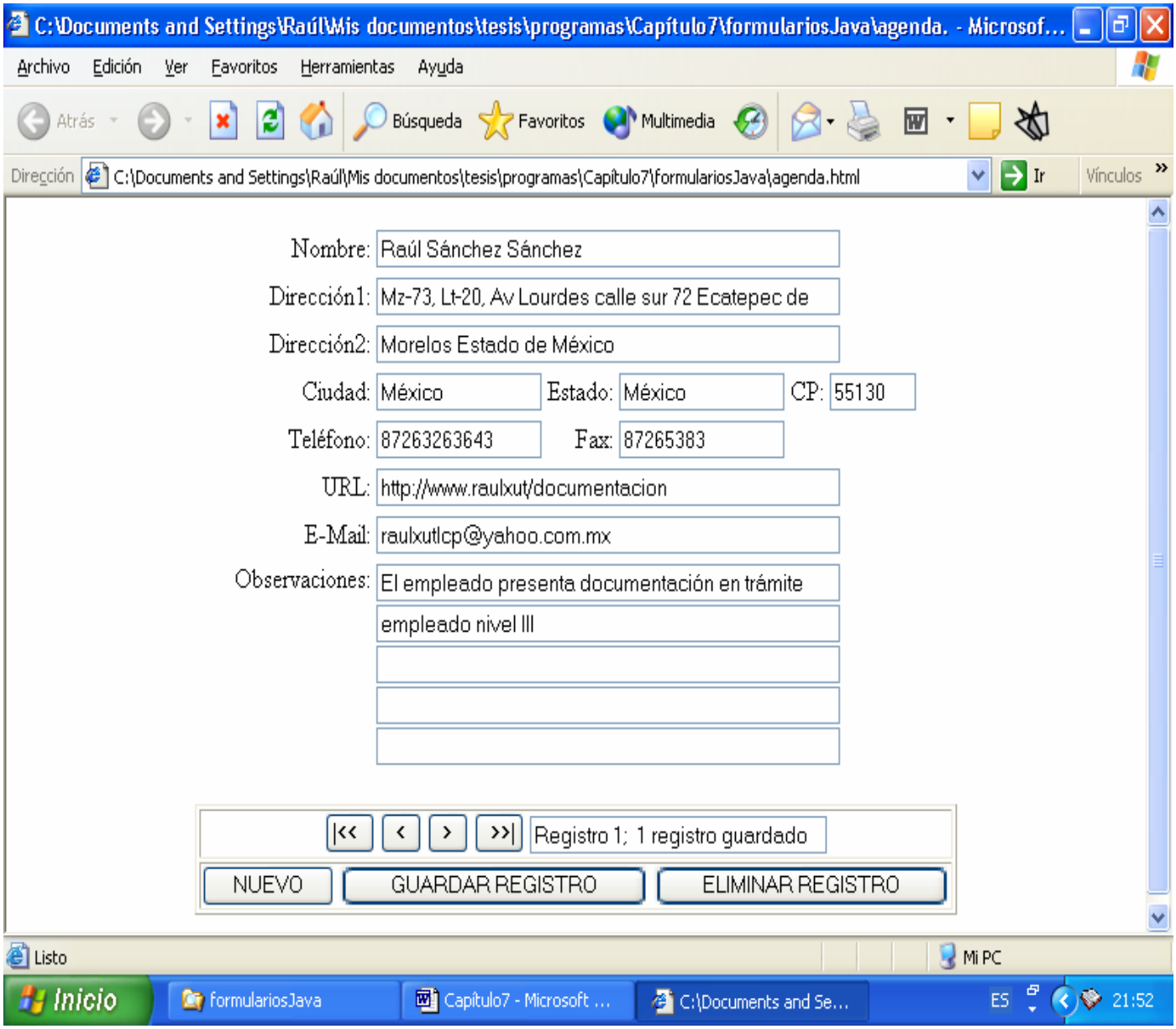


Figura 7.7

En la agenda anterior se pueden modificar los registros de nuestra base de datos desde una agenda electrónica. Esto nos da la comodidad de poder realizar nuestros compromisos de negocios y resolver los pendientes que tenemos en nuestra oficina. Como podemos ver, los controles de nuestra agenda son los usuales cuando realizamos altas, bajas y cambios desde cualquier oficina administrativa donde se maneja los datos de grande volúmenes de registros. Este es el código.

```
<SCRIPT LANGUAGE=”JavaScript”>
var arrRegistro = new Array();
var arrCookie = new Array();
var recCount = 0;
var strRecord=””;
expireDate = new Date;
expireDate.setDate(expireDate.getDate()+365);
function cookieVal(cookieName) {
thisCookie = document.cookie.split(“; “)
for (i = 0; i < thisCookie.length; i++) {
if (cookieName == thisCookie[i].split(“=”)[0]) {
return thisCookie[i].split(“=”)[1];
  }
}
return 0;
}
function loadCookie() {
if(document.cookie != “”) {
arrRegistro = cookieVal(“Records”).split(“,”);
currentRecord();
  }
}
function setRec() {
strRecord = “”;
for(i = 0; i < document.frm1.elements.length; i++) {
strRecord = strRecord + document.frm1.elements[i].value + “:”;
}
arrRegistro[recCount] = strRecord;
document.frm2.add.value = “  NUEVO  “;
document.cookie = “Records=”+arrRegistro+”;expires=” + expireDate.toGMTString();
}
function newRec() {
switch (document.frm2.add.value) {
case “  NUEVO  “ :
    varTemp = recCount;
    for(i = 0; i < document.frm1.elements.length; i++) {
    document.frm1.elements[i].value = “”
  }
    recCount = arrRegistro.length;
```

```
    document.frm2.add.value = “CANCEL”;
    break;
case “CANCEL” :
    recCount = varTemp;
    document.frm2.add.value = “  NUEVO  “;
    currentRecord();
    break;
  }
}
function countRecords() {
document.frm2.actual.value = “Registro “ + (recCount+1)+”;  “+arrRegistro.length+”
registro guardado”;
}
function delRec() {
arrRegistro.splice(recCount,1);
navigate(“anterior”);
setRec();
}
function currentRecord() {
if (arrRegistro.length != “”) {
strRecord = arrRegistro[recCount];
currRecord = strRecord.split(“:”);
for(i = 0; i < document.frm1.elements.length; i++) {
document.frm1.elements[i].value = currRecord[i];
     }
  }
}
function navigate(control) {
switch (control) {
case “primero” :
    recCount = 0;
    currentRecord();
    document.frm2.add.value = “  NUEVO  “;
    break;
case “last” :
    recCount = arrRegistro.length - 1;
    currentRecord();
    document.frm2.add.value = “  NUEVO  “;
    break;
case “siguiente” :
    if (recCount < arrRegistro.length - 1) {
    recCount = recCount + 1;
    currentRecord();
    document.frm2.add.value = “  NUEVO  “;
  }
```

```
break;
case “ultimo” :
    if (recCount > 0) {
    recCount = recCount - 1;
    currentRecord();
   }
    document.frm2.add.value = “  NUEVO  “;
    break;
    default:
   }
}

if (!Array.prototype.splice) {
function array_splice(ind,cnt) {
if (arguments.length == 0) return ind;
if (typeof ind != “number”) ind = 0;
if (ind < 0) ind = Math.max(0,this.length + ind);
if (ind > this.length) {
if (arguments.length > 2) ind = this.length;
else return [];
}
if (arguments.length < 2) cnt = this.length-ind;
cnt = (typeof cnt == “number”) ? Math.max(0,cnt) : 0;
removeArray = this.slice(ind,ind+cnt);
endArray = this.slice(ind+cnt);
this.length = ind;
for (var i = 2; i < arguments.length; i++) {
this[this.length] = arguments[i];
}
for(var i = 0; i < endArray.length; i++) {
this[this.length] = endArray[i];
}
return removeArray;
}
Array.prototype.splice = array_splice;
}
recCount = 0;
loadCookie();
countRecords();
</script>
</HEAD>
<BODY>

<center>
<form name=”frm1”>
```

```
<table align=”center” resize=”none” border=”0”>
<tr>
<td align=”right”>Nombre:</td>
<td colspan=”5”><input type=”box” name=”name” size=”49”></td>
</tr>
<tr>
<td align=”right”>Direcci&oacute;n1:</td><td colspan=”5”><input type=”box”
name=”address” size=”49”></td></tr>

<td align=”right”>Direcci&oacute;n2:</td><td colspan=”5” align=”left”><input
type=”box” name=”address2” size=”49”></td></tr>

<tr>
<td align=”right”>Ciudad:</td>
<td><input type=”box” name=”city” size=”15”></td>
<td>Estado:</td><td><input type=”box” name=”state” size=”15”></td>
<td>CP:</td><td><input type=”box” size=”6” name=”zip”></td>
</tr>
<td align=”right”>Tel&eacute;fono:</td>
<td align=”left”><input type=”box” name=”phone” size=”15”></td>
<td align=”right”>Fax:</td><td align=”left”><input type=”box” name=”fax”
size=”15”></td>

</tr>
<tr>
<td align=”right”>URL:<td colspan=”5” align=”left”><input type=”box” name=”address”
size=”49”></td></tr>

<td align=”right”>E-Mail:<td colspan=”5” align=”left”><input type=”box” name=”email”
size=”49”></td></tr>

<tr><td align=”right” valign=”top”>Observaciones:</td>
<td colspan=”5” align=”left”><input type=”box” name=”comment1” size=”49”><br>
<input type=”box” name=”comment2” size=”49”><br>
<input type=”box” name=”comment3” size=”49”><br>
<input type=”box” name=”comment4” size=”49”><br>
<input type=”box” name=”comment5” size=”49”>
</td></tr></table>
</form>
<form name=”frm2”>
<table align=”center” border=”1” resize=”none”><tr><td align=”center”>
<input type=”button” name=”primero” value=”|<< “
onClick=”navigate(‘primero’);countRecords()”>

<input type=”button” name=”anterior” value=” < “
onClick=”navigate(‘anterior’);countRecords()”>

<input type=”button” name=”siguiente” value=” > “
onClick=”navigate(‘siguiente’);countRecords()”>
```

```
<input type=”button” name=”ultimo” value=”  >>|”
onClick=”navigate(‘ultimo’);countRecords()”>

<input type=”box” name=”actual” size=30></td></tr>

<tr><td align=”center”><input type=”button” name=”add” value=”  NUEVO  “
onClick=”newRec();countRecords()”>

<input type=”button” name=”set” value=”GUARDAR REGISTRO”
onClick=”setRec();countRecords()”>

<input type=”button” name=”del” value=”ELIMINAR REGISTRO”
onClick=”delRec();countRecords()”></td></tr></table>

</form>
</center>
```

Listado 7.11

7.4. Visual Basic Script.

Es un lenguaje de programación para Internet que se define a partir Visual Basic, y junto con Java Script son los lenguajes que se utilizan en forma masiva en Internet. Ambos lenguajes se pueden utilizar de forma conjunta en el mismo navegador explorer. Ver listado 7.12:

```
<html><head><title>Mis combinaciones</title></head>

   <body>

<center><font size=”6” color=”blue” face=”Comic Sans MS”>Ejemplo de
muestra</font></center>

   <script language=”JavaScript”>

   function saludo()

            {
   alert(“Este es un saludo en JavaScript”);

            }
</script>


   <script language=”VBScript”>

   function saludo1()

   msgbox(“Este es un saludo en Visual Basic Script”)

   end function

</script>
<center><br><br><br>
<input type=”button” onClick=”saludo()” value=”Saludo en JavaScript”><br>
<input type=”button” onClick=”saludo1()” value=”Saludo en Visual Basic Script”>
```

```
</center>
</body>
</html>
```

Listado 7.12

A continuación presentamos el resultado en un navegador:

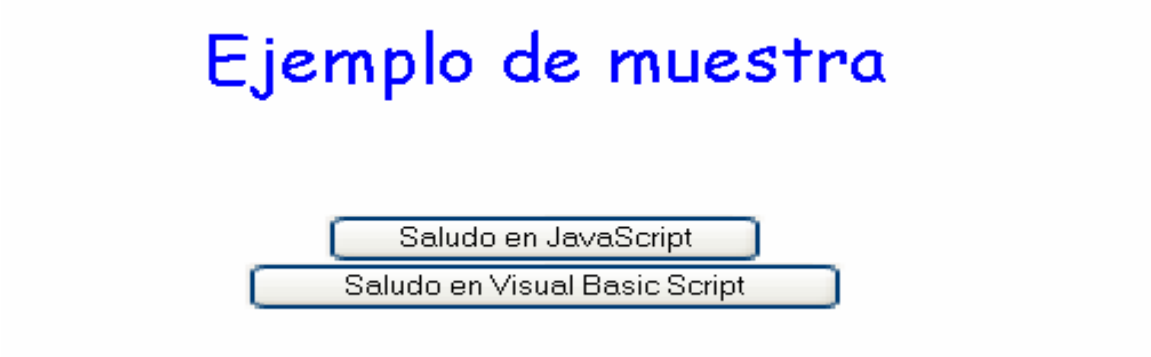


Figura 7.8

Formularios.

Los formularios en Visual Basic Script son similares a los que se utilizan en JavaScript, como siguen con la filosofía de la programación orientada a eventos y aparte hacen manejo de objetos, que hay que aclarar no es lo mismo que la programación Orientada a Objetos.

Orientación a Objetos.

Ninguno de los dos lenguajes puede ser considerado orientado a objetos. JavaScript soporta los paradigmas de identidad y encapsulación, pero no los de herencia y polimorfismo (Ver capítulo 4). Visual Basic no soportará ninguno de estos. Para definir un objeto en JavaScript, se debe escribir una función especial que dará nombre a la clase así como a las propiedades y métodos a través de la palabra clave This, que hace referencia al objeto. Por ejemplo, se define la clase Estudiante, en JavaScript:

```
function  Estudiante( Nombre, Edad, Matricula )
                        {

this.Nombre = Nombre;
this.Edad=Edad;
this.Matricula = Matricula;
                        }
```

Para instanciar un estudiante de la clase Estudiante se realiza lo siguiente:

datos_Personales= new Estudiante(“Andrés Manuel Jiménez”, “16, “, “88REA987”);

Las principales aplicaciones de los lenguajes Scripts en el entorno de un navegador son:

- ✓ Validación previa de los datos.
- ✓ Proporcionar o requerir información de los usuarios.
- ✓ Inicialización y control de los componentes Active X o de Applets de Java.
- ✓ Ejecución de cálculos.
- ✓ Utilización de Cookies.
- ✓ Añadir interactividad a la página respondiendo a las acciones de los usuarios.
- ✓ Modificar dinámicamente la apariencia de la página.
- ✓ Facilitar la navegación en varias instancias de un navegador simultáneamente.
- ✓ Creación de objetos reutilizables a partir del uso de Scriptlets en explorer 4.0 o superior.

Cómo averiguar un tipo de variable

Esto se realiza a través de la función VarType. Esta función toma como argumento el nombre de la variable y devuelve un número identificando al tipo al que pertenece ésta. Ver tabla 7.2

Constante	Valor	Significado
vbEmpty	0	La variable está sin inicializar
VbNull	1	Variable nula
vbInteger	2	Entero de dos bytes
vbLong	3	Entero largo(cuatro bytes)
vbSingle	4	Número de precisión simple
vbDouble	5	Número de presición doble
vbCurrency	6	Número de moneda
vbDate	7	Fecha
vbString	8	Cadena de texto
vbObject	9	Objeto
vbError	10	Número de error generado por VBScript
vbBoolean	11	Variable de tipo Verdadero/Falso
vbVariant	12	Tipo variante(en arrays)
vbDataObject	13	Un objeto de acceso a datos
vbByte	14	Un número de tipo Byte.
vbArray	8192	La variable contiene a una matriz

Tabla 7.2

En cualquier lenguaje de programación a veces es necesario identificar el tipo de variable con el que se está trabajando, esto es muy común en el uso de formularios, por ejemplo, cuando se nos obliga a introducir datos de tipo entero cuando introducimos la edad, o de tipo carácter cuando introducimos nuestro nombre, o bién cuando se realiza una operación

al introducir un tipo de dato, por ejemplo una fecha, si introducimos una tarjeta y la fecha está vencida, entonces se manda un mensaje de que la tarjeta se tiene que actualizar, o bién si la fecha coincide, entonces se nos da acceso al sistema. En Visual Basic se hace a través de funciones, para averiguar si una variable pertenece o no a un determinado tipo de dato, presentamos la tabla 7.3:

Función	Descripción
IsDate	Determina si el contenido de la variable es una fecha o una cadena que puede convertirse a una fecha.
IsEmpty	Determina si una variable es o no nula.
IsNumeric	Especifica si una variable contiene una expresión numérica (Las fechas no se consideran números). Es quizá la más usada en este grupo.
IsObject	Determina si la variable contiene una referencia válida a un objeto de automatización.

Tabla 7.3. Funciones que identifican un tipo de variable de interés.

Conversión de un tipo de dato a otro.

Las operaciones con fechas y caracteres son operaciones con datos de tipo entero, por lo menos es así como lo entiende el software por lo tanto que cuando realizamos una operación con una fecha, el resultado es un número entero, aquí es donde se hace necesario utilizar funciones de conversión, para esta situación, lo que necesitamos es una función que convierta una expresión cualquiera a un dato de tipo fecha, es decir, CDate. Ver tabla 7.4.

Buena parte de las operaciones que requieren procesamiento de bases de datos, requieren en buena medida de la conversión de los datos.

Función	Descripción
CBool	Convierte una expresión distinta de cero en Trae y al cero lo transforma en False.
Cbyte	Convierte una expresión al tipo Byte.
CCur	Convierte una expresión a tipo moneda.
CDate	Convierte una expresión en un dato tipo fecha.
CDbl	Convierte una expresión en tipo numérico de coma flotante y de doble precisión.
CInt	Convierte una expresión a entero.
CLng	Convierte una expresión en un entero largo.
CSng	Convierte una expresión en un número de coma flotante de precisión sencilla.
CStr	Convierte cualquier expresión válida en una cadena de texto.

Tabla 7.4. Funciones que convierten un tipo de dato en otro.

He aquí un listado de algunas de las funciones anteriores:

```
<html><head><title>Mis funciones de conversi&oacute;n</title></head>
<body>
<script language=”VBScript”>
sub expresion_onClick()
    msgbox “El número en tipo bool es:”&CBool(numeral.value)
    msgbox “El número en tipo moneda es:”&CCur(numeral.value)
    msgbox “El núммero en en tipo fecha es:”&CDate(numeral.value)
    msgbox “El número en tipo flotante y doble presición es:”&CDbl(numeral.value)
    msgbox “El número en tipo y a entero es:”&CInt(numeral.value)
    msgbox “El número en tipo entero largo:”&CLng(numeral.value)
    msgbox “El número en tipo coma flotante y sencilla:”&CSng(numeral.value)
    msgbox “El número en tipo cadena de texto:”&CStr(numeral.value)
end Sub
```

```
</script>
<font face=”Verdana” color=”red” size=”5”>N&uacute;mero:</font>
<input type=”text” size=”9” name=”numeral”>
<input type=”button”  value=”evaluar” name=”expresion” >
</body>
</html>
```

Listado 7.13

El programa funcionará correctamente siempre y cuando intrduzcamos datos, que estén dentro de los rangos aceptados por las funciones. Le recomiendo que pruebe este programa.

Diseño y análisis de formularios.

Los formularios en VBScript son similares a los que se usan en JavaScript. Aquí vamos a reproducir en VBScript el sistema de ecuaciones con dos incógnitas:

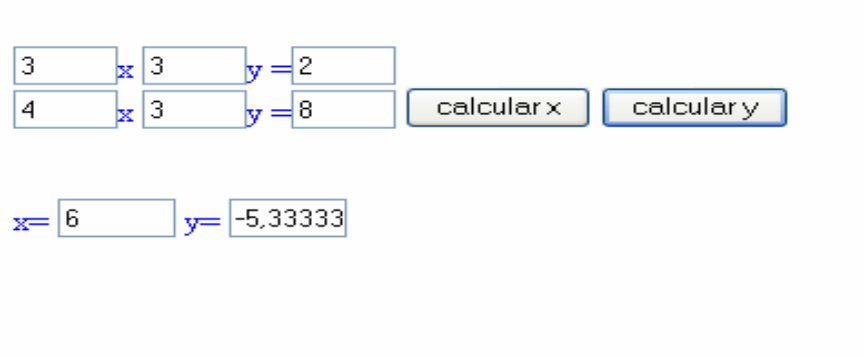


Figura 7.9

El análisis y el diseño del Script tiene la misma forma ver listado 7.14.

```
<html><head><title>Sistema de ecuaciones con dos inc&oacute;gnitas</title><head>
<body bgcolor=”#ffffff”>
<script language=”VBScript”>
sub ecuacion_onclick()

x.value=(cdbl(b1.value)*cdbl(a22.value)-
cdbl(b2.value)*cdbl(a12.value))/(cdbl(a11.value)*cdbl(a22.value)-
cdbl(a21.value)*cdbl(a12.value))

end sub
sub ecuacion1_onclick()
y.value=(cdbl(a11.value)*cdbl(b2.value)-
cdbl(a21.value)*cdbl(b1.value))/(cdbl(a11.value)*cdbl(a22.value)-
cdbl(a21.value)*cdbl(a12.value))

end sub

</script>
<br><input type=”text” size=”4” name=”a11”><font size=”3” color=”blue”>x</font>
<input type=”text” size=”4” name=”a12”><font size=”3” color=”blue”>y</font>
<font size=”3” color=”blue”>=</font><input type=”text” size=”4” name=”b1”>
<br><input type=”text” size=”4” name=”a21”><font size=”3” color=”blue”>x</font>
```

```
<input type=”text” size=”4” name=”a22”><font size=”3” color=”blue”>y</font>
<font size=”3” color=”blue”>=</font><input type=”text” size=”4” name=”b2”>
<input type=”button” name=”ecuacion” value=”calcular x”>
<input type=”button” name=”ecuacion1” value=”calcular y”>
<br><br><br><font size=”3” color=”blue”> x= </font><input type=”text” size=”5”
name=”x”>

<font size=”3” color=”blue”> y= </font><input type=”text” size=”5” name=”y”>

</body>
</html>
```

Listado 7.14

Como podemos apreciar en el programa anterior, en VBScript no es necesario declarar las variables, pero esto paradójicamente genera un problema, como es tan fácil de programar, se corre el riesgo de equivocarse. Es por eso que VBScript ofrece la posibilidad de declarar las variables con dim, a través de Option Explicit, que le obliga al usuario a declarar las variables.

Sentencias de control de flujo.

Sentencias de control de flujo. Son similares a las que se implementan en Java Script, es por eso que ya no las repetiremos pero si es necesario las explicaremos en los programas que vamos a desarrollar, para la interactividad con las bases de datos que va a manejar el usuario. Pero lo vamos a ilustrar con ejemplos representativos que nos ilustre lo que estamos explicando.

Vamos a diseñar un juego en VBScript, el problema del granjero.

Problema. Un granjero tiene tres pertenencias: Un lobo, una gallina y un saco de maíz. El problema consiste en que el granjero debe de pasar del otro lado del río con sus pertenencias, y para esto cuenta con una barca con capacidad de transportar a dos tripulantes, al granjero y a una de sus pertenencias. A continuación se ilustra una posible solución:

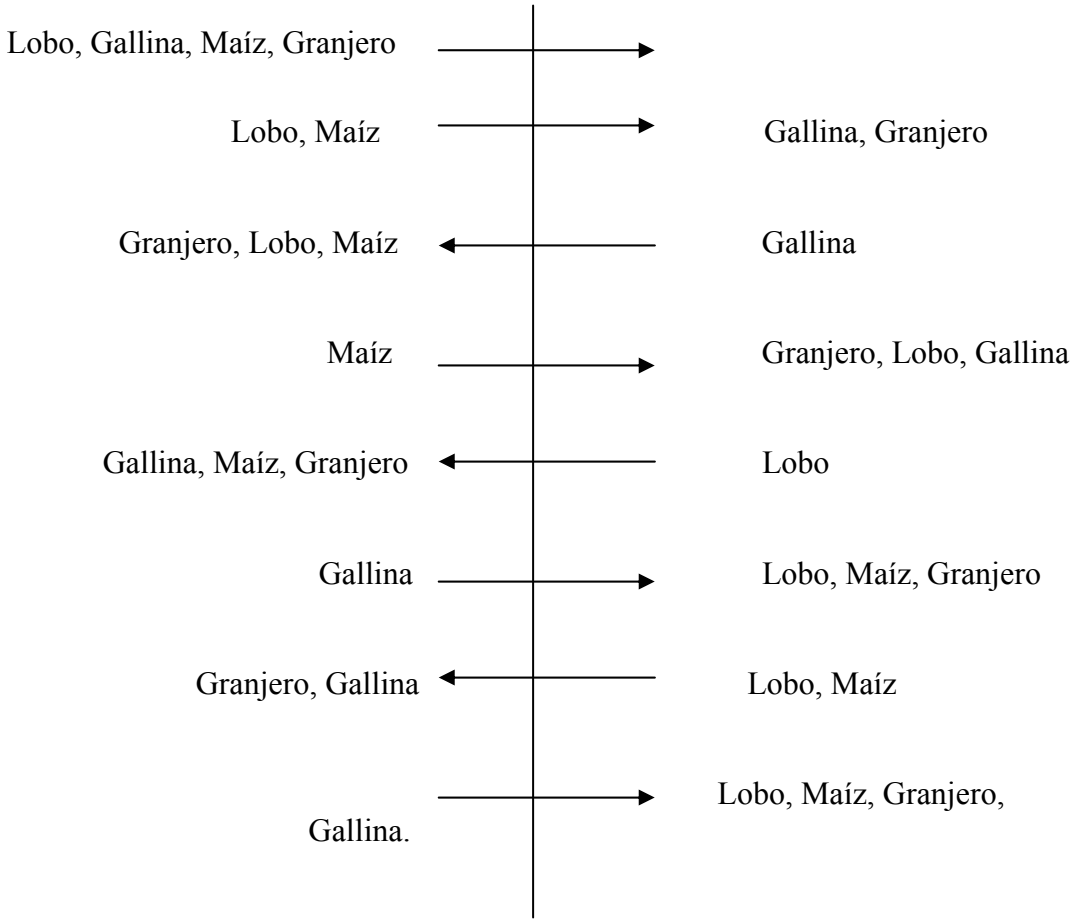


Figura 7.10

Lo primero que tenemos que analizar es que tenemos que escoger entre las tres opciones: lobo, gallina, maíz y para poder realizarlo, vamos a recurrir a un flujo **case,** ya que el case nos permite seleccionar entre varias opciones, ahora bién, cada que ganemos, tenemos que seguir seleccionando una de las tres posibles opciones, hasta terminar el juego. Ver listado 7.15.

```
<html><head><title>La granja</title></head>
    <body bgcolor=”#ffffff”>
<script language=”VBScript”>
opcion=inputbox(“a.-Lobo b.-Gallina c.-Maíz”)
select case opcion
case “a”
msgbox “Incorrecto...La Gallina Se Come el Maíz”
case “b”
msgbox “Correcto”
opcion=inputbox(“a.-Lobo b.-Gallina c.-Maíz”)
select case opcion
```

```
case “a”
msgbox “Correcto”
opcion=inputbox(“a.-Lobo b.-Gallina c.-Maíz”)
select case opcion
case “a”
msgbox “Incorrecto...Ya te llevaste al lobo”
case “b”
msgbox “Correcto”
opcion=inputbox(“a.-Lobo b.-Gallina c.-Maíz”)
select case  opcion
case “a”
msgbox “Incorrecto... El lobo ya está del otro lado del río”
case “b”
msgbox “Incorrecto...Repites la acción”
case “c”
msgbox “Correcto”
opcion=inputbox(“a.-Lobo b.-Gallina  c.-Maíz “)
select case opcion
case “a”
msgbox(“Incorrecto... El lobo ya está del otro lado del río”)
case “b”
msgbox(“Correcto...Ganaste el juego...”)
case “c”
msgbox(“Incorrecto...Ya te llevaste el maíz”)
end select
end select
case “c”
msgbox “Se complica más el problema”
end select
case “b”
msgbox “Incorrecto...Ya te llevaste a la Gallina”
case “c”
msgbox “Incorrecto...Se complica más”
end select
case “c”
msgbox “Incorrecto...El Lobo Se Come a la Gallina”
end select
</script>
</body></html>
```

Listado 7.15

Formularios en profundidad.

En casi todos los formularios que llenamos por Internet, se realiza algo que se conoce como validación de los datos, ventaja que obliga a los usuarios a introducir los datos correctamente, es común equivocarse cuanto se introducen por ejemplo las fechas en forma incorrecta, etc. En la tabla 7.2 existen unas funciones que determinan el tipo de dato

introducido. Nos vamos a valer de algunas de estas funciones para determinar la validez de los datos que vayamos a introducir desde el formulario.

En este ejemplo vamos a validar una entrada de tipo numérico y si introducimos un dato, no numérico, que no lo haga notar el formulario.

```
<html><head><title>Mi primera validaci&oacute;n</title></head>
    <body background=”#ffffff”>
    <script language=”VBScript”>
    sub mivalidacion_onclick()
dim error
if not isnumeric(salario.value) then
error=true
msgbox “El Salario es un valor numérico”
end if
    if not error then
    msgbox “Tu salario es:”&salario.value
    end if
    end sub
    </script>

<font size=”4” color=”red” face=”Verdana”>Salario:</font>
    <input type=”text” size=30 name=”salario”><br>
<input type=”button” name=”mivalidacion” value=” Enviar “>
</body><html>
```

Listado 7.16

en el formulario anterior, la estrategia consiste en utilizar una compuerta lógica **not**, con una función de verificación en este caso **isnumeric,** not es un operador de negación, y lo vamos a utilizar estratégicamente para negar todas las afirmaciones que realicemos con las funciones de verificación. Cuando realicemos un formulario completo, esto podría quedar como sigue:

```
<html><head><title>Mi primera validaci&oacute;n</title></head>
    <body background=”#ffffff”>
    <script language=”VBScript”>
    function mivalidacion()

if not isnumeric(salario.value) then
error=true
msgbox “El Salario es un valor numérico”
end if
        if not isdate(fecha.value) then
error=true
```

```
msgbox “Debes introducr una fecha”
        end if
if not error then
msgbox “Tu nombre es:”&nombre.value&_
apellidos.value&chr(13)&”Tu salario es:”&salario.value&chr(13)_
&”Tu fecha de nacimiento es:”&fecha.value&chr(13)_
&”Dirección:”&direccion.value

    end if

    end function

    </script>

<center><font size=”6” color=”blue” face=”Comic Sans MS”>Fomulario de
inscripci&oacute;n</font></center><br><hr border=”4” color=”orange”><br>

<font size=”4” color=”red” face=”Verdana”>Nombre:</font>

    <input type=”text” size=20 name=”nombre”><br><br>

<font size=”4” color=”red” face=”Verdana”>Apellidos:</font>

    <input type=”text” size=30 name=”apellidos”><br><br>

<font size=”4” color=”red” face=”Verdana”>Fecha de nacimiento:<font>

    <input type=”text” size=”12” name=”fecha”><br><br>

<font size=”4” color=”red” face=”Verdana”>Direcci&oacute;n:</font>

    <input type=”text” size=”60” name=”direccion”><br><br>


<font size=”4” color=”red” face=”Verdana”>Salario:</font>

    <input type=”text” size=10 name=”salario”><br><br>

<input type=”button” onclick=”mivalidacion” value=” Enviar “>
</body><html>
```

Listado 7.17

En el listado anterior estamos estableciendo el uso de los formularios y algunas verificaciones importantes. Ver figura 7.11.

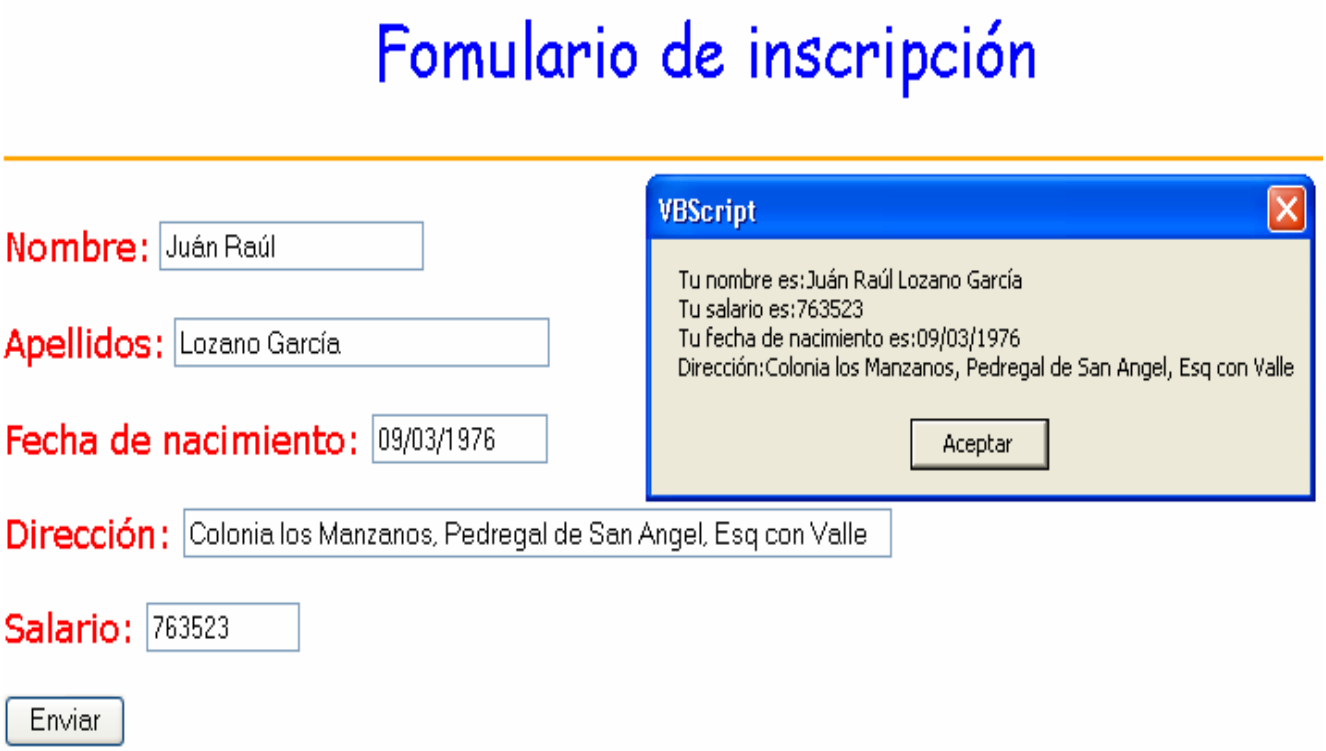


Figura 7.11

Hay varias formas de introducir datos en los formularios, una de ellas son las casillas de verificación tipo checkbox, estas casillas solo conocen dos posibles, respuestas, sí o no. La ventaja de utilizar este tipo de de casillas es que podemos realizar de forma cómoda las selecciones tales como la elección de algún curso, turno, materia etc. El listado siguiente es un ejemplo de lo que queremos explicar. Ver listado 7.18.

```
<html><head><title>Formulario de verificaci&oacute;n</title></head>
    <body>
    <script language=”VBScript”>
    function envio_datos()

        If envio.checkbox1.checked=true then
        a=a&”,”&”Acces”
        End If

        If envio.checkbox2.checked=true then
        a=a&”,”&”Excel”
        End If

        If envio.checkbox3.checked=true then
        a=a&”,”&”Word”
        End If
```

```
        If envio.checkbox4.checked=true then
        a=a&”,”&”Windows”
        End If

        If envio.checkbox5.checked=true then
        a=a&”,”&”Visual Basic”
        End If

    msgbox a
end function
</script>

<form name=”envio”>
Inscripción al curso de:<BR> <br>

          <INPUT id=checkbox1 name=checkbox1 type=checkbox>Access  
          <INPUT  id=checkbox2 name=checkbox2 type=checkbox>Excel  
          <INPUT  id=checkbox3 name=checkbox3 type=checkbox>Word  
          <INPUT id=checkbox4 name=checkbox4 type=checkbox>Windows  
          <INPUT  id=checkbox5 name=checkbox5 type=checkbox>Visual Basic</P>
<p><input  onclick=envio_datos()  id=button1 name=button1 type=button  value=Enviar>
    <input id=reset1 name=reset1  type=”reset” value=Reset></p>
</form>
</body>
</html>
```

Listado 7.18

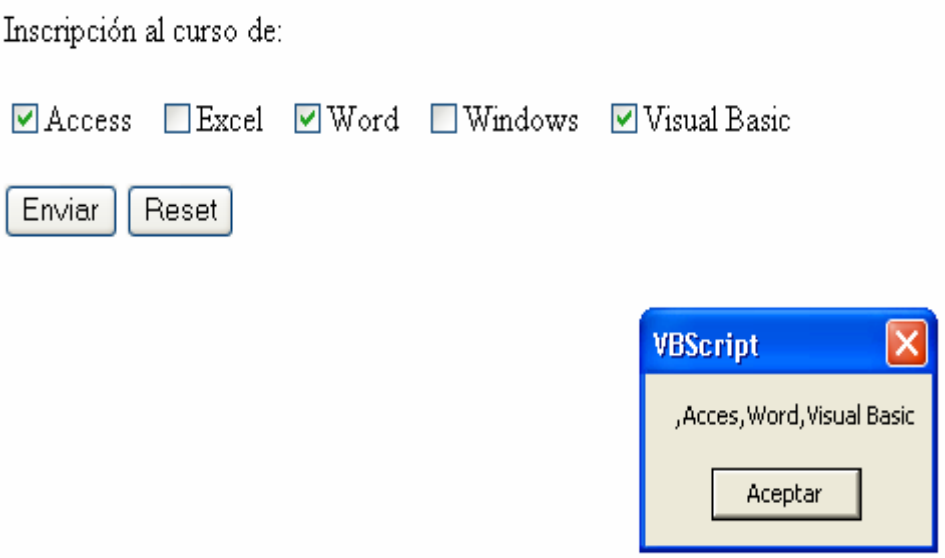


Figura 7.11

En la figura anterior se puede apreciar que se asocia el nombre del formulario con el nombre de la entrada de verificación, por ejemplo si tenemos un formulario de nombre envio y una entrada de verificación de nombre checkbox2, entonces cuando se lo asociamos a un bucle for quedaría del siguiente modo: envio.checkbox2.checked = true, podemos asociarle checked para evaluar si la expresión resulta cierta o falsa. Ver listado 7.19.

```
<html><head><title>Verificaci&oacute;n circular</title></head>
<body>
<script language=”VBScript”>
    function envio_datos()
    IF envio.radio1(0).checked=true THEN
    msgbox “Matutino”
    ELSE
    msgbox “Vespertino”
    end if
    end function
</script>
    <form name=”envio”>
    <p></LABEL>Turno:</p>
    <INPUT id=radio1 name= radio1 type=radio value=0>Matutino
    <INPUT id=radio1 name= radio1 type=radio value=1>Vespertino

<p><input  onclick=envio_datos()  id=button1 name=button1 type=button  value=Enviar>
        <input id=reset1 name=reset1  type=”reset” value=Reset></p>
</form>
        </BODY>
```

</HTML>

Listado 7.19

Bucles for – next.

Vamos a utilizar el contador y una variable que vaya almacenando los valores que se vayan generando en los contadores de un bucle for. Para realizar esto voy a utilizar un bucle **for** y una variable **cuenta** que vaya almacenando los valores. En el listado siguiente vemos como la variable **cuenta** va almacenando los valores del conteo. Ver listado 7.18.

```
<html><head><title>Mi flujo uno</title><head>
   <body>
<script language=”VBScript”>
        function cuentame_como_paso()
for i=1 to 10
cuenta=cuenta&i&chr(13)
        next
msgbox cuenta
end function



</script>
<input type=”button” onClick=”cuentame_como_paso()” value=”contar”>
</body>
</html>
```

Listado 7.20

Resalté con negritas el código que necesitamos para almacenar los valores de i, y chr(13) es un salto de línea. Este proceso lo introduje en un evento de ratón onClick(), para comodidad. Los procesos iterativos son muy útiles en las bases de datos y por eso hice un breve paréntesis en la utilización de los bucles for y variables para almacenar los valores de los procesos. Vamos a realizar otros dos ejemplos con for para ir familiarizándonos con los bucles y las variables que almacenan procesos. Ver listado

```
<html><head><title>cadena num&eacute;rica</title></head>
<body bgcolor=”#ffffff”>
   <script language=”VBScript”>
   sub cadena_de_numeros_onclick()
dim cuenta
for i=0 to 9
for j=0 to 9
cuenta= cuenta&i&j&” “
next
```

```
cuenta=cuenta&chr(13)
next
msgbox cuenta
end sub
                                                    </script>
<input type=”button” name=”cadena_de_numeros” value=”serie numérica”>
</body>
</html>
```

Listado 7.21

Aquí mostramos otra forma del mismo programa pero anidando el bucle for. La corrida queda del siguiente modo:

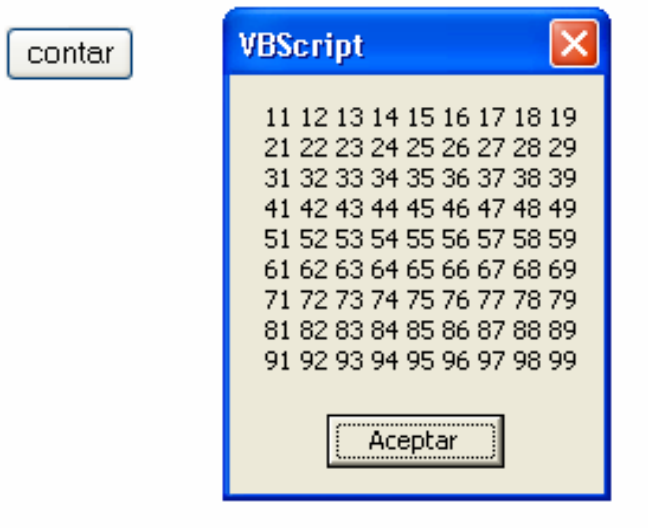


Figura 7.12

El siguiente listado tiene tres niveles de anidamiento:

```
<html><head><title>Mi Web</title></head>
        <body>
<script language=”VBScript”>
    function flujo_3()
    for i=0 to 9
    for j=0 to 9
    for k=1 to 9
    contador=contador&i&j&k&” “
    next
    contador=contador&chr(13)
```

```
    next contador=contador&chr(13)
    next
    msgbox contador
    end function
    </script>

<input type=”button” onClick=”flujo_3()” value=”contar”>
</body>
</html>
```

Listado 7.22

Matrices

Una matriz es un conjunto de elementos colocados en forma adyacente en la memoria de la computadora y son referidos por un nombre común. Ahora, cada uno de estos elementos de la matriz es referido a través un número que se le llama subíndice tal y como se indica a continuación. Ver figura 7.12.

Las matrices se pueden clasificar básicamente en dos categorías:

Estabilidad en el tamaño: *Matriz estática y matriz dinámica.* Una matriz es estática cuando su tamaño permanece inalterable a lo largo de todo el código. Si una matriz cambia el tamaño y lo puede adaptar a las condiciones del código, entonces estamos hablando de una matriz dinámica.

Según sus dimensiones: Si la matriz tiene tan solo un subíndice, entonces tenemos un grupo de elementos que están determinados mediante un único número que es la ubicación del elemento de la matriz, una matriz con estas características también se le llama arreglo, matriz unidimensional. Si la matriz tiene dos o más subíndices se le llama matriz multidimensional o simplemente matriz. La siguiente matriz X (6) consta de seis elementos. Ver figura 7.13.

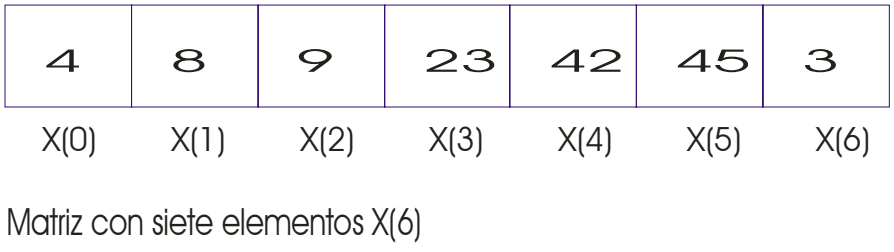


Figura 7.12. Representación de un arreglo en memoria

El ejemplo anterior, es una matriz unidimensional o arreglo, así, el número 23 está localizado en X(4), es decir, X(4)=23, X(5)=45, etc. Un ejemplo de matriz bidimensional es el siguiente: matriz (7, 8) que tienen 8 columnas y 9 filas, es decir:

matriz (7,8) = (Columna, Fila) . En Visual Basic Script a diferencia de otros lenguajes, son colecciones de datos heterogéneas. Ver listado 7.23.

```
<html><head><title>matriz uno</title></head>
    <body>
    <script language=”VBScript”>
    dim a(3)
        a(0)=”Hola”
        a(1)=3.1415927
        a(2)=89
        a(3)=”23”
        for i=0 to 3
contador=contador&”a(“&i&”)=”&a(i)&” “&”Es de tipo:”&typename(a(i))&chr(13)
    next
    msgbox contador
</script>
</body>
</html>
```

Listado 7.23

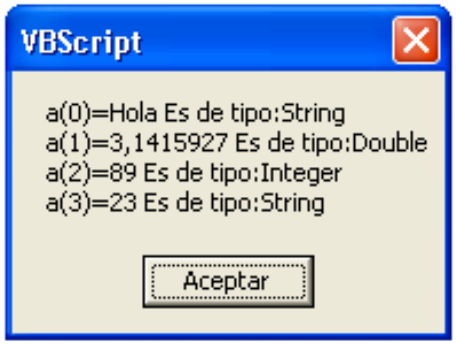


Figura 7.13. Resultado de la corrida del Listado 7.23

Importante. Las variables tipo matriz deben declararse obligatoriamente.

Vamos a realizar una suma de matrices, con lo que ya contamos para programar, con el propósito de profundizar el estudio, de este tema. A continuación mostramos el listado en donde utilizamos el concepto de matrices en VBScript con sentencias de control de flujo de datos.

```
<html><head><title>arreglo tres</title></head>
        <body bgcolor=”#ffffff”>
<Script language=”VBScript”>
   dim matriz1(2,2)
   dim matriz2(2,2)
   matriz1(0,0)=1
      matriz1(0,1)=0
      matriz1(0,2)=2
      matriz1(1,0)=3
      matriz1(1,1)=3
      matriz1(1,2)=5
      matriz1(2,0)=0
      matriz1(2,1)=5
      matriz1(2,2)=1
      matriz2(0,0)=0
      matriz2(0,1)=2
      matriz2(0,2)=0
      matriz2(1,0)=0
      matriz2(1,1)=1
      matriz2(1,2)=0
      matriz2(2,0)=5
      matriz2(2,1)=1
      matriz2(2,2)=3


   for i=0 to 2
   for j=0 to 2
            A=A&matriz1(i,j)&”      “
            B=B&matriz2(i,j)&”      “
              L=L&matriz1(i,j)+matriz2(i,j)&”     “
          next
```

```
    A=A&chr(13)
    B=B&chr(13)
        L=L&chr(13)
        next
msgbox « Matriz A « &chr(13)&chr(13)&A&chr(13)& »Matriz B
« &chr(13)&chr(13)&B&chr(13)& »La suma de matrices es: »&chr(13)&chr(13)&L
    </script>
</body>
</html>
```

Listado 7.24

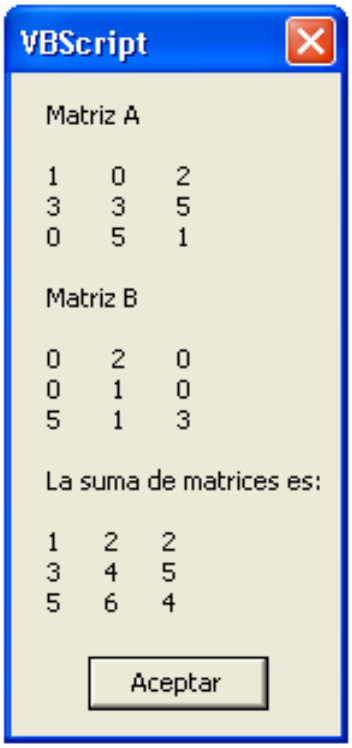


Figura 7.14. Corrida del listado 7.14

Los ejemplos anteriores, son ejemplos de matrices estáticas, se pueden realizar excelentes programas, pero el problema es que los elementos integrantes de la matriz no cambian a lo largo del programa. Hay veces que es necesario cambiar el tamaño de las matrices en los programas, para estos casos intervienen las matrices dinámicas.

Matrices dinámicas.

Las matrices dinámicas son mucho más flexibles que las estáticas, ya que nos proporciona la posibilidad de tener en la memoria de la computadora, listas de elementos de tamaño variable. Para declarar matrices de tamaño variable necesitamos declarar sin especificar el número de elementos que tendrá. Es decir:

```
dim  matriz(  )
```

Como no se conoce el tamaño tampoco se podrá ubicar un espacio para ella. En el momento en que se quiera modificar sus dimensiones, solo hay que hacerlo a través de la palabra reservada **redim** que indica al intérprete que existe una matriz que se va a redimensionar.

La matriz anterior queda del siguiente modo:

```
redim  matriz ( 6 )
```

ahora, si se introducen datos en los elementos de la matriz anterior y se quiere cambiar el tamaño de la misma hay volver a redimensionar la matriz, pero en el momento en que lo hagamos, los elementos que almacenaba desaparecerán. Ver listado 7.25.

```
redim matriz ( 7 )
```

```
<html><head><title>matriz cuatro ---Matrices din&aacute;micas---</title></head>
        <body bgcolor=”#ffffff”>
<script language=”VBscript”>
                                                dim arreglo()
   n=inputbox(“Tamaño del arreglo:”)
   redim arreglo(n-1)
   for i=0 to n-1

   arreglo(i)=inputbox(“a”&i+1&”=”)
   L=L&arreglo(i)&”  “
   next
   msgbox “Tu arreglo dinámico es:”&chr(13)&chr(13)&”( “&L&”)”
   </script>

   </body>
   </html>
```

Listado 7.25.

Figura 7.15. Corrida del listado 7.25

Esto se debe a que el intérprete al cambiar el tamaño de la matriz, éste busca un espacio libre en la memoria que pueda albergarla en su nuevo tamaño, liberando sin más, el espacio que ocupaba anteriormente, para que otros procesos lo puedan utilizar. Si nos interesara conservar los elementos de la matriz se debe incluir la palabra **preserve**. Tal y como se indica a continuación:

```
redim  preserve matriz ( 7  )
```

Ver listado 7.26.

```
<html><head><title>An&aacute;lisis de los redimensionamientos</title></head>
<body bgcolor=”#cccc66”>
<script language=”VBscript”>
dim matriz()
redim matriz(6)
for i=0 to 5
matriz(i)=inputbox(“a”&i+1&”=”)
resp=resp&matriz(i)&” “
next
msgbox(“La matriz fila es:”&”(“&resp&”)”)
redim preserve matriz(20)
for i=0 to 19
l=l&matriz(i)&” “
next
msgbox(“La matriz fila es:”&”(“&l&”)”)
redim matriz(10)
for i=0 to 9
mat=mat&matriz(i)
next
msgbox(“La matriz fila es: “&”(“&mat&”)”)
```

```
    </script>
</body>
</html>
```

Listado 7.26.

Las matrices especiales **ubound, y lbound** Se utilizan para ubicarnos en el inicio o en el fin de los subíndices de las dimensiones de una matriz. Por ejemplo si tenemos una matriz x (4,3) entonces, ubound(x,1) = 4 y ubound(x,2)=3. Ahora bién, lbound(x,1)=0 y lbound(x,2)=0. En el listado anterior se distingue mejor estas dos matrices especiales.

```
<html><head><title>Pruebas de Ubpound</title></head>
    <body bgcolor="#ffffff">
    <script language="VBScript">
        Option explicit
        dim a(5,6)
    msgbox "de a(5,6) ubound(a,2) devuelve un  "&ubound(a,2)
    msgbox "de a(5,6) ubound(a,1) devuelve un  "&ubound(a,1)
    dim b(6,1,7,9)
    msgbox " de b(6,1,7,9) lbound(b,1) devuelve un  "&lbound(b,1)
    msgbox " de b(6,1,7,9) lubound(b,2) devuelve un  "&lbound(b,2)
    msgbox " de b(6,1,7,9) lbound(b,3) devuelve un  "&lbound(b,3)
    msgbox " de b(6,1,7,9) lbound(b,4) devuelve un  "&lbound(b,4)
    msgbox " de b(6,1,7,9) ubound(b,1) devuelve un  "&ubound(b,1)
    msgbox " de b(6,1,7,9) ubound(b,2) devuelve un  "&ubound(b,2)
    msgbox " de b(6,1,7,9) ubound(b,3) devuelve un  "&ubound(b,3)
    msgbox " de b(6,1,7,9) ubound(b,4) devuelve un  "&ubound(b,4)
        </script>
        </body>
</html>
```

Listado 7.27.

En el listado siguiente se hace un uso extensivo del redimensionamiento de la matriz, para realizar la multiplicación matricial.

```
<html><head><title>Multiplicaci&oacute;n de matrices</title></head>
        <body bgcolor="#aa0045">
<script language="VBScript">
dim a(),b(),c()
    n=inputbox("Dame el tamaño de la matrices")
```

```
redim a(n-1,n-1),b(n-1,n-1),c(n-1,n-1)
for i=0 to n-1
for j=0 to n-1
producto=0
for k=0 to n-1
a(i,k)=inputbox("a"&i+1&k+1&"=")
    b(k,j)=inputbox("b"&k+1&j+1&"=")
producto=producto+a(i,k)*b(k,j)
c(i,j)=producto
next
        l=l&c(i,j)&"   "
        next
l=l&chr(13)
        next
        msgbox "El producto matricial es:"&chr(13)&chr(13)&l
        </script>
        </body>
</html>
```

Listado 7.28.

El siguiente listado nos muestra la ordenación de una lista de elementos de una matriz, el objetivo de esto es mostrar que las posibilidades de la forma en que podemos manejar los datos en la computadora. Ver listado 7.29

```
<html><head><title>Ordenaci&oacute;n de un conjunto de
n&uacute;meros</title></head>
    <body bgcolor="#ffffff">
    <script language="VBScript">
    dim x()
    n=inputbox("¿Cuántos números vas a ordenar?")
    redim x(n)
    for i=0 to n-1
    x(i)=inputbox("x["&i+1&"]=")
    next
    for elem=0 to n-1
for i=elem+1 to n
if x(i)<x(elem) then
temp=x(elem)
x(elem)=x(i)
```

```
                                    x(i)=temp
end if
    next
    next
    for i=0 to n
    l=l&x(i)&"  "
    next
    msgbox("La ordenación es la siguiente"&chr(13)&chr(13)&l)
    </script>
</body>
</html>
```

Listado 7.29

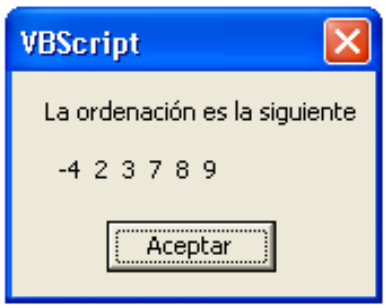


Figura 7.16

El siguiente programa es una agenda de empleados que suma todo lo visto hasta ahora sobre Visual Basic Script.

```
<html><head><title>Primera versi&oacute;n de la agenda
telef&oacute;nica</title></head>
<body bgcolor="#e6f59f">
    <script language="VBScript">
    Option explicit
        dim agenda()
'Observar aquí
        redim agenda(2,0)
sub capturar_clientes_agenda_onclick()
if nombre.value="  " or not isnumeric(telefono.value) or direccion.value="  " then exit sub
dim ultimo
    'Veo cuál va a ser el nuevo índice
    ultimo=ubound(agenda,2)+1
```

```
    'Redimensiono para aceptar uno nuevo
'Observar aquí
    redim preserve agenda(2,ultimo)
    agenda(0,ultimo)=nombre.value
    agenda(1,ultimo)=telefono.value
    agenda(2,ultimo)=direccion.value
    nombre.value=" "
    telefono.value=" "
    direccion.value=" "
end sub
sub listado_onclick()
dim listado_intranet
dim i
for i=1 to ubound(agenda,2)
listado_intranet=listado_intranet&agenda(0,i)&":"&agenda(1,i)&"  "&agenda(2,i)&chr(13)
next
msgbox "Datos enviados al servidor :"&chr(13)&chr(13)&listado_intranet
end sub
</script>
<h1><font color="blue" face="univers"><center>Intranets: Instalaci&oacute;n, cursos
y servicios</center></font></h1>
<h2><font color="blue" face="univers"><center>Agenda</center></font></h2>
<hr>
<p><font color="blue" face="antique olive">Nombre</font>
<input type="text" size="25" name="nombre">
        <font color="blue" face="antique olive">Tel&eacute;fono</font>
        <input type="text" size="20" name="telefono">
<br><br><font color="blue" face="antique olive">Direcci&oacute;n</font>
        <input type="text" size="61" name="direccion">
<br><br><input type="button" name="capturar_clientes_agenda" value="Añadir a la
agenda">
</p>
<p> </p>
<p align="center"><input type="button" name="listado" value="Generar listado"></p>
</body>
</html>
```

listado 7.30.

La ejecución del programa así se visualiza:

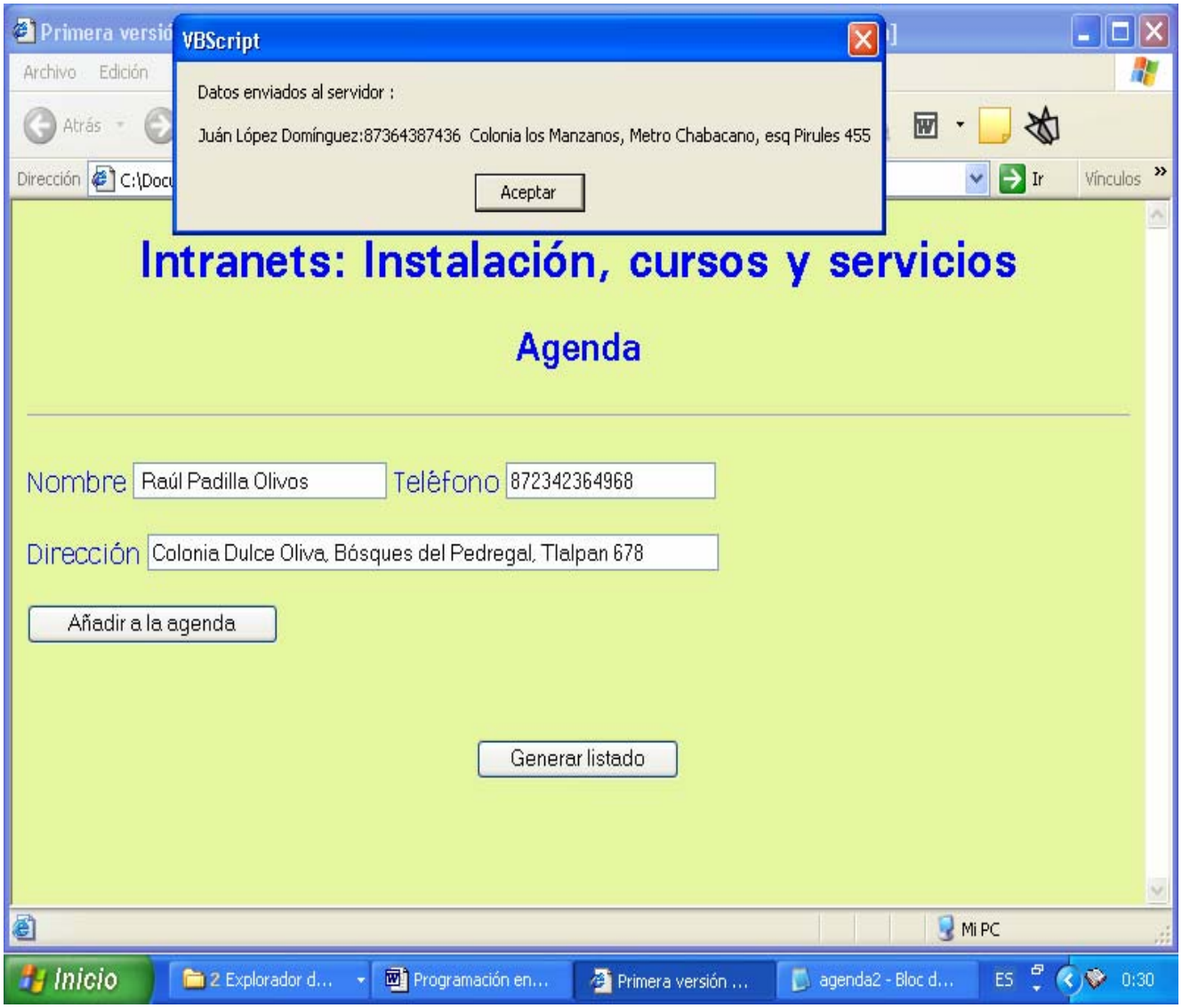


Figura 7.16. Ejecución del listado 7.30.

Algunas consideraciones acerca de la memoria y el rendimiento en matrices grandes.

¿Cuánta memoria ocupa una matriz?[1]

Primero que nada, cuando se define una matriz, ocupa 20 bytes mas 4 bytes por cada una de sus dimensiones. Esto es, si declaramos la siguiente matriz:

```
dim a(1,2,1)
```

Esta matriz ocupa 20 + 4 +4 +4 =32 bytes.

Pero cuando le empezamos a asignar valores a uno de sus elementos, entonces, cada uno de sus elementos ocupará el mismo espacio. Veamos este fragmento de programa:

```
dim  x ( 9, 9, 9)
x (0, 0, 0 ) = "el trabajo es el amor hecho visible"
```

Las dos líneas de código se ven inofensivas pero vamos a analizarlas a detalle.

Análisis:

a) Cuando se declara la matriz x (9, 9, 9) ocupará 20 + 4 + 4 +4 = 32 bytes.
b) Al primer elemento se le asigna la cadena
c) Cualquier cadena de texto ocupa 10 bytes + la longitud d e la cadena, o sea,

   10 + 35 = 45 bytes.

   Como la matriz tiene 1000 elementos, entonces todos sus elementos ocupan 45 000 bytes.
d) En total las dos líneas ocupan 45 000 + 32 = 45 032 bytes, casi 45 KB de memoria.

Analicemos las siguientes tres líneas de código aparentemente inofensivas:

dim x (70, 70, 70, 70, 70, 70, 70, 70) x(0, 0, 0, 0, 0, 0, 0, 0) = "bienvenidos al mundo de la programación de computadoras" msgbox x(0, 0, 0, 0, 0, 0, 0, 0)

Análisis:

a) Cuando se declara la matriz x (70, 70, 70, 70, 70, 70, 70, 70, 70,70) ocupará 20 + 4 + 4 + 4 + 4 + 4 + 4 + 4 + 4 = 52 bytes.

b) Al primer elemento se le asigna la cadena

c) Cualquier cadena de texto ocupa 10 bytes + la longitud de la cadena, o sea, "bienvenidos al mundo de la programación de computadoras" es:

   10 + 55 = 65 bytes.

d) El espacio que ocupa la matriz es de:

183, 608, 911, 850, 000, 000,000 bytes.
170, 999, 124,506.4884gigabytes.
166, 991,332.5258676 terabytes.

1.- Diseño de páginas Web interactivas con Java Script. Juán Carlos Orós. Editorial Alfaomega. RAMA. 2da Edición

VIII. Programación avanzada de bases de datos para Internet.

8.1. Programación Avanzada con Access.

El propósito de este capítulo es crear poderosas aplicaciones cliente servidor y orientadas a objetos creando interfaces gráficas que sean amigables para el usuario, o bien, una interfaz basada en Web.

Pasos para crear aplicaciones en Access:

a) Primero que nada cargue la aplicación de Office, Access.
b) Desplácese al menú archivo → Nuevo
c) Del panel de tareas seleccione la opción base de datos en blanco, cuando aparezca el panel de control Archivo nueva base de datos, ponga un nombre y oprima el botón Crear.
d) Seleccione el objeto módulos y haga clic en Nuevo.
e) Crear el código en el editor de Visual Basic.

Depuración

Hay varias formas para depurar el código, una de ellas es oprimiendo **la tecla F8**, ver figura 8.1.

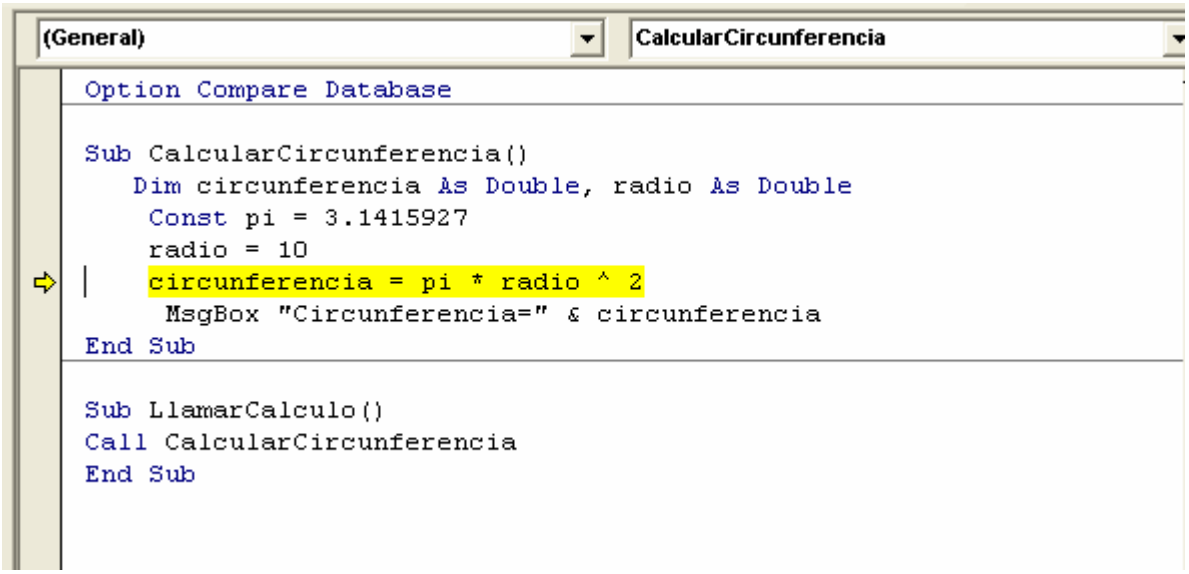


Figura 8.1

Uso de ventanas locales

Las ventanas locales muestran el comportamiento de las variables locales a lo largo de la ejecución del programa. Para abrir ventanas locales váyase al menú →Ver→Ventana locales, o bien, Alt + v, l.

Uso de la ventana de inspecciones.

Es similar a la ventana Locales, pero aquí solo aparecen las variables que se quieren examinar, aquí se pueden examinar variables locales y globales. Para acceder a esta ventana váyase al menú Ver → Ventana Inspección o bien Alt + v, n.

Cómo agregar un elemento a la ventana inspecciones.

Para agregar elementos a la ventana de inspecciones, váyase al menú →Depuración → Agregar Inspección, o bien, Alt + d, a.

Para ejecutar el programa simplemente oprima la **tecla F5.**

Edición de inspecciones.

Para modificar una inspección hay que realizar lo siguiente:

- ✓ Haga un clic en el valor de inspección que se va a modificar.
- ✓ Haga clic en el menú depuración.
- ✓ Haga clic en Modificar inspección.

Ejecución de código en la ventana inmediato.

La ventana inmediato permite probar el código fuera del programa, se puede probar código muy sencillo y complejo en esta ventana, para abrir la ventana inmediato váyase al menú → Ver →Ventana inmediato, o bien Alt + v, i. Ahora escribe en esta ventana ?pi y oprime <enter>.

El programa siguiente es ilustrativo, por que expone satisfactoriamente la importancia del empleo de las técnicas de depuración antes mencionadas.

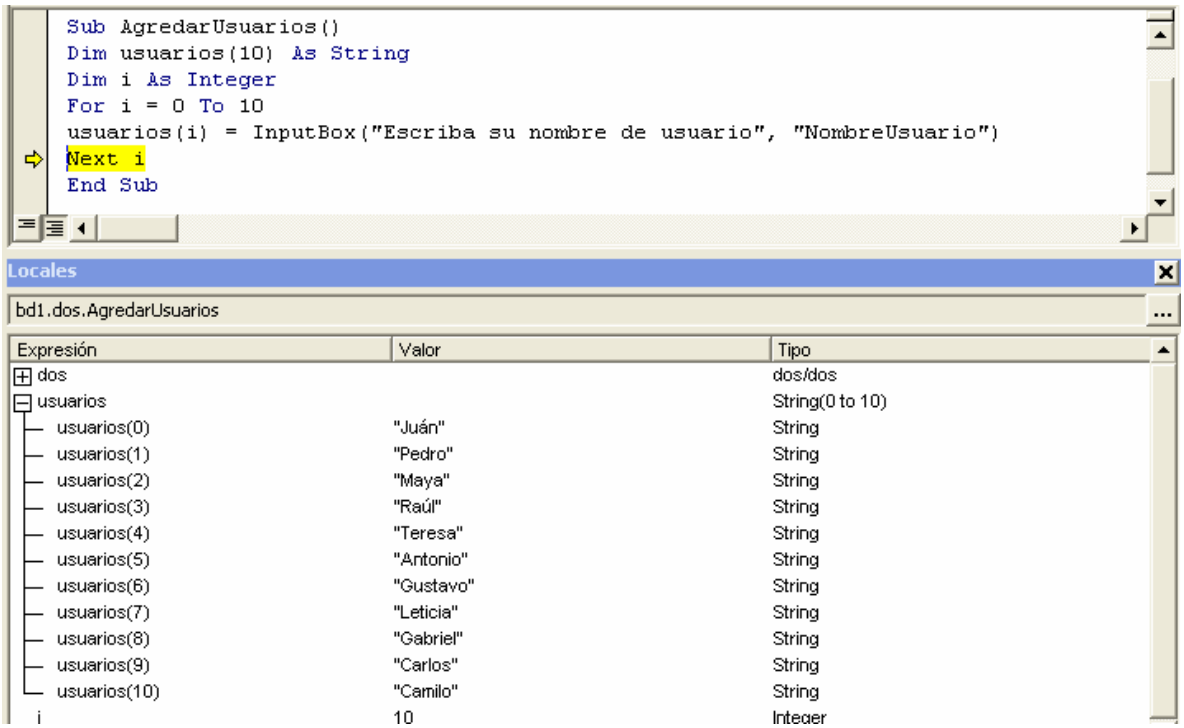


Figura 8.2

Ubound y Lbound

Ubound devuelve el elemento indexado más alto de la matriz y la función Lbound devuelve el elemento indexado más bajo. El siguiente listado ilustra correctamente la utilidad de estas dos funciones.

```
Sub altausuarios()
        Dim Usuarios(10) As String
          Dim i As Integer
       For i = UBound(Usuarios) To LBound(Usuarios) Step -1
    Usuarios(i) = InputBox(“Escriba su nombre de usuario”, “NombreUsuario”)
  Next i
End Sub
```

Listado 8.1

Iteración con colecciones de datos

La construcción for… each se diseñó exclusivamente para colecciones de datos y matrices. El bucle for … each necesita de un tipo de datos variant.

La sintaxis es la siguiente:

```
Dim Elemento as Variant

        For each Elemento in NombreMatriz
‘Procesar elemento
```

next elemento

El código siguiente demuestra la forma de utilizar la construcción anterior:

```
Option Compare Database
        Sub DesplegarListado()
Dim usuarios(2) As String
   usuarios(0) = “Lola”
   usuarios(1) = “Lucas”
   usuarios(2) = “Guadalupe”

   Dim usuario As Variant

   For Each usuario In usuarios

   MsgBox usuario

   Next usuario

   End Sub
```

Listado 8.2

8.2. Administración de bases de datos

Referencias a bibliotecas ActiveX.

Las referencias a las bibliotecas ActiveX, creación de tablas a través de código, se realiza lo siguiente:

- ✓ Abrir la tabla de datos en cuestión.
- ✓ Clic en el botón módulos.
- ✓ Doble clic en el módulo principal para que se abra el editor, o bien, alt + F11.
- ✓ Váyase a Herramientas→Referencias
- ✓ Del cuadro de diálogo

Uso de una conexión

Para abrir una nueva conexión, primero se debe declarar un objeto Connection. La sintaxis para declarar Connection es:

Dim NombreConexion as new ADODB.Connection

El prefijo ADODB con connection es necesario por que existen otros tipos de objetos Connection.

Cómo abrir una conexión

Para abrir una conexión, se debe tener una cadena de conexión. La cadena de conexión contiene la información del proveedor y del origen de datos.

8.3. Solución de problemas paso a paso

Una buena forma de resolver un problema complicado es dividirlo en un conjunto de problemas sencillos, en la programación actual se maneja mucho este sencillo concepto, es decir, un programa es un conjunto de código que resuelve uno o muchos problemas relacionados, o sea programa es un conjunto de bloques con una pequeña cantidad líneas de código.

```
Sub PruebaByVal(ByVal A As Integer)
        A = 5
        MsgBox A
   End Sub

Sub prueba()

   Dim A As Integer
   A = 10
   Call PruebaByVal(A)
   MsgBox A
End Sub
```

Listado 8.3

8.4. Reglas generales de programación:

Regla 1: Haga del código recurrente un procedimiento

La mejor forma de hacer más eficiente el código es escribiendo menos código. Si vemos que varias líneas se repiten constantemente en su programa, póngalas en un procedimiento.

Regla 2: Mantenga procedimientos cortos

Un procedimiento corto es aquél que tiene menos de cinco líneas (10 con código para el manejo de errores). En la mayoría de los casos, si hay más de 5 o 10 líneas, el procedimiento está haciendo demasiado. Mantener procedimientos cortos incrementa la oportunidad de reutilizar el código y un bajo costo.

Regla 3: Limite estrictamente el número de argumentos

Los mejores procedimientos son los que requieren unos cuantos argumentos, debido a que solo realizan una sola tarea.

Regla 4: Utilice calificadores de argumentos para inhibir el mal uso de variables

Los argumentos deben tener claramente definidos sus propósitos de tal forma que se le indique claramente al compilador lo que se pretende.

Regla 5: Usar programación bajo contrato

La programación bajo contrato es una forma de procurar que las funciones y variables que programemos sean confiables y que se cumplen ciertas condiciones preestablecidas.

Primero que nada vamos creando una tabla:

Primero que nada vamos a crear la tabla alumnos, el tamaño y las propiedades de los campos se dejan a las necesidades específicas del sistema que se vaya a desarrollar. Ver figura 8.2.

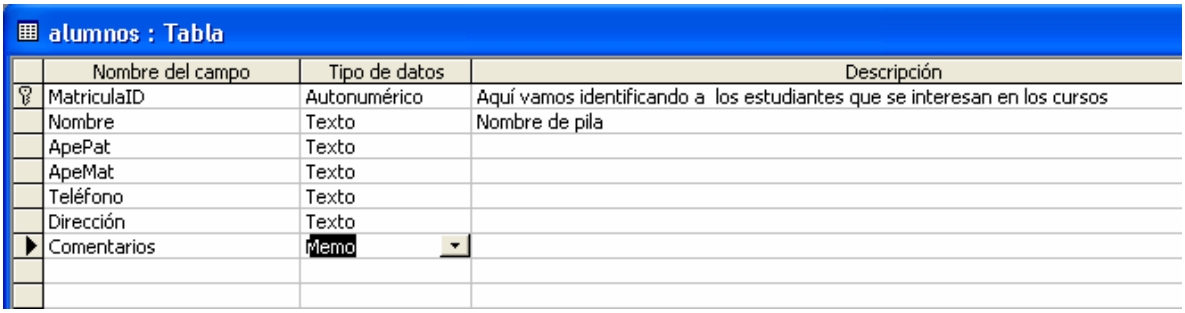


Figura 8.2.

Creación de código para formulario

Ya terminada la tabla de datos procederemos a utilizarla en un objeto de formulario, para tal efecto seleccionamos el **objeto formularios** y dentro de formularios seleccionamos **nuevo**, y en el panel **Nuevo formulario** seleccionamos **Vista Diseño** y seleccionamos la tabla que acabamos de crear y oprimimos el botón **aceptar.**

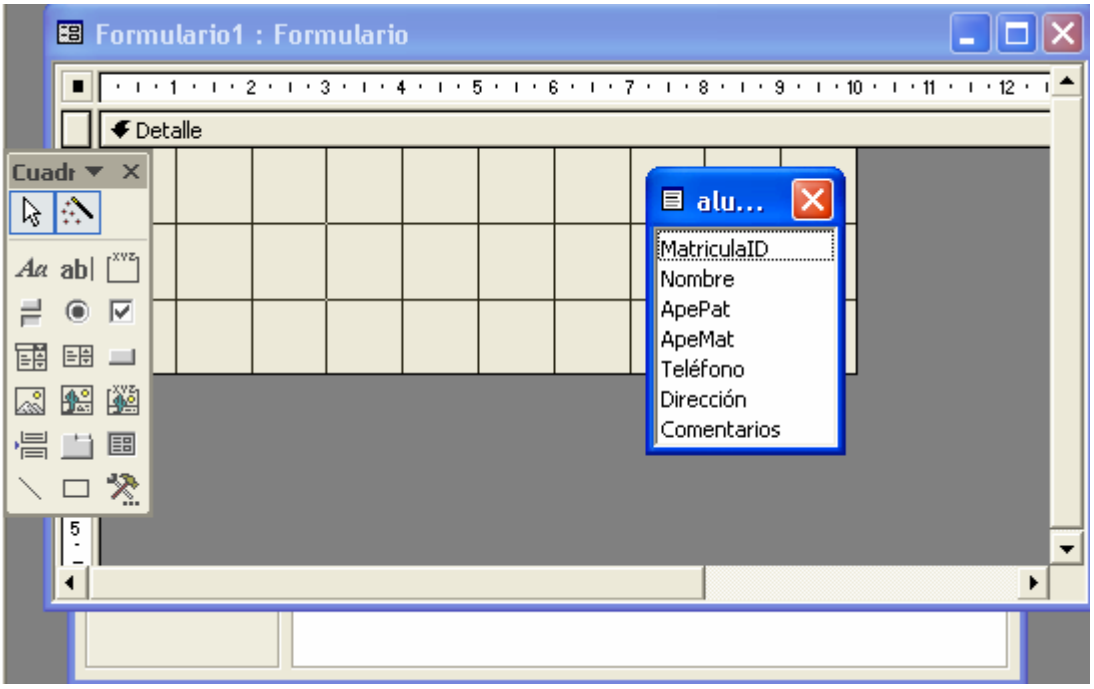


Figura 8.3.

En la figura anterior sobresalen el cuadro de herramientas y una ventana que contiene los campos de la tabla que acabamos de crear, a continuación vamos a seleccionar el control **botón de comando**, de un clic y posteriormente arrastre en diagonal sobre el objeto formulario trazando un botón de tamaño considerable, cuando termine aparecerá un **Asistente para botones de comando**, en el cuadro de **categorías** seleccionamos la opción **exploración de registros** y en el cuadro de **acciones** seleccionamos la opción **Ir al registro siguiente,** oprimimos el botón **siguiente,** volvemos a oprimir **siguiente** y en el cuadro de diálogo que aparece el nombre que le debemos dar al botón, en nuestro caso nosotros le vamos a poner **Siguiente** a nuestro botón y oprimimos el botón **Finalizar**, ahora vamos a acomodar los campos del lado izquierdo del formulario y cuando hayamos terminado, vamos a poner un botón que nos mande al **registro Anterior, Primero y Último**, para hacerlo recordemos que tenemos que tenemos que ir arrastrando los controles botón de comando del cuadro de herramientas, vamos a tratar de acomodarlos como se muestra a continuación:

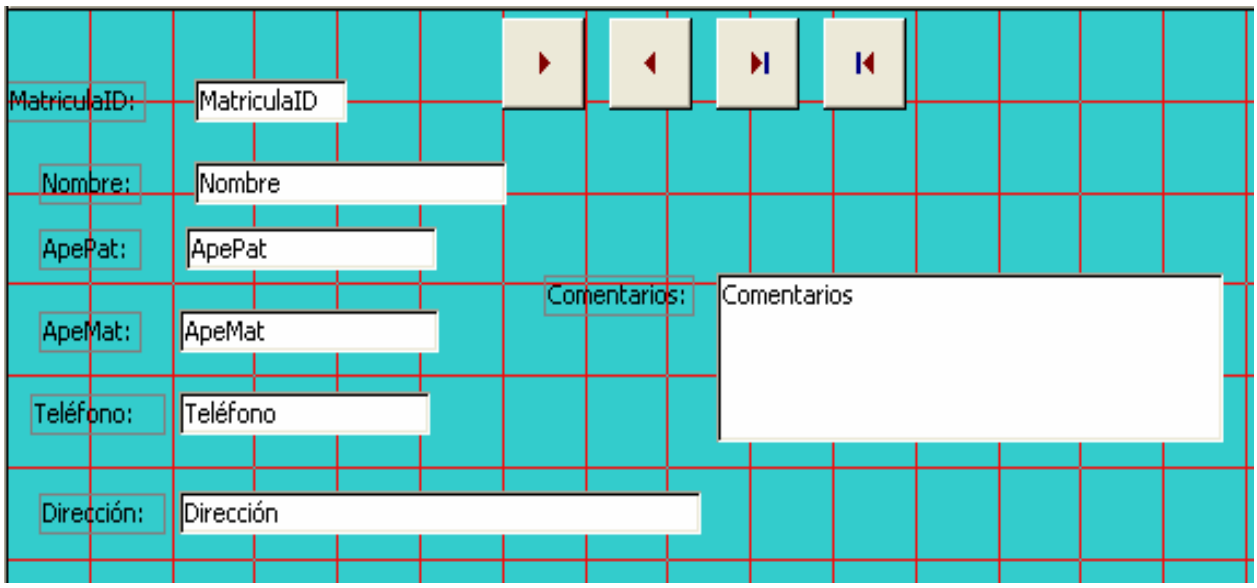


Figura 8.4.

Para visualizar nuestro formulario, basta con oprimir el botón ura que se muestre como se indica a continuación:

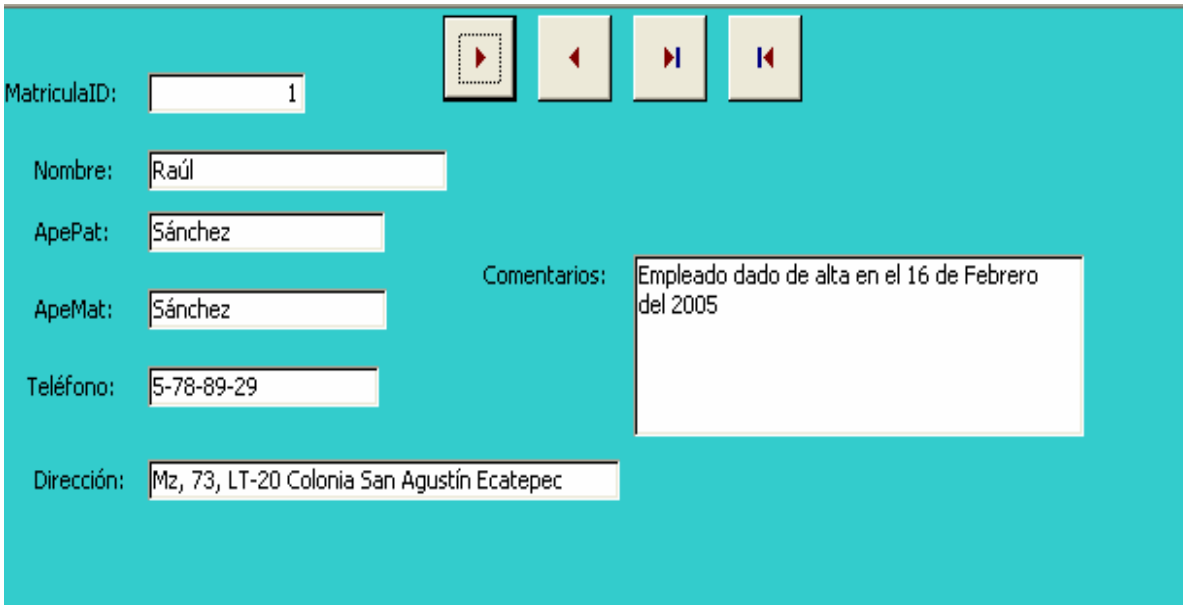


Figura 8.5.

Podemos ir al modo de diseño oprimimos el botón .

Agregando código Visual Basic Aplication a los controles botón de comando.

Para darles inteligencia a los botones hay que **seleccionar el botón** al que se va a añadir código y posteriormente darle clic con la **tecla derecha del ratón** y seleccionamos la opción Generar evento… Para nuestro caso Vamos a seleccionar el botón último, y le aplicamos los pasos anteriores, y ya dentro del subprocedimiento vamos a escribir la siguiente instrucción:

Msgbox " Último registro"

Ver figura 8.6.

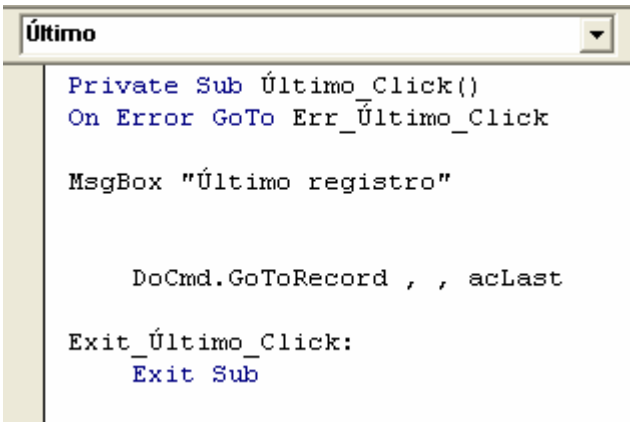


Figura 8.6.

Cuando terminemos y guardemos todos los cambios, vamos a probar nuestro formulario oprimiendo el botón Último. Ver figura 8.7.

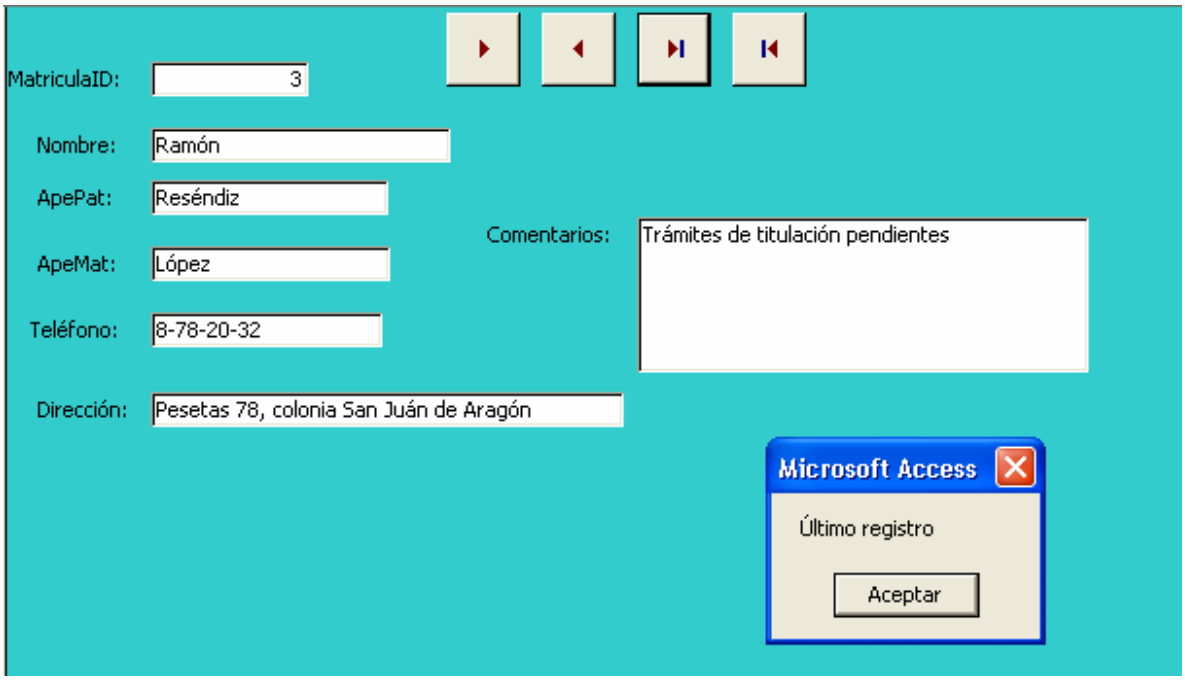


Figura 8.7.

A continuación vamos a realizar lo propio con el primer registro.

Creación de una página de acceso a datos

Las páginas de acceso a datos pueden utilizarse:

- ✓ Desde Access, mezcladas con formularios e informes estándar.
- ✓ En un sitio Web de Internet o de una intranet.
- ✓ Con una combinación de ambos.

Las páginas de acceso a datos son archivos HTML separados, con un acceso directo en la ventana de bases de datos de Access.

Vamos a crear un archivo HTML del ejercicio anterior. Para lo cuál realizaremos lo siguiente:

a) Primero abrimos las aplicaciones que creamos en el ejercicio anterior, una vez abiertas, seleccionamos el objeto Páginas, y posteriormente seleccionamos la opción Crear una página de acceso a datos en vista diseño. Ver figura 8.7.

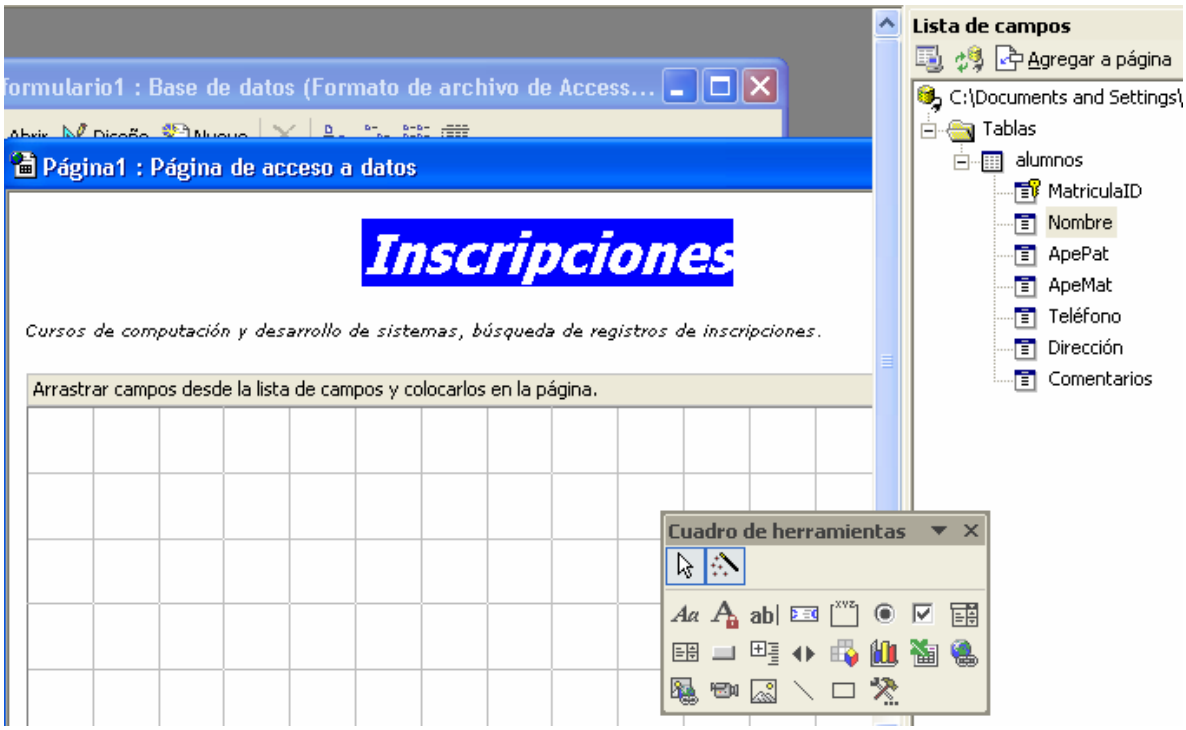


Figura 8.8.

b) En el panel de la lista de campos, le damos un doble clic a los campos de interés y los vamos acomodando en el formulario, para ir viendo como va quedando nuestro diseño de nuestra página Web, hay que ir intercalando las vistas y este es un consejo.

c) Una vez que el diseño haya completado nuestras expectativas, lo guardamos, si lo que hicimos es correcto se mostrará una página Web como la siguiente:

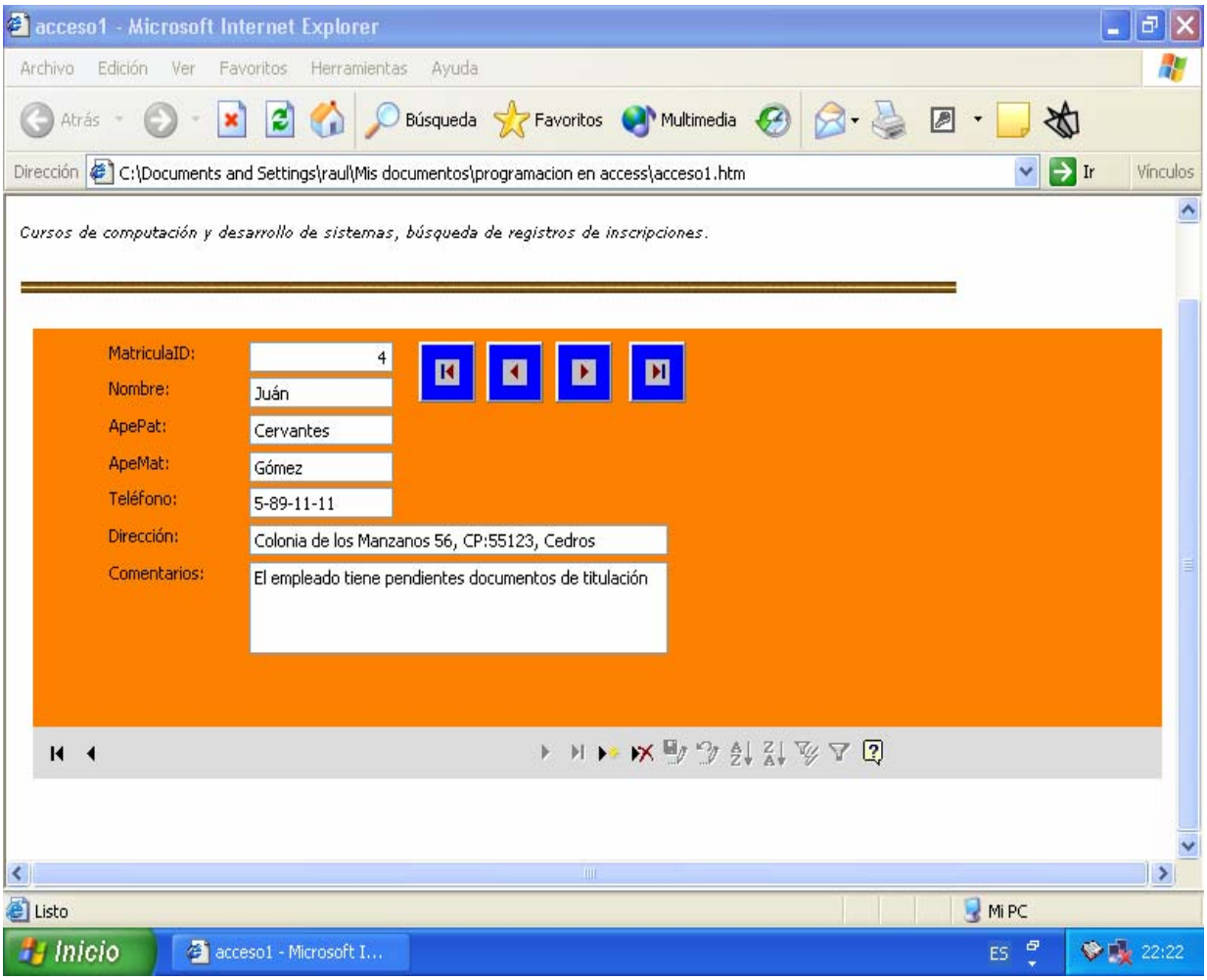


Figura 8.9.

d) Desde esta sencilla aplicación se pueden dar de alta los alumnos y también los podemos buscar, esta es una forma de inscribir a los alumnos desde una oficina.

Subir a un servidor de Internet nuestra página de acceso a datos.

Ya que tenemos terminada nuestra primera y sencilla obra de arte, vamos a subirla a un servidor de Internet, para tal efecto utilizaremos el Microsoft Information Server (Microsoft IIS Versión 5.1) que es la última versión de servidores de Internet de Microsoft. Hay que dejar claro que las páginas de acceso a datos de Access solamente funciona en servidores de Microsoft, es decir, estas páginas por el momento no será posible verlas en un servidor apache por ejemplo.

8.5. Pasos para subir una página de acceso a datos

1. Primero que nada hay que tener listo nuestro servidor de Internet de Microsoft. Ver figura 8.10.

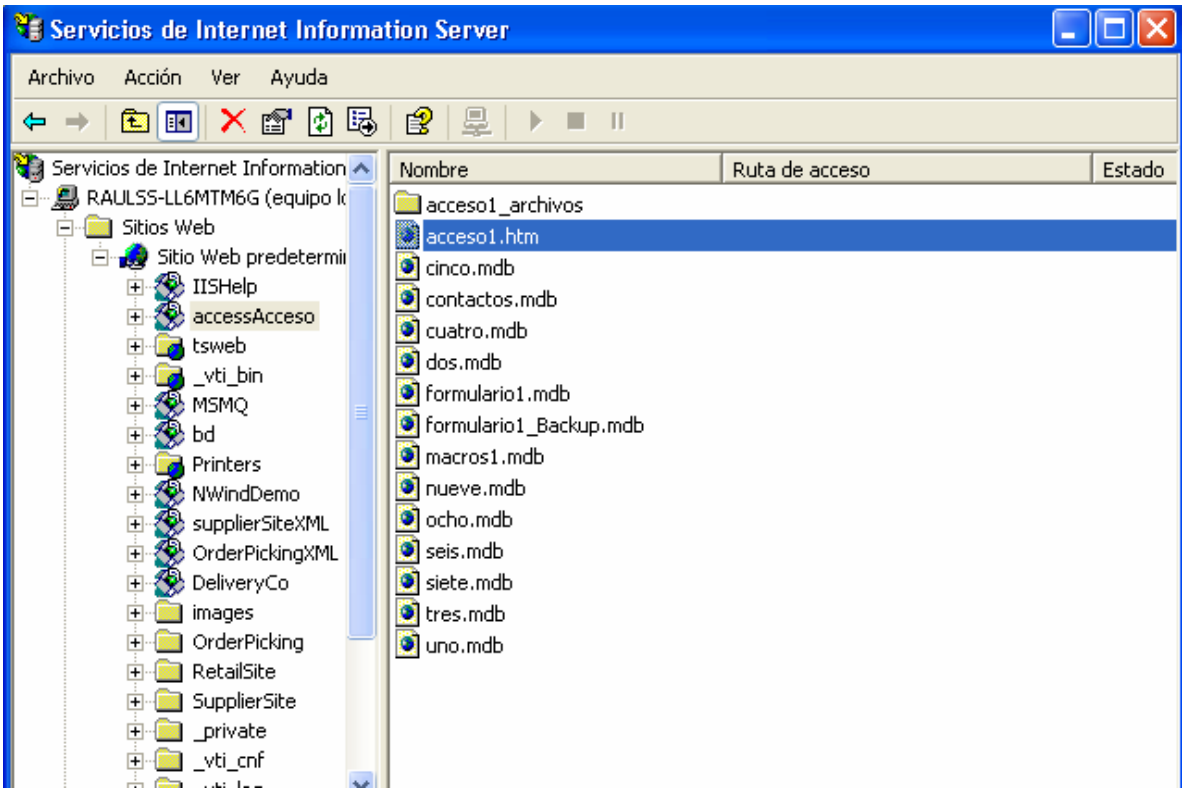


Figura 8.10.

2. Tenga listos los archivos que va a subir en una carpeta que sea descriptiva.
3. Posiciónese en el directorio **Sitio Web predeterminado**.
4. Váyase al menú Acción →Nuevo→Directorio Virtual. Ver figura 8.11:

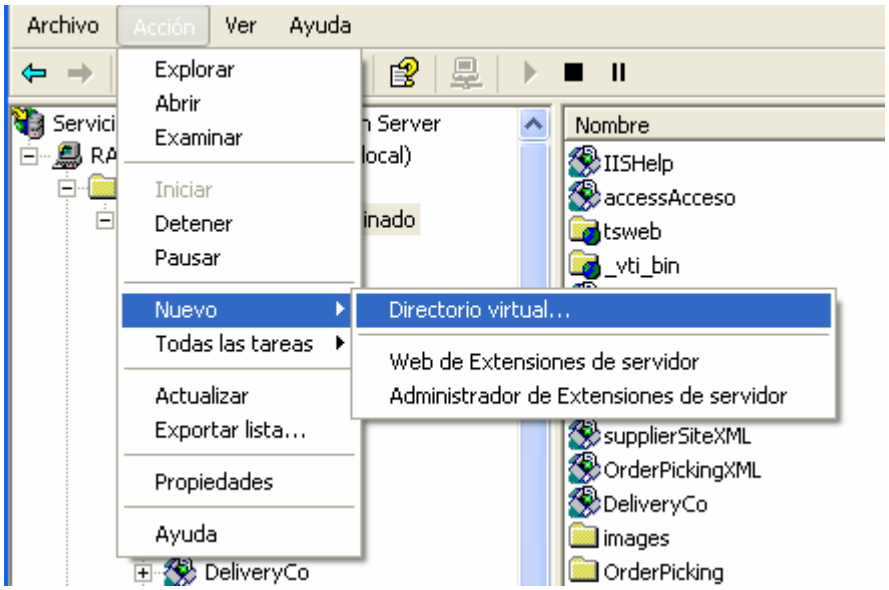


Figura 8.11.

5. Cuando haya aparecido el asistente <<Asistente para crear un directorio virtual>> oprimir el botón siguiente.
6. Se escribe un alias, que es el nombre de la carpeta en el servidor, para nuestro caso el alias es accessAcceso y a continuación se oprime el botón siguiente.
7. En la siguiente ventana hay que localizar el directorio donde tenemos nuestros archivos, oprime el botón examinar 

8. Cuando hayamos encontrado nuestra carpeta en mi caso mi carpeta se llama programación en access, oprimir el botón aceptar y después se oprime el botón siguiente.
9. Volver a oprimir siguiente.
10. Una vez finalizado el asistente para subir nuestros archivos al servidor, lo que resta es buscar nuestro sitio URL en Internet. Ver **figura 8.12**

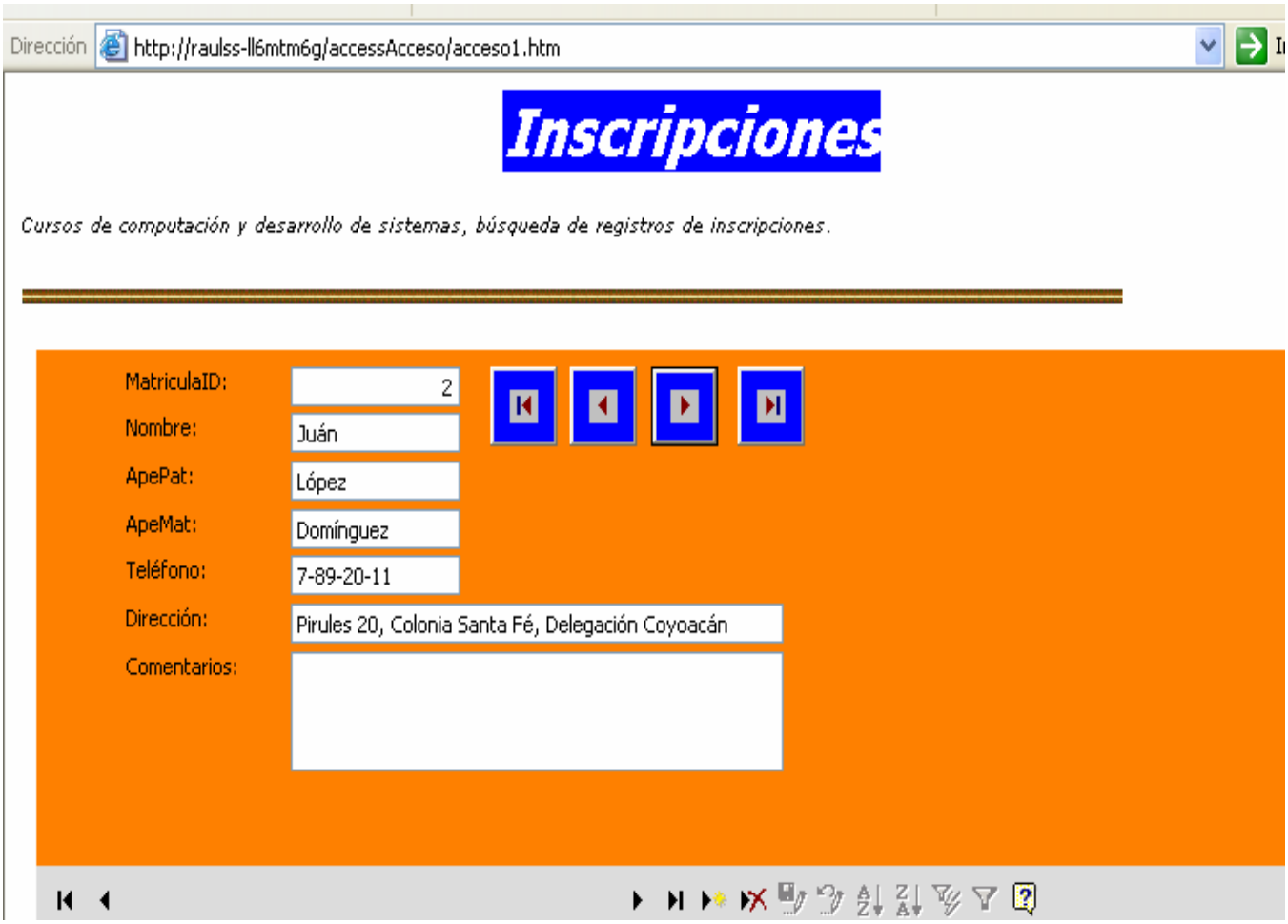


Figura 8.12.

Como se observa, hasta ahora la programación en access nos ha resultado sencilla, pero lo visto hasta este momento es la punta del iceberg, pues la complejidad del manejo de la información en el mundo actual obliga a realizar el desarrollo de sistemas Web en equipos de trabajo multidisciplinarios. Inclusive hay ocasiones que los asistentes de diseño son insuficientes y se tiene que programar si más recursos que la lógica y la habilidad de los programadores.

Programar en Visual Basic es muy sencillo y flexible, se puede programar sin necesidad de llevar una disciplina lógica, esto ocasiona que con el tiempo se realicen programas imposibles de mantener. En Basic no es necesario declarar las variables y sin embargo el intérprete no nos va a mandar ningún error. Para forzar a declarar las variables basta con colocar la instrucción Option Explicit.

Para asegurarse de que esta instrucción esté presente váyase al menú herramientas →Opciones ver figura 8.13.

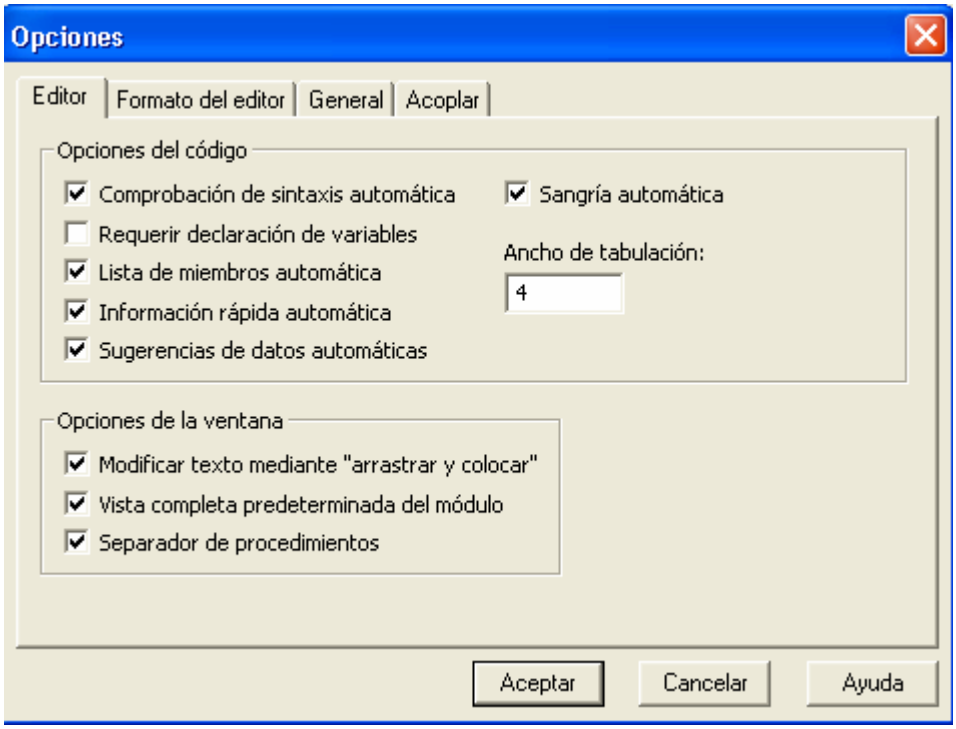


Figura 8.13.

Cuando aparezca la ventana, marque la casilla de verificación de Requerir declaración de variables.

Programación de Scripts Avanzados en Visual Basic

Vamos a realizar una aplicación para una Intranet escolar que consiste en lo siguiente:

- ✓ Los estudiantes van a poder hacer trámites escolares si cuenta con una conexión de Internet en cualquier lugar y a cualquier hora del día.
- ✓ Cada estudiante contará con una clave de acceso a sus registros (password), podrá hacer trámites escolares.
- ✓ Los bancos de datos estarán guardados en un servidor de bases de datos y se realizarán en ACCESS.
- ✓ Estas bases de datos se estarán disponibles en servidores de Internet de Microsoft.

Creando la base de datos cb

Vamos a empezar creando el origen de los datos, haciendo la base de datos cb que contengan las siguientes tablas: La tabla alumno, clientes y tblfolio. Ver figura 8.14:

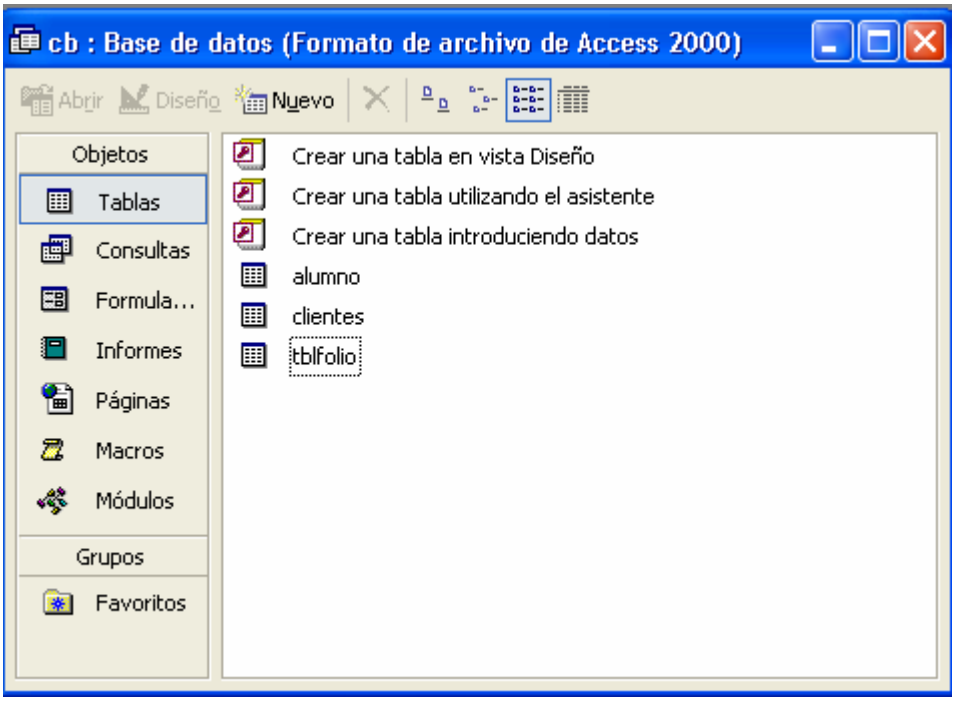


Figura 8.14 Bases de datos cb.

Ahora vamos a definir las propiedades de estos objetos alumno, clientes y tblfolio

Figura 8.14 a. Propiedades del objeto alumno.

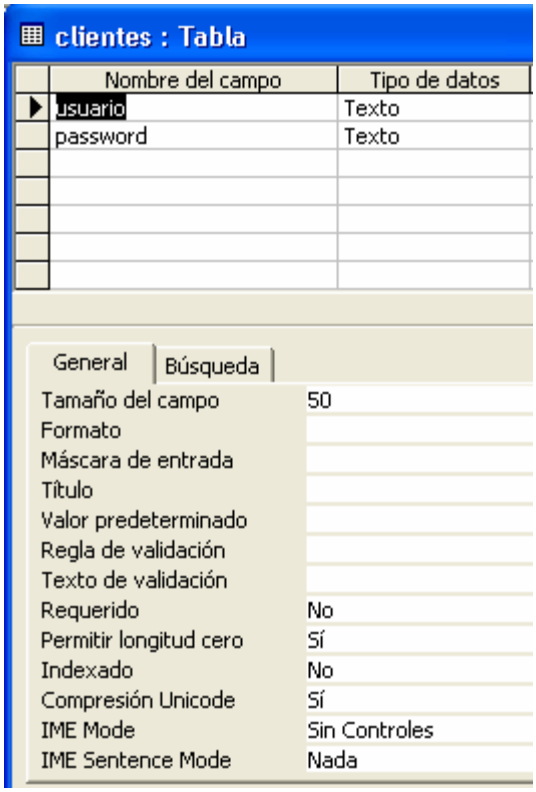


Figura 8.14 b. Propiedades del objeto clientes.

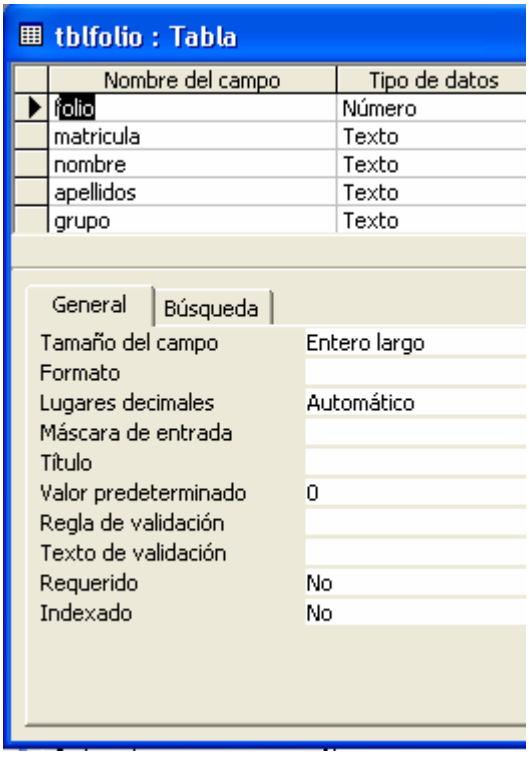


Figura 8.14 c. Propiedades del objeto alumno.

Scripts Avanzados en Visual Basic.

Terminando de realizar nuestras tablas, procederemos a crear, nuestros Scripts en visual Basic, en nuestro caso empezaremos creando el script de acceso al sistema, ver figura 8.15:

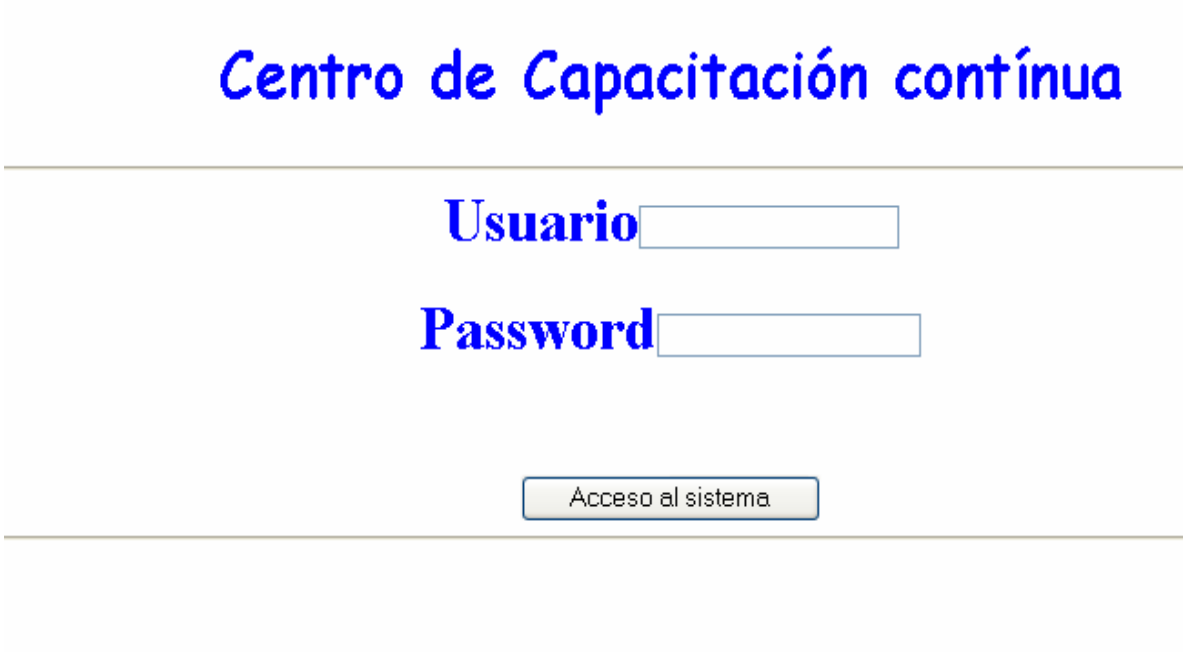


Figura 8.15

He aquí el código:

```
<HTML>
<HEAD>
    <TITLE>Acceso al sistema</TITLE>
</HEAD>

<SCRIPT language=VBScript>
public reko,conn,consulta
sub entrar()
    set conn=Createobject("ADODB.Connection")
    set reko=CreateObject("ADODB.Recordset")
    conn.Open "Provider=MSDASQL;DSN=union2"
    consulta="Select * From clientes"
    reko.open consulta,conn,1,2
end sub
sub entra()
    dim x
    reko.MoveFirst()
    do while (reko.EOF=false)
```

```
if (reko.fields(“usuario”).Value=clave.Text1.value)and
(reko.fields(“password”).Value=clave.text2.value) then
    window.open(“principal.html”)
    x=1
    end if
    reko.MoveNext()
    loop
    if x <>1 then
        msgbox”Usuario o password invalido”,,”Error”
    end if
end sub
</SCRIPT>

<BODY onload=entrar()>

<Form name=clave>

<h1><font color= »blue » face= »Comic Sans MS »><center>Centro de
Capacitaci&oacute;n cont&iacute;nua</center></font></h1>

<hr>

<center><font color=”blue” size=6><b>Usuario<INPUT type=”text”
ID=Text1></center></font><br>

<center><font color=”blue” size=6>Password<INPUT type=”password”
ID=text2></center></font><br><br><br></b>

<center><font color=”blue” size=6><INPUT onclick=entra() type=”button”
value=”Acceso al sistema” ID=Button1></font></center>

<hr>
</form>
</BODY>
</HTML>
```

Listado 8.4

8.6. El conector de bases de datos de Internet.

Los conocimientos previos del capítulo 6 y 7 nos dan la posibilidad de crear aplicaciones interactivas para Internet /Intranet a un nivel más aceptable, las aplicaciones vistas hasta este momento son aplicaciones del lado del cliente, en este capítulo vamos a desarrollar aplicaciones cliente/servidor que son las redes, y concretamente la red Internet o una Intranet.

8.7. Instalación del origen de datos ODBC.

Para poder instalar nuevos programas y controladores en su sistema, deberá tener categoría de administrador en el mismo. El único controlador ODBC que se suministra con IIS es el correspondiente a bases de datos SQL server que es el que nosotros vamos a utilizar.

Nosotros ya tenemos nuestra base de datos hecha, lo que sigue es instalar nuestro origen de datos, para esto hay que realizar los siguientes pasos:

1.- En el panel de control seleccionamos la opción rendimiento y mantenimiento, y después seleccionamos la opción herramientas administrativas ver figura 8.16:

Figura 8.16

2.- En el panel de herramientas administrativas seleccionamos la opción orígenes de datos ODBC.

3.- Ya activado este cuadro, seleccionamos la opción DNS del sistema:

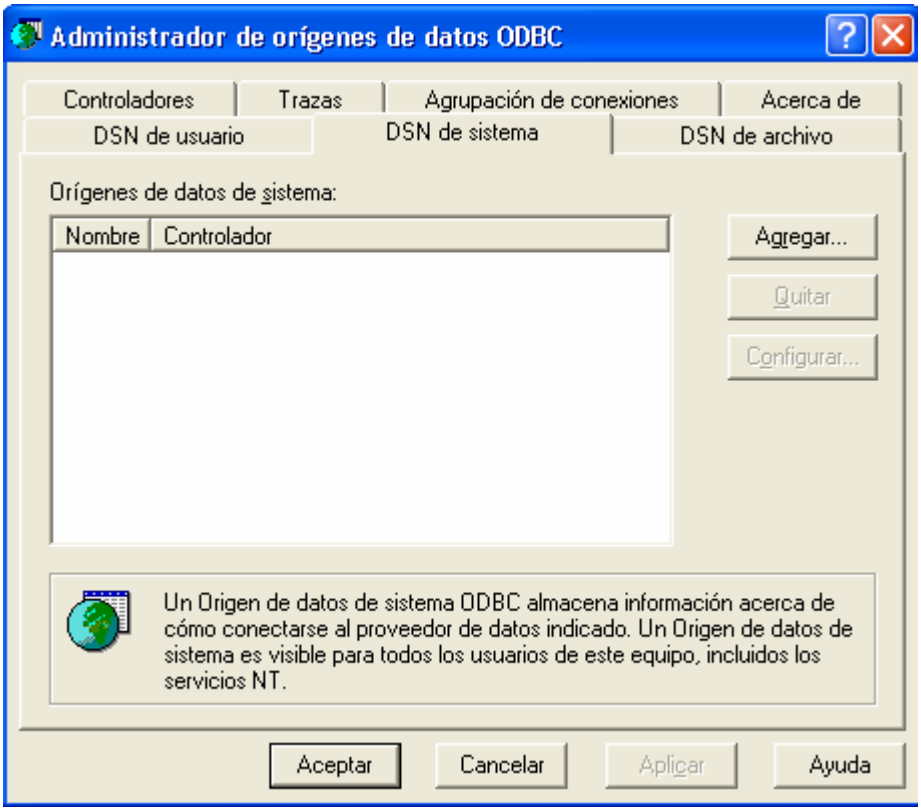


Figura 8.17

4.- Oprimimos el botón agregar y dentro de agregar seleccionamos la opción Microsoft Access Driver.

5.- Cuando aparezca el cuadro de configuración de ODBC, seleccionamos el nombre del origen de datos, oprimiendo el botón seleccionar buscamos nuestra base de datos y ponemos el nombre de nuestro origen de datos. Ver figura 8.18.

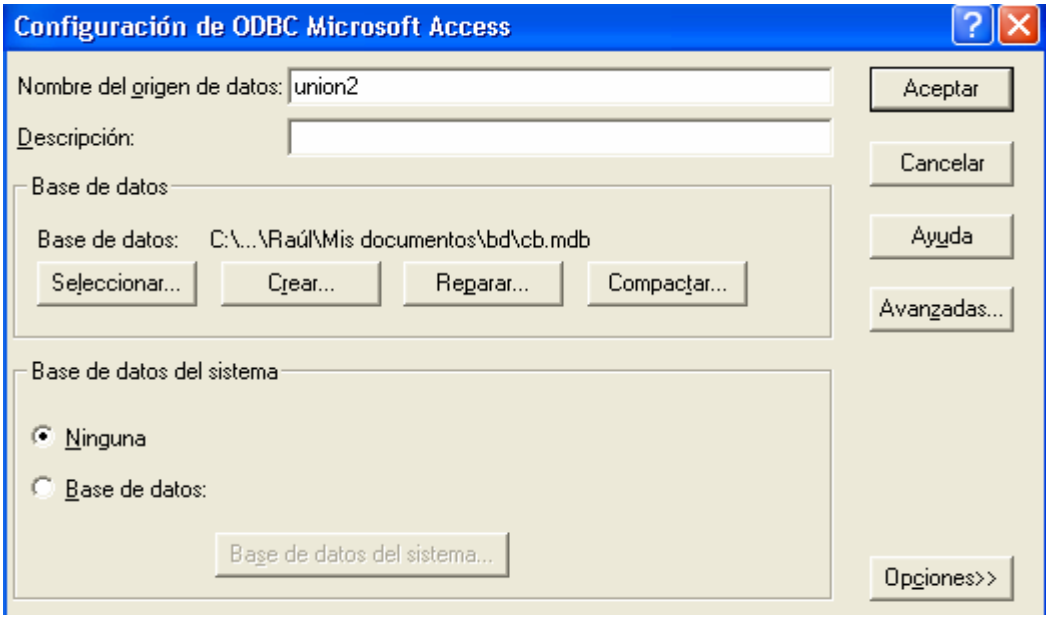


Figura 8.18

6.- Oprimimos el botón Aceptar.

Listados

Estos son los programas que vamos a cargar en nuestro servidor IIS de Microsoft, hay que tener presente que los programas escritos en Visual Basic Script, solamente funcionan en servidores de Microsoft, si queremos que estos programas corran en un servidor y en un navegador diferente, deberemos de guardarlos con la extensión asp y en algunos de servidores web, hay que realizar la configuración correspondiente. Ver listados 8.5, 8.5 y 8.7:

```
<HTML>
        <HEAD>
<TITLE>BUSCANDO REGISTROS</TITLE>


<script language=”VBScript”>
public reko,conn,consulta,rekofolio,consulta2
        sub entrar()
set conn=CreateObject(“ADODB.connection”)
    set reko=CreateObject(“ADODB.recordset”)
    set rekofolio=CreateObject(“ADODB.recordset”)
    conn.open “provider=msdasql;dsn=union2”
    consulta=”select * from alumno”
    reko.open consulta,conn,1,2
    reko.movefirst()
envio.text1.value=reko.fields(“folio”).value
```

```
envio.text2.value=reko.fields(“matricula”).value
envio.text3.value=reko.fields(“nombre”).value
envio.text4.value=reko.fields(“apellidos”).value
envio.Grupo.value=reko.fields(“grupo”).value

    end sub

    sub registro_adelante()

    reko.movenext()
    if reko.eof=true then
    msgbox”ultimo registro”
    reko.movelast()
    end if
    envio.text1.value=reko.fields(“folio”).value
    envio.text2.value=reko.fields(“matricula”).value
    envio.text3.value=reko.fields(“nombre”).value
    envio.text4.value=reko.fields(“apellidos”).value
    envio.Grupo.value=reko.fields(“grupo”).value
    envio.a.value=reko.fields(“inscripcion”).value
end sub

    sub registro_atras()

    reko.moveprevious()
    if reko.bof=true then
    msgbox”primer registro”
    reko.movenext()
    end if
    envio.text1.value=reko.fields(“folio”).value
    envio.text2.value=reko.fields(“matricula”).value
    envio.text3.value=reko.fields(“nombre”).value
    envio.text4.value=reko.fields(“apellidos”).value
    envio.Grupo.value=reko.fields(“grupo”).value
    envio.a.value=reko.fields(“inscripcion”).value
end sub
    sub comprobante()

    consulta2=”select * from tblfolio”

    rekofolio.open consulta2,conn,1,2
    rekofolio.addnew()

rekofolio.fields(“folio”).value=envio.text1.value
rekofolio.fields(“matricula”).value=envio.text2.value
rekofolio.fields(“nombre”).value=envio.text3.value
rekofolio.fields(“apellidos”).value=envio.text4.value
rekofolio.fields(“grupo”).value=envio.Grupo.value
rekofolio.fields(“inscripcion”).value=envio.a.value
```

```
    rekofolio.update()
    window.open(“comprobante3.html”)
    end sub
    </script>
        </HEAD>
        <BODY onload=entrar()>
<form name=”envio”>
    <p><label>Folio<input id=text1 name=text1><br>
    Matr&iacute;cula<INPUT id=text2 name=text2><BR>
    Nombre<INPUT id=text3 name=text3><BR>
    Apellidos<INPUT id=text4 name=text4><BR>
Grupo<SELECT id=select1 name=Grupo style=”HEIGHT: 22px; WIDTH: 133px”>
    <OPTION value=335>335</OPTION>
    <OPTION value=337>337</OPTION>
    <OPTION value=338>338</OPTION>
    </SELECT></p>
    Usted se ha inscrito a los siguientes cursos<input id=a name=a><br>

<p><input  onclick=registro_atras()  id=button1 name=button1 type=button  value=
Previo>
<input  onclick=registro_adelante()  id=button2 name=button2 type=button  value=
Siguiente>
<input  id=button3 name=button3 type=button value=Comprobante
onclick=comprobante()>
</form>
<a href=”principal.html”>Ir a la p&aacute;gina principal</a>

    </BODY>
</HTML>
```

Listado 8.5

```
<html>
<head>
<title>Comprobante de inscripci&oacute;n</title>
<script language=”VBScript”>
public reko,conn,consulta
    sub entrar()
    set conn=CreateObject(“ADODB.connection”)
    set reko=CreateObject(“ADODB.recordset”)
    conn.open “provider=msdasql;dsn=union2”
    consulta=”select * from alumno”
    reko.open consulta,conn,1,2
    reko.movelast()
```

```
    envio.text2.value=reko.fields(“matricula”).value
    envio.text3.value=reko.fields(“nombre”).value
    envio.text4.value=reko.fields(“apellidos”).value
    envio.text1.value=reko.fields(“folio”).value
    envio.grupo.value=reko.fields(“grupo”).value
    envio.a.value=reko.fields(“inscripcion”).value
    end sub



    </script>
</head>
<body onload=entrar()>
<h1>Comprobante</h1>
    <form name=”envio”>
    Folio<input id=text1 name=text1 ><br>
    <p><label>Matr&iacute;cula<INPUT id=text2 name=text2><BR>
    Nombre<INPUT id=text3 name=text3><BR>
    Apellidos<INPUT id=text4 name=text4><BR>
Grupo<SELECT id=select1 name=Grupo style=”HEIGHT: 22px; WIDTH: 133px”>
    <OPTION value=335>335</OPTION>
    <OPTION value=337>337</OPTION>
    <OPTION value=338>338</OPTION>
    </SELECT></p>
    Usted se ha inscrito a los siguientes cursos<input id=a name=a><br>
</form>
    </BODY>
</HTML>
```

Listado 8.6

```
<HTML>
        <HEAD>
<TITLE>Formulario de inscripci&oacute;n</TITLE>


<script language=”VBScript”>
public reko,conn,consulta
        sub entrar()
    set conn=CreateObject(“ADODB.connection”)
    set reko=CreateObject(“ADODB.recordset”)
```

```
    conn.open “provider=msdasql;dsn=union2”
    consulta=”select * from alumno”
    reko.open consulta,conn,1,2

    end sub

    sub envio_datos()

    reko.addnew()

    reko.fields(“matricula”).value=envio.text2.value
    reko.fields(“nombre”).value=envio.text3.value
    reko.fields(“apellidos”).value=envio.text4.value
    reko.fields(“grupo”).value=envio.Grupo.value
    IF envio.radio1(0).checked=TRUE THEN
    reko.fields(“turno”).value=”Matutino”
    ELSE
    reko.fields(“turno”).value=”Vespertino”
    END IF
    reko.fields(“comentarios”).value=envio.TEXTAREA1.value


    If envio.checkbox1.checked=true then
        a=a&”,”&”Acces”
        End If

    If envio.checkbox2.checked=true then
        a=a&”,”&”Excel”
        End If

    If envio.checkbox3.checked=true then
        a=a&”,”&”Word”
        End If

    If envio.checkbox4.checked=true then
        a=a&”,”&”Windows”
        End If

    If envio.checkbox5.checked=true then
        a=a&”,”&”Visual Basic”
        End If
reko.fields(“inscripcion”).value=a
reko.update()
reko.close
window.open (“comprobante2.html”)
    end sub
    </script>

</HEAD>
```

```
        <BODY onload=entrar()>
<form name="envio">
   <p><label>Matr&iacute;cula<INPUT id=text2 name=text2><BR>
   Nombre<INPUT id=text3 name=text3><BR>
   Apellidos<INPUT id=text4 name=text4><BR>
Grupo<SELECT id=select1 name=Grupo style="HEIGHT: 22px; WIDTH: 133px">
   <OPTION value=335>335</OPTION>
   <OPTION value=337>337</OPTION>
   <OPTION value=338>338</OPTION>
   </SELECT></p>
   <HR>
   <p></LABEL>Turno:</p>
   <INPUT id=radio1 name= radio1 type=radio value=0>Matutino
   <INPUT id=radio1 name= radio1 type=radio value=1>Vespertino
   <BR><BR><BR>
   Inscripción al curso de:<BR> <br>
   <INPUT  id=checkbox1 name=checkbox1 type=checkbox>Access  
   <INPUT  id=checkbox2 name=checkbox2 type=checkbox>Excel  
   <INPUT  id=checkbox3 name=checkbox3 type=checkbox>Word  
   <INPUT  id=checkbox4 name=checkbox4 type=checkbox>Windows  
   <INPUT  id=checkbox5 name=checkbox5 type=checkbox>Visual Basic</P>
   <P>Comentarios Adicionales.<BR>
<TEXTAREA id=TEXTAREA1 name=TEXTAREA1 style="HEIGHT: 89PX; Width:
317px">
   </TEXTAREA></p>
<p><input  onclick=envio_datos()  id=button1 name=button1 type=button  value=Enviar>
<input id=reset1 name=reset1  type="reset" value=Reset></p>
<a href="principal.html">Ir a la p&aacute;gina principal</a>
</form>
        </BODY>
</HTML>
```

Listado 8.7

A continuación se muestra la organización de los archivos, es recomendable que la organización sea descriptiva ver figura 8.19:

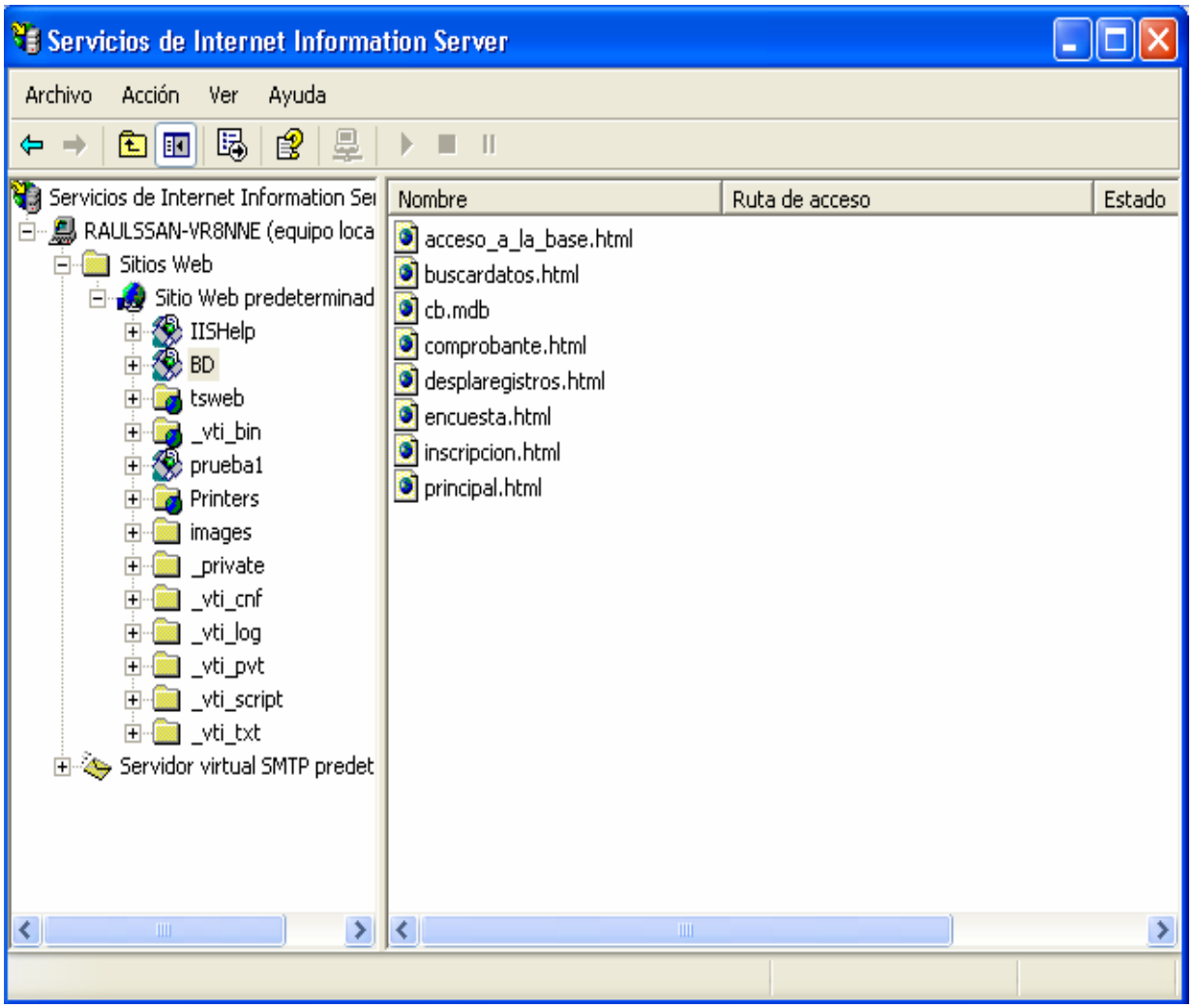


Figura 8.19

Con todos estos conocimientos, ya podemos dar servicios de Internet y ya estamos en posibilidades de usar de forma óptima nuestra base de datos.

Vamos a apreciar nuestra página principal en un servidor de Internet:

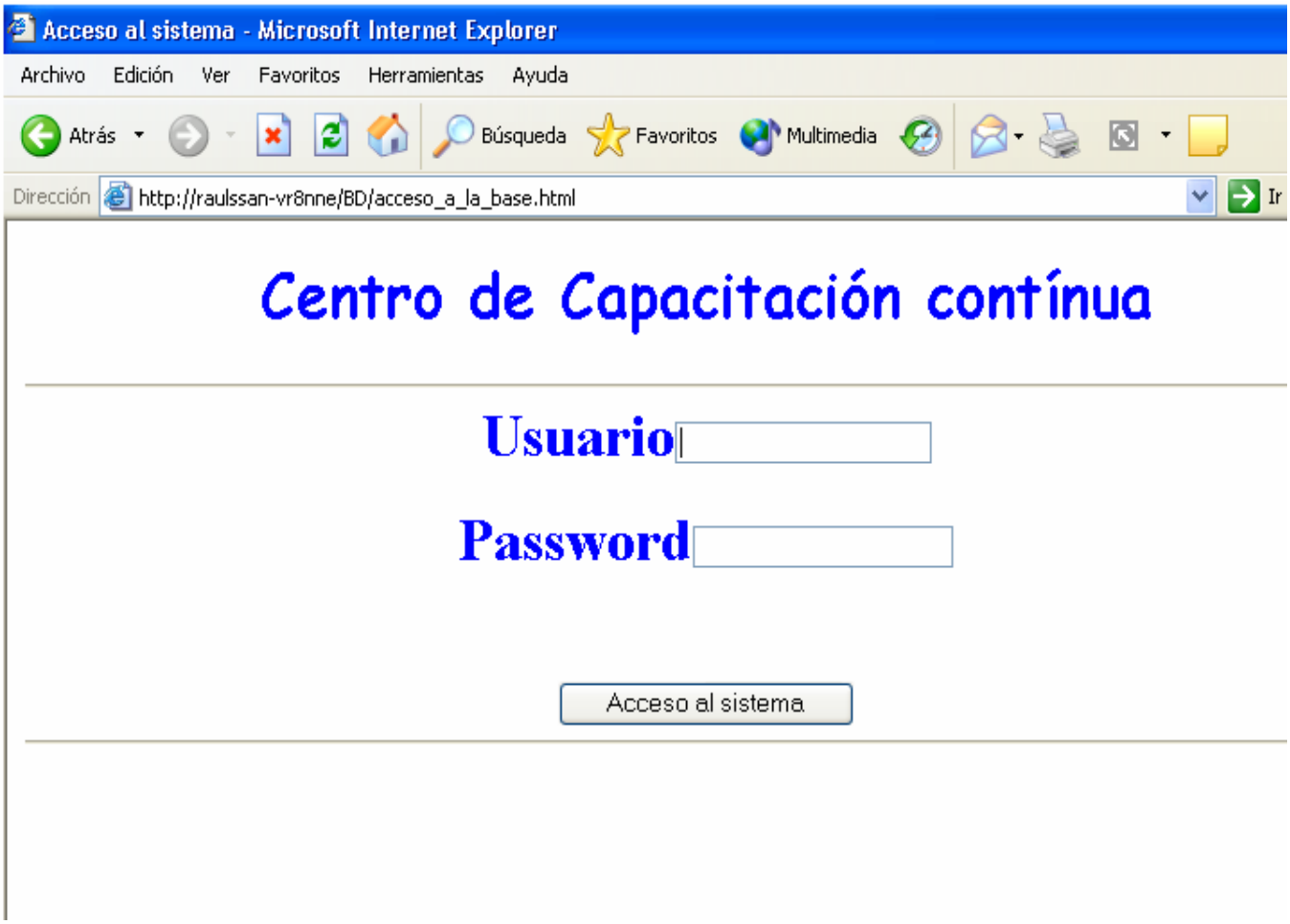
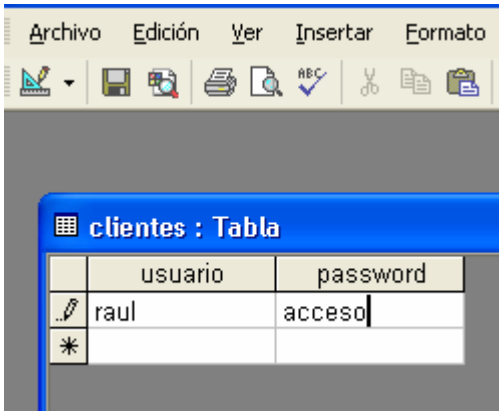


Figura 8.20

Como apenas acabamos de crear nuestro sistema, y ya está activado nuestro origen de datos, todavía no es posible acceder debido a que todavía no les hemos asignado una clave de acceso a nuestros usuarios.

Para realizar el acceso, proporcionamos una clave desde el servidor, en nuestra tabla clientes, ver figura 8.21.



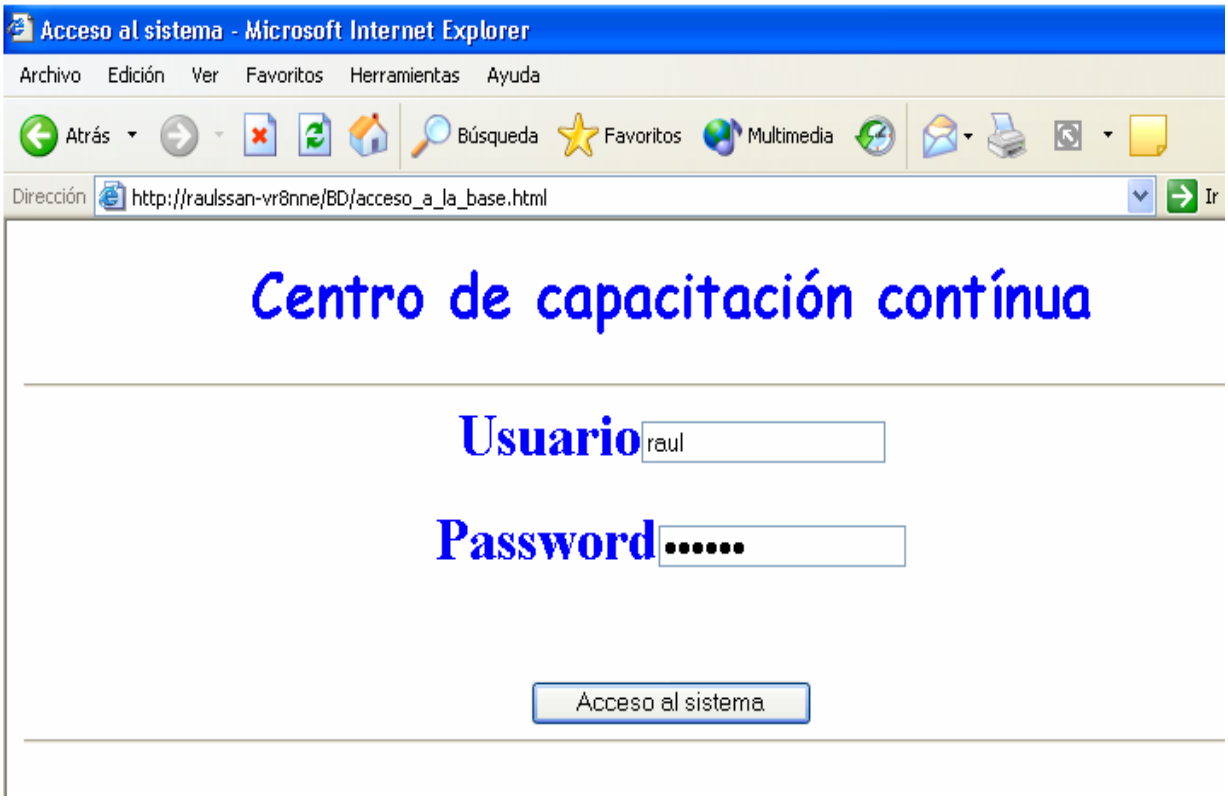


Figura 8.22

Como el administrador ya nos proporcionó una cuenta, ya es posible acceder desde Internet. Tecleamos nuestro nombre de usuario y después nuestro password para realizar una inscripción. Ver figura 8.23.

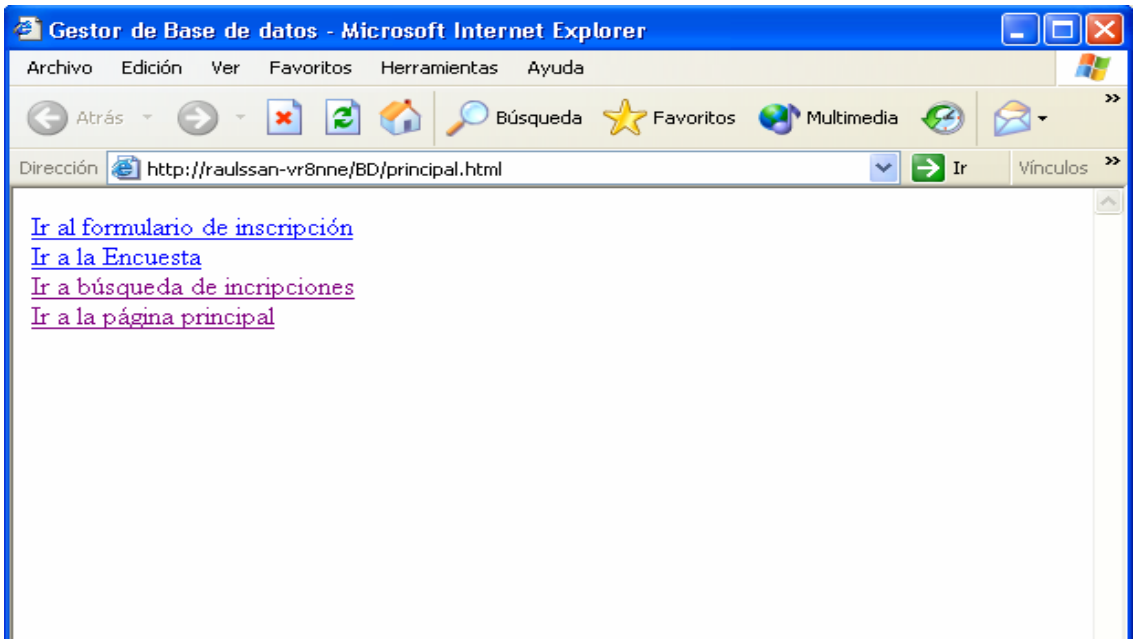


Figura 8.23

Oprimimos el hiperenlace Ir al formulario de inscripción, comenzamos a llenar nuestro formulario, cuando hayamos terminado, oprimimos el botón Enviar para que se nos dé el comprobante de inscripción.

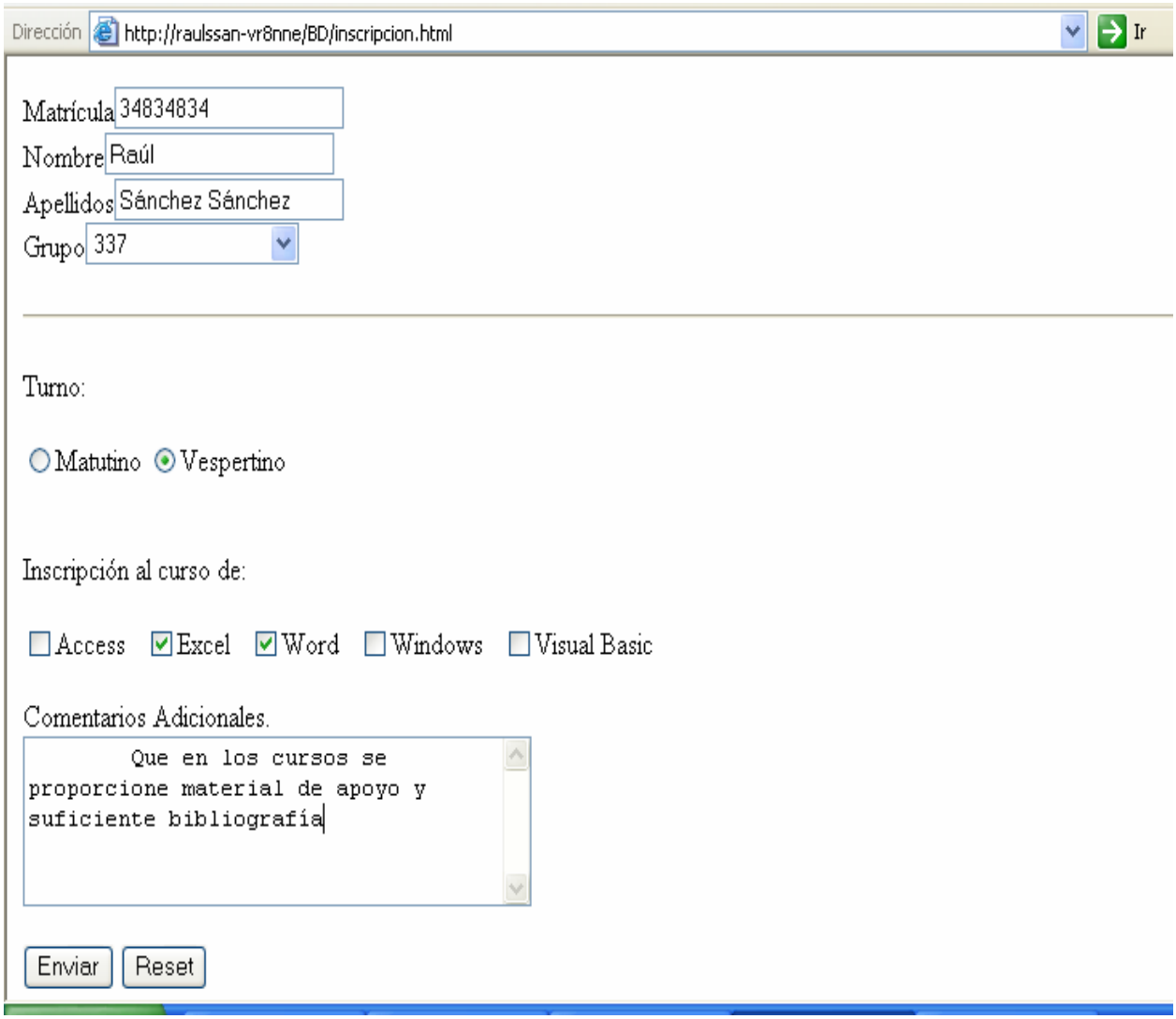


Figura 8.24

Cuando oprimimos el botón enviar de nuestro formulario de inscripción, aparecerá nuestro comprobante de inscripción. Ver figura 8.25.

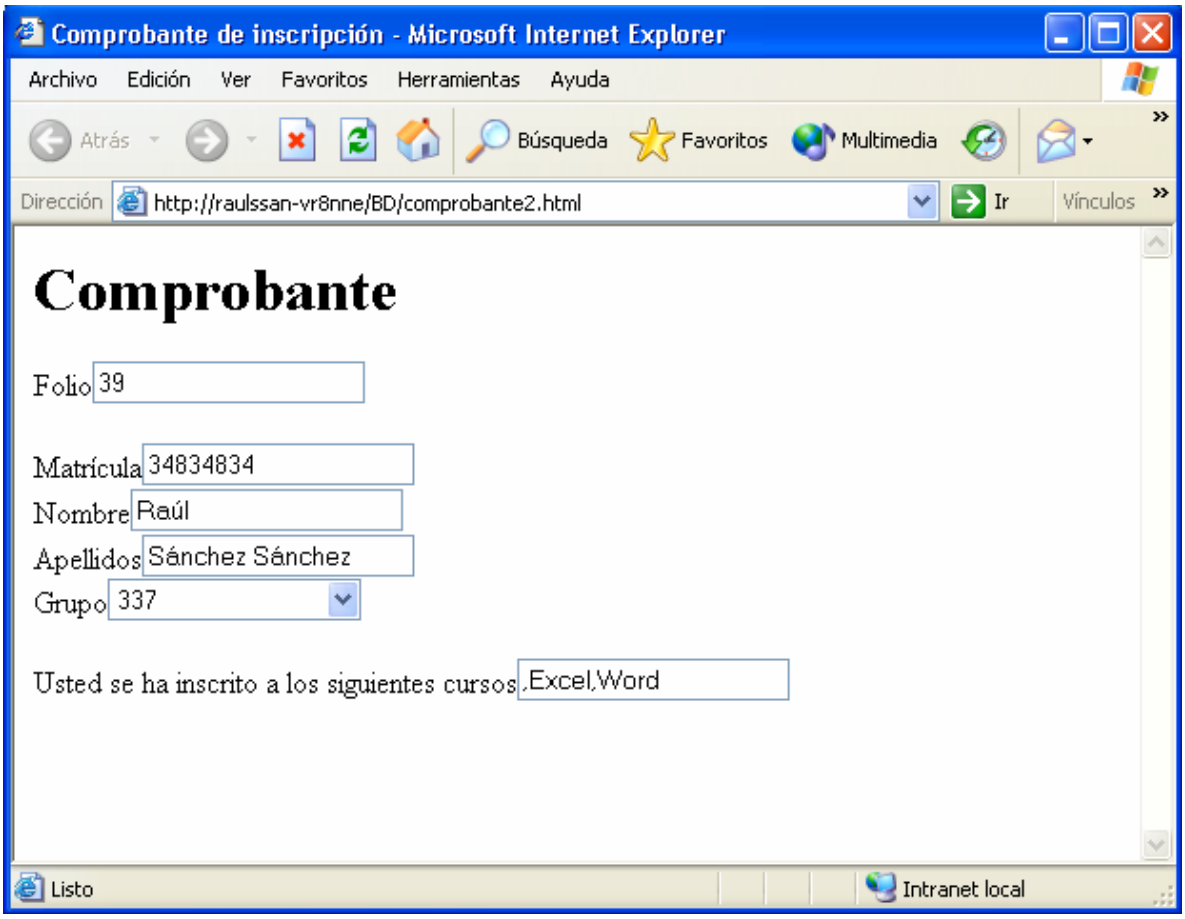


Figura 8.25

Al aparecer nuestro comprobante de inscripción, los datos que introducimos se guardarán en nuestro banco de datos.

En este capítulo utilizamos objetos para abrir bases de datos, para acceder a objetos de datos Active X (ADO) para esto, utilizamos los siguientes pasos:

- ✓ Crear una instancia de los objetos necesarios de ADO.
- ✓ Configurar sus propiedades.
- ✓ Llamar a sus métodos(Los comandos en Visual Basic Script).
- ✓ Recibir los resultados.

Hecho lo anterior, utilizamos el origen de datos ODBC, para tener acceso a los datos de la aplicación en cuestión.

En este caso fue fácil hacer uso de los objetos, cosa que se empieza a complicar cuando uno, empieza a crear sus propias clases y objetos, tal hecho requiere de un enorme esfuerzo de abstracción y empaparse del problema en cuestión, casi todos los lenguajes manejan objetos y algunos como Java permite crearlos por medio de clases. En estos últimos años han salido una buena cantidad de productos de software para aplicaciones orientadas a objetos, desafortunadamente, este no es el caso para las bases de datos más elaboradas, hay algunos buenos intentos como el Microsoft Access, pero solamente permite la creación de bases de datos a partir de objetos ya hechos por esta empresa. Además, la creación de datos en Web se limita a la administración básica de las bases de datos.

IX. Instalación y configuración de un servidor WWW

9.1. Servidor WWW.

Un servidor WWW es un tipo de servidor que alberga páginas Web para dar servicios dentro del WWW en Internet.

Estos son los servidores WWW más populares:

- ✓ Apache Web Server.
- ✓ Microsoft IIS.
- ✓ IPlanet.
- ✓ Zeus.

Si usted desea saber cuales son los servidores más populares le recomiendo visitar el sitio http://news.netcraft.com/ es un sitio de encuestas de los servidores más populares del mundo que se actualiza mes con mes.

1.2. Instalación de Servicios de Internet Information Server de Microsoft.

Primero que nada hay que asegurarse que se cuenta con el disco del sistema, en este caso nosotros disponemos del Sistema Operativo Windows XP. Lo que vamos hacer a continuación es similar a lo que se hace en las versiones recientes de Windows tales como Windows 2000, Me, XP y Windows.net Enterprise Server 2003 entre otros.

Ya que esté instalado en disco del sistema, nos vamos al panel de control, seleccionamos agregar o quitar programas, ya que esté abierto el cuadro Agregar o quitar componentes de Windows y damos un clic en la casilla de verificación de Servicios de Internet Information Server Ver figura 9.1

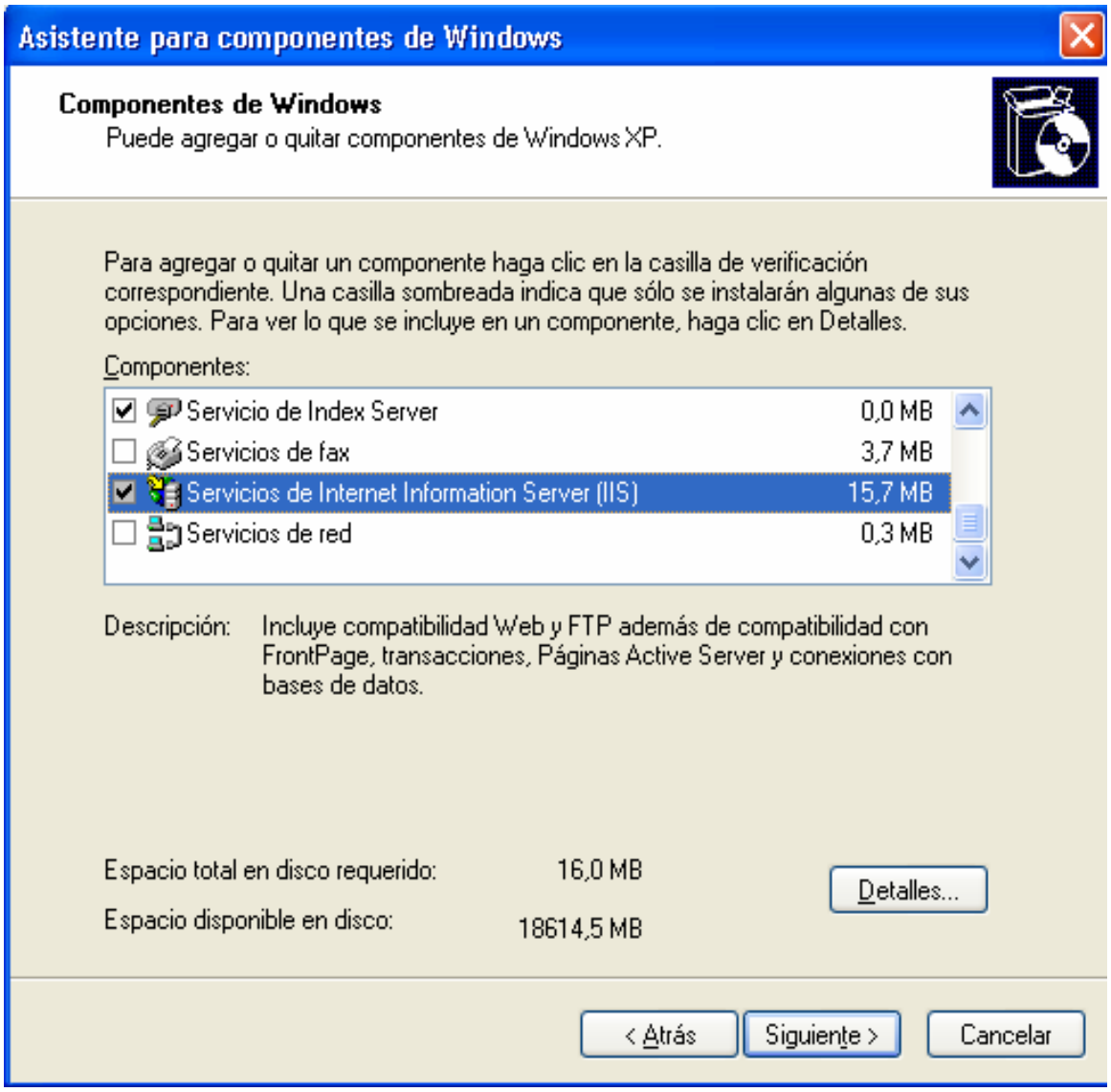


Figura 9.1

Oprimimos el botón siguiente del panel de la figura 9.1

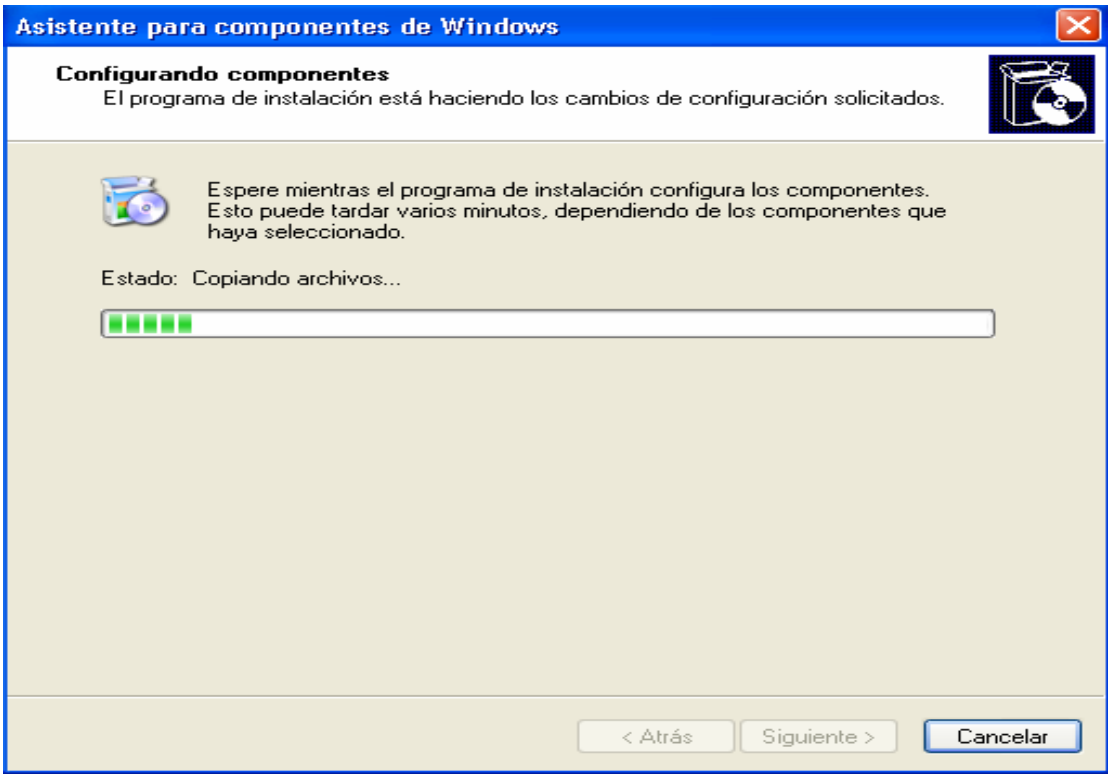


Figura 9.2

Cuando se hayan terminado de instalar todos los archivos se mostrará la ventana que finaliza el asistente cuando esto suceda oprima el botón finalizar. Ver figura 9.3.

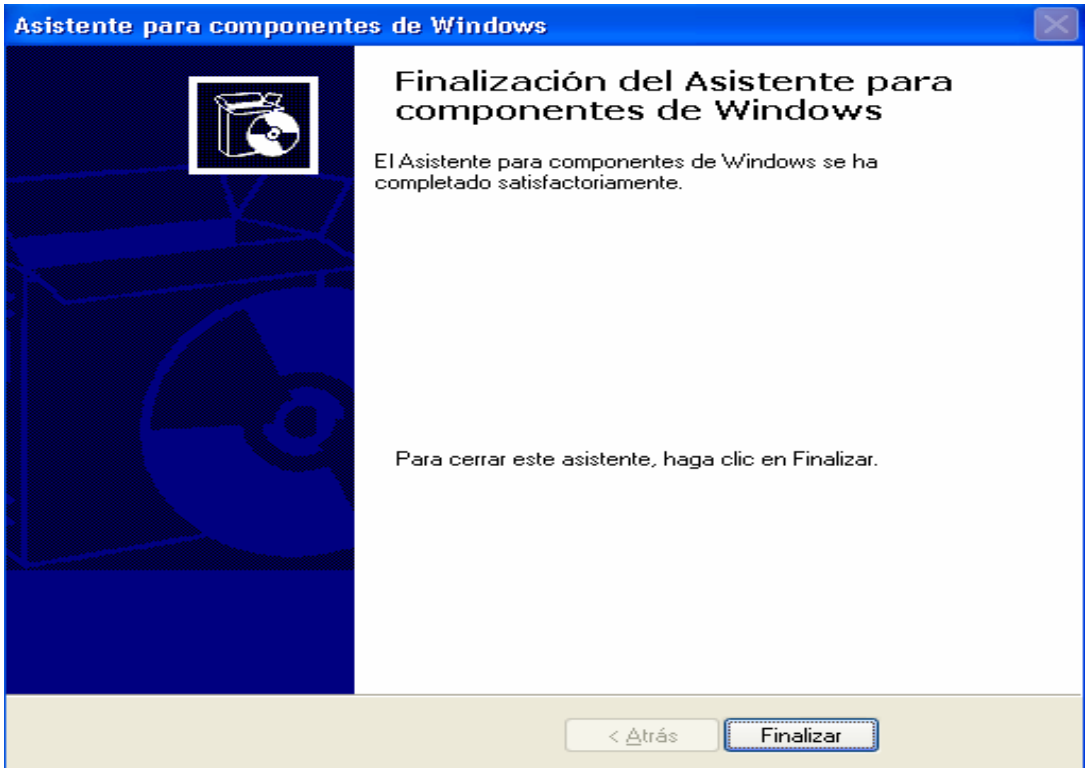


Figura 9.3

En la opción de herramientas administrativas aparece el servidor de Servicios de Internet Information Server de Microsoft de la figura 9.4, vamos a crear un acceso directo para que lo tengamos al alcance.

Figura 9.4

Cuando cargamos nuestro servidor, ya está listo para recibir cualquier tipo de documentos Web. Ver figura 9.5.

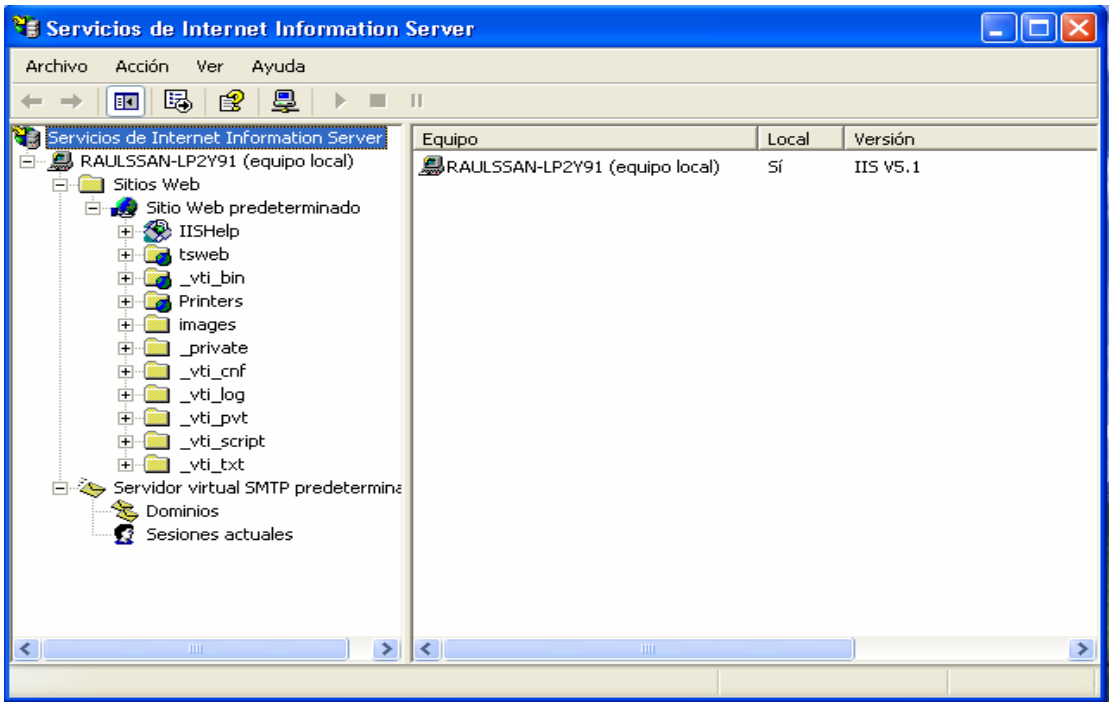


Figura 9.5

Servicios de Internet Information Server

Servicios de Internet Information Server 5.1 (IIS) es el servicio Web de Windows XP que facilita la publicación de información en una *Intranet.*

Servicios de Internet Information Server (IIS) simplifica la publicación de información en Internet o en la Intranet. IIS incluye una amplia gama de funciones administrativas para controlar sitios Web y el servidor Web. Con funciones de programación como páginas Active Server (ASP), puede crear e implementar aplicaciones Web flexibles y escalables. IIS no se instala de manera predeterminada, pero puede agregarlo mediante el cuadro de diálogo **Agregar o quitar programas** del Panel de control.

Como podemos apreciar el ambiente de Servicios de Internet Information Server consta de dos servidores:

- ✓ Un servidor Web: En este servidor que funge como el Sitio Web predeterminado es donde nosotros vamos almacenar nuestras páginas Web en lenguajes como HTML, Visual Basic Script, Java Script , PHP, ASP, MySQL, Applets de Java y un gran etcétera.
- ✓ Un servidor de correo electrónico: Este servidor de correo electrónico <<Servidor virtual SMTP predeterminado>>, el servicio SMTP de Microsoft utiliza el Protocolo simple de transferencia de correo (SMTP), estándar en Internet, para transportar y entregar mensajes. El servicio SMTP también le ofrece control sobre el enrutamiento y la entrega de mensajes, lo que proporciona comunicaciones seguras.

Vamos a comprobar si nuestro servidor se instaló correctamente, para esto vamos a teclear la siguiente dirección en nuestro navegador: http://localhost/localstart.asp , si lo que hicimos es correcto, nuestro navegador desplegará lo se muestra en la figura 9.6.

Figura 9.6

9.3. Servidor Apache.

El servidor apache es el servidor más popular y seguro del mundo, además de que es completamente gratuito e incluye el código fuente. El servidor apache se puede bajar gratuitamente de la siguiente dirección: http://www.apache.org/.

Ya que estemos en la página, procedemos a bajarlo y seguir las instrucciones.

Instalación del servidor apache en Windows.

La instalación del servidor apache en Windows es muy sencilla, simplemente ya que haya bajado el servidor Apache Web Server y lo haya guardado en una carpeta, simplemente hay que darle un doble clic al icono de instalación. Ver figura 9.7:


apache_2.0.5
2-win32-x86-n
o_ssl

Fig. 9.7

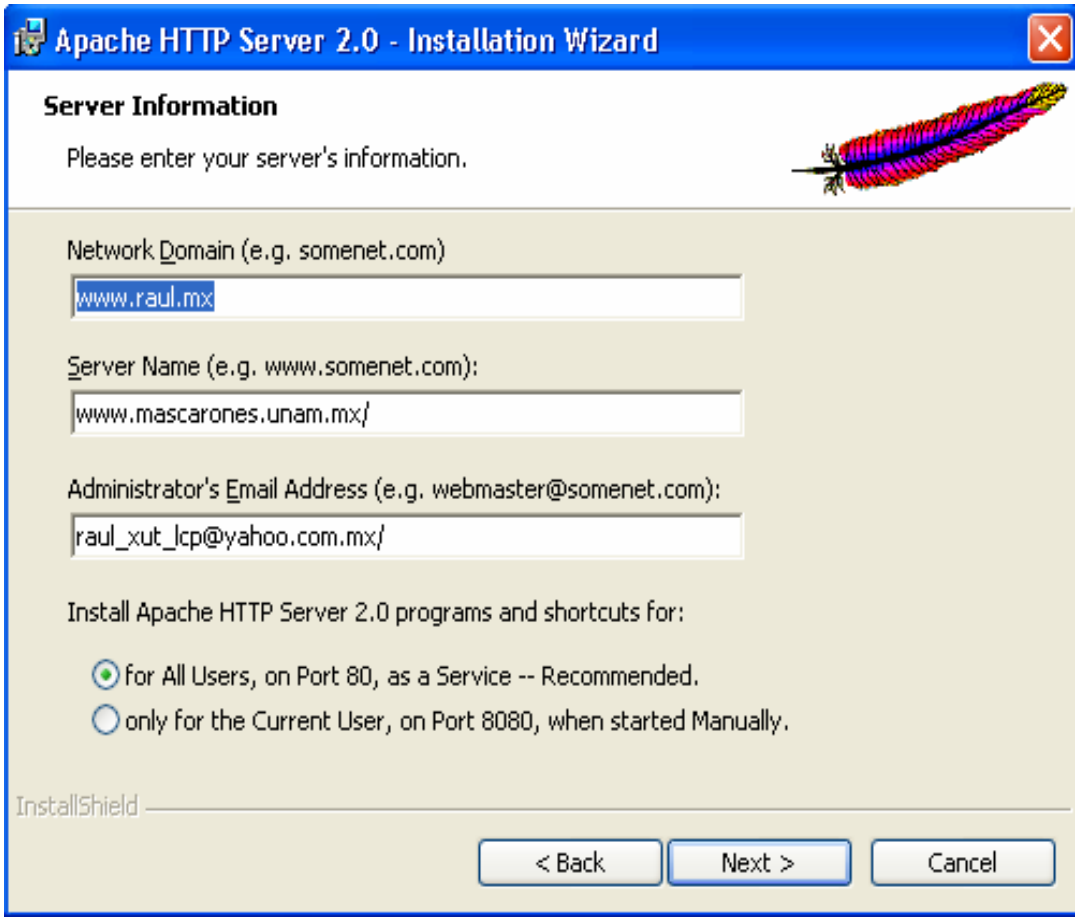

Apache HTTP Server 2.0 - Installation Wizard
Server Information
Please enter your server's information.
Network Domain (e.g. somenet.com)
www.raul.mx
Server Name (e.g. www.somenet.com):
www.mascarones.unam.mx/
Administrator's Email Address (e.g. webmaster@somenet.com):
raul_xut_lcp@yahoo.com.mx/
Install Apache HTTP Server 2.0 programs and shortcuts for:
for All Users, on Port 80, as a Service -- Recommended.
only for the Current User, on Port 8080, when started Manually.
InstallShield
< Back
Next >
Cancel

Fig. 9.8

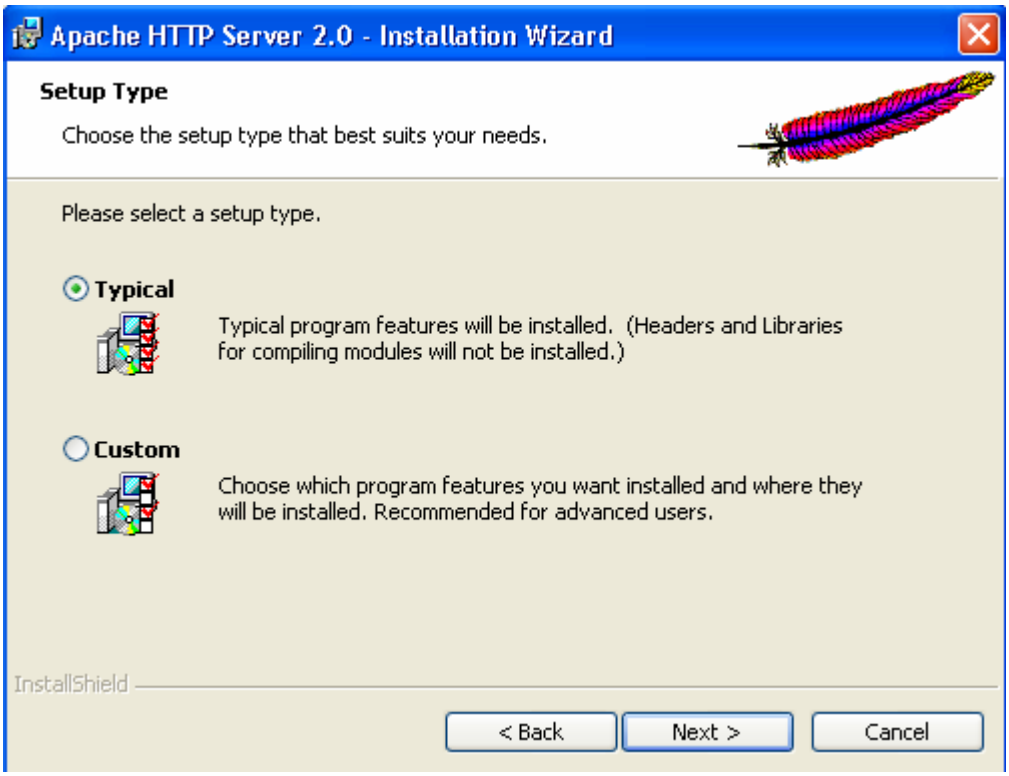


Fig. 9.9

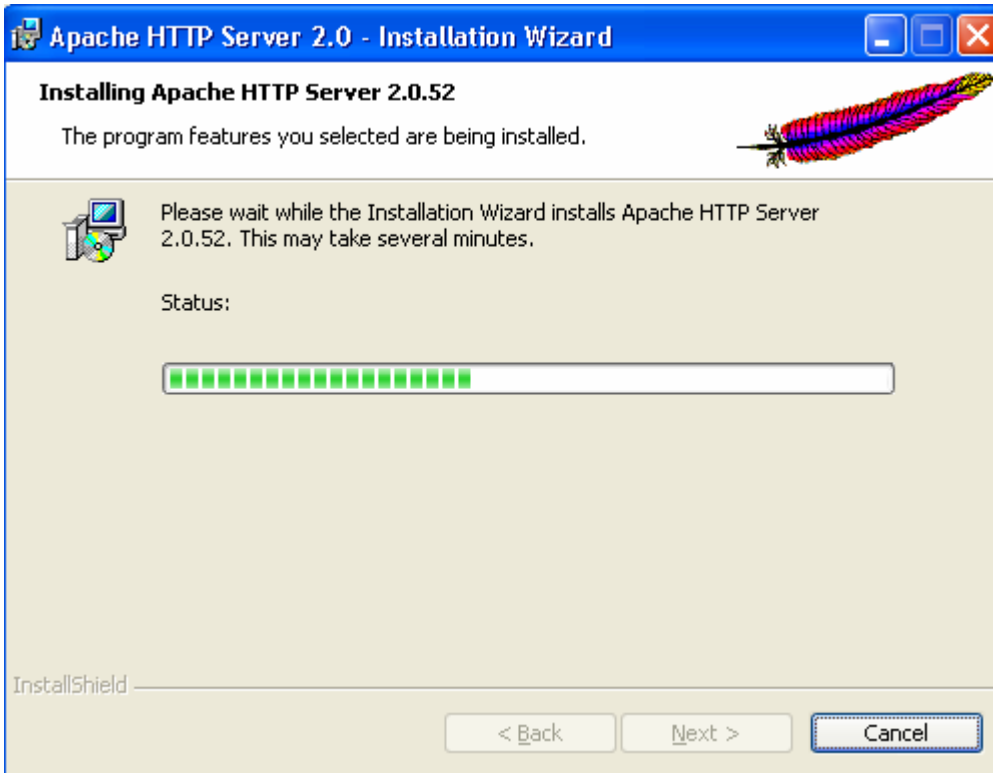


Fig.9.10

Ya que se haya terminado de instalar y hayamos aceptado los términos, y de haber leído la licencia, el servidor se visualiza así:

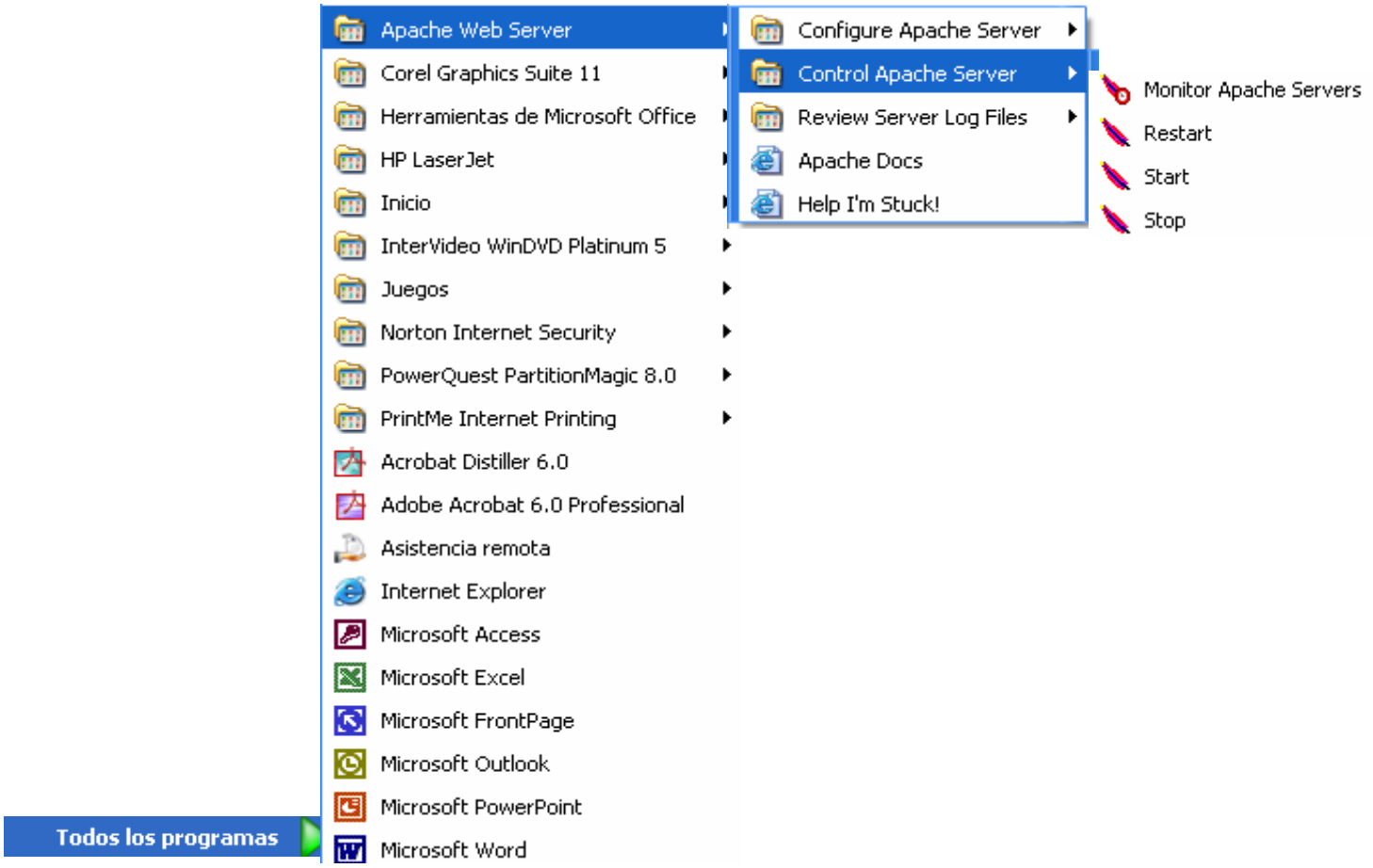


Fig.9.11

En la figura anterior se puede apreciar que es muy sencillo su manejo ya que si queremos activar el servidor apache, simplemente damos clic en Start Apache, si queremos desactivar el servidor, damos clic en Stop Apache. Aunque el IP de nuestro host puede variar, una forma de obtenerlo es con la instrucción de MS-DOS: ipconfig. Ver figura 9.11:

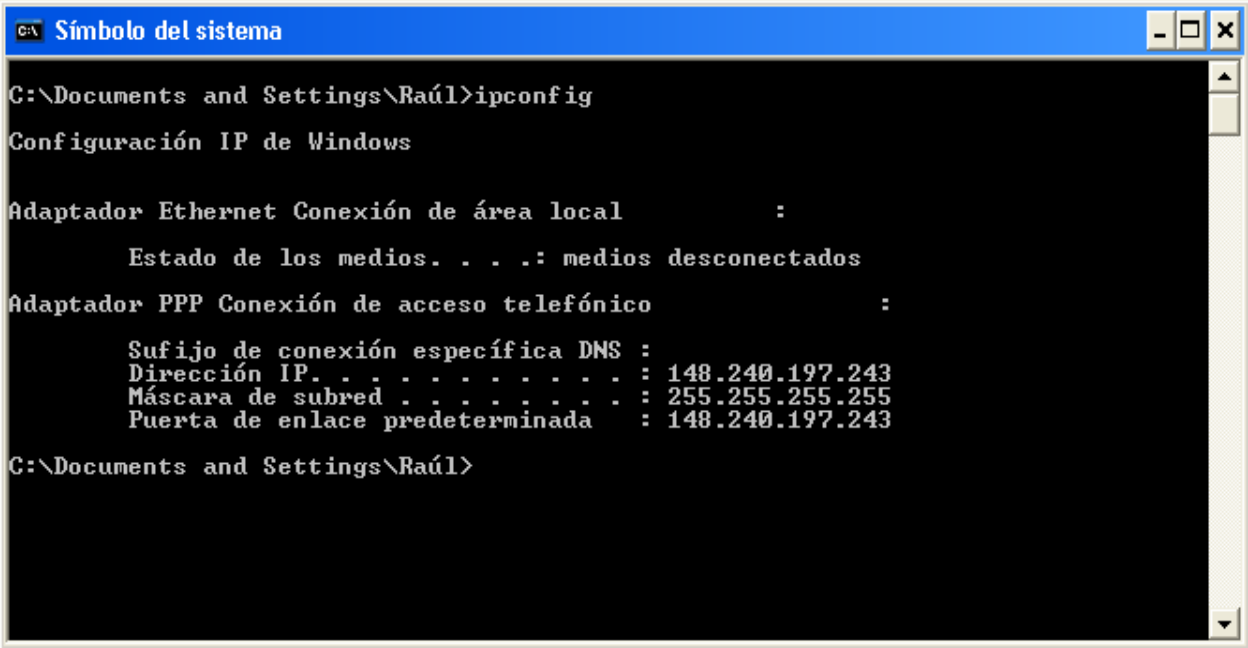


Fig. 9.12

Pruebas de apache.

En la figura 9.11 podemos apreciar los menús, centrémonos en el menú **Control Apache Server,** este menú tiene cuatro comandos:

- ✓ Monitor Apache Server. Es un panel desde el cuál podemos monitorear, al servidor. Ver figura 9.13:

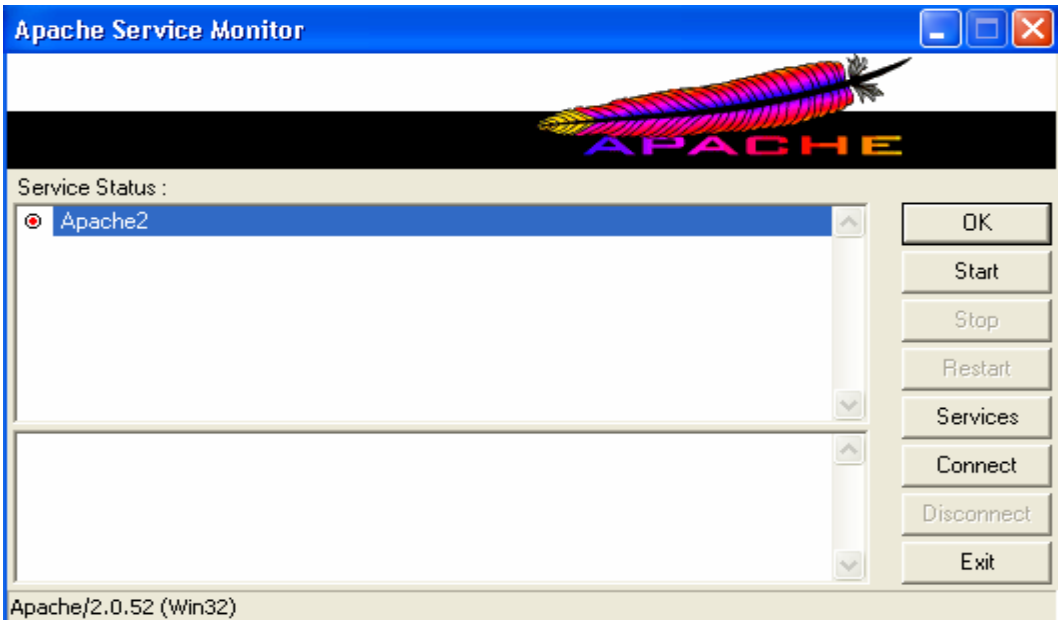


Fig. 9.13

- ✓ Restart. Restablecer el servidor Apache Web Server.

- ✓ Start. Arrancar el Servidor Apache Web Server.
- ✓ Stop. Desactivar el Servidor Apache Web Server.

Vamos a arrancar el servidor apache Web Server. Activando el comando **Start**. Cuando hayamos hecho esto, vamos a cargar el navegador de Internet de nuestra preferencia, colocando la dirección URL http://www.127.0.0.1, si lo que hicimos es correcto aparecerá lo siguiente:

Fig. 9.14

Es recomendable, probar con estas direcciones adicionales: http://localhost o bien con el IP de nuestra máquina como lo hicimos en la figura 9.12, con el comando ipconfig para localizar nuestro IP.

Apache, es un servidor altamente configurable, pues se administra por medio de aproximadamente 200 directivas. Una directiva es simplemente un comando al que obedece apache. Apache lee directivas de los archivos de configuración.

Conclusiones

1.- Con el Enfoque Ortientado a Objetos podemos representar cualquier problema de cualquier área del conocimiento, pero su limitación sigue siendo sus muchas pretenciones. Es poca la literatura que presente en forma concreta los alcances y las limitaciones del Enfoque Orientado a Objetos, y esto se debe a que este enfoque sigue siendo novedoso.

2.- Nos dimos cuenta de que las aplicaciones tanto las que manejan Objetos y las que permiten crearlos, tienen un gran alcance, aunque presentan limitaciones que en algunas ocasiones solo pueden ser cubiertas si se utilizan otras herramientas para programar, si bién en XML se trabaja con objetos, todavía hay serias deficiencias que pueden ser cubiertas con HTML y con HTML dinámico o bién haciendo uso de otros lenguajes de programación, lo que a la larga puede generar el famoso espagueti, y esto es una contradicción en la Programación Orientada a Objetos. Sin embargo XML es una herramienta que a medida que pase el tiempo va a ser cada vez más imprescindible. Algo similar ocurre con lenguajes como C++, Visual Basic y Java.

3.- La Orientación Objetos es buena pero hasta cierto punto.

4.- Con los objetos se pueden modelar y desarrollar aplicaciones aceptables, pero la reutilización del código que es lo que se persigue en un sentido estricto, esto es: desarrollar uno o más componentes que tienen una tarea específica y probada que pueda utilizarse en cualquier circunstancia y lugar, no es una garantía. Siendo consecuentes con esto, los objetivos se cumplieron en parte.

5.- La globalización de componentes comprobados que puedan ser utilizados en cualquier parte, tiempo y lugar por el usuario como si armara un pequeño avioncito, no es algo que sea frecuente en Internet.

Aportaciones

- Crear aplicaciones sencillas internet/intranet que permiten darnos cuenta de las limitaciones que presentan algunas aplicaciones que trabajan con objetos, por ejemplo XML trabaja objetos que nos permiten crear etiquetas al infinito, su limitación es permitir a un usuario ser el autor de la sintaxis de sus códigos, esto genera una suerte de torre de Babel, una contradicción si consideramos que actualmente al desarrollarse aplicaciones Internet/Intranet, se trabaja con grupos multidisciplinarios. Sin embargo el XML es el lenguaje del futuro para el diseño de sitios Web, son páginas seguras y confiables, es imposible que estos sitios sean invadidos por virus, experiencia que personalmente constaté, pero este futuro deseable todavía no nos alcanza, es fácil darse cuenta de esto al ver lo escasos que son los sitios Web diseñados con XML, ya no se diga los dinámicos utilizando JAVA.

Bibliografía.

1. Así son las Intranets. Cap 1, 2. Tyson Creer. 1997 Mc Graw-Hill.
2. Kit de recursos de Intranet Cap 1, 2. Prakash Ambegaonkar. Mc Graw-Hill. 1997.
3. Creación de Sitios Web con Access. James J. Hobuss, Prentice Hall, 2000. pp 9-14.
4. Curso IBM de Programación. Multimedia Ediciones, S.A. P 341-355.
5. Curso Práctico de Programación en Visual Basic. Número 1. Prensa Técnica. P 3-24.
6. Aprendiendo Visual Basic 6 en 21 Días. P 7-29.
7. HTML 4. Elisabeth Castro. Prentice Hall. 2000.
8. Perl, CGI y JavaScript. Anaya Multimedia. 2000.
9. Superutilidades para JavaScript. Jeff Frentzen y Henry Sobotka.1999
10. Diseño de páginas Web interactivas con JavaScript. Juán Carlos Orós. Alfaomega ra-ma. 2000.
11. Outlook 2002 Sin problemas. Don Gilbert y Julia Nelly. Mc Graw-Hill. 2002.
12. Programación con Access en 24 horas. Paul Kimmel. Prentice-Hall. 2002.
13. Programación Avanzada con Access 2000 F. Scout Barcker. Prentice Hall. 1999.
14. Access 2002. Manual Avanzado. Miguel Ángel Martín Tardío. Anaya Multimedia. 2001.
15. Programación con XML. Ricardo Eito Brun. Anaya Multimedia. 2001.
16. XML con ejemplos Benoit Marchal. Pearson Educación. 2001
17. La Biblia del Servidor Apache 2. Mohamed J. Kabir. Ed Anaya Multimedia.
18. Aprenda Desarrollo de bases de datos web ya. Jim Buyens. Mc-Graw Hill.
19. Aprendiendo UML en 24 horas. Joseph Schmuller. Prentice Hall.
20. El Lenguaje Unificado de Modelado. Grady Booch, James Rumbaugh e Ivar Jacobson. Addison Wesley. Cap1. 1999.
21. UML Gota a Gota. Martin Fowler con Kendall Scott. Addison Wesley. México 1999.