

ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL



**FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y
COMPUTACIÓN**

LICENCIATURA EN SISTEMAS DE INFORMACIÓN

TESIS DE GRADO

**PREVIO A LA OBTENCIÓN DEL TÍTULO DE:
LICENCIADO EN SISTEMAS DE INFORMACIÓN**

**TEMA: SISTEMA DE RESERVACIÓN DE
HABITACIONES, MESAS Y SALONES EN LÍNEA**

AUTORES:

**MANUEL RODDY GALARZA GALARZA
ROXANA ALEJANDRA GRANDA MEDINA
CLAUDIA TIYI MÁRQUEZ DE LA PRESA**

DIRECTOR:

ING. CARLOS MARTÍN BARREIRO

**AÑO
2002**

DECLARACIÓN EXPRESA

La responsabilidad de los hechos, ideas y doctrinas expuestas en esta Tesis de Grado nos corresponden exclusivamente; y el patrimonio intelectual de la misma a **L.S.I. (Licenciatura en Sistemas de Información)** de la Escuela Superior Politécnica del Litoral, según el Reglamento de Exámenes y Títulos Profesionales de la ESPOL.

FIRMA DE LOS AUTORES

Manuel Roddy Galarza Galarza

Roxana Alejandra Granda Medina

Claudia Tiyi Márquez de la Presa

FIRMA DEL TRIBUNAL CALIFICADOR

Ing. Mónica Villavicencio C.
Coordinadora

Ing. Carlos Martín Barreiro
Director de Tópico de Graduación

MIEMBROS DEL TRIBUNAL

Ing. Néstor Arreaga
Miembro Principal

Lic. Jorge Olaya
Miembro Principal

DEDICATORIA

A Dios, por darnos la oportunidad de disfrutar de este triunfo plasmado en la culminación exitosa de nuestras carreras, orgullo de nuestros padres, familiares y amigos, para ellos está dedicado con cariño ésta tesis.

*Claudia,
Manuel,
Roxana*

AGRADECIMIENTO

Gracias a Dios y a nuestra Madre Celestial María, que día a día nos han acompañado para nuestro bienestar.

A nuestros padres por la oportunidad de culminar ésta carrera con su apoyo fiel e incondicional.

A todos nuestros compañeros y amigos que nos han brindado su alegría, apoyo, consejo y conocimiento.

Y a todas aquellas personas que queremos y quienes de una u otra manera nos han ayudado.

Simplemente a todos,

Muchas Gracias!!

*Claudia,
Manuel,
Roxana*

RESUMEN EJECUTIVO

El Internet hoy en día, es una herramienta importante, que ayuda a las personas a realizar sus tareas, gracias a que posee una variedad ilimitada de buscadores, navegadores y aplicaciones fáciles de utilizar, ayudando a incrementar la productividad de los usuarios. Este fenómeno se ha incrementado mayormente en los últimos 5 años con la proliferación de ordenadores personales, de excelente rendimiento y de bajo costo en el mercado, y a su vez, la existencia de múltiples proveedores de servicios de Internet que ofrecen conexiones rápidas, con tarifas competitivas, a todo esto le sumamos que existe gran facilidad para encontrar temas diversos, siendo altamente explotada la red, con fines educativos, investigativos, de negocios, etc.

Por estas razones se decidió elaborar un Sistema de Información denominado *Mis.Reservaciones.com*, el mismo que comprende el registro de reservaciones en línea de habitaciones, mesas y salones a través del web, adicionalmente se desarrolló una aplicación standalone, que permitirá dar mantenimiento a las tablas referenciales. Todo esto fue realizado con arquitectura cliente/servidor, haciendo uso del modelo de tres capas, complementándolo con el uso de tecnología COM/DCOM, en el cual participaron tres tipos de procesos servidores: Web Server, Transaction Processing Server y Data Base Server.

Por último, debió emplearse una tecnología que permita la generación de páginas web dinámicamente. Utilizando para ello la tecnología AS(Active Server Pages).

INDICE GENERAL

RESUMEN EJECUTIVO

INTRODUCCIÓN

Capítulo 1

1.	ESPECIFICACIONES DEL PROYECTO	1
1.1.	Justificación	1
1.2.	Objetivos Generales	2
1.3.	Especificaciones Funcionales	3
1.3.1.	Funcionalidades del Sistema Web (Sistema on-line)	3
1.3.2.	Funcionalidades del Sistema Standalone	5
1.4.	Limitaciones y Restricciones	6
1.5.	Alcances	6

Capítulo 2

2.	ANÁLISIS DE LA INDUSTRIA Y EL PRODUCTO	8
2.1.	Definición de la industria y el producto	8
2.2.	Clientes	9
2.2.1.	Comportamiento del consumidor ante el mercado convencional	9
2.2.2.	Reacciones del consumidor ante el mercado virtual	10
2.2.3.	Mercado Virtual vs. Mercado Convencional	10
2.3.	Competidores	12
2.3.1.	Funcionalidad de los Competidores Convencionales Existentes	13
2.4.	Productos Sustitutos	14
2.5.	Análisis FODA del Sistema	14

Capítulo 3

3.	MODELO DE NEGOCIOS	16
3.1.	Definición del Modelo de Negocios	16
3.2.	Funcionalidad General	17
3.2.1.	Diagrama de Flujo de procesos	18
3.3.	Formas de Pago	19
3.4.	Seguridad	19

Capítulo 4

4.	ARQUITECTURA DEL SISTEMA	21
4.1.	Justificación de la Selección del Modelo	23
4.2.	Tipos de Procesos Clientes	24
4.3.	Tipos de Procesos Servidores	26
4.4.	Tipos de Middleware	29
4.5.	Tecnologías y Herramientas escogidas para la Implantación	30
4.6.	Requerimientos de Hardware y Software para la puesta en producción	32

Capítulo 5

5.	IMPLEMENTACIÓN DE LA CAPA DE DATOS	36
5.1.	Modelo Lógico de Datos	36
5.2.	Modelo Físico de Datos	41
5.3.	Diagrama Entidad - Relación	76
5.4.	Procedimientos almacenados (Stored Procedures)	79

Capítulo 6

6.	IMPLEMENTACIÓN DE LA CAPA DE NEGOCIOS	156
6.1.	Formatos para el Diseño de Componentes	156
6.2.	Formatos para variables de sistema	160
6.3.	Estructura de los Componentes de Acceso a Datos	162
6.4.	Objetos distribuidos de acceso a datos	165
6.5.	Comunicación con el proceso Servidor de Datos	165
6.6.	Estructura de los Componentes de Negocios	165
6.7.	Objetos distribuidos de negocios	169
6.8.	Comunicación entre objetos distribuidos	169
6.9.	Proceso Servidor Transaccional	169

Capítulo 7

7.	IMPLEMENTACIÓN DE LA CAPA DE PRESENTACIÓN	171
7.1.	Diseño del Sitio Web – Cliente Basado en Browser	171
7.2.	Cliente Standalone	178
7.3.	HTML	181
7.4.	Cliente Script - DHTML	182
7.5.	Server Script - ASP	183
7.6.	CSS	184
7.7.	Comunicación con Componentes de Negocios	185
7.8.	Proceso Servidor Web	186

CONCLUSIONES

RECOMENDACIONES

BIBLIOGRAFÍAS

APÉNDICE A – ARQUITECTURA CLIENTE/SERVIDOR

APÉNDICE B – PROGRAMACIÓN EN EL WEB

APÉNDICE C – COM / DCOM / COM+

APÉNDICE D – IIS Y ASP

INTRODUCCIÓN

La presente tesis de grado corresponde al Tópico para la Licenciatura en Sistemas de Información, la misma que implicaba que al final se desarrolle una aplicación Web con el uso de las tecnologías COM/DCOM y ASP. Como tema asignado tenemos el desarrollo de una aplicación Web que permita realizar reservaciones en línea de habitaciones, mesas y salones de un Hotel ó Restaurante.

Para ello se requería que el sistema de información sea implementado bajo la arquitectura Cliente/Servidor, haciendo uso del modelo de 3 capas, además se debía hacer uso de la tecnología COM/DCOM para el desarrollo de los componentes de acceso a datos y de negocios. También se debía emplear una tecnología que permita la generación de páginas Web dinámicamente: ASP. Tanto los componentes como las páginas Web debían ser administrados bajo tres servidores.

- Web Server.
- Transaction Processing Server.
- Database Server.

El Sistema de Información también debe implementar 3 diferentes tipos de lógica: lógica de presentación, lógica de negocios y lógica de acceso a datos. La arquitectura del sistema dependerá de cómo estos 3 diferentes tipos de lógica serán agrupados y posteriormente ubicados en las capas, y

cómo se complementarán con los 3 tipos de procesos servidores mencionados anteriormente. Además para cada capa se requiere que la aplicación Web realice lo siguiente:

- Para la Capa 1, el proceso cliente que implementaría la lógica de presentación debe ser de tipo dependiente, es decir, sólo podrá ser ejecutado dentro de un contenedor como es el caso del Web Browser.
- Para la Capa 2, el Web Server generará la lógica de presentación y el Transaction Processing Server permitirá la ejecución de las piezas de código que implementan la lógica de negocios y parte de la lógica de acceso a datos. El Web Server deberá poseer un motor generador de páginas Web en tiempo de ejecución. Además el Transaction Processing Server deberá permitir la administración de transacciones orientada a componentes y la ejecución de procesos.
- Para la capa 3, el Database Server permitirá la ejecución de procedimientos almacenados que implementarán la parte restante de la lógica de acceso a datos. Este Database Server deberá soportar un gran almacenamiento de datos, que permita implementar bases de datos relacionales.

Capítulo **1**

ESPECIFICACIONES DEL PROYECTO

Capítulo 1

1. ESPECIFICACIONES DEL PROYECTO

En este capítulo se describirá cuales son los objetivos, funciones y beneficios de la aplicación *Mis Reservasiones.com*.

1.1. Justificaciones

La idea de desarrollar *MisReservasiones.com* nació con la propuesta de mejorar el turismo en nuestro país, y con el objetivo de que nuestras empresas hoteleras obtengan un mejor reconocimiento, para que puedan competir en el mercado tanto Nacional, como Internacional.

Además, observamos que hoy en día, el medio que abre las puertas internacionales a cualquier empresa es la Internet, por esta razón hemos desarrollado este sistema que permite a los usuarios que navegar en nuestro WebSite desde cualquier parte del mundo, permitiéndoles conocer lugares de

interés de nuestro país, y a la vez promocionando nuestra reserva turística esperando despertarles la curiosidad de conocer las maravillas del Ecuador, aumentando la demanda de reservaciones de habitaciones en los diferentes hoteles de nuestro país. Cabe anotar que con esto obtendríamos beneficios adicionales de tipo económico y social, tanto para los Inversionistas que poseen cadenas hoteleras, restaurantes y/o salones para eventos, como para nuestra comunidad al generarse un ingreso de divisas adicional.

1.2. Objetivos Generales

Nuestro Sistema *MisReservaciones.com* fue desarrollado con la misión de permitir a nuestros clientes, que puedan proveer a sus usuarios un servicio cómodo para la reservación de habitaciones de Hoteles en un ambiente tranquilo y seguro, en el ámbito nacional desde cualquier lugar del mundo. Además mejorar sus demandas, convirtiéndolos en empresas competitivas a nivel internacional.

Entre los objetivos que persigue *MisReservaciones.com* mencionamos de manera general los más significativos:

Competitividad de nuestras empresas hoteleras en el mercado internacional. Gracias al Internet y a nuestro Website, las empresas cliente serán reconocidas en cualquier lugar del mundo donde haya acceso a la red.

Reconocimiento de Nuestra Reserva Turística. Nuestro WebSite permite conocer diferentes lugares de nuestro país y además, contiene enlaces a otras páginas que enseñan las bellezas de nuestra patria.

Facilidad para reservar en línea. El objetivo principal es permitir a los usuarios de nuestro WebSite realizar reservaciones en línea a habitaciones de hoteles, cafeterías y salones.

1.3. Especificaciones Funcionales

El diseño de la aplicación *Misreservaciones.com* le permite operar bajo distintos enfoques de gestión que se adaptarán a su modelo de negocio, todo ello aprovechando los últimos avances tecnológicos y de comunicaciones, el Sistema Web o Sistema on-line está dirigido a los usuarios de Internet y el Sistema Standalone que es una aplicación local, orientada a administradores del Sistema.

1.3.1. Funcionalidades del Sistema Web (Sistema on-line)

Entre las funcionalidades que permite realizar la aplicación Web de *MisReservaciones.com* está la de controlar la seguridad:

Registro de un nuevo usuario en el sistema. Permite a los navegadores de nuestro WebSite registrarse en el sistema, y así poder beneficiarse de las utilidades del mismo.

Ingreso al sistema. Cuando un usuario se registra el sistema le entrega un **login** o nombre de acceso y una **password** o clave que controla el acceso al sistema.

Cuando se permite el acceso al sistema a un usuario, este puede beneficiarse de funciones como:

Reservación en Línea de Habitaciones, Mesas y Salones. El sistema permite a los usuarios realizar reservaciones de habitaciones, salones y mesas de los clientes afiliados desde cualquier lugar de acceso.

Nuestro sistema permite también leer la información actualizada de ciertas tablas de nuestra base de datos por medio de consultas diversas.

Consulta de disponibilidad de Habitaciones, Mesas y Salones. El usuario puede confirmar a través de nuestra aplicación, si existe disponibilidad de habitaciones, salones o de la cafetería en los diferentes hoteles clientes.

Consulta de los hoteles en las diferentes ciudades. El usuario puede chequear las diferentes opciones de hoteles que se encuentre en la ciudad. Contando con una lista completa y de permanente actualización.

Consulta de tipos de habitaciones y de sus costos. El usuario puede consultar los costos de los distintos tipos de habitaciones de los hoteles clientes.

Además una vez realizada una reservación el sistema le permite:

Cancelación / Cambio de una reservación. Con esta opción, un usuario puede cancelar o cambiar la fecha de una reservación hasta 24 horas antes de la fecha inicial original.

1.3.2. Funcionalidades del Sistema Standalone

Entre las funcionalidades de la aplicación standalone de *MisReservaciones.com*, se encuentran:

Mantenimiento de las Tablas Maestras. Permite realizar ingreso, modificación y eliminación de registros, de todas las tablas maestras, por ejemplo tipos de clientes, categorías de hoteles, tipos de comida, etc.

Mantenimiento de Clientes. Permite llevar un control de nuestros clientes (hoteles) y de los usuarios de nuestro WebSite, permitiendo el ingreso, modificación o eliminación de los mismos.

Mantenimiento de las Seguridades del Sistema. Permite dar mantenimiento tanto para la aplicación Web como para la aplicación standalone.

Generación de Reportes varios. Permite generar reportes generales acerca de las reservaciones hechas, y de las principales tablas Maestras del Sistema.

1.4. Limitaciones y Restricciones

No permite el cobro de Facturas. El Sistema solo permite realizar reservaciones, no incluye el cobro de facturas o abono de reservaciones, eso solo lo realizará personalmente el usuario del servicio al hotel.

No permite la manipulación de datos por medio del Sistema Web. No permite realizar algún tipo de cambio a las tablas maestras por medio del sistema web, es decir, no se podrá eliminar o modificar la información, los únicos movimientos posibles serán: consultas, reservaciones, cancelaciones, modificaciones de reservaciones e ingreso de nuevos usuarios (explicado en el punto 1.3).

No permite realizar reservaciones por medio del Sistema Standalone. El sistema standalone no puede realizar una reservación, éstas solo pueden realizarse por medio del Sistema Web.

1.5. Alcances

Seguridad

El sistema mantiene la integridad de los datos de la aplicación Web, permitiendo que estos únicamente puedan manipularse a través del Sistema Standalone mediante los perfiles de usuario. Los perfiles permiten al usuario del sistema (según las restricciones y permisos) acceder a la información necesaria.

Procesos

El sistema *MisReservaciones.com* permite realizar los procesos de reservaciones de habitaciones de manera optimizada para las empresas, brindando facilidades para sus clientes.

Permite realizar consultas y generar reportes facilitando la toma de decisiones, además de dar mantenimientos a la base de datos.

Capítulo **2**

ANÁLISIS DE LA INDUSTRIA Y EL PRODUCTO

Capítulo 2

2. ANÁLISIS DE LA INDUSTRIA Y EL PRODUCTO

A continuación se dará una completa descripción acerca de la definición de la industria y el producto.

2.1. Definición de la industria y el producto

En la actualidad, un medio importante para el desarrollo empresarial es el Internet tanto para competir con otras empresas nacionales como también para introducirse al mercado internacional. Por esta razón es importante que las empresas brinden a sus clientes la facilidad de realizar cualquier tipo de transacción por este medio, que cada vez se incrementa. Tomando en cuenta esta idea hemos desarrollado el sistema *MisReservaciones.com* que como indicamos en el Capítulo 1 se orienta a realizar reservaciones on-line.

Existen sistemas que pueden realizar reservaciones de habitación y otros servicios pero no brindan los beneficios (explicados en el Capítulo 1) que nuestro sistema es capaz de entregar a sus clientes.

2.2. Clientes

Nuestro sistema está orientado hacia hoteles que deseen aumentar sus demandas de habitaciones, salones, cafeterías pero también a los salones y cafeterías independientes.

2.2.1. Comportamiento del consumidor ante el mercado convencional

Las reservaciones tradicionalmente eran realizadas por medio telefónico o personalmente, pero debido a múltiples factores (congestión, desinterés de parte de los encargados, limitante en cuanto a líneas de atención a los clientes, etc.), era muy difícil aumentar la demanda.

Los consumidores o clientes de este servicio que realizan las reservaciones por teléfono lo consideran como un costo adicional para ellos, que el hotel no cubre, peor aún si la reserva proviene del exterior.

Los clientes que realizan sus reservaciones personalmente corren el riesgo de no encontrar habitaciones disponibles al llegar a su hotel de preferencia, o de tener que conformarse con el tipo de habitación que estuviese disponible.

2.2.2. Reacciones del consumidor ante el Mercado Virtual

El hecho de poder realizar la reservación de habitaciones desde su hogar u oficina en cualquier punto geográfico, es una ventaja que le ahorra tiempo y dinero a los usuarios, siendo lo más importante la seguridad de que sus reservaciones estén confirmadas.

En cuanto a los hoteles, esta ventaja incrementará sus utilidades teniendo mayor número de clientes, dándole un mejor servicio, además de optimizar sus procesos.

2.2.3. Mercado Virtual vs. Mercado Convencional

Ventajas del Mercado Virtual sobre el Mercado Convencional

- El Mercado Virtual permite a la organización aumentar su alcance o expandir sus horizontes tanto nacional e internacionalmente. Con un mínimo de capital en comparación a la alternativa convencional, las empresas pueden fácil y rápidamente llegar a más clientes, conseguir los mejores proveedores y los socios estratégicos que necesitan sin importar su ubicación.
- Además de lo mencionado, el Mercado Virtual permite personalizar los servicios con un mínimo de costos e innovar constantemente el modelo de negocios para tener siempre ventajas competitivas con buenas utilidades.

- Otra de sus ventajas es reducir los costos de tener que crear una red privada, ya que emplear la Internet es lo más barato para llegar a todo el mundo.
- Permite a las organizaciones atender a los clientes las 24 horas del día.
- Permite reducir costos y gastos de transporte, incrementar el acceso a información y simplificar procesos.
- Ayuda a reducir los tiempos, a eliminar las actividades que no crean valor, a reducir costos en general, en resumen a hacer más productiva a la organización.

Desventajas del Mercado virtual ante el Mercado Convencional

- Los costos de implementar Mercado Virtual pueden ser muy altos y se pueden cometer errores debido a la falta de experiencia, lo que puede ocasionar retardos o implementaciones inconclusas. Hay algunas oportunidades para "outsourcing" dentro del mercado virtual, el problema es saber detectar dónde y cómo aplicarlo.
- Existen clientes que desconfían de efectuar transacciones con alguien que no conocen, sin papeles y con dinero electrónico.
- Existen clientes que prefieren realizar sus transacciones personalmente, para tener mayor conocimiento y seguridad del servicio.

- Algunos temas legales y regulaciones no están todavía resueltos para considerar todo lo que puede ocurrir dentro del comercio electrónico.
- Existe el temor de que se elimine la interacción social entre personas con el comercio electrónico, lo que a la larga puede perjudicar las relaciones humanas.
- Como una nueva disciplina y forma de hacer negocios, el Comercio Electrónico está evolucionando rápidamente. Algunas personas esperan que se “estabilice” para poder ingresar en el mercado electrónico.
- El acceso a la Internet es todavía caro para algunos y completamente inaccesible para otros.
- En algunos países o áreas, no hay una cantidad importante de clientes con deseos de participar del Comercio Electrónico como para que éste pueda ser exitoso, por ende, aún no existen vendedores y compradores que animen a las organizaciones a participar y obtener utilidades de los mercados electrónicos.

2.3. Competidores

Entre nuestros competidores virtuales, existen muchas empresas que brindan este servicio pero son empresas internacionales que tal como nosotros se dedican a brindar el servicio solo a los hoteles de sus respectivos países. Unisoft es pionero en ofrecer este tipo de servicio en el país.

El mayor de nuestros competidores convencionales, es el teléfono que por tradición está fuertemente ubicado en la mente de los consumidores de este servicio, pero posee las desventajas mencionadas ya en un capítulo anterior.

2.3.1. Funcionalidad de los Competidores Convencionales Existentes

Los competidores convencionales como dijimos anteriormente son el Sistema de Reservación Telefónica y Reservación personal.

Funcionalidades de la Reservación Telefónica

- Permite realizar la reservación directamente.
- Permite recibir una confirmación inmediata.
- Brinda seguridad de confirmación al usuario.
- Tiempo de respuesta inmediato.

Funcionalidades de la Reservación Personal

- Permite realizar la reservación directamente.
- Permite recibir una confirmación inmediata.
- Brinda seguridad de confirmación al usuario.
- Permite conocer las instalaciones de los hoteles.
- Permite realizar una selección de las diferentes opciones de habitaciones.
- Permite el cobro inmediato.

- Brinda a los hoteles, mayor seguridad que la reservación se cumpla.
- Tiempo de respuesta inmediato.

2.4. Productos Sustitutos

Los productos que pueden sustituir las funciones del sistema, como lo hemos mencionado a lo largo del capítulo son: la reservación telefónica y la reservación personal. Además en el mercado existen otros sistemas que cumple con estas funcionalidades, tales como los servicios de páginas web personalizados de cada empresa hotelera y de servicio.

2.5. Análisis de FODA del Sistema

FORTALEZAS	OPORTUNIDADES
▪ Reconocimiento Internacional	Gracias al Sistema de Reservaciones, las empresas participantes podrán abrirse campo tanto en el ámbito local como internacional, brindando una cartera de clientes muy variada.
▪ Publicidad	Internet tiene una gran cantidad de usuarios, que conocerán por medio de banners publicitarios nuestro sistema y por lo tanto a los hoteles.

▪ Facilidad para usuarios	No hace falta que el cliente del hotel se encuentre dentro de la ciudad para hacer sus reservas, ni requerirá de una costosa llamada telefónica.
▪ Maximización de Utilidades	La empresa tendrá una mayor utilidad, al ver disminuidos los gastos de empleados para la atención personal.

DEBILIDADES AMENAZAS	
▪ Tiempo de Respuesta	Molestias a los usuarios porque la red puede estar muy lenta.
▪ Reservas con un día de anticipación	Perder clientes que deseen realizar reservas inmediatas (para el mismo día).
▪ No permite el cobro de facturas	Tener que realizar otros procesos tanto el hotel como sus clientes.
▪ Cumplimiento de Reservación	Brindando poca seguridad para los hoteles, en cuanto a que la reservación se cumpla.
▪ Seguridad	El sistema marcará como reservada una habitación, sin la seguridad que la misma se cumpla, pudiendo perder a un cliente seguro que desee hacer una reservación personal, por no tener habitaciones disponibles.

Capítulo **3**

MODELO DE NEGOCIOS

Capítulo 3

3. MODELO DE NEGOCIOS

En este capítulo se dará una detallada explicación acerca de los procedimientos y funciones del negocio.

3.1. Definición del Modelo de Negocios

La idea de este sistema se basa en brindar a las Empresas Hoteleras y Restaurantes el servicio de reservaciones en línea de habitaciones, mesas y salones utilizando para ello las ventajas del Internet.

Para llegar a nuestros clientes, las Empresas Hoteleras y Restaurantes, realizaremos una presentación del sistema en cada una de sus sucursales principales de estas organizaciones, demostrando los beneficios y utilidades que obtendrían con este sistema. Además los interesados podrán obtener una versión demo del sistema como prueba por dos semanas. Si decidieran

comprar se implementarían las aplicaciones del Sistema standalone en sus instalaciones de cómputo.

Los navegadores de la red podrán enterarse y conocer nuestro Sistema on-line por medio de publicidades en banners.

3.2. Funcionalidad General

Su sistema de reservaciones en línea puede estar **operando en 48 horas**.

Reducción en sus costos administrativos. por atención a clientes por teléfono para confirmar disponibilidades, reservaciones, envío de faxes, etc.

Seguridad del manejo de Información. Únicamente usted como administrador del Sistema standalone podrá acceder para colocar inventarios, cambiar precios, bloquear temporadas. Recoger las reservaciones en su mail.

Su sitio web disponible a todo el mundo a través de la red de Internet, su sistema de reservaciones disponible las 24 horas del día, los 365 días del año en cualquier parte del mundo.

Administración y manejo de su información. Toda la información proveniente de sus reservaciones, será enviada hacia el computador central de nuestros clientes afiliados vía dial-up, para sus propósitos de mercadeo, estadísticas y seguimiento a clientes frecuentes para futuras promociones etc.

Para el manejo del Sistema no necesita ser experto en web, programación ni nada por el estilo, es fácil de manejar y acceder.

3.2.1. Diagrama de Flujo de Procesos

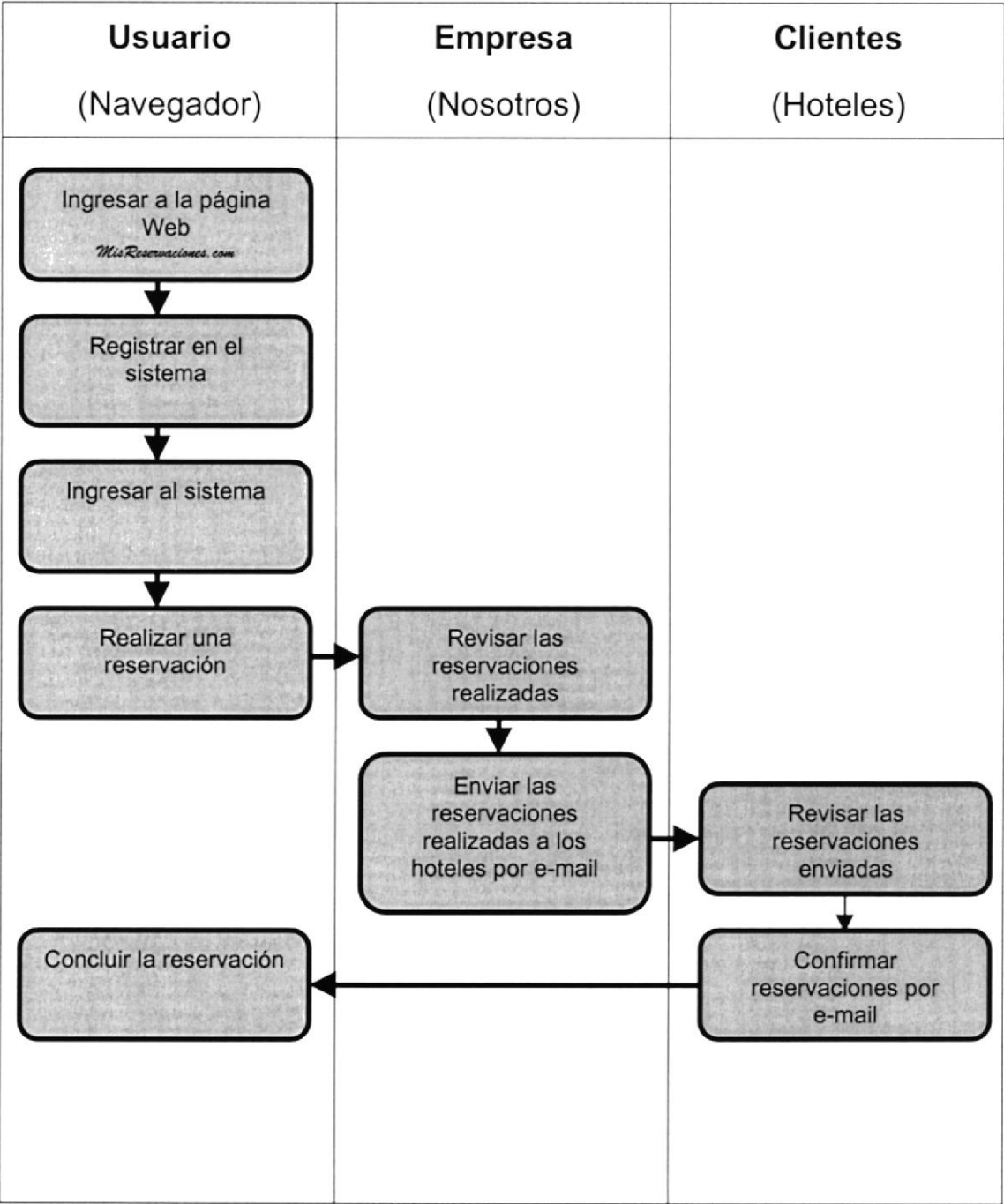


Tabla No. 3.1. Diagrama de Flujo de Procesos

3.3. Formas de Pagos

Nuestra empresa ofrecerá este servicio a los hoteles por un costo de implementación del sistema de \$ 800 esto es la instalación de la aplicación standalone y la afiliación al sistema on-line.

Sin tomar en cuenta el monto de sus ganancias provenientes de nuestro sistema *MisReservaciones.com* a nuestros clientes, el costo de este servicio (una vez instalado el sistema) será tan solo de \$80.00 mensuales, los cuales serán acreditados a nuestra cuenta bancaria por los clientes afiliados, todos los primeros días de cada mes.

Como puede observar, el costo mensual es bajo y se paga con tan solo una noche de alojamiento reservada por nuestro sistema. (Por ejemplo, si el precio de una noche de hotel es igual a \$80.00, con una noche reservada esta cubierta su inversión mensual). Usualmente todo viajero reserva como mínimo de 2 a 3 noches de alojamiento. Cuántas noches de alojamiento confirma su hotel al mes? Imagínese y todo eso por tan solo \$80.00/mes; es bastante económico.

3.4. Seguridad

Cobros de Tarifas. Por brindar mayor seguridad a los usuarios del Website y por ser un sistema de reservaciones, y no de facturación, *MisReservaciones.com* no permite el cobro de la reservación a los usuarios,

esto es, que no existe ningún riesgo para los clientes de los hoteles, ellos tienen que hacer su cancelación personalmente.

Acatamiento de las Reservaciones. El hotel corre el riesgo que la reservación no sea ejecutada, para esto nuestro sistema no permite a cualquier navegador realizar una reservación, este usuario que quiera beneficiarse con nuestro servicio deberá registrar sus datos al sistema. Además una vez hecha una reservación al hotel le corresponderá confirmar dicha reservación al e-mail del cliente.

Capítulo **4**

ARQUITECTURA DEL SISTEMA

Capítulo 4

4. ARQUITECTURA DEL SISTEMA

Este capítulo contiene una breve explicación acerca de la arquitectura del sistema de información *Mis Reservasiones*. Como introducción se puede mencionar que este sistema fue desarrollado con arquitectura Cliente/Servidor, haciendo uso del modelo de 3 capas. En este modelo se implementaron 3 diferentes tipos de lógica:

- Lógica de Presentación.
- Lógica de Negocios.
- Lógica de Acceso a Datos.

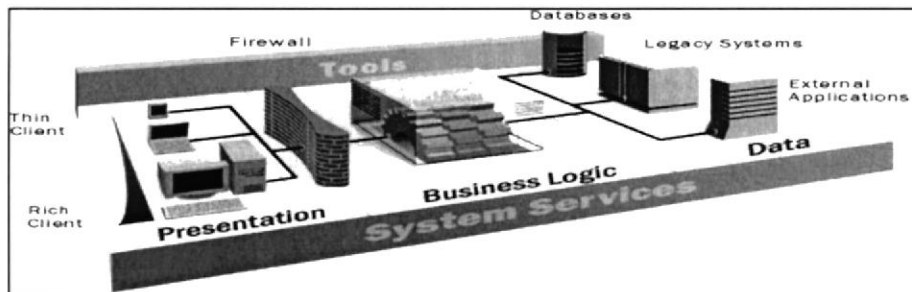


Figura No. 4.1. Tipos de Lógicas

Lógica de Presentación:

El proceso cliente que implementa la lógica de presentación presenta dos alternativas para la interacción con el Usuario Final, es decir, para nuestra aplicación existen dos tipos de Procesos Clientes implementados.

Es una aplicación independiente o StandAlone desarrollada en Microsoft Visual Basic 6.0.

Es un proceso contenedor como es el caso del Web Browser, el mismo que permitirá la ejecución de la aplicación web.

Lógica de Negocios:

Para este sistema la lógica de negocios es sumamente importante, debido a que en ella vamos a almacenar todas las reglas del negocio, es decir las políticas de la empresa. Para el cumplimiento de esta lógica se implementaron en componentes en proceso (dll) estas políticas.

Lógica de Acceso a Datos:

En esta lógica se implementaron componentes en proceso (dll) el acceso a datos y la invocación de los procedimientos almacenados que accedan a las tablas de la Base de Datos de Mis Reservaciones.

4.1. Justificación de la Selección del Modelo

Los desarrolladores de Mis Reservas hemos acordado utilizar el modelo descrito anteriormente por sus múltiples ventajas tales como:

- El uso de los procedimientos almacenados o stored procedures, además de facilitarnos el mantenimiento de las tablas, nos brinda un mejor tiempo de respuesta y una mejor organización de nuestro trabajo.
- Nuestro modelo al estar estructurado en procedimientos almacenados nos permite dividir el peso de la aplicación a nivel de la Capa de Datos, y esto a su vez nos ayuda a que la escalabilidad de la aplicación no se vea afectada. Logrando con ello que nuestra aplicación permita manejar un gran número de usuarios y no se vea afectada la escalabilidad ni el rendimiento.
- Contamos con el mejor Nivel de Seguridad, ya que nuestra estructura tanto a Nivel de Acceso a Datos y Reglas del Negocio no es accesada directamente por la Aplicación Cliente, de esta forma tanto la integridad de los datos como las políticas de la empresa se mantienen encapsuladas.
- Los procesos transaccionales son compartidos entre múltiples procesos clientes, es decir que múltiples usuarios pueden llamar al mismo componente, y acceder por ejemplo al mismo procedimiento almacenado.

- Otra ventaja es sin duda el balanceo de la carga de trabajo, por ejemplo los múltiples componentes pueden ser colocados en diferentes equipos, y con esto disminuir la carga, la centralización, y mejorando enormemente la confiabilidad y la disponibilidad de su uso. Solamente que esta disponibilidad puede verse afectada únicamente por la infraestructura de comunicaciones que pueda existir entre el proceso cliente y los procesos servidores que administran la aplicación.
- El tener un sistema de conexión a la Base de Datos resulta muy ventajoso, es decir conectarse lo más tarde posible, y liberar la conexión lo antes pronto posible, solucionamos el problema en cuánto a limitación de conexiones de acceso.
- Por último, una ventaja para los desarrolladores es que podemos realizar un fácil desarrollo, mantenimiento e implantación de la aplicación sin conocer muchas veces al 100% como está estructurado el Sistema, de manera que fácilmente podemos delegar a otro grupo de programadores para que continúen con su desarrollo, sin que las reglas y muchos detalles de la empresa y del sistema sean revelados.

4.2. Tipos de Procesos Clientes

En nuestra aplicación nosotros contamos con dos tipos de Procesos Clientes que implementan la Capa de Presentación y utilizan los mismos objetos que implementan la capa de acceso a datos y de negocio. Estos son:

- **Un Cliente Independiente:** Este cliente es una aplicación StandAlone, desarrollada en Microsoft Visual Basic 6.0. con el fin exclusivo de llevar el mantenimiento a las tablas maestras que se están utilizando en el Sistema de Mis Reservas. Además de permitirles a nuestros Clientes afiliados consultar todas las reservas realizadas por los usuarios web a sus servicios.



Figura No. 4.2. Pantalla del Cliente Independiente

- **Un Cliente Dependiente:** Este cliente es una aplicación Web que se ejecuta en un contenedor, que es un Browser, además que servirá como un portal o sitio web que estará abierto al público las 24 horas del día y los 365 días del año, en donde él podrá efectuar cualquier tipo de reservas que necesite.



Figura No. 4.3. Pantalla del Cliente Dependiente

4.3. Tipos de Procesos Servidores

En cuanto a procesos servidores, la aplicación tanto StandAlone como web interactúan según el caso con 3 procesos servidores, entre los cuales tenemos:

Proceso Servidor Web

El proceso servidor web es parte fundamental dentro de la arquitectura del sistema. El Web Browser es el proceso cliente que envía requerimientos HTTP al proceso servidor que es el Web Server. La solicitud que éste recibe es por una página estática HTML o por un programa que genere

dinámicamente el código HTML que posteriormente será enviado al browser. El producto utilizado para el desarrollo de la aplicación web y posteriormente para poner en producción al sistema es el Microsoft Internet Information Server (IIS), para más características técnicas véase Apéndice E.

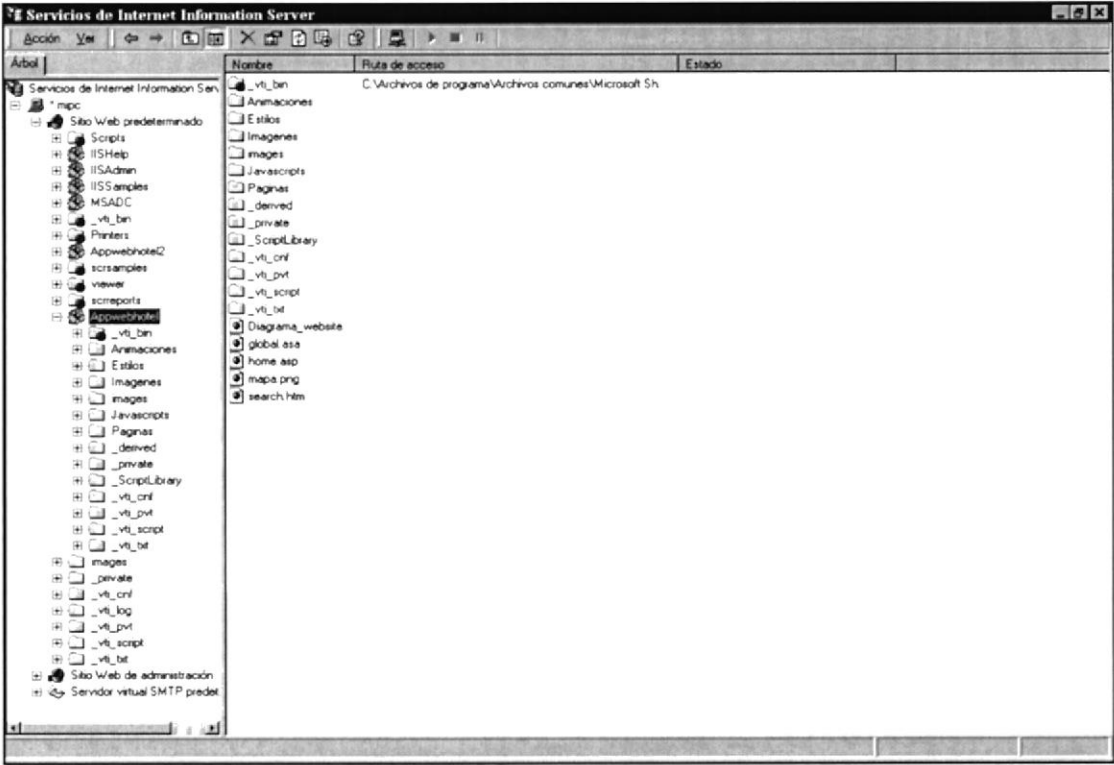


Figura No. 4.4. Consola del Microsoft Internet Information Server

Proceso Servidor de Transacciones:

El proceso servidor de transacciones es el encargado de administrar las transacciones y la seguridad en la Capa 2. Además maneja el pool de conexiones al Servidor de Base de datos y maneja threads agrupando los componentes en diferentes tipos de procesos. El producto utilizado para la administración y seguridad para los componentes de acceso a datos y de

negocios es el Microsoft Transaction Server (MTS), para más información véase Apéndice D.

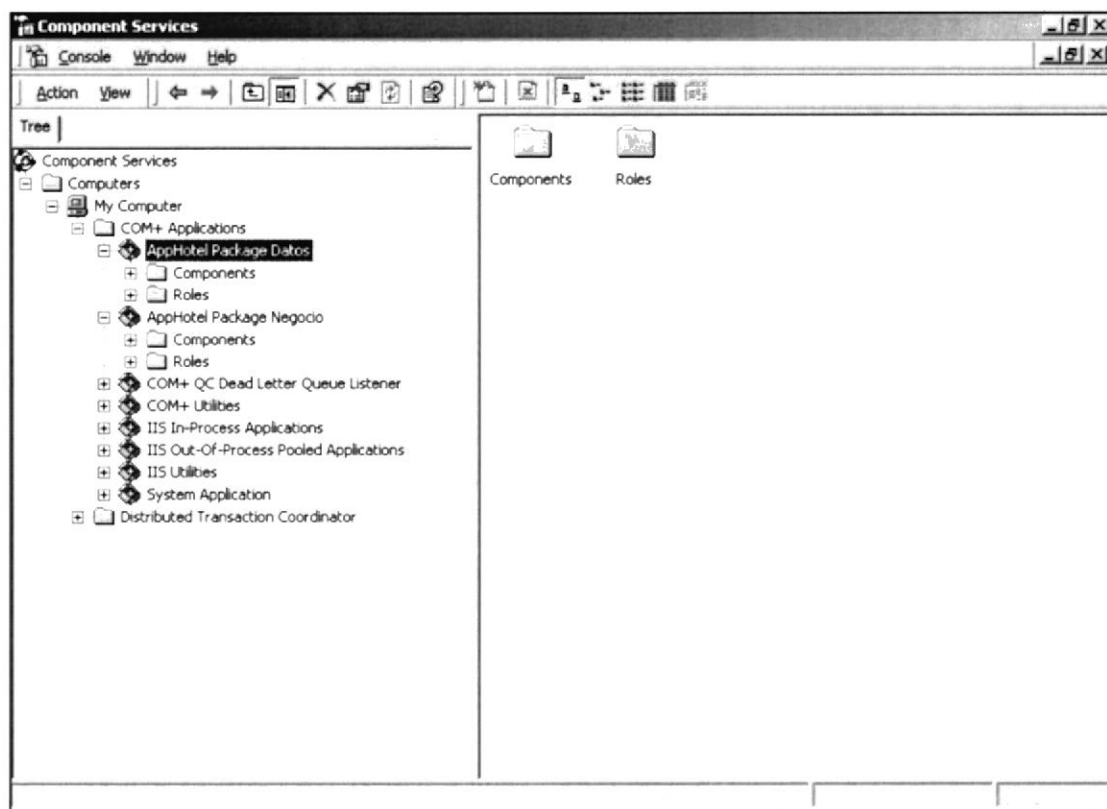


Figura No. 4.5. Consola del Microsoft Transaction Server

Proceso Servidor de Bases de datos:

El Proceso servidor de Bases de Datos es quién finalmente administra la base de datos de la aplicación permitiendo el acceso a los componentes de datos. Además contiene procedimientos almacenados que están escritos en lenguaje SQL que se encuentran precompilados y almacenados junto con los datos en la misma base, y estos componentes permite albergar la parte restante de la lógica de acceso a datos. Para la implementación de la base de datos de nuestra aplicación se utilizó como producto Microsoft SQL Server 7.0.

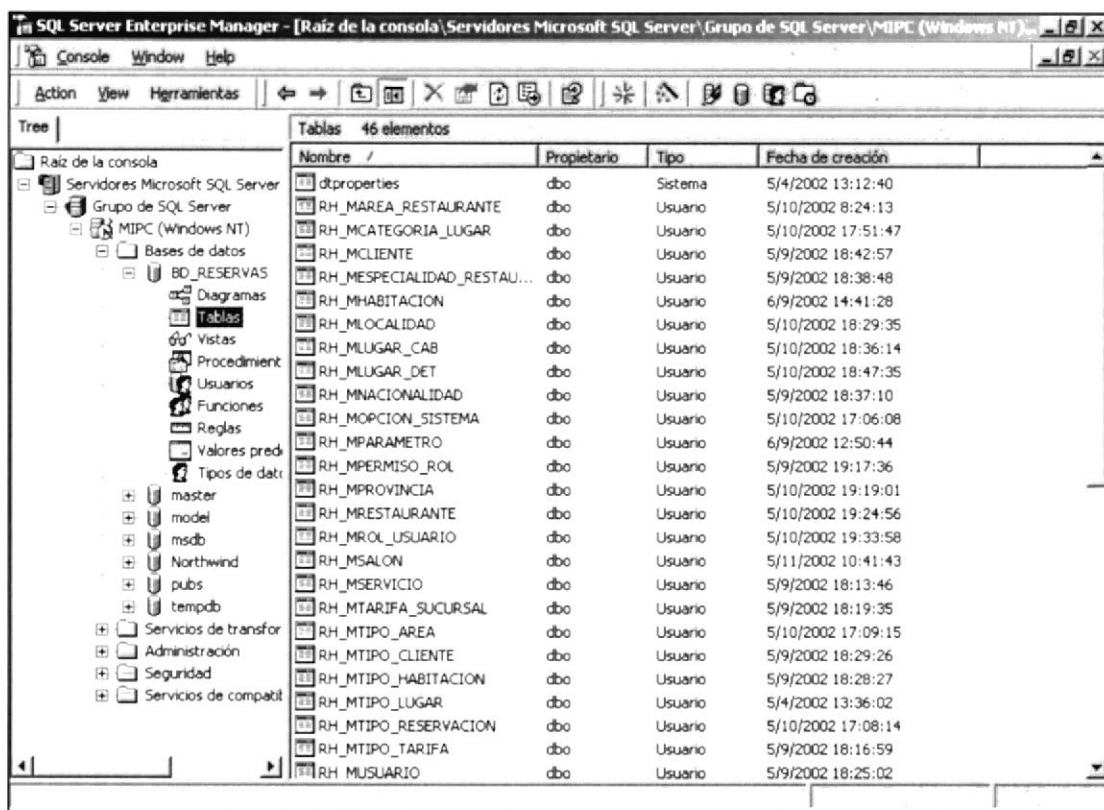


Figura No. 4.6. Consola del Administrador del SQL Server

4.4. Tipos de Middleware

Middleware es un software de conectividad, es decir, es todo el software que se encuentra entre el proceso cliente y el proceso servidor, además es el que soporta la interacción entre ambos, es decir su comunicación. Nuestro sistema necesita de tres tipos de middleware para su correcto funcionamiento: HTTP, COM/DCOM y COM+ y ADO/OLEDB

Hypertext Transfer Protocol (HTTP):

El primer tipo de middleware es HTTP, es un protocolo a nivel de capa de aplicación dentro del modelo OSI, empleado para la interacción entre el Web browser y el Web Server. El usuario del sistema debe emplear un browser

para invocar al Web Server que es el encargado de genera dinámicamente el código HTML que implementa la lógica de presentación.

COM / DCOM y COM+:

El segundo tipo de middleware es COM/DCOM y COM+, que es usado para la comunicación entre el proceso que se encarga de generar dinámicamente el código HTML que es enviado al Browser y los componentes de negocios que corren dentro del entorno del MTS, también es empleado para la comunicación entre los componentes de negocios y los componentes de datos, para más información véase Apéndice C.

ActiveX Data Objects (ADO):

El tercer tipo de middleware empleado por el sistema es ADO/OLEDB como el API para acceso a la Base de Datos. ADO es un API fácil de usar e independiente del lenguaje de programación. A través de está API nos permite almacenar y recuperar datos de una gran variedad de proveedores de datos y mejorar el tiempo de respuesta de la aplicación.

4.5. Tecnologías y Herramientas Escogidas para la Implantación

Para la construcción del sistema de información se utilizaron las siguientes tecnologías:

- COM+, para la administración de los componentes.

- ASP, para la creación de páginas web dinámicas.

En cambio como productos o herramienta que nos permitieron la construcción de ambas aplicaciones tenemos las siguientes:

Desarrollo de Cliente Independiente, componentes de datos y de negocios:

- Herramientas del paquete de Microsoft Visual Studio 6.0
 - Microsoft Visual Basic 6.0:
 - Microsoft Visual C++ 6.0
- GuidGen, para la asignación de códigos identificadores únicos.
- Compilador MIDL, para la generación de las .tlb.

Desarrollo de Cliente Dependiente:

- Herramientas del paquete de Microsoft Visual Studio 6.0
 - Microsoft Visual InterDev 6.0:
- Productos Macromedia.
 - Macromedia Dreamweaver 4.0:
 - Macromedia Flash 4.0.

Implementación de Procesos Servidores:

- Microsoft Component Services (COM+), como proceso servidor transaccional.
- Microsoft Internet Information Server, como proceso servidor web.
- Microsoft SQL Server 7.0, como proceso servidor de Base de datos.

Web Browser

- Microsoft Internet Explorer 6.0

Sistema Operativo

- Microsoft Windows 2000 Server.

4.6. Requerimientos de Hardware y Software para la puesta en producción

Entre los requerimientos como mínimo recomendados para que las aplicaciones tanto web como standalone puedan ser soportadas para su correcta ejecución y puesta en producción tenemos los siguientes:

Requerimientos de Hardware.

Descripción	Características
 1 Servidor	Sun Ultra Enterprise 450 Procesador Ultra Sparc II (2) de 296 Mhz. 512 Mb de Memoria RAM 2 Mb de Memoria Cache 45Gb (5 discos de 9 Gb., Ultra SCSI III, 4 unidades X canal) Nivel RAID 0+1 STRIPPED Velocidad de Acceso 40Mb/seg Este servidor se lo utilizará como Web Server, DataBase Server, Transaction Server.
 Computador para Usuarios	Procesador Pentium 4, 1.6 Ghz Memoria 256 Mb. Disco Duro 40 Gb. CD ROM 52x Floppy disk 1.44 Tarjeta de Red 10/100 Mbps Un usuario que utilice cualquiera de las dos aplicaciones su PC deberá contar con estas características.

 1 Router	• Marca: Motorola Tipo: Vanguard 320 Procesador Motorola 68360/25 MHz Memoria Flash 2MB Memoria RAM 8MB Puertos: 1 Ethernet 10BaseT
 1 HUB	• Marca: 3COM Modelo: Super Stack II Baseline Dual Speed HUB Velocidad 10/100 Mh. 24 puertos
 1 Módem	• Marca: Tellabs Tipo: Martis DXX STU-160 Velocidad: 64 Kpbs

Tabla No. 4.1. Requerimientos de Hardware

Requerimientos de Software.

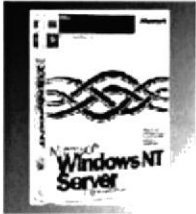
Descripción	Características
	• Paquete de productos de Microsoft Visual Studio 6.0
	• Microsoft Windows 2000 Server
	• Microsoft SQL Server 7.0
	• Macromedia Dreamweaver

Tabla No. 4.2. Requerimientos de Software

Capítulo **5**

IMPLEMENTACIÓN DE LA CAPA DE DATOS

Capítulo 5

5. IMPLEMENTACIÓN DE LA CAPA DE DATOS

Este capítulo contiene una explicación acerca de la implementación de la capa de datos, presentando para ello modelos a nivel de datos y descripción de los procedimientos almacenados desarrollados.

5.1. Modelo Lógico de Datos.

El Modelo Lógico de Datos diseñado para el sistema de Mis Reservas muestra la relación entre las entidades de datos, las mismas que se encuentran identificadas con nombres lógicos.

A continuación mostramos este modelo:

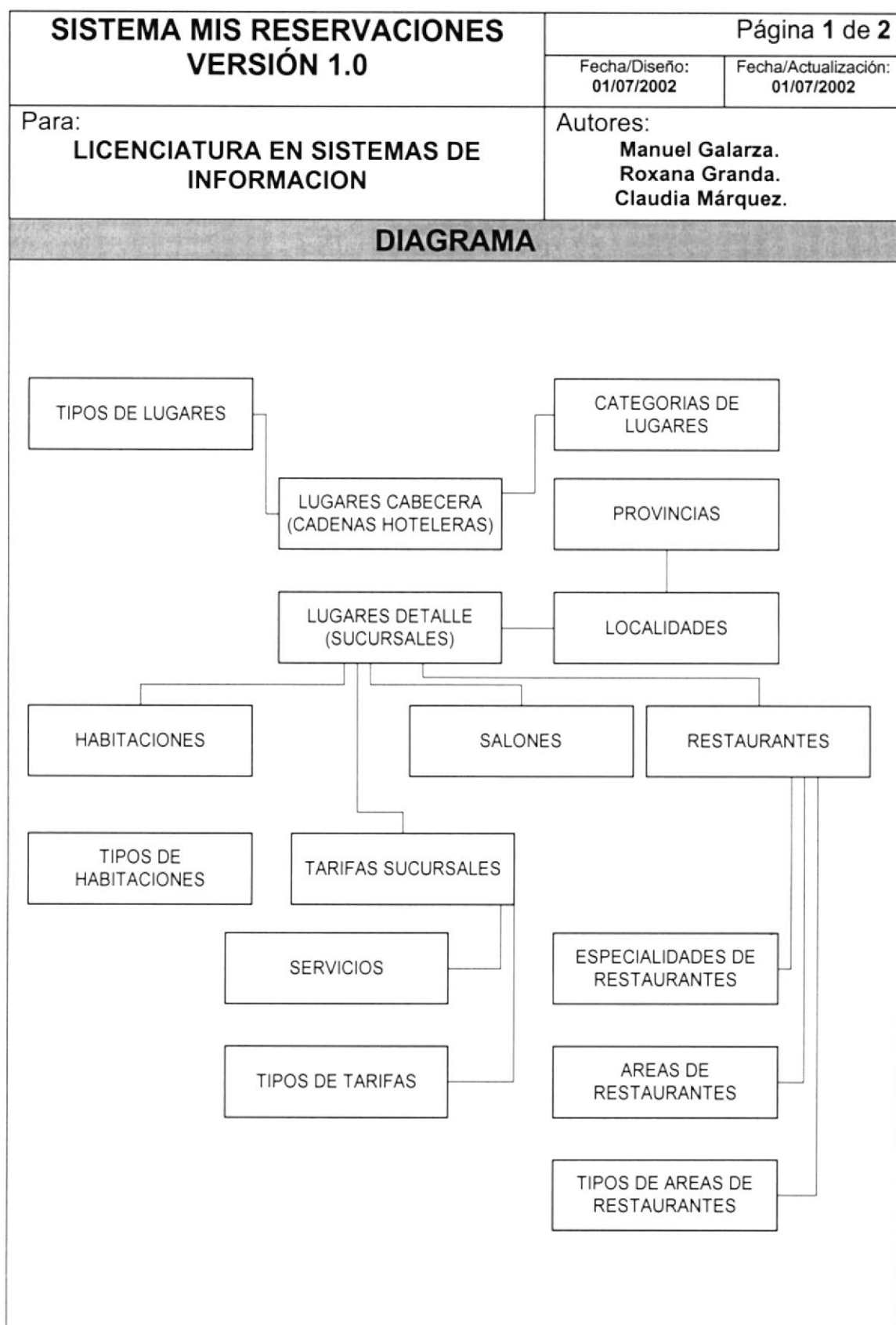


Tabla No. 5. 1. Diagrama del Modelo Lógico de Datos.

Explicación:

En esta primera parte del Modelo Lógico de datos se muestra como primera relación a la Entidad Cadena Hotelera (Lugar Cabecera), la misma que posee relaciones con Categoría de Lugar, Tipo de Lugar y a su vez un detalle de Sucursales (Lugar Detalle).

La Entidad Lugar Detalle posee solo relaciones con Localidad debido a que sólo se inserta en esta tabla las sucursales que existan en una determinada ciudad. Cada registros de la Entidad Lugar Detalle esta asociado a las Entidades Habitaciones, Restaurante, Salones de eventos y a las Tarifas de Sucursales en donde se registran los precios que el usuario tiene que cancelar por una reservación.

La Entidad Habitaciones que contiene el registro de las habitaciones que existe en cada sucursal, a su vez posee relaciones con la Entidad Tipos de Habitaciones.

La Entidad Tarifas Sucursales que contiene los precios a cobrar por los servicios que prestan en cada sucursal, posee relaciones con la Entidad Servicios y la Entidad Tipos de Tarifas.

La Entidad Restaurante contiene el registro de los restaurantes que posee un determinado Lugar Detalle, este a su vez posee relaciones con la Entidad Areas del Restaurante y Especialidad Gastronómica.

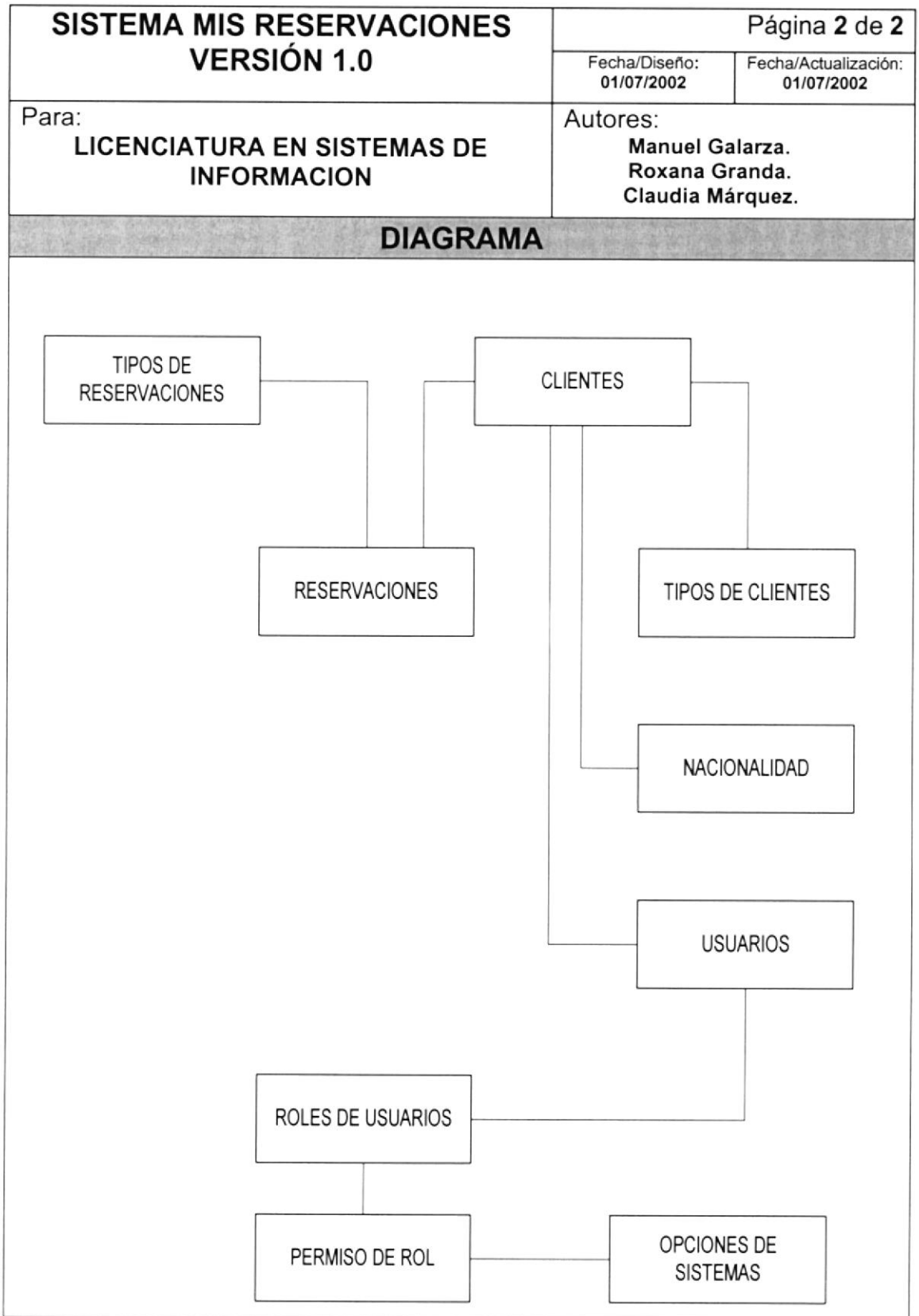


Tabla No. 5. 2. Diagrama del Modelo Lógico de Datos.

Explicación:

En esta segunda parte del Modelo Lógico de datos se muestra las siguientes relaciones de entidades:

La Entidad Reservas que contiene registro de las reservas registradas en el Sistema, posee relaciones con la Entidad Cliente, Tipos de Reservas.

La Entidad Cliente que contiene la información de los usuarios que realizan las reservas, posee relaciones con la Entidad Nacionalidad, Tipos de Clientes y Usuarios.

La Entidad Usuarios, que contiene información de los usuarios que pueden ingresar al sistema, posee relaciones con la Entidad Roles de Usuarios, la misma a su vez con la Entidad Permisos por Rol de las diferentes opciones del sistema que un usuario puede acceder.

5.2. Modelo Físico de Datos.

El Modelo Físico de Datos diseñado para el sistema de Mis Reservas muestra la relación entre las entidades de datos, las mismas que se identifican con nombres físicos, es decir como se encuentran identificadas a nivel del Database Server.

5.2.1. Formatos para el Almacenamiento de Información.

NOMBRE DE LA BASE DE DATOS	
Descripción: La estandarización del nombre de la base de datos es una identificación a nivel del DataBase Server de nuestra repositorio de datos.	
BD_XXXXXXXXXXXXX.....	
Ejemplo:	Nombre descriptivo.
BD_RESERVAS	

Tabla No. 5. 3. Estandarización de la Base de Datos.

NOMBRE DE TABLAS	
Descripción: La estandarización de los nombres de las tablas nos permitira identificar fácilmente que tipo de tabla es y que registros contiene	
XX_ XXXXXXXXXX.....	
	Identificador Descriptivo en Singular.
	Tipo de Tabla.
	Abreviación del Nombre del Módulo.
Ejemplo:	
RH_MRESERVACION	

Tabla No. 5. 4. Estandarización de las Tablas.

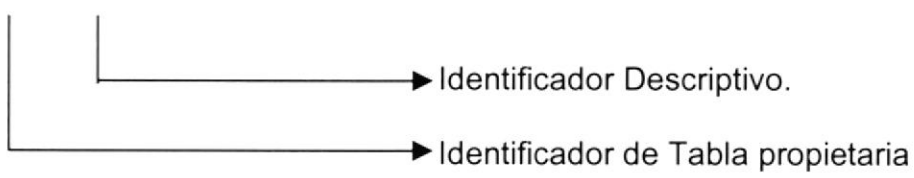
NOMBRE DE CAMPOS	
Descripción:	
La estandarización de los nombres de los campos nos permitira identificar fácilmente el contenido que guarda el campo y a que tabla pertenece	
XXXX_XXXXXXXXX..... 	
Ejemplo:	
PROV_CODIGO	

Tabla No. 5. 5. Estandarización de los campos.

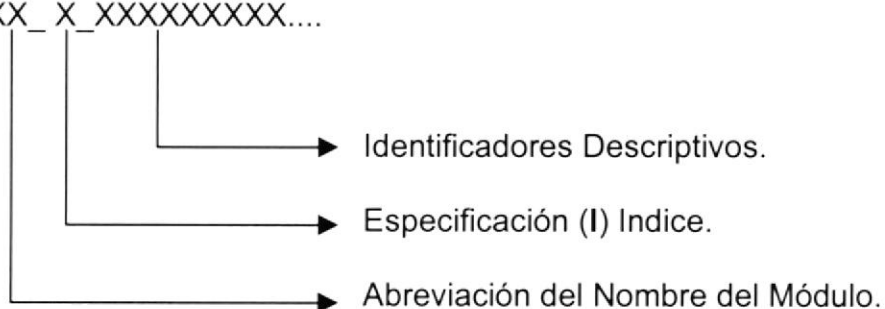
NOMBRE DE INDICES	
Descripción:	
La estandarización de los nombres de los índices nos permitira identificar fácilmente el la tabla y el campo que se encuentra indexado.	
XX_ X_XXXXXXXXX.... 	
Ejemplo:	
RH_I_CLIENTE	

Tabla No. 5. 6. Estandarización de los índices.

NOMBRE DE PROCEDIMIENTOS ALMACENADOS	
Descripción: La estandarización de los nombres de los procedimientos almacenados nos permitira identificar fácilmente que actividad realizan cuando se los ejecute.	
XX _XXX_ XXXXXXXX..... Nombre descriptivo. Tipo de Transacción que ejecuta. Stored Procedure (SP) Ejemplo: SP_IU_USUARIO	

Tabla No. 5. 7. Estandarización de los procedimientos almacenados.

5.2.2. Diagrama del Modelo Físico de Datos.

Una vez realizado una breve explicación de la estandarización utiliza para los nombre de los objetos de la Base de Datos a continuación le mostramos el diagrama del Modelo Físico de Datos.

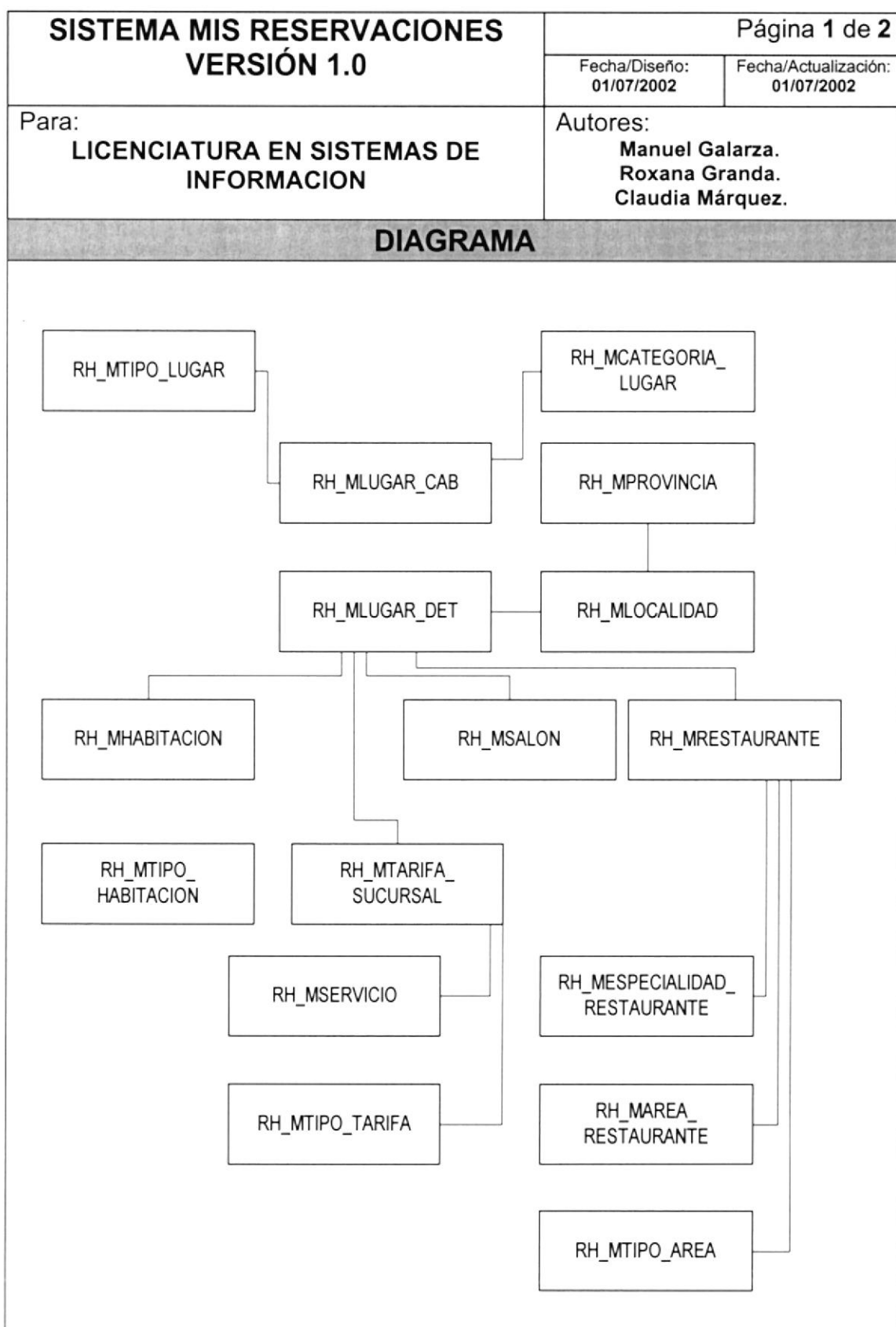


Tabla No. 5. 8. Diagrama del Modelo Físico de Datos.

Explicación:

En esta primera parte del Modelo Físico de Datos se muestra como primera relación a la Entidad RH_MLUGAR_CAB (Cadena Hotelera), la misma que posee relaciones con RH_MCATEGORIA_LUGAR, RH_MTIPO_LUGAR y a su vez con la Entidad RH_MLUGAR_DET (Detalle de Sucursales).

La Entidad RH_MLUGAR_DET posee solo relaciones con RH_MLOCALIDAD (Localidad) debido a que sólo se inserta en esta tabla las sucursales que existan en una determinada ciudad. Cada registro de la Entidad RH_MLUGAR_DET está asociado a las Entidades RH_MHABITACION, RH_MRESTAURANTE, RH_MSALON y a las RH_MTARIFA_SUCURSAL (Tarifas de Sucursales) en donde se registran los precios que el usuario tiene que cancelar por una reservación.

La Entidad RH_MHABITACION (Habitaciones) que contiene el registro de las habitaciones que existe en cada sucursal, a su vez posee relaciones con la Entidad RH_MTIPO_HABITACION (Tipos de Habitaciones).

La Entidad RH_MTARIFA_SUCURSAL (Tarifas Sucursales) que contiene los precios a cobrar por los servicios que prestan en cada sucursal, posee relaciones con la Entidad RH_MSERVICIO (Servicios) y la Entidad RH_MTIPO_TARIFA (Tipos de Tarifas).

La Entidad RH_MRESTAURANTE (Restaurante) contiene el registro de los restaurantes que posee un determinado RH_MLUGAR_DET (Lugar Detalle), este a su vez posee relaciones con la Entidad RH_MAREA_RESTAURANTE (Áreas del Restaurante) y RH_M_ESPECIALIDAD_RESTAURANTE (Especialidad Gastronómica).

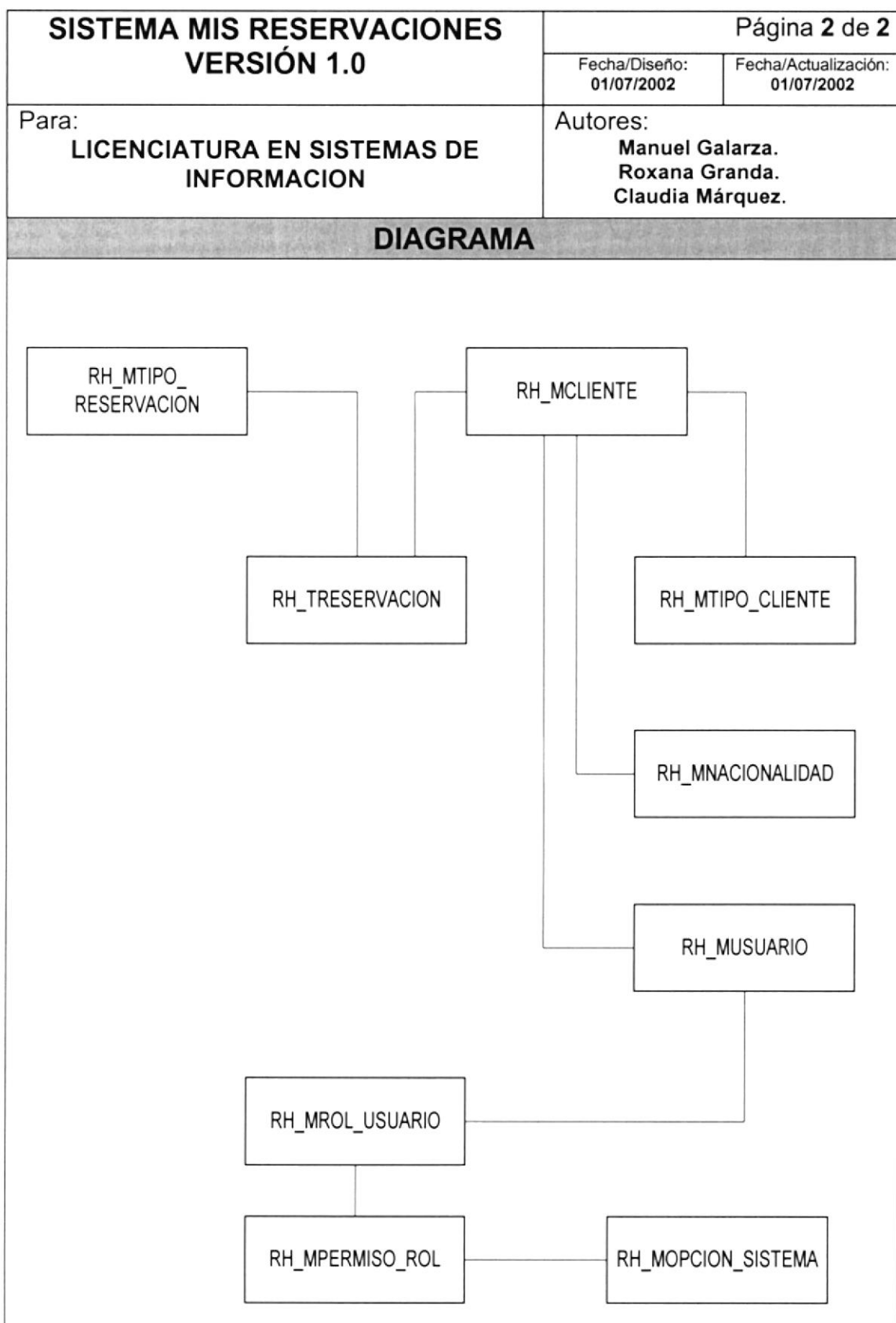


Tabla No. 5. 9. Diagrama del Modelo Físico de Datos.

Explicación:

En esta segunda parte del Modelo Físico de Datos se muestra las siguientes relaciones de entidades:

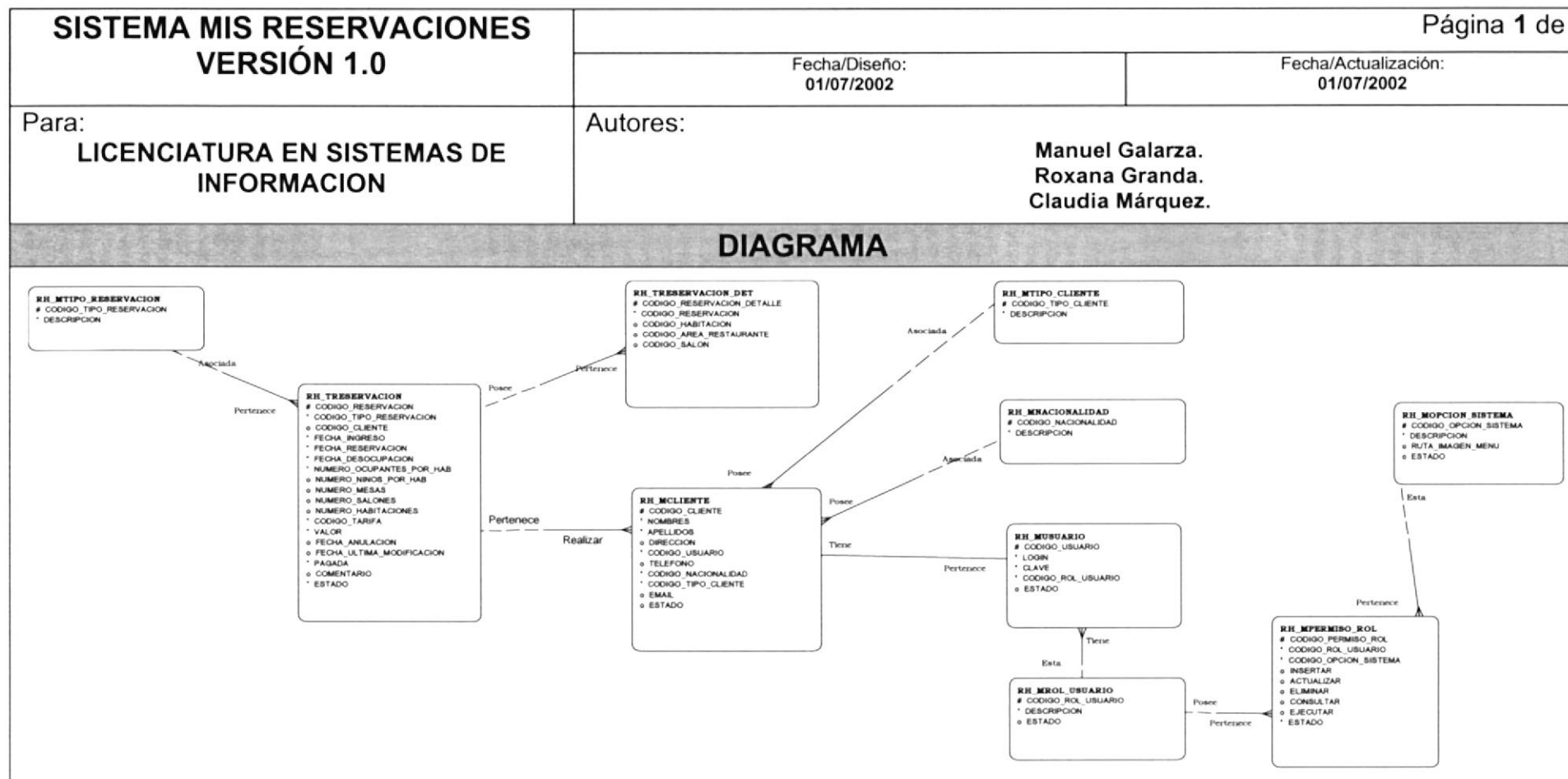
La Entidad RH_TRESERVACION (Reservaciones) que contiene información de las reservaciones registradas en el Sistema además posee relaciones con la Entidad RH_MCLIENTE (Cliente), RH_MTIPO_RESERVACION (Tipos de Reservaciones).

La Entidad RH_MCLIENTE (Cliente) que contiene la información de los usuarios que realizan las reservaciones, posee relaciones con la Entidad RH_MNACIONALIDAD (Nacionalidad), RH_MTIPO_CLIENTE (Tipos de Clientes) y RH_MUSUARIO (Usuarios).

La Entidad RH_MUSUARIO (Usuarios), que contiene información de los usuarios que pueden ingresar al sistema, posee relaciones con la Entidad RH_MROL_USUARIO (Roles de Usuarios), la misma a su vez con la Entidad RH_MPERMISO_ROL (Permisos por Rol) de las diferentes opciones del sistema registradas en la entidad RH_MOPCION_SISTEMA que un usuario puede acceder.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 2			
				Fecha/Diseño: 01/07/2002		Fecha/Actualización: 01/07/2002	
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.			
Nombre de la Tabla: RH_TRESERVACION		Base de Datos: BD_RESERVAS		Tipo: MAESTRA		Longitud de Registros: 24 Bytes	
Descripción: Diccionario de Datos Maestro de las diferentes reservaciones registradas en el sistema							
DESCRIPCIÓN DEL REGISTRO							
No.	Nombre	Tipo	Long.	Descripción	P.K.	F.K.	Nulo
1	Rese_codigo	Smallint	2	Código de la reservación	✓		No
2	Tres_codigo	Tinyint	1	Código del tipo de reservación		✓	No
3	Clie_codigo	Smallint	2	Código del cliente		✓	No
4	Rese_fecha_ingreso	Smalldatetime	8	Fecha de ingreso			No
5	Rese_fecha_reservacion	Smalldatetime	8	Fecha desde de la reservación			No
6	Rese_fecha_desocupacion	Smalldatetime	8	Fecha hasta de la reservación			No
7	Rese_numero_ocupantes_por_hab	Smallint	2	Número de ocupantes			Si
8	Rese_numero_ninos_por_hab	Smallint	2	Número de niños por habitación			Si
9	Rese_numero_mesas	Smallint	2	Número de mesas			Si
10	Rese_numero_salones	Smallint	2	Número de salones			Si
11	Rese_numero_habitaciones	Smallint	2	Número de habitaciones			Si
12	Tari_codigo	Smallint	2	Código de la tarifa		✓	No
13	Rese_valor	Int	4	Valor a pagar en la reservación			No
14	Rese_fecha_anulacion	Smalldatetime	8	Fecha de anulación			Si
15	Rese_fecha_ultima_modificacion	Smalldatetime	8	Fecha ultima modificación			Si
16	Rese_pagada	Char	1	Si esta pagada o no			Si

5.3. Diagrama Entidad - Relación.



Explicación:

En el Modelo Entidad Relación del Sistema de Mis Reservas explicamos con más detalle la relación a nivel físico de las entidades de datos o tablas con todos sus campos.

Incluido la representación de sus claves primarias y foráneas, además sus campos obligatorios y nulos.

5.4. Procedimientos almacenados (Stored Procedures).

A continuación se describirá la definición de los parámetros que reciben cada uno de los procedimientos almacenados utilizados por nuestro sistema.

5.4.1. Procedimientos Almacenados de Eliminación.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_AREA_RESTAURANTE		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un área de un restaurante específico actualizando su estado de “Activo” a “Inactivo”.					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	spv_codigo	Smallint	2	Código del Area del Restaurante	INPUT
2	spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 37. Eliminación Área Restaurante.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_CATEGORIA_LUGAR		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar una categoría de lugar físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	spv_codigo	Smallint	2	Código de la categoría de lugar	INPUT
2	spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 38. Eliminación Categoría de lugar.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_CLIENTE		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de cliente actualizando su estado de “Activo” a “Inactivo”					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	spv_codigo	Smallint	2	Código del Cliente	INPUT
2	spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 39. Eliminación Cliente.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0			Página 1 de 1		
			Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002	
Para: LICENCIATURA EN SISTEMAS DE INFORMACION			Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.		
Nombre del Procedimiento: SP_DEL_ESPECIALIDAD_RESTAU RANTE		Base de Datos: BD_RESERVAS	Tipo: TRANSACCIONAL		
Descripción: Este procedimiento permite eliminar un registro de la especialidad del restaurante físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	spv_codigo	Smallint	2	Código de la especialidad del restaurante	INPUT
2	spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 40. Eliminación Especialidad Restaurante.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_HABITACION		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de habitación de un hotel físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	spv_codigo	Smallint	2	Código de la habitación	INPUT
2	spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 41. Eliminación Habitación.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_LOCALIDAD		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de localidad físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	spv_codigo	Tinyint	1	Código de la localidad	INPUT
2	spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 42. Eliminación Localidad.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_LUGAR_CAB		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de lugar cabecera [cadenas hoteleras] actualizando su estado de “Activo” a “Inactivo”					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	spv_codigo	Smallint	2	Código del Lugar Cabecera	INPUT
2	spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 43. Eliminación Lugar Cabecera.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_LUGAR_DET		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de lugar detalle [sucursales hoteleras] actualizando su estado de “Activo” a “Inactivo”					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	spv_codigo	Smallint	1	Código del Lugar Detalle	INPUT
2	spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 44. Eliminación Lugar Detalle.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_NACIONALIDAD		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de Nacionalidad físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	spv_codigo	Tinyint	1	Código del Lugar Detalle	INPUT
2	spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 45. Eliminación Nacionalidad.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_OPCION_SISTEMA		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de Opciones del Sistema actualizando el estado de “Activo” a “Inactivo”					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de la Opción del Sistema	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 46. Eliminación Opción Sistema.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_PARAMETRO		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de Parámetros actualizando el estado de “Activo” a “Inactivo”					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Parámetros	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 47. Eliminación Parámetro.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_PERMISO_ROL		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de Permiso Rol actualizando el estado de “Activo” a “Inactivo”					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Permiso Rol	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 48. Eliminación Permiso Rol.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_PROVINCIA		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de Provincia físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Provincia	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 49. Eliminación Provincia.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1			
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002		
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.			
Nombre del Procedimiento: SP_DEL_RESERVACION	Base de Datos: BD_RESERVAS	Tipo: TRANSACCIONAL			
Descripción: Este procedimiento permite eliminar un registro de reservación actualizando el estado de “Activo” a “Inactivo”					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	1	Código de la reservación	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 50. Eliminación Reservación.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_RESTAURANTE		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de restaurante actualizando el estado de “Activo” a “Inactivo”					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	1	Código del restaurante	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 51. Eliminación Restaurante.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_ROL_USUARIO		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de restaurante actualizando el estado de “Activo” a “Inactivo”					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	1	Código del restaurante	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 52. Eliminación Rol Usuario.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_SALON		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de salon actualizando el estado de “Activo” a “Inactivo”					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	1	Código del Salon	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 53. Eliminación Salón.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0			Página 1 de 1		
			Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002	
Para: LICENCIATURA EN SISTEMAS DE INFORMACION			Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.		
Nombre del Procedimiento: SP_DEL_SERVICIO		Base de Datos: BD_RESERVAS	Tipo: TRANSACCIONAL		
Descripción: Este procedimiento permite eliminar un registro de servicio físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	1	Código del Servicio	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 54. Eliminación Servicio.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_TARIFA_SUCURSAL		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de tarifa sucursal actualizando el estado de “Activo” a “Inactivo”					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	1	Código de la tarifa sucursal	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 55. Eliminación Tarifa Sucursal.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL TIPO AREA		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de tipo de área físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código del Tipo de Area	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 56. Eliminación Tipo Área.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_TIPO_CLIENTE		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de tipo de cliente físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código del Tipo de Cliente	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 57. Eliminación Tipo Cliente.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_TIPO_HABITACION		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de tipo de habitación físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	2	Código del Tipo de Habitación	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 58. Eliminación Tipo Habitación.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_TIPO_LUGAR		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de tipo de lugar físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	2	Código del Tipo de Lugar	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 59. Eliminación Tipo Lugar.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_TIPO_RESERVACION		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de tipo de reservación físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código del Tipo de Lugar	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 60. Eliminación Tipo Reservación.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_DEL_TIPO_TARIFA		Base de Datos: BD_RESERVAS		Tipo: TRANSACCIONAL	
Descripción: Este procedimiento permite eliminar un registro de tipo de tarifa físicamente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código del Tipo de Tarifa	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 61. Eliminación Tipo Tarifa.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1			
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002		
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.			
Nombre del Procedimiento: SP_DEL_USUARIO	Base de Datos: BD_RESERVAS	Tipo: TRANSACCIONAL			
Descripción: Este procedimiento permite eliminar un registro de usuario actualizando el estado de “Activo” a “Inactivo”					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código del Usuario	INPUT
2	Spv_codigo_error	Varchar	600	Mensaje de Retorno	OUTPUT
Observación:					

Tabla No. 5. 62. Eliminación Usuario.

5.4.2. Procedimientos almacenados de consulta por descripción.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_CATEGORIA_LUGAR		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Categoría de Lugar por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 63. Consulta por descripción a Categoría Lugar.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_ESPECIALIDAD_R ESTAURANTE		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Especialidad de Restaurante por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 64. Consulta por descripción a Especialidad.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1			
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002		
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.			
Nombre del Procedimiento: SP_GETBYDS_LOCALIDAD	Base de Datos: BD_RESERVAS	Tipo: CONSULTA			
Descripción: Este procedimiento permite consultar registros a la tabla de Localidad por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 65. Consulta por descripción a Localidad.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1			
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002		
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.			
Nombre del Procedimiento: SP_GETBYDS_LUGAR_CAB	Base de Datos: BD_RESERVAS	Tipo: CONSULTA			
Descripción: Este procedimiento permite consultar registros a la tabla de Lugar Cabecera por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 66. Consulta por descripción a Lugar Cabecera.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1			
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002		
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.			
Nombre del Procedimiento: SP_GETBYDS_NACIONALIDAD	Base de Datos: BD_RESERVAS	Tipo: CONSULTA			
Descripción: Este procedimiento permite consultar registros a la tabla de Nacionalidad por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 67. Consulta por descripción a Nacionalidad.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1			
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002		
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.			
Nombre del Procedimiento: SP_GETBYDS_OPCION_SISTEMA	Base de Datos: BD_RESERVAS	Tipo: CONSULTA			
Descripción: Este procedimiento permite consultar registros a la tabla de Opción del Sistema por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 68. Consulta por descripción a Opción del Sistema.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_PARAMETRO		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Parámetros por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 69. Consulta por descripción a Parámetro.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0			Página 1 de 1		
			Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002	
Para: LICENCIATURA EN SISTEMAS DE INFORMACION			Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.		
Nombre del Procedimiento: SP_GETBYDS_PROVINCIA		Base de Datos: BD_RESERVAS	Tipo: CONSULTA		
Descripción: Este procedimiento permite consultar registros a la tabla de Provincia por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 70. Consulta por descripción a Provincia.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_RESTAURANTE		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Restaurante por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 71. Consulta por descripción a Restaurante.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_ROL_USUARIO		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Rol Usuarios por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 72. Consulta por descripción a Rol Usuario.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_SALON		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Salón por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 73. Consulta por descripción a Salón.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_SERVICIO		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Servicio por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 74. Consulta por descripción a Servicio.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_TIPO_AREA		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tipo de Área por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 75. Consulta por descripción a Tipo Área.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_TIPO_CLIENTE		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tipo de Cliente por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 76. Consulta por descripción a Tipo Cliente.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_TIPO_HABITACION		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tipo de Habitación por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 77. Consulta por descripción a Tipo Habitación.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_TIPO_LUGAR		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tipo de Lugar por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 78. Consulta por descripción a Tipo Lugar.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_TIPO_RESERVACION		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tipo de Reservación por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 79. Consulta por descripción a Tipo Reservación.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYDS_TIPO_TARIFA		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tipo de Tarifa por descripción					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_descripcion	Varchar	200	Descripción	Input
Observación:					

Tabla No. 5. 80. Consulta por descripción a Tipo Tarifa.

5.4.3. Procedimientos almacenados de consulta por código.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_AREA_RESTAURANTE		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Area de Restaurante por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	2	Código del Area de Restaurante	Input
Observación:					

Tabla No. 5. 81. Consulta por código a Área Restaurante.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_CATEGORIA_LUGAR		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Categoría de Lugar por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de la Categoría de Lugar	Input
Observación:					

Tabla No. 5. 82. Consulta por código a Categoría de Lugar.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_CLIENTE		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Categoría de Lugar por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	2	Código del Cliente	Input
Observación:					

Tabla No. 5. 83. Consulta por código a Cliente.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_ESPECIALIDAD_RE STAURANTE		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Especialidad Restaurante por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de la Especialidad Restaurante	Input
Observación:					

Tabla No. 5. 84. Consulta por código a Especialidad Restaurante.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_HABITACION		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Habitación por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Habitación	Input
Observación:					

Tabla No. 5. 85. Consulta por código a Habitación.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_LOCALIDAD		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Localidad por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Localidad	Input
Observación:					

Tabla No. 5. 86. Consulta por código a Localidad.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_LUGAR_CAB		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Lugar Cabecera por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	2	Código de Lugar Cabecera	Input
Observación:					

Tabla No. 5. 87. Consulta por código a Lugar Cabecera.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_LUGAR_DET		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Lugar Detalle por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Lugar Detalle	Input
Observación:					

Tabla No. 5. 88. Consulta por código a Lugar Detalle.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_NACIONALIDAD		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Nacionalidad por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Nacionalidad	Input
Observación:					

Tabla No. 5. 89. Consulta por código a Nacionalidad.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_OPCION_SISTEMA		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Opción del Sistema por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Opción del Sistema	Input
Observación:					

Tabla No. 5. 90. Consulta por código a Opción Sistema.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_PARAMETRO		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Parámetro por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	2	Código de Parametro	Input
Observación:					

Tabla No. 5. 91. Consulta por código a Parámetro.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_PERMISO_ROL		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Permiso Rol por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Permiso Rol	Input
Observación:					

Tabla No. 5. 92. Consulta por código a Permiso Rol.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_PROVINCIA		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Provincia por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Provincia	Input
Observación:					

Tabla No. 5. 93. Consulta por código a Provincia.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_RESERVACION		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Reservación por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	2	Código de Reservación	Input
Observación:					

Tabla No. 5. 94. Consulta por código a Reservación.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_RESTAURANTE		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Restaurante por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	2	Código de Restaurante	Input
Observación:					

Tabla No. 5. 95. Consulta por código a Restaurante.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_ROL_USUARIO		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Rol Usuarios por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Rol Usuario	Input
Observación:					

Tabla No. 5. 96. Consulta por código a Rol Usuario.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0			Página 1 de 1		
			Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002	
Para: LICENCIATURA EN SISTEMAS DE INFORMACION			Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.		
Nombre del Procedimiento: SP_GETBYID_SALON		Base de Datos: BD_RESERVAS	Tipo: CONSULTA		
Descripción: Este procedimiento permite consultar registros a la tabla de Salon por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Salon	Input
Observación:					

Tabla No. 5. 97. Consulta por código a Salón.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_SERVICIO		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Servicios por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	2	Código de Servicio	Input
Observación:					

Tabla No. 5. 98. Consulta por código a Servicio.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_TARIFA_SUCURSAL		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tarifa Sucursal por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Smallint	2	Código de Tarifa Sucursal	Input
Observación:					

Tabla No. 5. 99. Consulta por código a Tarifa Sucursal.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_TIPO_AREA		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tipo de Area por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código del Tipo de Area	Input
Observación:					

Tabla No. 5. 100. Consulta por código a Tipo Área.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_TIPO_CLIENTE		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tipo de Cliente por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código del Tipo de Cliente	Input
Observación:					

Tabla No. 5. 101. Consulta por código a Tipo Cliente.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_TIPO_HABITACION		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tipo de Habitación por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código del Tipo de Habitación	Input
Observación:					

Tabla No. 5. 102. Consulta por código a Tipo habitación.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_TIPO_LUGAR		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tipo de Lugar por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código del Tipo de Lugar	Input
Observación:					

Tabla No. 5. 103. Consulta por código a Tipo Lugar.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_TIPO_RESERVACION		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tipo de Reservación por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código del Tipo de Reservación	Input
Observación:					

Tabla No. 5. 104. Consulta por código a Tipo Reservación.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_TIPO_TARIFA		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Tipo de Tarifa por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código del Tipo de Tarifa	Input
Observación:					

Tabla No. 5. 105. Consulta por código a Tipo Tarifa.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_GETBYID_USUARIO		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar registros a la tabla de Usuarios por código					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
1	Spv_codigo	Tinyint	1	Código de Usuario	Input
Observación:					

Tabla No. 5. 106. Consulta por código a Usuario.

5.4.4. Procedimientos almacenados de Consulta Masiva.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_LISTALL_AREA_RESTAURANTE		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar todos los registros de la tabla de Área restaurante					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
Observación:					

Tabla No. 5. 107. Consulta masiva a Área de Restaurante.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_LISTALL_CATEGORIA_LUGAR		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar todos los registros de la tabla de Categoría de Lugar					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
Observación:					

Tabla No. 5. 108. Consulta masiva a Categoría Lugar.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_LISTALL_CLIENTE		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar todos los registros de la tabla de Cliente					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
Observación:					

Tabla No. 5. 109. Consulta masiva a Cliente.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1			
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002		
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.			
Nombre del Procedimiento: SP_LISTALL_ESPECIALIDAD_RESTAURANTE	Base de Datos: BD_RESERVAS	Tipo: CONSULTA			
Descripción: Este procedimiento permite consultar todos los registros de la tabla de Especialidad Restaurante					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
Observación:					

Tabla No. 5. 110. Consulta masiva a Especialidad Restaurante.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1	
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_LISTALL_HABITACION	Base de Datos: BD_RESERVAS	Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar todos los registros de la tabla de Habitación			
DESCRIPCIÓN DEL REGISTRO			
No.	Nombre	Tipo	Long. Descripción IN-OUT-[IN/OUT]
Observación:			

Tabla No. 5. 111. Consulta masiva a Habitación.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1	
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_LISTALL_LOCALIDAD	Base de Datos: BD_RESERVAS	Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar todos los registros de la tabla de Localidad			
DESCRIPCIÓN DEL REGISTRO			
No.	Nombre	Tipo	Long. Descripción IN-OUT-[IN/OUT]
Observación:			

Tabla No. 5. 112. Consulta masiva a Localidad.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1	
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_LISTALL_LUGAR_CAB	Base de Datos: BD_RESERVAS	Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar todos los registros de la tabla de Lugar Cabecera			
DESCRIPCIÓN DEL REGISTRO			
No.	Nombre	Tipo	Long.
Descripción			
IN-OUT-[IN/OUT]			
Observación:			

Tabla No. 5. 113. Consulta masiva a Lugar Cabecera.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1	
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_LISTALL_LUGAR_DET	Base de Datos: BD_RESERVAS	Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar todos los registros de la tabla de Lugar Detalle			
DESCRIPCIÓN DEL REGISTRO			
No.	Nombre	Tipo	Long.
Descripción			
IN-OUT-[IN/OUT]			
Observación:			

Tabla No. 5. 114. Consulta masiva a Lugar Detalle.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_LISTALL_NACIONALIDAD		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar todos los registros de la tabla de Nacionalidad					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
Observación:					

Tabla No. 5. 115. Consulta masiva a Nacionalidad.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0				Página 1 de 1	
				Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION				Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_LISTALL_OPCION_SISTEMA		Base de Datos: BD_RESERVAS		Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar todos los registros de la tabla de Opción del Sistema					
DESCRIPCIÓN DEL REGISTRO					
No.	Nombre	Tipo	Long.	Descripción	IN-OUT-[IN/OUT]
Observación:					

Tabla No. 5. 116. Consulta masiva a Opción Sistema.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1	
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_LISTALL_PARAMETRO	Base de Datos: BD_RESERVAS	Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar todos los registros de la tabla de Parametro			
DESCRIPCIÓN DEL REGISTRO			
No.	Nombre	Tipo	Long.
IN-OUT-[IN/OUT]			
Observación:			

Tabla No. 5. 117. Consulta masiva a Parámetro.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1	
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Nombre del Procedimiento: SP_LISTALL_PERMISO_ROL	Base de Datos: BD_RESERVAS	Tipo: CONSULTA	
Descripción: Este procedimiento permite consultar todos los registros de la tabla de Permiso Rol			
DESCRIPCIÓN DEL REGISTRO			
No.	Nombre	Tipo	Long.
IN-OUT-[IN/OUT]			
Observación:			

Tabla No. 5. 118. Consulta masiva a Permiso Rol.

Capítulo **6**

IMPLEMENTACIÓN DE LA CAPA DE NEGOCIOS

Capítulo 6

6. IMPLEMENTACIÓN DE LA CAPA DE NEGOCIOS.

Este capítulo contiene una explicación acerca de la implementación de la capa de negocio, además la estructura de los componentes de acceso a datos y de negocios.

6.1. Formatos para el Diseño de Componentes.

El formato para el diseño de los componentes comprende una estandarización para los nombres de los mismos y de sus librerías, entre estos formatos tenemos:

NOMBRE DE LOS COMPONENTES DE ACCESO A DATOS	
Descripción:	
La estandarización a los nombres de los componentes de acceso a datos nos permitirá identificarlos fácilmente cuando se los enlace al componente de negocio.	
DB _XXXXXXXXX.DLL Nombre Descriptivo. Indicador de Componente de Datos	
Ejemplo: DB_CLIENTE.DLL	

Tabla No. 6.1. Estandarización de los Componentes de datos.

NOMBRE DE LOS COMPONENTES DE NEGOCIOS	
Descripción:	
La estandarización a los nombres de los componentes de negocios nos permitirá identificarlos fácilmente cuando se los enlace a la Aplicación Cliente.	
BU _TRXXXXXXXXXX.DLL Nombre Descriptivo. Indicador de Componente de Datos	
Ejemplo: BU_TRXCLIENTE.DLL	

Tabla No. 6.2. Estandarización de los Componentes de Negocios.

NOMBRE DE LAS INTERFASES DE DATOS	
Descripción: La estandarización a los nombres de las interfases de datos nos permitirá identificarlos fácilmente cuando se los enlace al Componente de Datos, una vez generado el TLB.	
XXXXXXXXXXXXXXXX.IDL 	Nombre Descriptivo.
Ejemplo: CLIENTE.IDL	

Tabla No. 6.3. Estandarización de las Interfases de Datos.

NOMBRE DE LAS INTERFASES DE NEGOCIOS	
Descripción: La estandarización a los nombres de las interfases de negocios nos permitirá identificarlos fácilmente cuando se los enlace al Aplicativo, una vez generado el TLB.	
TRX XXXXXXXXXXXXXXXX.IDL 	Nombre Descriptivo.
Ejemplo: TRXCLIENTE.IDL	

Tabla No. 6.4. Estandarización de las Interfases de Negocios.

NOMBRE DE LOS MÓDULOS DE CLASE	
Descripción: La estandarización de los nombres de los módulos de clase nos permite una mejor identificación de su fuente.	
M_XXXXXXXXXXXXXXXXX.CLS Nombre descriptivo del Módulo. Ejemplo: M_CLIENTE.CLS	

Tabla No. 6.5. Estandarización de los Módulos de Clase

NOMBRE DE LOS MÉTODOS	
Descripción: Para la estandarización de los métodos que se programen en un módulo de clase no se encuentran definidos en un estándar.	
XXXXXXXXXXXXXXXXX Nombre descriptivo del Método. Ejemplo: CREACION_USUARIO	

Tabla No. 6.6. Estandarización de los Métodos

6.2. Formatos para variables de sistema.

El formato para los nombres de los programas y fuentes que utilizarán los componentes de datos y de negocios son los siguientes:

NOMBRE DE L PROYECTO	
Descripción: La estandarización para el nombre del proyecto, este comenzará con las siglas APP y la parte restante será un nombre descriptivo a su contenido	
APP_XXXXXXXXXX..... 	
Ejemplo: APP_HOTEL	

Tabla No. 6.7. Estandarización del Proyecto

NOMBRE LÓGICO Y FÍSICO DE FORMAS	
Descripción: La estandarización para los nombres físicos y lógicos de la formas, estas deberán comenzar con el Tipo de Forma y el número de nivel según la opción en el sistema.	
INGFR_XXXX.frm 	
Ejemplo: INGFR0101.FRM	

Tabla No. 6.8. Estandarización de las formas.

NOMBRE DE VARIABLES	
Descripción: Para la estandarización de los nombres de las variables tendrá que revisar si la variable al momento de declararla es de ámbito local o privada.	
XX_XXXXXXXXX → Descriptivo. → Indicador Tipo de Variable. → Indicador de Ámbito de la Variable. Indicador de Ámbito de la variable: P = Global L = Local Indicador Tipo de variable N = Numérico D = Fecha V = Varchar Ejemplo: LV_ESTADO LN_VALOR PV_CUENTA PN_PROCESO	

Tabla No. 6.9. Estandarización de las variables.

6.3. Estructura de los Componentes de Acceso a Datos.

La estructura de un componente de acceso a datos de esta aplicación está basada en todo lo establecido en las especificaciones COM, para conocer algo más acerca de la estructura de estos componentes, a través de los siguientes puntos se realizará una explicación más detallada.

6.3.1. Introducción a los Componentes de Acceso a Datos.

Los componentes de acceso a datos son pequeñas piezas de código reutilizables que contienen la Lógica de Negocios y parte de la Lógica de Acceso a Datos.

Los componentes de datos encapsulan el acceso a los datos, donde se incluyen peticiones inserción, modificación, eliminación y consulta dirigidas al Proceso Servidor de Base de Datos, para que sea este el último quién finalmente accese a la Base de Datos.

Una vez que los componentes de datos reciben la respuesta a la solicitud del Proceso Servidor de Base de datos, éstos también contestan a los componentes de negocios.

Acerca de la estandarización empleada en los nombres de los componentes de acceso a datos estos comenzarán con el prefijo "DB" como se explicó anteriormente en el punto 6.1. "Formatos para el diseño de componentes".

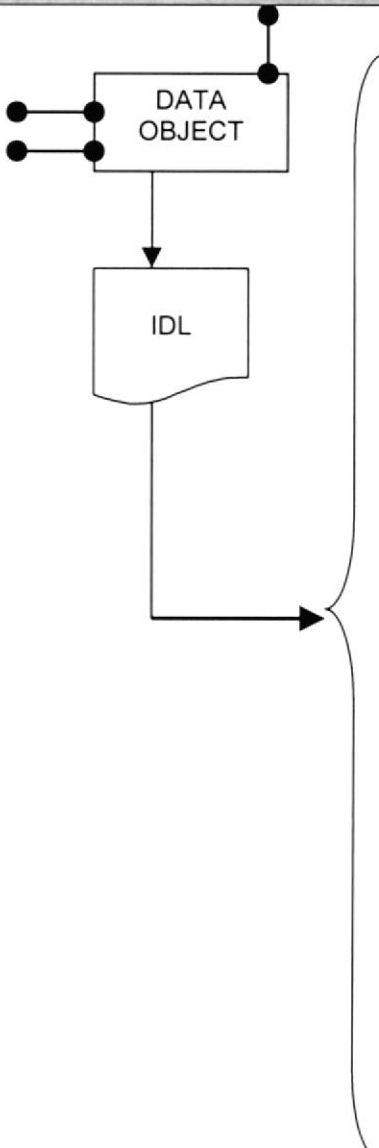
SISTEMA MIS RESERVACIONES		Página 1 de 1	
VERSIÓN 1.0		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Descripción: Para el diseño de las interfases (IDL), su codificación fue creada en Microsoft Visual C++, además cada interfase se le fue asignada una identificación única a través del GUIGEN.exe y compiladas con el Compilador MIDL para generar las librerías TLB para su distribución e implementación en los componentes de datos.			
DISEÑO DE INTERFASES			
 <pre>[/** ASIGNACION DEL GUID A LA LIBRERIA uuid(B749A00D-1D03-4771-AB4F-8AAB7F3A127D), version(1.0)] /* NOMBRE DE LA LIBRERIA library AreaRestaurante{ /* IMPORTACION DE LIBRERIAS EXTERNAS importlib("msado21.tlb"); importlib("stdole2.tlb"); /* DECLARACION DE INTERFASES interface IAreaRestaurante; /*ASIGNACION DEL GUID A LA INTERFAZ [odl, uuid(AE3189B7-42D2-47c0-971C-5510EF5D1DF2), version(1.0), dual, nonextensible, oleautomation] /* DECLARACION DE INTERFASES interface IAreaRestaurante:IDispatch { [id(0)] HRESULT AddUpdDlt([in] long vartipo, [in, out] long* codigo, [in] long codrestaurante, [in] long codtipoarea, [in] long numeromesas, [in] long numeromesasdisponibles, [in] DATE fechaultreserva, [in] BSTR estado, [out, retval] long*); }; };</pre>			

Tabla No. 6.10. Diseño de Interfases de componentes de datos.

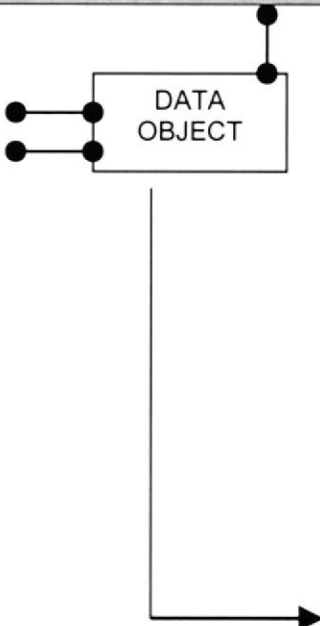
SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1	
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Descripción: La codificación de cada componente de dato fue realizada en Microsoft Visual Basic integradas a su vez con la librería del MTS para la administración del uso del mismo y implementado los métodos de las interfases anteriormente creadas.			
DISEÑO DE COMPONENTES			
 <pre>Option Explicit 'Definicion de Constantes Const database_name = "BD_RESERVAS" 'Implementacion de la IDL - AreaRestaurant Implements AreaRestaurante.IAreaRestaurante 'Declaración de Métodos Private Function IAreaRestaurante_AddUpdDlt(ByVal vartipo As Long, _ ByRef Codigo As Long) As Long 'Definición de Variables de trabajo ' 'Manejo de Excepcion de Error 'Inicializacion de Variables '..Inicializacion de variables de trabajo '...Instanciando variable del MTS '...Instanciando Objeto Conexion '....Llamando metodo de conexion a la Base de Datos '...Instanciando Objeto ADODBCommand '...Verificando Tipo de Accion a ejecutar por el Componente 'Añadiendo parametros para el stored procedure 'Asignacion y ejecucion del stored procedure a la función 'Manejo de validaciones de error ' Se indica al MTS la transaccion exitosa 'Manejo de la excepcion de error ' Inicializacion de variables ' Si la transaccion no fue exitosa se indica al MTS el error 'Mensaje de Error a retornar</pre>			

Tabla No. 6.11. Estructura de un componente de dato.

6.4. Objetos distribuidos de acceso a datos.

Los objetos distribuidos se encuentran embebidos en los componentes. Los componentes son piezas de código modulares cuya principal característica es la reusabilidad.

Cada componente de acceso a dato contiene en su interior todo un modelo de objetos que son los que le permiten ofrecer los servicios hacia el mundo exterior encapsulando el código en ellos.

6.5. Comunicación con el Proceso Servidor de Datos.

El sistema de Mis Reservas utiliza como middleware para la comunicación con el proceso servidor de datos, COM+. (Véase Apéndice C). Otro tipo de middleware empleado por el sistema es Microsoft ActiveX Data Objects (ADO) como la interfaz para el acceso a los datos permitiendo la llamada a los procedimientos almacenados creados en Microsoft SQL SERVER.

6.6. Estructura de los Componentes de Negocios.

La estructura de un componente de negocio de esta aplicación está basado en todo lo establecido en las especificaciones COM, para conocer algo más acerca de la estructura de estos componentes, a través de los siguientes puntos se realizará una explicación más detallada.

6.6.1. Introducción a los Componentes de Acceso a Datos.

Los componentes de negocios encapsulan las reglas de negocios del sistema de Mis Reservaciones, estas reglas son un conjunto de prácticas y políticas de la empresa que deben ser reflejadas en la aplicación y en la base de datos.

Todos los componentes de negocios de la aplicación cuando reciban un requerimiento por parte del cliente en ASP o Microsoft Visual Basic que necesite leer, actualizar, eliminar o crear un registro en la Base de Datos, estos se verán en la obligación de solicitar este trabajo a los componentes de acceso a datos ya que estos son los encargados de implementar la lógica de acceso a los datos.

Similar a los componentes de acceso a datos, los componentes de negocios también obedecen al estándar binario de COM.

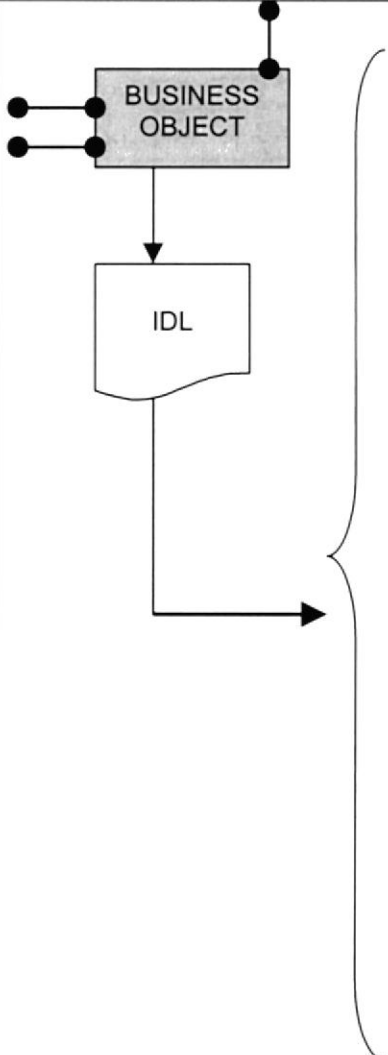
SISTEMA MIS RESERVACIONES		Página 1 de 1	
VERSIÓN 1.0		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Descripción: Para el diseño de las interfases de negocio (IDL), su codificación fue creada en Microsoft Visual C++, además cada interfase se le fue asignada una identificación única a través del GUIGEN.exe y compiladas con el Compilador MIDL para generar las librerías TLB para su distribución e implementación en los componentes de negocio.			
DISEÑO DE INTERFASES			
 <pre> [/** ASIGNACION DEL GUID A LA LIBRERIA uuid(B749A00D-1D03-4771-AB4F-8AAB7F3A127D), version(1.0)] /* NOMBRE DE LA LIBRERIA library Trxarearestaurante{ /* IMPORTACION DE LIBRERIAS EXTERNAS importlib("msado21.tlb"); importlib("stdole2.tlb"); /* DECLARACION DE INTERFASES interface ITrxarearestaurante; /*ASIGNACION DEL GUID A LA INTERFAZ [odl, uuid(AE3189B7-42D2-47c0-971C-5510EF5D1DF2), version(1.0), dual, nonextensible, oleautomation] /* DECLARACION DE INTERFASES interface ITrxAreaRestaurante:IDispatch { [id(0)] HRESULT AddUpdDlt([in] long vartipo, [in, out] long* codigo, [in] long codrestaurante, [in] long codtipoarea, [in] long numeromesas, [in] long numeromesasdisponibles, [in] long fechaaltreserva, [in] BSTR estado, [out, retval] long*); }; } </pre>			

Tabla No. 6.12. Diseño de las interfases de los componentes de negocios.

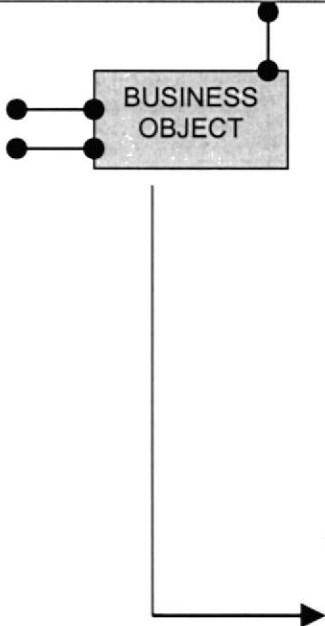
SISTEMA MIS RESERVACIONES		Página 1 de 1	
VERSIÓN 1.0		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Descripción: La codificación de cada componente de dato fue realizada en Microsoft Visual Basic integradas a su vez con la librería del MTS para la administración del uso del mismo y implementado los métodos de las interfases anteriormente creadas.			
DISEÑO DE COMPONENTES			
			
<pre>Option Explicit 'Definicion de Constantes Const modname = "bu_trxCategoriaLugar.m_trxCatLugar" 'Implementacion de la IDL - TRXCategoriaLugar Implements trxcategoriaLugar.ITrxcategoriaLugar 'Declaración de Métodos Private Function ItrxCategoriaLugar_AddUpdDlt(ByVal vartipo As Long, _ ByVal codigo As Long, _ ByVal descripcion As String, _ ByVal numeroestrellas As String) As Long 'Definición de Variables de trabajo ' 'Manejo de Excepcion de Error 'Inicializacion de Variables '...Instanciando variable del MTS '...Instanciando Objeto de Datos 'Asignacion del resultado de la transaccion a la funcion ' Se indica al MTS la transaccion exitosa 'Manejo de la excepcion de error ' Inicializacion de variables ' Si la transaccion no fue exitosa se indica al MTS el error 'Mensaje de Error a retornar</pre>			

Tabla No. 6.13. Estructura de un componente de negocio.

6.7. Objetos distribuidos de negocios.

Similarmente a los componentes de acceso a datos, los componentes de negocios empleados en el Sistema también poseen incluido un modelo de objetos de negocios, los mismos que se encuentran embebidos en el componente.

6.8. Comunicación entre objetos distribuidos.

En el sistema de Mis reservaciones se utiliza como middleware para la comunicación con los componentes de acceso a datos, COM+. (Véase Apéndice C).

Además otro tipo de middleware empleado en el sistema es Microsoft ActiveX Data Objects (ADO) como la interfaz para la invocación de los métodos de consulta y luego recibir como dato de retorno los Recordset devueltos por los componentes de acceso a datos con los registros recuperados.

6.9. Proceso Servidor Transaccional.

El proceso servidor de transacciones es el “encargado de administrar las transacciones que se realizan en cada uno de los componentes desarrollados en la aplicación, tanto de acceso a datos como de negocio a nivel de la capa 2.

Cabe indicar además, que el proceso servidor de transacciones también ejecuta procesos y maneja el pool de conexiones al servidor de Base de datos, además de los “threads”.

Este proceso servidor transaccional es un TP Monitor orientado a componentes y que hace las veces de proceso discriminador o intermediario entre los procesos de la capa 1 y capa3.

Acerca del producto empleado en la aplicación para la administración de las transacciones, procesos y seguridades en los componentes es el Microsoft Transaction Server.

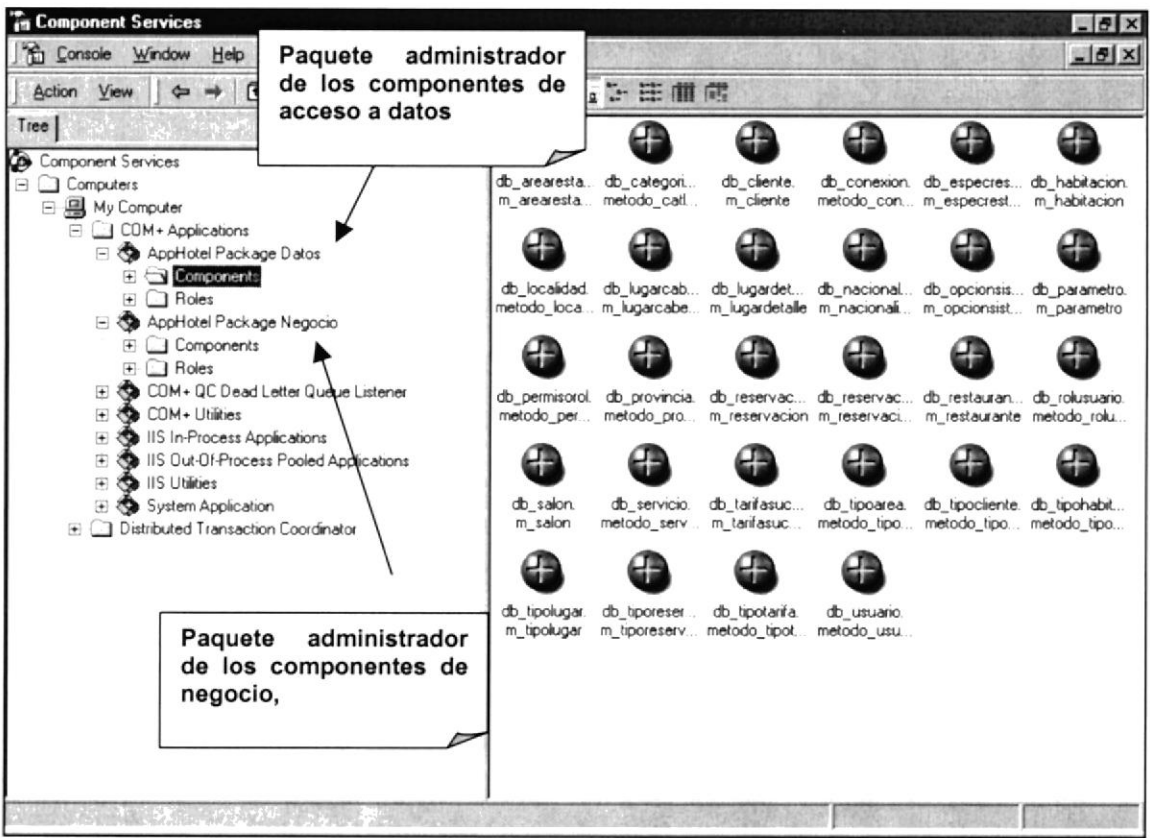


Figura 6.1. Paquetes de la Aplicación administrado por MTS.

Capítulo 7

IMPLEMENTACIÓN DE LA CAPA DE PRESENTACIÓN

Capítulo 7

7. IMPLEMENTACIÓN DE LA CAPA DE PRESENTACIÓN.

Este capítulo contiene una explicación acerca de la implementación de la capa de presentación y como está estructurada nuestra aplicación Web y standalone.

7.1. Diseño del Sitio Web – Cliente Basado en Browser.

El desarrollo de nuestro Sitio Web trae consigo muchas consideraciones realizadas al entorno considerando características y herramientas empleadas por los usuarios para la realización de reservaciones de habitaciones, mesas y salones de eventos. Para lo cual se desarrollo una aplicación web que cumple todas su expectativas y que a continuación a través de los siguientes puntos ustedes conocerán más al detalle.

7.1.1. Introducción a la Capa de Presentación.

La arquitectura del sistema está desarrollado en un ambiente de 3 capas: Capa de presentación, Capa de Negocio y Capa de Acceso a Datos.

La Capa 1 la compone el Browser que es el encargado de enviar requerimientos HTTP al Servidor Web que se encuentra en la Capa 2. Estos requerimientos son solicitudes por una página estática HTML o por la ejecución de algún programa que generará código HTML de forma dinámica.

En ambos casos una vez que el Servidor Web termina su trabajo, éste envía código HTML al Web Browser quién finalmente se encargará de interpretar aquel código HTML recibido y de mostrar la página Web apropiada al usuario del sistema.

7.1.2. Estructura del WebSite (Mapa del Sitio).

La aplicación web de Mis Reservas está estructurado de la siguiente manera:

- La aplicación web consta de una página de inicio llamada **Home.asp**, la misma que invoca al Menú Introductorio de la aplicación **Principal.htm**

- Está página contiene diferentes opciones que permitirá al Usuario conocer algo más acerca del sistema y de la empresa.
- La opción *[Ingresar al Sistema]* nos permitirá ingresar a un menú de opciones transaccionales, **Menu_principal.htm** en donde podemos realizar nuevas reservaciones, consultar, actualizar información o eliminar reservaciones.

A través del siguiente gráfico mostramos el Mapa del Sitio de Mis Reservaciones.

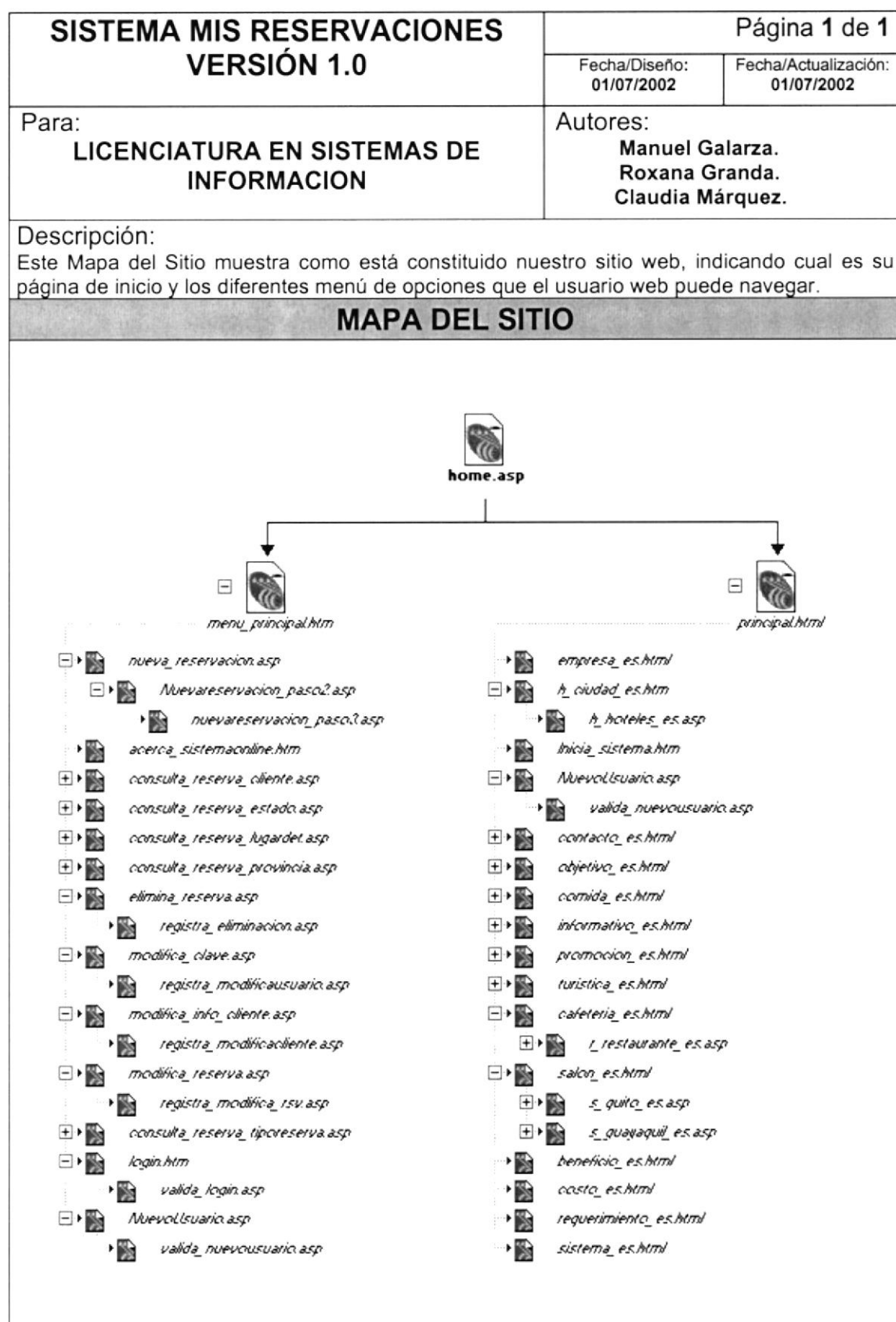


Tabla No. 7.1. Mapa del Sitio.

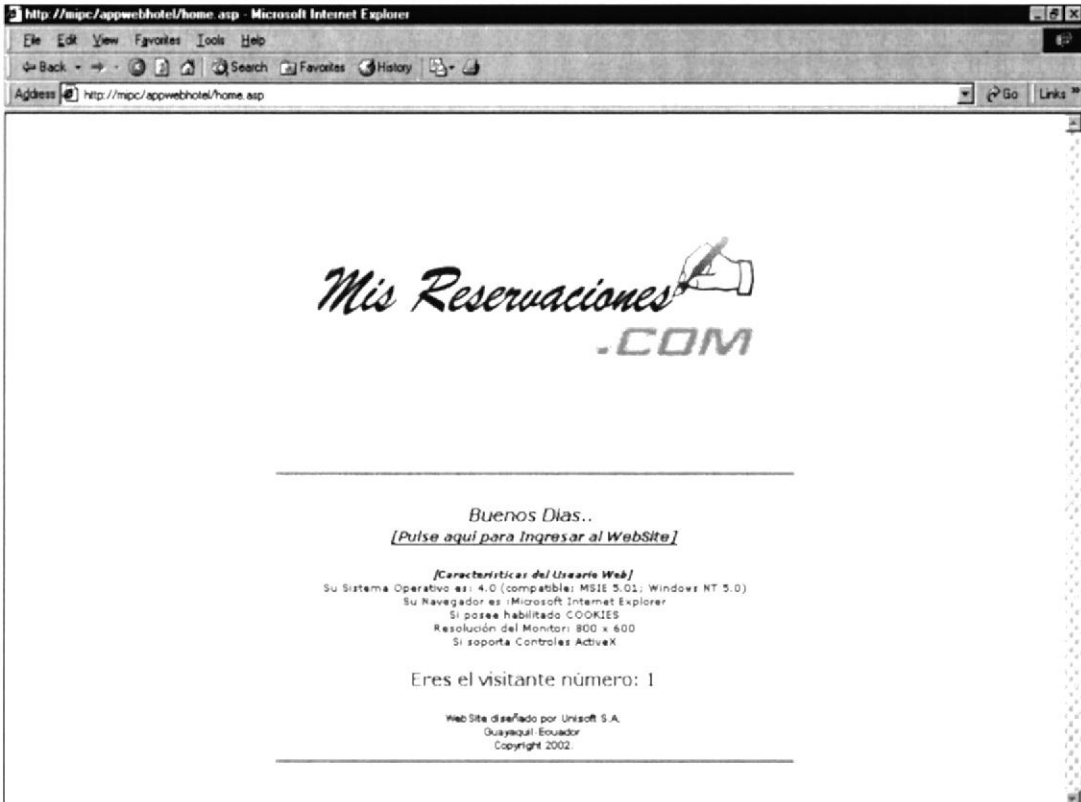
SISTEMA MIS RESERVACIONES VERSIÓN 1.0	Página 1 de 1	
	Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION	Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Página: home.asp		
Descripción:		
Página de Inicio de la Aplicación Web de Mis Reservasiones.		
HOMEPAGE DE INICIO		
		

Tabla No. 7.2. Página de Inicio del Sitio Web.

SISTEMA MIS RESERVACIONES		Página 1 de 1	
VERSIÓN 1.0		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Página: <i>principal.html</i>			
Descripción: Menú principal introductorio de la Aplicación de Mis Reservaciones			
MENÚ PRINCIPAL			
			

Tabla No. 7.3. Menú introductorio al Sitio Web.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0	Página 1 de 1			
	Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002		
Para: LICENCIATURA EN SISTEMAS DE INFORMACION	Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.			
Página: <i>nueva_reservacion.asp</i>				
Descripción: Está página Web dinámica nos permite realizar diferentes tipos de reservaciones				
TRANSACCIONES DE RESERVACIÓN				
				

Tabla No. 7.4. Menú transaccional del Sitio Web.

7.2. Cliente Standalone.

La aplicación standalone de Mis Reservas se encuentra estructurado de la siguiente manera:

- La aplicación consta de una forma contenedora y incluye un menú de opciones para invocación de las demás pantallas.

A través del siguiente gráfico mostramos algo más acerca de este aplicativo.

SISTEMA MIS RESERVACIONES VERSIÓN 1.0		Página 1 de 1	
		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Programa: <i>ResPy0000.fmb</i>			
Descripción: Está pantalla contenedora de opciones realiza las invocaciones de las demás pantallas del aplicativo standalone			
PANTALLA PRINCIPAL			
			

Tabla No. 7.5. Pantalla principal de la aplicación standalone.

SISTEMA MIS RESERVACIONES		Página 1 de 1	
VERSIÓN 1.0		Fecha/Diseño: 01/07/2002	Fecha/Actualización: 01/07/2002
Para: LICENCIATURA EN SISTEMAS DE INFORMACION		Autores: Manuel Galarza. Roxana Granda. Claudia Márquez.	
Programa: <i>ResPy0000.fmb</i>			
Descripción: Entre las opciones que se encuentran agrupadas tenemos Archivos, Procesos y seguridades, en donde el usuario podrá realizar mantenimientos a las tablas y consultar reservaciones.			
MENÚ DE OPCIONES			

Tabla No. 7.6. Menú de Opciones de la aplicación standalone.

7.3. HTML.

El principio esencial del lenguaje HTML, es el uso de las etiquetas (TAGS), lo que se encuentre entre ambas etiquetas estará influenciada por ellas. Cabe indicar además que el lenguaje HTML provee una gran variedad de TAGS que permiten dar formato al texto incluido en las páginas

De esta manera el siguiente código muestra la programación realizada:

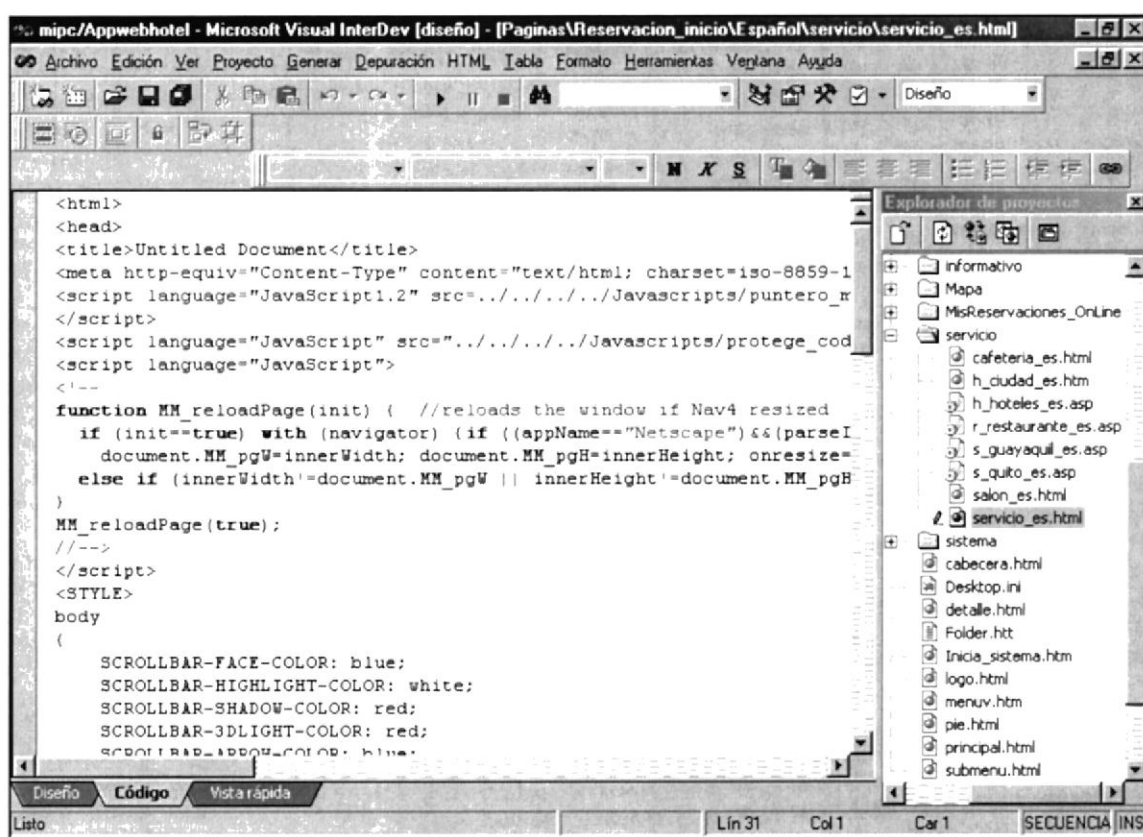


Figura 7.1. Codificación de una página HTML en nuestro Sitio Web

7.4. CLIENTE SCRIPT - DHTML.

DHTML (Dynamic HTML), no es una definición tangible como lo es HTML o Javascript, puestos que estos últimos son elementos específicos fácilmente identificados como tecnologías del Web. DHTML por otro lado es un término comercial utilizado por Internet Explorer y Netscape Navigator para describir un conjunto de tecnologías introducidas en sus navegadores, con la idea de ampliar las capacidades dinámicas al mostrar contenido Web.

Se define también como mejora al HTML; esto es, incluye elementos que desde ahora son tratados como objetos programables. A estos objetos se les puede asignar una identificación (ID) y utilizar código script para alterar la página en la que están incluidos.

La siguiente tabla muestra la codificación de una página ASP utilizando código DHTML.

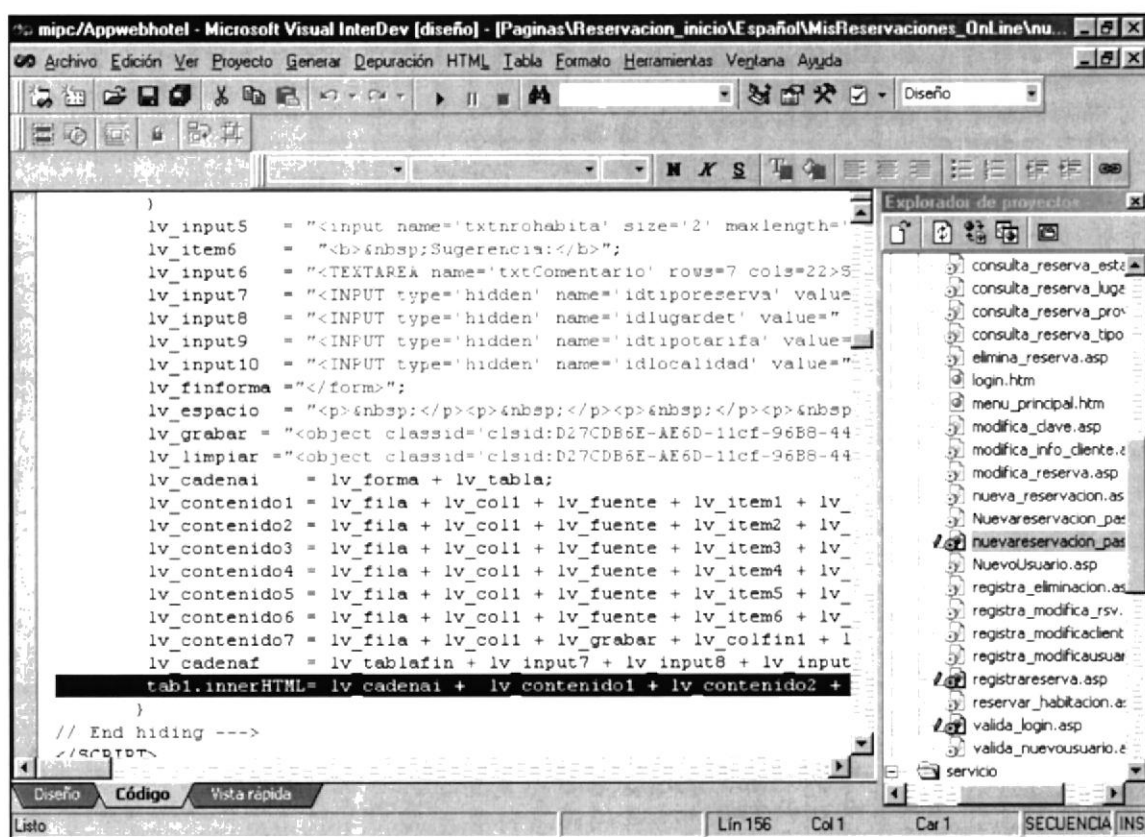


Figura 7.2. Codificación de una página Web dinámica

7.5. SERVER SCRIPT – ASP.

Los programas Microsoft Active Server Pages (ASP), son una serie de archivos con extensión ASP y lo que realizan estos programas durante su ejecución es la generación dinámica del código HTML, implementando la lógica de presentación. Entre los programas ASP desarrollados en nuestra aplicación tenemos los siguientes:

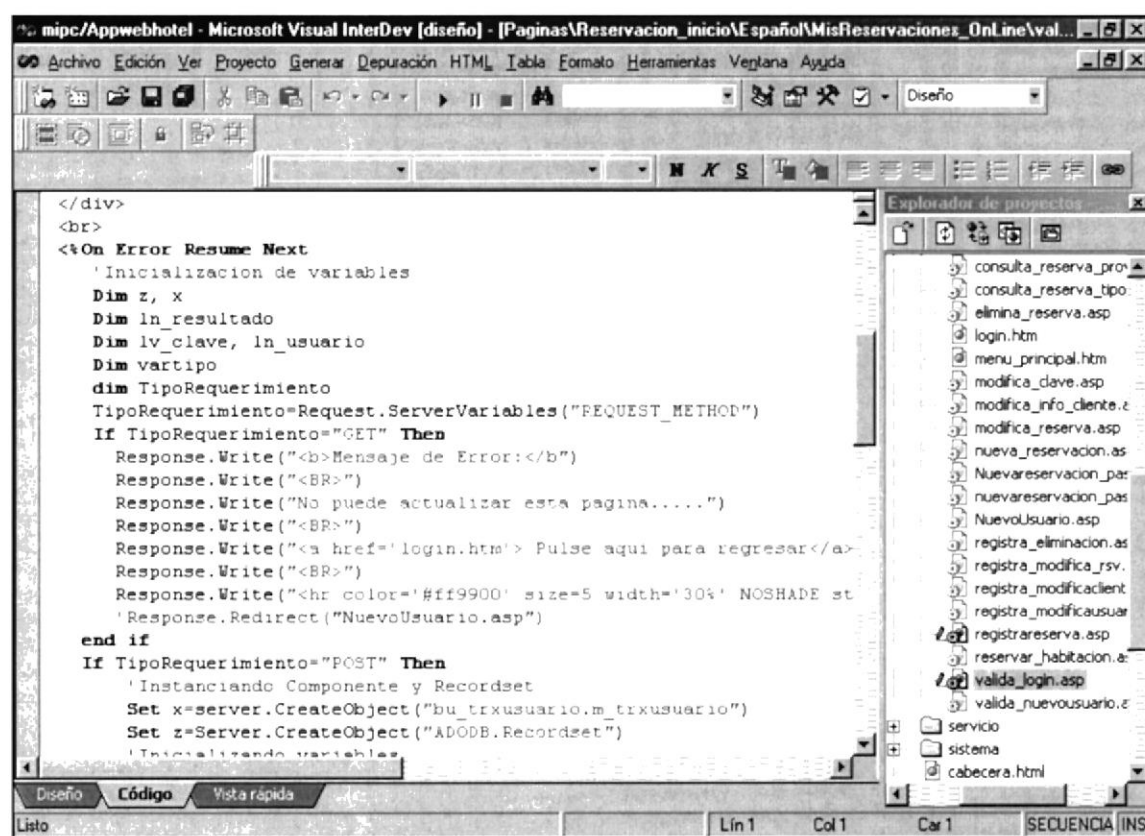


Figura 7.3. Codificación de una página Web ASP

7.6. CSS.

Las hojas CSS permiten definir un conjunto de reglas de estilo sobre las páginas HTML, esta herramienta permite establecer un control más generalizado sobre el estilo o formato de los elementos de una página o grupos de páginas. Entre muchas cualidades, las CSS proveen un mecanismo sencillo que permite a navegadores que no las pueden implementar.

Entre los estilos más utilizados tenemos:

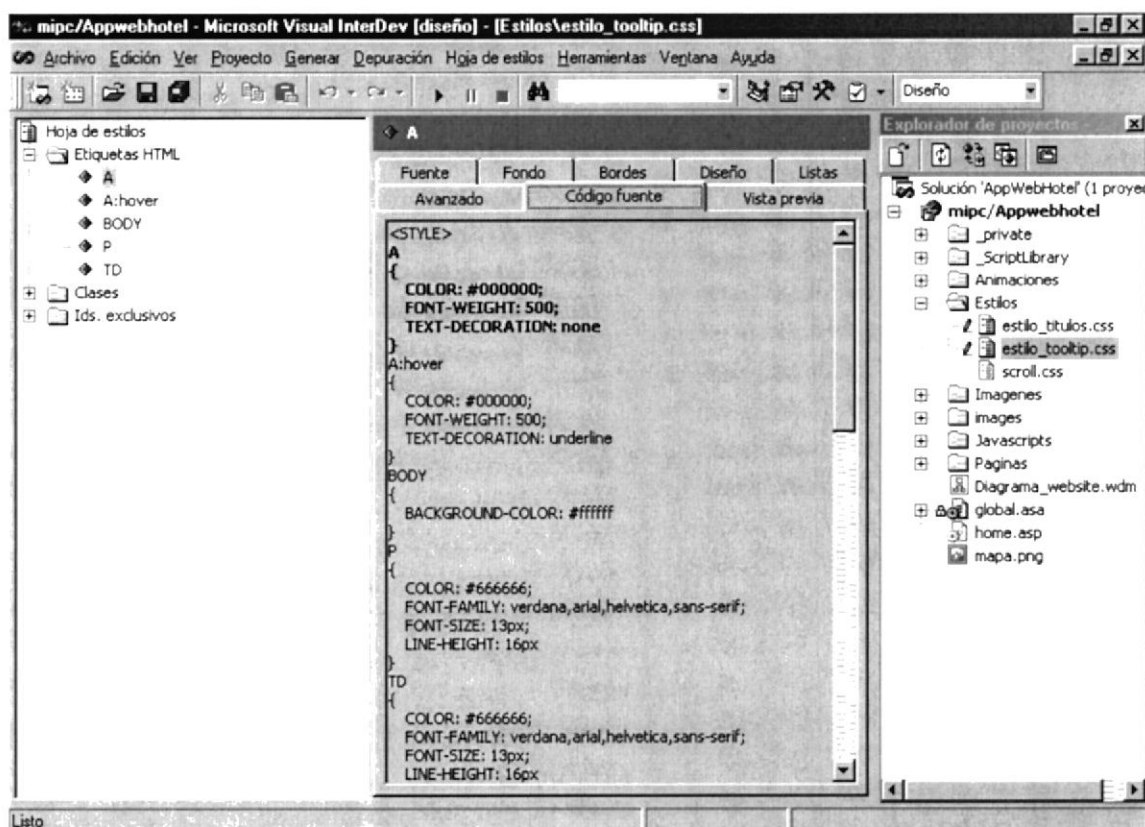


Figura 7.4. Codificación de un CSS

7.7. Comunicación con Componentes de Negocios.

Para la comunicación entre las páginas HTML y ASP con los componentes de negocio tenemos HTTP (HiperText Transfer Protocol), es un protocolo a nivel de la capa de aplicación dentro del Modelo OSI, empleado para la interacción entre el browser y el Web Server. El Usuario del sistema debe emplear un Browser para invocar al Web Server que es el encargado de generar dinámicamente el código HTML que implementa la Lógica de Presentación.

7.8. Proceso Servidor Web.

El Proceso Servidor Web es parte fundamental dentro de la arquitectura de la aplicación, para nuestra aplicación el producto que se ha utilizado es Microsoft Internet Information Server (IIS), véase Apéndice E. El Proceso Servidor Web generará la Lógica de Presentación y será además el encargado de enviar los requerimientos al Microsoft Transaction Server para permitir la ejecución de los componentes que implementan la Lógica de Negocio y la Lógica de Acceso a Datos.

El Usuario que desee usar el sistema debe “levantar” el Web Browser, en este caso puede ser Microsoft Internet Explorer o Netscape Navigator y luego digitar la siguiente dirección URL:

- Si es dentro de una Intranet: <http://mipc/appwebhotel/>
- Si es con un dominio: www.misreservaciones.com

El Web Browser es el proceso cliente y envía requerimientos HTTP al proceso servidor que es el Web Server. La solicitud es por una página estática HTML o por un programa que se encargue de generar dinámicamente el código HTML que posteriormente será enviado al Browser, para el desarrollo de esta aplicación nosotros hemos utilizado Microsoft Active Server Pages (ASP), para más información véase Apéndice E.

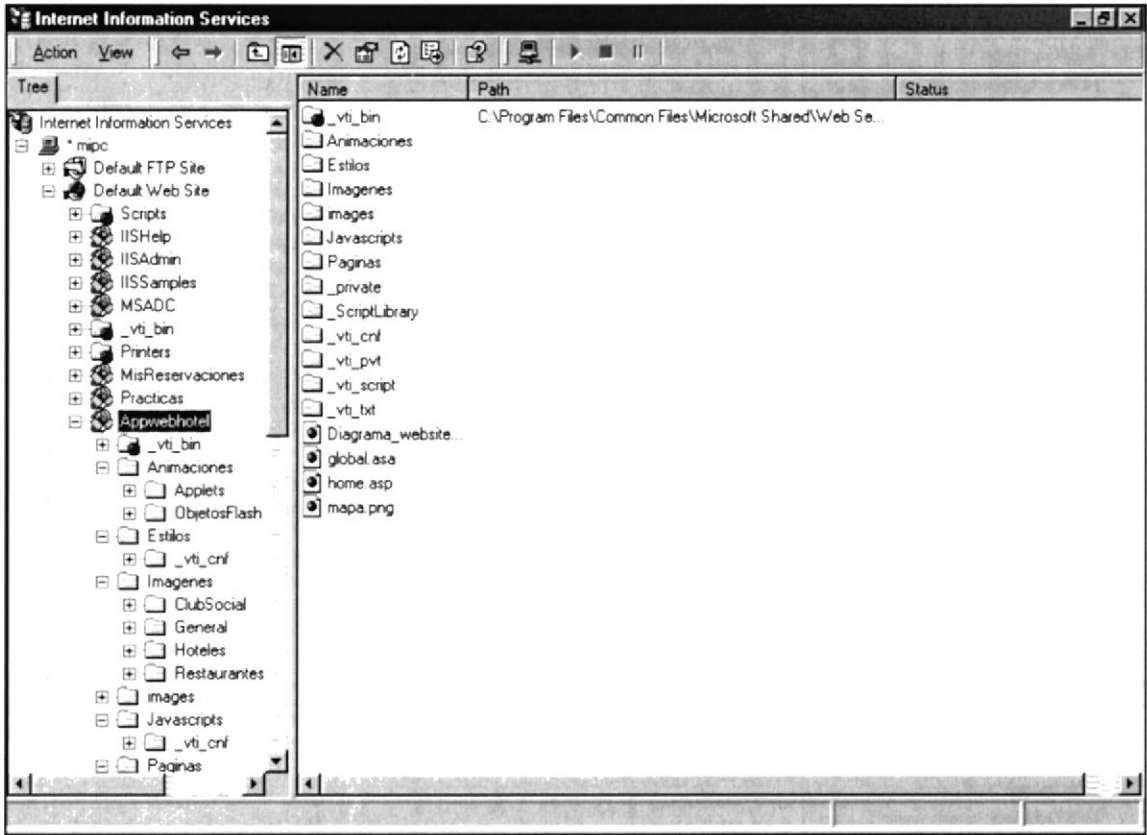


Figura 7.5. Administrador del Website (Internet Information Server–IIS)

CONCLUSIONES

Al concluir este proyecto, hemos pensado sus autores que ha mejorado con el presente proyecto, **la calidad y la eficacia** de las comunicaciones, ya que al utilizar un portal tan grande como lo es el Internet para un fin tan empleado, hemos conseguido varias ventajas competitivas sobre el procedimiento común empleado, entre estas ventajas destacan.

Costo Hora – Hombre: Ya que al no interactuar directamente con otra persona, se reducen a cero los costos por reservaciones directas, ya no es necesario que nuestros usuarios del portal dispongan de una o varias personas junto a un teléfono permanentemente para atender los requerimientos del cliente. Ni tiempo de máquina y sus sucesivos gastos (energía eléctrica, mantenimiento, etc.), adecuación del lugar de trabajo, etc.

Datos confiables para nuestros clientes: Nuestro cliente siempre va a contar con datos confiables y actualizados, así entonces reduciremos al mínimo posible los errores que comúnmente puedan darse por duplicación y olvido de reservaciones de diversos tipos.

Facilidad y Flexibilidad: Ya que nuestro cliente no tendrá que limitarse a un horario de atención para hacer sus reservaciones, ya que tendrá un portal disponible las 24 horas del día, desde cualquier punto geográfico sin que le cueste más.

Claro que, todavía el mercado de la Internet a pesar de su amplia cobertura y crecimiento, no es accesible para todos, y por supuesto el manejo de la tecnología no está tampoco al alcance de todos. Otra desventaja del mismo es sin duda las costumbres de la gente, ya que todavía la gente no se acostumbra a este medio y prefieren un servicio personalizado.

RECOMENDACIONES

Por todo lo mencionado, recomendamos el uso de las tecnologías anteriormente explicadas, por su alto rendimiento, escalabilidad, soporte a un mayor número de usuarios, seguridad en las transacciones, fácil mantenimiento de las aplicaciones y reutilización de código.

El uso del sistema, es recomendado tanto para lugares que posean una infraestructura tecnológica, como para aquellos que no la posean, ya que gracias a que nosotros, contamos con tecnología disponible, que puede brindar el mismo servicio, tanto a los unos, como a los otros.

BIBLIOGRAFÍA

Para referencia del mercado a nivel nacional e internacional visitamos algunos sitios de la World Wide Web, entre los que podemos anotar:

a. Artículo en el Internet.

1. <http://www.hiltoncolon.com>, Website del Hotel Hilton Colón.
2. <http://www.grandhotelguayaquil.com>, Website del Grand Hotel Guayaquil.
3. <http://www.microsoft.com>, Website de Microsoft.
4. <http://www.orohotels.com>, Website de la Cadena Hotelera Orohotels.

b. Libro con edición.

5. **Robert Orfali**, Cliente – Servidor. Guía de Supervivencia. (2da. Edición).
6. **Nicolás Evans**, Microsoft Visual Interdev 6.0.
7. **Ted Patison**, Programming Distributed Applications with COM and Microsoft Visual Studio.
8. **Mary Kirthand**, Designing Component Application.

c. Ayuda en línea.

Apéndice **A**

ARQUITECTURA CLIENTE / SERVIDOR

APÉNDICE A

Arquitectura Cliente/Servidor

Con la proliferación de ordenadores personales de bajo coste en el mercado, los recursos de sistemas de información existentes en cualquier organización se pueden distribuir entre ordenadores de diferentes tipos: ordenadores personales de gama baja, media y alta, estaciones de trabajo, miniordenadores o incluso grandes ordenadores.

El concepto de cliente/servidor proporciona una forma eficiente de utilizar todos estos recursos de máquina de tal forma que la seguridad y fiabilidad que proporcionan los entornos mainframe se traspasa a la red de área local. A esto hay que añadir la ventaja de la potencia y simplicidad de los ordenadores personales.

La arquitectura cliente/servidor es un modelo para el desarrollo de sistemas de información en donde las transacciones se dividen en procesos independientes que cooperan entre sí para intercambiar información, servicios o recursos. Se denomina cliente al proceso que inicia el diálogo o solicita los recursos y servidor al proceso que responde a las solicitudes.

En este modelo las aplicaciones se dividen de forma que el servidor contiene la parte que debe ser compartida por varios usuarios, y en el cliente permanece sólo lo particular de cada usuario.

Los clientes realizan generalmente funciones como:

- Manejo de la interfaz de usuario.
- Captura y validación de los datos de entrada.
- Generación de consultas e informes sobre las bases de datos.
- Por su parte los servidores realizan, entre otras, las siguientes funciones:
 - Gestión de periféricos compartidos.
 - Control de accesos concurrentes a bases de datos compartidas.
 - Enlaces de comunicaciones con otras redes de área local o extensa.

Siempre que un cliente requiere un servicio lo solicita al servidor correspondiente y éste le responde proporcionándolo. Normalmente, pero no necesariamente, el cliente y el servidor están ubicados en distintos procesadores. Los clientes se suelen situar en ordenadores personales y/o estaciones de trabajo y los servidores en procesadores departamentales o de grupo.

Entre las principales características de la arquitectura cliente/servidor se pueden destacar las siguientes:

- El servidor presenta a todos sus clientes una interfaz única y bien definida.

- El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.
- El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.
- Los cambios en el servidor implican pocos o ningún cambio en el cliente.

Características funcionales

Esta arquitectura se puede clasificar en cinco niveles, según las funciones que asumen el cliente y el servidor, tal y como se puede ver en el siguiente diagrama:

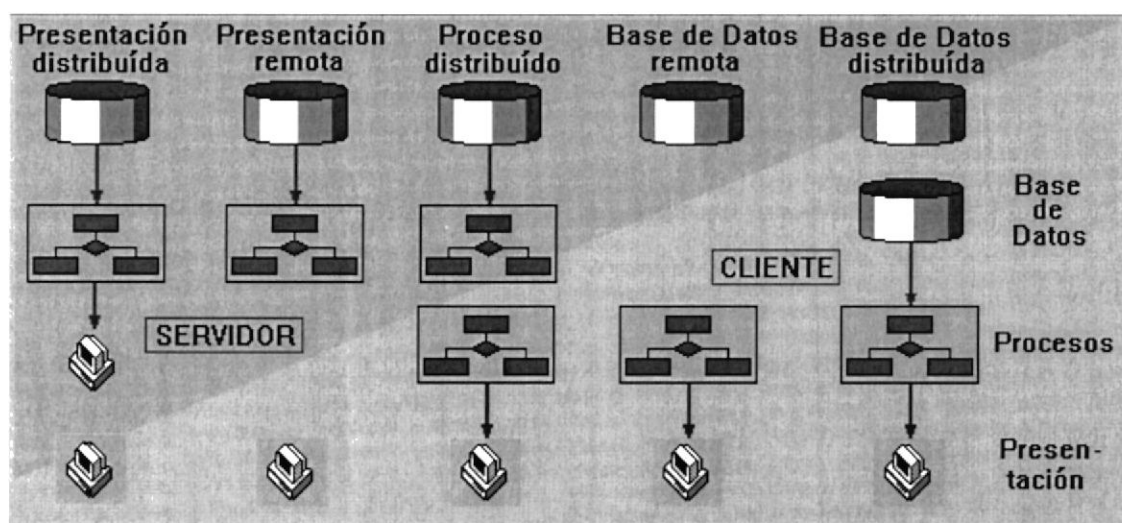


Figura A.1. Esquema Cliente / Servidor

En el primer nivel el cliente asume parte de las funciones de presentación de la aplicación, ya que siguen existiendo programas en el servidor dedicados a

esta tarea. Dicha distribución se realiza mediante el uso de productos para el "maquillaje" de las pantallas del mainframe. Esta técnica no exige el cambio en las aplicaciones orientadas a terminales, pero dificulta su mantenimiento. Además, el servidor ejecuta todos los procesos y almacena la totalidad de los datos. En este caso se dice que hay una presentación distribuida o embellecimiento.

En el segundo nivel la aplicación está soportada directamente por el servidor, excepto la presentación que es totalmente remota y reside en el cliente. Las terminales del cliente soportan la captura de datos, incluyendo una validación parcial de los mismos y una presentación de las consultas. En este caso se dice que hay una presentación remota.

En el tercer nivel la lógica de los procesos se divide entre los distintos componentes del cliente y del servidor. El diseñador de la aplicación debe definir los servicios y las interfases del sistema de información de forma que los papeles de cliente y servidor sean intercambiables, excepto en el control de los datos que es responsabilidad exclusiva del servidor. En este tipo de situaciones se dice que hay un proceso distribuido o cooperativo.

En el cuarto nivel el cliente realiza tanto las funciones de presentación como los procesos. Por su parte, el servidor almacena y gestiona los datos que permanecen en una base de datos centralizada. En esta situación se dice que hay una gestión de datos remota.

En el quinto y último nivel, el reparto de tareas es como en el anterior y además el gestor de base de datos divide sus componentes entre el cliente y el servidor. Las interfases entre ambos están dentro de las funciones del gestor de datos y, por lo tanto, no tienen impacto en el desarrollo de las aplicaciones. En este nivel se da lo que se conoce como bases de datos distribuidas.

Características físicas

El diagrama del punto anterior da una idea de la estructura física de conexión entre las distintas partes que componen una arquitectura cliente / servidor. La idea principal consiste en aprovechar la potencia de los ordenadores personales para realizar sobre todo los servicios de presentación y, según el nivel, algunos procesos o incluso algún acceso a datos locales. De esta forma se descarga al servidor de ciertas tareas para que pueda realizar otras más rápidamente.

También existe una plataforma de servidores que sustituye al ordenador central tradicional y que da servicio a los clientes autorizados. Incluso a veces el antiguo ordenador central se integra en dicha plataforma como un servidor más. Estos servidores suelen estar especializados por funciones (seguridad, cálculo, bases de datos, comunicaciones, etc.), aunque, dependiendo de las dimensiones de la instalación se pueden reunir en un servidor una o varias de estas funciones.

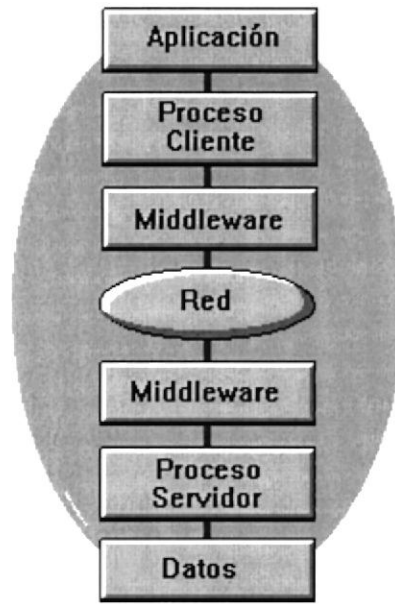


Figura A.2. Características Físicas del modelo Cliente / Servidor

Las unidades de almacenamiento masivo en esta arquitectura se caracterizan por incorporar elementos de protección que evitan la pérdida de datos y permiten multitud de accesos simultáneos (alta velocidad, niveles RAID, etc.).

Características lógicas

Una de las principales aportaciones de esta arquitectura a los sistemas de información es la interfaz gráfica de usuario. Gracias a ella se dispone de un manejo más fácil e intuitivo de las aplicaciones mediante el uso de un dispositivo tipo ratón. En esta arquitectura los datos se presentan, editan y validan en la parte de la aplicación cliente.

En cuanto a los datos, cabe señalar que en la arquitectura cliente/servidor se evitan las duplicidades (copias y comparaciones de datos), teniendo siempre una imagen única y correcta de los mismos disponible en línea para su uso inmediato.

Todo esto tiene como fin que el usuario de un sistema de información soportado por una arquitectura cliente/servidor trabaje desde su estación de trabajo con distintos datos y aplicaciones, sin importarle dónde están o dónde se ejecuta cada uno de ellos.

Ventajas e inconvenientes

Ventajas

- Aumento de la productividad:
 - Los usuarios pueden utilizar herramientas que le son familiares, como hojas de cálculo y herramientas de acceso a bases de datos.
 - Mediante la integración de las aplicaciones cliente/servidor con las aplicaciones personales de uso habitual, los usuarios pueden construir soluciones particularizadas que se ajusten a sus necesidades cambiantes.
 - Una interfaz gráfica de usuario consistente reduce el tiempo de aprendizaje de las aplicaciones.

- Reducción de los costos de operación:
 - Permiten un mejor aprovechamiento de los sistemas existentes, protegiendo la inversión. Por ejemplo, la compartición de servidores (habitualmente caros) y dispositivos periféricos (como impresoras) entre máquinas clientes permite un mejor rendimiento del conjunto.
 - Proporcionan un mejor acceso a los datos. La interfaz de usuario ofrece una forma homogénea de ver el sistema, independientemente de los cambios o actualizaciones que se produzcan en él y de la ubicación de la información.
 - El movimiento de funciones desde un ordenador central hacia servidores o clientes locales origina el desplazamiento de los costes de ese proceso hacia máquinas más pequeñas y por tanto, más baratas.
- Mejora en el rendimiento de la red:
 - Las arquitecturas cliente/servidor eliminan la necesidad de mover grandes bloques de información por la red hacia los ordenadores personales o estaciones de trabajo para su proceso. Los servidores controlan los datos, procesan peticiones y después transfieren sólo los datos requeridos a la máquina cliente. Entonces, la máquina cliente presenta los datos al usuario mediante interfases amigables. Todo esto reduce el tráfico de la

red, lo que facilita que pueda soportar un mayor número de usuarios.

- Tanto el cliente como el servidor pueden escalarse para ajustarse a las necesidades de las aplicaciones. Las UCPs utilizadas en los respectivos equipos pueden dimensionarse a partir de las aplicaciones y el tiempo de respuesta que se requiera.
- La existencia de varias UCPs proporciona una red más fiable: un fallo en uno de los equipos no significa necesariamente que el sistema deje de funcionar.
- En una arquitectura como ésta, los clientes y los servidores son independientes los unos de los otros con lo que pueden renovarse para aumentar sus funciones y capacidad de forma independiente, sin afectar al resto del sistema.
- La arquitectura modular de los sistemas cliente/servidor permite el uso de ordenadores especializados (servidores de base de datos, servidores de ficheros, estaciones de trabajo para CAD, etc.).
- Permite centralizar el control de sistemas que estaban descentralizados, como por ejemplo la gestión de los ordenadores personales que antes estuvieran aislados.

Inconvenientes

- Hay una alta complejidad tecnológica al tener que integrar una gran variedad de productos.
- Requiere un fuerte rediseño de todos los elementos involucrados en los sistemas de información (modelos de datos, procesos, interfaces, comunicaciones, almacenamiento de datos, etc.). Además, en la actualidad existen pocas herramientas que ayuden a determinar la mejor forma de dividir las aplicaciones entre la parte cliente y la parte servidor.
- Es más difícil asegurar un elevado grado de seguridad en una red de clientes y servidores que en un sistema con un único ordenador centralizado.
- A veces, los problemas de congestión de la red pueden degradar el rendimiento del sistema por debajo de lo que se obtendría con una única máquina (arquitectura centralizada). También la interfaz gráfica de usuario puede a veces reducir el tiempo de funcionamiento de la aplicación.
- El quinto nivel de esta arquitectura (bases de datos distribuidas) es técnicamente muy complejo y en la actualidad hay muy pocas implantaciones que garanticen un funcionamiento totalmente eficiente.

Fases de implantación

Una arquitectura cliente/servidor debe mostrar los sistemas de información no como un cliente que accede a un servidor corporativo, sino como un entorno que ofrece acceso a una colección de servicios. Para llegar a este estado pueden distinguirse las siguientes fases de evolución del sistema:

Fase de Iniciación

Esta etapa se centra sobre todo en la distribución física de los componentes entre plataformas. Los dos tipos de plataforma son:

- Una plataforma cliente para la presentación (generalmente un ordenador personal de sobremesa).
- Una plataforma servidora (como por ejemplo el servidor de una base de datos relacional) para la ejecución de procesos y la gestión de los datos.

Un ejemplo sería el de una herramienta de consulta que reside en un ordenador personal a modo de cliente y que genera peticiones de datos que van a través de la red hasta el servidor de base de datos. Estas peticiones se procesan, dando como resultado un conjunto de datos que se devuelven al cliente.

En esta fase pueden surgir los siguientes problemas:

- Cómo repartir la lógica de la aplicación entre las plataformas cliente y servidor de la forma más conveniente.
- Cómo gestionar la arquitectura para que permita que cualquier cliente se conecte con cualquier servidor.

Fase de Proliferación

La segunda etapa de una arquitectura cliente/servidor se caracteriza por la proliferación de plataformas clientes y servidoras. Ahora, el entorno para la interacción entre clientes y servidores se hace mucho más complejo. Puede hacerse una distinción entre:

- Datos de servidores a los que se accede a través de una red de área extensa (WAN) y,
- Datos a los que se accede a través de una red de área local (RAL).

Los mecanismos de conexión son muy variados y suelen ser incompatibles.

En esta fase los problemas que se pueden plantear son:

- La gestión de accesos se convierte en crítica y compleja debido a la estructura del organismo donde se está implantando la arquitectura. El mercado ofrece algunas soluciones que mejoran

la interoperabilidad y que se basan en conexiones modulares que utilizan entre otros:

- Drivers en la parte cliente
 - Pasarelas (gateways) a bases de datos
 - Especificaciones de protocolos cliente/servidor, etc.
- Los requisitos de actualización de datos pasan a formar parte de los requisitos solicitados al sistema cliente/servidor. Ahora no sólo se consultan datos, sino que se envían peticiones para actualizar, insertar y borrar datos.

Fase de Control

En esta fase se consolidan los caminos de acceso desde una plataforma cliente particular a una plataforma servidor particular.

Los conceptos en los que se debe poner especial énfasis son los siguientes:

- **Transparencia en la localización.** Significa que la aplicación cliente no necesita saber nada acerca de la localización (física o lógica) de los datos o los procesos. La localización de los recursos debe estar gestionada por servidores y estar representada en las plataformas adecuadas de forma que se facilite su uso por parte de las plataformas cliente.

- **Gestión de copias.** El sistema se debe configurar de forma que se permita copiar la información (datos o procesos) de los servidores.
- **Especialización de los equipos servidores en servidores de bases de datos o en servidores de aplicaciones.** Los servidores de bases de datos continúan ofreciendo servicios orientados a datos a través de llamadas SQL o a través de procedimientos almacenados. En cualquier caso, los servicios se orientan a mantener la integridad de los datos. Por otro lado, los servidores de aplicaciones se centran en los procesos implementando partes de la lógica de la aplicación en la parte servidor.

Fase de Integración

Esta etapa se caracteriza por el papel conjunto que juegan la gestión de accesos, la gestión de copias y la gestión de recursos. La gestión de la información se debe realizar de forma que se pueda entregar la información controlada por los servidores que contienen los datos a las plataformas clientes que los requieran. El concepto en que se basa este tipo de gestión es la distinción entre dos tipos de datos: datos de operación y datos de información. Para ajustarse a los posibles cambios en los procesos, los datos de operación varían continuamente mientras que los datos de información son invariables porque son de

naturaleza histórica y se obtienen tomando muestras en el tiempo de los datos de operación.

Fase de Madurez

Esta es la etapa final de una arquitectura cliente/servidor. Se caracteriza por una visión más flexible de las plataformas físicas del sistema que se contemplan como una única unidad lógica. Este estado también se caracteriza porque la tecnología cliente/servidor se ha generalizado en la empresa. Ya no es un problema saber qué componentes se distribuyen en qué plataformas, porque los recursos se pueden redistribuir para equilibrar la carga de trabajo y para compartir los recursos de información. Lo fundamental aquí es saber quién ofrece qué servicios. Para ello es necesario distinguir qué tipo de servicios y recursos se demandan y conocer las características de esta arquitectura basada en servicios.

En la fase de integración veíamos que se establecía una distinción entre datos de operación y datos de información histórica. Por contra, en un entorno de operación cliente/servidor que se encuentre en la fase de madurez, lo interesante es distinguir entre dos nuevos términos: organismo y grupo de trabajo. Esta distinción se establece basándose en sus diferencias organizativas. El grupo de trabajo es el entorno en el que grupos organizados de personas se centran en tareas específicas de la actividad del organismo al que pertenecen. Estos equipos de personas requieren una información

propia y unas reglas de trabajo particulares, que pueden ser diferentes de las del organismo en su globalidad.

Una arquitectura basada en servicios es la que se contempla como una colección de consumidores de servicios poco relacionados entre sí y los productores de dichos servicios. La utilización de este tipo de arquitectura permite pensar en nuevos retos de diseño:

- Desarrollo de componentes reutilizables entre distintas aplicaciones y distintos grupos de trabajo
- Desarrollo de aplicaciones distribuidas
- Gestión del desarrollo de aplicaciones entre distintos equipos, etc.

Criterios de utilización

El mercado de los sistemas cliente/servidor está marcando nuevos caminos porque:

- La información puede ahora residir en redes de ordenadores personales.
- Los usuarios pueden tener un mayor acceso a los datos y a la capacidad de proceso.

El marketing también juega un papel importante. Muchos sistemas que se denominan cliente/servidor en realidad distan bastante de serlo y muchas aplicaciones aseguran ser tan fiables como sus homólogas en el host.

En realidad, el cambio hacia tecnologías cliente/servidor está aún en sus comienzos, pero de ninguna manera debe ignorarse.

La forma de asegurar la futura utilización productiva de sistemas cliente/servidor, asumiendo un bajo riesgo, debe considerar:

- Comenzar por el downsizing: utilizar redes de área local y familiarizar a los usuarios con el uso de ordenadores personales.
- Estudiar las herramientas cliente/servidor que se encuentren disponibles y aquellas que se encuentren en fase de desarrollo; la mayoría están basadas en algún sistema de gestión de base de datos en red local.
- Permitir el acceso de los usuarios a los datos de la organización conectando las redes locales entre sí.
- Añadir interfaces de usuario amigables al equipo lógico del ordenador central y desarrollar prototipos.

Una organización tiene que decidir cuándo y cómo debe migrar en su caso, hacia un entorno cliente/servidor, teniendo en cuenta la evolución de las herramientas cliente/servidor y desarrollando una estrategia que se base en los estándares predominantes en el mercado.

Relación con otros conceptos

Arquitectura cliente/servidor y *downsizing*

Muchas organizaciones están transportando sus aplicaciones a plataformas más pequeñas (*downsizing*) para conseguir la ventaja que proporcionan las nuevas plataformas físicas más rentables y la arquitectura cliente/servidor. Este transporte siempre supone un coste, debido a la necesidad de rediseñar las aplicaciones y de re-entrenar a los usuarios en los nuevos entornos.

Independencia de Bases de Datos

Las arquitecturas cliente/servidor permiten aprovechar los conceptos de cliente y servidor para desarrollar aplicaciones que accedan a diversas bases de datos de forma transparente. Esto hace viable cambiar la aplicación en la parte servidora, sin que la aplicación cliente se modifique. Para que sea posible desarrollar estas aplicaciones es necesaria la existencia de un estándar de conectividad abierta que permita a los ordenadores personales y estaciones de trabajo acceder de forma transparente a bases de datos corporativas heterogéneas.

Relación con los Sistemas Abiertos

Las arquitecturas cliente/servidor se asocian a menudo con los sistemas abiertos, aunque muchas veces no hay una relación directa entre ellos. De hecho, muchos sistemas cliente/servidor se pueden aplicar en entornos propietarios.

En estos entornos, el equipo físico y el lógico están diseñados para trabajar conjuntamente, por lo que, en ocasiones se pueden realizar aplicaciones cliente/servidor de forma más sencilla y fiable que en los entornos que contienen plataformas heterogéneas.

El problema surge de que los entornos propietarios ligan al usuario con un suministrador en concreto, que puede ofrecer servicios caros y limitados. La independencia del suministrador que ofrecen los entornos de sistemas abiertos, crea una competencia que origina mayor calidad a un menor precio.

Pero, por otra parte, debido a la filosofía modular de los sistemas cliente/servidor, éstos se utilizan muchas veces en entornos de diferentes suministradores, adecuando cada máquina del sistema a las necesidades de las tareas que realizan. Esta tendencia está fomentando el crecimiento de las interfaces gráficas de usuario, de las bases de datos y del *software* de interconexión.

Debido a esto, se puede afirmar que los entornos cliente/servidor facilitan el movimiento hacia los sistemas abiertos. Utilizando este tipo de entornos, las organizaciones cambian sus viejos equipos por nuevas máquinas que pueden conectar a la red de clientes y servidores.

Los suministradores, por su parte, basan uno de los puntos clave de sus herramientas cliente/servidor en la interoperabilidad.

Relación con Orientación a Objetos

No hay una única forma de programar aplicaciones cliente/servidor; sin embargo, para un desarrollo rápido de aplicaciones cliente/servidor y para obtener una reducción importante de costos, la utilización de la tecnología cliente/servidor puede considerarse en conjunción con la de orientación a objetos.

Apéndice **B**

PROGRAMACIÓN EN EL WEB

APÉNDICE B

PROGRAMACIÓN EN EL WEB

Existen diferentes opciones para el desarrollo de página web, este apéndice le explicará acerca de estas opciones.

HTML

El *HTML* (*Hyper Text Markup Language*) es un sistema para estructurar documentos. Estos documentos pueden ser mostrados por los visores de páginas Web en Internet, como Netscape, Mosaic o Microsoft Explorer. Por el momento no existe un estándar de *HTML* ya que tanto Netscape como Microsoft se empeñan en incluir directivas que solo funcionan con sus respectivos navegadores. De cualquier manera existen diferentes revisiones o niveles de estandarización, el 1.0, el 2.0 y el 3.0, lo que produce que algunos visores no "comprendan" en su totalidad el contenido de un documento. En este manual se ha utilizado la revisión 3.0 de HTML. Esto quiere decir que algunas de las órdenes de HTML que aquí se indican puede que no sean reconocidas por algunos visores de páginas Web. Netscape 2.x y Microsoft Explorer 3.x reconocen prácticamente todas las órdenes HTML vistas en este apéndice.

Básicamente, el HTML consta de una serie de órdenes o directivas, que indican al visor que estamos utilizando, la forma de representar los

elementos (texto, gráficos, etc...) que contenga el documento. En este apéndice nos referiremos a estas órdenes con la palabra "directiva".

Las directivas de HTML pueden ser de dos tipos, cerradas o abiertas. Las directivas cerradas son aquellas que tienen una palabra clave que indica el principio de la directiva y otra que indica el final. Entre la directiva inicial y la final se pueden encontrar otras directivas. Las directivas abiertas constan de una sola palabra clave. Para diferenciar las directivas del resto del texto del documento se encierran entre los símbolos < y >. Las directivas cerradas incluyen el carácter / antes de la palabra clave para indicar el final de la misma. Una directiva puede contener "parámetros". Estos parámetros se indican a continuación de la palabra clave de la directiva.

Ejemplos :

Directiva cerrada

<CENTER> Mi página Web </CENTER>

Directiva abierta

<HR>

Directiva con parámetros

<BODY bgcolor="#FFFFFF"> </BODY>

Los ficheros que contienen documentos HTML suelen tener la extensión *.html* o *.htm*. En este manual se han fijado los siguientes criterios a la hora de escribir la sintaxis de las directivas de HTML:

1. Las directivas se indican en letra mayúscula y en negrilla.

2. Los parámetros de las directivas se indican en letra minúscula y negrilla.
3. El resto de elementos se indican en letra normal.
4. Las palabras a resaltar en el texto se indican en cursiva y negrilla.

Estructura básica de un documento HTML

Un documento escrito en HTML contendría básicamente las siguientes directivas :

<code><HTML></code>	Indica el inicio del documento.
<code><HEAD></code>	Inicio de la cabecera.
<code><TITLE></code>	Inicio del título del documento.
<code></TITLE></code>	Final del título del documento.
<code></HEAD></code>	Final de la cabecera del documento.
<code><BODY></code>	Inicio del cuerpo del documento.
<code></BODY></code>	Final del cuerpo del documento.
<code></HTML></code>	Final del documento.

El documento se hallará situado en algún ordenador al que se pueda acceder a través de Internet. Para indicar la situación del documento en Internet se utiliza el *URL (Uniform Resource Locator)*. El *URL* es el camino que ha de seguir nuestro visor a través de Internet para acceder a un determinado

recurso, bien sea una página Web, un fichero, un grupo de noticias, etc. Es decir, lo que el visor de páginas Web hace es acceder a un fichero situado en un ordenador que está conectado a la red Internet. La estructura de un URL para una página Web suele ser del tipo *http://dominio/directorio/fichero*. El dominio indica el nombre del ordenador al que accedemos, el directorio es el nombre del directorio de ese ordenador y fichero el nombre del fichero que contiene la página Web escrita en HTML.

Cabecera del documento

La directiva `<HEAD></HEAD>` delimita la cabecera del documento. Dentro de la cabecera es importante definir el título de la página por medio de la directiva `<TITLE></TITLE>`. Este título será el que aparezca en la barra de nuestro visor de páginas Web.

Ejemplo :

```
<TITLE>La página Web de AcMark</TITLE>
```

Dentro de la cabecera de nuestro documento podemos incluir otras directivas adicionales. La directiva `<META>` indica al visor de Internet las palabras clave y contenido de nuestra página Web. Muchos de los buscadores de páginas Web de Internet (Yahoo, Lycos, etc...) utilizan el contenido de esta directiva para incluir la página en sus bases de datos. La directiva `<META>` lleva generalmente dos parámetros, `name` y `content`.

Ejemplos :

```
<META name = "Página de José" content = "Mi página personal, Música y Películas">
```

Indica al visor el nombre de la página y sus contenidos principales.

```
<META name = "keywords"content = "José música películas links España">
```

Indica al visor las palabras clave para los buscadores de Internet.

Otro uso de la directiva <META> es indicar documentos con "refresh automático". Si se indica un *URL* se sustituirá el documento por el indicado una vez transcurridos el número de segundos especificados en el tag. Si no se incluye ningún *URL* se volverá a cargar en el visor el documento en uso transcurridos los segundos indicados. Esto es útil para páginas que cambian de contenido con mucha frecuencia o para redireccionar a la persona que visita nuestra página Web a una nueva dirección donde se encuentra una versión actualizada de nuestra página Web.

Ejemplo :

```
<META http-equiv= "refresh"content = "15;URL=http://www.microsoft.com">
```

Transcurridos 15 segundos se accederá a la página Web de Microsoft.

La directiva <BASE> indica la localización de los ficheros, gráficos, sonidos, etc... a los que se hace referencia en nuestra página Web. Si no se incluye esta directiva el visor entiende que dichos elementos se encuentran en el mismo lugar donde se encuentra nuestra página Web.

Ejemplo :

```
<BASE href = "http://www.jet.es/jose/">
```

Cuerpo del documento

La directiva <BODY></BODY> indica el inicio y final de nuestra página Web. Será entre el inicio y el final de esta directiva donde pongamos los contenidos de nuestra página, textos, gráficos, enlaces, etc. Esta directiva tiene una serie de parámetros opcionales que nos permiten indicar la "apariencia" global del documento :

background= "nombre de fichero gráfico"

Indica el nombre de un fichero gráfico que servirá como "fondo" de nuestra página. Si la imagen no rellena todo el fondo del documento, esta será reproducida tantas veces como sea necesario.

bgcolor = "código de color"

Indica un color para el fondo de nuestro documento. Se ignora si se ha usado el parámetro background.

text = "código de color"

Indica un color para el texto que incluyamos en nuestro documento. Por defecto es negro.

link = "código de color"

Indica el color de los textos que dan acceso a un Hyperenlace. Por defecto es azul.

vlink = "código de color"

Indica el color de los textos que dan acceso a un Hyperenlace que ya hemos visitado con nuestro visor. Por defecto es púrpura.

El *código de color* es un número compuesto por tres pares de cifras hexadecimales que indican la proporción de los colores "primarios", *rojo*, *verde* y *azul*. El código de color se antecede del símbolo #.

Ejemplos :

#000000	Color	Negro
#FF0000	Color	Rojo
#00FF00	Color	Verde
#0000FF	Color	Azul
#FFFFFF	Color	Blanco

El primer par de cifras indican la proporción de color Rojo, el segundo par de cifras la proporción de color Verde y las dos últimas la proporción de color Azul. Cada par de cifras hexadecimales nos permiten un rango de 0 a 255. Combinando las proporciones de cada color primario obtendremos diferentes colores.

De cualquier forma la mayoría de los editores de HTML nos permiten obtener el código de color correspondiente escogiendo directamente el color de una paleta.

Espaciados y saltos de línea.

En HTML solo se reconoce un espacio entre palabra y palabra, el resto de los espacios serán ignorados por el visor.

Ejemplo	Se verá como
Esto es una frase	Esto es una frase

Tabla B.1. Reconocimiento de un espacio entre palabras

Tampoco se respetan las tabulaciones, retornos de carro, etc. Para ello existen una serie de directivas que indican estos códigos. La directiva `<PRE></PRE>` obliga al visor a visualizar el texto tal y como ha sido escrito, respetando tabulaciones, espacios, retornos de carro, etc.

Ejemplo	Se verá como
<code><PRE></code> Este texto ha sido preformateado . <code></PRE></code>	Este texto ha sido preformateado .

Tabla B.2. Reconocimiento de un espacio entre palabras

Para indicar un salto de línea se utiliza la directiva
 y para un cambio de párrafo (deja una línea en blanco en medio) se utiliza la directiva <P>.

Ejemplo	Se verá como
Este texto tiene saltos de línea y <P> de párrafo.	Este texto tiene saltos de línea y de párrafo.

Tabla B.3. Para indicar un salto de línea

La directiva <P> puede usarse también como directiva "cerrada" <P></P> indicando de esta manera los atributos de un párrafo en concreto. Cuando se usa de esta manera tiene el parámetro ALIGN que indica al visor la forma de "justificar" el párrafo. Los valores posibles de este parámetro son LEFT, RIGHT y CENTER, estando aún en estudio el valor JUSTIFY.

Ejemplo	Se verá como
<P Align=right>Este es un ejemplo de un párrafo de texto justificado a la derecha.</P>	Este es un ejemplo de párrafo de texto justificado a la derecha
<P Align=center>Este es un ejemplo de párrafo de texto centrado.</P>	Este es un ejemplo de párrafo de texto centrado

Tabla B.4. Ejemplo de la Directiva ALIGN

La directiva <HR> muestra una línea horizontal de tamaño determinable. Tiene los siguientes parámetros opcionales :

align = posición

Alinea la línea a la izquierda (left), a la derecha (right) o la centra (center).

noshade

Nos muestra sombra, evitando el efecto en tres dimensiones.

size = número

Indica el grosor de la línea en pixels.

width = num / %

Indica el ancho de la línea en tanto por ciento en función del ancho de la ventana del visor. También se puede especificar un número que indicaría el ancho de la línea en pixels.

Ejemplo :

<HR align= center size= 20 width= 50%>



La directiva <HR> sin ningún parámetro mostraría una línea horizontal que ocuparía todo el ancho de la página. Estas líneas sencillas son las que usted puede observar en este manual para separar las diferentes secciones

Cabeceras. En un documento de HTML se pueden indicar seis tipos de cabeceras (tamaños de letra) por medio de las directivas <H1><H2><H3><H4><H5> y <H6>. El texto que escribamos entre el inicio y

el fin de la directiva será el afectado por las cabeceras. La cabecera <H1> será la que muestre el texto en mayor tamaño.

Ejemplo	Se vería como
<H1>Texto de Prueba</H1>	Texto de prueba
<H2>Texto de Prueba</H2>	Texto de Prueba
<H3>Texto de Prueba</H3>	Texto de Prueba
<H4>Texto de Prueba</H4>	Texto de Prueba
<H5>Texto de Prueba</H5>	Texto de Prueba
<H6>Texto de Prueba</H6>	Texto de Prueba

Tabla B.5. Ejemplo de la Directiva H1..H6

Los textos marcados como "cabeceras" provocan automáticamente un retorno de carro sin necesidad de incluir la directiva
. Por ejemplo

Ejemplo	Se vería como
<H3>Pagina de José</H3>Esta es mi página personal.	Página de José. Esta es mi página personal

Tabla B.6. Ejemplo del Retorno de Carro

Atributos del Texto. Para indicar atributos del texto (negrita, subrayado, etc.) tenemos varias directivas. Algunas de ellas no son reconocidas por determinados navegadores, es por ello que según el navegador que usted este utilizando, verá el resultado correctamente o no.

Atributo	Directiva	Ejemplo	Resultado
Negrita		Texto de prueba	Texto de prueba
Cursiva	<I></I>	<I>Texto de prueba</I>	<i>Texto de prueba</i>
Teletype	<TT></TT>	<TT>Texto de prueba</TT>	Texto de prueba
Subrayado	<U></U>	<U>Texto de prueba</U>	Texto de prueba
Tachado	<S></S>	<S>Texto de prueba</S>	Texto de prueba
Parpadeo	<BLINK></BLINK>	<BLINK>Texto de prueba</BLINK>	de Texto de prueba
Superíndice		^{Texto de prueba}	de Texto de prueba
Subíndice		_{Texto de prueba}	de Texto de prueba
Centrado	<CENTER></CENTER>	<CENTER>Texto de prueba</CENTER>	de Texto de prueba

Tabla B.7. Atributos de Texto

Por otro lado la directiva nos permite variar el tamaño, el color, y el tipo de letra de un texto determinado. Utiliza para ello los parámetros size, bgcolor y face.

size = valor

Da al texto un tamaño en puntos determinado.

size = +/- valor

Da al texto un tamaño tantas veces superior (+) o inferior (-) como indique el valor.

color = "código de color"

Escribe el texto en el color cuyo código se especifica.

face = "nombre de font"

Escribe el texto en el tipo de letra especificado. Si este tipo de letra no existe en el ordenador que "lee" la página se usara el font predeterminado del navegador.

Ejemplo	Se vería como
 Texto de prueba 	Texto de prueba

Tabla B.8. Ejemplo de la Directiva FONT.

Existen otras directivas que realizan las mismas operaciones que las antes vistas en los atributos del texto.

Directiva	Hace lo mismo que
<code></code>	<code></code>
<code><CITE></CITE></code>	<code><I></I></code>
<code><STRIKE></STRIKE></code>	<code><S></S></code>

Tabla B.9. Ejemplo de otras directivas para texto

Para incluir comentarios en la página Web se utiliza la directiva `<!-- -->`.

Ejemplo :

`<!-- Esto es un comentario sobre mi página Web -->`

Los comentarios no serán mostrados por el visor y son útiles para realizar anotaciones en el documento HTML que nos indiquen lo que estamos haciendo en una determinada parte del documento. Como veremos más adelante que la directiva de comentario nos será de utilidad para incluir código JavaScript en nuestra página Web.

JavaScript

JavaScript es un lenguaje compacto, y basado en objetos, diseñado para el desarrollo de aplicaciones cliente-servidor a través de Internet. Netscape Navigator 2.0 es capaz de interpretar sentencias JavaScript embebidas en programas CGI.

En una aplicación cliente para un browser, las sentencias JavaScript embebidas en un documento HTML pueden reconocer y responder a eventos generados por el usuario, como clicks del mouse, información en formularios y navegación de documento a documento.

Por ejemplo, se puede escribir una función JavaScript que verifique que la información ingresada por el usuario sea correcta. Sin que haya transmisión de main por la red, un documento HTML con JavaScript embebido es capaz de interpretar la información ingresada por el usuario, verificar que sea correcta y alertar al usuario en caso que no lo sea. Se puede también ejecutar archivos de audio, applet's o comunicar con una extensión de Netscape (plug-in's) en respuesta a la apertura o cierre de una página.

JavaScript y Java

El lenguaje JavaScript imita a Java, pero sin verificación estática de tipos ni restricciones fuertes en tiempo de ejecución. JavaScript soporta la mayoría de las expresiones y sintaxis válidas en Java, así como las sentencias de control de flujo de Java. En contraste con el sistema de clases definidas en tiempo de compilación en Java, Javascript trabaja en base a un simple sistema en tiempo de ejecución, el cual permite el uso de tipos de main básicos como enteros, números de punto flotante, booleanos y strings. JavaScript soporta un sistema simple de objetos basado en instancias (no en clases), que provee capacidades significativas.

Además, JavaScript soporta funciones, pero de nuevo sin ningún requerimiento especial al declararlas. Estas funciones pueden ser propiedades de objetos (aunque el usuario no puede definir clases), y se ejecutan como métodos bastante flexibles debido a que no se verifican los tipos.

JavaScript complementa a Java al exponer muchas características de las applets en Java a los autores de HTML. JavaScript puede además obtener y establecer propiedades de los applets, esto es, sólo aquellas que alteran el estado de los applets o plug-ins.

Java es un lenguaje de extensión diseñado, en particular, para ejecución rápida y seguridad de tipos, que se ve reflejada en la imposibilidad de transformar un entero en una referencia, por ejemplo.

Los programas en Java consisten exclusivamente de clases y sus métodos, lo que, junto a los requerimientos de declaraciones de clases, métodos y tipos hacen que la programación sea más compleja que en JavaScript. Otro punto importante es que la herencia que provee Java crea jerarquías muy acopladas de objetos.

En contraste, JavaScript posee el espíritu de un lenguaje pequeño y verificado dinámicamente. Estos lenguajes ofrecen herramientas de programación a una audiencia mayor, ya que son más fáciles de utilizar, y proveen de una interfase más especializada. Como consecuencia, necesitan sólo de un mínimo de requerimientos para creación de objetos.

La siguiente tabla compara y contrasta JavaScript con Java:

JavaScript	Java
Interpretado en el Cliente.	Compilado en el Servidor antes de la ejecución en el Cliente.
Basado en Objetos. El código usa objetos incorporados del sistema, pero no provee creación de clases o herencia.	Orientado a Objetos. El código consiste de clases con herencia, permitiendo crear objetos y crear jerarquías.
Código embebido en HTML.	Applets referenciadas desde HTML.
Variables y tipos de main no se declaran.	Variables y tipos de main se declaran.
Enlazado dinámico. Referencias a objetos se validan en tiempo de ejecución.	Enlazado estático. Referencias a objetos se validan en tiempo de compilación.
Ninguno puede escribir a archivos en disco.	

Tabla B.10. Comparación entre Javascript y Java

Apéndice **C**

COM / DCOM / COM+

APÉNDICE C

Tecnologías COM / DCOM Y COM+

Introducción

La programación moderna

Algunas de las modernas técnicas de programación están creando un pequeño caos en la industria del desarrollo de software. Normalmente, este caos se produce en quienes hemos programado con metodologías anteriores y que, obviamente, nos causa algunos dolores de cabeza entender las nuevas.

Con lo vertiginoso que es el ámbito de la computación, es difícil aceptar un nuevo dolor de cabeza prácticamente cada mes (o menos). Sin embargo, las nuevas tecnologías de programación no son sólo instrumentos para quebrarse el cerebro, sino que han sido pensadas para hacer más llevadera la vida, a la comunidad de desarrolladores.

La Programación Orientada a Objetos

La Programación orientada a objetos (POO) es una nueva capa de esta evolución que se basa en la ideología de "muchas hormigas juntas mueven

una montaña". En realidad, la propuesta es sencilla: crear individuos con características y facultades propias, y unirlos con un propósito común.

Trataré de redondear esto: imagine que existe un Pancho Kaputnik que tiene la habilidad de manejar la computadora. Si un empresario lo contrata, será exactamente para aprovechar sus capacidades en pro de los intereses de la empresa. De hecho, dado que Pancho Kaputnik tiene estas habilidades, será más fácil adaptarlo a las necesidades expresas que en tal rubro se tienen. Esto, por supuesto, es más fácil que entrenar a uno nuevo.

Pues, estimado lector, esto es, de manera muy general, la ideología que define la POO. El uso de entidades propias que pueden ejecutar satisfactoriamente una tarea, o aprovechar sus capacidades para adaptarlas a las necesidades de la tarea, es más fácil que crear una entidad desde el inicio.

Cada entidad se llama objeto, el cual envuelve una serie de características y acciones para llevar a cabo una tarea. Un objeto Individuo puede tener entre sus características distintivas el color de sus ojos, de su piel, de su cabello, su complexión, estatura, etcétera. Las acciones del individuo definen las cosas que puede hacer, como caminar, comer, hablar. Estas acciones pueden afectar exclusivamente al individuo que las lleva a cabo o a terceros, por ejemplo, cuando los golpee. El tercero puede reaccionar a la acción proveniente del primer individuo.

Bien, pues la POO llama Propiedades a las características, Métodos a las acciones y Eventos a las reacciones a los estímulos externos provenientes de otros objetos o del ambiente que les rodea. Tanto las Propiedades como los Métodos están envueltos en la entidad llamada Objeto. A esto se le llama Encapsulamiento u Ocultación, dado que el código inherente a cada uno está fuera de la vista del ambiente de desarrollo, bajo la ideología de "cada quién sus cosas".

La comunicación entre estos objetos se logra mediante interfases, las cuales proporcionan un medio que permite la interacción entre estos objetos. Todos aquellos que se encuentren bajo una misma interfase son polimórficos. El polimorfismo describe el efecto de poder invocar al método de un objeto, sin conocer exactamente el tipo de objeto, con lo que se logra que se ejecute la acción apropiada de ese objeto en particular.

Finalmente, la Herencia es la capacidad de poder crear otros objetos a partir de otro. ¿Recuerda al Sr. Kaputnik? Sólo se le adaptó para alguna(s) tarea(s) específica(s) de la empresa y no se tuvo que formar a un nuevo Kaputnik. La herencia es un punto clave en la POO y de suma importancia para el desarrollo de sistemas basados en POO. Microsoft Visual C++, Borland Delphi y Java son claros representantes de esta ideología. VB queda, aunque no del todo, fuera de esta clasificación.

Puntos en contra de la POO.

El soporte de la POO para el desarrollo de módulos de software tipo plug-and-play se aplica sólo a nivel del código fuente; no es un estándar binario. Esto significa que no existe ningún mecanismo estándar a través del cual los objetos puedan trabajar juntos. Los módulos de desarrollo individuales podrían requerir sus propias interfases en código fuente y así, ejercer una carga muy fuerte en tiempo y costos de desarrollo. No sólo es necesario considerar las diferentes interfases; además, la selección del lenguaje de programación se convierte en un obstáculo potencial. Un desarrollador podría desear utilizar un objeto escrito en un lenguaje no familiar, lo que requerirá, por tanto, aún más inversión en la curva de aprendizaje.

El hecho de que la compatibilidad está limitada a nivel del código fuente trae consigo que la POO no enfrenta el importante problema de las revisiones. Cuando un objeto es actualizado, revisado y reemplazado, cualquier programa que lo utilice necesita volver a compilarse y probablemente a distribuirse. Mientras que los programadores están ya acostumbrados a compilar y enlazar de nuevo cada vez que actualizan una aplicación, los problemas de la redistribución elevan el nivel de sus preocupaciones. Por ejemplo, esto trae la desagradable posibilidad de que las DLLs se sobrescriban unas a otras.

Ciertamente, el problema de la compatibilidad de software se resuelve parcialmente gracias a la POO, pero ésta no es una respuesta completa.

Este apartado de las desventajas baja un poco la moral, pero si ante todo ya se usan lenguajes como Visual C++ o Delphi, pues parte del problema está resuelto.

Bien, en esta ocasión trataremos una propuesta interesante que parece mirar un nuevo esquema a los problemas inherentes a la POO: COM.

Un objeto

Un objeto es una unidad de código y datos. Este objeto representa un grupo de variables y procedimientos relacionados que pueden ser manipulados como una sola unidad. Los procedimientos en un objeto definen sus acciones y son llamados Métodos. Las variables son usadas normalmente para establecer y regresar atributos o comportamiento y son llamadas Propiedades.

Descripción de una clase.

Si un objeto es una entidad propia, la clase es su molde. La clase sirve para definir todas las características, comportamiento y acciones de un objeto en particular. A partir de ésta es donde se detallarán los objetos que resulten de la misma. Una misma clase puede definir a varios objetos con acciones similares, aunque cambien un poco en cuanto a comportamiento y características.

Características extra de las clases.

Abstraer las nuevas técnicas de programación como entidades individuales que, en conjunto, permiten lograr un objetivo común, es la forma más sencilla de hacerse a la idea de lo que es la programación moderna. Ya se explicó qué son estas entidades y a partir de qué se crean, específicamente los objetos y las clases. Sin embargo, para el mejor aprovechamiento de las capacidades de una clase, se debe echar mano de un par de características más, definidas en el entorno de la programación orientada a objetos, pero implantadas de forma distinta en Visual Basic.

Una de estas características es la herencia. Podría decirse que esta es una de las más útiles prestaciones de la POO. Gracias a esto, podemos crear una clase basada en una existente a la que se le desea extender su funcionalidad. Con esto, podemos crear subclases. Muy práctico, sí, pero en VB no está implantada esta funcionalidad, aunque se puede emular.

Otra de las características, y que sí está completamente implantada en VB (mucho más evidente en Visual Basic 5.0), es el Polimorfismo. La mayoría de los lenguajes de programación orientada a objetos proporcionan polimorfismo mediante la herencia. Esta característica proviene del hecho de que puede llamar a un método determinado de un objeto que pertenezca a cualquier clase derivada de una superclase, sin conocer la clase a la que pertenece el objeto.

Dado que Visual Basic no utiliza la herencia para proporcionar polimorfismo, esta característica se otorga mediante múltiples interfases ActiveX. En el modelo de objetos basado en componentes (COM) que forma la

infraestructura de la especificación ActiveX, las múltiples interfases permiten que los sistemas de componentes basados en software evolucionen sin afectar el código existente. Una interfase es un conjunto de propiedades y métodos relacionados. Gran parte de la especificación ActiveX se ocupa de la implementación de interfases estándar para obtener servicios del sistema o para proporcionar funcionalidad a otros programas.

De este modo, en VB podría crear una interfase "Animal" e implantarla en las clases Ave y Perro. Después podría ejecutar el método Comer de cualquiera de los dos tipos de objetos sin conocer su tipo. En VB cada clase tiene una interfase predeterminada que tiene el mismo nombre de la clase. Una referencia a una interfase también es una referencia al objeto que la implementa. Además, un objeto puede tener múltiples interfases y seguir siendo el mismo objeto en cada una de ellas.

COM

Definición

COM (Modelo de objetos y componentes) es una solución creada por Microsoft que provee una alternativa viable a los problemas originados por la recompilación y distribución de objetos programados o componentes. Esto se logra mediante una norma de compatibilidad y reusabilidad establecida a nivel binario. Estas características requieren de un soporte a nivel del sistema operativo. COM, sin embargo, difiere del estilo del API tradicional de compartir los servicios del sistema, en que describe el método por el cual los objetos se conectan. Posterior a la conexión, COM ya no es necesario y deja de consumir recursos del sistema. Compare esto con un API, donde un sistema operativo tiene que manejar la conexión entre los componentes todo el tiempo.

Modelo de Objeto

El Modelo de objetos y componentes también establece otro factor clave, necesario para la conexión fácil entre objetos, llamado manejo de procesos. Un objeto que puede ser usado por cualquier aplicación cliente debe tener

algunos medios para manejar su propio tiempo de vida. Cuando la aplicación cliente ha dejado de utilizar al objeto, éste debe saber cómo removerse a sí mismo de la memoria, claro está, si no está siendo usado por alguna otra aplicación cliente.

A pesar de su complejo papel, el modelo COM no es una tecnología complicada. En realidad, los fundamentos de la especificación COM son muy simples y elegantes:

COM es un estándar binario, así que la selección del lenguaje de programación es (con la consideración de un pequeño detalle) un problema del programador. Su codificación también se deja al programador, siempre que un programa siga las reglas de COM. El detalle es que el lenguaje de programación debe ser capaz de utilizar apuntadores indirectos para llamar funciones en un objeto COM. Aunque esto elimina a algunos lenguajes, un programador no está obligado a utilizar C++ en lugar de C, por ejemplo.

Interfaces

Las interfases se identifican de forma única en el tiempo y espacio a través de un identificador que se garantiza único: un número de 128 bits. Una vez que se publica una interfase con este identificador no puede modificarse. Las actualizaciones se manejan mediante la adición de nuevas interfases al objeto, cada una con su propio identificador. Esto garantiza que las aplicaciones cliente no tengan que ser recompiladas. Como los clientes sólo

tratan sobre las interfases de la versión previa de un objeto, sencillamente no se enteran de las nuevas características de éste.

Un objeto es responsable de manejar su propio tiempo de vida. Esto se logra mediante un contador interno de referencias que, en esencia, contiene el número de clientes que están conectados al objeto en todo momento. Cuando el contador de referencias llega a cero, el objeto puede descargarse de la memoria. Es un útil y compacto mecanismo que libera al desarrollador de muchos dolores de cabeza asociados con los recursos compartidos.

Conexión

El mecanismo de conexión entre el cliente y el servidor es el mismo. No importa si el objeto servidor se utiliza dentro de la memoria de proceso del cliente, fuera de las fronteras de proceso del cliente o a través de la frontera de red. De hecho, la aplicación cliente, en la mayor parte de los casos, ni siquiera necesita saber dónde está el componente, porque COM se encargará de establecer la conexión. La aplicación cliente puede desear tomar sus medidas para la utilización de componentes a través de una red, para lidiar con los problemas de latencia, pero esto no es un requisito.

La especificación COM también proporciona una "fundación flexible de seguridad a nivel del objeto". Esencialmente, los objetos y datos COM están sujetos a los permisos de los sistemas operativos y a nivel de aplicación. Al mismo tiempo, puesto que los objetos COM están en línea con los

estándares DCE, las aplicaciones servidor COM tienen la opción de implementar sus propios mecanismos de seguridad.

Así pues, COM es una evolución de POO, simple y llana. Uno de los mejores representantes de este concepto es Visual Basic, dado que toda su interfaz y características basan su operación en este modelo de objetos. Ello explica por qué Visual Basic no apoya la herencia de manera formal, sino a través de la composición. ¿Cuál es la diferencia entre una y otra? A continuación se lo explicó.

Imagine que quiere crear un objeto "SalonDeClases". ¿Qué hace para ello? Toma un objeto "Persona" y reparte varias copias de este objeto en el área. Toma otros objetos "Silla", "Escritorio", "Pupitre", "Suelo", "Pared", "Techo" y de igual manera los coloca en los lugares adecuados. Con esto tiene un objeto compuesto de varios objetos contenidos dentro de sí.

Ahora bien, suponga que quiere construir un objeto "Ave". El objeto "Ave" es inherentemente un animal, sin embargo no lleva un animal dentro de sí, sino que forma parte del reino animal. Esta es la herencia formal. La herencia mediante composición permite crear objetos a través de la contención de otros con todas sus propiedades y métodos inherentes dentro de sí. La herencia formal permite la creación de objetos con base en el aprovechamiento de las propiedades y métodos de otro, los hace parte del primero y permite "pasar por alto" propiedades y métodos clave para el objeto resultante.

La herencia por composición no es sino la única manera, hasta ahora, de crear objetos mediante el aprovechamiento de las capacidades de COM, por sus características binarias. Hace poco, Microsoft liberó una nueva norma llamada DCOM (Modelo de objetos y componentes distribuidos). En DCOM, el programador ya no programa para una computadora individual, sino para el entorno o red en donde operan las computadoras. En el modelo COM, un programador solamente necesita seguir un conjunto de reglas sobre cómo interactúa la computadora y el usuario. Con DCOM, al programador se le proporciona un conjunto de reglas mediante las cuales, las computadoras, en general, interactúan con las demás a través de Internet o intranet.

Desde que Visual Basic vio la luz a principios de esta década, ha sido un lenguaje basado en objetos. Esto es, se crean aplicaciones usando objetos previamente creados como formularios, controles y otros del sistema (como App, Clipboard, Printer, etcétera). Bajo esta ideología, para usar un cuadro de texto en un programa, lo único que hay que hacer es seleccionarlo desde un cuadro de herramientas (que concentra a todos los objetos o controles de este tipo) y colocarlo en un formulario, para después manipular sus propiedades y métodos, de modo que se pueda personalizar su comportamiento y apariencia.

Esto hace que el desarrollo de aplicaciones sea mucho más sencillo que antes, dado que el diseño de interfaces y el desarrollo de tareas se reduce significativamente mediante el uso de objetos prediseñados, que funcionan en una ideología de Conectar y usar (una verdadera implantación de la

tecnología plug and play) que otros lenguajes empezaron a aprovechar posteriormente, como Delphi y Visual C++.

Sin embargo, la industria de las computadoras no considera necesariamente a los lenguajes de programación basados en objetos como lenguajes de Programación orientada a objetos. Esto es porque la POO se refiere a mucho más que tan sólo usar objetos existentes durante el desarrollo. La POO se refiere a una cierta cantidad de principios de diseño, así como a la utilización de una serie de técnicas de desarrollo.

Aunque Visual Basic no cumple cabalmente con todos los requerimientos, sí le proporciona un acercamiento a esta tecnología, al permitirle, a partir de la versión 4 y aún más en la versión 5.0, la creación de sus propios objetos. Si quiere verlo más de cerca consulte en la siguiente dirección de Internet: <http://www.microsoft.com/vstudio>. Ahí encontrará el Service Pack 3 para Visual Basic 5.0, y otras herramientas del Visual Estudio. Deberá bajar 12mb aproximados de información, así que ármese de paciencia. Sin embargo es muy recomendable instalarlo, no importa si su edición de VB es en inglés o en español. Ahora echemos un vistazo a esto.

El papel de COM

Los servidores externos COM pueden intercambiar información aunque se ejecutan en un espacio de dirección diferente al del programa cliente. El proceso COM copia la información y transfiere esta copia al espacio de dirección del servidor para que los métodos del servidor puedan acceder a él.

COM sustituye el indicador hacia la información original con un indicador hacia la copia en el espacio de dirección del servidor. Con IIS, es posible indicar que IIS actúe como cliente y las aplicaciones Web externas que se ejecutan en un paquete MTS separado como servidor.

Cuando finaliza el método del servidor, COM copia los datos del componente en el espacio de dirección del cliente. El proceso es transparente a las aplicaciones y puede incluso transformar tipos de datos distintos de forma automática. Ni el servidor ni el cliente se percatan de las acciones del proceso COM que se está realizando. Microsoft se refiere a este tipo de procesos que copian información de un proceso a otro como marshaling, lo que ocurre cuando una aplicación accede a una propiedad o método de un componente externo. Con IIS, el marshaling ocurre cuando se presenta una interacción con una aplicación Web.

DCOM

Definición

La evolución tuvo su continuación natural en DCOM, COM-distribuido. Un paso en la evolución del COM de Microsoft para su plataforma Windows; con la misma filosofía que aquel, pero pensado para que el sistema pueda funcionar bajo cualquier tipo de red, ya sea LAN, WAN o Internet [4], de forma que se solucionasen los múltiples problemas que añaden estos entornos [1]. DCOM es capaz de transferir la petición de un objeto local a uno remoto ejecutándose en una máquina diferente.

Si bien en sus principios DCOM solo estaba disponible para Windows 95/98 y NT, actualmente existen también implementaciones Unix. Junto con CORBA y Java RMI, DCOM es uno de los tres principales sistemas o modelos de objetos distribuidos. Software AG, Digital y HP lo están transportando a plataformas no Windows.

Al mismo tiempo, el modelo de programación hacia el que se evoluciona, propició la aparición del denominado MTS (Microsoft Transaction Server).

Un software que se ejecuta bajo la forma de servicios del sistema; un conjunto de servicios diseñados para integrar los objetos COM en un modelo mayor de cómputo distribuido, en el que MTS ofrece un background transaccional y un modelo de programación orientado a objetos COM.

Modelo de Objetos

DCOM (Distributed Component Object Model), el Modelo de Objeto Componente Distribuido, es un juego de conceptos e interfaces de programa de Microsoft en el cual los objetos de programa del cliente pueden solicitar servicios de objetos de programa servidores en otros ordenadores dentro de una red. El Component Object Model (COM) proporciona un juego de interfases que permiten a los clientes y servidores comunicarse dentro del mismo ordenador (si están corriendo bajo un sistema Windows 95 o NT).

Por ejemplo, uno puede crear una página para un sitio web que contenga un script o programa que puede ser procesado (antes de ser enviado a otro usuario que lo solicite) no en el servidor del sitio web, sino en otro servidor más especializado de la red. Usando interfaces DCOM, el servidor del programa del sitio web (actuando ahora como un objeto cliente) puede iniciar una Llamada de Procedimiento Remoto (Remote Procedure Call, RPC) al objeto del servidor especializado, que proporciona el procesamiento necesario y devuelve el resultado al servidor del sitio web. Este a su vez pasa el resultado al usuario que está viendo la página web.

El DCOM también puede funcionar dentro de una empresa o en redes distintas de la Internet pública. Usa protocolos TCP/IP y HTTP. El DCOM viene como parte de NT 4.0 y es una actualización gratuita para Windows 95. El DCOM está ya, o pronto lo estará, disponible en todas las principales plataformas UNIX y en los productos de servidores grandes de IBM. El DCOM reemplaza la Automatización Remota del OLE (OLE Remote Automation).

El DCOM es en lo general equivalente a una Arquitectura de Intermediación de Solicitud de Objetos Comunes (Common Object Request Broker Architecture, CORBA) por cuanto proporciona un juego de servicios distribuidos. El DCOM es la aproximación de Microsoft a un ambiente de programas y objetos de datos para toda una red. La CORBA es patrocinada por el resto de industria de tecnología de la información bajo los auspicios del Grupo de Administración de Objetos (Object Management Group OMG).

DCOM es un protocolo de Objetos que permite a los componentes ActiveX comunicarse directamente entre sí a través de una red.

COM+

COM plus, son las extensiones a COM introducidos con Windows 2000. Integran la tecnología que existía previamente de forma separada como MTS en los servicios estándar COM.

El resultado de toda esta evolución que empezó con OLE, es una nueva capa de software situada sobre el antiguo corazón del sistema; un sistema operativo dentro del sistema operativo, como se le ha denominado en ocasiones, que si presenta características de orientación a objetos así como una amplia colección de estos. Actualmente una auténtica plataforma software que basa su filosofía y su potencia en la encapsulación de código dentro de objetos y en ofrecer una especificación y una implementación para la interoperabilidad de dichos objetos. Así, esta tecnología afecta cada vez más a las aplicaciones Windows, permitiendo una comunicación e interacción más sencilla entre ellas, de forma que la mayoría de las operaciones que realiza el PC están controlada en algún momento por COM, desde el uso del portapapeles al navegador.

Windows 2000

Estos servicios, integrados en la plataforma de aplicaciones Microsoft, permiten un enlace de alto rendimiento con la capa de presentación. El producto  Microsoft Internet Information Server (IIS) permite el desarrollo de aplicaciones de negocio basadas en Web que se pueden extender a Internet o distribuir en intranets corporativas. Con Internet Information Server, Microsoft introduce un nuevo concepto en Internet; las aplicaciones transaccionales. Las transacciones son los conductos que hacen posible ejecutar aplicaciones reales de negocio con desarrollos rápidos, escalables y fiables.

Active Server Pages (ASP), un componente de Internet Information Server, es el entorno de scripting de servidor, sin necesidad de compilar e independiente del lenguaje, que se usa para crear y ejecutar aplicaciones Web dinámicas e interactivas. Mediante la combinación de DHTML, scripts y componentes, ASP permite que los desarrolladores de aplicaciones generen contenido Web interactivo y potentes aplicaciones Web.

Con la actual tendencia a distribuir los procesos en los entornos empresariales, es importante tener comunicaciones flexibles y fiables entre las aplicaciones. Los negocios requieren a menudo que aplicaciones independientes en sistemas distintos se comuniquen entre sí e intercambien mensajes incluso aunque las aplicaciones no funcionen al mismo tiempo. Las aplicaciones construidas combinando scripts ASP y en comunicación con los

componentes cooperativos pueden interactuar con los sistemas existentes, aplicaciones y datos.

Utilizar COM+

Las tecnologías COM+ y Microsoft Windows® 2000 proporcionan una sólida plataforma de desarrollo. Esta plataforma se compone de diversas tecnologías esenciales que proporcionan los componentes básicos para generar aplicaciones empresariales de varios niveles.

COM+ ofrece a las empresas las ventajas de las aplicaciones de varios niveles y reduce su complejidad inherente.

En este apéndice vamos a ver un análisis general de los servicios distribuidos que se han integrado en la plataforma COM+. Cualquier aplicación de varios niveles que pueda parecer irrelevante requiere elementos como un soporte para las transacciones, seguridad integrada, un servidor Web, un sistema de mensajería y una entrega de notificaciones de eventos. En este artículo se identificarán dónde se ubican cada uno de estos servicios COM+. De este modo, obtendremos una visión general de estas nuevas tecnologías.

Necesidad de mensajería

Un gran número de protocolos que se utilizan en las aplicaciones distribuidas, como RPC y HTTP, son sincrónicos, es decir, que las aplicaciones que se ejecutan en el equipo cliente deben esperar

pacientemente durante cada petición mientras que el equipo servidor realiza la tarea solicitada. Asimismo, si el servidor tiene numerosas peticiones atrasadas, el cliente no tiene otra opción que enfrentarse a un largo tiempo de espera.

Las aplicaciones creadas mediante protocolos sincrónicos dependen de todos los equipos que están en línea y que forman parte del procesamiento de una petición de forma simultánea. El servidor debe estar en línea para que el cliente realice sus peticiones y, de igual forma, los clientes deben estar en línea enviando peticiones para que se utilicen los ciclos de procesamiento disponibles que ofrece el servidor. Si se desconecta el servidor, los clientes no podrán enviar sus peticiones. Cuando se desconectan los clientes, el servidor permanece expectante sin realizar ninguna actividad. En resumen, todo el mundo debe estar conectado al mismo tiempo para que el sistema en general pueda funcionar.

Suele ocurrir que varias aplicaciones necesiten comunicarse sin conexión y de forma asincrónica. La mensajería es el mecanismo que permite establecer una comunicación asincrónica entre dos o más aplicaciones. Las aplicaciones cliente envían mensajes asincrónicos a las colas de la red. Generalmente cada mensaje representa una petición del cliente o algún tipo de notificación. Las aplicaciones del servidor supervisan estas colas y procesan los mensajes en el momento en que se reciben o con posterioridad. Los mensajes y las colas proporcionan, simplemente, una capa de

indirección entre las aplicaciones que realizan las peticiones y las que las procesan.

Las aplicaciones que se crean sobre un protocolo de mensajería pueden continuar funcionando cuando varios equipos están desconectados. No obstante, no es tarea fácil la creación de una infraestructura sólida de mensajería que pueda enviar cada mensaje hacia su cola de destino con las garantías de envío necesarias. La infraestructura de mensajería debe proporcionar un amplio conjunto de subsistemas a fin de almacenar mensajes de forma transparente cuando los equipos estén fuera de línea y enviarlos con posterioridad a sus destinos cuando se pueda establecer una conexión.

La mayoría de las empresas no están dispuestas a invertir el tiempo y dinero necesarios para escribir y depurar el código de una infraestructura de mensajería personalizada. Afortunadamente, se encuentran a la venta varios productos de mensajería que ofrecen las ventajas de la comunicación asincrónica y sin conexión, como es el caso de MQSeries de IBM y MSMQ Server de Microsoft. Gracias a ello no es necesaria una infraestructura de mensajería personalizada.

Descripción de los servicios COM+

Además de proporcionar un entorno en tiempo de ejecución, COM+ y Windows 2000 también incluyen varios servicios integrados de gran importancia para los programadores que generan aplicaciones de varios

niveles. Puede que algunas aplicaciones de varios niveles necesiten aplicar uno o dos de estos servicios. Otros, sin embargo, necesitarán la utilización de todos ellos. Resulta conveniente tener una perspectiva global de la interrelación entre estos componentes para poder tomar las decisiones adecuadas durante la fase inicial de diseño. En las siguientes secciones se incluye una descripción general sobre los servicios más importantes de la plataforma para las aplicaciones distribuidas.

Componentes en cola

MSMQ se incorporó por primera vez a Windows NT Server en noviembre de 1997 con la versión de Windows NT Option Pack. Desde entonces, los programadores de Visual Basic se han servido de MSMQ para aprovechar las ventajas de la comunicación asincrónica y sin conexión. Sin embargo, la programación en MSMQ requiere código adicional para crear, preparar y enviar mensajes desde las aplicaciones cliente. También es necesario que escriba una aplicación de escucha (listener) del lado del servidor que reciba estos mensajes. Si se compara con la comunicación que se realiza con las llamadas a métodos de COM, en la comunicación con MSMQ resulta más difícil enviar una petición y obtener una respuesta.

COM+ proporciona un servicio denominado "Componentes en cola" que permite obtener mayores ventajas de MSMQ sin tener que programar de forma explícita con la API MSMQ. El servicio de componentes en cola no es más que una capa de productividad generada sobre MSMQ. Los

componentes en cola se modifican exactamente de la misma forma que los componentes estándar de COM+ con una o dos limitaciones. Por ejemplo, no se pueden diseñar métodos con parámetros de salida o que devuelvan valores.

Una vez que se crea y se instala un componente en cola en una aplicación COM+, es preciso configurar un atributo que indique que la interfaz que se está utilizando se supone que está en la cola. También hay que configurar la aplicación COM+ para incluirla en la cola y que reciba mensajes. Al realizar esta operación, el servicio de componentes en cola crea automáticamente una cola especial para la aplicación. Además, configura un servicio proporcionado por el sistema que reciba mensajes para administrar los mensajes entrantes a medida que se reciben.

Una vez que el componente en cola se ha configurado adecuadamente en el servidor, se pueden escribir aplicaciones cliente que lo utilicen. Las aplicaciones cliente no solicitan directamente el objeto de un componente en cola, sino que crean un objeto especial proxy en el lado del cliente denominado "recorder" (grabadora), cuyo aspecto y funcionamiento es el mismo que el del objeto real en lo que se refiere al cliente. A continuación, el cliente comienza a invocar llamadas a métodos de la forma normal. El servicio de componentes en cola proporciona una infraestructura para grabar estas llamadas a métodos en un mensaje MSMQ y transportar estos datos a través de la red hasta el equipo donde se encuentra instalado el componente en cola. La parte del servidor del servicio de componentes en cola recibe el

mensaje, crea una instancia del componente en cola y vuelve a realizar las llamadas a métodos.

El principal objetivo del diseño de los componentes en cola consiste en proporcionar las llamadas a métodos de COM más adecuadas junto con las ventajas que ofrece la comunicación asincrónica y sin conexión. En resumen, los componentes en cola utilizan MSMQ como protocolo subordinado de transporte en lugar de RPC. Los componentes en cola pueden evitar de este modo la gran cantidad de limitaciones de un protocolo sincrónico y orientado a la conexión.

No es necesario utilizar este tipo de servicio para aprovechar las ventajas de MSMQ. Siempre podemos programar directamente con MSMQ. El servicio de componentes en cola constituye un marco de trabajo orientado a la productividad que elimina los incómodos detalles de la programación en MSMQ. Como suele ocurrir con cualquier otro marco de trabajo, se proporciona una mayor productividad a costa de la flexibilidad.

Los componentes en cola no admiten todas aquellas posibilidades que sí se pueden realizar con MSMQ. Se debe utilizar el servicio de componentes en cola únicamente si éste admite las funciones necesarias y supone un ahorro considerable de tiempo de programación en MSMQ. Dentro de las funciones de mayor utilidad se encuentra el servicio de escucha (listener) del lado del servidor. Resulta casi imposible crear un servicio de escucha multiproceso utilizando únicamente Visual Basic, por lo que, si desea una aplicación de escucha del lado del servidor que pueda procesar una gran cantidad de

mensajes por segundo sin recurrir al desarrollo con C++, los componentes en cola es justo lo que necesitamos.

Servicio de eventos COM+

Algunas aplicaciones de varios niveles requieren que determinados usuarios o aplicaciones reciban notificaciones de eventos interesantes como, por ejemplo, el contrato de un nuevo empleado, la modificación en el precio de la mercancía o la falta de existencias en el nivel de inventario donde se necesita un reaprovisionamiento. Los eventos COM+ proporcionan un servicio para el envío y entrega de notificaciones de eventos entre las aplicaciones.

En el modelo de eventos COM+, aquellas aplicaciones que envían notificaciones de eventos se denominan editores, mientras que las aplicaciones que reciben las notificaciones de evento se llaman suscriptores. Normalmente a los eventos COM+ se les conoce como LCE (eventos poco asociados) debido a que los editores y suscriptores no se conocen entre sí. En cambio, un conjunto de eventos se define dentro de una clase evento, que se encuentra instalada en una aplicación COM+. Los editores y suscriptores conocen las clases de eventos, sin embargo, éstos no se conocen entre sí.

La separación de los editores respecto de los suscriptores ofrece algunas ventajas. No es preciso controlar quién recibe los mensajes cuando se crea una aplicación de editor, por lo que se escribe menos código. No es preciso

modificar la aplicación del editor si se desea agregar o eliminar suscriptores una vez incluidos en la producción. Es más, tampoco se tienen que modificar los suscriptores al agregar o eliminar los editores asociados a una clase evento.

A continuación, se analiza brevemente el funcionamiento de este modelo de eventos. Primero se crea una clase evento que implemente una o varias interfaces. Cada interfaz define un grupo de métodos que representa un conjunto de eventos. Para implementar estas interfaces y responder a los eventos cuando se activan se escriben componentes de suscriptores. Finalmente, se escribe una aplicación del editor para crear objetos a partir de la clase evento y llamar a varios métodos cuando ésta desea activar eventos. El servicio de eventos se ocupa del procesamiento del método y de la entrega de eventos a cada suscriptor.

Existen dos tipos de suscripciones: suscripciones persistentes y suscripciones temporales. Cuando se entrega un evento a una suscripción persistente, el servicio de eventos COM+ crea un nuevo objeto a partir de un componente suscriptor configurado y solicita el método asociado al evento. El servicio de eventos activa la aplicación suscriptor si aún no se está ejecutando. Para una suscripción persistente, se crea y se destruye un objeto cada vez que se activa un evento.

El segundo tipo especificado es una suscripción temporal. Una vez que se ejecuta una aplicación, ésta puede registrarse a sí misma con una clase evento específica para recibir los eventos cuando se activan. En este caso, el

servicio de eventos no crea ni destruye un objeto para cada evento. La principal diferencia que existe entre los dos tipos de suscripciones radica en que el suscriptor temporal debe crear una nueva suscripción mediante la programación y registrar una devolución de llamada para recibir los eventos.

La creación de un suscriptor temporal requiere más código que la de un suscriptor persistente. La creación de este último resulta más sencilla porque se pueden utilizar las herramientas de administración de COM+. Si se reinicia el servidor que contiene la clase evento, los suscriptores persistentes continuarán en funcionamiento sin ningún problema. No obstante, todos los suscriptores temporales deberán crear y registrar una nueva suscripción.

Una arquitectura gradual de notificación de eventos requiere una comunicación asíncrona. Si el editor espera que cada suscriptor procese su notificación de eventos, el sistema podría resultar inadecuado. Asimismo, algunas aplicaciones puede que necesiten que se envíen notificaciones de eventos sin una conexión consistente con MSMQ. Los eventos COM+ se han integrado totalmente con los componentes en cola para proporcionar las ventajas de una comunicación asíncrona y sin conexión. Tanto los editores como los suscriptores se pueden configurar para utilizar componentes en cola en lugar de RPC. Para muchas aplicaciones, los eventos COM+ sólo resultan útiles cuando se utilizan con el servicio de componentes en cola.

ActiveX

Contra lo que pudiera parecer, ActiveX no hace referencia a una tecnología bien definida, de hecho "ActiveX es un nombre de marca (de Microsoft), como Calvin Klein o Ford. Y tal como ocurre con cualquier nombre de marca, aquello a lo que se aplica puede variar con el tiempo".

Todas las tecnologías ActiveX se basan en el Component Object Model (COM) de Microsoft, pero no puede decirse que ActiveX sea algo que engloba todas las tecnologías COM; todo lo más que podría afirmarse es que ActiveX es un conjunto mal definido de tecnologías con base COM.

Integración de Visual Basic con COM, DCOM y ActiveX.

El que Visual Basic se ciña a las características antes descritas, le permite, de una manera muy fluida, manejarse bajo el modelo de objetos y componentes (COM) y el modelo de objetos y componentes distribuidos (DCOM), y utilizar eficientemente la propuesta ActiveX de Microsoft (es muy

posible que sea el lenguaje que aprovecha con mayor eficiencia estas características).

Como ya se mencionó, este modelo permite el uso de objetos sin importar en qué lenguaje fueron creados, dado que establece una integración a nivel binario y no a nivel código, mismo que puede ser una limitante de la POO.

El COM le permitirá aprovechar en su código controles u objetos creados por terceros sin tener que preocuparse de su lenguaje de origen. Si el objeto se ciñe a las normas de COM y DCOM, podrá utilizarlo sin problemas como utilizaría cualquier objeto o control propio de Visual Basic.

En Visual Basic 5 (aunque desde VB4 ya se podía hacer con una extensión especial) se pueden crear este tipo de controles de modo que los pueda implantar en cualquier entorno que soporte COM, como Excel, Access, Delphi, Visual C++, etcétera. De esta forma, ya no es requerido utilizar un lenguaje más especializado para poder crearlos.

Esta versión de VB le permitirá crear sus propios controles ActiveX de una manera sencilla, rápida y eficiente. Podrá, a su vez, hacer pruebas del control en el entorno de desarrollo y contar con todas las herramientas de depuración disponibles con sus hermanos mayores. Sin embargo, no podrá crear programas ejecutables.

Para obtener absolutamente toda la funcionalidad del lenguaje, es preferible obtener las ediciones Profesional o Empresarial de VB, aunque podría obtener suficiente funcionalidad en algunos casos, conjuntando las

capacidades de las ediciones VB5CCE y de Aprendizaje de este lenguaje. Cabe hacer notar que la edición de Aprendizaje no incluye herramientas de desarrollo avanzado como ODBC, DAO, RDO y una diversidad de controles que sí ofrecen las otras ediciones.

En este apéndice, que habla de la programación moderna, se ha establecido que Visual Basic no es un lenguaje que cumpla con el paradigma de la Programación orientada a objetos (POO), pero ello no le resta poder para ser considerado un lenguaje de programación moderno, dado que aquella "deficiencia" se contrarresta por su completo soporte a COM.

Apéndice **D**

I.I.S Y A.S.P

APÉNDICE D

Microsoft Internet Information Server 4.0 (IIS)

Introducción

Las empresas buscan tecnologías de Web basadas en estándares para, de una manera más eficiente, intercambiar información con clientes, socios y empleados de todo el mundo. Además, intentan implementar soluciones que aprovechen al máximo las inversiones existentes en conocimientos, formación y tecnología. Microsoft Internet Information Server (IIS) está diseñado para cubrir estas necesidades a un amplio rango de usuarios, desde grupos de trabajo y departamentos de una intranet corporativa hasta proveedores de servicios Internet que alojan sitios Web que reciben millones de visitas diarias. IIS 4.0 revoluciona las capacidades de Web del sistema operativo Microsoft Windows NT Server versión 4.0, proporcionando la forma más fácil de compartir información, crear y distribuir aplicaciones de negocio, y alojar y administrar sitios.

Ventajas

Las características innovadoras de publicación de Web, herramientas personalizables y tecnologías de nuevos asistentes, hacen de Windows NT Server con IIS 4.0 la forma más fácil de publicar y compartir información, de forma segura, a través de intranets corporativas y a través de Internet.

Las aplicaciones Web pueden simplificar procesos como la creación de informes de gastos, administración de beneficios, compras, administración de inventario, marketing y ventas. IIS 4.0 proporciona todo lo necesario para distribuir aplicaciones Web fiables y escalables en Windows NT Server. Las herramientas personalizables, opciones flexibles de administración y herramientas de análisis hacen de Windows NT Server con IIS 4.0 el servidor de Web más fácil de manejar.

Servicios de Internet Information Server

Los servicios de Internet Information Server (IIS) constituyen el servidor Web de Microsoft. En un principio, este producto se creó para Windows NT Server. La versión más reciente es IIS 5.0, que se distribuye con Windows 2000. Al igual que los demás productos del servidor Web, IIS administra las peticiones entrantes de HTTP enviadas por las aplicaciones cliente. Las versiones anteriores de IIS se utilizaron fundamentalmente para páginas Web estáticas. Actualmente, numerosas aplicaciones basadas en Web se sirven de IIS para ejecutar procesamientos personalizados en todas las peticiones entrantes.

IIS exhibe una API registrada denominada Internet Server API (ISAPI) para los desarrolladores que desean crear aplicaciones Web con una lógica personalizada del lado del servidor. Aquellos programadores que programan con ISAPI crean directamente módulos de software denominados extensiones ISAPI y filtros ISAPI. Aunque la escritura de software basado en

ISAPI proporciona los niveles más elevados de rendimiento y flexibilidad, este enfoque también presenta costos significativos, ya que es preciso el desarrollo en C o C++ en lugar de en Visual Basic, con lo cual, los programadores se verán obligados a tratar con detalles de infraestructura de bajo nivel, como la escritura de un administrador de agrupamiento de subprocesos.

Numerosas empresas se niegan a programar directamente con ISAPI debido a que no disponen de la experiencia necesaria o no desean invertir el tiempo o el dinero necesarios para ponerla en práctica. IIS ofrece una alternativa a ISAPI con un marco de trabajo denominado páginas Active Server (ASP). El marco de trabajo de ASP es, en sí mismo, una extensión de ISAPI que permite a los programadores escribir una lógica del lado del servidor utilizando lenguajes de secuencias de comandos y Visual Basic.

Las aplicaciones Web que se crean mediante ISAPI o ASP proporcionan soluciones HTML, ahora bien, está en nuestras manos la decisión de con qué exploradores será compatible la aplicación. Podemos aprovechar al máximo las posibilidades de exploradores, como Internet Explorer en un entorno de intranet utilizando dichas aplicaciones web.

En un gran número de empresas se han generado sitios bastante sofisticados mediante el uso de ASP únicamente. Resulta muy sencillo escribir una lógica empresarial en el servidor y un código de acceso a datos utilizando lenguajes de secuencia de comandos como Visual Basic, Scripting Edition (VBScript) y JavaScript además de una herramienta de desarrollo

como Microsoft Visual InterDev®. No obstante, las empresas también han reparado en la dificultad que conlleva la reutilización, el mantenimiento y la extensión de la lógica difundida en distintas páginas, dificultad que se hace más evidente a medida que aumenta el tamaño del sitio. Un enfoque más adecuado consiste en encapsular la lógica empresarial y el código de acceso a datos dentro de componentes compilados.

La integración entre ASP y COM+ facilita la creación y la ejecución de objetos empresariales personalizados desde una página ASP. De este modo, se puede distribuir la mayor parte de la lógica del servidor utilizando componentes en lugar de secuencias de comandos incrustados en las páginas ASP. El uso de componentes facilita la reutilización, el mantenimiento y la extensión del código. Dichos componentes se pueden crear con Visual Basic, que ofrece mejores mecanismos de comprobación y depuración que las demás herramientas de desarrollo basadas en ASP. Numerosas empresas han reparado en el hecho de que el uso de ASP con Visual Basic proporciona el mejor equilibrio entre la productividad, el mantenimiento y el rendimiento.

Active Server Page (ASP)

Hasta no hace mucho tiempo se ha utilizado las aplicaciones CGI para hacer posible el intercambio de información entre el usuario y el servidor, situación que cada vez se ha hecho más común. En la actualidad cualquier empresa que tenga una web en Internet la utiliza para atender a sus clientes lo más personalizadamente posible. Esto abarca desde consultas inmediatas de los productos o servicios que ofrece la empresa a libros de sugerencias, peticiones, etc., que se graban en una base de datos para poder gestionar la información según sea necesario.

Pero la realidad es que en Internet cualquier pequeño defecto de código se paga con webs de lentísima carga, servidores saturados y habituales errores en cuanto a la gestión de los datos.

Pues bien, ahí está el problema; La ejecución de programas CGI's es muy ineficiente, principalmente por que el proceso de carga del código en memoria del servidor cada vez que un usuario lo invoca. O sea, que si hay 30 usuarios bajando un fichero al servidor mediante un CGI, esté está cargado 30 veces en memoria.

Pero no hace demasiado se buscó la alternativa en otra tecnología (ISAPI (Internet Server API)) y la versión de Netscape para sus servidores web: NSAPI.

Una aplicación ISAPI es una DLL de Windows que se ejecuta en el mismo espacio de direcciones que el servidor de web. Estas aplicaciones pueden

soportar las peticiones simultáneas de diversos clientes con una sola imagen en memoria.

Pero esta tecnología no se libra de sus inconvenientes. La creación de estas aplicaciones es costosa por su complejidad técnica, a la vez que las compilaciones pertinentes se han de hacer en el servidor web. Además las pruebas no se pueden realizar más que en una máquina que esté dando servicios de red.

Hasta ahora no hemos sacado demasiado beneficio profesional con las técnicas de las que disponemos.

ASP (Active Server Pages). Microsoft desarrolla esta tecnología que combina la facilidad de desarrollo de los CGI's y la eficacia de la tecnología ISAPI (el núcleo de funcionamiento de ASP es una aplicación ISAPI).

Para explicar mejor el funcionamiento del ASP se presenta la siguiente gráfica:

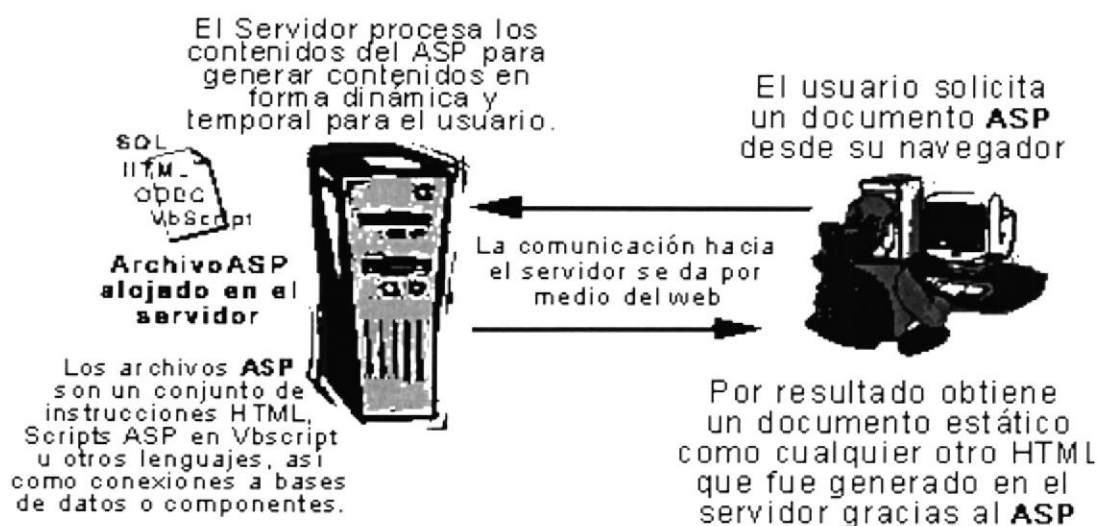


Figura D.1. Funcionamiento del ASP

Aprovechemos esta tecnología

Llevamos un año de atraso con respecto a los países mas avanzados en desarrollo basados en ASP. Y lo peor es que no solo no esta implantado, sino que los proveedores de Internet lo ofrecen en sus conexiones como algo nuevo y que por cierto incrementa bastante el precio anual de sus servicios.

ASP hace un uso importante del micro del servidor pero a estas horas debería ser un estándar.

Más cosas sobre las Active Server Pages.

- Los puntos a favor de las Active Server Pages es la corrección de errores, que es muy cómoda.
- El proceso de desarrollo es extremadamente sencillo, basta con guardar la página en el directorio virtual mientras se va escribiendo.
- Si se hace una petición al servidor para que la procese y nos atienda esta solicitud, se detecta un error, con corregirlo en el editor de texto y grabarlo ya está listo para volver a ser ejecutada.

Otra de las ventajas es que se puede crear páginas web en casa con tan solo un ordenador y una tarjeta de red.

Seguridad y tipos de variables de control

La seguridad de nuestra web es un punto que debemos de tener muy cuenta, hay que ocuparse en buen grado para decir que una web está terminada.

Todos sabemos la inseguridad que corremos cuando navegamos por esta Red y pretendemos aportar nuestros datos a una web para que nos de una conexión gratuita, o nos den de alta en un chat, o, cuando queremos pagar con el número de nuestra tarjeta ese libro que no tenemos tiempo de ir a comprar personalmente.

Si tenemos que realizar una web que acceda a datos, estamos casi obligados, según la importancia de estos, de realizar un estudio de seguridad para evitar que accedan intrusos a nuestros datos más importantes.

Mediante ASP podemos gestionar una seguridad basada en bases de datos, en la que residen los usuarios que tenemos en nuestra web, los permisos de cada uno, etc.

Uno de los puntos clave de este aspecto son las sesiones:

Sesión es una instancia de un objeto (Session) que se crea en el servidor cada vez que un usuario entra por primera vez a un directorio virtual con páginas ASP. Este objeto no se destruye hasta que no lo indiquemos nosotros o hasta que pasa un tiempo sin que el usuario realice ninguna nueva petición. Mediante este tipo de objeto se tiene control sobre aspectos deseados del usuario en todo momento. Su mantenimiento es sencillo, pero debemos saber que el servidor almacena en memoria cada variable de sesión que se cree. Si solo se crean algunas no pasa nada, pero si hablamos de 300 visitantes al mismo tiempo, el servidor lo puede presentar una terrible caída de rendimiento).

Por otro lado tenemos las cookies, son ficheros que se graban en el disco duro del cliente, que contienen cierta información de este que evita hacer continuos accesos al servidor. Lo malo de estos objetos es que la integridad de los datos no puede asegurarse por estar en el cliente y no en el servidor.

El objeto opuesto al objeto Session es Application. Las variables de aplicación no son individuales de cada usuario, sino que son generales, es decir, mediante una variable de aplicación podemos hacer que se comuniquen u obtengan información unos usuarios de otros. Son objetos muy útiles, y se eliminan por el servidor cuando finaliza la aplicación (cuando se desconecta el último usuario visitante).

¿Porqué utilizamos ASP?

Porque ASP es una amigable y muy eficiente forma de hacer aplicaciones de comercio electrónico. Es muy poderosa y también muy fácil de usar. Si se requiere una aplicación "Ecommerce" sobre la web pero se necesita aprender rápidamente un lenguaje de procesamiento de datos, la tecnología ASP lo es. Es también seguro que esta tecnología permanecerá durante mucho tiempo.

Active Server Pages (ASP) es un componente de Microsoft Internet Information Server(IIS). ASP fue liberado por Microsoft en 1996. ASP tiene muchas ventajas. La mayor es que reemplaza la forma tradicional de intercambiar información entre usuarios. Los métodos tradicionales de intercambio cliente-servidor involucraban la tecnología Common Gateway

Interface (CGI). CGI requiere una aplicación separada desde un documento web para procesar todos los datos regresados al servidor. Esto tiene desventajas tanto en la seguridad como en el procesamiento de recursos.

Active Server Pages permiten que el código sea incrustado en un documento HTML y que corra en el servidor, es un concepto simple. El código que necesita correr en el servidor se coloca entre especiales script tags . Estas tags le dicen al servidor que no envíe el código que está entre ellas al cliente. El código no será necesario hasta que se reciba una solicitud del cliente, luego éste procesará el código en el servidor basado en la información enviada de regreso desde el cliente, el código nunca deja el servidor haciendo por lo tanto a ASP muy seguro y fácil de programar.

ASP se ejecuta sobre el servidor y envía datos de regreso al cliente (a través del browser). ASP es compatible con múltiples plataformas.

Existen varios lenguajes que se pueden usar para hacer ASP. El más comúnmente utilizado es VBScript y es por mucho el más fácil, ya que es nativo de Microsoft. ASP se puede hacer también en Perl and Jscript (no JavaScript).

ASP ha logrado llegar a ser el líder en la construcción de aplicaciones ECommerce. Es más rápido, más eficiente, poderoso y extremadamente fácil de utilizar. Los métodos tradicionales de comercio electrónico conocidos como CGI aún existen y son usados ampliamente, sin embargo utilizan más

recursos y son difíciles de implementar. También involucran el uso de sistemas operativos que son un desafío el aprenderlos.

Funcionamiento de ASP

Cuando se trabaja con IIS y ASP el servidor Web analiza las peticiones de página que le llegan. Cuando la petición tiene extensión .ASP se apoya en la aplicación ISAPI que sirve de soporte de ejecución de paginas ASP. ISAPI diferencia entre líneas HTML y los scripts. Cuando determina el lenguaje en el que encuentran los scripts (o programas) da paso a los motores de ejecución de scripts adecuados, Visual Basic Script, Java Script... Estos últimos se encargan de hacer al análisis sintáctico y la compilación de las instrucciones ejecutables. Una vez realizado esto se procede a ejecutar las instrucciones. El usuario recibe como respuesta un fichero ".htm" que se ha formado uniendo las instrucciones HTML originales de la página ".asp" con las instrucciones HTML que se han generado tras los scripts.

Tecnologías estrechamente integradas

El Option Pack para Windows NT Server 4.0 incluye tecnologías que le permiten crear y administrar Internet y aplicaciones distribuidas: Internet Information Server (IIS) 4.0 de Microsoft es un servidor de sedes Web; Microsoft Transaction Server 2.0 (MTS) aloja y ejecuta objetos COM y Microsoft Message Queue Server (MSMQ) añade la estructura NT para construir aplicaciones distribuidas utilizando tecnologías de colas de datos.

Aunque las tecnologías del Option Pack operan de forma independiente, éstas interactúan entre sí más de lo que imagina.

La conexión ASP-IIS-MTS

MTS es un sistema transaccional pensado para desarrollar, desplegar y administrar aplicaciones escalables y de alto rendimiento para Internet e intranets. Define un modelo de programación de aplicaciones para desarrollar aplicaciones distribuidas. MTS también proporciona un entorno para desplegar y administrar estas aplicaciones. Este sistema es la base de los procesos transaccionales Active Server Pages (ASP) de Microsoft.

Microsoft ha integrado estrechamente el IIS y el MTS, de forma que IIS se vale de MTS para ejecutar todas las aplicaciones. MTS usa un paquete para definir el espacio de un proceso. Un paquete contiene los componentes que residen dentro de un proceso particular y define los atributos compartidos por sus componentes. Los componentes de un paquete se ejecutan en el mismo espacio de procesos (los componentes de otro paquete se ejecutan en distinto espacio). Los paquetes permiten a los administradores y desarrolladores repartir los componentes y aplicaciones de un servidor para que se ejecuten de forma conjunta o por separado.

La aplicación In-Process Application de IIS es un paquete predeterminado de MTS para IIS. Todas las aplicaciones que se gestionan en un proceso IIS se ejecutan en este paquete. El paquete incluye cualquier programa Internet Server API (ISAPI), por ejemplo ASP.DLL, utilizado por una aplicación.

Cada servidor definido como ordenador dentro del Explorer de MTS tendrá normalmente muchos paquetes instalados. Para examinar los paquetes de un servidor, seleccione la carpeta Paquetes Instalados.

Cada componente es una aplicación ASP. Por ejemplo, IISWAM.1_Root_Charting apunta hacia a la sede web creada por nosotros con Visual InterDev, por ejemplo. El administrador de aplicaciones Web (WAM) se refiere al archivo C:\winnt\system32\inetsrv\wam.dll. Si usted selecciona las Propiedades de un registro de In-Process Application de IIS, se mostrará una referencia a este archivo, como se observa en la figura.

El WAM administra las aplicaciones IIS. IIS genera las instancias de WAM dinámicamente a medida que recibe solicitudes. Cuando IIS recibe la solicitud de una dirección URL, IIS debe determinar a qué espacio de nombre debe dirigir esta solicitud. IIS busca el nombre de espacio y envía esta solicitud al WAM para su proceso.

Aplicaciones Aisladas

IIS 4.0 le permite ejecutar una aplicación como un proceso aislado. Con esta opción, la aplicación aislada (es decir, una aplicación fuera de proceso) se ejecuta en un espacio de memoria separado, por lo tanto no afecta a las demás aplicaciones o DLL. Este método vuelve a las aplicaciones y al servidor más tolerante a fallos, ya que el fallo de una aplicación no afectará las demás aplicaciones o procesos IIS.

Cuando usted transforma una aplicación no aislada para que se ejecute de forma individual, IIS establece un nuevo paquete MTS para esa aplicación y transfiere su WAM a la carpeta de componentes del nuevo paquete. Por ejemplo, puede crear una aplicación aislada llamada Charting bajo la sede Web predeterminada y la aplicación aparecerá como un paquete IIS-(Default Web Site//Root/Charting). Puede observar el cambio de configuración de la aplicación desde el Explorer de MTS.

IIS puede ejecutar una combinación de aplicaciones aisladas y no aisladas; cada aplicación se ejecuta normalmente y pueden vincularse entre ellas. No existen diferencias en el código de aplicaciones aisladas o no liberadas.

Se dará cuenta de la ventaja que significa poder disponer de las aplicaciones aisladas el día en que alguna de ellas le falle. La aplicación aislada parará, sin afectar a los demás procesos o paquetes, porque éstos se ejecutan en un espacio de procesos o espacio IIS distinto. Asimismo, como es posible ejecutar aplicaciones Web en un espacio de procesos separado, también podrá configurar opciones (como ajustes de limpieza) que se aplicarán únicamente a esa aplicación y no a las demás o al programa IIS completo.

El archivo mt.exe ejecuta una aplicación de forma aislada. Para ver este proceso, abra el Administrador de tareas y localice mt.exe en la lista de procesos, mientras la aplicación se ejecuta. Se mostrará únicamente un suceso de mt.exe con las aplicaciones no aisladas y otro suceso por cada aplicación aislada.

El trabajo conjunto de IIS y ASP

Es necesario conocer cómo se integran las aplicaciones ASP con IIS 4.0. Una raíz virtual puede contener más de una aplicación. IIS 3.0 limita cada raíz virtual a una aplicación definida por el archivo global.asa. IIS 4.0 le ofrece más flexibilidad, pudiendo tener más de una aplicación, cada una de las cuales tendrá una carpeta con sus definiciones global.asa.

Un beneficio adicional de ejecutar una aplicación por separado es que puede descargar y reinstalar componentes que se ejecutan como parte de la aplicación Web sin tener que parar el servicio IIS. Sin embargo, sustituir componentes dentro de una aplicación ASP es problemático, ya que si la aplicación está ejecutando un proceso con IIS, deberá parar el servidor Web para sustituir el componente. Esta interrupción puede significar un gran inconveniente si necesita recopilar y sustituir un componente utilizado por la aplicación ASP. Puesto que la aplicación ASP está utilizando el componente, podrá borrarlo o sustituirlo mientras IIS esté ejecutando el componente.

Para parar IIS, deberá hacer algo más que parar el servidor Web con ISM (Internet Services Manager). Diríjase a la aplicación Servicios del Panel de control e interrumpa el servicio IIS Admin (que a su vez interrumpe el servicio Web Publishing). A continuación, sustituya el componente y reinicie el servicio. La consecuencia es que la aplicación Web se interrumpe cuando el servicio está apagado. (Este método puede causar problemas si utiliza componentes en su servidor de producción.)

Si desea parar una aplicación aislada, abra ISM, seleccione con el botón derecho del ratón el nombre de la aplicación y seleccione Propiedades. Seleccione Descargar para parar la aplicación. A continuación, sustituya el componente. La aplicación se iniciará cuando los usuarios accedan a la aplicación.

Ejecutar aplicaciones Web dentro del espacio de procesos compartidos de IIS también produce beneficios. Experimentará una mejoría en el rendimiento del sistema, porque este método evita mucho del consumo de recursos de sistema asociados a aplicaciones externas.

Los beneficios de las aplicaciones aisladas

Los componentes internos (in-process) se ejecutan más rápido que los componentes externos, porque todo el código y la información están localizados en el espacio de dirección del cliente. En el caso de los componentes internos, no es necesaria ninguna acción para copiar información desde un espacio de dirección a otro. Puesto que las aplicaciones Web internas se ejecutan en un espacio de proceso compartido con el IIS, la aplicación e IIS pueden compartir mejor los recursos.

Las aplicaciones aisladas ayudan muchísimo durante el proceso de desarrollo. Se pueden ejecutar aplicaciones de un proceso aislado mientras se construye la aplicación y sus componentes. Este método proporciona protección de una aplicación potencialmente defectuosa. Asimismo, puede

cargar y descargar también componentes sin tener que interrumpir el servidor.

Cuando la aplicación está lista para la producción, puede transformarla en un proceso interno de IIS para maximizar el rendimiento. También es posible convertir un proceso de producción en un proceso aislado cuando necesite sustituir componentes. Después de completar la sustitución, podrá volver a configurar la aplicación como aplicación interna.

Integración de las piezas

En este apéndice, hemos explicado por qué los desarrolladores y administradores de sistemas deben entender cómo se integran las tecnologías y lo que sucede detrás del telón. Con la gran variedad de asistentes, interfases y sencillas herramientas disponibles, es fácil olvidar la necesidad de comprender los detalles. La estrecha integración de MTS e IIS demuestra cómo conocer los detalles es importante.