UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE ESTUDIOS SUPERIORES
A C A T L A N

SISTEMA DE INFORMACION ESCOLAR EN LINEA CON TECNOLOGIA ASP PARA UNA UNIVERSIDAD PUBLICA DEL PAIS

SEMINARIO TALLER EXTRACURRICULAR

QUE PARA OBTENER EL TITULO DE LICENCIADO EN MATEMATICAS APLICADAS Y COMPUTACION

P R E S E N T A :

Manuel Alejandro Montes de Oca Yemha

Asesor: Lic. Maritza Nova Juárez.

Octubre 2005

Índice

Pág.

Introducción

Este trabajo trata de la elaboración de un prototipo de sistema. El prototipo es de un sistema de información escolar en línea, desarrollado con tecnología ASP. Además de tratar los aspectos de cómo se desarrolla el prototipo, como requerimientos, modelo, diseño y desarrollo, se explica los elementos utilizados en la tecnología ASP para su desarrollo, y de la seguridad en la autentificación para el ingreso de datos.

El primer capítulo describe que es la tecnología ASP, cómo funciona y una breve historia de esta. También se tratan tecnologías que trabajan de forma similar y otras tecnologías para desarrollo de aplicaciones Web, con el fin de tener presente las ventajas y desventajas de ASP. Se plantean posibles formas para llegar a una autentificación segura.

El segundo capítulo muestra cómo desarrollar una aplicación con la tecnología ASP. Expone también los elementos necesarios para entender el desarrollo del prototipo, como la creación de formularios y el acceso a bases de datos.

En el tercer capítulo se establecen los requerimientos del sistema, explicando porque la tecnología ASP es viable para el desarrollo del prototipo. Se explica en forma general como trabajan los servicios principales del prototipo y cómo está formada la base de datos.

El cuarto capítulo explica a detalle el funcionamiento de los servicios principales, mediante secciones de código. También se muestra el proceso de implantación y la forma de autentificación del prototipo.

El seminario es de gran ayuda para el proceso de titulación. Traté con otras opciones, pero hubo bastantes tropiezos en el camino. En algún momento llegue a pensar que este proceso era prácticamente imposible. Si ahora estas leyendo estas líneas, de un trabajo de un exalumno que se tituló en 2005, es gracias a un seminario, las personas que lo organizan, y quienes lo imparten.

Capítulo 1: Conceptos Base

Este capítulo es el marco teórico del trabajo. Incluye una explicación de qué es la tecnología ASP, su funcionamiento en breve y tecnologías relacionadas a esta.

1.1 Breve historia de ASP

En los comienzos de la World Wide Web (WWW), a principios de los noventas, no había páginas dinámicas, solo páginas con contenido estático. El contenido no podía ser cambiado más que por el Webmaster (Administrador Web).

Era bueno tener acceso a toneladas de datos, pero la gente necesitaba que la información desplegada fuera personalizada, es decir, que estas páginas tuvieran solo lo que estaban buscando.

Gente de negocios, deseaba poder mostrar a un posible cliente, una lista de productos específicamente diseñada para ese cliente en lugar de una lista genérica de productos diseñada para todos.

Para resolver este problema, fue creada la tecnología CGI. Esto permitió al Webmaster usar un pequeño programa para crear una nueva página Web a partir de las entradas del usuario, la fecha o la hora, información almacenada en la computadora del usuario, entre otras. Entonces, las páginas Web ya podían ser dinámicas.

Los Scripts CGI necesitaban código en un lenguaje de programación y los más populares eran C y Perl. El problema era que estos lenguajes no eran tan sencillos para muchos administradores.

Una alternativa para el diseño dinámico de páginas Web fue introducida por Microsoft, diseñada por Denalí en 1996. La nueva tecnología estaba basada en Visual Basic Script, además se agregó una aplicación "Servidor Web" a las versiones de Windows (IIS y PWS). Debido a que este Script era más accesible, permitiría que más gente creara páginas Web dinámicas. Esta tecnología llamada ASP (Active Server Pages) fue bien aceptada.

Actualmente ASP corre en otras plataformas además de Windows. Se pueden correr páginas con esta tecnología en plataformas como Unix, Linux, Solaris y otros sistemas operativos.

Un problema para ASP fue que las aplicaciones Web habían literalmente sobrecrecido a lo que es la tecnología ASP. Los sitios Webs eran tan grandes y requerían tanto código que acababa pareciendo un plato de spaghetti en vez de una operación organizada. En otras palabras, era difícil de leer este código, por tanto, anexar código para alguna función requería de mucho tiempo o de la ayuda de quien lo escribió.

ASP no se prestaba bien para estos sitios grandes desde el punto de vista de diseño. La razón principal es que ASP no soportaba la Programación Orientada a Objetos. Entre más larga sea una aplicación, existe una mayor necesidad de una Programación Orientada a Objetos. Por eso ASP ha sido reescrito como ASP.NET. Se recomienda en esta tecnología que uno escriba el código en C#, el nuevo lenguaje Microsoft para .NET. Sin embrago, se puede usar cualquier lenguaje en cuanto sea .NET, como VB.NET.

Aun con esto, ASP no es hoy el entorno de programación número uno para páginas Web Dinámicas, esa corona pertenece a PHP; ASP es un sólido número dos. Esto no significa que se deba utilizar la herramienta número uno, ni que esta sea mejor. Se debe analizar la plataforma (tipo de servidor y sistema operativo) donde se ejecutará la página Web, si existen otras páginas Web del sitio diseñadas con alguna tecnología, o incluso el lenguaje que se quiere usar.

1.2 Conceptos relacionados, tecnologías similares y afines.

En esta parte, se tratarán términos relacionados a tecnología Web, para que dentro de este panorama, se pueda ubicar a la tecnología ASP.

Empecemos por Internet y Web, que muchas veces se utilizan de manera indistinta, cuando en realidad no son sinónimos, sino dos términos relacionados. La Internet es una red masiva de redes, una infraestructura de redes. Conecta millones de computadoras de forma global, formando una red en la cual cualquier computadora se puede comunicar con la otra en tanto éstas estén conectadas a la Internet. La información que viaja a través de Internet lo hace a través una variedad de protocolos. La World Wide Web, o simplemente Web, es una forma de acceder a la información por medio de Internet. Es un modelo para compartir información muy usado en Internet, por lo que a veces se piensa que es la Internet. El punto es que no hay que confundir Web con la Internet.

La Web usa el protocolo HTTP (Hyper Text Transfer Protocol), solo uno de los empleados en la Internet para transmitir datos. Se usa este protocolo para permitir a las aplicaciones

intercambiar información. La Web también utiliza navegadores, como el Internet Explorer o el Netscape para acceder a documentos Web llamados Páginas Web que están ligadas una con otra a través de Hipervínculos. Los documentos Web pueden contener gráficas, sonidos, texto y video.

La Web es solo una de las formas en que la información puede ser compartida sobre la Internet. La Internet, no la Web, es también usada por el correo electrónico que depende de un SMTP (Simple Mail Transfer Protocol), grupos de noticias Usenet, mensajes instantáneos y FTP (File Tranfer Protocol). En resumen, la Web es solo una porción de la Internet, una muy gran porción,

Entonces, la tecnología ASP, utiliza la Internet, específicamente la Web, para poder establecer una conexión entre el servidor y el usuario por medio de un explorador, donde el usuario puede ver la información que produjo un servidor

Una página ASP además de los Scripts, puede contener código HTML (Hyper Text Markup Language) que es un código para la creación de páginas Web Estáticas. Este código define la estructura y distribución de un documento Web usando una variedad de etiquetas y atributos.

Una página ASP debe tener código HTML además de los Scripts, ya que este código da el formato de visualización al documento, y el Script corresponde a lo que es la aplicación. También se puede incluir código XML (Extensible Markup Language), que es casi lo mismo que HTML, solo que nos permite personalizar las etiquetas que se manejan, lo que permite la definición, transmisión, validación e interpretación de datos entre aplicaciones y entre organizaciones.

Como era de suponerse, y se menciona anteriormente, ASP no es una tecnología única en su clase, tenemos las tecnologías PHP y CGI.

CGI (Common Gateway Interface),

Un programa CGI es diseñado para aceptar y regresar datos. El programa puede ser escrito en un lenguaje de programación, incluyendo C, Perl, Java. Los programas CGI son una forma común para interactuar de forma dinámica con los usuarios. Muchas páginas HTML que contienen formularios, usan un programa CGI para procesar los datos una vez que son enviados. Un problema con CGI es que cada vez que un Script CGI es ejecutado, un nuevo proceso es ejecutado. Para sitios Web muy concurridos, esto puede bajar la velocidad de respuesta notablemente.

PHP (Hypertext Preprocesor)

Es un Script incrustado al código HTML, de uso libre para crear páginas Web dinámicas. En un documento HTML, el Script PHP es encerrado con etiquetas especiales., así el autor puede brincar entre HTML y PHP, y como el código es ejecutado en el servidor, el cliente no puede ver el código PHP, como ocurre en las páginas ASP. PHP puede realizar cualquier tarea que un programa CGI, pero su fuerza radica en su compatibilidad con muchas bases de datos.

Otras de las tecnologías que ejecutan código en el servidor, es la de los Java Servlets. Estas son básicamente Java Applets que se ejecutan en el servidor en vez de en un navegador Web. Esta tecnología se esta volviendo popular como una alternativa a las aplicaciones CGI. La más grande diferencia entre los dos es que un Java Servlet es persistente. Esto significa que una vez que se ha iniciado, se mantiene en memoria y responde a varias peticiones. En contraste, un programa CGI desaparece una vez que responde a una petición. La persistencia hace a los Java Servlets más rápidos porque no gastan tiempo en cargar y cerrar un proceso.

Como se puede ver en las definiciones anteriores, ambas son soluciones del lado del servidor, al igual que ASP. Por lo mencionado antes, es conveniente usar PHP, desde un punto de vista personal. Pero, ¿cuándo saber si usar PHP o ASP? , esto va en relación al

tipo de servidor que se quiera usar. Como en la mayoría de las cosas, no se puede establecer una regla que funcione en todos los casos. Es necesario un análisis del problema para una solución particular a este.

Por otro lado, hablando de aplicaciones Web, tenemos las soluciones del lado del cliente (client side), como Java Applet. Un Applet es un programa diseñado para ser ejecutado dentro de otra aplicación. Los Applets no pueden ser ejecutados directamente desde el Sistema Operativo. Con la popularidad creciente de los Objetos Vinculados e incrustados (OLE), las Applets se están volviendo más comunes. Un Applet bien diseñado puede ser invocado de muchas aplicaciones diferentes. Los navegadores Web, generalmente están equipados con Maquinas Virtuales Java y pueden interpretar la Applets de los servidores Web. Porque las Applets son pequeñas en tamaño, compatibles con distintas plataformas, y altamente seguras (no pueden acceder al disco duro del usuario) son ideales para pequeñas aplicaciones de Internet accesibles desde un navegador Web.

Por parte de Microsoft, también existe una solución del lado del cliente, esta es el ActiveX. Un control ActiveX puede ser automáticamente bajado y ejecutado por un navegador Web. No es un lenguaje de programación, sino un conjunto de reglas de cómo una aplicación debe compartir información. Los programadores pueden desarrollar controles ActiveX en una variedad de lenguajes, incluyendo C, C++, Visual Basic y Java.

Esta tecnología es similar al Java Applet, pero a diferencia de esta, ActiveX tiene acceso total al sistema Operativo Windows. Esto le da mucho más capacidad que a las Java Applets, pero con esto viene el riesgo de que la aplicación pueda dañar software o datos de la maquina. Para controlar este riesgo, Microsoft desarrollo un sistema de registro para que los navegadores puedan identificar y autentificar un control ActiveX antes de bajarlo. Otra diferencia importante con Java Applet es que ActiveX solo puede ser ejecutado en ambiente Windows, mientras que las Applets son multiplataforma.

En lo que se refiere a la elección para una aplicación Web, si del tipo client side o del tipo Server side, para el objetivo de nuestro sistema esto se simplifica, ya que en cuanto a manejo de bases de datos se refiere, la opción es aplicaciones del tipo server side. Esto recae un poco en el hecho de que las bases de datos en una aplicación Web, deben estar en el servidor.

1.3 ¿Qué es ASP y que hace?

ASP: Siglas para Active Server Pages o Páginas con Servidor Activo. Es una tecnología desarrollada por Microsoft para realizar páginas Web dinámicas que usualmente utilizan código VBscript o Jscript. Un Script es una lista de comandos que pueden ser ejecutados sin la intervención del usuario.

Páginas Web dinámicas se refiera a páginas Web que no solo muestran información, ya sea tipo texto o multimedia, páginas que no son estáticas, donde la única interacción con el usuario es la de ir navegando a través de los vínculos a otras páginas Web. Una página dinámica puede variar el contenido, información que despliega, como resultado de alguna entrada del usuario o de otro medio.

Esta interacción usuario – página Web, hace que la página sea una aplicación, de hecho, lo es desde el momento que se inserta el Script (lista de comandos). Con esto se permite que el usuario vea solo la información que le interesa.

Este tipo de páginas puede acceder a bases de datos dentro del servidor, lo que es de suma importancia, ya que muchas de las aplicaciones actuales requieren de operaciones que se realizan en un lugar remoto, y aprovechando que Internet es fácil acceso para todos, se pueden realizar aplicaciones para ser ejecutadas desde un servidor donde el usuario que requiera éstas solo tendrá que tener acceso a una computadora con Internet y un navegador Web.

La forma en que estas páginas trabajan es la siguiente:

- El usuario solicita la página que desea ver. Esta solicitud se hace por medio del navegador, ingresando la dirección Web. Hasta aquí, funciona de la misma manera que una página HTML, solo que estas páginas tienen extensión ASP.

- La solicitud llega al servidor donde se encuentra alojada la página requerida. Es aquí donde se utiliza la conexión a Internet. También se pueden visualizar páginas en una LAN o incluso en la misma computadora, pero a diferencia de las páginas HTML, no basta con que se pueda acceder al archivo, tiene que estar activa la aplicación servidor para poder visualizarla.

- El servidor procesa la página ASP y devuelve código HTML. Esta es la parte importante de esta tecnología. El código es ejecutado en el servidor. Los archivos Java Applet y Flash, por mencionar algunos, se tienen que descargar en la computadora que hizo la solicitud, y se ejecutan en esta. Una razón para que estos archivos de aplicación sean descargados, es por que resultan ser muy pesados en comparación con otras aplicaciones para ser ejecutados en forma remota desde el servidor. Tal es el caso de los juegos en línea. Siendo de otra forma, nos conviene que las aplicaciones se ejecuten en servidor y solo devuelvan código HTML, ya que es más seguro de esta forma.

- El código HTML se envía a la computadora que solicito la página, y este se visualiza por medio del navegador.

Una página ASP puede tener texto, código HTML, código XML (todos estos se visualizan por medio del navegador) y los Scripts (estos se ejecutan en el servidor).

El software de servidor Web que se encarga de ejecutar el código ASP en una página Web es el IIS (Internet Information Server) en los sistemas Windows NT, 2000 y XP. Windows 9X cuenta con el PWS (Personal Web Server) una versión más pequeña pero funcional del IIS. Estas aplicaciones vienen con el CD de instalación o en los Service Pack, según sea el sistema. También se pueden bajar del sitio de Microsoft de forma gratuita.

Las páginas ASP también pueden ser ejecutadas en otras plataformas, en las cuales no se ejecuta IIS o PWS, dependiendo de la plataforma hay software para realizar esta tarea. En Linux, por ejemplo, se cuenta con Chilisoft ASP. InstantASP es otro software para ejecutar páginas ASP sin utilizar Windows. En mi opinión, a menos que sea necesario, esto no es conveniente, ya que es mejor utilizar las tecnologías nativas de cada sistema.

El código puede ser trabajado en cualquier editor HTML o de texto, este último esta incluido en todos los sistemas operativos. Existen herramientas para trabajar profesionalmente el ASP en modo visual, como lo son el Drumbeat 2000 y el Visual Interdev de Microsoft.

Como ayuda para los desarrolladores de aplicación, existen las API (Application Program Interface). Es un conjunto de rutinas, protocolos y herramientas para construir aplicaciones de Software. Una buena API hace más fácil desarrollar un programa al proveer todos los bloques para el desarrollo. Un programador pone estos bloques juntos.

La mayoría de los Sistemas Operativos, proveen una API para que los programadores puedan escribir aplicaciones consistentes con el ambiente del Sistema Operativo. Aunque las API son diseñadas para los programadores, acaban también sirviendo a los Usuarios. porque garantizan que todos los programas usando API's en común, tendrán interfaces similares. Esto hace más fácil para los usuarios aprender a usar nuevos programas.

Para el caso de desarrollo de aplicaciones Web, se tiene API's como ISAPI y NSAPI. ISAPI (Internet Server API) es una API para el Microsoft IIS. Además del IIS, muchos servidores Web de otras compañías diferentes a Microsoft soportan ISAPI. NSAPI es la API que se usa para desarrollar aplicaciones Web para servidores Web Netscape.

1.4 Autentificación para el ingreso de datos

Debido a la naturaleza de muchos sistemas, es necesario tener un método de seguridad para proteger la información, que esta no sea modificada o robada. La complejidad de esta dependerá del tipo de datos que se quieran proteger. En el caso de la tecnología ASP, cuyas páginas son ejecutadas en el IIS, se cuenta con configuraciones dentro del mismo programa. También se puede establecer "un certificado digital" ya sea con la adquisición de algún producto o con el desarrollo de este.

Dentro de la configuración del IIS para la autentificación tenemos tres básicamente:

1) Autentificación Básica: Es un estándar que la mayoría de navegadores soporta. Cuando se accede a un sitio y aparece una ventana con un dibujo de una llave con una etiqueta de planeta tierra, la cual pregunta nombre de usuario y contraseña, se esta usando la configuración básica.

2) Autentificación por cookie: Esta se aplica a nivel de Script, insertando código que funcione como autentificador.

3) NTLM / Windows Integrated Authentification: La Autentificación integrada en Windows es muy similar a la básica. La principal diferencia es que la información ingresada es encriptada y llevada de forma segura. Pero para completarse el navegador debe tener ciertas características. De hecho, esta función solo trabaja con Internet Explorer.

Por otro lado, tenemos los certificados digitales. Un certificado digital es un conjunto de datos, algo como credenciales cuando haces negocios u otras transacciones en la Web. Contiene nombre, número de serie, fecha de expiración, una copia de la llave pública del certificado y la firma digital de la persona que generó el certificado, para que se sepa que este certificado es válido. Es producida por una Autoridad de Certificación (CA), que es la persona en una red que administra llaves públicas para la encriptación de mensajes y firmas digitales.

En criptografía, una llave pública es un valor provisto por alguna autoridad designada (CA) como una llave de encriptación, que combinada con una llave privada derivada de la llave pública, puede ser usada efectivamente para encriptar mensajes y firmas digitales. El uso combinado de llave pública y privada es conocido como criptografía asimétrica. Algunos certificados digitales, cumplen con el estándar X.509.

Una firma digital (no confundir con un certificado digital) es una firma electrónica que puede ser usada para autentificar la identidad del que envía un mensaje y posiblemente asegurarse que el contenido original del mensaje o documento no ha sido cambiado. Una firma digital puede ser usada en cualquier mensaje, sea encriptado o no.

En los procesos de almacenamiento y transmisión de la información normalmente aparece el problema de la seguridad. En el almacenamiento, el peligro lo representa el robo del soporte del mensaje o simplemente el acceso no autorizado a esa información, mientras que en las transmisiones lo es la intervención del canal.

La protección de la información se lleva a cabo variando su forma. Se llama cifrado (o transformación criptográfica) a una transformación del texto original (llamado también texto inicial o texto claro) que lo convierte en el llamado texto cifrado o criptograma. Análogamente, se llama descifrado a la transformación que permite recuperar el texto original a partir del texto cifrado.

Capítulo 2: Programación en ASP

En este capítulo se describen algunos elementos de programación utilizados por la tecnología ASP. Desde "cómo empezar" hasta "escribir código para la conexión a bases de datos".

2.1 Software para ejecutar el código

Se ha mencionado qué es ASP, y cómo funciona. Del lado del usuario, se necesita un explorador Web, para poder leer la respuesta del servidor, el servidor ejecuta el código del Script y lo devuelve como HTML. El software del servidor que hace esto es el IIS (Internet Information Server), en el sistema operativo Windows 2000. El IIS también se incluyó en el sistema operativo Windows XP profesional. Sin necesidad de tener un servidor Web dedicado, se puede utilizar este software para una red de área local.

Si se quiere instalar este componente, tanto en el sistema Windows XP, como en Windows 2000, se tiene que abrir *Panel de control*, después en *agregar y quitar programas*, se selecciona el *IIS*. Se pueden modificar los detalles de este componente, al oprimir el botón *detalles*: Servicio World Wide Web, Servicio SMTP, servidor FTP, entre otros. Se instala el componente.

Para versiones anteriores de Windows, como Windows 98 y Millennium Editon (ME), se cuenta también con Software Servidor. El nombre de este software para estos sistemas es Personal Web Server. Es una versión más sencilla del IIS, es usada como servidor Web para redes corporativas y para visualizar páginas ASP antes de subirlas a un servidor.

Al ser instalado el PWS, aparece un ícono en la barra de tareas, el programa puede ser ejecutado desde este ícono, y acceder a la configuración del programa. En el caso del IIS, no se genera este ícono en barra de tareas. Para realizar configuraciones, se tiene que acceder a *panel de control*, *herramientas administrativas y Servicios de Internet Information Server*. Aunque no aparezca el ícono, el programa es ejecutado cada vez que se inicia Windows.

En ambos casos, se utiliza un directorio de inicio. En este directorio, se almacenan las páginas que van a ser compartidas, o las páginas que necesitan ser ejecutadas. El directorio predeterminado es C:\Inetpub\wwwroot, este puede ser cambiado si se quiere otro directorio o se tiene un sitio Web en algún otro. En el PWS, esto se hace en la ventana de *Opciones avanzadas*, seleccionando el directorio *Home*, y con el botón *Modificar propiedades*. En IIS, se selecciona el sitio Web al que se quiere modificar el directorio de inicio, se presiona el botón *Propiedades*, y en la pestaña directorio de inicio se modifica este valor.

Se puede agregar un directorio virtual al sitio Web. Esto es, un directorio en cualquier ubicación del disco duro, pero al que se le asigna un alias. Este alias, es el nombre con el que se ingresará desde el explorador. Se puede hacer desde el IIS o PWS, pero la forma

más sencilla es con el botón derecho del mouse en el directorio que se desea agregar como virtual, en la opción *Compartir* y en la pestaña *Uso compartido del Web* se modifica esta opción.

Una de las opciones que podemos modificar en estos dos programas es la de existencia de un documento predeterminado. Si se solicita una página Web almacenada en el servidor, y en la dirección no aparece el nombre de documento, se abrirá un documento que se haya establecido con anterioridad. Puede ser una lista separada por comas, en donde si no se encuentra el primer documento predeterminado en el directorio, se busque el segundo y así sucesivamente hasta encontrar un documento en la lista. Si no existe documento predeterminado en el directorio, se mandará un mensaje diciendo que no se puede explorar el directorio, a menos que se haya habilitado esta opción. Al habilitar la exploración de directorio, permite seleccionar el archivo a mostrarse de una lista de archivos existentes en el directorio del servidor.

Para el PWS, ambas opciones se encuentran en la ventana de opciones avanzadas, con el texto *habilitar documento predeterminado* y *permitir exploración de directorio*. En el IIS, para hacer que un documento Web sea el predeterminado, tenemos que seleccionar el sitio Web en el programa, abrir *Propiedades* y en la pestaña de *Documentos* anexar los documentos predeterminados. Al seleccionar una carpeta, en la ventana de propiedades de esta, se puede activar la opción de exploración de directorio.

Además de las modificaciones mencionadas, los directorios pueden tener atributos, dependiendo la tarea que realizan los archivos que están contenidos dentro de la carpeta. Estos atributos pueden ser de lectura, ejecución y de archivo de comandos (Script). El atributo de lectura permite acceder a las páginas Web y debe estar activado para directorios que contengan información que va a ser desplegada. El atributo ejecución permite a los usuarios correr aplicaciones en el directorio. El atributo archivo de comandos, permite correr Scripts en un directorio y debe estar activada para un directorio que contenga páginas ASP. Un atributo puede hacer que se generen mensajes de error si un usuario solicita una acción que no esta permitida por el atributo.

El PWS tiene soporte a ASP 2.0, mientras que IIS v5.0 tiene soporte a ASP 3.0, la cual es la última versión de ASP, sin considerar a ASP.NET. Windows 2003 Server incluye el software IIS 6.0. Estos programas son de fácil manejo, un usuario con nociones de diseño Web, aprende a utilizarlo en el momento de estar operándolo.

El software mencionado arriba nos permite responder a solicitudes de un usuario. Por otra parte, el usuario hace estas solicitudes por medio de un navegador Web, que seria lo necesario, refiriéndonos a software, que se necesita para que la página sea desplegada. Para el desarrollo de páginas ASP, bastará con el escritor de texto simple para escribir el código.

2.2 Elementos básicos de la programación en ASP.

Impresión de texto y declaración de variables

Se mencionó que la Tecnología ASP es básicamente la inserción de un código Script, que es ejecutado en el servidor. En esta parte del capítulo se tratarán los elementos básicos de programación en el Script, el Script que se va a emplear es el Visual Basic Script. Se podrá notar que en los primeros ejemplos, el código no se ejecuta en el servidor, las páginas generadas no son ASP. Esto es solo para introducir los elementos básicos del Script, sin utilizar ASP, solo la inserción de Script. En ejemplos posteriores se insertarán en las páginas ASP.

La inserción del Script se hace dentro del código HTML por medio de una etiqueta inicial y una etiqueta final.

```
<SCRIPT LANGUAGE="VBScript">
</SCRIPT>
```

Lo que se encuentre dentro de estas etiquetas, será código para ejecución. Se agregan etiquetas de comentario (<!--,-->).

En VBscript se utiliza la instrucción *Document.Write* para desplegar información. Al igual que en otras variantes del Basic, se tiene que poner el texto entre comillas. Con la instrucción *Dim*, se declara una variable, se coloca la instrucción, seguida del nombre de variable.

La instrucción OPTION EXPLICIT obliga a que se declare una variable antes de ser usada. Esto es conveniente para evitar errores de programación. Cuando un valor se va a quedar estático durante la ejecución del programa se declara como constante. Se utiliza la instrucción *Const*, seguida del nombre de variable y el valor que se desea asignar. Una constante puede ser cualquier valor alfanumérico, que no sea resultado de un cálculo o de otra función. El siguiente ejemplo muestra lo mencionado.

```
<HTML>
<HEAD>
<TITLE>PRECIO DEL ARTICULO 1</TITLE>
</HEAD>
<BODY>
En esta parte se puede agregar ia descripción del producto<BR>
<SCRIPT LANGUAGE="VBScript">
<!--
OPTION EXPLICIT
Dim Precio01
Const Iva=.15
Precio01=200
Document.Write "Precio del Articulo 1: $",Precio01,"<BR>"
Document.Write "I.V.A.: $",Precio01*Iva,"<BR>"
Document.Write "Total: $",Precio01+Precio01*Iva
-->
</SCRIPT>
</BODY>
</HTML>
```

Ejemplo 01: Declaración de variables e impresión de Datos

El Ejemplo 01 muestra como salida, el precio del producto, el IVA. y la suma, con texto para el usuario. Se pudo haber almacenado la suma en una variable y luego haber impreso este valor. También hay que notar que el Script se encuentra solo entre las etiquetas Script, y que pueden agregarse etiquetas HTML de formato de texto en el Script, siempre y cuando estén entre comillas, por consiguiente, en una cadena de texto.

Arreglos estáticos y dinámicos

La creación de un arreglo se realiza por medio de la instrucción para declarar una variable, agregando el nombre de variable y la dimensión del arreglo entre paréntesis, esta dimensión debe ser un número entero. Este arreglo funciona como un arreglo estático, quiere decir que trabajará con la dimensión que se introdujo en el código mientras se ejecute el programa. Se ingresaran valores al arreglo mediante el nombre de variable y el número de elemento del arreglo entre paréntesis. Estos valores asignados con el signo de igual, pueden ser tipo carácter o numérico, incluso mezclados. El primer elemento del arreglo será el elemento con índice 0.

Una forma rápida de ingresar elementos a un arreglo es con la instrucción Array. Se tiene que declarar el nombre del arreglo, sin incluir en los paréntesis la dimensión. Después de esta instrucción, se escribe el nombre del arreglo, un signo igual para la asignación, la instrucción *Array* y entre paréntesis los elementos del arreglo separados por coma. Esto es útil cuando queremos llenar un arreglo desde el principio utilizando una sola instrucción.

Se puede crear un arreglo dinámico. En este tipo de arreglos la dimensión puede ser cambiada cuando el programa de ejecución, dependiendo de las necesidades de este. Se empieza por declarar un arreglo sin especificar la dimensión, después se utiliza la instrucción *Redim* seguida del nombre del arreglo y la dimensión. Puede volver a ser asignada la dimensión con la misma instrucción, en alguna otra parte del programa, teniendo en consideración que se necesita la instrucción *Redim Preserve* cuando se quiera conserva los elementos ya ingresados del arreglo.

La instrucción *Ubound,* sin argumento, devuelve el índice del elemento con mayor índice del arreglo, por lo que se puede utiliza para saber el número de elementos en el arreglo. Se utiliza la instrucción, seguida del nombre del arreglo entre paréntesis. Esto se puede utilizar para redimensionar el arreglo a su número de elementos más uno a la hora de la captura de otro elemento, en conjunto con la instrucción R*edim*.

Instrucciones para el control de flujo

Al igual que en todos los lenguajes, existen instrucciones para el control de flujo. Para VBScript se mencionarán dos, la instrucción *If* y la instrucción *Select Case*. La instrucción *If* maneja una sintaxis, compuesta por la instrucción, seguida de la condición y la instrucción *Then*. Básicamente se evalúa la condición, de resultar verdadera, ejecuta las instrucciones encerradas entre *Then* y el *Endif*. De resultar falsa, se ejecutará la línea posterior a *Endif*. Si se requiere que se ejecute alguna acción en caso de que la condición resulte falsa, se puede agregar *Else* dentro del bloque *IF*, después del *Then*, esta instrucción ejecutará el bloque de líneas que se encuentre entre *Else* y el *Endif*. Esta instrucción puede anidarse, poner instrucciones *If* dentro de una instrucción *If*. Resulta difícil verificar este tipo de código, por lo que es mas propicio en algunas ocasiones agregar la instrucción *Elseif* dentro del bloque *If*. Con esta instrucción, se ejecuta una sección de código, dependiendo de los resultados de verificación de condiciones múltiples.

La instrucción *Select Case* ejecuta una sección de código específica dependiendo del valor que tome una variable. Es parecido a lo que hace la instrucción *Elseif*, con la diferencia que en esta instrucción solo se evalúa el valor de una variable mientras que en el caso *Elseif* son condiciones. Su sintaxis es la instrucción *Select Case* seguida del nombre de variable, la instrucción *Case* y el valor de cada variable para ejecutar ese código. Se termina con la Instrucción *Endcase* para evaluar una condición se pueden utilizar operadores lógicos (And, Or, Not, Xor) y de comparación.(<,>,=,<>).

Ciclos

Las instrucciones que nos permiten hacer ciclos en VBScript son la instrucción *Do* y la instrucción *For*. La instrucción *For* es un ciclo con contador, mientras que la Instrucción *Do* es un ciclo con condición. La sintaxis de la instrucción *For* es la instrucción, la variable contador con un valor inicial, después se escribe el valor máximo al que tiene que llegar el contador. Se puede agregar la instrucción *Step*, la cual indica el incremento al contador. Se termina con la instrucción *Next*. Todo lo que este dentro de *For* y *Next* constituirá las instrucciones del ciclo. La instrucción *For Each Next* se utiliza para acceder a elementos almacenados en una colección de datos, como un arreglo o una colección de registros.

La sintaxis de la instrucción *Do* se forma de la instrucción *Do While* y la condición. Se cierra con la instrucción *Loop*. Las líneas de código que se encuentren entre estas dos instrucciones son la que se ejecutarán mientras la condición sea verdadera. La instrucción *EXIT Do* hará que se ejecute el código fuera del ciclo. De manera similar, la instrucción *EXIT For* termina un ciclo *For*.

Un ejemplo de código para un programa con un ciclo, decisión y arreglo dinámico es el siguiente.

```
<HEAD>
<TITLE></TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE=VBScript>
<!--
OPTION EXPLICIT
Dim Personas()
Dim busqueda,Nombre_a_buscar,Bandera
Nombre_a_buscar="Gina"
Redim Personas(2)
Personas(0)="Alejandro" : Personas(1)="Guillermo" : Personas(2)="Laura"
Redim Preserve Personas(4)
Personas(3)="Gina" : Personas(4)="Daniel"
Redim Preserve Personas(Ubound(Personas)+1)
Personas(5)="Leonora"
```

```
For each busqueda in Personas
  If busqueda=Nombre_a_buscar Then
    Document.Write "El elemento si esta en la lista"
    Bandera=True
    Exit For
  End if
Next
        If Bandera=False Then Document.Write "El elemento no esta en la lista"


        -->
        </SCRIPT>
        </BODY>
        </HTML>
```

Ejemplo 02: Arreglo dinámico y ciclo.

En el ejemplo anterior, se almacenan elementos a una lista, se varía la dimensión del arreglo en la ejecución del programa. Después, mediante un ciclo, se busca la coincidencia de una cadena, con el elemento de un arreglo. Si esta cadena es encontrada, el programa entra a un segmento de código en el que imprime que la cadena de búsqueda existe en nuestro arreglo, asigna el valor de verdadero a una variable llamada Bandera, y sale del ciclo. Podría modificarse fácilmente el código para que encontrara cuantos elementos coinciden con la cadena dentro del arreglo. Si al final del ciclo, no se encuentra ninguna coincidencia de elementos, la variable bandera no tiene asignado el valor de verdadero, entonces se imprime el valor de que no se encontró el elemento.

2.3 Otros elementos de programación.

En el tema anterior se mencionaron algunos elementos importantes de VBScript. Estos elementos de programación no se utilizaron como código para una página ASP. Al ingresar código Script en una página HTML, este puede ser visualizado, y se ejecuta en la computadora que realiza la solicitud. El código ASP, se ejecuta en el servidor, y se visualiza como los elementos HTML resultantes.

El nombre de este tema en el capítulo, hace referencia a que se dará tratado a elementos propios de aplicaciones Web ejecutadas en Servidor. Para empezar esta mención, se tendrían los objetos de ASP. Los cinco principales objetos son:

- *Respose*: Envía y controla la información del servidor Web a un usuario.
- *Request*: Extrae y controla datos enviados de un usuario al servidor Web.
- *Server*: crea objetos y subministra acceso a métodos y propiedades en el servidor Web.
- *Session*: Almacena información de sesión para usuarios individuales conforme navegan en un sitio Web.
- *Application*: Almacena y comparte información para su uso durante una aplicación activa.

Por la gran cantidad de instrucciones, solo se retoman los elementos a utilizar en la aplicación resultante de este trabajo.

El código ASP es insertado dentro del código HTML mediante los delimitadores <%, %>. Estos indican al servidor la parte a procesar antes de enviar el código HTML. Estos pueden ser insertados en la parte Body del código HTML.

Las páginas deben guardarse en un servidor Web que soporte esta tecnología. Son archivos de texto con la extensión .asp. En el VBScript se utilizó la instrucción *Document.Write* para escribir algún mensaje. En ASP se utiliza el objeto Response y la instrucción *Write* para realizar esta tarea (*Response.Write("Texto")*). Si se quiere escribir mas de un elemento con la instrucción *Write*, se debe incluir cada elemento dentro del paréntesis separado por el carácter &. Los comentarios se pueden insertar con el carácter '.

Formularios

Hasta ahora, no se ha dicho cómo una página Web puede recopilar valores de un usuario. Esto se hace por medio de Formularios. Este código es HTML, pero los resultados del formulario, si así se indica, son enviados a una página ASP en la cual los datos pueden ser tratados. Las formas de obtener información dentro de un formulario puede ser mediante:

cuadro de contraseña, lista desplegable, cuadro de texto, área de texto, cuadro de verificación y botón de opción. Estos datos son enviados a una página ASP por medio de un Botón de envío. También existe un botón de restablecimiento que restaura el formulario a su forma inicial. Existen dos métodos de envío. El método Get y el método Post. El método Get envía los datos adjuntos al URL de la página, en la dirección de la página, con un signo de interrogación y el nombre de los datos seguidos de un igual y su valor, cada uno de esos separado por un carácter &. El método Post no adjunta esta información en el URL.

La sintaxis de la etiqueta HTML para iniciar formulario consta de la palabra *Form*, seguido de *Action="archivo.asp"*. Archivo.asp es el nombre del archivo que usará la información del formulario. *Method="Get"* selecciona el método de envío de datos. Puede ser Get o Post. Dentro de estas etiquetas se selecciona el método de entrada por medio de la etiqueta *Input Type* y el nombre del método de entrada, seguido de sus atributos y/o elementos. Los atributos principales son:

- *Name:* Nombre del elemento. Se utiliza para identificar los datos.
- *Value:* Permite especificar un valor para un elemento. Si el elemento es un botón, será el texto en el botón.
- *Maxlength:* Número máximo de caracteres.
- *Checked:* Selecciona un elemento de forma predeterminada.

La siguiente tabla muestra ejemplos de código para los métodos de entrada nombrados.

Cuadro de contraseña	*<INPUT TYPE="password" name="Clave" maxlength="20">*
Lista desplegable	*<SELECT NAME="Nacionalidad"> <OPTION VALUE="Mexicana">Mex</OPTION> <OPTION VALUE="Americano">Amer</OPTION> <OPTION VALUE="Canadiense">Can</OPTION>*
Cuadro de texto	*<INPUT TYPE="text" NAME="Nombre">*
Área de texto	*<TEXTAREA NAME="Opiniones"></TEXTAREA>*
Cuadro de verificación	*<INPUT TYPE="checkbox" NAME="info" VALUE="Autos">*
Botón de opción	*Femenino<INPUT TYPE="radio" NAME="genero"*

	VALUE="Femenino" Checked> Masculino<INPUT TYPE="radio" NAME="genero" VALUE="Masculino" Checked>
Botón de envio	<INPUT TYPE="submit" NAME="envoi" Value="Enviar">
Botón de reinicio	<INPUT TYPE="RESET" value"Restablecer"

Ejemplo 03: Etiquetas para métodos de entrada.

La instrucción *QueryString* del objeto *Request* extrae información anexa en el URL. Su sintaxis es *Request.QueryString("Nombre_Dato")*, donde Nombre_Dato es el nombre de variable donde se dejo almacenado el valor desde el formulario. Esto se utilizará para el siguiente ejemplo, en donde se presentará un formulario, que al ser enviado, se utilizará esta información para ser desplegada en una página Web.

```
<HTML>
<HEAD>
<TITLE>Formulario</TITLE>
</HEAD>
<BODY>
<H1>Datos Personales</H1>
<FORM ACTION="http://fiona/tesis/Datos.asp" METHOD="GET">
Nombre: <INPUT TYPE="text" NAME="Nombre"><BR><BR>
Sexo:          Masculino<INPUT TYPE="radio" Name="Sexo" Value="Masculino"
CHECKED>
               Femenino<INPUT TYPE="radio" Name="Sexo"
Value="Femenino"><BR><BR>
Nacionalidad: <SELECT NAME="Nacionalidad">
               <OPTION VALUE="Mexicana">Mexicana</OPTION>
               <OPTION VALUE="Americana">Americana</OPTION>
               <OPTION VALUE="Canadiense">Canadiense</OPTION>
             </SELECT><BR><BR>
Pasatiempos: Viajar<INPUT TYPE="checkbox" NAME="Hobbies"
VALUE="Viajar">
          Cine<INPUT TYPE="checkbox" NAME="Hobbies" VALUE="Cine">
          Deportes<INPUT TYPE="checkbox" NAME="Hobbies"
VALUE="Deportes"><BR><BR>
<INPUT TYPE="submit" NAME="envio" Value="Enviar">
</FORM>
</BODY>
</HTML>
```

Ejemplo 04: Archivo HTML de formulario

Si el elemento almacena más de un valor, como es el caso de un cuadro de verificación, al momento de obtener los valores de este, se obtendrán como una sola cadena con los elementos separados por comas. Si se desea obtener el valor de un elemento específico del cuadro de verificación, se tendrá que anexar el índice adelante del nombre del cuadro de verificación. Se debe tener cuidado en este índice, ya que no es fabricado con el numero de elementos en el cuadro de verificación, sino con el número de elementos seleccionados. Si se quiere saber el número de elementos seleccionados, se agrega la instrucción Count adelante del nombre del cuadro de verificación. Incluso estos datos pueden ser acomodados, si se necesitará, en un arreglo que nombre todos los valores y que asigne falso o verdadero según la verificación.

```
<HTML>
<HEAD>
<TITLE>Datos del formulario</TITLE>
</HEAD>
<BODY>
<%
'OPTION EXPLICIT
Dim i
Response.Write("Nombre: "&Request.QueryString("Nombre")&"<BR>")
Response.Write("Sexo: "&Request.QueryString("Sexo")&"<BR>")
Response.Write("Nacionalidad: "&Request.QueryString("Nacionalidad")&"<BR>")
Response.Write("Hobbies:<BR>")
If Request.QueryString("Hobbies").Count=0 then
        Response.Write("Ningun Hobbie listado<BR>")
Else
        For i=1 to Request.QueryString("Hobbies").Count
                Response.Write(Request.QueryString("Hobbies")(i)&"<BR>")
        Next
End If

%>
</BODY>
</HTML>
```

Ejemplo 05: Archivo datos.asp que utiliza el formulario ejemplo anterior

El ejemplo 5 recupera información del formulario, solo imprime los datos recuperados en la página. Estos datos pueden usarse en algún proceso o para llenar un a base de datos.

Lo anterior, mostró como recuperar elementos capturados en un formulario, cuando se usa el método Get. Para un formulario, también se puede utilizar el método Post. Para recuperar datos ingresados en un formulario por un método Post, se sigue utilizando el objeto Request, en vez de utilizar la instrucción *QueryString,* se utiliza la instrucción *Form*. La forma en que funciona el código es la misma, se puede utilizar también la instrucción *Count*. Del ejemplo 04, si se cambia el método *Get* por *Post*, solo se tendría que cambiar en el ejemplo 5 la instrucción *QueryString* por *Form* y la página seguiría funcionando de la misma forma.

Búfer

Por medio del objeto response también se puede controlar el búfer. El búfer es una sección de memoria dentro del servidor ASP donde una página puede ser almacenada temporalmente, con el fin de una respuesta más rápida para el usuario. Sin este búfer, se tendría que procesar la página ASP en el servidor cada vez que se solicita. La instrucción para activar el búfer es Response.Buffer="true" y debe de colocarse antes de cualquier etiqueta HTML, al principio del documento. Estas y otras instrucciones del manejo de buffer no pueden ser ejecutadas en PWS, solo en IIS. Por default, el búfer se encuentra activado.

Como el servidor realiza el procesamiento de las instrucciones ASP, no despliega información hasta haber terminado de procesar el bloque. Si es un proceso muy largo, la página permanecerá en blanco hasta que el proceso se lleve a cabo sin mostrar información. A veces es conveniente, que muestre información de manera parcial, y siga con otras partes del proceso. La instrucción Response.Flush obliga a que se muestre el contenido del búfer procesado hasta ese momento y vacía el búfer. Por medio de esta instrucción se controla el despliegue de información hacia el usuario. La instrucción Response.Clear limpia el contenido del búfer desde la última instrucción Flush o desde el principio del búfer. Esto es útil cuando se presenta un error. El proceso ejecutado y almacenado en el búfer hasta ese momento no se despliega. La instrucción Response. End detiene el procesamiento del búfer

y envía la información procesada hasta el momento. Ninguna instrucción posterior a esta es ejecutada.

Expiración y variables de servidor

Uno de los manejos que se pueden hacer por medio del objeto Response es el de tiempo de expiración de una página. Los exploradores Web por lo general utilizan una memoria caché para almacenar las páginas vistas por un usuario. Estas páginas almacenadas, son llamadas cuando el usuario vuelve a solicitar la misma página Web sin el botón actualizar. Esto genera problemas cuando una página Web tiene actualizaciones muy frecuentes en cortos periodos de tiempo, ya que se podría estar omitiendo información actualizada. Con la instrucción Response.Expires = n se controla el número de minutos en la que expira una página Web, forzando al explorador a volver a cargar la página. Esta instrucción, al igual que las de control del búfer, debe ser colocada al principio del documento.

Se pueden mostrar las variables del servidor por medio de la instrucción ServerVariables del objeto Response. Se puede crear una página que por medio de un ciclo *For Each* muestre las variables disponibles por el servidor. Entre las más importantes se encuentran:

- Server_Name: Nombre del servidor
- Local_Addr: Dirección IP del servidor
- Server_Software: Software del servidor
- Remote_Addr: Dirección del cliente
- http_User_Agent: Tipo de explorador Web del Cliente

Cookies

Una Cookie es un archivo de texto de tamaño pequeño que se almacena en la computadora del usuario que solicita la página. Consta de un nombre y valor(es) que se almacenan dentro de ella. Se puede especificar una fecha de vencimiento. Esto es conveniente, ya que por lo general una Cookie sin fecha de vencimiento es borrada cuando se cierra el explorador. También se puede establecer un dominio y ruta de estas para que sean usadas solo por el servidor que las depositó.

Su sintaxis es Response.Cookies("Nombre")("Subclave")="Valor". No es conveniente que una misma Cookie tenga muchas subclaves o valores. En vez de esto, se puede generar una solo clave, para recuperar los elementos en una base de datos. Esta instrucción también debe ser colocada antes de las etiquetas HTML.

Para recuperar una Cookie se utiliza el objeto Request y la instrucción Cookie, utilizando el nombre de la Cookie. Si la Cookie contiene subclaves, se deberá anexar el nombre de las subclaves a recuperar. Si no existe la Cookie, regresara valores vacíos al hacer la petición. Para no confundir una Cookie con valores vacíos con una Cookie inexistente, se tiene que asegurar que en la creación de esta no se generen valores vacíos.

El siguiente ejemplo muestra como llenar una Cookie en base a la captura de datos en un formulario. La página de inicio extrae nombre y apellidos de una Cookie. En la página de captura hay un formulario con dos entradas de textos que manda la información a la página cookie.asp, la cual es la que genera la Cookie.

```
<HTML>
<HEAD>
<TITLE>Inicio</TITLE>
</HEAD>
<BODY>
<H2>Bienvenido
<%
If (Request.Cookies("Nombre")("Nombre"))<>"" Then
        Response.Write(Request.Cookies("Nombre")("Nombre")&" ")
        Response.Write(Request.Cookies("Nombre")("Apellidos"))
Else
        Response.Write("<BR>Debe llenar Formulario")

End if %>
</H2>
Usuario nuevo?No es su nombre?
<A HREF="Http://Fiona/tesis/Captura.htm">Captura</A>
</BODY>
</HTML>
```

Ejemplo 06: archivo de inicio, inicio.asp

```
<HTML>
<HEAD>
<TITLE>Captura Datos</TITLE>
</HEAD>
<BODY>
<H1>Captura de datos</H1><BR><BR>
<FORM ACTION="http://fiona/tesis/cookies.asp" METHOD="Post">
Nombre: <INPUT TYPE="text" NAME="Nombre"><BR><BR>
Apellidos: <INPUT TYPE="text" NAME="Apellidos"><BR><BR>
<INPUT TYPE="submit" NAME="envio" Value="Enviar">
</FORM>
</BODY>
</HTML>
```

Ejemplo 07: Archivo de formulario, captura.htm

```
<%
If Request.Form("Nombre")<>"" and Request.Form("Apellidos") <>"" Then
        Response.Cookies("Nombre")("Nombre")=Request.Form("Nombre")
        Response.Cookies("Nombre")("Apellidos")=Request.Form("Apellidos")
        Response.Cookies("Nombre").Expires=#Oct 23,2004#

Else
        Response.Write("Debe de completar los Campos")
        Response.End
End if
%>
<HTML>
<HEAD>
<TITLE>Enviando Datos</TITLE>
</HEAD>
<BODY>
<H2>Datos Recibidos</H2>
<A HREF="Http://Fiona/tesis/Inicio.ASP">Inicio</A>

</BODY>
</HTML>
```

Ejemplo 08: Archivo que genera cookie,cookie.htm

Bases de Datos

Una parte importante para la aplicación que se va a desarrollar es el acceso a bases de datos. ASP nos permite consultar y modificar una base de datos desde un explorador Web. Las bases de datos que se manejan con más frecuencia para ASP son las bases de datos creadas por el SMBD (Sistema Manejador de Base de Datos) Microsoft Access y Microsoft SQL Server. Lo primero que se necesita es establecer una conexión a la base de datos. Se debe crear un DSN o Nombre de fuente de datos para indicar a la página ASP el tipo de base de datos que se está utilizando y la ubicación de esta en el servidor.

Para agregar un DSN en el sistema operativo Windows 2000 y XP, se tiene que ejecutar el icono ODBC dentro de las herramientas administrativas. En Windows 9X, ODBC se encuentra en el panel de control. Aquí se visualizan tres pestañas DSN dependiendo del tipo de DSN que se quiera agregar.

DSN de sistema: La información se almacena en un registro del servidor Web. Cualquier usuario con acceso al servidor será capaz de usar el sistema DSN para acceder a la base de datos.

DSN de usuario: La información se almacena en un registro del servidor Web, pero solo una cuanta específica de usuario puede usar el DSN.

DSN de Archivo: La información se almacena en un archivo de texto en el servidor Web. Cualquier usuario con acceso al servidor será capaz de usar el sistema DSN para acceder a la base de datos.

Con el botón Agregar, aparece otra ventana donde tenemos que especificar el nombre de la conexión y la ubicación y nombre de la base da datos.

Para configurar una conexión a una base de datos, una instancia del objeto *Connection* debe ser creada y asignarle un nombre. Después se pueden usar sobre esta las propiedades y

métodos del objeto *Connection*. Timeout asigna el número de segundos para intentar abrir la base antes de generar un error. *Open* abre la base indicando el DSN. *Close* cierra la base. Para extraer datos de una base por medio de ASP, contamos con dos métodos, el método Execute, que es recomendado para bases con pocos elementos, y usando el objeto Recordset para bases con un gran número de registros y que son actualizadas frecuentemente.

Primero se establece la colección de registros, sobre la conexión ya realizada y la instrucción *Execute*, en la cual se ingresa la instrucción SQL *Select* que incluye el nombre de la tabla en la base de datos. Al iniciar, nuestra base se encuentra en el registro número 1, se puede llamar a una parte del registro por medio del nombre de la colección de registros y un índice, el cual especifica el nombre del campo, empezando con 0 el primero y así sucesivamente. Para moverse al siguiente registro se utiliza la instrucción *MoveNext*. Para hacer esto por medio de un ciclo, se necesita saber el número de campos, esto se puede hacer con la instrucción *Fields*. El siguiente ejemplo imprime el número de campos en la tabla, el campo 1 del registro 1 y el campo 2 del registro 3.

```
<HTML>
<HEAD>
<TITLE>Base de Datos</TITLE>
</HEAD><BODY><%
SET connectionToDataBase=Server.CreateObject("ADODB.Connection")
connectionToDatabase.Connectiontimeout=60
connectionToDatabase.Open "DSN=telefonos"
Set recordcollection=ConnectiontoDatabase.execute("Select * from agenda")

Response.Write(recordcollection.fields.count&<BR>)
Response.Write(recordcollection(1)&<BR>)
recordcollection.movenext
recordcollection.movenext
Response.Write(recordcollection(2)&<BR>)

connectionToDatabase.close
Set connectiontodatabase=Nothing
%>
</BODY></HTML>
```

Ejemplo 09: Abrir y consultar una base de datos, método Execute.

Para editar un registro, usando el objeto RecordSet, se establece la conexión a la base de datos, igual que en el ejemplo 09. El conjunto de registros antes establecido en *RecordCollection*, se asigna a RecordSet, también delimitando los registros por medio de la instrucción *Select* y la instrucción *Where* seguido de una condición. También se imprime el campo de un registro utilizando un subíndice, solo que este, a diferencia del método Execute, no es un número, sino el nombre del campo del cual se quiere recuperar la información. Para modificar un elemento, se tiene que estar en el registro deseado a modificar, se coloca como índice el nombre de campo a modificar, y se asigna el valor por medio de un signo igual y el nuevo valor deseado. Se agrega una instrucción *Update* para que los cambios se realicen. El ejemplo 10, es código para la modificación de registro.

```
<HTML>
<HEAD>
<TITLE>Base de Datos</TITLE>
</HEAD>
<BODY>
<%
SET connectionToDataBase=Server.CreateObject("ADODB.Connection")
connectionToDatabase.Connectiontimeout=60
connectionToDatabase.Open "DSN=telefonos"

Set recordset=Server.CreateObject("ADODB.recordset")
recordset.Open "select * From agenda  where
telefono='58732863'", connectiontodatabase,1,2

response.write(recordset("Nombre"))
recordset("Nombre")="Alejandro"
recordset.Update

connectionToDatabase.close
Set connectiontodatabase=Nothing
%>
</BODY>
</HTML>
```

Ejemplo 10: modificación de un registro.

La forma mediante la cual se agrega y elimina un registro, es por medio de las instrucciones *SQL INSERT* y *DELETE* Respectivamente. Estas son ejecutadas por medio del método EXECUTE. Estas instrucciones pueden almacenarse en una variable para hacer el programa más fácil de leer. La instrucción *INSERT* lleva las dos cláusulas. *INTO* especifica el nombre de la tabla y los campos afectados, *VALUES* lleva los valores encerrados en apostrofes y en el orden que se especificaron los campos. La instrucción *DELETE* también lleva dos cláusulas. *FROM* para la tabla que se usará y *WHERE* que indica los valores que debe tener el campo de los registros a borrar. El código se hace un poco confuso al agregar las *comillas* y *ampersens* necesarios para la construcción de variables de cadena en el Script. El ejemplo 11 agrega un registro y lo borra.

```
<%
nombre="Alejandro"
Direccion="Venus #47"
Telefono="54678900"

SET connectionToDataBase=Server.CreateObject("ADODB.Connection")
connectionToDatabase.Connectiontimeout=60
connectionToDatabase.Open "DSN=telefonos"

instruccion="INSERT INTO Agenda(Nombre,Direccion,Telefono)"&_
"Values('"&Nombre&"','"&Direccion&"','"&Telefono&"')"
Set recordset=connectiontodatabase.execute(instruccion)

instruccion2="DELETE FROM Agenda WHERE nombre='"&nombre&"'"
Set recordset=connectiontodatabase.execute(instruccion2)

connectionToDatabase.close
Set connectiontodatabase=Nothing
%>
```

Ejemplo 11: Agregar y borrar registros.

Las instrucciones utilizadas para codificar una página ASP, son relativamente sencillas, más para alguien familiarizado con el lenguaje Visual Basic. Lo mencionado en este capítulo, pretende ser una base de para el desarrollo de la aplicación que se generará en este trabajo, en cuanto a programación se refiere. A pesar de haber más objetos e instrucciones, solo se tratan los de interés para el trabajo.

Capítulo 3: Modelado del sistema de información escolar en línea

En este capítulo se determina la necesidad del sistema y sus requerimientos. También se describe cómo funcionan las aplicaciones, qué datos reciben, qué datos se producen, la manera en que lo hacen y las herramientas de desarrollo que se utilizarán.

3.1 Análisis de requerimientos

Planteamiento del problema.

Para un estudiante, es importante tener acceso a la información escolar (en particular a sus calificaciones). Esta información le ayuda a la toma de decisiones respecto a su horario o para saber su avance académico actual. Por lo general este tipo de información se le proporciona al alumno de manera impresa.

Sin embargo, el problema de la información impresa, es que se tendrían que generar grandes volúmenes de esta para entregarla a cada alumno. Además no es práctico un historial académico por período escolar, en la mayoría de los casos se requiere información de este tipo en un período específico.

Las listas de calificaciones colocadas en las instalaciones no son la mejor solución. Por lo general, esta información se coloca en lugares clave, para ser vista por varias personas, pero muchas veces es destruida por estudiantes o gente externa que no tiene conciencia de la importancia de estas.

Otro problema con la información impresa es que no puede ser consultada en cualquier hora y en cualquier lugar.

Por último, se tiene los módulos de consulta escolar. Su problema es que en ciertas fechas del periodo escolar se saturan, debido al número de módulos que existen y el número de personas que quieren consultar esta información. Sería poco práctico tener un número grande de módulos, ya que esto es costoso, y la mayoría del tiempo estarían en desuso. Un problema extra de estos módulos, es que están sujetos al horario de la escuela, además de que hay que desplazarse a estos.

Requerimientos

De usuario obligatorios (el sistema debe hacer)

- El alumno podrá consultar sus calificaciones por medio de un navegador de Internet
- El profesor podrá ingresar las calificaciones a la base de datos existente.
- Se verificará, en el ingreso de calificación, que esta sea hecha por un profesor.

De usuario deseables (el sistema podría tener)

- Se podrán visualizar instrucciones sencillas para la consulta e ingreso de datos, una ayuda en línea.
- Podrá haber un despliegue de datos generales del alumno.

De sistema Funcionales (especificación para alguna función)

- El despliegue de calificaciones será individual, usando un arreglo en donde los renglones son las asignaturas y las columnas datos de estas. El password será la fecha de nacimiento y como logín la matricula.
- Para el ingreso de calificaciones se requerirá de un login y un password que no sea de fácil acceso a otras personas diferentes al usuario, y que sea protegido por medio de la aplicación.

De sistema no funcionales (especificación técnica para alguna función)

- Se podrá tener acceso a una base de datos por medio de un navegador de Internet
- Se generará información estadística en el despliegue, como el promedio y porcentaje de créditos.
- Se generará información estadística como el porcentaje de aprobación y promedio del grupo
- El diseño de la aplicación utilizará tecnología ASP.

La necesidad de la consulta de calificaciones es la que motiva la creación de este sistema. Aunque esta consulta exista, no se puede realizar a cualquier hora, y necesita el traslado a la institución. Básicamente, se podría consultar la calificación desde Internet. Los profesores podrán ingresar calificaciones al sistema desde cualquier computadora con acceso a Internet, para que después los alumnos consulten esta. Para la captura se necesitará la información existente acerca de los alumnos integrados al grupo y de los profesores integrados al grupo.

Con esto se mejorará la atención al estudiante, dándole herramientas para la creación de su siguiente grupo a mejor tiempo, y ahorrando tiempo y esfuerzo para el desarrollo de otras actividades.

Definición de requerimientos de usuario

Los servicios que ofrecerá el sistema básicamente son:

- Captura de calificaciones
- Consulta de calificaciones
- Consulta de promedio grupal.

Como necesidades de estos, se generan otros servicios, como son: correcciones, petición para lista de calificación, autentificación, despliegue de información estadística para calificaciones.

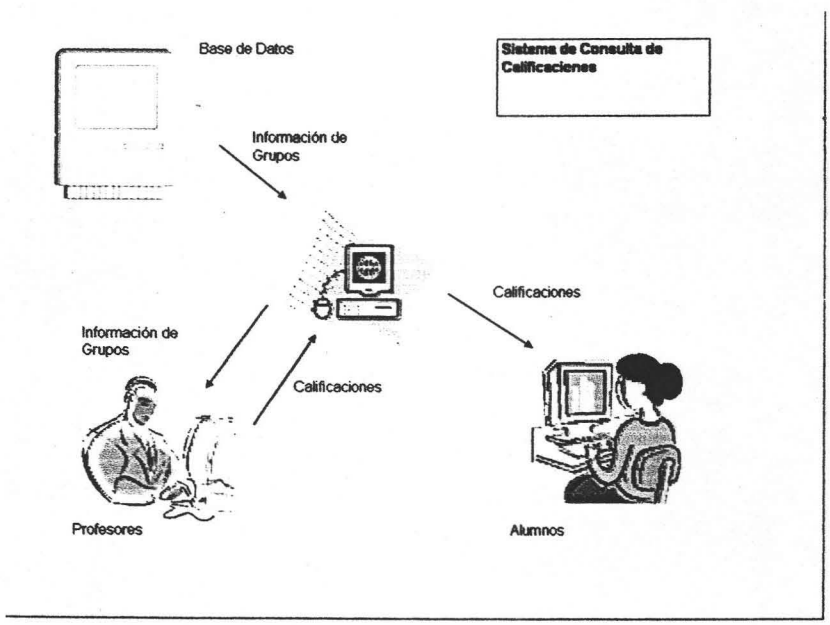


Imagen 01: Sistema de calificaciones

Para la consulta de una calificación, se ingresará un login. Este no será validado de una forma compleja. A continuación se presentará un arreglo, mostrando las calificaciones hasta el momento obtenidas. Esta tabla tendrá información como clave de la materia, nombre de la materia, calificación obtenida, y periodo en el que se presentó. Como información adicional, se encontrará el promedio y porcentaje de créditos.

En el ingreso de calificación se hará mediante un formato que tendrá los datos de matrícula, nombre del alumno, como encabezado vendrá el grupo y la asignatura. La parte a llenar en

el formato será de la calificación. Se añadirá el promedio de grupo y porcentaje de aprobación, el cual después se ingresará a una base para un reporte que se podrá consultar al final de cada periodo. El ingreso a esta parte del sistema estará restringido por medio de una validación.

Además, se podrá consultar el aprovechamiento por asignatura, para identificar en qué asignaturas hay mas reprobados, con esto, se podría decidir de manera más rápida y pronta la creación de grupos de apoyo para estas asignaturas.

Estas operaciones se realizaran por medio de un navegador Web, lo que facilitará y hará mas cómoda su realización.

3.2 Selección de herramientas de desarrollo

Para el desarrollo de nuestro sistema se empleará la tecnología ASP. Pero, ¿Porqué esta tecnología?. Debido a la naturaleza de la aplicación, un sistema en línea con una base de datos, ASP puede ser utilizado para el desarrollo de la aplicación debido a sus características, que son las siguientes:

- **Crear sitios Web Interactivos**: Sitios Web que intercambian información entre el sitio y el usuario. ASP permite a los administradores Web crear páginas que procesen información de un usuario y luego generen contenido dependiendo de los datos subministrados por este. Los sitios Web interactivos le permiten a los administradores Web personalizar el contenido de sus páginas para atraer mejor al usuario.
- **Crear sitios Web dinámicos**: Contienen páginas que muestran constantemente contenido cambiante. Cuando se usa ASP, se puede determinar el contenido que muestra una página Web dependiendo de muchos factores diferentes, fecha, hora, lugar de conexión, usuario que se conecta.

- **Crear aplicaciones:** muchas personas están acostumbradas a trabajar con aplicaciones, como agendas electrónicas y lectores de correo. Con ASP es posible crear aplicaciones que son albergadas en un sitio Web y son accedidas por un explorador Web.
- **Usar componentes incorporados**: ASP incluye componentes que son fáciles de usar y que pueden emplearse para tareas como rastrear estadísticas de usuario o verificar las capacidades del explorador Web de un usuario para que la página sea personalizada a este. ASP permite a los diseñadores Web crear y usar sus propios componentes.
- **Trabajar con bases de datos**: Una opción importante en ASP es su capacidad de conectarse a bases de datos. Las páginas ASP pueden usarse para que la información almacenada en una base de datos este disponible a los usuarios que visitan la página. Este es un método eficiente para mostrar información actualizada en un sitio Web. Además, se puede manipular esta información, se puede agregar, borrar o editar los registros de la base.
- **Mayor seguridad:** Debido a que el código ASP es ejecutado en el servidor de la Web, el usuario no puede acceder al código usado para crear una página ASP. Esto permite trabajar de forma más segura con datos sensitivos, como nombre de registro y contraseñas. Si un usuario visualiza el código fuente de una página ASP dentro de un explorador Web, todo lo que vera es el código HTML que fue generado por el servidor de la Web para crear la página y no el código ASP en sí.

Teniendo en cuenta las características mencionadas, es fácil saber porque no se escogen ciertas herramientas para desarrollar este sistema, pero aun con esto, hay herramientas que tiene características similares, entonces, ¿cómo saber cuando usar ASP?. El primer factor sería el tipo de servidor en el cual va a ser albergada la página. ASP es una tecnología Microsoft, por tanto, si el servidor donde se va a albergar esta página es un servidor Microsoft Windows 2000, ASP seria una opción natural, ya que este sistema soporta esta tecnología. Aunque estas páginas pueden ser almacenadas en servidores Unix, se pueden tener problemas de compatibilidad, además, para que usar tecnología ASP en una plataforma Unix, siendo que esta cuenta con tecnologías nativas similares.

Otro de los factores en consideración, concerniente al desarrollador o desarrolladores, es el de los Scripts. Cada tecnología de este tipo maneja un conjunto de Scripts, un desarrollador puede basarse en esto para la selección de alguna de las tecnologías. No sería conveniente si se tiene un conjunto de desarrolladores para un cierto Script, se tenga que cambiar este conjunto por cambiar de tecnología sin una razón justificada.

En lo que se refiere a bases de datos, las más utilizadas en la tecnología ASP son Microsoft Access y Microsoft SQL Server. Las dos son SMBD de Microsoft. Access es utilizado para bases de datos relativamente pequeñas, y Microsoft SQL es de utilidad para crear bases de datos grandes y de alto tráfico. Algunas características de SQL Server son:

- **Integración:** SQL Server proporciona compatibilidad con administración de datos y capacidades de programación a distintos niveles, desde grandes servidores a estaciones de trabajo de escritorio. Proporciona robustas capacidades de administración de los datos a los dispositivos. Presenta un modelo operativo y de programación coherente con el resto de la familia SQL Server, por lo que garantiza fácil integración con los sistemas existentes y el aprovechamiento de los conocimientos de desarrollo que ya se tienen. Puede integrarse con el Internet Information Server.

- **Base de datos relacional**: Aunque los dispositivos avanzan rápidamente, los recursos del sistema como la memoria disponible suelen ser escasos, así que resulta vital que un sistema de base de datos relacional sea lo más compacto posible al tiempo que sigue manteniendo la funcionalidad esencial. El rendimiento se mejora con un procesador de consultas optimizado.

Por otra parte, si se tiene una base de datos ya hecha, para evitar el problema de migrar esta a otro manejador, se puede buscar el componente para que esta se de de alta en el ODBC.

3.3 Carga de la información

Para ingresar estas calificaciones, se necesita de datos iniciales. Estos datos son generados por un Sistema de Admisión Escolar, lo que en la imagen 01 esta etiquetado como

"información de los grupos". Sería poco práctico que un profesor, para ingresar calificaciones, ingresara la asignatura, el nombre de esta, el grupo correspondiente y además las matriculas de sus alumnos. Esto podría generar errores en los datos. Además, esta información es capturada en algún momento de la inscripción a cursos o a exámenes especiales. Por tanto, es conveniente obtener esta información ya capturada. La información que necesita el sistema para iniciar, es la siguiente:

Clave del profesor: Llave única por la cual se identifica al profesor dentro del personal escolar. Asignada en el momento de darlo de alta en la base de datos del personal.

Asignación n: Asignatura(s) que imparte. n es un número entero que puede correr hasta el número máximo de asignaturas que puede tener un profesor en la institución. El valor n, depende de las normas, para el sistema se utilizara el valor de 3. El nombre de asignatura esta compuesto por la clave del grupo y la clave de materia. Para ser único este nombre dentro de la historia escolar, hace falta el periodo en que se imparte, este lo asigna el sistema de forma automática para la creación de historiales, por medio de una aplicación para indicar al sistema el cambio de periodo y permitir que nuevas asignaciones se hagan a las tablas que funcionan por periodo (asignación y tira de materias)

Elementos que tiene que validar el sistema de admisión escolar :

- Existencia del profesor en la base de datos de personal
- Existencia de las asignaturas en la base de datos de asignaturas.
- No repetir la asignatura.

Se mencionaron dos bases de datos, ligadas a la base de asignación por periodo, la de Personal, y la de Asignaturas. Estas contienen datos referentes a las entidades que están manejando. El diagrama de la página 47 muestra la estructura y relaciones de la base de datos.

Por medio del atributo clave se puede relacionar la entidad asignación con la entidad materias, el cual contiene el nombre de la materia, créditos, tipo, clave, plan.

Otro dato necesario antes de ingresar calificaciones es el de los alumnos pertenecientes a una asignatura. Un alumno puede estar inscrito a un número máximo de asignaturas dependiendo de la política escolar. Para nuestro sistema, este número será 6. Se tiene que utilizar la entidad asignatura. Es conveniente, por el tamaño de registros, tener bases de datos por plantel y carrera.

Estos son algunos datos requeridos, previamente validados por el sistema de admisión escolar

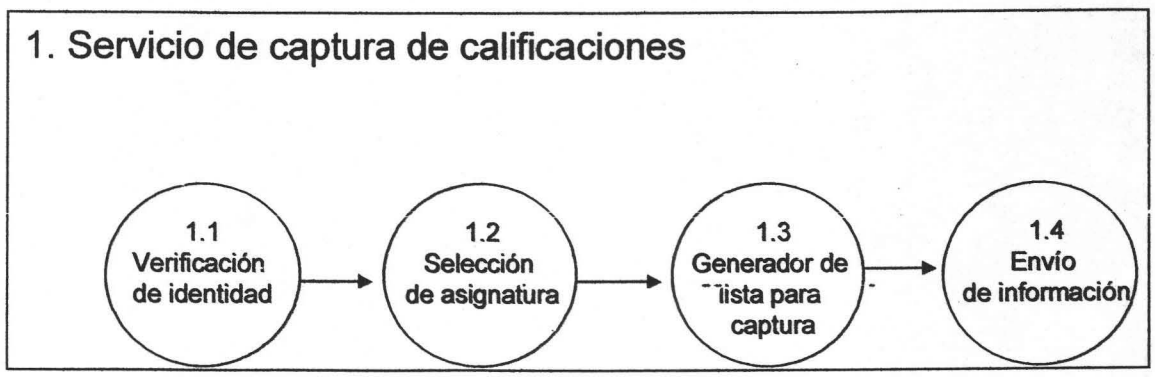
- Existencia de la matricula para inscripción
- Existencia de la asignatura a inscribir
- Seriación
- Requisitos
- Disponibilidad en grupo
- Asignatura del plan correspondiente
- No inscripción a materias aprobadas

Respecto a las validaciones mencionadas, para el sistema de admisión escolar, se debe tener un servicio, el cual permita ingresar planes de estudio, este debe de contemplar el ingreso para cada asignatura de materias antecedentes, número de créditos necesarios, verificación de no adelanto de un número de semestres y otras que se necesiten según el plan de estudio que se este manejando. Por lo mencionado, se puede notar que se pueden agregar más atributos a la entidad asignatura, de momento, solo se anexan los necesarios para el servicio de captura de calificaciones, además de que el sistema concerniente al trabajo es de Información escolar y no de admisión.

La información de qué alumno está inscrito a qué grupo, genera un documento en el momento de la inscripción llamado tira de materias, de ahí el nombre de la entidad. Estas entidades, almacenadas como tablas en una base de datos, son lo necesario para ejecutar el servicio de captura de calificaciones.

Hay que notar que estos datos son generados de manera periódica, en cada momento de inscripción a cursos o exámenes especiales, por lo que se tiene que decidir si se elimina la información de inscripción cuando no sea requerida (terminación de un periodo e inicio de otro, por ejemplo) o si se desea almacenar para formar una historia de que profesor-materia-fecha o de que alumno-materia-profesor-grupo. Esto puede no ser necesario en el caso alumnos, ya que el historial puede almacenar el Folio de un acta donde se asentaron las calificaciones. La información referente a que asignatura tuvo un profesor en cierto periodo, no es muy recurrente, por lo que también se puede buscar de forma manual actas. Una buena razón para almacenar la información antes mencionada, sería que el sistema de actas se llevará de forma electrónica también.

Lo anterior, es la entrada necesaria para el servicio de captura de calificaciones. Este servicio, comienza con el ingreso de un profesor al sistema, al ingresar su Clave y password, el sistema realiza una verificación, primero de existencia de clave y luego de correspondencia con el password. En este punto, se tienen que tomar consideraciones de seguridad, ya que esta información es de vital importancia. Una vez ingresado esto, se obtendrá información de las asignaturas correspondientes al profesor que ingresa, por medio de un acceso a la base de datos de asignación, dando a seleccionar de qué asignatura se quiere ingresar calificaciones. El sistema obtendrá la lista de alumnos de la base tira de materias para obtener la lista de los alumnos inscritos en este periodo. El profesor tendrá que seleccionar una calificación de las que se presenta a selección delante de la matricula. Esta puede ser 10, 9, 8,..., 6, 5 para este sistema, pudiendo agregar otra, dependiendo de políticas escolares. En caso de existir esta calificación, a causa de haber sido presentada con anterioridad la materia por el alumno, el sistema sustituirá la calificación no aprobatoria por la nueva calificación. No se puede sustituir calificaciones aprobatorias, ya que el sistema de admisión escolar no debe permitir inscripción a asignaturas ya aprobadas. Se almacena el número de veces cursada la materia y el número de exámenes presentados. Si se desea, se puede almacenar datos referentes a calificaciones no aprobatorias antes de sustituirlas por la nueva. El sistema calcula y almacena el aprovechamiento del grupo para su posterior uso. Se despliega un mensaje de envío satisfactorio.



Servicio de captura de calificaciones

Otra forma de ingreso de dato es la de corrección de una calificación. Esta se da después de una revisión de un examen mal calificado o cuando el profesor se da cuenta después de entregar actas de algún error en calificación. Esta corrección podrá hacerse por medio de un servicio, para el cual se necesitará login y password también. De hecho, es oportuno mencionar que la entidad encargada de administración escolar, tiene acceso a este servicio, para modificar la calificación en un historial o para ingresar calificaciones por un profesor en caso de que este no pueda por alguna razón.

Las Entidades y atributos pueden ser visualizados en este esquema. En la parte de historial no se queda hasta dos, sino que corre hasta el número de materias que un alumno puede cursar en la carrera.

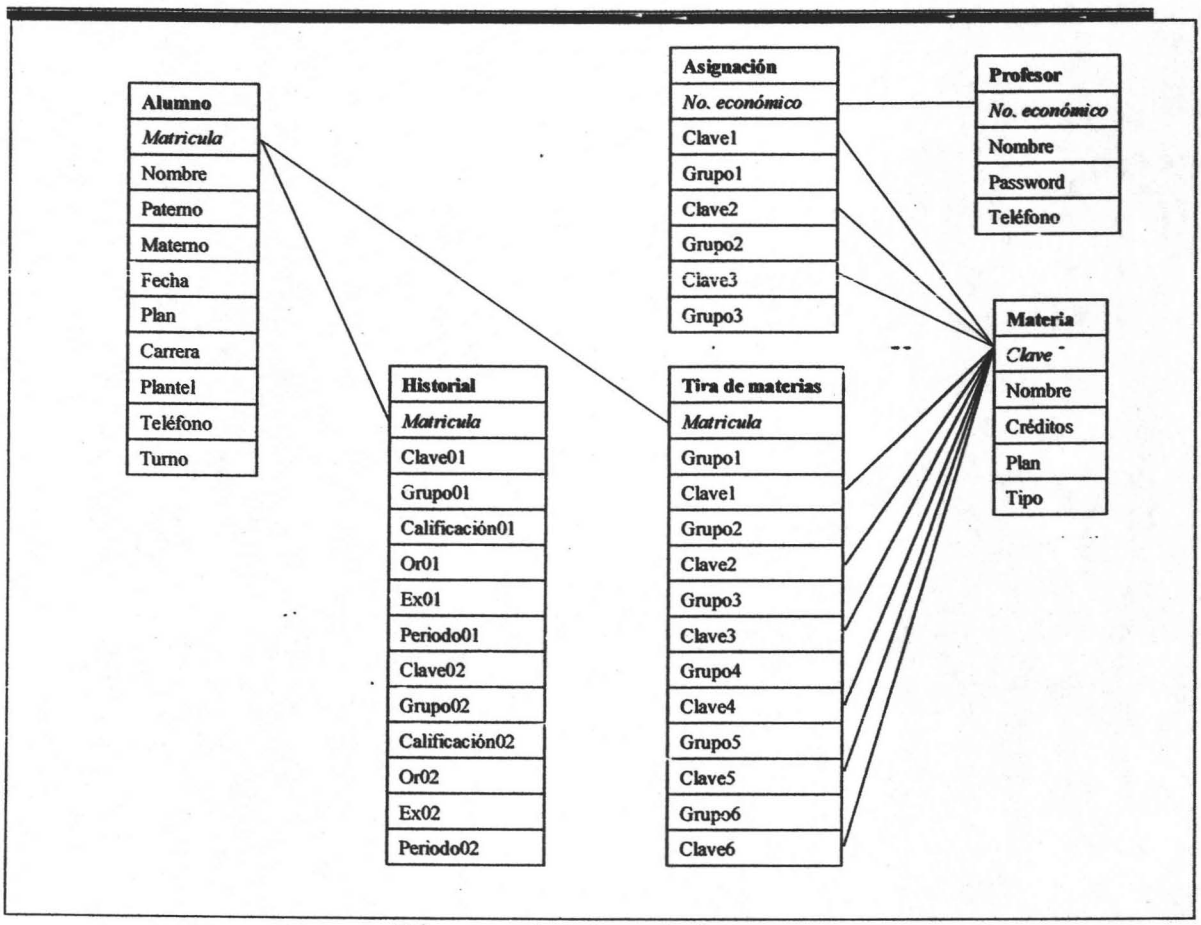


Diagrama de la base de datos

3.4 Consulta de la información.

Con el Historial, generado por el servicio captura, se puede hacer una consulta de calificaciones. El servicio de captura, de una lista de calificaciones, agrega cada calificación de la lista al correspondiente registro y campo del historial. En el momento que el listado de calificaciones es ingresado, se va llenando el historial de cada alumno que aparece en la lista. El proceso de captura se completa en el momento de que todas las listas de calificación son ingresadas. En este momento, se puede consultar el historial de completo de las calificaciones ingresadas.

El servicio funciona de la siguiente forma:

- Se pide matricula y password

- Se verifican las entradas
- Se accesa al registro en el historial.
- Se muestra la información

Como se puede ver, el servicio de consulta es mucho más sencillo en funcionamiento que el de captura, Anexo al servicio de consulta de historial, se puede agregar uno de consulta por materia, en caso de ser requerido.

El servicio de consulta por materia también genera un arreglo con el promedio y porcentaje aprobación. Para fines de análisis académico, este arreglo puede ser consultado, empleando como entrada la materia y periodo. Esta información ayuda para saber si alguna modificación como la forma de calificar, la implantación de cursos extra o algún otro método sirvió para mejorar el desempeño académico de los estudiantes.

En breve, es así como funciona el sistema. Se hizo más énfasis en las bases de datos ya que para este sistema son la parte principal. Las aplicaciones, son solo una forma de acceder a estas bases.

Capítulo 4: Desarrollo del prototipo de sistema

En este capítulo se explicará como funcionan los servicios del sistema. Además se explicará el plan de implantación del prototipo, para evaluar si las funciones están trabajando de manera correcta y a partir de esto poder desarrollar un sistema.

4.1 Plan de desarrollo

Un prototipo es un modelo (representado o simulado) fácilmente ampliable y modificable de un sistema específico, probablemente incluyendo su interfaz y su funcionalidad de entradas y salidas.

Este prototipo no requiere de un gran número de personas para su desarrollo ni de un tiempo excesivo. La parte que requiere esfuerzo humano considerable es la de la captura de información. Además de la captura de alumno y profesor ingresado, cada periodo escolar se debe almacenar una tira de materias y asignaciones a profesores. Esta cantidad de personal dependerá del número de alumnos que se tengan en la institución.

La metodología que se uso para el desarrollo de este prototipo fue el Modelo de Cascada. Cuyas fases son: Secuencia de requerimiento, diseño del sistema, diseño de programa, codificación, pruebas, operación y mantenimiento. Para este trabajo, se llega hasta la fase de codificación y pruebas.

Captura de calificaciones

Esta es la aplicación que permite captura de calificaciones a nuestro sistema. Para iniciar este proceso, se necesita información de una tabla con información del profesor. La información específica de esta tabla es la del número económico, que funciona como clave única para esta tabla, y la de password. Al ingresar un número económico y un password, primero se verifica la existencia de este password en la tabla profesor, y después la correspondencia de este número con el password. De no existir el número económico, o de no ser el password correspondiente, se mostrará un mensaje. Al ser verificadas ambas entradas, se continúa con la ejecución de la aplicación.

```
<HTML>
<HEAD>
<TITLE>Inserte login del profesor</TITLE>
</HEAD>
<BODY>

<Form Action="grupos.asp" Method="Post">
Login <Input type="text" Name="login"><BR>
Password <Input type="Password" name="password" maxlength="20"><BR>
<Input type="submit" Name="enviar" Value="Enviar ahora">
</Form>
</BODY>
</HTML>
```

código htm para ingresar login y password

El código anterior no inserta instrucciones ASP, y utiliza elementos básicos de formulario

```
<%
Set conexion=Server.Createobject("ADODB.Connection")
conexion.ConnectionTimeOut=60
conexion.Open "DSN=escuela"

ins="Select* From profesor where neconomico='"+Request.Form("login")+"'"
Set profesor=conexion.Execute(ins)


If profesor.eof=False Then
  If profesor(3)=Request.Form("password") then
    Response.Write("Bienvenido Profesor, Seleccione el Grupo a Calificar:")
    Server.Execute("Selgrupo.asp")
  Else
    Response.Write("Password Incorrecto")
  End If
Else
  Response.Write("Login incorrecto")
End If

conexion.Close
Set conexion=Nothing
%>
```

código en página grupos.asp

La primera página Web nos manda a esta página ASP, donde se abre la tabla profesor para verificar el password y la existencia de la matricula, de resultar ambas verdaderas, ejecuta en esa misma página el código de selgrupos.ASP, el cual genera los botones de opción correspondientes a los grupos del profesor. De este parte de código en adelante, se eliminan las etiquetas HTML iniciales y finales para ahorrar espacio.

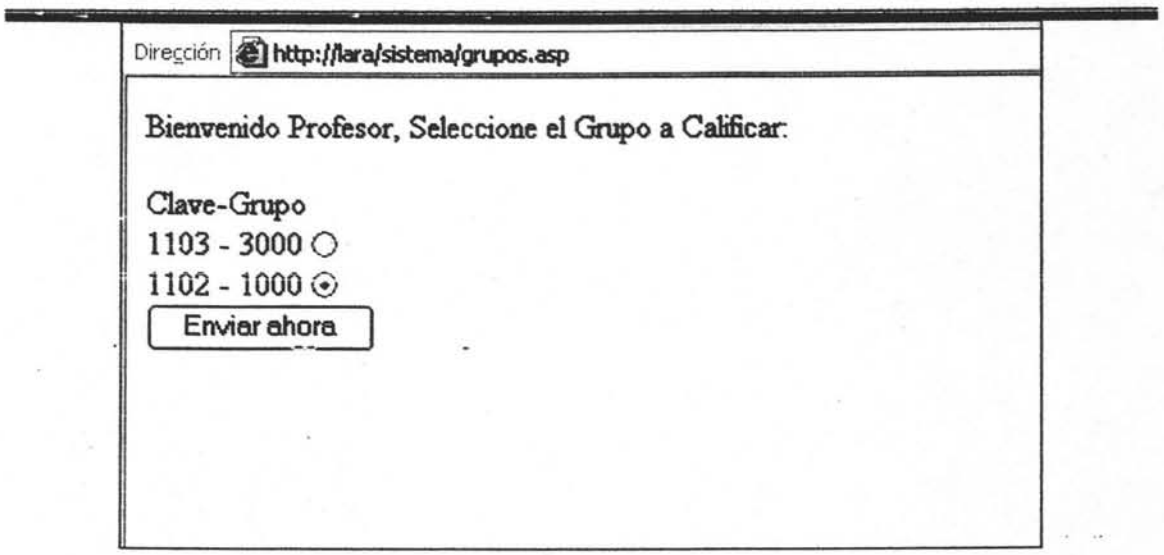
1.- Mostrar nombre del profesor, extraído de la tabla profesor del campo nombre, correspondiente al registro que coincide con el número económico.

2.- Mostrar la lista de asignaciones del profesor, extraída de la tabla asignación, correspondiente al registro que coincide con el número económico. Una asignación consta

de un grupo y una clave de materia. Esta lista puede ser extraída mediante un ciclo con tres iteraciones, debido a que es el número máximo de asignaciones a un profesor en el prototipo. Este número puede ser modificado según necesidades. Esta lista es mostrada con botones de opción.

```
<Form Action="lista.asp" Method="Post">
<%
co=Chr(34)

Set conexion=Server.Createobject("ADODB.Connection")
conexion.ConnectionTimeOut=60
conexion.Open "DSN=escuela"
ins2="Select* From asignacion where neconomico='"+Request.Form("login")+"'"
Set asignacion=conexion.Execute(ins2)
If asignacion.eof=true Then
  Response.Write("sin asignación")
  Response.End
End If

Response.Write("Clave-Grupo<BR>")
For i=1 to 3
  If asignacion(2*i)<>"" then
    Response.Write(asignacion(2*i-1)+" - "+asignacion(2*i))
  Else
    Exit For
  End if
  Response.Write("<Input Type="+co+"Radio"+co+" Name="+co+"Grupo"+co+_
  " Value="+co+asignacion(2*i-1)+"-"+asignacion(2*i)+co+" Checked"+">"+"<BR>")
Next

conexion.Close
Set conexion=Nothing

%>
```

selgrupos.asp genera los botones de opción de selección de grupos.



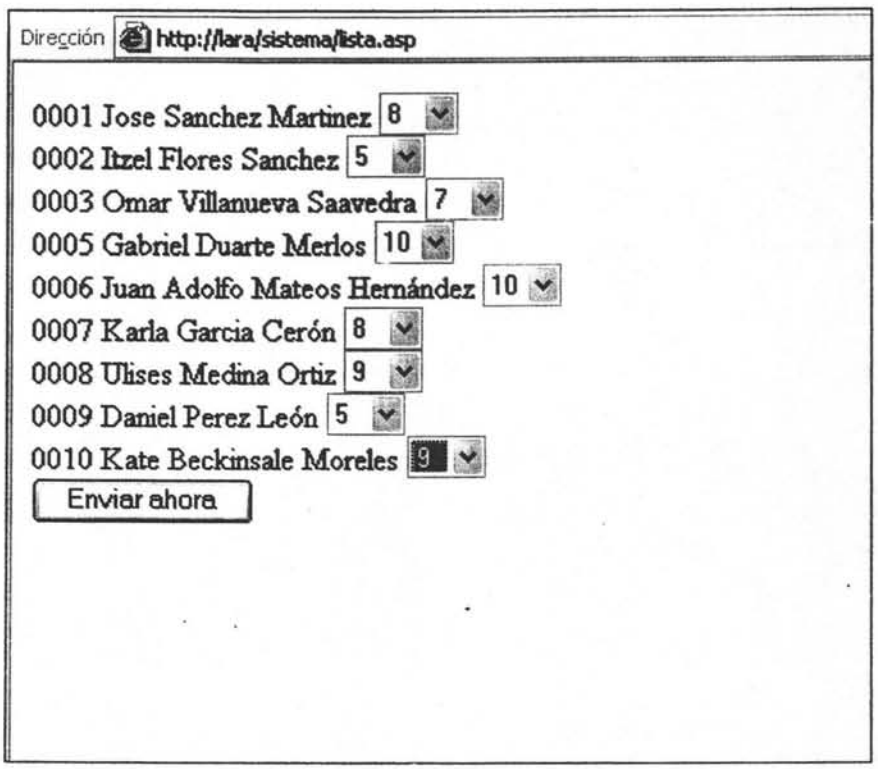
Pantalla de selección de grupo

Esta parte de código abre la tabla profesor para obtener las asignaturas de este y desplegarlas. Se mencionó que se puede incluir código ASP en el HTML. También que en el código ASP se pueden anexar etiquetas HTML siempre y cuando estén entre comillas. Cuando estas etiquetas HTML tienen comillas propias, las sustituimos por una función que nos devuelve las comillas insertando código ASCII de estas. En pocas palabras, tenemos que armar la cadena de texto de la etiqueta HTML utilizando concatenación y la función CHR.

3.- Al tener la asignación, se busca esta coincidencia en la tabla tira de materias, la cual contiene información de las matriculas de los alumnos inscritos, y la clave y grupo de las materias inscritas. Se tiene que buscar esta coincidencia en cada uno de los seis pares de campos que contienen información de las asignaciones en la tira de materias. Al seleccionar los registros que cumplen con la coincidencia de asignación, se muestra en pantalla una lista de las matriculas de estos alumnos y su nombre, extraído de la tabla alumno del campo nombre. Una lista desplegable acompaña a cada una de estos registros, cuyas opciones son las posibles calificaciones asignadas a la materia.

```
<%
Response.Write("<Form
Action="+chr(34)+"captura.asp?asig="+request.form("grupo")+chr(34)+"
Method="+chr(34)+"Post"+chr(34)+">")
Set conexion=Server.Createobject("ADODB.Connection")
conexion.ConnectionTimeOut=60
conexion.Open "DSN=escuela"
If len(request.form("grupo"))=9 then
  Clave=Mid(request.form("grupo"),1,4)
  grupo=Mid(request.form("grupo"),6,4)
Else
  Clave=Mid(request.form("grupo"),1,4)
  grupo=Mid(request.form("grupo"),6,5)
End if
ins="Select * From tiramaterias where (clave1='"+clave+"' and grupo1='"+grupo+"')
or"+_
"(clave2='"+clave+"' and grupo2='"+grupo+"') or"+_
"(clave3='"+clave+"' and grupo3='"+grupo+"') or"+_
"(clave4='"+clave+"' and grupo4='"+grupo+"') or"+_
"(clave5='"+clave+"' and grupo5='"+grupo+"') or"+_
"(clave6='"+clave+"' and grupo6='"+grupo+"')"

Set lista=conexion.Execute(ins)

If lista.eof=True then
  Response.Write("Asignatura sin Alumnos")
  Response.end
End if
Do while not lista.eof
  Response.write(lista(0)+" ")
  ins2="Select nombre,paterno,materno From alumno where
matricula='"+lista(0)+"'"
  Set nomcom=conexion.Execute(ins2)
  Response.Write(nomcom(0)+" "+nomcom(1)+" "+nomcom(2)+" ")

  Response.Write("<Select Name="+chr(34)+"m"+lista(0)+chr(34)+">")

  For I=5 to 10

    response.write("<Option Value="+lista(0)+cstr(i)+">"+cstr(i)+"</option>")
  Next
  Response.Write("</Select>")
  Response.Write("<BR>")
  lista.movenext
Loop%>
```

lista.asp. Despliega alumnos con una lista desplegable para las calificaciones



Pantalla de la generación de la lista para calificar

En esta parte del código la instrucción larga Select hace el trabajo. Esta fue separada con guiones bajos para su mejor entendimiento. Busca dentro de la tabla tira de materias la coincidencia de asignatura. Después, a cada elemento se le asigna una lista desplegable con calificaciones del 5 al 10.

4.- Cada una de las calificaciones se almacena en la tabla historial, en el registro correspondiente a la matricula, por lo que este proceso se repite el número de elementos en la lista de grupo. Se busca si esta materia ya se ha almacenado anteriormente, de no ser así, se almacena en un campo nuevo la información correspondiente a la calificación: clave, grupo, periodo, calificación y tipo de examen (Or: Ordinario o Ex: Extraordinario). Clave y grupo fueron extraídos con el historial al generar la lista, periodo es una variable del sistema, que es almacenada por un usuario antes del momento de una inscripción, y que funciona durante el periodo escolar. Ex y Or almacenan el número de veces que se presento cada tipo de examen. De encontrarse ya una calificación, y de ser reprobatoria, esta se cambiará por la nueva calificación y se incrementará el contador correspondiente,

cambiando el grupo y periodo. Si la calificación es aprobatoria, solo se incrementara el contador en la base de datos. La forma de almacenaje puede cambiarse según alguna otra política escolar. Este historial se utiliza también en el momento de inscripción para validar la misma.

```
<%

Set conexion=Server.Createobject("ADODB.Connection")
conexion.ConnectionTimeOut=60
conexion.Open "DSN=escuela"

set per=conexion.execute("select * from info where tipo='periodo'")
periodo=per(1)



If len(request.querystring("asig"))=9 then
  Clave=Mid(request.querystring("asig"),1,4)
  grupo=Mid(request.querystring("asig"),6,4)

Else
  Clave=Mid(request.querystring("asig"),1,4)
  grupo=Mid(request.querystring("asig"),6,5)

End if
suma=0
aprobados=0

For i=1 to Request.Form.count-1
bandera=False

  If Mid(request.form(i),Len(request.form(i)),1)<>0 Then
    matricula=Mid(request.form(i),1,Len(request.form(i))-1)
    Cal=Mid(request.form(i),Len(request.form(i)),1)
  Else
    matricula=Mid(request.form(i),1,Len(request.form(i))-2)
    Cal=Mid(request.form(i),Len(request.form(i))-1,2)
  End if
  suma=suma+cal
  if cal>=6 then
    aprobados=aprobados+1
  end if
```

```
Set Base=Server.CreateObject("ADODB.Recordset")
Base.Open "Select * From historial Where matricula='"+matricula+"'", conexion,1,2

  For j=1 to 15
    campo="clave"+cstr(j)
    If base(Campo)=Clave Then

      Base("grupo"+cstr(j))=grupo
      Base("Periodo"+cstr(j))=periodo
      Base("calificacion"+cstr(j))=cal
      If Len(grupo)=5 then
        Base("ex"+cstr(j))=Base("ex"+cstr(j))+1
      Else
        Base("or"+cstr(j))=Base("or"+cstr(j))+1
      End if
      Base.update
      bandera=True
        Exit For

    End If
  Next

if bandera<>True then

  For j=1 to 15
    campo="calificacion"+cstr(j)
    If base(Campo)=0 Then
      Base("Clave"+cstr(j))=clave
      Base("grupo"+cstr(j))=grupo
      Base("Periodo"+cstr(j))=periodo
      Base("calificacion"+cstr(j))=cal
      If Len(grupo)=5 then
        Base("ex"+cstr(j))=Base("ex"+cstr(j))+1
      Else
        Base("or"+cstr(j))=Base("or"+cstr(j))+1
      End if
      Base.update
      Exit For
    End If
  Next
end if

next
```

```
response.write(ins2)
Set recordset=server.createobject("ADODB.recordset")
recordset.open "Select * from grupal where clave='"+clave+"' and
grupo='"+grupo+"' and periodo='"+periodo+"'",conexion,1,2

If recordset.EOF=true then

  ins="Insert Into grupal(clave,Grupo,Periodo, suma,aprobacion,nal) Values('"+_
  clave+"','"+grupo+"','"+periodo+"','"+cstr(suma)+"','"+cstr(aprobados)+"','"+cstr(i-
1)+"')"
  Set registros=conexion.execute(ins)
Else
  recordset("suma")=suma
  recordset("aprobacion")=aprobados
  recordset("nal")=i-1
  recordset.update
End if


base.close
conexion.close
Set conexion=nothing
Response.Write("Su información ha sido cargada exitosamente")


%>
```

captura.asp es la parte final. Lleva las calificaciones a la base de datos

5. Para llevar a cabo una consulta de promedios, se tiene que almacenar en una tabla, llamada promedios, el periodo, grupo, clave, promedio grupal, número de aprobados, total de alumnos y carrera. Para el promedio grupal se toman las calificaciones NA como 5, NP no se contabiliza.

La corrección de calificaciones lleva esta misma estructura, a diferencia de que no se tiene que validar un profesor, sino un password de administrador. Para corregir algún error en la calificación, la aplicación se ubica en el registro y campo de los cuales se desea hacer la corrección. El registro esta dado por la matricula del alumno, el campo es la calificación

correspondiente a la clave y grupo ingresados. A diferencia de cuando se ingresa una calificación, el contador no se incrementa. Si se necesitara llevar una maca para esta corrección en el historial electrónico, se puede agregar una marca en la clave de grupo, ya que agregar otro campo sería demasiado tomando en cuenta el número de correcciones que se realizan.

Consulta de calificaciones

La consulta de calificaciones resulta sencilla por la forma en que fue construida la base de datos. La tabla historial contiene la información base para esta tarea.

1.- Se piden como datos de entrada la matricula del estudiante y su fecha de nacimiento para ser usada como password. Si se desea se puede hacer una aplicación para que este password sea modificado, contemplando futuros problemas por perdidas del mismo. Este formulario es muy parecido al del ingreso del login del profesor, por lo que no se anexa.

2.- Se corrobora la existencia de a matricula y la correspondencia de la fecha de nacimiento, utilizando la tabla alumno de la base de datos. De ser correctos ambos datos, se continúa con la ejecución del sistema.

```
<%
Set conexion=Server.Createobject("ADODB.Connection")
conexion.ConnectionTimeOut=60
conexion.Open "DSN=escuela"

ins="Select* From Alumno where matricula='"+Request.Form("login")+"'"
Set tablaalumno=conexion.Execute(ins)



If tablaalumno.eof=False Then
  password=mid(tablaalumno("Fecha"),1,2)+mid(tablaalumno("Fecha"),4,2)+_
  mid(tablaalumno("Fecha"),9,2)
  If password=Request.Form("password") then
    Response.Write(tablaalumno("nombre")+""+tablaalumno("paterno")+_
tablaalumno("materno"))
    Server.Execute("creartabla.asp")
```

```
  Else
    Response.Write("Password Incorrecto")
  End If
Else
  Response.Write("Login incorrecto")
End If

conexion.Close
Set conexion=Nothing

%>
```

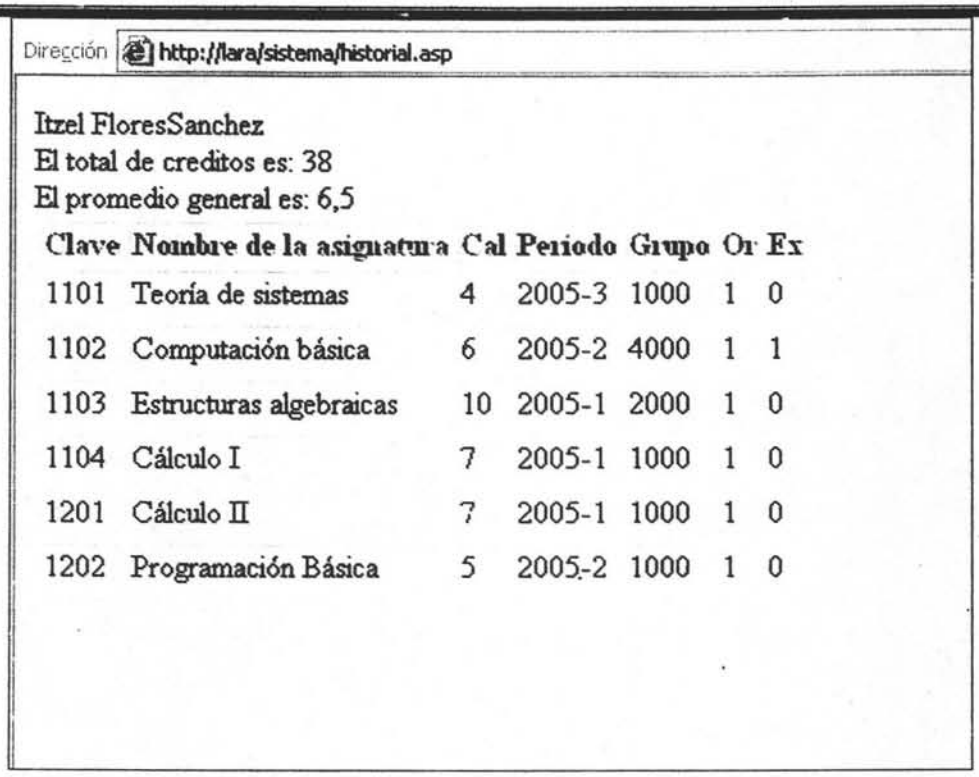
historial.asp valida y ejecuta otra página para generar la tabla.

3.- Se despliega en pantalla la lista de calificaciones de las asignaturas cursadas. Las calificaciones cursadas se encuentran en todos los campos hasta encontrar una cadena vacía en algún campo, lo que indica que la anterior fue la última calificación registrada Esta información es obtenida de la tabla historial. Información adicional se obtiene de la tabla materia, como el número de créditos, el nombre de la materia y si es obligatoria u optativa.

```
<%
Set conexion=Server.Createobject("ADODB.Connection")
conexion.ConnectionTimeOut=60
conexion.Open "DSN=escuela"

ins="Select * From historial where matricula='"+Request.Form("login")+"'"
Set historial=conexion.Execute(ins)

ins="Select * From materia where clave='"+Request.Form("login")+"'"
Set materia=conexion.Execute(ins)

Creditos=0
sumatoria=0
Cont=0

Response.Write("<Table Border=1><Tr><Th>Clave</Th><Th>Nombre de la
asignatura</Th><Th>Cal</Th><Th>Periodo</Th><Th>Grupo</Th><Th>Or</Th><
Th>Ex</Th></Tr>")

For i=1 to 15
  If historial("Calificacion"+cstr(i))<>0 Then
    response.write("<Tr>")
```

```
    Response.Write("<Td>")
    Response.Write(historial("clave"+cstr(i)))
    Response.Write("</Td>")
    ins="Select * From materia where clave='"+historial("clave"+cstr(i))+"'"
    Set materia=conexion.Execute(ins)

    Response.Write("<Td>")
    Response.Write(materia("nombre"))
    Response.Write("</Td>")
    Response.write("<Td>")
    Response.Write(historial("calificacion"+cstr(i)))
    If historial("calificacion"+cstr(i))>5 then
      creditos=creditos+materia("creditos")
    End if
    sumatoria=sumatoria+historial("calificacion"+cstr(i))
    cont=cont+1
    Response.Write("</Td>")
    Response.write("<Td>")
    Response.Write(historial("periodo"+cstr(i)))
    Response.Write("</Td>")
    Response.write("<Td>")
    Response.Write(historial("grupo"+cstr(i)))
    Response.Write("</Td>")
    Response.write("<Td>")
    Response.Write(historial("or"+cstr(i)))
    Response.Write("</Td>")
    Response.write("<Td>")
    Response.Write(historial("ex"+cstr(i)))
    Response.Write("</Td>")


    response.write("<Tr>")
  Else
    Exit For
  End If
next

response.Write("El total de creditos es: "+cstr(creditos)+"<BR>")
Response.Write("El promedio general es: "+cstr(sumatoria/cont))

%>
```

creartabla.asp Genera la tabla en base al historial del alumno.



Pantalla de Historial

4.- Se tiene que utilizar un contador para saber cuantas materias tuvieron calificación, ya que muchos de los campos de historial estarán vacíos hasta el momento de que el alumno concluya el plan de estudios. Con este se ejecutará un ciclo, con el cual, del registro obtenido, se obtendrá el promedio de las calificaciones aprobatorias y del porcentaje de créditos. Para obtener un porcentaje de créditos, se tiene que saber cuantos créditos obligatorios y optativos son 100%. Si esta cantidad es la misma para las distintas carreras, se agrega como constante en el código del programa, de lo contrario, se tiene que generar una tabla que contenga esta información.

Consulta por materia

Esta consulta se realiza utilizando la tabla que contiene información de grupo, clave, periodo, número de alumnos, suma de calificaciones y número de aprobados, generada en el momento de capturar calificaciones. Esta tabla tiene promedios grupales, con los cuales se puede obtener el promedio por materia en cierto periodo.

Esta consulta es alimentada por medio de la entrada clave y periodo. Entonces se despliega las asignaturas que cumplen con esta característica. Requiere de un formulario de entrada, cuyo código es muy parecido a los formularios de entrada de los procesos anteriores, por lo que no se anexa.

El siguiente es el código que genera la tabla de consulta.

```
<%

Set conexion=Server.Createobject("ADODB.Connection")
conexion.ConnectionTimeOut=60
conexion.Open "DSN=escuela"

ins="Select * From grupal where clave='"+Request.Form("clave")+"' and
periodo='"+request.form("periodo")+"'"
Set tabla=conexion.Execute(ins)
response.write("Clave de materia: "+request.form("clave")+", periodo:
"+request.form("periodo")+"<BR>")
if tabla.eof=True then
  Response.write("Sin coincidencias")
Else
  response.write("<Table
border=1><Tr><Th>Grupo</Th><Th>promedio</Th><Th>Aprobación</Th></Tr>")
  Do while not tabla.eof
    response.write("<Tr>")
    Grupo=tabla(1)
    suma=tabla(3)
    aprobados=tabla(4)
    nal=tabla(5)

    response.write("<Td>")
    response.write(grupo)
    response.write("</Td>")

    response.write("<Td>")
    response.write(suma/nal)
    response.write("</Td>")

    response.write("<Td>")
    response.write(aprobados/nal)
    response.write("</Td>")
```

```
    response.write("<Tr>")
    tabla.movenext
  loop
  response.write("</Table>")

end if

conexion.close
set conexion=nothing
```

consulta.asp, genera consulta de clave

Este servicio de consulta pretende ser usado por administración escolar para la decisión en la construcción de grupos y de ayuda a la creación de exámenes, aunque es un servicio que si se desea puede dejarse abierto para cualquier estudiante que lo quiera consultar.

4.2 Plan de Implantación

El prototipo será probado en una maquina con el sistema operativo Windows XP, utilizando IIS. Para este primer prototipo la base será realizada en Access XP, que es el SMBD con que se cuenta. EL SMBD que se escoja no altera el funcionamiento del programa, ya que si se desea que sea otro, solo se da de alta esta nueva base en el ODBC con el nombre de la base que se probó el prototipo, y no se necesita hacer ninguna alteración en el sistema. De realizarse de esta forma, el sistema estaría trabajando en forma local. Este prototipo también se almacenara en un servidor que soporte el software mencionado, para su verificación de funcionamiento en línea. Para el prototipo, se seleccionará un servidor de hospedaje Web gratuito que permita alojar una base de datos. De hecho, el uso de Access fue motivado a que en la mayoría de los servidores mencionados soportan bases de este tipo.

Por cuestiones de seguridad, esto debe funcionar solo para el prototipo, ya que es necesario cuando esté el sistema terminado, se aloje en un servidor dedicado dentro de la institución, o de menos en un servidor que garantice la seguridad y estancia del sistema. Este servidor, por la tecnología Web utilizada, deberá preferentemente contar con Sistema Operativo

Windows enfocado a Red (Windows 2000 o 2003 Server). El tener un servidor Web dentro de la institución, implicaría además de tener la computadora, una dirección IP fija, el respectivo acceso y un dominio. La implantación de este sistema no requiere de adquisición de equipo y/o servicios adicionales, porque en muchas de las instituciones educativas ya se cuenta con este tipo de servidor para alojar sus páginas Web, por lo que solo resta la tarea de crear y alojar el sistema dentro de este servidor.

Otra ventaja de este prototipo, es que, sin tomar en cuenta las computadoras necesarias para la captura en inscripción escolar, no se necesita de instalación de terminales, ya que están serian las computadoras de cada usuario que requiera del sistema, o cualquier computadora con acceso a Internet. Es necesario, una vez creado el sistema, difundir la dirección Web de este para su uso.

4.3 Problemas para la implantación de la autentificación

La autentificación de este sistema corrobora la existencia y correcta correspondencia de un login y password, que pueden ser interceptados en el momento de la transacción. A continuación se mencionaran algunos de los métodos de autentificación, con seguridad incorporada mediante el encriptamiento de los datos críticos. Aunque no se mencione en este trabajo el como realizarlo, se debe tener cuidado en la seguridad del servidor, ya que este puede también ser atacado.

La forma más fácil de hacer esta autentificación segura es por medio la autentificación integrada por Windows NTLM, pero no es la más conveniente. Cuando se accesa a un sitio con esta autentificación, aparece una ventana pidiendo login y password. A diferencia de la autentificación básica, esta no envía el texto plano como información, lo envía encriptado. Tiene como ventajas que no requiere de software adicional ni de un proceso complejo, solo es una configuración. La desventaja es que solo Internet Explorer soporta esta característica y de que hay que hacer una lista de grupos y/o usuarios que van a tener acceso al sitio Web.

Una forma de proteger la información que se envía es encriptando esta. Existen varios tipos y algoritmos para realizar este proceso, entonces queda la tarea de desarrollarlos dentro de la aplicación. Otra forma es comprando algún producto de los existentes en el mercado.

La forma de encriptación que se seleccionó para el prototipo es mediante una librería. Esta es CriptoASP, distribuida de forma gratuita por el Instituto para la Seguridad en Internet. CriptoASP 1.0 es una DLL ActiveX pensada para ser utilizada en servidores Web en entornos IIS/ASP, aunque también puede ser utilizada por cualquier otra aplicación compatible con ActiveX. Su finalidad es proporcionar mecanismos criptográficos de alta seguridad a las aplicaciones Web que los necesiten. En concreto permite realizar operaciones de cifrado y descifrado, creación de resúmenes criptográficos y generación de números pseudo aleatorios.

Cualquier aplicación que utilice criptografía, necesite proteger datos sensibles o crear números aleatorios puede beneficiarse de CriptoASP:

- Cifrado o resumen criptográfico de las contraseñas de usuario en la base de datos
- Generación de salts (números aleatorios que se utilizan junto con las contraseñas para evitar que dos usuarios con una misma contraseña obtengan el mismo resultado) para almacenar junto a las contraseñas en las bases de datos.
- Generación de tickets, testigos o identificadores de sesión aleatorios.
- Almacenamiento cifrado en la base de datos de la información sensible de los usuarios.

Para crear objeto y utilizar las opciones de CriptoASP se tiene que instalar la librería en el servidor donde se va a ejecutar las aplicaciones. Esto se hace por medio del comando *msgsvr32*. Se tiene que ejecutar con un único parámetro, la ubicación del archivo DLL.

msgsvr32 ruta_de_acceso\instisec.dll

Los siguientes son métodos para el uso de CriptoASP. Estos pueden considerarse como una sintaxis y una breve descripción de la instrucción.

Aleatorio(Longitud)

Devuelve un String con la cadena aleatoria de Longitud bytes.

Longitud Requerido: Integer. Representa la longitud expresada en bytes del número aleatorio que se desea generar.

Hash(Texto[, Algoritmo])

Devuelve un String con el resumen criptográfico.

Texto Requerido: String. El texto cuyo hash se desea calcular.

Algoritmo Opcional: Integer. El tipo de algoritmo de hash a utilizar: 0 - MD2; 1 - MD4; 2 - MD5; 3 - SHA. Por defecto se utiliza MD5 si no se suministra este parámetro.

Cifrar(TextoClaro, Clave[, Algoritmo])

Devuelve un String con el texto cifrado.

TextoClaro Requerido: String. El texto en claro que se desea cifrar.

Clave Requerido: String. La clave que se utilizará para el cifrado/descifrado.

Algoritmo Opcional: Integer. El tipo de algoritmo de cifrado a utilizar: 0 - RC2; 1 - RC4; 2 - DES. Por defecto se utiliza DES si no se suministra este parámetro.

Descifrar(TextoCifrado, Clave[, Algoritmo])

Devuelve un String con el texto descifrado.

TextoCifrado Requerido: String. El texto cifrado que se desea descifrar.

Clave Requerido: String. La clave que se utilizará para el cifrado/descifrado.

Algoritmo Opcional: Integer. El tipo de algoritmo de cifrado a utilizar: 0 - RC2; 1 - RC4; 2 - DES. Por defecto se utiliza DES si no se suministra este parámetro.

Los siguientes son algunos ejemplos de cómo utilizar esta sintaxis:

```
<%
' Creación del objeto CripoASP
Set oCripto = Server.CreateObject("Instisec.Cripto")

' Generación de un número aleatorio de 128 bits
strAleatorio = oCripto.Aleatorio(16)

' Resumen criptográfico con el algoritmo SHA del texto enviado por el usuario
strHash = oCripto.Hash(strTexto,3)

' Cifrado de un texto con una clave introducida por el usuario utilizando DES
strCifrado = oCripto.Cifrar(strClaro,strClave,2)

' Descifrado del texto anterior
strDescifrado = oCripto.Descifrar(strCifrado,strClave,2)

' Liberación del objeto CriptoASP
Set oCripto = Nothing
%>
```

El desarrollo del prototipo, sin todas las características finales que presentaría un sistema, es un buen punto de partida para la evaluación y modificación. Se ha establecido las funciones de este prototipo, en que va ha ser desarrollado e implementado. Resta verlo en funcionamiento para determinar si es todo lo que se espera del sistema si se necesita agregar o corregir algo.

Conclusiones

El objetivo de este trabajo, el de realizar un prototipo de sistema de información escolar con la tecnología ASP, se satisface al termino del mismo. Básicamente, el poder ingresar datos a una base en línea, para que estos puedan ser recuperados y produzcan información escolar, es la finalidad del prototipo, la cual fue alcanzada. Además, este prototipo servirá como punto de partida llegar a un objetivo superior al propuesto, el de un Sistema de administración escolar.

El desarrollo de aplicaciones en línea tiene cada vez más popularidad, debido a que estas aplicaciones son más cómodas para los usuarios ya que permiten un acceso a distancia y con un mejor horario. Este tipo de aplicaciones se ejecutan en un servidor, por lo que solo deben ser instaladas en él, no teniendo que hacer instalaciones en las máquinas que solamente usan la aplicación. Esto implica que cualquier actualización o mantenimiento, solo se le dará a la aplicación alojada en el servidor. Este prototipo de sistema permite por medio de interfaz de fácil entendimiento, que un usuario pueda almacenar y consultar de forma sencilla información almacenada en la base de datos.

Un punto a considerar, es en cuanto a la tecnología, no se puede afirmar que existe una tecnología mejor para todos los casos. Cada implantación requiere de un análisis que proporcione la tecnología más adecuada dependiendo de las condiciones del problema, y restricciones la implantación.

El desarrollo de software es más que el conocimiento de la tecnología que será utilizada. Hay que tener una visón lógica de cómo atacar el problema. Si de principio los requerimientos no están completos, se escoge una forma inadecuada de realizar las funciones o de como almacenar los datos, o no se consideran ciertos sucesos especiales que puedan ocurrir en la ejecución del sistema, aunque participen los mejores desarrolladores, este software no realizará sus funciones, y podrá causar más problemas de los que resuelve.

Bibliografía

Lisa Lopuck, "Web Design for Dummies", For Dummies, Estados Unidos, 2001.

Dick Oliver, Michael Morrison, "Sams Teach Yourself HTML & XHTML in 24 Hours, Sixth Edition", Sams, Estados Unidos, 2003

Doug Lowe, "Networking for Dummies, Sixth Edition", For Dummies, Estados Unidos,2002

Paul Whitehead, "Active Server Pages 3.0", St editorial Inc.,Panama, 2001.

Mark L. Gillenson, "Introducción a la bases de datos", McGraw-Hill, México, 1988