BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

Facultad de Ciencias de la Computación

Sistema para seguimiento de órdenes de servicio basado en WEB

Tesis que para obtener el título de:

Licenciado en Ciencias de la Computación

Presenta:

Moisés Guzmán Arias

Asesor:

Dr. David Eduardo Pinto Avendaño

Noviembre 2009

Índice

Introducción

En la actualidad existen empresas dedicadas a proporcionar servicios de mesa de ayuda a los usuarios de las organizaciones. Cuando se menciona el término “servicio”, generalmente es entendido como la reparación de objetos, soporte vía telefónica, consultoría, etc.

En la realidad, la mayoría de estas empresas realiza el seguimiento de las órdenes de servicio de forma manual. Se expide un documento de la solicitud de servicio a realizar y se verifica (en algunos casos) su seguimiento que incluye estado del avance, conclusión, cancelación o reapertura.

La solución de automatización se ha hecho mediante la creación de un sistema de cómputo basado en WEB bajo el esquema Intranet, lo que implica una aplicación cliente-servidor con restricción de acceso a la información, mediante módulos independientes que son invocados y liberados para trabajar con ellos mediante la autenticación del usuario que inicia la aplicación.

Se han definido diversos niveles de acceso al sistema donde cada uno representa un rol de trabajo e interacción con él y la información que se administra.

- Administrador del sistema.
- Encargado de área.
- Técnico.
- Recepción.
- Cliente.

La información que se manipula en el sistema es la siguiente.

- Clientes.
- Órdenes de Servicio.
- Empleados.
- Áreas.
- Tipos de servicio
- Asignación de órdenes de servicio.
- Mensajes (Avisos por medios electrónicos)

Este documento de tesis se ha dividido de la siguiente forma:

En el Capítulo I, se presentan conceptos que el lector puede usar para comprender a detalle el resto del trabajo realizado. Esto conceptos incluyen tópicos relacionados con Internet, bases de datos y UML. Dado que el trabajo fue desarrollado en base a una metodología, el Capítulo II se desarrolla la documentación del análisis y diseño del proyecto. EL Capítulo III muestra la implementación del sistema y por último se presentan las conclusiones y perspectivas de este trabajo de tesis.

Capítulo I. Marco Teórico

Puesto que se desarrollará un sistema de seguimiento de órdenes de servicio basado en WEB, a continuación se presentan algunos conceptos necesarios para que el lector comprenda mejor el contenido de éste documento; dando inicio con redes de computadoras.

1.1 Redes de computadoras [1]

Al hablar de redes de computadoras, hablamos de computadoras autónomas, capaces de realizar tareas de manera independiente, pero que se encuentran interconectadas entre sí[2], con la finalidad de compartir recursos, información o procesar tareas de manera compartida. El uso de redes de cómputo ha cambiado completamente la manera de trabajar de las empresas y en general de las personas, antiguamente las grandes empresas disponían de una o dos computadoras de gran tamaño, en las cuales se montaban las aplicaciones de los usuarios.

Las redes de computadoras se pueden clasificar de muy diversas maneras, pero una de las más acertadas es de acuerdo a su cobertura geográfica:

Tipo de red	Cobertura
Redes de área local	Cuarto, edificio y campus
Red de área metropolitana	Ciudad
Red de área amplia	País o continente
Planeta	Internet

Cada una de estas redes tiene diferentes tipos de interconexión y especificaciones, peculiares, pero se ha tenido el cuidado de poder interconectar un tipo de red con otra.

Conforme los dispositivos activos y pasivos de redes de computadoras han evolucionado, la velocidad de transferencia de información y cobertura han mejorado de manera directa.

Con el abaratamiento de la infraestructura de telecomunicaciones y equipos de cómputo, la aplicación de redes de computadoras hacen posible hoy en día que teleconferencias, transferencias de información de un lado al otro del planeta y un sinnúmero de cosas que hace algunos años se consideraron como parte de la ciencia ficción, son una realidad; esto ha traído consigo un conjunto de beneficios económicos y sociales muy importantes que se ha impactado a empresas y personas en la manera de realizar sus procesos y tareas cotidianas respectivamente.

Internet, es una de las redes con mayor difusión y aplicación hoy en día en todo el mundo, es una red de redes que llega a una gran cantidad de hogares y la cual sustenta su funcionamiento en el protocolo de comunicación TPC/IP.

1.2 TCP/IP

Para poder hablar de TCP/IP, primero debemos decir que un protocolo es un conjunto de reglas y convenciones que se siguen para poder llevar a cabo una conversación o transferencia de información.

TCP e IP fueron desarrollados inicialmente en 1973 por Vinton Cerf en un proyecto dirigido por Roberth Kahn y patrocinado por la Agencia de Programas Avanzados de Investigación (ARPA) del Departamento de Defensa Estadounidense.

[1] Información obtenida en la referencia [12]
[2] La interconexión de computadoras se puede dar de muchas maneras, cobre, microondas, fibra óptica y satélites de comunicaciones.

Este protocolo es utilizado en todas las computadoras conectadas a Internet, es un protocolo muy flexible partiendo del hecho de que en Internet existen equipos de cómputo con hardware y software talmente incompatibles entre sí, además de que se manejan muchos medios de conexión entre un punto y otro, TCP/IP se encarga de conectar todo esto de manera transparente.

TCP/IP no es un protocolo único, si no que es un conjunto de protocolos que colaboran entre sí y que cubren diferentes niveles del Modelo OSI[3] , estos protocolos son:

- **FTP**: Sistema de transferencia de archivos.
- **SMTP:** Es una aplicación para correo electrónico.
- **Telnet**: Permite la conexión a una terminal remota desde un proceso o terminal.
- **RPC**: Permite el llamado al procedimientos situados en equipos remotos, tratándose como su fueran procesos locales.
- **NFS:** Permite la utilización de archivos distribuidos por los programas de la red.
- **X-Windows**: Es un programa para el manejo e interfaces de usuarios.
- **http**: Protocolo de transferencia de hipertexto

Con la aparición de http, se genera el gran auge de Internet ya que es éste el protocolo sobre el cual se basa la WEB (www) y es el encargado de resolver y atender las peticiones de para desplegar una página de este tipo.

1.3 Internet

Internet es un tipo de red bastante especial que tiene sus orígenes en la red creada por la Nacional Science Fundation de Estados Unidos en 1986 para conectar los centros de de supercomputación. De esta iniciativa posteriormente surge Merit que es resultado de la unión de MCI e IBM que desarrollaron una dorsal de 1.5 Mb.

De manera independiente a Merit, se creó una corporación no lucrativa denominada NFS, encargada de crear la red TCP/IP más veloz del mundo con un ancho de banda de 45 Mbps.

Internet nace como tal en los años 1990, creciendo de aproximadamente 50,000 servidores hasta 10 millones en el año de 1996, actualmente el número de equipos conectados a Internet es mucho mayor y crece día a día.

Pero todo lo anterior no basta para poder explicar el por que es que Internet ha tenido tanta penetración, son necesarios otros elementos tales como lenguajes de programación, lenguajes para generar páginas Web, lenguajes de validación, bases de datos e ingeniería de software para generar aplicaciones basadas en Web.

Existen diversas combinaciones para generar una plataforma de desarrollo en la que se involucran al sistema operativo, los medios de comunicación y las herramientas de desarrollo; como ya se mencionó TCP/IP es un protocolo que permite la interconexión y transferencia de información de manera independiente al software y al hardware pues así fue concebido desde su creación, por lo tanto podemos inmiscuir plataformas Unix, Windows, Mac, etc., con lenguajes

[3] Es un modelo de desarrollo propuesto por ISO como un paso hacia la estandarización internacional de los protocolos que se usan en diversas capas para la interconexión de sistemas abiertos.

como php, perl c, java, etc., sin dejar fuera al elemento principal que es html (Hipertext Markup Language) que es el lenguaje en el cual se escriben las páginas WEB y los navegadores que permiten visualizarlas. Para el caso de estudio, se ha propuesto utilizar la siguiente plataforma de desarrollo y producción:

- HTML
- Java Script
- ASP
- MS SQL
- Internet Information Server

A continuación se muestra un panorama breve de cada una de estas herramientas y como se integran para el desarrollo de aplicaciones avanzadas basadas en WEB.

1.4 HTML[4]

HTML (HyperText Markup Language) es un lenguaje muy sencillo que permite describir hipertexto, es decir, texto presentado de forma estructurada y agradable, con enlaces que conducen a otros documentos o fuentes de información relacionadas, y con inserciones multimedia (gráficos, sonido...) La descripción se basa en especificar en el texto la estructura lógica del contenido (títulos, párrafos de texto normal, enumeraciones, definiciones, citas, etc) así como los diferentes efectos que se quieren dar (especificar los lugares del documento donde se debe poner cursiva, negrita, o un gráfico determinado) y dejar que luego la presentación final de dicho hipertexto se realice por un programa especializado (como Opera, Safari o Internet Explorer).

Un documento HTML comienza con la etiqueta <html>, y termina con </html>. Dentro del documento (entre las etiquetas de principio y fin de html), hay dos zonas bien diferenciadas: el encabezamiento, delimitado por <head> y </head>, que sirve para definir diversos valores válidos en todo el documento; y el cuerpo, delimitado por <body> y </body>, donde reside la información del documento.

La única utilidad del encabezado en la que nos detendremos es la la etiqueta <title>, que permite especificar el título de un documento HTML. Este título no forma parte del documento en sí: no aparece, por ejemplo, al principio del documento una vez que este se presenta con un programa adecuado, sino que suele servir como título de la ventana del programa que nos la muestra.

El cuerpo de un documento HTML contiene el texto que, con la presentación y los efectos que se decidan, se presentará al usuario. Dentro del cuerpo son aplicables todos los efectos que se definen en HTML. Dichos efectos se especifican exclusivamente a través de directivas. Esto quiere decir que los espacios, tabulaciones y retornos de carro que se introduzcan en el fichero fuente no tienen ningún efecto a la hora de la presentación final del documento.

En general HTML es un lenguaje muy sencillo que permite mostrar información estática que se almacena en un servidor y que se puede visualizar con un navegador de Internet, ésta plataforma de desarrollo con limitaciones de alimentación y actualización detuvieron por algún tiempo la creatividad de los desarrolladores para WEB. Al ver esto, se propuso la incorporación de diversas tecnologías, lenguajes de script que permitieron una mejor interacción con los usuarios. Los pioneros en esta tecnología fueron C, C++ y Perl que ejecutándose en el lado del servidor, se tuvo ya la posibilidad de recuperar información de forma dinámica, trabajar en conjunto con bases

[4] Información recopilada de la referencia [1L], [3L] y [5L]

de datos y otros recursos de los servidores. Para tal efecto en 1996 Microsoft introduce en su versión 3 de Internet Information Server una DLL[5] del tipo ISAPI[6] que da al servidor la capacidad de ejecutar archivos con extensión ASP.

En el siguiente punto se analizaran más a fondo las características, utilidad y funcionamiento de ASP (Active Server Page).

1.5 ASP[7]

No se debe definir a ASP como un lenguaje de programación, si no como un marco de desarrollo que permite construir aplicaciones basadas en Internet que se apoya en el lenguaje HTML y lenguajes de script, en motores de bases de datos y el lenguaje de consulta SQL.

Las principales ventajas de ASP son:

- Es gratuito
- Mezcla de código ASP, HTML y script del lado del cliente sin necesidad de compilación
- No requiere forzosamente de editores caros o complejos, basta un simple block de notas para editar archivos ASP.
- ASP se ejecuta del lado del servidor, generando documentos HTML que se pueden visualizar en cualquier navegador.
- Permite incorporar componentes programados en otros lenguajes como Visual Basic, Delphi, C, C++ y otros que se pueden invocar desde el mismo código ASP.
- Permite acceso rápido y sencillo a bases de datos.
- Permite el uso de componentes OLE, acceso a archivos, login al sistema, envío de correo electrónico, etc.
- Tiene persistencia de variables en memoria en diferentes invocaciones de la página, permitiendo con ello manejar sesiones de usuario, resolviendo el problema de la no orientación a conexión del protocolo http.

Para poder trabajar con ASP solo necesitamos un servidor de WEB que lo soporte, como es el caso de Personal WEB Server e Internet Information Server, para ejecutar un archivo ASP, es necesario situarlo en un directorio virtual del servidor de WEB e invocarlo desde un navegador de Internet.

Como se mencionó, ASP puede utilizar lenguajes de Script como JavaScript y VBScript, en esta ocasión se ha elegido utilizar VBScript,

1.6 VBScript[8]

El Visual Basic Script (en adelante VBScript) es un lenguaje de script, directamente derivado de Visual Basic. Los lenguajes de script son versiones recortadas de otros lenguajes. Estas versiones se usan para su integración en páginas WEB. Un código escrito en un lenguaje de script se incorpora directamente dentro de un código HTML y se ejecuta interpretado, no

5 Dynamic Link Library o más comúnmente DLL (en español biblioteca de enlace dinámico) es el término con el que se refiere a los archivos con código ejecutable que se cargan bajo demanda de un programa por parte del sistema operativo. Esta denominación se refiere a los sistemas operativos Windows siendo la extensión con la que se identifican los ficheros, aunque el concepto existe en prácticamente todos los sistemas operativos modernos.

6 Es una interfaz de programación de aplicaciones (API) para el servidor web de Microsoft, IIS (Internet Information Server). La ISAPI permite que los programadores puedan desarrollar aplicaciones basadas en web que se procesen mucho más rápidamente que los programas CGI. Esto es así porque están más integrados con el servidor web.

7 Información recopilada de la referencia [1L], [4] y [7]

8 Información recopilada de la referencia [4] y [7]

compilado. Lo cual significa que ninguna línea de código fuente sufre modificación alguna previa a su ejecución y que esta traducción a lenguaje máquina se lleva a cabo en el momento en que es requerido por el interprete y una ves utilizada la traducción, no se guarda ninguna copia de ella en el equipo que la utilizó o el servidor del que provino.

1.6.1 Ventajas y desventajas de la interpretación frente a la compilación

Interpretación

Ventajas	Desventajas
• El código es cómodo para depurar, ya que no es necesario volver a compilar tras un cambio. • No es necesario disponer de un compilador, ya que el intérprete (que forma parte del navegador) ejecuta el script. • El mantenimiento es fácil y rápido, por parte del autor o de otro programador.	• La ejecución se ralentiza, al ser necesaria la interpretación línea a línea cada vez. • El código es visible y puede ser objeto de plagio por parte de otras personas. • El usuario tiene acceso al código y puede modificarlo, estropeando alguna operación.

Compilación

Ventajas	Desventajas
• El código compilado se ejecuta muy rápido, al no ser necesaria una traducción cada vez que se le invoca. • El código compilado no puede ser "abierto" por otras personas. No es necesario transmitir el código fuente. • El código compilado puede estar, íntegramente, incluido en un solo fichero.	• Es necesario disponer de un compilador-linkador para el proceso de la compilación. • El código compilado suele ocupar bastante espacio en disco, ya que incorpora en el propio código algunas librerías del sistema. • Depurar un programa implica volver a compilar tras los cambios.

El código en VBScript puede, además, estar diseñado para su ejecución en el lado del cliente o en el del servidor. La diferencia es que un código que se ejecuta en el lado del servidor no es visible en el lado del cliente. Este recibe los resultados, pero no el código. El código que se debe de ejecutar en el lado del servidor estará incluido en la página web correspondiente entre los tags <% y %>

1.6.2 ActiveX Scripting[9]

Para poder comunicar a VBScript con otros recursos del lado del servidor, Mircrosoft ha incorporado el uso de ActiveX Scripting, el cual evita que los exploradores y otras aplicaciones host necesiten escribir código especial de integración para cada componente de la secuencia de comandos. ActiveX Scripting permite a un host compilar una secuencia de comandos, obtener y llamar puntos de entrada, y administrar el espacio de nombres disponible para el programador. Con ActiveX Scripting, los distribuidores de lenguajes pueden crear run times de lenguaje estándar para la secuencia de comandos. Microsoft proporcionará soporte de tiempo de ejecución para VBScript.

[9] Información recopilada de la referencia [7]

Microsoft está trabajando con varios grupos de Internet para definir el estándar de ActiveX Scripting de modo que se puedan intercambiar los motores de automatización. ActiveX Scripting se utiliza en Microsoft® Internet Explorer y en Microsoft® Internet Information Server.

VBScript, como el Visual Basic del que deriva, incorpora la POO, aunque en menor grado que otros lenguajes actuales. En realidad la estructura de este lenguaje es, hasta cierto punto, un poco anárquica, vestigio de las antiguas versiones de Basic, que eran totalmente procedimentales.

1.6.3 VBScript en otras aplicaciones y exploradores

Como programador, puede licenciar su implementación de origen de VBScript sin cargo para utilizarla en sus productos. Microsoft proporcionará implementaciones binarias de VBScript para la API de Windows de 32 bits, la API de Windows de 16 bits y la de Macintosh. VBScript se integra con exploradores de World Wide Web. VBScript y ActiveX Scripting también se pueden utilizar como un lenguaje de automatización general en otras aplicaciones.

Hasta el momento hemos hablado de los lenguajes de programación que permiten el desarrollo de una interfaz basada en WEB que además permiten acceder a bases de datos, pero precisamente no hemos hablado de las herramientas necesarias para manipular estructuras de datos y darles una debida organización para recuperación de información bajo un esquema bien definido que permita tareas de inserción, recuperación, actualización y eliminación de dicha información, razón por la cual a continuación se presenta una breve descripción y conceptos acerca de MS SQL Server que es el manejador de bases de datos de Microsoft y el cual se propone utilizar para el manejo de la base de datos de la aplicación que se plantea para el manejo de ordenes de servicio.

1.7 MS SQL Server Express 2008[10]

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

La versión MS SQL Server Express es una versión diseñada para crear aplicaciones robustas y confiables a los desarrolladores, es gratuita y permite implementación de aplicaciones que no requieren una alta carga transaccional que requieren de una base de datos sencilla y confiable. Está disponible para descarga directamente en Internet en el sitio MSDN de Microsoft o se instala directamente de forma nativa en todas las herramientas de visual estudio express.

Entre otras cosas, las principales características son:

- Una base de datos robusta para crear aplicaciones dinámicas
- Fuerte soporte de XML
- Herramientas y características para mejorar la gestión y facilidad de uso
- Instalación y setup sencillos
- Escalabilidad y rendimientos
- Gran integración con Visual Studio 2008
- Robusta seguridad

[10] Información recopilada de la referencia [13]

1.7.1 Arquitectura Cliente/Servidor

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

- El Cliente es responsable de la parte lógica y de presentar la información al usuario. Generalmente, el cliente corre en una o más computadoras Cliente, aunque también puede correr en una computadora Servidor con SQL Server.

- SQL Server administra Bases de Datos y distribuye los recursos disponibles del servidor (tales como memoria, operaciones de disco, etc) entre las múltiples peticiones.

La arquitectura Cliente /Servidor permite desarrollar aplicaciones para realizar en una variedad de ambientes.

1.7.2 Sistema Administrador para bases de datos relacionales (RDBMS)

El RDBMS es responsable de:

- Mantener las relaciones entre la información y la Base de Datos.
- Asegurarse de que la información es almacenada correctamente, es decir, que las reglas que definen las relaciones ente los datos no sean violadas.
- Recuperar toda la información en un punto conocido en caso de que el sistema falle.

1.7.3 TRANSACT – SQL

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

1.7.4 Plataformas para MS SQL Server

Los componentes Cliente y Servidor de SQL Server corren en los Sistemas Operativos mostrados en la siguiente tabla:

PLATAFORMA	COMPONENTE SERVER	COMPONENTE CLIENTE
Microsoft Win 95/98/XP/Vista/7	Si	Si
Microsoft Windows NT Workstation 4.0 y posteriors	Si	Si
Microsoft Windows NT Server 4.0 y posteriors	Si	Si
Microsoft Windows NT Server Enterprise Edition 4.0 y posteriors	Si	Si
Windows 3.X	No	Si
MS-DOS	No	Si

Third party	No	Si (Unix, apple Macintosh)
Internet browsers	No	Si

1.7.5 Base de la seguridad

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

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

SQL Server corre como un servicio dentro de Windows, permitiendo operarlo remotamente.

1.7.6 Servicios de SQL Server

Los servicios de SQL Server incluyen MSSQLServer, SQLServerAgent, Microsoft Distributed Transaction Coordinator (MSDTC), y Microsft Search. Aunque estos servicios de SQL generalmente corren en Windows NT y posteriores, también pueden correr como aplicaciones.

MS SQL Server:

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

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

SQLServerAgent:

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

Microsoft Distributed Transaction Coordinator

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

Servicio de Microsoft Search:

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

1.7.7 Software de SQL Server Express

SQL Server incluye una variedad de software para administrar y mantener al servidor, encontrando ayuda acerca de temas específicos, diseñando y creando Bases de Datos y buscando información.

SQL Server Enterprise Manager SNAP-IN:

SQL Server está provisto de un cliente administrativo, que es el SQL Server Enterprise Manager, el cual es una Consola de Administración de Microsoft (MMC) de tipo Snap-in. MMC es una interfase de usuario compartida para administración de servidor usada por Back Office. Esta consola compartida, provee un ambiente consistente para administración de herramientas.

1.7.8 Herramientas y asistentes para la administración de SQL Server:

Sql Server provee un número de herramientas administrativas y asistentes que atienden aspectos particulares de SQL Server. La siguiente tabla describe las herramientas y asistentes de SQL Server mas importantes de la versión Express 2008:

HERRAMIENTA GRÁFICA	APLICACIÓN
SQL Server Configuration Manager	Utilidad para administrar la configuración del servidor Incluye el monitoreo del servicio Configuración de red
SQL Server Management Studio	Herramienta de administración de: • Bases de datos • Seguridad • Objetos del servidor • Replicación • Adinistración

1.7.9 Arquitectura

Comunicación:

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

- APLICACIÓN: Una aplicación es desarrollada usando una aplicación de interfaz de programación para Base de Datos (API). La aplicación no tiene conocimiento de los protocolos internos de red usados para la comunicación con SQL Server.
- INTERFAZ DE LA BASE DE DATOS: Esta es una interfaz usada por una aplicación para mandar peticiones a SQL Server y procesar los resultados devueltos por SQL Server.
- LIBRERÍA DE RED: Este es un componente de Software de comunicación que empaqueta las peticiones de la Base de Datos y los resultados para transmitirlos por medio del protocolo de red apropiado. Una librería de Red, también conocida como Net-Library, debe ser instalada tanto en el cliente como en el servidor. Tanto Clientes como Servidores pueden usar más de una Net-Library al mismo tiempo, pero deben usar una Librería de Red común para comunicarse satisfactoriamente. SQL Server soporta protocolos de red tales como TCP/IP, Novell, IPX/SPX, Banyan VINES/IP, Named Pipes,y Apple Talk ADSP.

- TABULAR DATA STREAM: (TDS) Es un protocolo por niveles de aplicación usado para la comunicación entre un Cliente y SQL Server. Los paquetes TDS son encapsulados en los paquetes de red hechos por la protocol stak usada por las Net-Libraries.
- SERVICIOS OPEN DATA: Este es un componente de SQL Server que se encarga de las conexiones de red, pasando las peticiones del cliente al SQL Server para procesar y regresar cualquier resultado a los Clientes. Open Data escucha automáticamente en todas las Net-Libraries que están instaladas en el servidor.

Desarrollo de aplicaciones:

Los usuarios acceden al SQL Server a través de una aplicación que está escrita con una interfaz de objetos de datos o con una API. SQL Server soporta interfaces comunes y APIs nativos de bajo nivel.

Interfaces de programación de aplicaciones:

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

- OLE DB: Esta es una interfaz de acceso a datos basada en el COM (Component Object Model). Soporta aplicaciones escritas usando OLE DB o Interfaces de Objetos de Datos basadas en OLE DB. Puede acceder a la información en SQL Server, otras Bases de Datos relacionales y otras fuentes de datos.
- OPEN DATABASE CONNECTIVITY: 8ODBC) Es una interfaz por capas. Accede directamente al protocolo SQL Server TDS y soporta aplicaciones o componentes que estén escritos usando ODBC o interfaces basadas en ODBC. Puede acceder a los datos en SQL Server, y otras Bases de Datos relacionales, pero generalmente no puede ser usado para controlar otras fuentes de datos.

Data objects interfaces:

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

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

Administración:

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

Administración de SQL Server:

SQL Server puede ser administrado usando:

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

Administración distribuida de objetos SQL:

(SQL-DMO) Es una colección de objetos de administración basados en COM, usados por SQL Server. SQL-DMO oculta los detalles de las operaciones Transact-SQL y es apropiado para escribir scripts de administración para SQL Server. Las herramientas de administración incluidas en SQL Server están escritas usando SQL-DMO.

SQL Server Agent:

Es un servicio que trabaja en conjunto con SQL Server para desempeñar las siguientes tareas administrativas:

- Administración de Alertas: Las alertas brindan información acerca del estado de un proceso, tal como cuando un trabajo está completo o cuando ocurre un error. El agente de SQL Server monitorea la aplicación de Windows NT y genera alertas.
- Notificación: El agente de SQL Server puede enviar e-mails, o iniciar otra aplicación cuando ocurre una alerta, por ejemplo, se puede programar una alerta para que ocurra cuando una Base de Datos o cuando una transacción está casi completa o cuando un respaldo de la Base de Datos ha terminado exitosamente.
- Ejecución de Tareas: El agente de SQL Server incluye un motor de creación y planeación de tareas. Las tareas pueden ser simples operaciones de un solo paso, o pueden ser tareas complejas de varios pasos que requieren planeación. También se pueden crear pasos de las tareas con Transact-SQL, leguajes script, o comandos del Sistema Operativo.
- Administración de Réplicas: La replicación es el proceso de copiar datos o transacciones de un SQL Server a otro. El agente de SQL Server es responsable de sincronizar los datos entre los servidores, monitorear los datos para buscar cambios y replicar la información en otros servidores.

1.7.10 Seguridad en SQL Server

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

Roles:

Permiten reunir a los usuarios en una sola unidad a la cual se le pueden aplicar permisos. SQL Server contiene roles de servidor y de Base de Datos predefinidos, para tareas administrativas comunes, de manera que pueden asignársele determinados permisos administrativos a un usuario en particular. También se pueden crear roles de Base de Datos definidos por el usuario. En SQL Server, los usuarios pueden pertenecer a varios roles:

- Roles fijos del Servidor: Proveen agrupamientos con privilegios administrativos a nivel del Servidor. Son administrados independientemente de las Bases de Datos de usuarios a nivel servidor.
- Roles fijos de la Base de Datos: Proveen agrupamientos con privilegios administrativos a nivel de Base de Datos.
- Roles de usuarios definidos en la Base de Datos: También se pueden crear roles para Base de Datos, para representar un trabajo desarrollado por un grupo de empleados

dentro de una organización. No es necesario asignar y quitar permisos a cada persona. En función de que cambia un rol, se pueden cambiar fácilmente los permisos del rol y hacer que los cambios se apliquen automáticamente a todos los miembros del rol.

Validación de permisos

Dentro de cada Base de Datos, se asignan permisos a las cuentas de usuarios y a los roles para permitir o limitar ciertas acciones. SQL Server acepta comandos después de que un usuario ha accedido a la Base de datos.

SQL Server realiza los siguientes pasos cuando valida permisos:

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

1.7.11 Bases de datos en SQL Server

Cada SQL Server tiene dos tipos de Bases de datos: Bases de Datos del Sistema y Bases de Datos del usuario. Las Bases de Datos del sistema almacenan información acerca de SQL Server como un total. SQL Server usa la Base de Datos del sistema para operar y administrar al sistema. Las Bases de Datos de usuarios, son Bases de Datos creadas por los usuarios. Una copia del SQL Server puede administra una o más Bases de datos de usuario.

Bases de datos de sistema y de usuario:

Cuando SQL Server es instalado, el setup crea 4 bases de datos de sistema 2 y 2 de usuario, de ejemplo. La Base de Datos de distribución es instalada cuando se configura SQL Server para actividades de replicación.

Objetos de la base de datos

Una Base de Datos, es una colección de datos, tablas y otros objetos. Los objetos de la Base de Datos ayudan a estructurar los datos y definir mecanismos para la integridad de datos.

En resumen, un manejador de bases de datos como SQL Server es una herramienta muy poderosa, capaz de almacenar grandes volúmenes de información de manera estructurada bajo un conjunto de reglas, pero de que sirve esto si no se cuenta con un mecanismo de manipulación de información o de definición de datos. En el inciso 1.6.3, se habla de Transact-SQL como una versión de SQL, a continuación hablaremos de SQL y sus características principales.

1.8 SQL [11]

Hasta la década de los 80, las personas que preparaban las consultas e informes de una base de datos debían ser programadores. Al aparecer las bases de datos con lenguajes de

[11] Información recopilada de la referencia [8]

consulta sencillos y estandarizados, semejantes al lenguaje natural, el proceso de consulta puede hacerlo cualquier usuario mediante un lenguaje escrito asequible.

El lenguaje de gestión de bases de datos más conocido en la actualidad es el SQL, Structured Query Language, que es un lenguaje estándar internacional, comúnmente aceptado por los fabricantes de generadores de bases de datos.

El SQL trabaja con estructura cliente/servidor sobre una red de ordenadores. El ordenador cliente es el que inicia la consulta; el ordenador servidor es que atiende esa consulta. El cliente utiliza toda su capacidad de proceso para trabajar; se limita a solicitar datos al ordenador servidor, sin depender para nada más del exterior. Estas peticiones y las respuestas son transferencias de textos que cada ordenador cliente se encarga de sacar por pantalla, presentar en informes tabulados, imprimir, guardar, etc., dejando el servidor libre.

SQL permite:

- Definir una base de datos mediante tablas.
- Almacenar información en tablas.
- Seleccionar la información que sea necesaria de la base de datos.
- Realizar cambios en la información y estructura de los datos.
- Combinar y calcular datos para conseguir la información necesaria.

Estas tareas se encuentran divididas en dos bloques, uno es el DDL (Lenguaje de definición de datos), que es el encargado de generar y definir la estructura que almacenará la información en la base de datos y el DML (Lenguaje de Manipulación de datos) que permite agregar, recuperar, modificar o eliminar información de la estructura definida.

Comandos DDL	
Create	Utilizado para crear nuevas tablas, campos e índices.
Drop	Empleado para eliminar bases de datos, tablas e índices.
Alter	Utilizado para modificar la estructura de datos.

Comandos DML	
Select	Utilizado para consultar registros de la base de datos que satisfagan un criterio determinado
Insert	Utilizado para cargar lotes de datos en la base de datos en una única operación.
Update	Utilizado para modificar los valores de los campos y registros especificados
Delete	Utilizado para eliminar registros de una tabla de una base de datos

SQL cuenta con otros conjuntos de instrucciones que permiten realizar acotaciones, agrupaciones y operaciones desde el mismo manejador de bases de datos, estas instrucciones son operadores lógicos, cláusulas, operadores de comparación y funciones de agregado.

Hasta el momento se han mencionado herramientas para desarrollo de aplicaciones que permiten generar en conjunto una plataforma de desarrollo, conceptos como manejadores de bases de datos y el lenguaje estructurado de consulta, pero no se ha tratado nada acerca de bases de datos e un nivel mas profundo en conceptualización ni tampoco sobre herramientas para el diseño de sistemas que permitan definir estructuras robustas y sistemas confiables, es por ello que a continuación se hablará de ello, empezando por una introducción que dará al lector una idea mas clara de los que son las bases de datos.

1.9 Bases de datos[12]

Existen diferentes modelos para el diseño de bases de datos, el modelo de red, el modelo jerárquico y el modelo entidad-relación, en la actualidad el modelo entidad-relación es el mas difundido por su facilidad de uso, comprensión y manera de poder modelar en él problemas del mundo real en un modelo computacionalmente factible, robusto y bien estructurado guardando la definición de las entidades (objetos del mundo real) y sus relaciones con otras entidades.

El modelo relacional fue propuesto originariamente por E.F. Codd en un ya famoso artículo de 1970. Gracias a su coherencia y facilidad de uso, el modelo se ha convertido en los años 80 en el más usado para la producción de DBMS.

La estructura fundamental del modelo relacional es precisamente esa, "relación", es decir una tabla bidimensional constituida por líneas (tuplas) y columnas (atributos). Las relaciones representan las entidades que se consideran interesantes en la base de datos. Cada instancia de la entidad encontrará sitio en una tupla de la relación, mientras que los atributos de la relación representarán las propiedades de la entidad.

En realidad, siendo rigurosos, una relación es sólo la definición de la estructura de la tabla, es decir su nombre y la lista de los atributos que la componen. Cuando se puebla con las tuplas, se habla de "instancia de relación".

Las tuplas en una relación son un conjunto en el sentido matemático del término, es decir una colección no ordenada de elementos diferentes. Para distinguir una tupla de otra, se recurre al concepto de "llave primaria", o sea a un conjunto de atributos que permiten identificar unívocamente una tupla en una relación. Naturalmente, en una relación puede haber más combinaciones de atributos que permitan identificar unívocamente una tupla ("llaves candidatas"), pero entre éstas se elegirá una sola para utilizar como llave primaria. Los atributos de la llave primaria no pueden asumir el valor nulo (que significa un valor no determinado), en tanto que ya no permitirían identificar una tupla concreta en una relación. Esta propiedad de las relaciones y de sus llaves primarias está bajo el nombre de integridad de las entidades (entity integrity).

A menudo, para obtener una llave primaria "económica", es decir compuesta de pocos atributos fácilmente manipulables, se introducen uno o más atributos ficticios, con códigos identificativos unívocos para cada tupla de la relación.

Cada atributo de una relación se caracteriza por un nombre y por un dominio. El dominio indica qué valores pueden ser asumidos por una columna de la relación. A menudo un dominio se define a través de la declaración de un tipo para el atributo (por ejemplo diciendo que es una cadena de diez caracteres), pero también es posible definir dominios más complejos y precisos. Por ejemplo, para el atributo "sexo" de "Personas" se puede definir un dominio por el cual los únicos valores válidos son 'M' y 'F'; o bien por el atributo "fecha_nacimiento" se definirá un dominio por el que se consideren válidas sólo las fechas de nacimiento después del uno de enero de 1960, si en la base de datos no está previsto que haya personas con fecha de nacimiento anterior a esa. El motor de datos se ocupará de controlar que en los atributos de las relaciones se incluyan sólo los valores permitidos por sus dominios. Característica fundamental de los dominios de una base de datos relacional es que sean "atómicos", es decir que los valores contenidos en las columnas no se puedan separar en valores de dominios más simples. Más formalmente se dice que no es posible tener atributos multivalor (multivalued).

[12] Información recopilada de la referencia [3L], [3], [8]

Una de las grandes ventajas del modelo relacional es que define también un álgebra, llamada "álgebra relacional". Todas las manipulaciones posibles sobre las relaciones se obtienen gracias a la combinación de tan sólo cinco operadores: RESTRICT, PROJECT, TIMES, UNION y MINUS. Por comodidad, se han definido también tres operadores adicionales que de todos modos se pueden obtener aplicando los cinco fundamentales: JOIN, INTERSECT y DIVIDE. Los operadores relacionales reciben como argumento una relación o un conjunto de relaciones y restituyen una única relación como resultado.

A continuación se presenta una breve descripción de estos operadores.

RESTRICT
: restituye una relación que contiene un subconjunto de las tuplas de la relación a la que se aplica. Los atributos se quedan como estaban.

PROJECT
: restituye una relación con un subconjunto de los atributos de la relación a la que viene aplicado. Las tuplas de la relación resultado se componen de las tuplas de la relacion original, de manera que siguen siendo un conjunto en sentido matemático.

TIME
: se aplica a dos relaciones y efectúa el producto cartesiano de las tuplas. Cada tupla de la primera relación está concatenada con cada tupla de la segunda.

JOIN
: se concatenan las tuplas de dos relaciones de acuerdo con el valor de un conjunto de sus atributos.

UNION
: aplicando este operador a dos relaciones compatibles, se obtiene una que contiene las tuplas de ambas relaciones. Dos relaciones son compatibles si tienen el mismo número de atributos y los atributos correspondientes en las dos relaciones tienen el mismo dominio.

MINUS
: aplicado a dos relaciones compatibles restituye una tercera que contiene las tuplas que se encuentran sólo en la primera relación.

INTERSECT
: aplicado a dos relaciones compatibles restituye una relación que contiene las tuplas que existen en ambas.

DIVIDE
: aplicado a dos relaciones que tengan atributos comunes, restituye una tercera que contiene todas las tuplas de la primera relación que se puede hacer que correspondan con todos los valores de la segunda relación.

Las bases de datos relacionales efectúan todas las operaciones en las tablas usando el álgebra relacional, aunque normalmente no le permiten al usuario usarla.

Al hablar de bases de datos se dispone de una cantidad de material enorme, con el cual se pueden construir varias tesis, aquí la idea es dar al lector una idea breve y concisa de que y como funcionan las herramientas que se utilizarán para crear el sistema que se menciona al inicio de éste capítulo, ahora se dará una introducción de lo que es la Programación Orientada a Objetos y posteriormente se hablará del Lenguaje de Modelado Unificado (UML).

1.10 Programación Orientada a Objetos

Actualmente una de las áreas más candentes en la industria y en el ámbito académico es la orientación a objetos. La orientación a objetos promete mejoras de amplio alcance en la forma de diseño, desarrollo y mantenimiento del software ofreciendo una solución a largo plazo a los problemas y preocupaciones que han existido desde el comienzo en el desarrollo de software: la falta de portabilidad del código y reusabilidad, código que es difícil de modificar, ciclos de desarrollo largos y técnicas de codificación no intuitivas.

Un lenguaje orientado a objetos ataca estos problemas. Tiene tres características basicas: debe estar basado en objetos, basado en clases y capaz de tener herencia de clases. Muchos lenguajes cumplen uno o dos de estos puntos; muchos menos cumplen los tres. La barrera más difícil de sortear es usualmente la herencia.

El concepto de programación orientada a objetos (OOP) no es nuevo, lenguajes clásicos como SmallTalk se basan en ella. Dado que la OOP. se basa en la idea natural de la existencia de un mundo lleno de objetos y que la resolución del problema se realiza en términos de objetos, un lenguaje se dice que está basado en objetos si soporta objetos como una característica fundamental del mismo.

El elemento fundamental de la OOP es, como su nombre lo indica, el objeto. Podemos definir un objeto como un conjunto complejo de datos y programas que poseen estructura y forman parte de una organización.

Esta definición especifica varias propiedades importantes de los objetos. En primer lugar, un objeto no es un dato simple, sino que contiene en su interior cierto número de componentes bien estructurados. En segundo lugar, cada objeto no es un ente aislado, sino que forma parte de una organización jerárquica o de otro tipo.

1.10.1 ESTRUCTURA DE UN OBJETO

Un objeto puede considerarse como una especie de cápsula dividida en tres partes:

- Relaciones
- Propiedades
- Métodos

Cada uno de estos componentes desempeña un papel totalmente independiente:

Las relaciones permiten que el objeto se inserte en la organización y están formadas esencialmente por punteros a otros objetos.

Las propiedades distinguen un objeto determinado de los restantes que forman parte de la misma organización y tiene valores que dependen de la propiedad de que se trate. Las propiedades de un objeto pueden ser heredadas a sus descendientes en la organización.

Los métodos son las operaciones que pueden realizarse sobre el objeto, que normalmente estarán incorporados en forma de programas (código) que el objeto es capaz de ejecutar y que también pone a disposición de sus descendientes a través de la herencia.

1.10.2 Encapsulamiento y ocultación

Como se ha visto, cada objeto es una estructura compleja en cuyo interior hay datos y programas, todos ellos relacionados entre sí, como si estuvieran encerrados conjuntamente en una cápsula. Esta propiedad (encapsulamiento), es una de las características fundamentales en la OOP.

Los objetos son inaccesibles, e impiden que otros objetos, los usuarios, o incluso los programadores conozcan cómo está distribuída la información o qué información hay disponible. Esta propiedad de los objetos se denomina ocultación de la información.

Esto no quiere decir, sin embargo, que sea imposible conocer lo necesario respecto a un objeto y a lo que contiene. Si así fuera no se podría hacer gran cosa con él. Lo que sucede es que las peticiones de información a un objeto deben realizarse a través de mensajes dirigidos a él, con la orden de realizar la operación pertinente. La respuesta a estas órdenes será la información requerida, siempre que el objeto considere que quien envía el mensaje está autorizado para obtenerla.

El hecho de que cada objeto sea una cápsula facilita enormemente que un objeto determinado pueda ser transportado a otro punto de la organización, o incluso a otra organización totalmente diferente que precise de él. Si el objeto ha sido bien construído, sus métodos seguirán funcionando en el nuevo entorno sin problemas. Esta cualidad hace que la OOP sea muy apta para la reutilización de programas.

1.10.3 Organización de los objetos

En principio, los objetos forman siempre una organización jerárquica, en el sentido de que ciertos objetos son superiores a otros de cierto modo.

Existen varios tipos de jerarquías: serán simples cuando su estructura pueda ser representada por medio de un "arbol". En otros casos puede ser más compleja.

En cualquier caso, sea la estructura simple o compleja, podrán distinguirse en ella tres niveles de objetos.

La raíz de la jerarquía. Se trata de un objeto único y especial. Este se caracteríza por estar en el nivel más alto de la estructura y suele recibir un nombre muy genérico, que indica su categoría especial, como por ejemplo objeto madre, Raíz o Entidad.

Los objetos intermedios. Son aquellos que descienden directamente de la raíz y que a su vez tienen descendientes. Representan conjuntos o clases de objetos, que pueden ser muy generales o muy especializados, según la aplicación. Normalmente reciben nombres genéricos que denotan al conjunto de objetos que representan, por ejemplo, VENTANA, CUENTA, FICHERO. En un conjunto reciben el nombre de clases o tipos si descienden de otra clase o subclase.

Los objetos terminales. Son todos aquellos que descienden de una clase o subclase y no tienen descendientes. Suelen llamarse casos particulares, instancias o ítems porque representan los elementos del conjunto representado por la clase o subclase a la que pertenecen.

Ahora se verán en detalle los tres elementos mencionados en "Estructura de un Objeto".

Relaciones:

Las relaciones entre objetos son, precisamente, los enlaces que permiten a un objeto relacionarse con aquellos que forman parte de la misma organización, habiendo dos tipos fundamentales de relaciones.

Relaciones jerárquicas. Son esenciales para la existencia misma de la aplicación porque la construyen. Son bidireccionales, es decir, un objeto es padre de otro cuando el primer objeto se encuentra situado inmediatamente encima del segundo en la organización en la que ambos forman parte; asimismo, si un objeto es padre de otro, el segundo es hijo del primero (en la fig. 2, B es padre de D,E y F, es decir, D,E y F son hijos de B; en la fig. 3, los objetos B y C son padres de F, que a su vez es hijo de ambos).

Una organización jerárquica simple puede definirse como aquella en la que un objeto puede tener un solo padre, mientras que en una organizacion jerárquica compleja un hijo puede tener varios padres).

Relaciones semánticas. Se refieren a las relaciones que no tienen nada que ver con la organización de la que forman parte los objetos que las establecen. Sus propiedades y consecuencia solo dependen de los objetos en sí mismos (de su significado) y no de su posición en la organización.

Propiedades:

Todo objeto puede tener cierto número de propiedades, cada una de las cuales tendrá, a su vez, uno o varios valores. En OOP, las propiedades corresponden a las clásicas "variables" de la programación estructurada. Son, por lo tanto, datos encapsulados dentro del objeto, junto con los métodos (programas) y las relaciones (punteros a otros objetos). Las propiedades de un objeto pueden tener un valor único o pueden contener un conjunto de valores mas o menos estructurados (matrices, vectores, listas, etc.). Además, los valores pueden ser de cualquier tipo (numérico, alfabético, etc.) si el sistema de programación lo permite.

Pero existe una diferencia con las "variables", y es que las propiedades se pueden heredar de unos objetos a otros. En consecuencia, un objeto puede tener una propiedad de maneras diferentes:

- Propiedades propias. Están formadas dentro de la cápsula del objeto.
- Propiedades heredadas. Estan definidas en un objeto diferente, antepasado de éste (padre,"abuelo", etc.). A veces estas propiedades se llaman propiedades miembro porque el objeto las posee por el mero hecho de ser miembro de una clase.

Métodos:

Una operación que realiza acceso a los datos. Podemos definir método como un programa procedimental o procedural escrito en cualquier lenguaje, que está asociado a un objeto determinado y cuya ejecución sólo puede desencadenarse a través de un mensaje recibido por éste o por sus descendientes.

Son sinónimos de 'método' todos aquellos términos que se han aplicado tradicionalmente a los programas, como procedimiento, función, rutina, etc. Sin embargo, es conveniente utilizar el término 'método' para que se distingan claramente las propiedades especiales que adquiere un programa en el entorno OOP, que afectan fundamentalmente a la forma de invocarlo (únicamente a través de un mensaje) y a su campo de acción, limitado a un objeto y a sus descendientes, aunque posiblemente no a todos.

Si los métodos son programas, se deduce que podrían tener argumentos, o parámetros. Puesto que los métodos pueden heredarse de unos objetos a otros, un objeto puede disponer de un método de dos maneras diferentes:

- Métodos propios. Están incluídos dentro de la cápsula del objeto.
- Métodos heredados. Estan definidos en un objeto diferente, antepasado de éste (padre,"abuelo", etc.). A veces estos métodos se llaman métodos miembro porque el objeto los posee por el mero hecho de ser miembro de una clase.

1.10.4 Polimorfísmo

Una de las características fundamentales de la OOP es el polimorfísmo, que no es otra cosa que la posibilidad de construir varios métodos con el mismo nombre, pero con relación a la clase a la que pertenece cada uno, con comportamientos diferentes. Esto conlleva la habilidad de enviar un mismo mensaje a objetos de clases diferentes. Estos objetos recibirían el mismo mensaje global pero responderían a él de formas diferentes; por ejemplo, un mensaje "+" a un objeto ENTERO significaría suma, mientras que para un objeto STRING significaría concatenación ("pegar" strings uno seguido al otro)

1.10.5 Consideraciones

Beneficios que se obtienen del desarrollo con OOP:

Día a día los costos del Hardware decrecen. Así surgen nuevas áreas de aplicación cotidianamente: procesamiento de imágenes y sonido, bases de datos multimediales, automatización de oficinas, ambientes de ingeniería de software, etc. Aún en las aplicaciones tradicionales encontramos que definir interfases hombre-máquina "a-la-Windows" suele ser bastante conveniente.

Lamentablemente, los costos de producción de software siguen aumentando; el mantenimiento y la modificación de sistemas complejos suele ser una tarea trabajosa; cada aplicación, (aunque tenga aspectos similares a otra) suele encararse como un proyecto nuevo, etc.

Todos estos problemas aún no han sido solucionados en forma completa. Pero como los objetos son portables (teóricamente) mientras que la herencia permite la reusabilidad del código orientado a objetos, es más sencillo modificar código existente porque los objetos no interaccionan excepto a través de mensajes; en consecuencia un cambio en la codificación de un objeto no afectará la operación con otro objeto siempre que los métodos respectivos permanezcan intactos. La introducción de tecnología de objetos como una herramienta concepual para analizar, diseñar e implementar aplicaciones permite obtener aplicaciones más modificables, fácilmente extendibles y a partir de componentes reusables. Esta reusabilidad del código disminuye el tiempo que se utiliza en el desarrollo y hace que el desarrollo del software sea mas intuitivo porque la gente piensa naturalmente en términos de objetos más que en términos de algoritmos de software.

1.10.6 Problemas derivados de la utilización de OOP en la actualidad

Un sistema orientado a objetos, por lo visto, puede parecer un paraíso virtual. El problema sin embargo surge en la implementación de tal sistema. Muchas compañías oyen acerca de los beneficios de un sistema orientado a objetos e invierten gran cantidad de recursos luego comienzan a darse cuenta que han impuesto una nueva cultura que es ajena a los programadores actuales. Específicamente los siguientes temas suelen aparecer repetidamente:

Curvas de aprendizaje largas. Un sistema orientado a objetos ve al mundo en una forma única. Involucra la conceptualización de todos los elementos de un programa, desde subsistemas a los datos, en la forma de objetos. Toda la comunicación entre los objetos debe realizarse en la forma de mensajes. Esta no es la forma en que están escritos los programas orientados a objetos actualmente; al hacer la transición a un sistema orientado a objetos la mayoría de los programadores deben capacitarse nuevamente antes de poder usarlo.

Dependencia del lenguaje. A pesar de la portabilidad conceptual de los objetos en un sistema orientado a objetos, en la práctica existen muchas dependencias. Muchos lenguajes orientados a objetos están compitiendo actualmente para dominar el mercado. Cambiar el lenguaje

de implementación de un sistema orientado a objetos no es una tarea sencilla; por ejemplo C++ soporta el concepto de herencia multiple mientras que SmallTalk no lo soporta; en consecuencia la elección de un lenguaje tiene ramificaciones de diseño muy importamtes.

Determinación de las clases:

Una clase es un molde que se utiliza para crear nuevos objetos. En consecuencia es importante crear el conjunto de clases adecuado para un proyecto. Desafortunadamente la definición de las clases es más un arte que una ciencia. Si bien hay muchas jerarquías de clase predefinidas usualmente se deben crear clases específicas para la aplicación que se este desarrollando. Luego, en 6 meses ó 1 año se da cuenta que las clases que se establecieron no son posibles; en ese caso será necesario reestructurar la jerarquía de clases devastando totalmente la planificación original.

Performance:

En un sistema donde todo es un objeto y toda interaccion es a través de mensajes, el tráfico de mensajes afecta la performance. A medida que la tecnología avanza y la velocidad de microprocesamiento, potencia y tamaño de la memoria aumentan, la situación mejorará; pero en la situación actual, un diseño de una aplicación orientada a objetos que no tiene en cuenta la performance no será viable comercialmente.

Idealmente, habría una forma de atacar estos problemas eficientemente al mismo tiempo que se obtienen los beneficios del desarrollo de una estrategia orientada a objetos. Debería existir una metodología fácil de aprender e independiente del lenguaje, y facil de reestructurar que no drene la performance del sistema.

1.11 UML[13]

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

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

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

[13] Información recopilada de la referencia [5], [9] y [10]

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

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

Como el lector habrá podido darse cuenta, el sistema que se plantea desarrollar involucra tecnologías de telecomunicaciones, lenguajes de programación y herramientas de diseño bastante robustas que se han posicionado en el ámbito computacional de tal manera que la conjugación de todas ellas permite obtener una plataforma de desarrollo bastante fiable que con un adecuado diseño de la aplicación y conocimiento eficiente de las herramientas tendrán que dar como resultado una aplicación abierta que tenga la flexibilidad de trabajar en diferentes escenarios, manejando la información de la manera que se espera y soportada en un desarrollo bien estructurado y diseñado.

Capítulo II. Análisis y diseño

Durante el presente capítulo se habla sobre el análisis y diseño del sistema. Estos elementos son utilizados por los programadores para construir el sistema bajo una metodología orientada a objetos para el desarrollo de aplicaciones de WEB, esta metodología se compone de diferentes elementos que permiten conocer a detalle cada uno de los componentes que intervienen durante la implementación, pruebas, implementación y mantenimiento del producto final.

2.1 Descripción del problema

Se desea automatizar el proceso de seguimiento de ordenes de servicio, entiéndase al concepto de ordenes de servicio como la reparación de un objeto, soporte en línea, consultoría y en general cualquier proceso del cual se tenga que llevar seguimiento.

La solución de automatización se hará mediante la creación de un sistema de cómputo basado en WEB bajo el esquema de Intranet, lo cual implica una aplicación cliente-servidor con restricción de acceso a la información, mediante módulos independientes que serán invocados y liberados para trabajar con ellos mediante la autentificación del usuario que inicia la aplicación.

Tipos de usuarios

- Administrador del sistema
- Encargado de área
- Técnico
- Recepción
- Cliente

Cada tipo de usuario estará ligado a un perfil de usuario que será el que se encargue de guardar la información sobre los módulos e información a la que tendrán acceso.

La información que se manipulará por medio del sistema, es la siguiente:

Entidades

- Cliente
- Orden de servicio
- Empleado
- Área
- Tipos de servicio (Con los pasos que los componen).

Procesos

- Asignación de órdenes de servicio.
- Seguimiento de órdenes de servicio.
- Envío de mensajes

La información estará disponible de acuerdo a la siguiente tabla:

Tipo de usuario	crear	modificar	visualizar	eliminar	concluir	aceptarConclusion	mensaje	seguimientoAgregar	seguimientoVisualizar	asignacion	aceptación	modificarAceptacion	clienteAgregar	clienteModificar	clienteEliminar	empleados	tipoServicio	areas
Administrador	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S
Encargado de area	N	S	S	S	S	S	S	S	S	S	N	S	N	N	N	N	N	N
Tecnico	N	N	S	N	N	N	S	S	S	N	S	N	N	N	N	N	N	N
Recepcion	S	N	S	N	N	N	S	N	S	N	N	N	S	N	N	N	N	N
Cliente	S	N	S	N	N	N	S	N	S	N	N	N	N	N	N	N	N	N

Tabla 2.1 Permisos y privilegios de acceso al sistema para los diferentes tipos de usuarios.

S=Acceso permitido
N=Acceso denegado

En la Tabla 2.1 se describen diferentes acciones generales que se encontrarán disponibles en el sistema, se deberá poder activar una tarea se manera automática con tan solo habilitarla en la estructura que guarde la información de la tabla anterior.

El lector se preguntará cual es la necesidad de 5 diferentes tipos de usuarios. Esto se se debe a que el Administrador será el encargado de manipular toda la información por el carácter mismo que tiene de administrador y fungirá en un momento dado como un mediador entre los otros cuatro tipos de usuario supervisando el proceso y el seguimiento.

El usuario Encargado de Área, estará a cargo de un área de producción dentro de la empresa, en particular en una de las áreas en donde se presten servicios administrados por el presente sistema, realizando las tareas estipuladas en la tabla y solo sobre la información que involucra a su área.

El usuario Técnico será el encargado de llevar a cabo el servicio y dar seguimiento a nivel de ejecución, la palabra seguimiento emplea desde un contexto en el cual se deben cumplir cada uno de los pasos específicos de una tarea de a cuerdo a la definición del tipo de servicio.

Recepción es un tipo de usuario especial que permite crear órdenes de servicio y dar de alta clientes para recepción de peticiones vía telefónica por medio de un Call Center o recurso similar. El tipo Recepción puede ser visto como un caso especial de cliente, solo que éste podrá asignar un cliente específico a una nueva orden de servicio y en el caso del cliente, su nombre estará preestablecido cuando se cargue la forma de captura, por lo cual será manejado como un cliente más.

El Cliente es la persona que solicita un servicio por parte de la empresa.

A continuación se presentan documentos resultado de la aplicación de metodologías y herramientas de desarrollo de software que permiten analizar la información existente y arrojar elementos que permiten describir en un argot computacional y entendible a las personas en general el funcionamiento esperado del sistema y el manejo de cada elemento y proceso que se busca automatizar o controlar.

2.2 Análisis

El análisis al igual que el diseño es un proceso importante en el desarrollo de sistemas, en él se hará uso de herramientas especializadas para el diseño de bases de datos, análisis de procesos y flujo de información que permitan generar a detalle un mapa claro del funcionamiento del sistema.

Con la finalidad de identificar de adecuadamente cada uno de los elementos que intervienen en el sistema, éste se dividirá en su análisis y diseño de acuerdo a la información que se maneja tanto para la manera de trabajar internamente como para la forma en que el usuario interactuará con él.

Los elementos principales sobre los cuales se trabajará son los siguientes.

- ORDENES DE SERVICIO: Esta es la parte fundamental y la razón de existir del sistema, aquí habrá tareas a las que los diferentes usuarios tendrán acceso de acuerdo a su perfil y será el módulo al que más tiempo podría consumir en desarrollo por la cantidad de validaciones y procesos que incluye.
- CLIENTES: Permitirá administrar los usuarios dentro del sistema que son clientes de la empresa para la cual se pondrá en marcha el sistema.
- EMPLEADOS: Aquí se incluirán todos los elementos que permitan administrar a los empleados que se encuentren laborando dentro de la empresa, es un módulo exclusivamente para acceso del administrador general del sistema y además permitirá establecer relación entre un empleado y el área a la que pertenece así como definir al encargado de un área.
- AREA: Define cada una de las áreas que prestan servicio a los clientes como área de Staff o producción dentro de la empresa y hacia fuera de la empresa.
- TIPOS DE SERVICIO: Permite definir los servicios asociados a un área. Está es una entidad que incluye la definición de la estructura de un tipo de servicio, con la finalidad de dar seguimiento paso a paso de la ejecución de la orden de servicio.

La información que el sistema manejará, debe de almacenarse de manera adecuada. Para tal efecto existen diferentes maneras de llevar a cabo tal tarea, en éste caso, se ha optado por utilizar un Modelo Entidad-Relación. En la Figura 2.1 se presenta el diagrama Entidad- Relación que modela la estructura en la cual se almacenará la información del sistema.

2.2.1 Modelo E-R

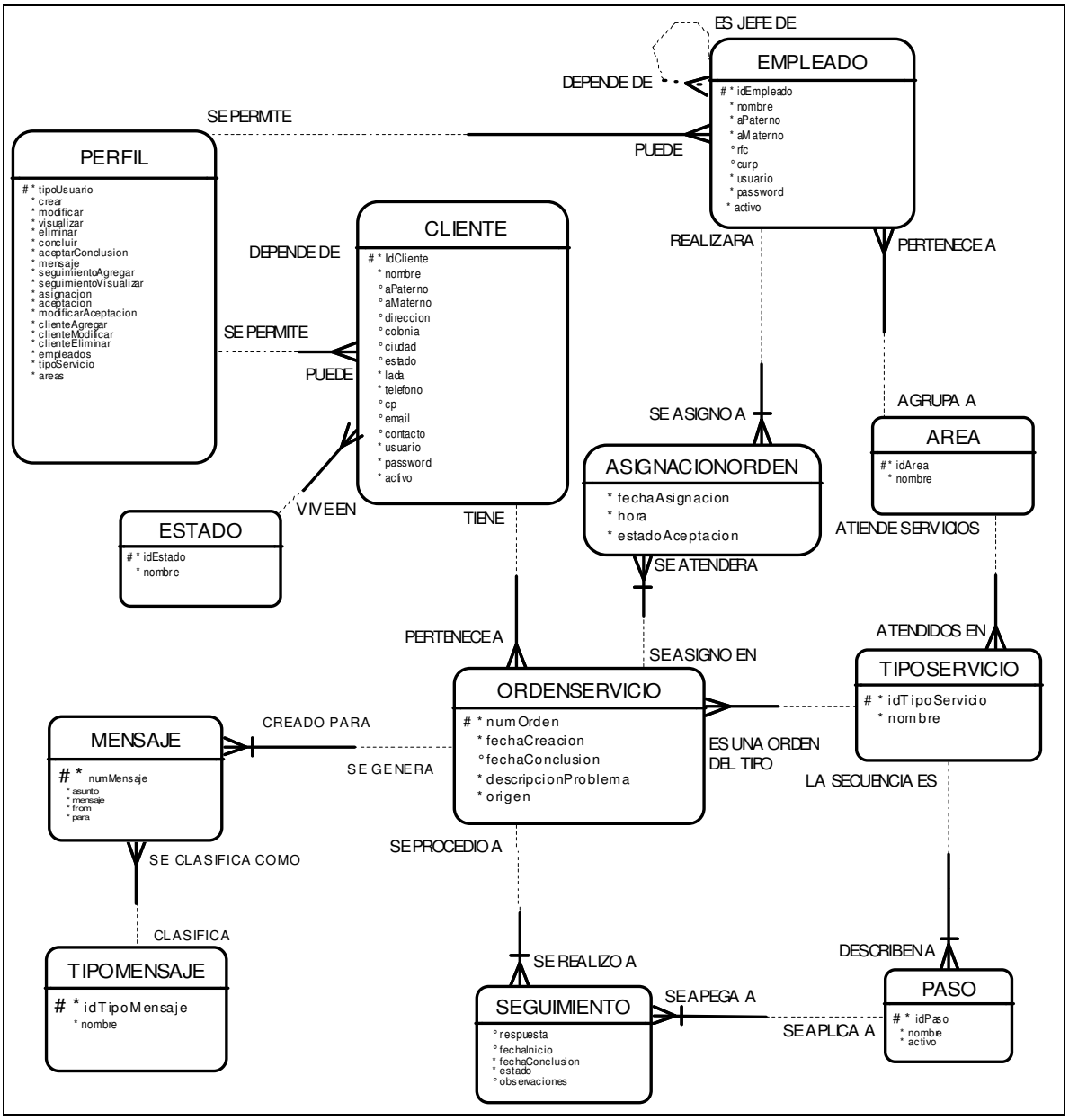

Figura 2.1: Diagrama Entidad-Relación de la Base de datos

En el diagrama Entidad-Relación (Figura 2.1) se aprecian cada una de las entidades que formaran parte del sistema, las entidades guardan relaciones entre si y en algunas existen relaciones recursivas ya que por la naturaleza del sistema resultan necesarias.

Las entidades principales de la estructura son:

- ORDENSERVICIO: Almacenará la información referente a cada orden de servicio que se de de alta en el sistema.
- EMPLEADO: Permitirá almacenar la información de los empleados a los cuales se les asignará la ejecución de una orden de servicio, cada empleado está aginado a

un área, cada área tiene un jefe de área (Relación recursiva) y hay un usuario administrador el cual tiene a su cargo a cada jefe de área.
- CLIENTE: Almacenará la información de los clientes dados de alta en el sistema.

Entidades de catalogo:
- AREA: Guardará la información de cada área de servicio, permite identificar a que área pertenece cada servicio, a que área está asignado cada empleado, tomando en cuanta que un empleado solo puede estar asignado a un área.
- TIPOSERVICIO: Manejará la información de los servicios que se tienen disponibles para cada área y sus pasos están definidos en la entidad PASO.
- PASO: Almacenará los pasos que deben cumplirse para cada servicio que se tiene definido en la entidad TIPOSERVICIO.
- SEGUIMIENTO: Permite guardar la información de cada paso por el cual ha pasado una orden de servicio, estableciendo un candado para no ejecutar un paso sin antes haber cumplido con los que le preceden de acuerdo a la tabla PASO.
- PERFIL: Establece los privilegios de acceso de cada uno de los clientes y empleados a cada una de las tareas y estructuras del sistema.
- TIPOMENSAJE: Permite definir cuales son los tipos de mensajes que se pueden enviar por medio del sistema entre los usuarios para una orden de servicio en particular.

Entidades auxiliares en el proceso
- AGINACIONORDEN: Es una relación que permitirá relacionar a una orden de servicio con el empleado que la ejecutará, de igual manera mantendrá un histórico de rechazos por parte de los empleados a la asignación y ejecución de una orden de servicio y de esta manera se podrá medir el rendimiento de los empleados.
- MENSAJE: Guardará los mensajes enviados entre los usuarios por medio del sistema de acuerdo a la tabla TIPOMENSAJE.

2.2.2 Casos de uso

Anteriormente se planteó que el sistema se dividirá en 4 módulos principales para administrar las tareas y acceso al manejo de las estructuras, enseguida se presentan los diferentes casos de uso relacionados con cada uno de éstos módulos y algunos adicionales como sucede con el acceso al sistema.

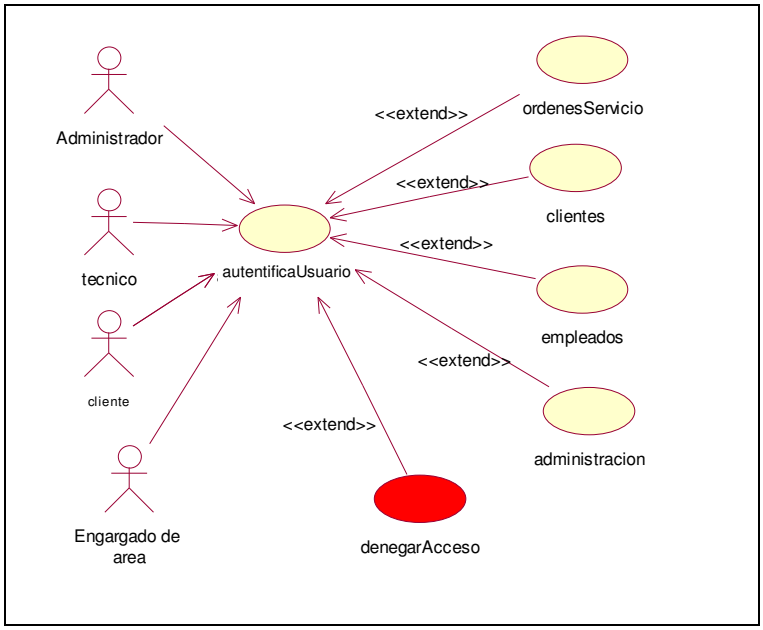

Figura 2.2 Casos de uso para el acceso al sistema

En la Figura 2.2 se presenta el caso de uso para el acceso al sistema, cuando un usuario intenta entrar al sistema, deberá de dar un nombre de usuario y password que permitan autentificarlo, en caso de que ambos parámetros sean válidos, el usuario podrá elegir cualquiera de los módulos hacia los cuales se extiende el caso de uso, solo en el caso de que no sean válidos, se extiende el caso de uso de validación hacia denegarAcceso.

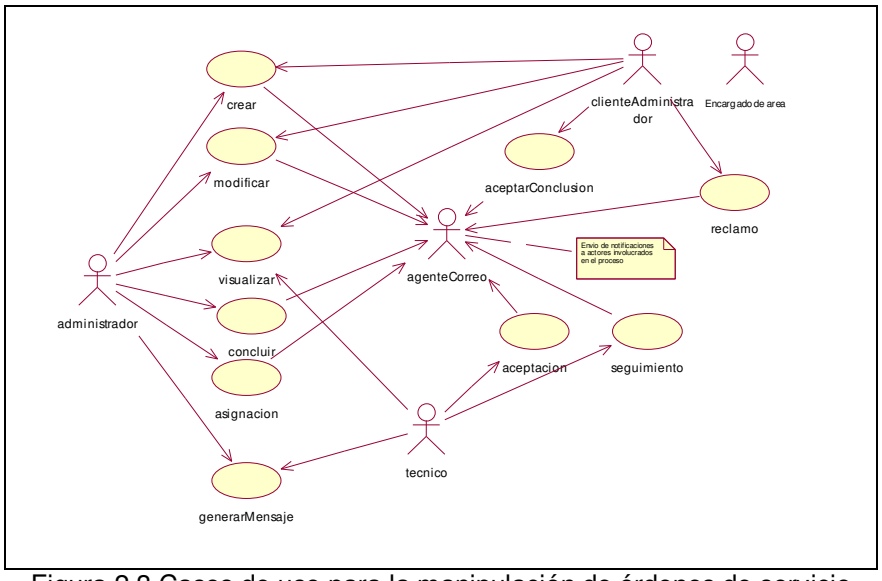

Figura 2.3 Casos de uso para la manipulación de órdenes de servicio

La Figura 2.3 muestra los actores principales del sistema, que son los diferentes tipos de usuarios que pueden tener acceso al sistema y cuales son las acciones que puede realizar cada uno dentro del módulo de administración de órdenes de servicio. Se aprecia un actor más que es un agente de correo, que no es más que el servidor de correo con que se encontrará en el servidor en que montará el sistema y que permitirá que se envíen correos electrónicos a cada uno de los participantes en una orden de servicio.

Aquí se puede preguntar el lector en que parte se han incluido elementos como asignar y dar seguimiento a una orden de servicio, pero estas tareas están implícitas en la tarea modificar, pues al realizar cualquiera de estas tareas, se está modificando a la orden de servicio.

También aparece un caso de uso llamado reclamo, esto establece que el único que puede generar un reclamo es el cliente cuando algo no está de acuerdo a lo estipulado de acuerdo a la definición del servicio en tiempos o forma de llevarlo a cabo.

Conviene recordar que existen encargados y técnicos de diferentes áreas, clientes que no tienen ningún vínculo y un administrador general del sistema, cada usuario del sistema deberá de tener acceso únicamente a las órdenes de servicio que tiene vinculadas ya sea como cliente o como empleado de la empresa. Para el caso de los clientes resulta claro que solo tendrá acceso a aquellas órdenes que tiene vinculadas y los clientes de consulta que estén subordinados a ellos, para el caso de los empleados de la empresa, existe una estructura, en donde el administrador general tiene acceso a todas las ordenes de servicio existentes, para el caso de los encargados de área, solo tendrán acceso a aquellas órdenes de trabajo vinculadas a tipos de servicio de sus áreas y para los técnicos, estos solo pueden visualizar y manipular la información de las órdenes de trabajo que tienen vinculadas por medio de la entidad ASIGNACIONORDEN.

Se puede dar el caso en que un encargado de área se deba encargar de la ejecución de una orden de servicio, por lo cual se deben de tomar las previsiones necesarias e incluso se puede dar el caso para el administrador general.

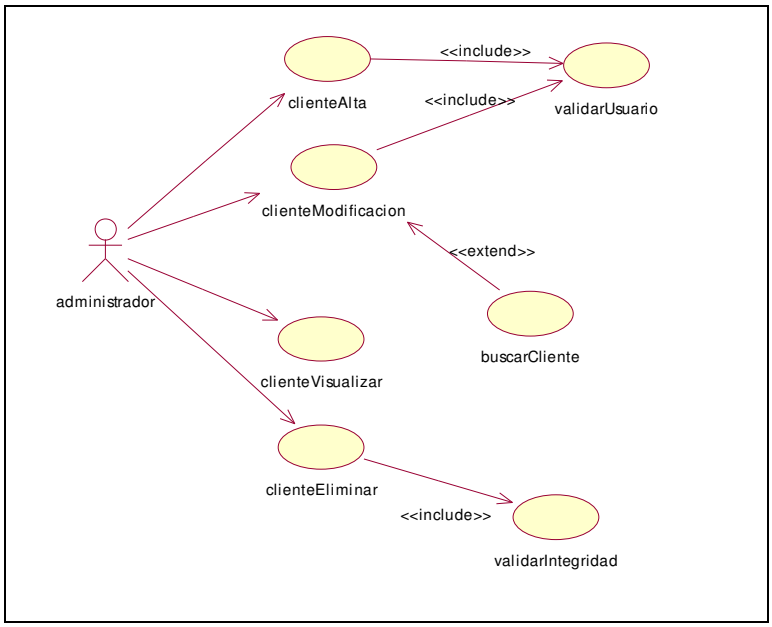

Figura 2.4 Casos de uso para la manipulación de clientes

La Figura 2.4 muestra las tareas principales para la administración de clientes, aquí se hace hincapié en que se trata de un módulo protegido del sistema, con esto se quiere decir que no

cualquier usuario tiene acceso, pues es responsabilidad del administrador del sistema dar de alta a los usuarios, así como clasificarlos.

Se han incluido algunos casos de uso para inclusión que permiten realizar validaciones, esto con la finalidad de mantener integridad en la información para el caso de clienteAlta y clienteEliminar. Es muy importante que se haga notar que un cliente no puede eliminarse si ya existe información relacionada, para tal caso se debe marcar como inactivo mediante la bandera activo en para tal efecto en la entidad CLIENTE.

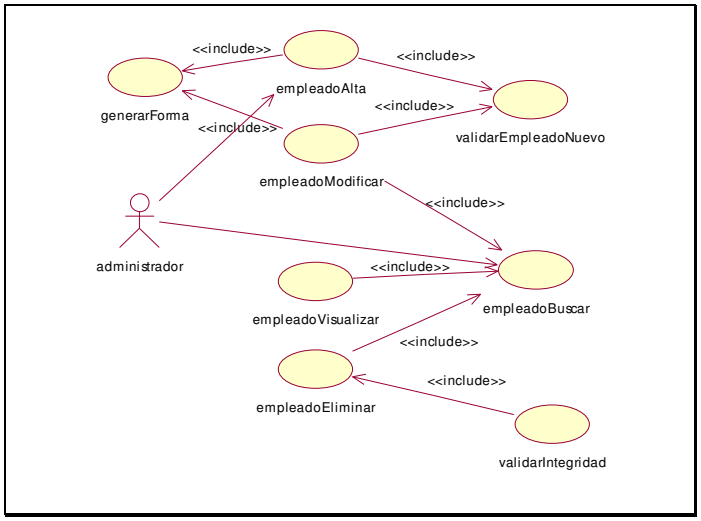

Figura 2.5 Casos de uso para la manipulación de empleados

La Figura 2.5 muestra las diferentes tareas involucradas con la manipulación de los empleados en la base de datos. Como en los casos de uso se muestran las operaciones básicas de una entidad (Alta, baja, modificación, visualización), aquí se debe tener cuidado de definir de manera adecuada a los empleados de acuerdo al tipo de empleado que se desea definir dependiendo si se trata de un técnico o un encargado de área, se deben definir los mecanismos adecuados para que esto se pueda validar o manipular de manera adecuada en la implementación. El único usuario que se dará de alta al momento de crear la base de datos será el administrador general, de ahí en adelante, los demás usuarios deberán darse de alta por medio de la aplicación.

Al igual que el caso de los clientes, se debe tener especial cuidado cuando se realice la eliminación de un empleado pues no puede hacerse si se da el caso que el empleado ya haya tenido información relacionada y lo cual implicaría inconsistencias en la información, para ello y evitar que el usuario pueda seguir teniendo acceso al sistema, se marcará como inactivo en la bandera para ello incluida en la entidad EMPELADO.

A continuación se presentan los diagramas de casos de uso para el módulo de administración, en el cual se manipulan las entidades que tienen que ver con los catálogos del sistema, haciendo notar que éste es un módulo exclusivo para el administrador al igual que el módulo de empleados.

En éste módulo se administrarán las siguientes entidades:

- AREA
- TIPOSERVICIO
- TIPO DE MENSAJES
- REPORTES

En la Figura 2.6 se aprecian las tareas de administración de la entidad AREA, la cual como se menciona anteriormente es una entidad de catálogo, que sirve para clasificar a los empleados,

los servicios, las ordenes de servicio, esto con la finalidad de posteriormente poder generar reportes que permitan saber el rendimiento de un área en especial o de algun empleado específico.

No está por demás tomar en cuenta que con la finalidad de mantener la integridad de la información, la opción de eliminación de un área solo se podrá llevar a cabo cuando el área no tenga información relacionada, en caso contrario, el área no podrá ser eliminada.

Como en algunos casos de uso anteriores, se incluye en éste caso una rutina que crea el formulario de captura, puesto que es un elemento que se utiliza tanto en el alta como en la modificación.

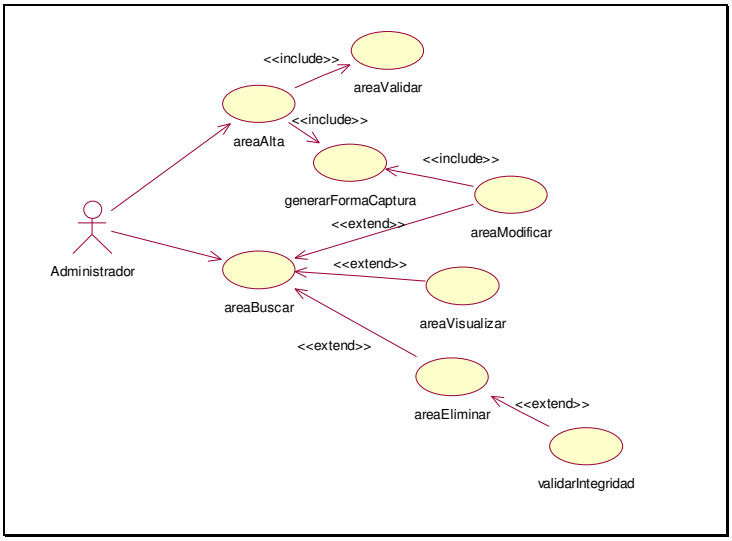

Figura 2.6 Casos de uso para la manipulación de áreas

Los tipos de servicio no son más que los servicios que se tienen disponibles por cada área, en la Figura 2.7 se muestran las diferentes tareas asociadas a ésta entidad. Si se analiza a fondo, se verá que involucra no solo la administración de los tipos de servicio, so no que también se incluye el manejo de los pasos que componen a un servicio.

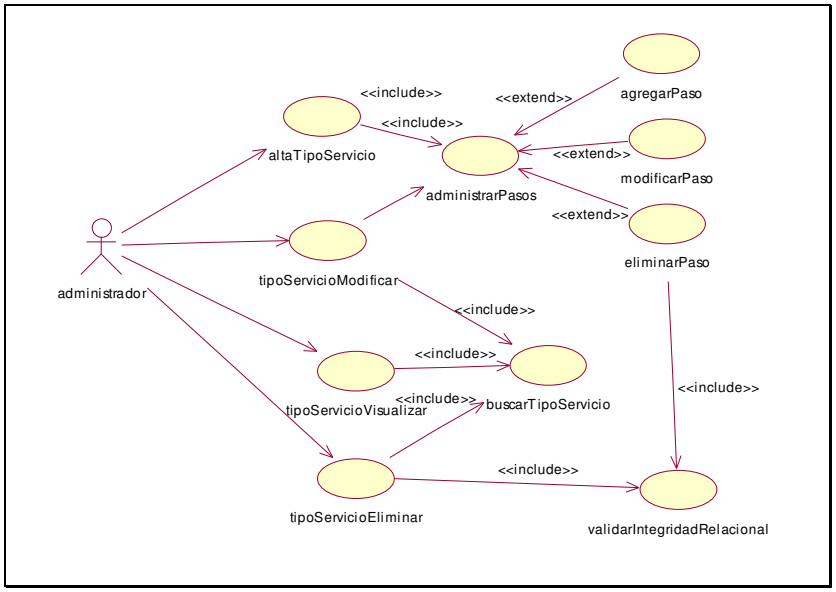

Figura 2.7 Casos de uso para la manipulación de tipos de servicio

Primero que nada el administrador debe de dar de alta el servicio con su información general y una vez realizado esto tendrá que empezar a definir cada uno de los pasos que lo componen. Cada paso tiene un peso, teniendo como menor peso aquel cuyo identificador sea cero y el de mayor peso será el que tenga mayor valor en el identificador, el orden de los pasos se podrá modificar, pudiéndose insertar nuevos pasos intermedios para un servicio ya definido. Aquí se presenta un problema cuando el tipo de servicio ya haya sido utilizado en alguna orden de servicio, cuando esto suceda, deberán de cargarse al seguimiento de las órdenes de servicio los nuevos pasos con la leyenda NA para marcarlos como que no aplicaron pues son resultado de una modificación del proceso.

Para el caso de la eliminación de pasos, esto deberá manipularse por medio de una bandera de estado activo en la entidad paso para poder marcarle a la orden de servicio que el paso ya no se aplica y el mismo será marcado como NA en el seguimiento de la orden de servicio marcando la causa (NA: por actualización del proceso).

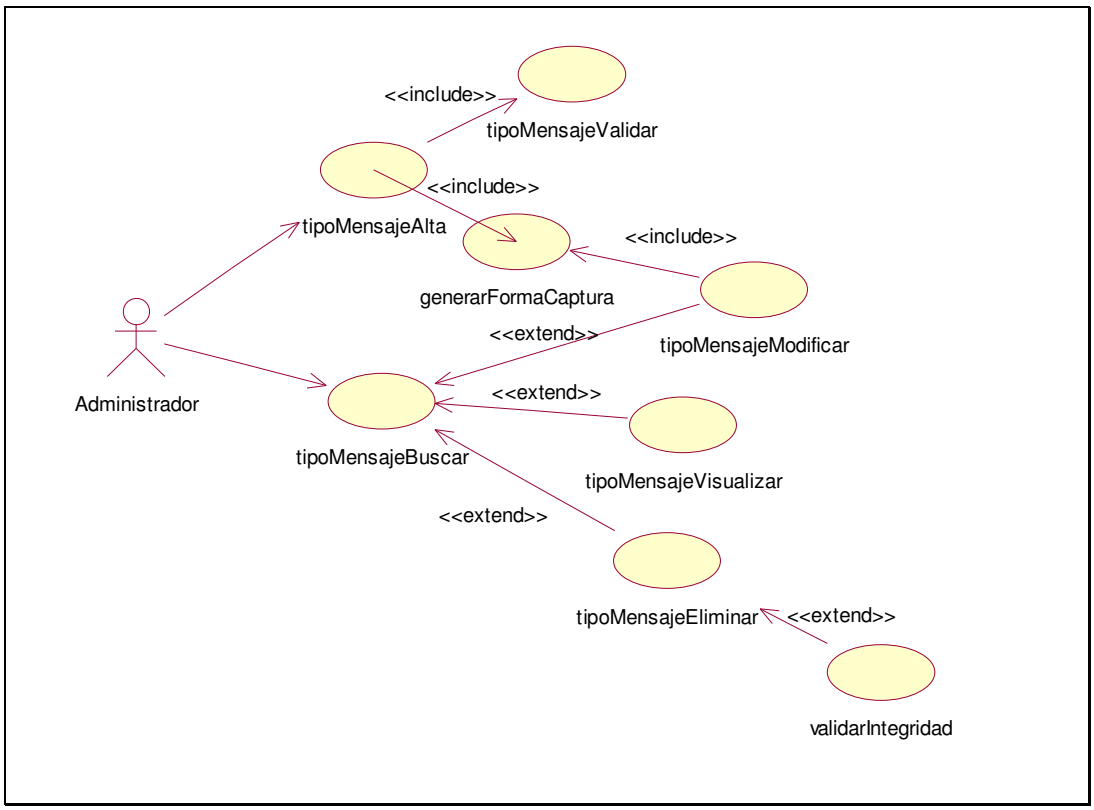

Figura 2.8 Casos de uso para la manipulación de tipos de mensaje

En la Figura 2.8 se muestra el diagrama de casos de uso para la manipulación de tipos de mensaje, como se trata de una entidad de catálogo, su funcionamiento y casos de uso al igual que las entidades anteriores que caen en la misma clasificación, presentan las tareas triviales de alta, modificación, visualización, búsqueda y eliminación o en su caso desactivación para cuando existe alguna regla de integridad que pudiese violarse.

Para la Figura 2.9 se presentan los casos de uso para los reportes que generará el sistema. Cuando se desarrolla un sistema, no basta con automatizar procesos, todo sistema debe generar reportes que permitan evaluar procesos y agilizar la toma de desiciones.

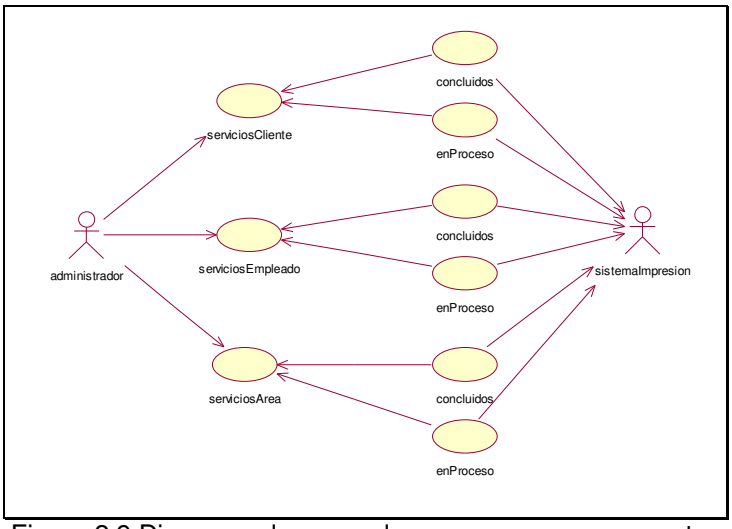

Figura 2.9 Diagrama de casos de uso para generar reportes

Este es un bloque que pertenece al módulo de administración, es exclusivo para uso del administrador, se presentan tres acciones principales que simbolizan las formas que permitirán introducir criterios de búsqueda para rangos por fecha e información que permita acotar los criterios de búsqueda para generar reportes mas puntuales, una vez que la información introducida, el sistema puede mostrar ordenes de servicio concluidas o en proceso de acuerdo a los criterios introducidos por el usuario. Una vez hecho esto, el usuario tiene la opción de poder imprimir el resultado del reporte si así lo requiere.

Hasta el momento se han identificado los procesos principales del sistema, ahora con ayuda de la estructura de datos definida en el modelo entidad relación y los diagramas de casos de uso se presenta la identificación de clases que permitirá aplicar los conceptos de abstracción, encapsulación, modularidad y jerarquía de la programación orientada a objetos ya que el diseño del sistema esta orientado a objetos.

2.2.3 Identificación de clases

A continuación se presentan las clases que se han identificado, con sus métodos asociados.

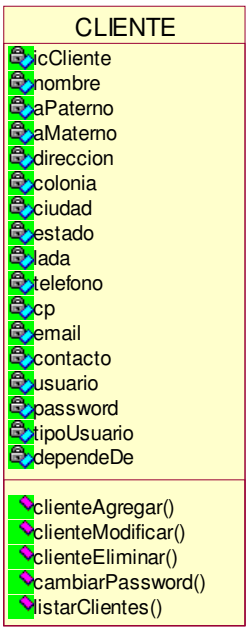

Figura 2.10 Clase CLIENTE

La clase cliente mostrada en la Figura 2.10 permite manipular la información referente a los clientes que tendrán acceso al sistema, respetando las tareas y definiciones hasta ahora estipuladas a lo largo de la documentación para tal efecto.

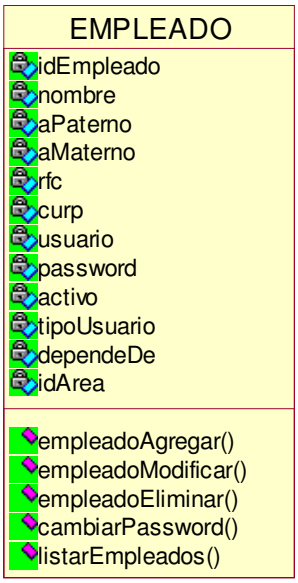

Figura 2.11 Clase EMPLEADO

En la Figura 2.11 se muestra la clase EMPLEADO que incluye los métodos y atributos necesarios para manipular la información de los empleados.

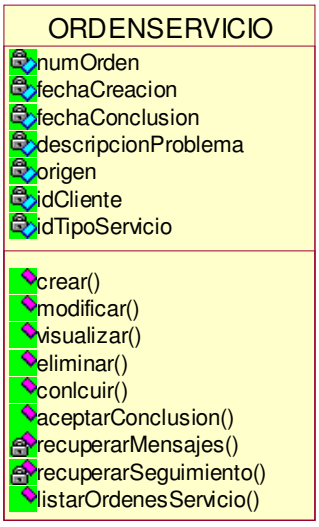

Figura 2.12 Clase ORDENSERVICIO

En la Figura 2.12 se muestran la clase ORDENSERVICIO que permite manipular la información referente a las órdenes de servicio, la cual incluye métodos que permiten manipular información para los mensajes relacionados con una orden de servicio, seguimiento y entidades relacionadas.

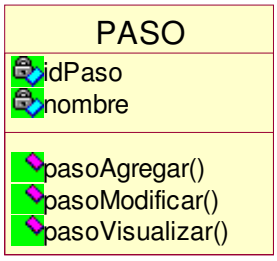

Figura 2.13 Clase PASO

La clase PASO mostrada en la Figura 2.13 permite administrar la entidad de catálogo PASO, la cual permite definir los diferentes pasos que forman parte de un TIPOSERVICIO, el cual no es mas que la definición de un servicio relacionado con un área.

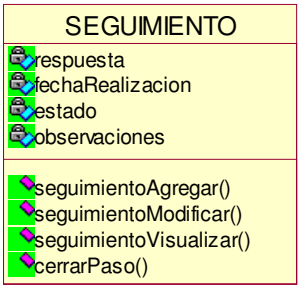

Figura 2.14 Clase SEGUIMIENTO

En la Figura 2.14 se muestra la clase SEGUIMIENTO que es parte fundamental de la razón de ser del sistema, permite dar seguimiento paso a paso del cumplimiento cabal y ejecución

sistematizada de una orden de servicio, esta clase se avoca a generar la información que se almacenará en la entidad SEGUIMIENTO.

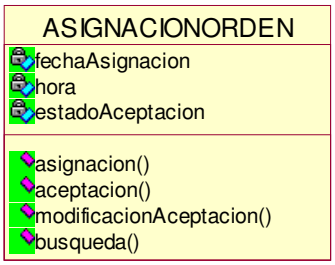

Figura 2.15 Clase ASIGNACIONORDEN

La clase asignación orden, la cual se muestra en la Figura 2.15, es la clase que permite realizar las asignaciones, no aceptación o aceptación de una orden de servicio por parte de un empleado, también permitirá al administrador modificar el valor de la aceptación por medio del método modificación cuando así sea necesario.

Figura 2.16 Clase AREA

La clase AREA que se muestra en la Figura 2.16 permite administrar la información que se almacena en la entidad AREA que es una entidad de catálogo.

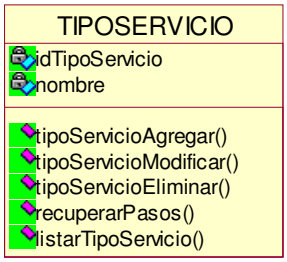

Figura 2.17 Clase TIPOSERVICIO

La clase TIPOSERVICIO es la clase encargada de manipular la información que se almacenará en la entidad TIPOSERVICIO.

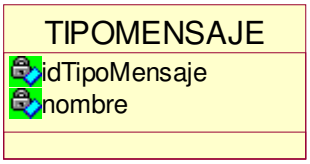

Figura 2.18 Clase TIPOMENSAJE

La clase TIPOMENSAJE almacena la información para catalogar los tipos de mensaje que se pueden generar en el sistema, no es administrable y por ello no cuenta con métodos para tal efecto, se define en la instalación y puesta a punto del sistema.

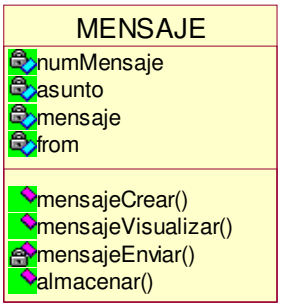

Figura 2.19 Clase MENSAJE

La clase MENSAJE permite manipular la información que se almacenará en la entidad mensaje, dicha entidad almacenará los mensajes que se hayan generado en base a una orden de servicio por los usuarios participantes en la ejecución de la orden en cuestión.

2.3 Diseño

A partir del detalle con que se representa un modelo en UML, por medio del diagrama de clases se peude realizar el mapeo en un DBMS (Data Base Management System, Sistema Administrador de Bases de Datos) por un programador, el diagrama contiene la visibilidad de las características (atributos y operaciones) de las clases. A partir de esto, se presenta la traslación hacia el modelo relacional.

2.3.1 Modelo relacional

NOMBRE DE LA TABLA: EMPLEADO (CATALOGO-ADMINISTRABLE)

NOMBRE DE LA COLUMNA	TIPO DE LLAVE	NULO	DATOS DE EJEMPLO	OBSERVACIONES
#(1)idEmpleado	PK	NN	1,2,3,4	ENTERO AUTOINCREMENT (4)
nombre		NN	Juan, Pedro, Armando, Ricardo	CADENA (30)
aPaterno		NN	Guzmán, Pérez, Rosas, Martínez	CADENA (30)
aMaterno		NN	Pérez, Rosas, Martínez, Victoria	CADENA (30)
Rfc			GUAM750724DSA SASL77061DX1 EAFG580912JS	CADENA (15)
curp			GUAM750724HPLNSS0 SALS770611HPRSKIT54 EAFG580912HOTGDF87 SAVR671105HOIEGW98	CADENA (18)
#(2)usuario	UK2	NN	jguzman, pperez, aarmenta, rsanchez	CADENA (15)
password		NN	8765kikd, 9816kisa, lod651so, fadkjka2	CADENA (16) ENCRIPTADA
idArea	FK	NN	1,2,3,4,5	ENTERO (2) Relación con la tabla AREA
activo		NN	0,1	BIT 0=No activo 1=Activo
idTipoUsuario	FK	NN	Administrador, etc	CADENA (25) Relación con la tabla PERFIL
dependeDe			1,2,3,4	Relación recursiva

Tabla 2.1 Modelo tabular de la entidad EMPLEADO

En la Tabla 2.1 se almacenará la información de los empleados que laboran en la empresa, permitiendo guardar la información como el departamento al que pertenece, quién es su jefe directo en el área, su perfil de acceso al sistema y un histórico de las ordenes que ha realizado, rechazado y en las que se ha visto involucrado como encargado de llevar a cabo la ejecución de la orden de trabajo.

El atributo activo permite habilitar o deshabilitar un empleado cuando sea necesario cuando las reglas de integridad puedan ser corrompidas, permitiendo guardar un histórico de que empleado atendió una orden de servicio aún cuando éste ya no se encuentre laborando en la empresa.

Desglose de tablas relacionadas:
AREA
PERFIL
ASIGANACIONORDEN

NOMBRE DE LA TABLA: AREA (CATALOGO-ADMINISTRABLE)

NOMBRE DE LA COLUMNA	TIPO DE LLAVE	NULO	DATOS DE EJEMPLO	OBSERVACIONES
idArea	PK	NN	1,2,3,4	ENTERO AUTOINCREMENT(4)
Nombre		NN	Administración de sistemas Desarrollo de sistemas Soporte y redes Mantenimiento	CADENA(30)

Tabla 2.2 Modelo tabular de la entidad AREA

La Tabla 2.2 es una tabla de catálogo que permite saber cuales son los departamentos que proveen servicios dentro de la empresa o institución, y permite agrupar a empleados y a los tipos de servicios que se brindan en la empresa bajo el esquema de órdenes de servicio.

Desglose de tablas relacionadas:
EMPLEADO
TIPOSERVICIO

NOMBRE DE LA TABLA: CLIENTE (CATALOGO-ADMINISTRABLE)

NOMBRE DE LA COLUMNA	TIPO DE LLAVE	NULO	DATOS DE EJEMPLO	OBSERVACIONES
idCliente	PK	NN	20030001,20040004, 20040099, 20040234	ENTERO (4)
nombre		NN	Moisés, Maria Luisa, Héctor, Industrias nacionales	CADENA (100)
aPaterno		NN	Guzmán, Sánchez, Méndez	CADENA (30)
aMaterno		NN	Arias, Sosa, Villa	CADENA (30)
direccion			Calmecac 117-B, Privada 6 Nte. 808, 7 Norte 910,	CADENA (80)
colonia			Fracc.El Pilar, Guadalupe Caleras, Centro	CADENA (50)
ciudad			San Andrés Cholula, Puebla, Tlaxcala	CADENA (40)
estado			Puebla, Oaxaca, Nayarit Puebla	CADENA (40)
lada			222, 189, 155	CADENA (3)
telefono		NN	8762732, 9876781, 9865241	CADENA (7)
cp			72656, 72100,72100	CADENA (5)
email			mguzman@correo.com mluisa@correo.com hmendez@correo.com contacto@correo.com	CADENA (100)
contacto			Juan Pérez Luis García Antonio Gómez	CADENA (90)
usuario	UK1	NN	mguzman mluisa hmendez industrias	CADENA (12)
password		NN	**********	CADENA(16) ENCRIPTADA
idTipoUsuario	FK	NN	clienteAdministrador clienteUsuario clienteAdministrador	CADENA(25) Relación con la tabla PERFIL
activo		NN	0,1	BIT

Tabla 2.3 Modelo tabular de la entidad CLIENTE

La tabla 2.3 representa una entidad que permite almacenar la información de los clientes que tienen acceso y almacena dirección, ciudad, teléfono, etc.; se puede ver como una entidad de catálogo para el manejo de las órdenes de servicio.

Desglose de tablas relacionadas:
PERFIL
ORDENSERVICIO

NOMBRE DE LA TABLA: PERFIL (CATALOGO)

NOMBRE DE LA COLUMNA	TIPO DE LLAVE	NULO	DATOS DE EJEMPLO	OBSERVACIONES
tipoUsuario	PK	NN	Administrador, tecnico cliente, etc.	CADENA(25)
crear		NN	1, 0	BIT
modificar		NN	1, 0	BIT
visualizar		NN	1, 0	BIT
eliminar		NN	1, 0	BIT
concluir		NN	1, 0	BIT
aceptarConclusion		NN	1, 0	BIT
mensaje		NN	1, 0	BIT
seguimientoAgregar		NN	1, 0	BIT
seguimientoVisualizar		NN	1, 0	BIT
asignación		NN	1, 0	BIT
aceptación		NN	1, 0	BIT
modificarAceptacion		NN	1, 0	BIT
clientegregar		NN	1, 0	BIT
clienteModificar		NN	1, 0	BIT
clienteEliminar		NN	1, 0	BIT
empleados		NN	1, 0	BIT
tipoServicio		NN	1, 0	BIT
areas		NN	1, 0	BIT

Tabla 2.4 Modelo tabular de la entidad PERFIL

La Tabla 2.4 se encarga de guardar los perfiles de tipos de usuarios que tienen acceso al sistema, guardando un registro de a que opciones tiene acceso cada uno y a que opciones no, esto se almacena en una matriz en donde la mayor parte de los atributos son de tipo BIT y deberá ser consultada para saber que opciones se habilitan y cuales no al los usuarios. Los usuarios pueden ser clientes y empleados.

Desglose de tablas relacionadas:
CLIENTE
EMPLEADO

NOMBRE DE LA TABLA: ORDENSERVICIO (ADMINISTRABLE-PROCESO)

NOMBRE DE LA COLUMNA	TIPO DE LLAVE	NULO	DATOS DE EJEMPLO	OBSERVACIONES
numOrden	PK	NN	200400001 200400002 200400030	ENTERO (9)
fechaCreacion		NN	2004/06/11 2004/07/25 2004/08/30	DATE
fechaConclusion			2004/08/07 2004/08/12	DATE
descripcionProblema		NN	El equipo tiene algunos problemas de …. El CPU no entra a la red y la navegación no …. El sistema operativo se traba con frecuencia y…	MEMO
Origen		NN	1,2	Por recepción=1, En linea=2
idCliente			20040001, 20040034, 20040056	ENTERO(8) Relación con la tabla CLIENTE
idTipoServicio			001, 045	ENTERO(3) Relación con la tabla TIPOSERVICIO
Estado			1	1=Terminada

Tabla 2.5 Modelo tabular de la entidad ORDENSERVICIO

En la Tabla 2.5 se guardan las generalidades de una orden de servicio, ligándola con el cliente y el tipo de servicio que se realizará, los pormenores y pasos a seguir durante la realización de la orden de servicio, serán manipuladas y almacenadas en la tabla SEGUIMIENTO (Seguimiento de ordenes) tomando como base los pasos relacionados al tipo de servicio manejado en la tabla TIPO SERVICIO

Desglose de tablas relacionadas:
ASIGNACIONORDEN
TIPOSERVICIO
SEGUIMIENTO
CLIENTE

NOMBRE DE LA TABLA: SEGUIMIENTO (ADMINISTRABLE-PROCESO)

NOMBRE DE LA COLUMNA	TIPO DE LLAVE	NULO	DATOS DE EJEMPLO	OBSERVACIONES
respuesta		NN	Se encontró dañado un elemento de la… El cable de la tarjeta esta dañado… El equipo se embarco y envia al cliente	MEMO Se escribe una descripción de la tarea realizada y se Informa del avance en la tarea.
fechaInicio			2004/07/09, 2004/07/15, 2004/06/20	DATE
fechaConclusion			2004/08/09, 2004/08/11, 2004/08/26	DATE
estado		NN	0, 1, 2	ENTERO(2) 0=Pendiente 1=En proceso 2=Concluido
observaciones			En espera de materiales Se sustituyo la tarjeta de red Se procede a cierre de orden Se procede a cierre de orden	MEMO Permite poner observaciones adicionales al paso Realizado, mensajes personales al encargado.
idPaso	PK	NN	2,3,4,5,6	ENTERO(2) Relación con la tabla PASO
numOrden	PK	NN	200400030, 200400025, 200600130	ENTERO(9)

Tabla 2.6 Modelo tabular de la entidad SEGUIMIENTO

La Tabla 2.6 que almacena en si cada uno de los pasos de la orden de trabajo, permitiendo al los usuarios agregar información a cada paso, esto por lo general a los técnicos, aunque también estará permitido al clienteAdministrador y al administrador del sistema, una vez que un texto sea agregado a un paso, este no podrá modificarse, salvo por intervención del administrador, la llave primaria está definida por la conjunción de idPaso y idTipoServicio que es a su son llaves foráneas que hace referencia a la tabla PASO y también se referencia a la tabla ORDENSERVICIO por medio del atributo numOrden que en conjunto forman una llave única para cada paso perteneciente a una orden de servicio.

Desglose de tablas relacionadas:
PASO
ORDENSERVICIO

NOMBRE DE LA TABLA: PASO (CATALOGO-ADMINISTRABLE)

NOMBRE DE LA COLUMNA	TIPO DE LLAVE	NULO	DATOS DE EJEMPLO	OBSERVACIONES
idTipoServicio	PK	NN	002, 125, 456	ENTERO(3) Relación con la tabla TIPOSERVICIO
idPaso	PK	NN	1,2,3,4,5	ENTERO(2) Relación con la tabla PASO
Nombre		NN	Generación de orden de servicio Recepción de equipo y diagnóstico Reparación	CADENA(100)
activo		NN	0,1	BIT

Tabla 2.7 Modelo tabular de la entidad PASO

En la Tabla 2.7 se almacena cada paso que se desee destacar por cada servicio que se tenga disponible y sobre el cual se generará el seguimiento de los servicios respetando el peso que determina idPaso y hace uso de idTipoServicio para en conjunto generar la llave primaria de la entidad, así se permite que cada tipo de servicio tenga definida la secuencia de pasos para sí mismo sin interferir con otros. Es conveniente hacer esto ya que de otra manera se podría haber metido los pasos en la entidad TIPOSERVICIO, pero el problema es que no todos los servicios tienen el mimo número de pasos ni se definen de la misma manera.

Desglose de tablas relacionadas:
SEGUIMIENTO
TIPOSERVICIO

NOMBRE DE LA TABLA: TIPOSERVICIO (CATALOGO-ADMINISTRABLE)

NOMBRE DE LA COLUMNA	TIPO DE LLAVE	NULO	DATOS DE EJEMPLO	OBSERVACIONES
IdTipoServicio	PK	NN	001, 002, 003, 004	ENTERO(3)
IdArea		NN	01, 02, 03	ENTERO(2) Relación con la tabla AREA
Nombre		NN	Desarrollo de sistema Soporte redes Problemas con sistema operativo	CADENA(100)

Fallas con impresoras

Tabla 2.8 Modelo tabular de la entidad TIPOSERVICIO

La Tabla 2.8 maneja el catálogo de los servicios disponibles por la empresa para los usuarios, maneja de 1 a 999 servicios diferentes, permite saber a que área pertenece cada servicio, es decir, en que área se le dará atención y seguimiento a la orden.

Desglose de tablas relacionadas:
AREA
PASO
ORDENSERVICIO

NOMBRE DE LA TABLA: ESTADO (CATALOGO-ADMINISTRABLE)

NOMBRE DE LA COLUMNA	TIPO DE LLAVE	NULO	DATOS DE EJEMPLO	OBSERVACIONES
idEstado		NN	1,2,3,4…	ENTERO(2)
Nombre		NN	Puebla, Oaxaca, Yucatán…	CADENA(40)

Tabla 2.9 Modelo tabular de la entidad AREA

La Tabla 2.9 maneja el catálogo de las áreas que prestan servicios en la empresa y permite clasificar o saber que servicios son proporcionados por cada área al relacionarse con la tabla TIPOSERVICIO y además se relaciona con la tabla empleado que permite conocer cuales son los empleados de la empresa que pertenecen a las áreas para realizar la asignación de las ordenes de servicio.

Desglose de tablas relacionadas:
TIPOSERVICIO
EMPLEADO

NOMBRE DE LA TABLA:ASIGNACIONORDEN (PROCESO)

NOMBRE DE LA COLUMNA	TIPO DE LLAVE	NULO	DATOS DE EJEMPLO	OBSERVACIONES
IdEmpleado	PK	NN	1, 2, 3, 4	ENTERO (4)
IdOrdenServicio	PK	NN	200400001, 200400001, 200400034	ENTERO (9)
FechaAsginación		NN	2004/08/09, 2004/08/09, 2004/08/10	DATE
EstadoAceptacion		NN	0,1, 2	ENTERO (1)

Tabla 2.10 Modelo tabular de la entidad ASIGNACIONORDEN

La Tabla 2.10 se encarga de guardar un registro de asignación de órdenes de servicio, se debe tomar en cuenta que las ordenes de servicio manejan estado de aceptación por la persona a la que se le asigna el trabajo (0 en espera de respuesta, 1 aceptado, 2 no aceptado), la llave primaria la generan en conjunto el idEmpleado y el idOrdenServicio que permiten identificar de manera única a cada elemento de la tupla, en donde ambos componentes de la llave primaria son llaves foráneas y la tabla en si es el resultado de una relación muchos a muchos.

Desglose de tablas relacionadas:
ORDENSERVICIO
EMPLEADO

NOMBRE DE LA TABLA:MENSAJE (PROCESO)

NOMBRE DE LA COLUMNA	TIPO DE LLAVE	NULO	DATOS DE EJEMPLO	OBSERVACIONES
numMensaje	PK	NN	20040001, 20040002,…	ENTERO (4)
numOrden	PK	NN	200400001, 200400001, 200400034	ENTERO (9)
Mensaje		NN	Por favor revisar completamente el equipo…	TEXT
from		NN	mguzman, fperez, msanchez	CADENA(12)
para		NN	mguzman, fperez, msanchez	CADENA(12)
idTipoMensaje	FK	NN	1,2,3,4,5…	ENTERO (2)

Tabla 2.11 Modelo tabular de la entidad MENSAJE

En la Tabla 2.11 se almacenarán todos los mensajes relacionados a una orden de servicio, en donde cada mensaje será identificado por el numero de orden al que pertenece ya a un número consecutivo que permitirá identificar el orden que tienen los mensajes dentro de la orden de servicio siempre guardando quien envió el mensaje en el atributo from.

Desglose de tablas relacionadas:
ORDENSERVICIO
TIPOMENSAJE

NOMBRE DE LA TABLA:TIPOMENSAJE (CATALOGO)

NOMBRE DE LA COLUMNA	TIPO DE LLAVE	NULO	DATOS DE EJEMPLO	OBSERVACIONES
idTipoMensaje	PK	NN	1, 2, 3, 4…	ENTERO (2)
Nombre		NN	Reclamo, Aviso, Alerta, Mensaje simple	VARCHAR (9)
Activo		NN	0,1	BIT

Tabla 2.12 Modelo tabular de la entidad TIPOMENSAJE

Mediante entidad mostrada en la tabla 2.12 se pueden clasificar los mensajes y priorizar o darles un peso administrativo dentro de la orden de servicio para los usuarios del sistema.

Desglose de tablas relacionadas:
MENSAJE

2.3.2 Diagrama general de clases

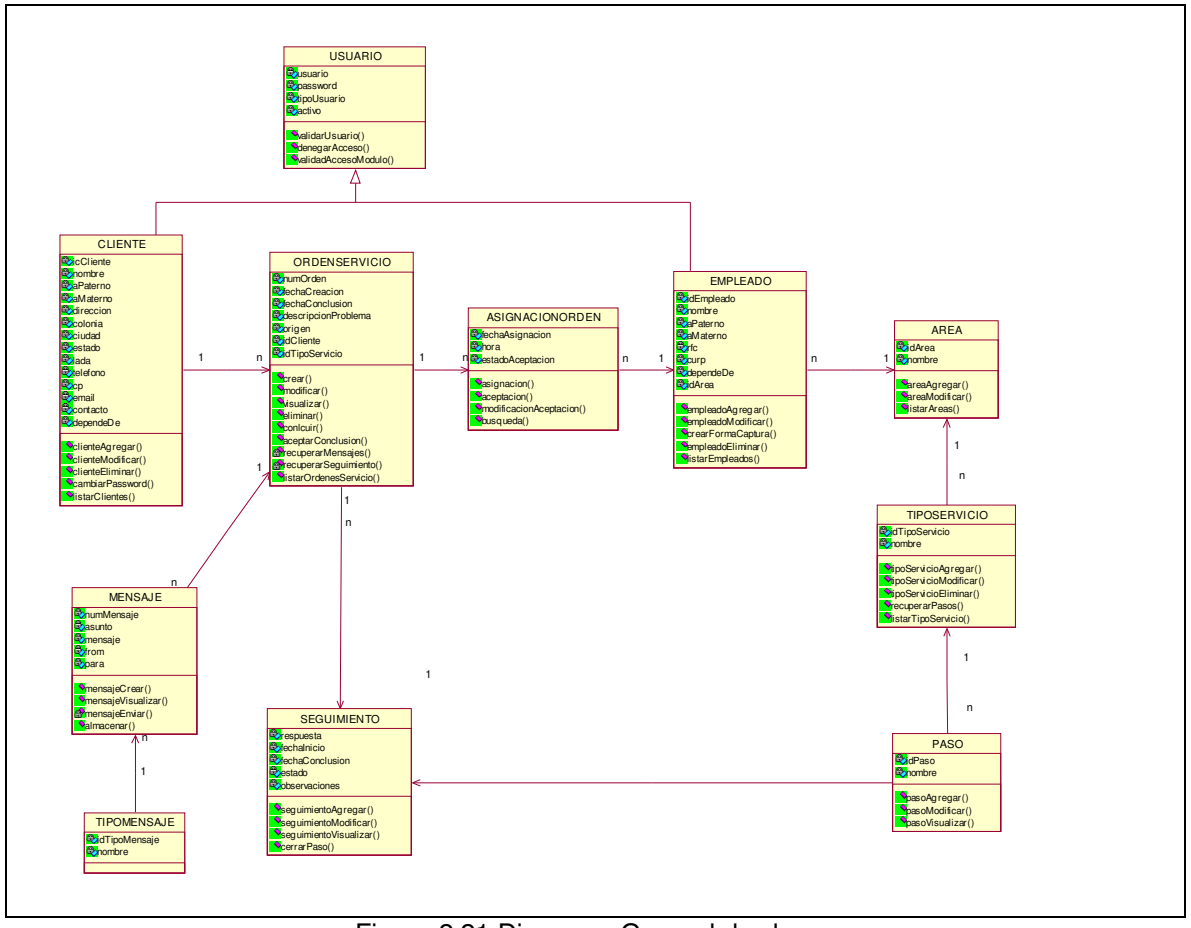

Figura 2.21 Diagrama General de clases

En la Figura 2.2 se muestra el Diagrama General de Clases de la aplicación. El diagrama contiene el modelo del dominio del sistema. Muestra las clases y la forma en la cual se relaciona una clase con otras. Por ejemplo muestra que un cliente pueden tener una o más órdenes de servicio asociadas, una orden de servicio puede tener uno o más mensajes relacionados, que existen una o más entradas de seguimiento para una orden de servicio.

2.3.3 Diagramas de secuencia

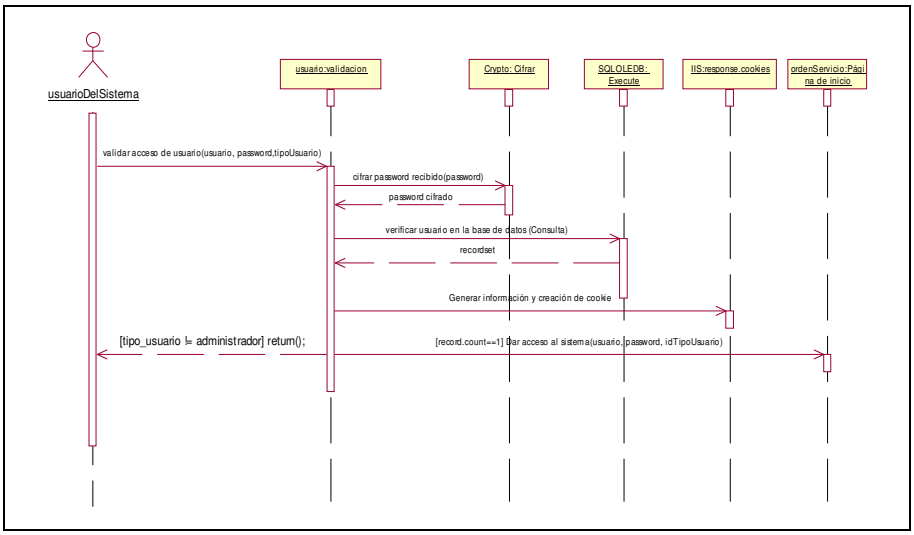

Figura 2.22 Diagrama de secuencia

En la Figura 2.22 se muestra el diagrama de secuencia que permite mostrar los diferentes escenarios que se pueden presentar durante la validación de los usuarios.

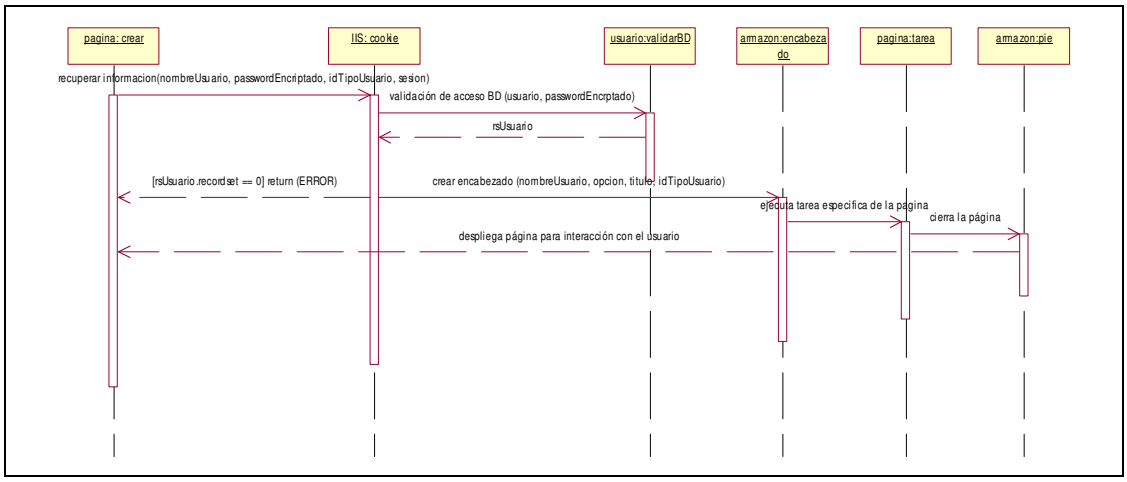

Figura 2.23 Diagrama de secuencia

Ya que se adopta un esquema gráfico uniforme para desplegar la información en la interfaz del sistema, se debe desarrollar un módulo que realice esta tarea, para ello se maneja un esquema de vestimentas. La Figura 2.23 Muestra la secuencia de generación de vestimentas dinámicas. Es muy importante hacer notar que en el diagrama de secuencia se involucra la validación del usuario, esto se hace con la finalidad de que se mantenga la seguridad de acceso al sistema en todo momento y cada ocasión que se invoca un nuevo script de ASP.

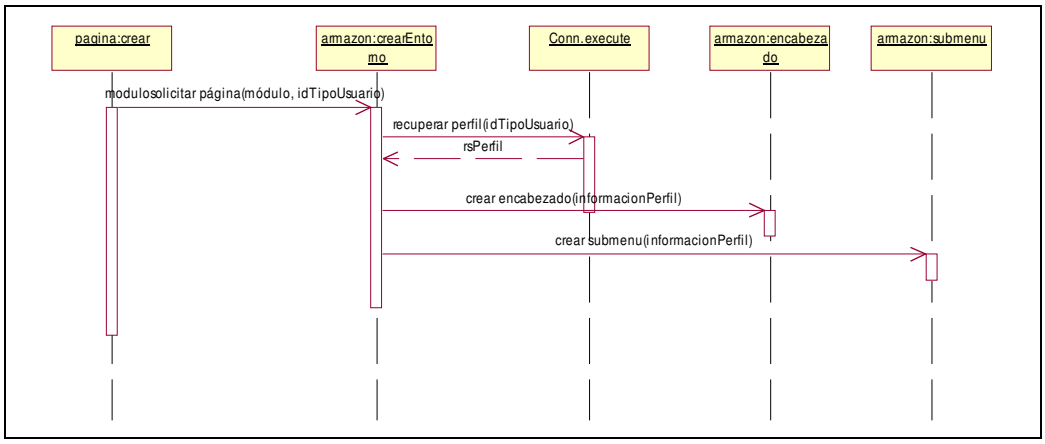

Figura 2.24 Diagrama de secuencia

Se debe mantener en mente que el sistema además de validar el acceso de los usuarios, también mantiene un rol de estos para el acceso y manipulación de la información. En la Figura 2.24 se muestra el diagrama de secuencia para la creación del encabezado (Donde se localiza el menú principal) y el submenú para cada script que se invoca y de esta forma asegurar que el usuario solo accede a la información que se le ha autorizado.

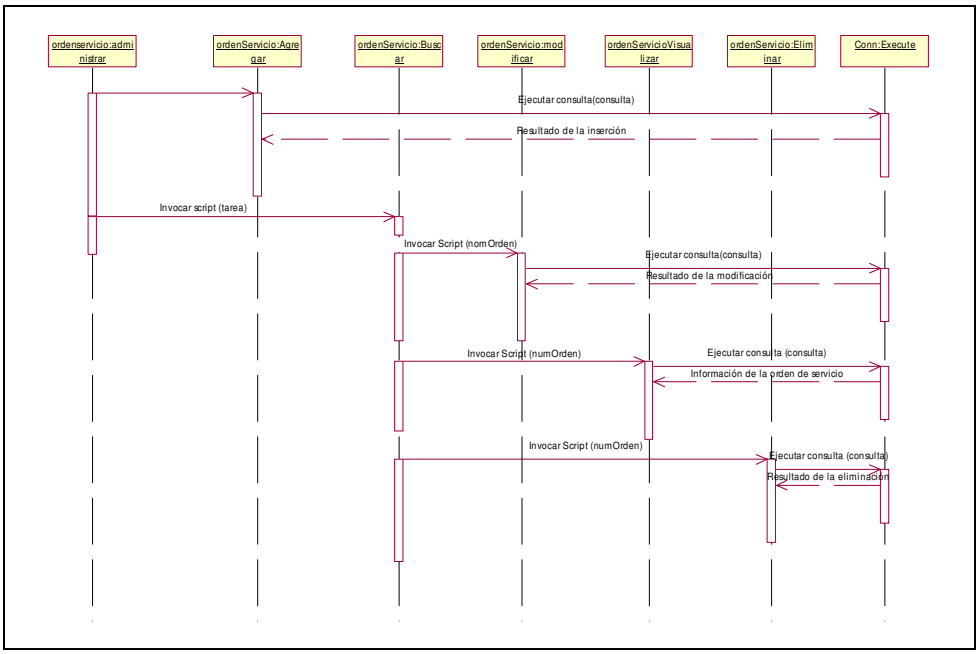

Figura 2.25 Diagrama de secuencia

El manejo de órdenes de servicio que es la parte esencial del sistema se asocia al diagrama de secuencia mostrado en la Figura 2.25 en donde se destacan las operaciones principales de una clase (Agregar, Modificar, Eliminar y Visualizar) cuando se accede a ellas para administrarlas.

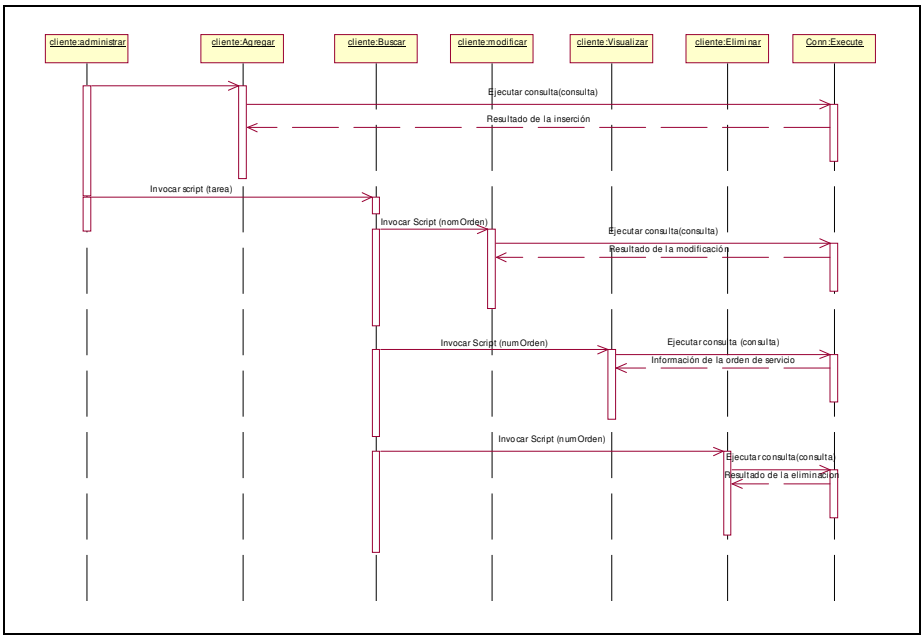

Figura 2.26 Diagrama de secuencia

De manera similar al manejo de las órdenes de servicio, los clientes son parte importante del sistema, el diagrama de secuencia de la administración de esta clase se muestra en la Figura 2.26.

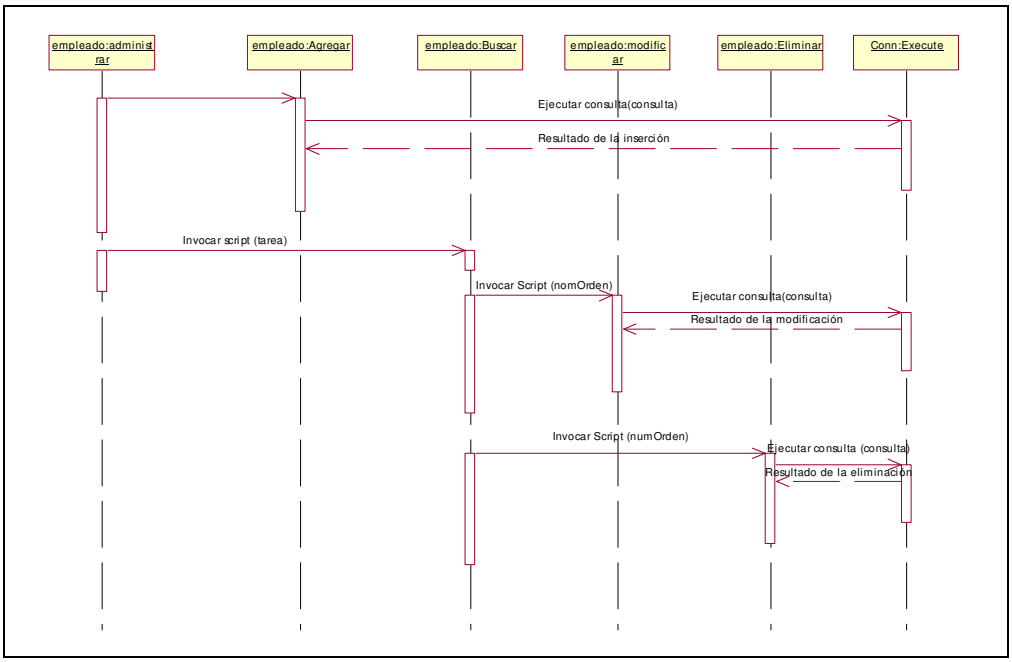

Figura 2.27 Diagrama de secuencia

Los empleados de la empresa prestadora del servicio son actores del sistema. Es importante que se cuente con un mecanismo de administración que permita acceder información

acerca de ellos, modificarla y asignarles roles dentro del sistema, la Figura 2.27 muestra el diagrama de secuencia que describe el comportamiento de este caso de uso.

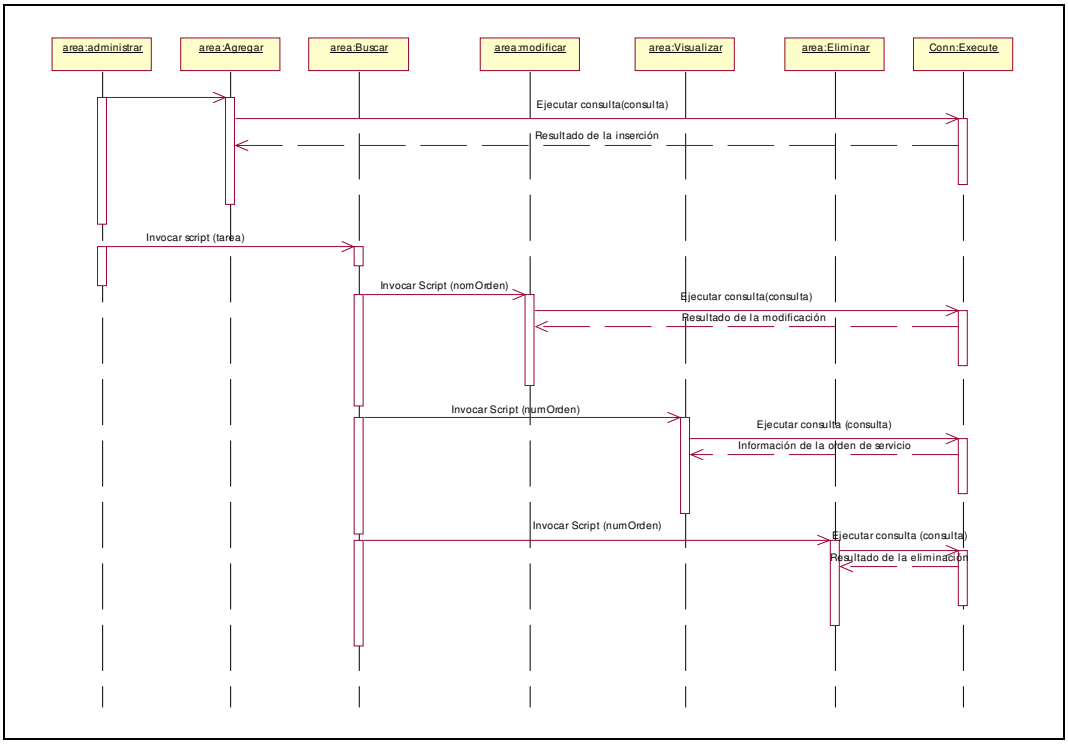

Figura 2.28 Diagrama de secuencia

Una empresa puede disponer de una o más áreas en las que se presten diferentes tipos de servicios, la Figura 2.28 muestra el diagrama de secuencia que describe el escenario de las transacciones que se realizan con esta clase.

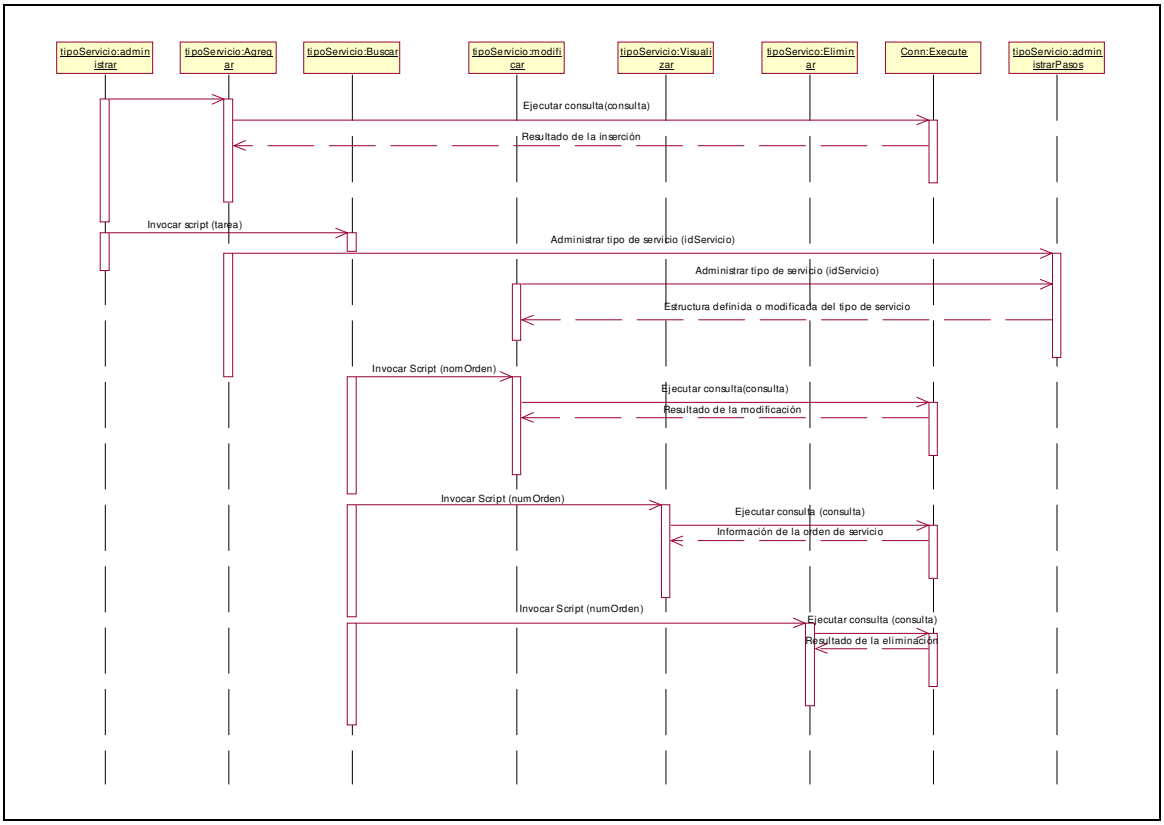

Figura 2.29 Diagrama de secuencia

Se ha mencionado que una empresa dispone de diversas áreas, pero también se da el caso que pueden existir uno o más tipos de servicio. La Figura 2.29 muestra el diagrama de secuencia que describe el escenario de las transacciones que se realizan con esta clase.

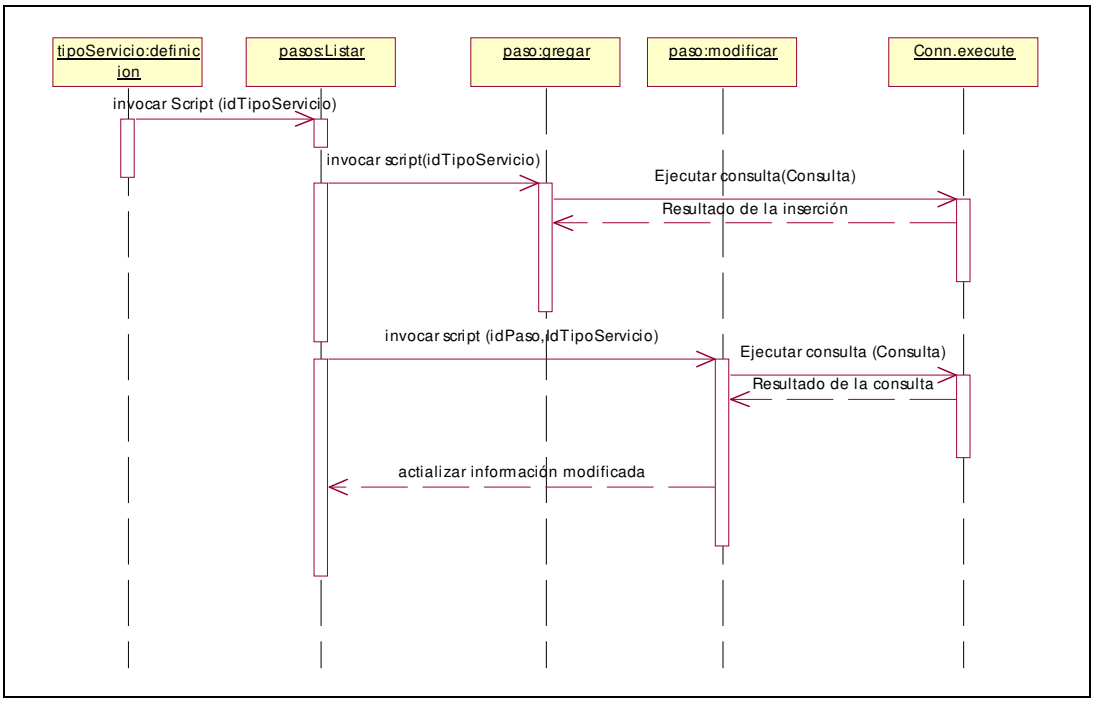

Figura 2.30 Diagrama de secuencia

La idea de realizar un sistema como el que se describe en este trabajo de tesis es dar seguimiento a un tipo específico de servicio, el cual está asociado a un área dentro de una empresa. Para dar seguimiento a un servicio, se precisa describir cuales son los pasos que conforman un servicio, la Figura 2.30 muestra el diagrama de secuencia que describe el escenario de las transacciones que se realizan con esta clase..

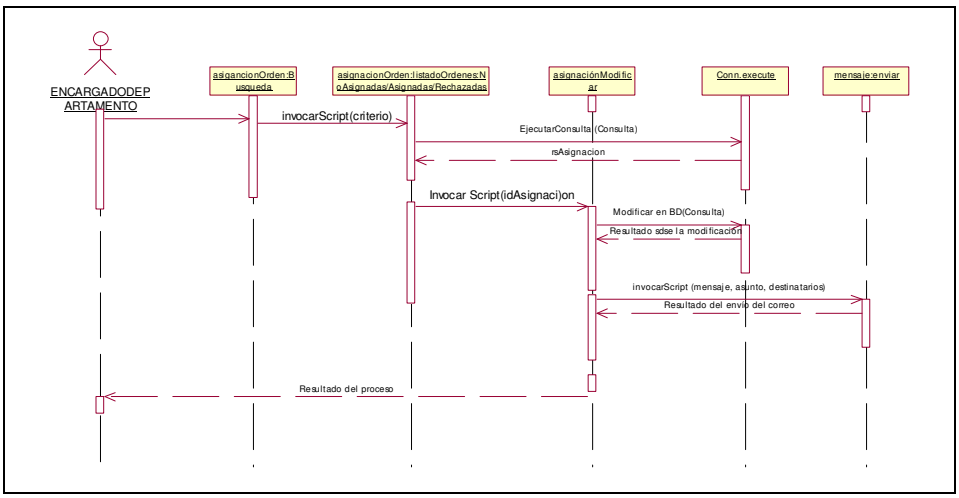

Figura 2.31 Diagrama de secuencia

Para la asignación de órdenes de servicio por el encargado de departamento del área a la que pertenece el servicio se definen diversas tareas. Para tal efecto en la Figura 2.31 se muestran los diferentes escenarios que se dan en dicha tarea.

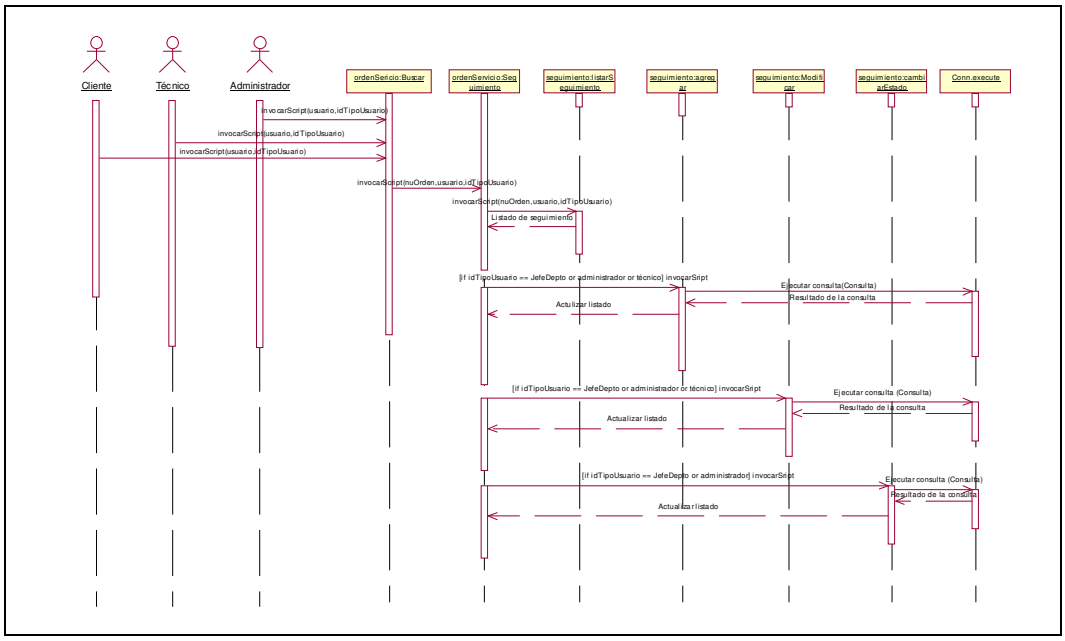

Figura 2.32 Diagrama de secuencia

La figura2.32 muestra los diferentes escenarios que se dan cuando los diferentes actores del sistema dan seguimiento a una orden de servicio dentro del sistema.

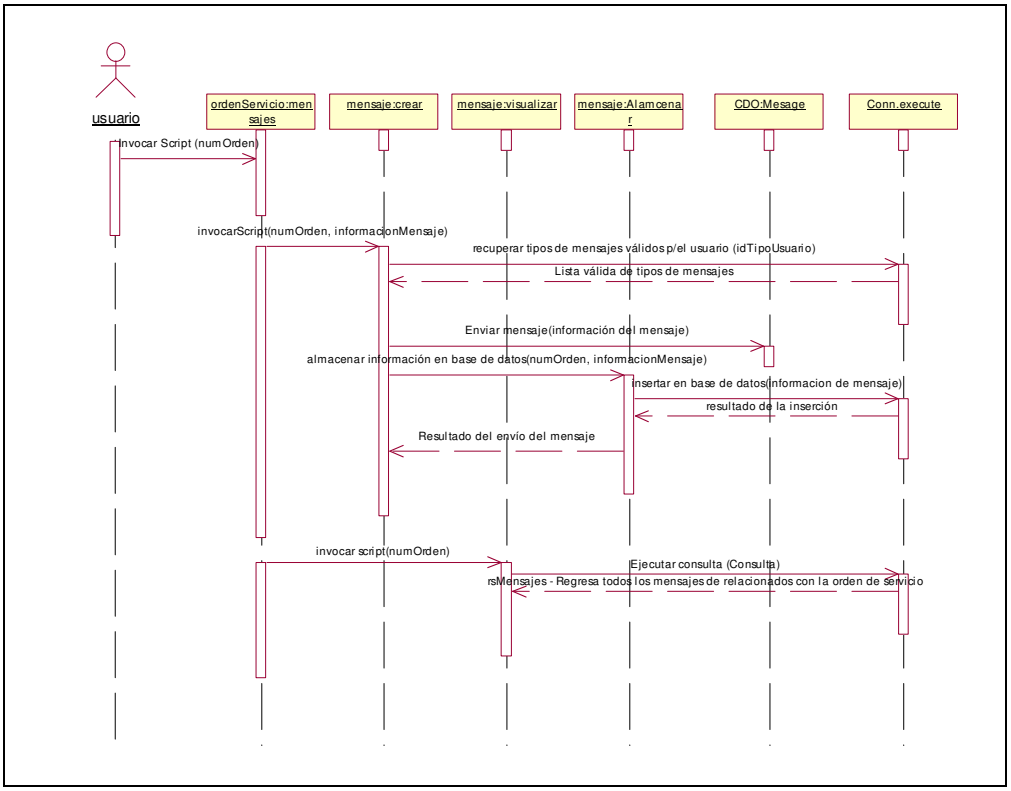

Figura 2.33 Diagrama de secuencia

Durante manejo la ejecución de una orden de servicio se generan diferentes mensajes de aviso de, hacia y entre los actores que se involucran en la ejecución. La figura 2.33 muestra los escenarios que se presentan para la clase usuario.

2.3.4 Diagrama funcional

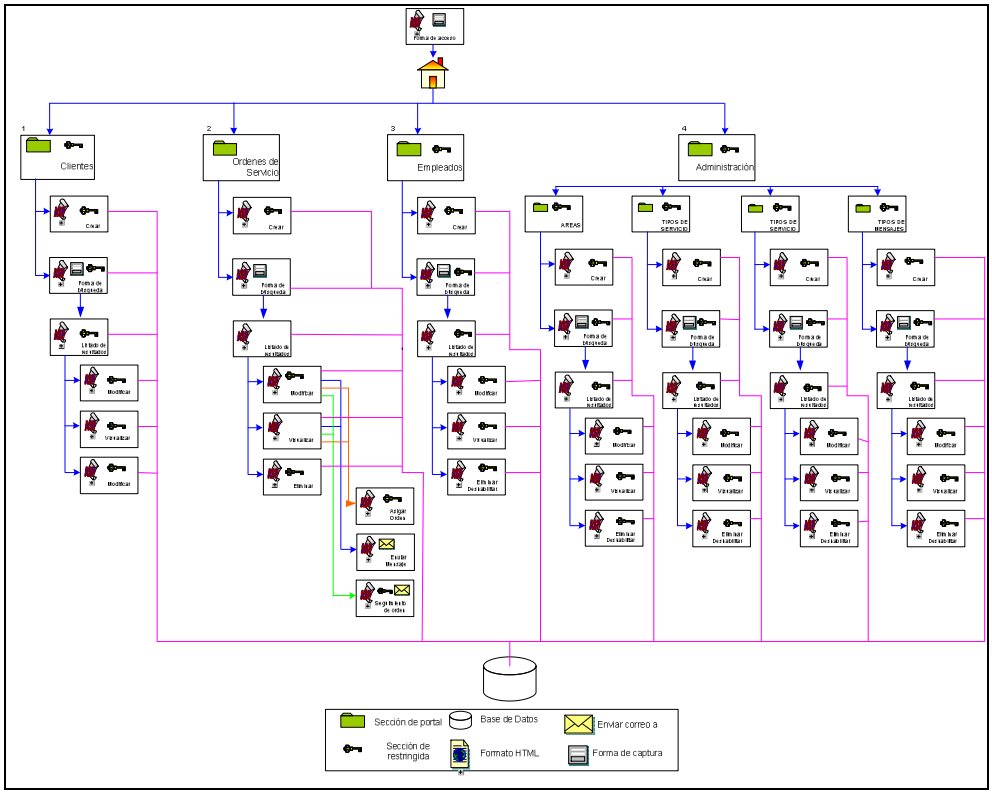

Figura 2.31 diagrama funcional del sistema

En la Figura 2.31 se muestra la manera en como el usuario navegará dentro del sistema para poder realizar la tareas que requiera y a las cuales tenga permiso de acceso de acuerdo a su perfil.

2.3.5 Diseño de interfaces

La definición de las interfaces se debe hacer apegado a normas de usabilidad con la finalidad de crear una aplicación que haga sentir cómodo al usuario dentro de ella. Tomando como base lo anterior, en las normas de usabilidad se indica que los esquemas o interfaces deben mantener la estructura de trabajo, menús, contenidos, etc. y solo la página principal puede variar del resto de las páginas internas de la aplicación. Esta ocasión haremos uso de éste recurso pues la página principal no muestra información en especial que permita generar contenidos relevantes para los usuarios, ésta página deberá de mostrar la forma de acceso al sistema preguntando el nombre de usuario, password(clave de acceso) y tipo de usuario.

La estructura interior se definirá de acuerdo a la Figura 2.32, en la quese define el área para información de acceso, tareas secundarias de navegación, opciones especiales, menú principal y submenús.

Se debe tener especial cuidado de definir mediante programación un esquema que permita mantener y administrar de manera eficiente la interfaz completa del sistema de tal suerte que todos los elementos se encuentren normalizados en estilo y definición.

El esquema debe ser claro para el usuario, indicarle mediante el seguimiento de posición y tarea seleccionada o tarea en ejecución, mantener la posición de las formas, etiquetas y componentes como botones, cajas de texto, cajas de selección, areas de texto y demás elementos.

El esquema de definición de la interfaz interna deberá ser parametrizado para poder desplegar variantes de la información manteniendo el la estructura completa para evitar confusiones al usuario. El área de contenido será la que varíe dentro del esquema interno y será donde se desplieguen los componentes necesarios para interactuar con la aplicación y el resultado de la gestión de la información.

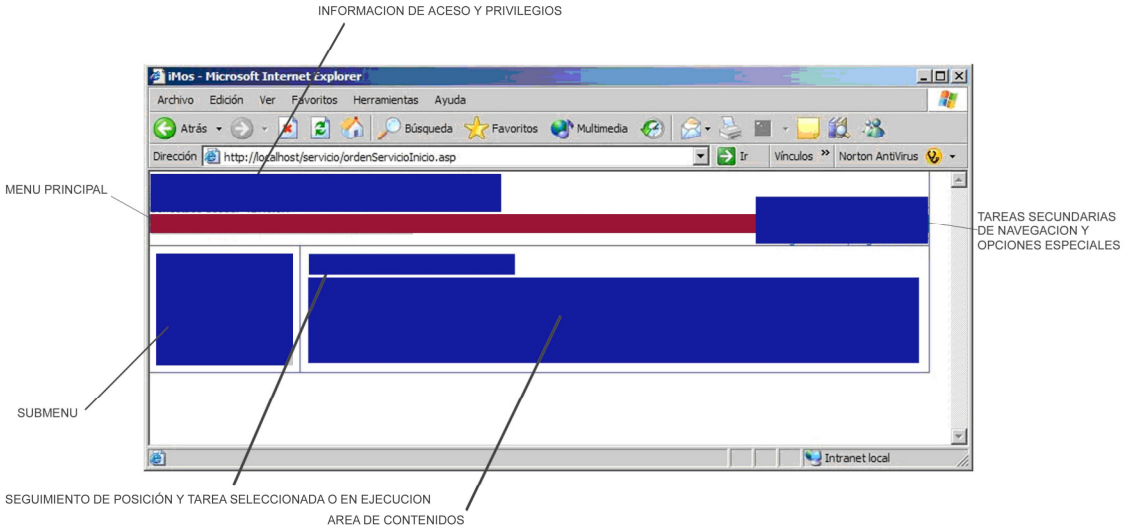

Figura 2.32 Esquema que deberá utilizarse en la definición de las interfáces internas de la aplicación.

3 Implementación

3.1 Instalación y puesta a punto de recursos.

El sistema objeto de éste documento de tesis se ha trabajado y modelado como un sistema de gestión de contenidos, a lo que se define como el conjunto de tareas capaces de adquirir, manipular y entregar información de tal manera que ésta sea útil para un fin específico. Cuando se implementa un sistema para gestionar información, se deben adoptar tecnologías que permitan almacenar y manipular dicha información a partir de una estructura debidamente definida por un arquitecto de información, para tal efecto, sistema se adoptó una plataforma de desarrollo basada en las siguientes herramientas:

- IIS 7.0 como servidor de de páginas WEB.
- Servicio SMTP con uso de CDOSYS para envío de correo electrónico.
- MS SQL Server Express 2008 como sistema manejador de bases de datos (DBMS).

3.1.1 Instalación de servidor WEB

La computadora que se utilizó, cuenta con Windows Vista Home Premium, sistema operativo que incluye el servidor de WEB llamado IIS 7 (Internet Information Server), pero cuando se instala el sistema operativo no lo instala de manera predeterminada y se debe agregar manualmente. Para llevar a cabo la instalación del servidor, se debe realizar el siguiente procedimiento:

1. Abrir el panel de control de Windows Vista.
2. Entrar en “Agregar y quitar programas”
3. En éste punto se elige “Agregar componentes de Windows”
4. Habilitar “Servicios de Internet Information Server (IIS), debiendo verificar que en los detalles de instalación quede habilitada la casilla de “Servicio de SMTP”.

Cuando esto se ha hecho, el servicio se encuentra instalado en el sistema, pero esto no es suficiente pues resulta necesario que el servicio se habilite de manera automática para que se inicie cada ocasión que el sistema se reinicie. Para validar que esto ha quedado debidamente configurado, se debe llevar a cabo el siguiente procedimiento.

1. Abrir el panel de control de Windows Vista
2. Entrar en “Herramientas administrativas”
3. Elegir la opción “Servicios”
4. Verificar que se encuentre presente el servicio de “Publicación World Wide WEB” y que éste tenga habilitado el inicio automático.
5. Ver que el servicio “Administración de IIS” esté presente y con tipo de inicio automático.
6. Por último se abre un navegador local escribiendo “localhost” en la dirección, inmediatamente se verá la página de inicio de IIS.

3.1.2 Instalación del manejador de bases de datos (MS SQL Server Express 2008)

Ya que se tiene el servicio de WEB disponible, el directorio por defecto donde se localizan las aplicaciones, es “C:\Inetpub\wwwroot”, hecho esto, se realiza la instalación del manejador de la base de datos.

1. Iniciar la instalación de MSSQL Server 2008 desde el siguiente enlace http://www.microsoft.com/express/sql/download/ en donde se muestran las diversas modalidades de instalación, la opción utilizada para este sistema fue “Runtime with Management Tools”. Es importante hacer notar al lector que la instalación se realiza en línea y que se debe disponer de un enlace a Internet de al menos 1 Mbit y que como requisito se instala inicialmente el componente Web Platform Installer (WPI).
2. Dejar habilitada la opción de instalación “Default” para una nueva instalación.
3. Existen dos diferentes maneras de manejar los servicios de cuentas de usuarios, para éste caso de elige la opción “Use the same account for each service. Auto start SQL Server service ” pues con ésta opción se permite que los servicios de SQL Server inicien automáticamente y se especifica que utilizará una cuenta del dominio para tal efecto (Use a Domain User Account) con nombre de usuario “Administrador”, password “notiene” (Usuario definido en el manejador de dominio) y dominio “LAPMGA” (Nombre de la computadora).
4. Para mejorar la seguridad del sistema, se elige la opción “Mixed Mode (Windows Authentication and SQL Server Authentication)” pues se tienen que definir usuarios dentro del manejador de bases de datos para acceder a él que son independientes a los del sistema, quedando definido el “sa” (System Administrator) como administrador del DBMS con password “2407espa”.
5. Se define el diccionario y para búsquedas y coincidencias de cadenas con el diccionario “Tradicional_Spanish” y las opciones “Case sentive” y “Acent sensitive”.
6. Posteriormente se deben definir las librerias de red necesarias para la conexión al DBMS, se utilizaron para este proyecto las liberias “Named pipes” con \\.\pipe\sql\query y “TCP/IP Sockets” en el puerto 1433.
7. Reiniciar el equipo.
8. Cuando reinicie, el servicio estará funcionando, esto se puede comprobar haciendo uso de la aplicación /inicio/Todos los programas/Microsoft SQL Server 2008/Herramientas de configuración/Administrador de Configuración de SQL Server

3.2 Implementación y puesta a punto de la base de datos.

3.2.1 Creación de la base de datos

La base de datos creada recibió el nombre de “imos” (Intranet para el Manejo de Ordenes de Servicio), definida con un crecimiento automático del 10% del tamaño que ocupe el archivo de almacenamiento y sin restricciones de tamaño para máximo crecimiento, para el LOG de transacciones se tomó el mismo criterio.

Esto generalmente se hace para asegurar que al aumentar la cantidad de información almacenada en la base de datos, el archivo que la contiene, no se sature y la única limitante sea el disco duro, igual aplica para el sistema de LOG, el cual guarda un registro de todas las transacciones hechas a la base de datos.

La definición de la base de datos permite que el manejador de base de datos genere un receptáculo que físicamente albergará a las tablas y los metadatos inherentes a las mismas, así como todos los elementos relacionados tales como disparadores, procedimientos anidados, relaciones, vistas, usuarios, roles, etc.

Una vez que se han puesto a punto los recursos necesarios para implementar el sistema, se deben definir las estructuras que almacenaran la información que se gestione por medio del sistema a desarrollar, dichas estructuras deben definirse de acuerdo al modelo diseñado en el Capitulo 2.

3.2.2 Creación de tipos definidos por el usuario

Por el tipo de DBMS con el que se dispone, permite la creación de tipos de datos definidos por el usuario (User Definied Data Types), con los cuales se asegura la integridad de definición de atributos que comparten el mismo tipo de datos a almacenar.

```
EXEC sp_addtype N'nUser', N'varchar (12)', N'not null'
```

nUser es una variable de 12 caracteres, de longitud variable, con la cual se almacena el nombre de un usuario que tenga acceso al sistema, este tipo se utilizará en las tablas EMPLEADO y CLIENTE.

```
EXEC sp_addtype N'pwdCad', N'varchar (16)', N'not null'
```

pwdCad es un tipo cadena que almacena una cadena encriptada mediante Crypto de 16 caracteres, de longitud variable que almacena el password de un usuario definido.

3.2.3 Definición de tablas

A continuación se presentan los script's necesarios para la creación de las tablas asociadas a cada entidades definidas.

Primero que nada se definen las entidades de catalogo, de las cuales dependen otras entidades para poder existir y con las que se encuentran relacionadas.

TABLA AREA:

```
CREATE TABLE [dbo].[AREA] (
	[idArea] [int] NOT NULL ,
	[nombre] [varchar] (30) COLLATE Traditional_Spanish_CI_AS NOT NULL
) ON [PRIMARY]
```

Esta tabla almacena la información relacionada con las áreas o departamentos que componen la estructura funcional encargada de ejecutar las ordenes de servicio que se generan.

TABLE TIPOMENSAJE

```
CREATE TABLE [dbo].[TIPOMENSAJE] (
	[idTIpoMensaje] [int] NOT NULL ,
	[nombre] [varchar] (9) COLLATE Traditional_Spanish_CI_AS NOT NULL
) ON [PRIMARY]
```

Guarda la información que define los diferentes tipos de mensajes que se pueden enviar entre los usuarios, resultado del seguimiento y ejecución de las órdenes de servicio.

ESTADO

```
CREATE TABLE [dbo].[ESTADO] (
	[idEstado] [int] NOT NULL ,
	[nombre] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NOT NULL
```

```
) ON [PRIMARY]
```

Por ésta entidad se cataloga a que estados pertenece cada cliente.

```
CREATE TABLE [dbo].[PERFIL] (
	[tipoUsuario] [varchar] (25) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[alta] [bit] NULL ,
	[visualizar] [bit] NULL ,
	[modificar] [bit] NULL ,
	[concluir] [bit] NULL ,
	[aceptarCliente] [bit] NULL ,
	[reclamo] [bit] NULL ,
	[mensaje] [bit] NULL ,
	[seguimiento] [bit] NULL ,
	[asignacion] [bit] NULL ,
	[clienteAlta] [bit] NULL ,
	[clienteModificar] [bit] NULL ,
	[clienteEliminar] [bit] NULL ,
	[tipoServicioAlta] [bit] NULL ,
	[tipoServicioVisualizar] [bit] NULL ,
	[tipoServicioModificar] [bit] NULL ,
	[tipoServicioEliminar] [bit] NULL ,
	[empleadoAlta] [bit] NULL ,
	[empleadoVisualizar] [bit] NULL ,
	[empleadoModificar] [bit] NULL ,
	[empleadoEliminar] [bit] NULL
) ON [PRIMARY]
```

Mediante esta tabla se mantendrán los diferentes tipos de usuarios y los privilegios de acceso a cada parte del sistema ya que los usuario tienen áreas restringidas de acceso de acuerdo al perfil definido.

```
CREATE TABLE [dbo].[MENSAJE] (
	[numMensaje] [int] NOT NULL ,
	[numOrden] [int] NOT NULL ,
	[mensaje] [text] COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[de] [nUser] NOT NULL ,
	[para] [nUser] NOT NULL ,
	[idTipoMensaje] [int] NOT NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Se debe entender por mensaje a los correos electrónicos que el sistema genere de manera automática o que los usuarios mismos del sistema envién como resultado de la ejecución o seguimiento de la orden de servicio.

```
CREATE TABLE [dbo].[EMPLEADO] (
	[idEmpleado] [int] NOT NULL ,
	[nombre] [varchar] (30) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[aPaterno] [varchar] (30) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[aMaterno] [varchar] (30) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[rfc] [varchar] (15) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[curp] [varchar] (18) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[usuario] [nUser] NOT NULL ,
	[password] [pwdCad] NOT NULL ,
	[idArea] [int] NOT NULL ,
```

```
	[activo] [bit] NOT NULL ,
	[idTipoUsuario] [varchar] (25) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[dependeDe] [int] NULL
) ON [PRIMARY]
```

Existen dos tipos principales de usuarios del sistema, el primero son los empleados, y el otro son los clientes, para el primer caso se ha definido una tabla capaz de almacenar la información necesaria para identificar a un empleado y asignarle la ejecución de órdenes de servicio, aquí se ha implementado una relación redundante, con la cual se permite al sistema realizar la agrupación o dependencia de algunos empleados a un encargado de área o jefe de departamento, según sea el caso.

En general tomando en cuenta la definición de las tablas EMPLEADO y PERFIL, se tendrán 3 tipos de usuario para empleados:

- Administrador del sistema
- Encargado de área
- Técnico

Resulta importante hacer notar al lector que el Administrador será quién se encargue administrar en su totalidad todo el sistema, el encargado de área dará seguimiento a las diferentes órdenes de servicio que se generen hacia su área y se asegurará que éstas sean debidamente resueltas, el sistema le enviará los avisos correspondientes para tal efecto. Una tarea más del encargado de área, será el asignar las diferentes órdenes de servicio a sus subordinados.

```
CREATE TABLE [dbo].[CLIENTE] (
	[idCliente] [int] NOT NULL ,
	[nombre] [varchar] (100) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[aPaterno] [varchar] (30) COLLATE Traditional_Spanish_CI_AS NULL ,
	[aMaterno] [varchar] (30) COLLATE Traditional_Spanish_CI_AS NULL ,
	[direccion] [varchar] (40) COLLATE Traditional_Spanish_CI_AS NULL ,
	[colonia] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NULL ,
	[ciudad] [varchar] (40) COLLATE Traditional_Spanish_CI_AS NULL ,
	[idEstado] [int] NULL ,
	[lada] [char] (3) COLLATE Traditional_Spanish_CI_AS NULL ,
	[telefono] [char] (7) COLLATE Traditional_Spanish_CI_AS NULL ,
	[cp] [char] (5) COLLATE Traditional_Spanish_CI_AS NULL ,
	[email] [varchar] (100) COLLATE Traditional_Spanish_CI_AS NULL ,
	[contacto] [bit] NULL ,
	[usuario] [nUser] NOT NULL ,
	[password] [pwdCad] NOT NULL ,
	[activo] [bit] NOT NULL ,
	[idTipoUsuario] [varchar] (25) COLLATE Traditional_Spanish_CI_AS NOT NULL
) ON [PRIMARY]
```

La tabla definida almacenará la información de los diferentes clientes con capacidad de levantar órdenes de servicio.

```
CREATE TABLE [dbo].[TIPOSERVICIO] (
	[idTipoServicio] [int] NOT NULL ,
	[nombre] [varchar] (50) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[idArea] [int] NOT NULL
) ON [PRIMARY]
```

Así como se definen áreas de servicio para la empresa, cada área tiene asignados servicios específicos que se brindan a sus clientes, es por ello que se crea la tabla TIPOSERVICIO, en ella se definen los diferentes tipos de servicio que se atenderán en cada área.

```
CREATE TABLE [dbo].[PASO] (
	[idPaso] [int] NOT NULL ,
	[idTipoServicio] [int] NOT NULL ,
	[nombre] [varchar] (100) COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[activo] [bit] NOT NULL
) ON [PRIMARY]
```

Ya que se definen la estructura para almacenar los tipos de servicio que se atienden en cada área, los diferentes tipos de servicio, están compuestos por un conjunto de pasos válidos, mediante los cuales se podrá dar seguimiento a la orden de servicio de manera estructurada y con restricciones de no poder avanzar a un paso determinado si no se han cumplido los requisitos anteriores al paso definido. En la definición se encuentra un atributo llamado “activo” que permite que uno de los pasos del proceso se desactive, pero no se elimine, con esto se permite que una orden de servicio que haya hecho uso del paso definido, pero desactivado, pueda seguir usándola y una nueva orden no la tome en cuenta para su seguimiento.

```
CREATE TABLE [dbo].[ORDENSERVICIO] (
	[numOrden] [int] NOT NULL ,
	[fechaCreacion] [datetime] NOT NULL ,
	[fechaConclusion] [datetime] NULL ,
	[descripcionProblema] [text] COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[origen] [int] NOT NULL ,
	[tipoServicio] [int] NOT NULL ,
	[idCliente] [int] NOT NULL ,
	[estado] [int] NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

La entidad anterior es la que se encarga de guarda la información general de las órdenes de servicio, se debe hacer énfasis que las fechas manejadas en las entidades hasta el momento definidas y que se definirán aún en las siguientes, se definieron del tipo “datetime”, con lo cual se agrega un mejor control histórico de la información del sistema.

```
CREATE TABLE [dbo].[ASIGNACIONORDEN] (
	[idEmpleado] [int] NOT NULL ,
	[numOrden] [int] NOT NULL ,
	[fechaHora] [datetime] NOT NULL ,
	[estadoAceptacion] [int] NOT NULL
) ON [PRIMARY]
```

Como se puede apreciar esta entidad queda definida como una entidad que permite eliminar una relación de muchos a muchos, ya que permite que una orden de servicio se asigne a uno o mas empleados y un empleado puede tener una o mas ordenes de servicio asignadas con control de fecha y hora de asignación y un estado de aceptación de acuerdo al diseño planteado en el Capitulo 2.

```
CREATE TABLE [dbo].[SEGUIMIENTO] (
	[idPaso] [int] NOT NULL ,
	[numOrden] [int] NOT NULL ,
	[respuesta] [text] COLLATE Traditional_Spanish_CI_AS NOT NULL ,
	[fechaInicio] [datetime] NULL ,
	[fechaConclusion] [datetime] NULL ,
	[estado] [int] NOT NULL ,
```

[observaciones] [text] COLLATE Traditional_Spanish_CI_AS NOT NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

Esta tabla queda definida como la encargada de relacionar a una orden de servicio con cada uno de los pasos que la componen de acuerdo al tipo de servicio solicitado con un estado independiente de ejecución por el atributo “estado” y fechas de inicio y conclusión.

3.2.4 Definición relaciones entre tablas

Tal como se menciona en el capítulo 2, el diseño de la estructura de datos se implementó en un manejador de bases de datos relacional, por lo cual varias de las entidades mapeadas a tablas, se encuentran relacionadas mediante llaves foráneas y llaves primarias respectivamente, estableciéndose restricciones en la creación de tuplas para conservar la integridad referencial del modelo generando las siguientes restricciones.

1. Existencia de tupla en la tabla referida cuando se desee crear una nueva tupla en una tabla que hace referencia a la primera para asegurar la integridad de la información. (Checking existing data on creation)
2. Forzar la repliación en una relación entre tuplas. (Enforce relationship for replication).
3. Forzar la existencia de la relación en inserción y actualización de información.

Para lo anterior se utilizó la definición de diagramas que incluye la herramienta denominada “SQL Server Enterprise Manager-Diagrams” y que generó el diagrama que se presenta en la Figura 3.1.

De acuerdo al modelo E-R presentado en el capítulo 2, se han conservado las cardinalidades de las relaciones entre tablas de tal manera que ninguna relación ha quedado definida de muchos a muchos de manera directa, cuando éste caso se presentó, se utilizaron entidades de intersección que permiten un mejor manejo de éstas situaciones.

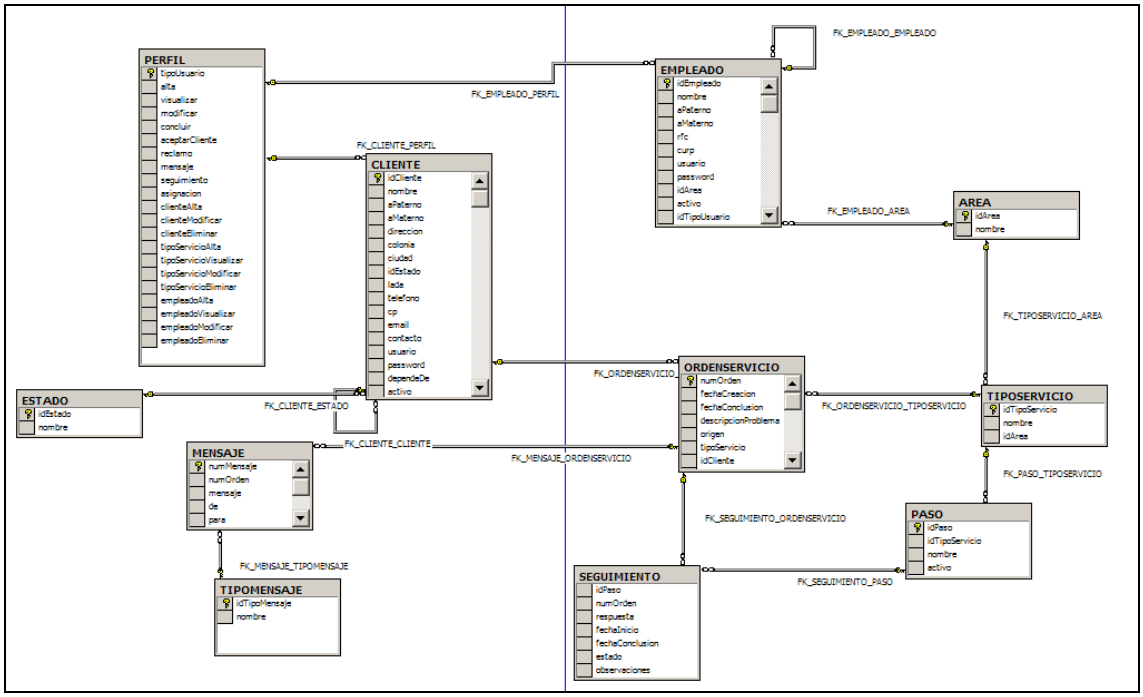

Figura 3.1 Diagrama Entidad-Relación implementado en MSSQL Server 2000

3.2.5 Poblado de tablas de catálogo

Toda vez que se han definido las estructuras de datos que permitirán al sistema poder almacenar información persistente y gestionarla, se deben alimentar las tablas de catálogo y aquellas que aun cuando no son de catálogo directamente y que son administrables, es necesario poblarlas pues otras entidades dependen de ellas para poder existir.

En primer lugar se debe poblar la tabla de áreas (AREA) en la base de datos, para éste caso el sistema se configurará de tal manera que funcione para un centro de soporte técnico de una empresa X.

Poblado de la tabla AREA:

1	Desarrollo de sistemas
2	Soporte técnico
3	Administración de sistemas
4	Redes y telecomunicaciones

Poblado de la tabla TIPOMENSAJE

1	Orden de Servicio: Creación-Notificación
2	Orden de Serviico: Asignación
3	Orden de Servicio: Asignación-Rechazo
4	Orden de Servicio: Asiganción-Aceptación
5	Reclamo
6	Notificación al cliente
7	Aviso
8	Mensaje
9	Orden de Servicio: Conlusión
10	Orden de Servicio: Aceptar conclusión
11	Seguimiento: Reporte
12	Seguimiento: Paso concluido

Poblado de la tabla PERFIL de acuerdo a la tabla

Administrador	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Encargado de area	0 1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 0 0 0
Tecnico	0 0 1 0 0 0 1 1 1 0 1 0 0 0 0 0 0 0 0
Recepcion	1 0 1 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0
Cliente	1 0 1 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0

Poblado de la tabal ESTADO

1	Aguascalientes
2	Baja California
3	Baja California Sur
4	Campeche
5	Chiapas
6	Chihuahua
7	Coahuila
8	Colima
9	Distrito Federal
10	Durango
11	Guanajuato
12	Guerrero
13	Hidalgo
14	Jalisco

15 México
16 Michoacán
17 Morelos
18 Nayarit
19 Nuevo León
20 Oaxaca
21 Puebla
22 Querétaro
23 Quintana Roo
24 San Luis Potosí
25 Sinaloa
26 Sonora
27 Tabasco
28 Tamaulipas
29 Tlaxcala
30 Veracruz
31 Yucatán
32 Zacatecas

Para poblar esta tabla, se hizo uso de la herramienta de importación de datos de MS SQL Server 2000 que permite importar datos desde diferentes formatos y manejadores de bases de datos. La tabla de estados se tenia disponible en formato de texto separado por tabuladores, temporalmente se tuvo que romper la restricción de relación entre CLIENTE y ESTADO para poder hacer la importación de datos y posteriormente se reinstauró para mantener la integridad relacional de la estructura.

Ya creados los catálogos, se deben implementar las rutinas y elementos de programación necesarias para dar estructura a la aplicación a nivel de interfaces.

Para asegurar que una aplicación basada en WEB tenga éxito, se deben asegurar los siguientes elementos de tal manera que el usuario pueda trabajar de manera adecuada con la aplicación desarrollada.

- Infraestructura capaz de soportar el tráfico de información y usuarios en accesos concurrentes.
- Buena velocidad de acceso.
- Disponibilidad.
- Información debidamente estructurada, que soporte modificaciones sin necesidad de alterar código de la aplicación y modificación de la aplicación sin necesidad de alterar la información.
- Facilidad de uso
- Diseño atractivo y funcional.
- Equilibrio entre gráficos e información.
- La información debe mantenerse al día.
- Mantenimiento rápido de la aplicación y la información.

De acuerdo a como las aplicaciones para WEB, llámense “Página WEB”, “WEB Site” y Portales de Internet de baja y alta penetración, así como los sistemas Intranet han evolucionado y tomando en cuenta las observaciones anteriores, se hace necesario establecer rutinas permitan definir una estructura adecuada para crear las interfaces con la que trabajará el usuario. Estas rutinas deben asegurar un mantenimiento eficiente de la aplicación para actualizaciones o expansiones de la misma.

Se dispone de un servidor WEB capas de soportar un lenguaje de programación como ASP para la creación de páginas dinámicas, también en el lado del cliente se tienen elementos como las hojas de estilos y Java Script, con los cuales se abre una gamma impresionante de

posibilidades para poder desarrollar rutinas que permitan automatizar y normalizar diferentes elementos dentro de la aplicación.

A continuación se aborda la manera en que se crea la primera definición de la estructura que deberán mantener las interfaces con cuales trabajen los usuarios.

3.3 Creación del armazón y definición de plantilla de desarrollo

Lo primero que se debe hacer es conocer cual será la apariencia de la aplicación, generalmente se dan dos casos:

1. La página de inicio puede diferir del resto de las páginas internas.
2. Tanto la página de inicio como las interiores son iguales.

En ésta ocasión se eligió que la página de inicio fuera diferente a las internas. Esto se debe a que solo se solicitan los parámetros de acceso al sistema.

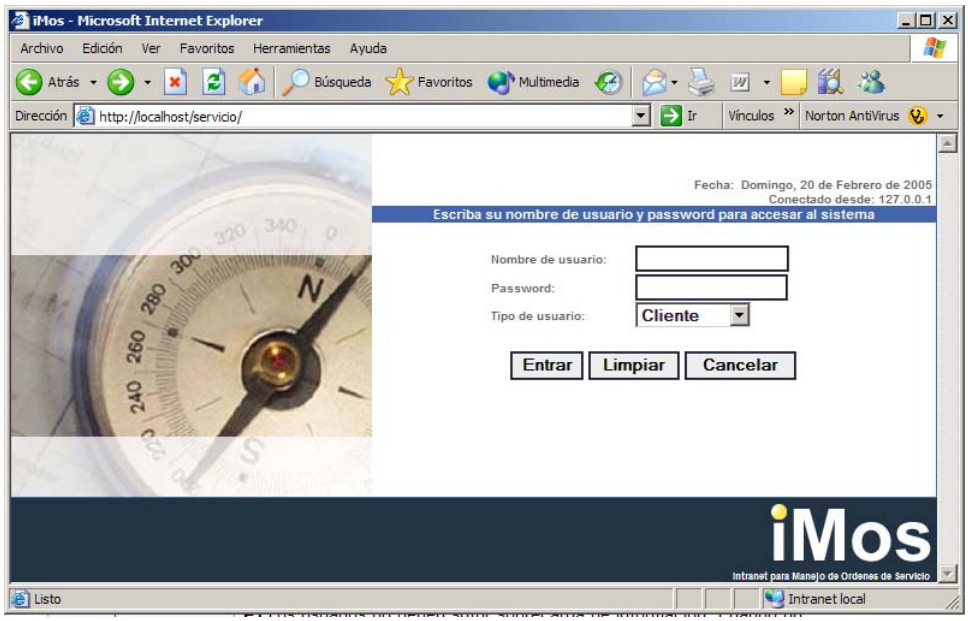

Figura 3.2 Apariencia de la página de inicio de la aplicación.

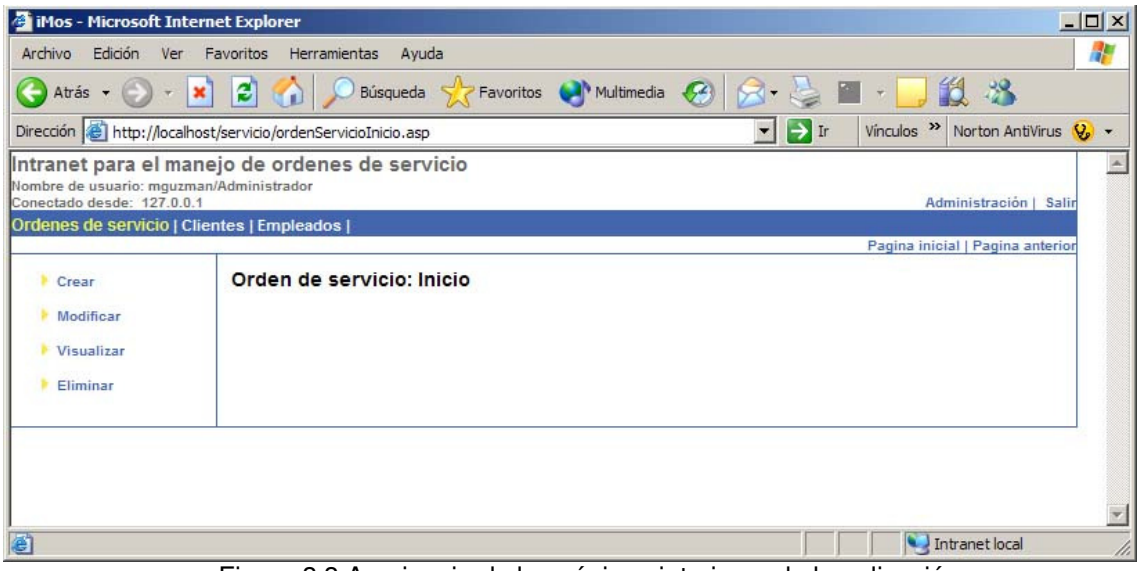

Figura 3.3 Apariencia de las páginas interiores de la aplicación.

Las páginas interiores se han dibujado primero en Photoshop y se diseñaron apegándose a normas de usabilidad e interfaces para Internet, las cuales buscan anticiparse a las necesidades del usuario permitiéndole que perciba al sistema como si el lo controlara dentro de un entorno abarcable y no infinito, normalizado y estandarizado en colores y posición de los elementos de navegación y contenidos.

Se debe además asegurar que la aplicación sea rápida. Se debe recordar que por el tipo de infraestructura con la que se dispone en nuestro país, muchos usuarios no disponen de Internet de Banda Ancha, por lo que una vez que se define la apariencia de una aplicación basada en WEB, se deben elegir cuales elementos a nivel gráfico son susceptibles de optimización por código y cuales no. Un ejemplo de esto, se podría ver haciendonos la siguiente pregunta:

¿Qué tarda más tiempo en viajar por la red, una línea de 720 px x 1px o un elemento de 1px x1px con la finalidad de desplegar una línea?

Desde luego que viajará más rápido un elemento de 1px x 1px, en donde por medio de código se puede lograr que del lado del cliente se dibuje la línea con la ayuda de éste elemento gráfico, y no solo se pueden representar líneas, si no áreas completas con este tipo de recursos.

Una vez que se han identificado los elementos gráficos susceptibles de optimización por código, se procede a realizar los cortes necesarios para poder hacer la primera representación y armado de la plantilla inicial mediante código HTML. En la Figura 3.2 se muestra la página de acceso al sistema. Para generar esta página y las páginas interiores de la aplicación (Figura3.3), solo se necesitan de 4 elementos gráficos:

1. La brújula de la página principal
2. Una flecha a manera de pleca en color amarillo
3. El logo iMos
4. Un píxel de color azul en RGB 66,101,173

Aquí se hace la puntualización que la página principal se codifica en HTML de manera independiente al resto de las páginas interiores por ser diferente. La figura 3.4 muestra la apariencia y distribución de las páginas interiores del sistema creadas por medio de una plantilla y un sistema de vestimentas dinámicas con la distribución de los elementos de información que permitirán al usuario conocer cual es su posición dentro del sitio y la estructura visual de la misma.

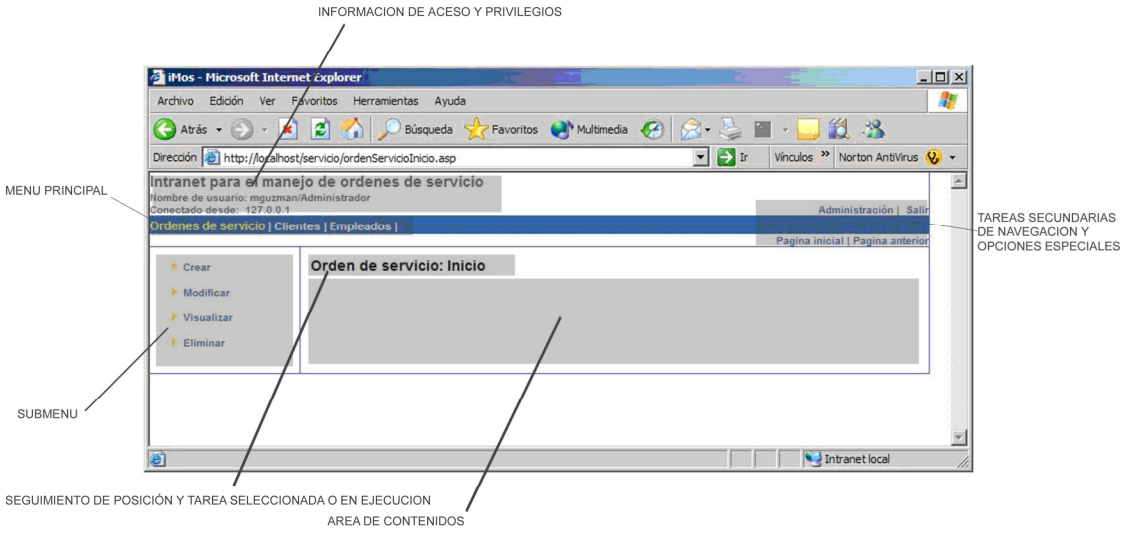

Figura 3.4 Identificación de elementos dentro de la plantilla

A pesar de que algunos elementos dentro de la plantilla pudiesen verse como información variable, tal como sucede en el caso del "Seguimiento de posición y tarea seleccionada o en ejecución" o la manera en como se ilumine la opción principal seleccionada, todos estos elementos pueden ser parametrizados e incluidos en el armazón que permitirá generar una plantilla mas ligera y fácil de mantener que al final dará una estructura con mantenimiento bastante eficiente bajo un esquema que el autor ha denominado "Vestimentas de aplicaciones WEB".

Esta forma de desarrollar aplicaciones para WEB, permite generar aplicaciones en las cuales el código de armado de todo el sitio se encuentra centralizado en una sola librería del lenguaje de desarrollo elegido para el proyecto específico, en este caso ASP, de tal suerte que no se tenga que repetir el código de los menús, los submenús, ligas, llamados a librerias Java Script, etc. En cada una de las "páginas" que forman parte de la aplicación.

Se escribe "páginas" entre comillas ya que dichas páginas serán el resultado de la ejecución de un programa escrito en ASP con la cuales interactuará el usuario.

En estos momentos ya se debe tener creada una página en HTML simple con la definición de hoja de estilos (hojaDeEstilo.css) en la cual se definen las tipografías y apariencias de los componentes de formas, ligas y demás elementos susceptibles de manipular por éste medio con la finalidad de dar aún un mejor control sobre el mantenimiento de la aplicación.
La parte medular de éste estilo de desarrollo se basa en la identificación clara de los elementos parametrizables, los elementos que solo sirven para dar estructura visual a la aplicación y por último definir e identificar el lugar en donde se presentará la información que variará entre "página" y "página" también conocida como área de contenidos tal como se describe en la Figura 3.4.

Una vez hecho esto se obtiene una librería (armazón.asp) con las siguientes subrutinas:

- encabezado(usuario,opcion,titulo,idTipoUsuario)
- pie()
- errorParse()
- llenaCombo(nombre,SQL)
- llenaComboAccion(nombre,SQL)
- llenaComboAccionSeleccionado(nombre,SQL,idComparacion)

- llenaComboSeleccionadoId(nombre,SQL,habilitado,cadenaComparacion)
- enviarAvisos(email,cc,mensaje,asunto)
- ip()

La rutina de encabezado permite que se construya la parte superior de cada una de las páginas generadas para el usuario con parámetros parámetros:

usuario: permite que se despliegue el nombre del usuario que está utilizando el sistema.
opcion: cambia el color de la opción principal elegido por el usuario en el menú principal y permite generar las ligas al submenu que le corresponde.
titulo: muestra en que parte del sistema se encuentra el usuario como seguimiento de posición o tarea seleccionada.
idTipoUsuario: muestra el tipo de usuario que hace uso del sistema y habilita las opciones a las que tiene acceso el mismo.

La rutina pie() construye la parte de cierre de la página, no requieres parámetros.

errorParse() se encarga de desplegar un error cuando la página que se invoca no ha sido invocada desde otra página del sistema con pase de parámetros con método Parse.

Las rutinas de llenado de combos se definieron pues se requieren en diferentes formas para el llenado de componentes de selección para el usuario y permiten automatizar dicho proceso con diferentes parámetros que construyen el objeto dentro de una forma con un nombre de componente, una consulta a la base de datos y parámetros opcionales que dan al objeto diferentes propiedades como habilitado, construcción del objeto con un elemento del conjunto recuperado por la consulta seleccionado de manera predeterminada.

enviarAvisos() es una rutina de envío de correo electrónico que se puede invocar desde diferentes rutinas o scripts generados en ASP para generar notificaciones del sistema a los diferentes involucrados en el seguimiento de las órdenes de servicio, está rutina se basa en CDOSYS (Collaboration Data Objects, Objetos de datos de colaboración) que está incluido en Windows Vista.

ip() recupera el número ip de la computadora desde la cual se está conectando el usuario para desplegarlo en la página que se esté utilizando y hacer algunas validaciones de acceso.

A partir de esto, la plantilla final de construcción de scripts generados en ASP para la aplicación queda definida de acuerdo al Código 3.1.

```
1.  <%@LANGUAGE="VBSCRIPT" CODEPAGE="1252"%>
2.  <!--#include VIRTUAL ="/servicio/armazon.asp" -->
3.  <% dim nombreUsuario
4.     dim sesionOriginal
5.     dim sesionActual
6.     if request.TotalBytes > 0 then
7.       nombreUsuario=Request.Cookies("usuario")("nombreUsuario")
8.       passwdEncriptado=Request.Cookies("usuario")("passwdEncriptado")
9.       idTipoUsuario=request.Cookies("usuario")("idTipoUsuario")
10.      sesionOriginal=request.Cookies("usuario")("sesion")
11.      sesionActual=Session.SessionID
12.      call encabezado(nombreUsuario,1,"Titulo",idTipoUsuario)
13.      if sesionOriginal <> sesionActual then
14.        response.redirect "http://localhost/servicio/html/sesionExpirado.html"
15.      else
16.              'CONTENIDO DE LA PÁGINA AQUÍ SE DEFINE LA TAREA DEL SCRIPT
17.      end if
18.      call pie()
19.    else
20.      call errorParse()
21.   end if
22.%>
```

Código 3.1

En el Código 3.1 se puede ver que de las líneas 6 a la 14 se han insertado algunas líneas de código que aparentemente no tienen una relación con lo que hasta el momento se ha dicho, pero estas líneas están directamente vinculadas con la seguridad de acceso al sistema, la cual es muy importante y de la que se hablará en las siguientes líneas.

3.4 Elementos y rutinas de seguridad de acceso a la aplicación.

Como todo sistema de gestión de información, el sistema debe mantener un acceso restringido a la información, en donde los usuarios de acuerdo a la definición del sistema solo deben ver la información que les concierne según a la clasificación o tipo de usuario que se les ha asignado.

Con la finalidad de evitar que personas no autorizadas, no vinculadas con el sistema puedan hacer uso de él o que los usuarios vean información no relacionada con ellos, se definieron diferentes mecanismos de autentificación y validación de acceso.

1. Validación de usuarios para permitir acceso al sistema.
2. Implementación de un mecanismo de encriptación para claves de acceso.
3. Establecimiento de sesiones con caducidad.
4. Manejo de cookies para validación en cada página o forma generada.
5. Verificación de parseo de información.
6. Relación a nivel de estructura de datos con la información que corresponde a cada usuario.

La validación de usuario se lleva a cabo en un script llamado “validación.asp”, el cual internamente funciona de acuerdo a la Figura 3.4, el script recibe la información de la clasificación de usuario, el nombre de usuario y clave de acceso en texto plano, esta se recupera del parse de la forma default.asp. Se hace uso de las funciones pwdencrypt() y pwdcompare() disponibles en el DBMS y que permite que la base de datos guarde las claves de acceso encriptadas de cada usuario.

Una vez que se tiene encriptada la clave de acceso que se recibió de la forma inicial, se procede a realizar la consulta de validación a la base de datos mediante pwdcompare(), seleccionando la consulta adecuada de dos diferentes opciones ya que se puede realizar la consulta a la tabla de clientes o a la de empleados, esta decisión se toma dependiendo de la

selección realizada por el usuario en la página inicial en la opción “Tipo de usuario” que puede ser Cliente o Empleado.

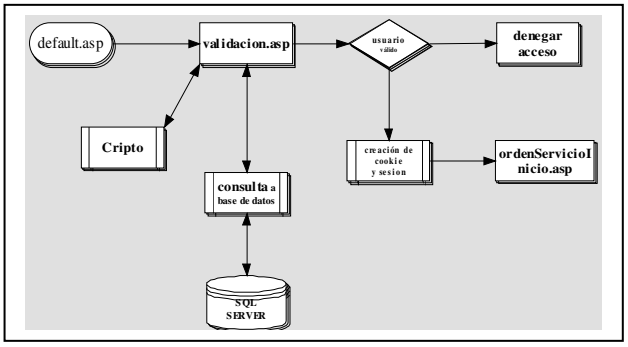

Figura 3.5 Funcionamiento del mecanismo de validación y acceso de usuarios

Se verifica si el recordset asociado a la consulta trae o no un valor asociado de acuerdo a los parámetros enviados, si el recordset está vacío, se enviará el mensaje de acceso denegado, de lo contrario, se genera una cookie que se envía a la computadora desde la que se conecta el usuario para la autentificación en cada script generado en ASP y se establece la sesión con caducidad 20 (vida máxima de 20 minutos) y por ultimo se invoca al script ordenServicioInicio.asp que es la página inicial dentro del sistema una vez autentificado el usuario, en éste script se realiza una validación de “parse” al momento de ser llamado a ejecución, si el arreglo de parámetros viene vacío, se desplegará un mensaje de error al usuario pues esto definirá un llamado directo al script sin haber pasado por la validación de acceso, este proceso de validación se realiza en todos los scripts generados en ASP que forman parte del sistema.

El algoritmo de validación y manejo de perfil para habilitar y deshabilitar opciones de acceso en el sistema trabaja de manera similar, pero en este caso no se hace generación de una cookie, el proceso consiste en recuperar la información que proviene de la cookie, se envían los parámetros a la rutina, hace la verificación a la base de datos y se recupera el recordset y se valida la existencia de registros en el vector de resultados, si no hay coincidencia, se regresa el error y se termina la ejecución del script mediante el error correspondiente, si existe el usuario, se procede a realizar la construcción del encabezado y submenu de acuerdo al perfil del usuario.

Para la construcción del encabezado se debe hacer una consulta a la base de datos para recuperar el perfil del usuario de la tabla PERFIL, una vez hecho esto se debe ir validando cada una de las ligas que se despliegan en el script seleccionado habilitando aquellas a las que el usuario tiene acceso en el color que les corresponda de acuerdo a la definición de la interfaz, dejando inhabilitadas aquellas a las que no tiene derecho a acceder en color gris (DDDDDD).

Ya que se ha validado al usuario con la sesión y con la consulta a la base de datos, se construye el cuerpo o contenido de la página a generar como Front End para el usuario.

3.5 Construcción del Front-End para la Gestión de la Información

En estos momentos ya se tienen pobladas las tablas de catálogo (AREA, PERFIL, ESTADO, TIPOMENSAJE), que son necesarias para empezar a construir los scripts que permitirán gestionar la información.

La construcción de los scripts realiza de tal manera que se vayan poblando las tablas que permiten definir a otras tablas que dependen de ellas.

El lector debe recordar que el diseño está orientado a Objetos, de lo cual se establecieron clases, que tienen métodos variables de instancia que al bajarlas al modelo de desarrollo y herramientas de programación, se mapean de la siguiente manera:

- Las variables de instancia como formas que se alimentarán o permitirán modificar información.
- Los métodos como scripts o rutinas dentro de librerías que se invocan cuando son requeridas y que permiten manipular la información.

3.5.1 Definición del manejo de empleados

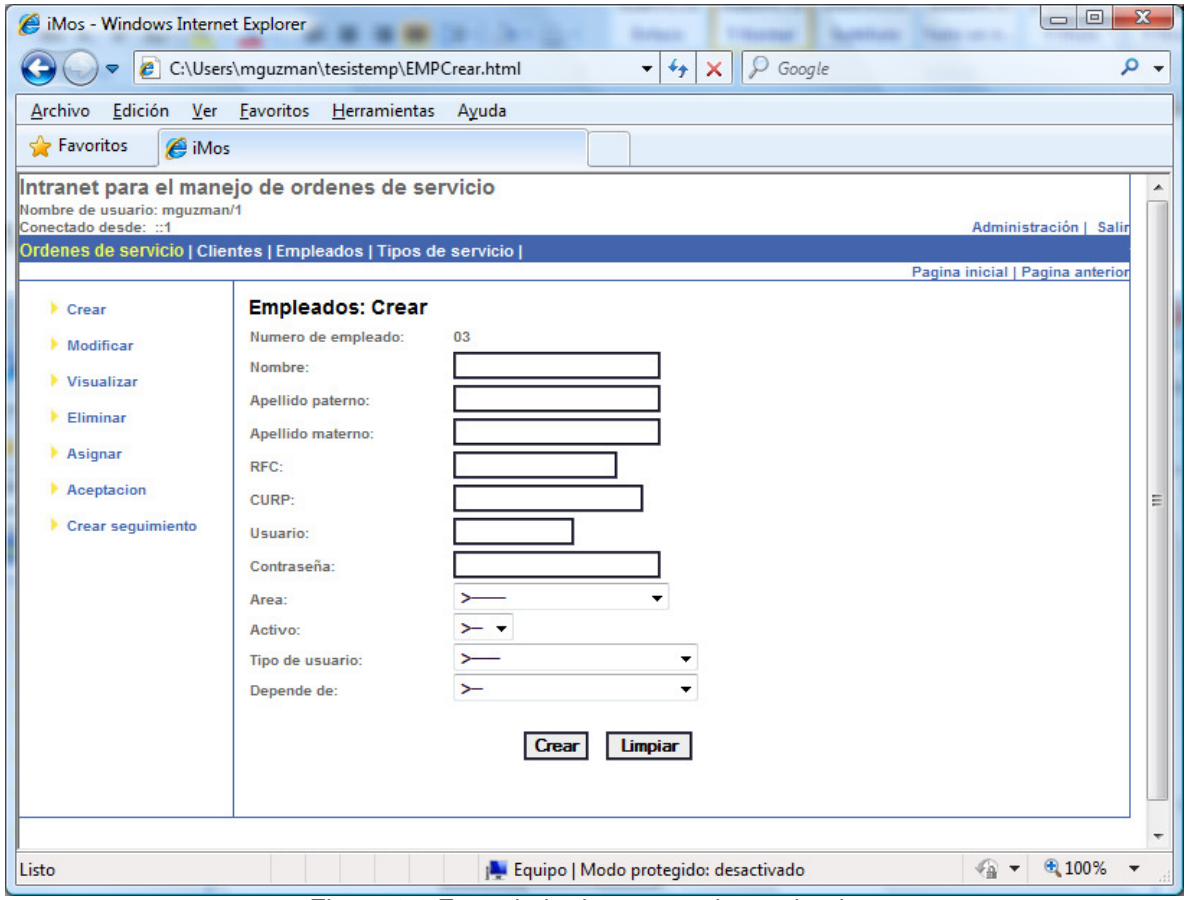

Figura 3.6 Formulario de captura de empleados.

La primera clase a implementar con sus métodos será la clase EMPLEADO (Figura 3.6) cuya información se almacena en la tabla EMPLEADO, para la cual ya existen las tablas de catalogo necesarias para su implementación como son AREA y PERFIL.

Para ésta entidad se definen los scripts siguientes:

- empleadoAgregar.asp: se encarga de dar de alta a un empleado en la base de datos con sus atributos tal cual se definen en el diseño.
- empleadoModificar.asp: es el script que permite la modificación de los atributos de un empleado y desde él se puede invocar al script cambiaPassword.asp que es una rutina que permite cambiar el password de los usuarios del sistema (EMPLEADO o CLIENTE).
- empleadoEliminar.asp: permite habilitar o inhabilitar a un empleado de la base de datos con la finalidad de conservar la integridad de la información.
- empleadoListar.asp: Muestra un listado de todos los empleados que están dados de alta en la base de datos para su manipulación.

Se debe hacer especial énfasis en que la rutina de alta de empleados (empleadoAgregar.asp) así como la de modificación de empleado (empleadoModificar.asp) tienen un nivel especial de complejidad pues incluyen mecanismos especiales para la asignación un empleado a un área en especial, se debe recordar que esto se define por medio de la relación entre AREA y EMPLEADO y un empleado es encargado de un área cuando éste se asigna al área y no está subordinado a otro empleado con la relación recursiva depende de.

Un usuario administrador se distingue del resto por tener asignada el área Administración que solo se asigna por medio de la puesta a punto de la base de datos con el identificador 0.

3.5.2 Definición del manejo de los mensajes

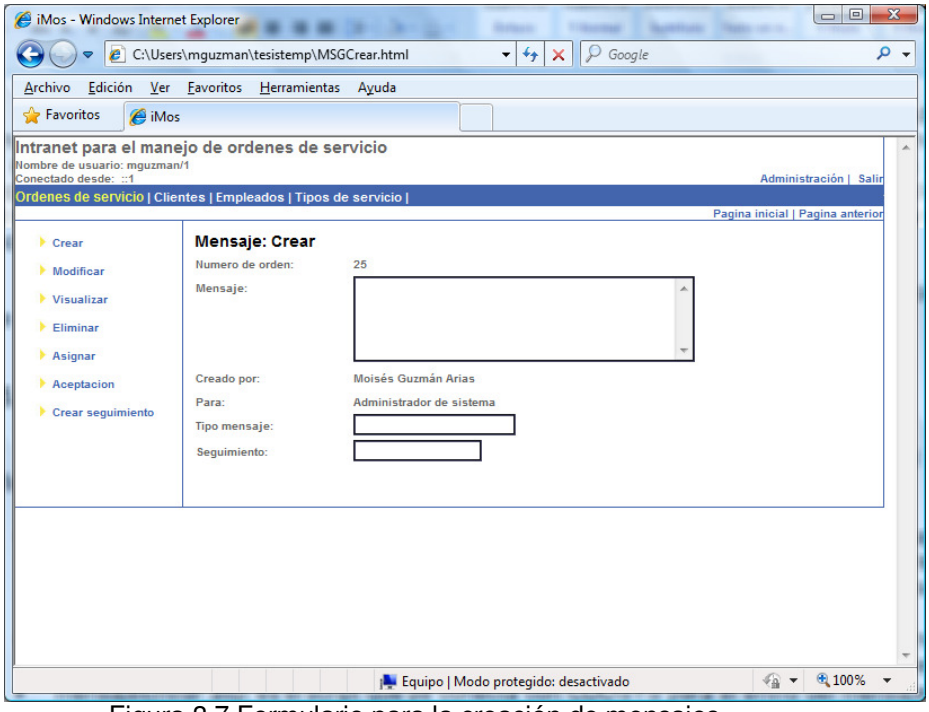

Figura 3.7 Formulario para la creación de mensajes.

Hecho lo anterior, se procede a la definición de la clase mensaje, para ésta entidad se definen los siguientes scripts:

- mensajeCrear.asp: es el encargado de desplegar la forma para la redacción del mensaje, muestra los tipos de mensaje válidos para selección del usuario (reclamo, mensaje aviso) ya que dentro de la definición de la tabla TIPOMENSAJE existen algunos que son protegidos y solo el sistema por si mismo es quien los puede generar y catalogar.
- mensajeVisualizar.asp muestra todos encabezados de los mensajes que se tienen relacionados con una orden de servicio con la opción de poder ver cada mensaje completo.
- mensajeEnviar.asp: es el script que se conecta con CDOSYS para el envío del mensaje al o los destinatarios.
- mensajeAlmacenar.asp: se encarga de almacenar y catalogar en la base de datos todos los mensajes relacionados son una orden de servicio.

3.5.3 Definición del manejo de clientes

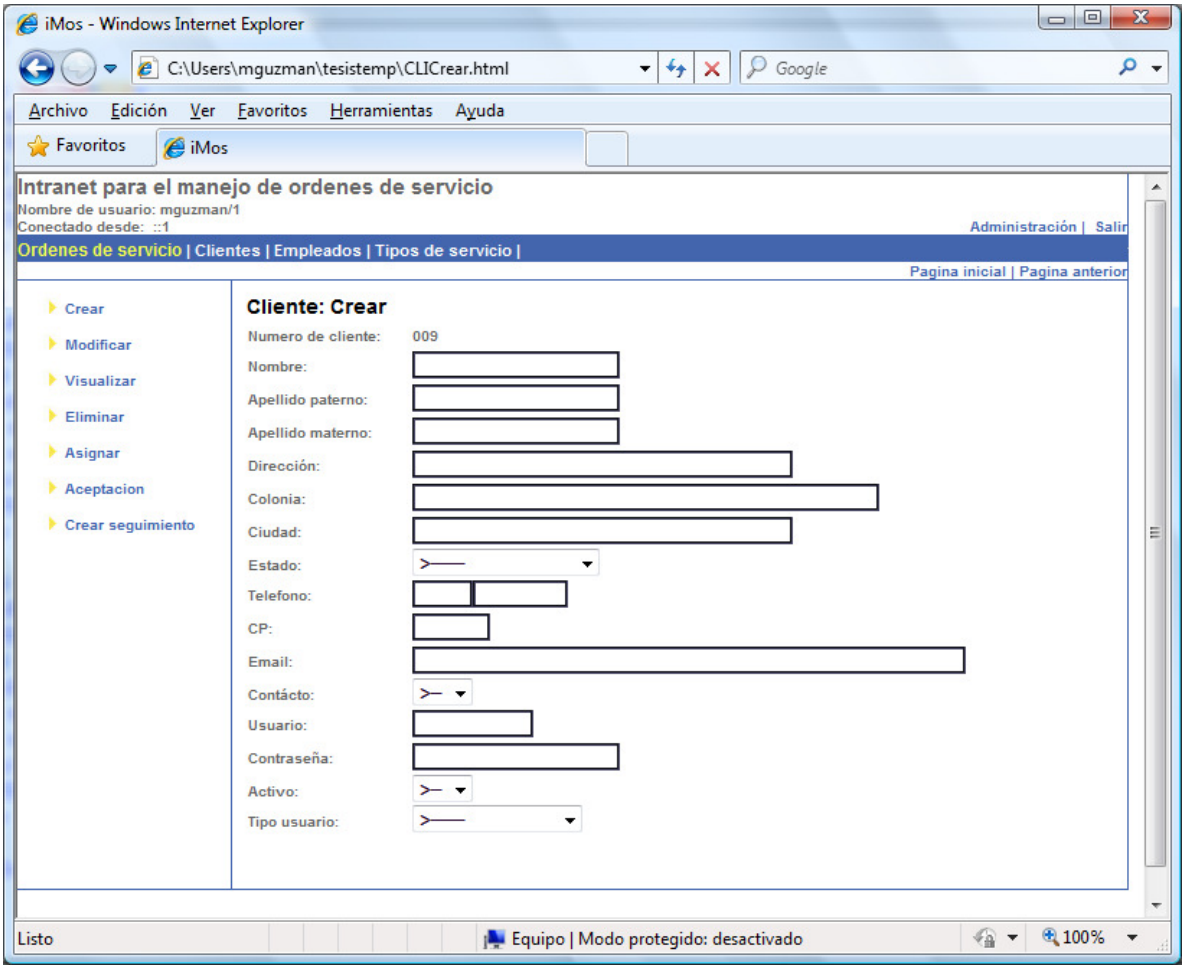

Figura 3.8 Formulario para la creación de clientes.

El manejo de los clientes es un punto medular del sistema pues sin ellos no habría razón de ser del mismo al igual que el manejo de lar órdenes de servicio, para el manejo de los clientes se definen los siguientes scripts:

- clienteAgregar.asp: permite agregar a un cliente en la base de datos con toda su definición estipulada en el Capítulo 2.

- clienteModificar.asp: modifica los atributos de un cliente definido en la base de datos conservando la integridad de la información y manteniendo los estandares definidos en las secciones anteriores.
- cleinteVisualizar.asp: permite visualizar la información de un cliente por el mismo cliente y por un usuario definido como tipo recepción para validar sus datos cuando se levante una orden de servicio.
- clienteEliminar.asp, permite habilitar o deshabilitar a un cliente con la finalidad de no romper la integridad de la información que pudiese estar vinculada por el esquema de la base de datos.
- clienteLibrerias.asp: incluye las rutinas de cambio de password de un cliente, y listar clientes para su selección y manipulación.

3.5.4 Definición de la manipulación de áreas

Aún cuando las áreas se definen en un momento como una entidad de catálogo, se requiere implementar los mecanismos de administración para las mismas ya sea por expansión de la empresa o por eliminación de áreas, por lo cual se definen los siguientes scripts:

- areaAgregar.asp: define las tareas necesarias para crear un área nueva.
- areaModificar.asp: manipula la información de un área sin que esto implique romper con la integridad de la estructura de datos definida incluso inhabilitándola si es necesario cuando un área ya no sea necesaria operacionalmente con la finalidad de no crear ordenes de servicio para un área que ya no preste servicios.
- prealistar.asp: muestra todas las áreas definidas para la empresa y permite seleccionar alguna para su manipulación.

3.5.5 Definición de la manipulación y creación de tipos de servicio

Un tipo de servicio más que definir un tipo, permite definir servicios asignados a las diferentes áreas de la empresa, incluye un identificador de servicio almacenado en idTipoServicio, el nombre del servicio y el identificador del área a la que se asigna, pero esto no es todo, existe una tabla adicional que permite asignar pasos a un servicio en especial, lo que se hace en la parte de modificación del tipo de servicio en donde se puede crear toda la estructura de los pasos que conforman el servicio, agregar, deshabilitar, etc.

- tipoServicioAgregar.asp: agrega la nueva definición de un servicio a la base de datos, de acuerdo a la estructura estipulada en la estructura de datos y si se desea definir los pasos que conforman al tipo de servicio, se hace un llamado a tipoServicioModificar.asp en el cual se procede a la manipulación de los pasos que lo constituyen.
- tipoServicioModificar.asp: permite manipular la información que define a un tipod e servicio incluyendo los pasos que lo conforman invocándose desde aquí a las librerías pasoAgregrar, pasoModificar y pasoVisualizar.
- tipoServicioEliminar.asp: desactiva un tipo de servicio que ya no está disponible, pero del cual se debe conservar la información para referencia de órdenes de servicio creadas.
- recuperarPasos es una librería que permite mostrar los pasos de un tipo de servicio definido en la ventana de modificación.
- tipoServicioListar.asp permite mostrar todos los tipos de servicio existentes y seleccionar uno para su manipulación.

3.5.6 Definición de la manipulación de órdenes de servicio

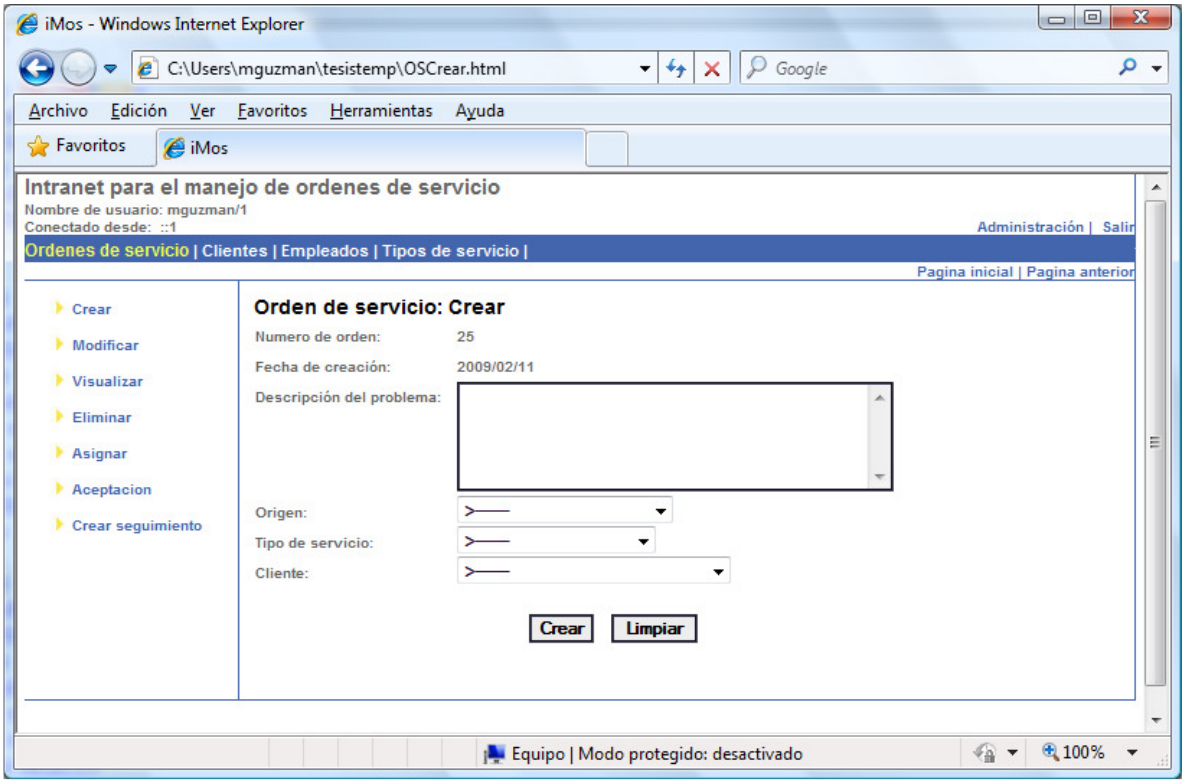

Figura 3.9 Formulario para la creación de una orden de servicio

Las órdenes de servicio son la parte principal del sistema y existen otras clases que interactúan con ella como es el caso del seguimiento y asignación de órdenes. La estructura de las órdenes de servicio permite crear y manipular la definición de una orden de servicio más no su proceso o asignación.

- ordenServicioAgregar.asp: crea la estructura de una orden de servicio asignada a un cliente (Figura 3.X), aquí se establecen algunas normas especiales para variar la manera en como se genera la forma de captura pues es una forma a la cual tienen acceso diferentes tipos de usuarios (Encargado de área, recepción y clientes), para el caso de selección de cliente para la creación de la orden de servicio pues un recepcionista o encargado de área pueden seleccionar el cliente al que se le asigna la orden y un cliente no puede hacerlo y de manera predeterminada debe aparecer preseleccionado su nombre.
- ordenServicioModificar.asp: permite al usuario que tiene acceso a él de acuerdo a la tabla PERFIL manipular una orden de servicio creada sin romper con la integridad de la información.
- ordenServicioVisualizar.asp: muestra la definición de una orden de servicio y de manera adicional permite acceder al seguimiento de la misma para los usuarios que así lo tengan permitido.
- ordenServicioEliminar.asp: elimina una orden de servicio cuando es necesario, solo para los usuarios que así lo tienen permitido.
- ordenServicioCouncluir es una rutina que se incluye en la modificación de una orden de servicio y que solo está disponible para los usuarios administrador y encargado de área.
- ordenServicioRecuperarMsg: es una rutina que permite ver los mensajes relacionados con la orden de servicio hasta el momento en que se está revisando.

- recuperarSeguimiento: es la rutina que permite visualizar el seguimiento de la orden, o lo que es lo mismo como va su proceso de ejecución.
- ordenServicioListar.asp: Muestra un listado de órdenes de servicio a partir de uncliente determinado a un área definida, éstas opciones se habilitan dependiendo del usuario que accede al sistema. Para un técnico, le mostrará las ordenes de servicio que son o fueron ejecutadas por él. Para un cliente solo verá las ordenes de servicio que tiene vinculadas, para un encargado de área le mostrará las órdenes de servicio de su área y para el administrador se maneja una forma de selección y categorización de la información, lo mismo que para un recepcionista que tendrá la opción de hacer búsquedas por un tipo de cliente específico para visualización solamente mas no para manipulación. Todo esto de acuerdo a la tabla PERFIL.

3.5.7 Definición de la manipulación para el seguimiento de órdenes de servicio.

El seguimiento de órdenes de servicio es el encargado de dar seguimiento a la ejecución de un servicio solicitado por un cliente y se basa fuertemente en los pasos definidos para un tipo de servicio manteniendo la secuencia de ejecución de cada paso y no permitiendo que se ejecute un paso sin antes ejecutar los anteriores.

- seguimientoAgregar.asp: se encarga de generar el seguimiento para un paso específico de una orden de servicio asociada a un tipo de servicio cuidando los prerrequisitos del mismo debiéndose observar las restricciones de la tabla de PERFIL.
- seguimientoModificar.asp. permite modificar el paso en ejecución, no permitiéndolo así para los pasos que ya se encuentren marcados como concluidos.
- cerrarPaso.asp: bloquea la modificación o actualización del seguimiento en un paso de una orden de servicio para permitir la ejecución de seguimiento del que le sucede.
- seguimientoVisualizar.asp Genera un reporte del seguimiento de una órden de servicio hasta el momento de la consulta.

3.5.8 Definición de la manipulación de la asignación de órdenes de servicio

La asignación de las órdenes de servicio permite definir quién es el encargado de ejecutar físicamente la orden de servicio, para lo cual se definen los siguientes scripts y rutinas:

- asiganacion.asp: permite definir y asignar quien es encargado de ejecutar la orden de servicio en cuestión, pero esto no queda totalmente definido hasta que a quién se le asignó cambie el estado de asignación a aceptado.
- aceptación.asp permite a un técnico o encargado de área aceptar la ejecución física de una órden de servicio o rechazarla.
- modificarAceotacion.asp: permite al encargado o administrador modificar el estado de aceptación para reasignar la orden de servicio a la persona que la rechazó o a una nueva persona.
- asignacionBusqueda.asp: muestra todas las asignaciones disponibles por técnico y área, permitiendo además seleccionar por que tipo de estado de asignación se desea buscar.

Todas las rutinas aquí descritas y los scripts definidos dan acceso a los usuarios solo a la información a la que tienen derecho, con la finalidad de mantener la confidencialidad de la información.

Conclusiones y perspectivas

El objetivo de este trabajo de tesis fue el desarrollo de un sistema basado en WEB para el seguimiento de órdenes de servicio. Para cumplir con el objetivo, se planteó una metodología para el desarrollo del mismo. En este caso se hizo uso del modelo de cascada para la generación de la base de datos, mientras que para el desarrollo de la aplicación se utilizó el modelo de prototipos.

La metodología planteada garantizó la conclusión del proyecto en tiempo con la calidad deseada. Se usó el modelo entidad-relación para el modelado conceptual de la base de datos en su etapa de análisis y diseño, mientras que para el modelo conceptual de la planeación se usaron casos de uso e identificación de clases de UML. Por otra parte, se utilizaron diagramas de clase y diagramas de secuencia para modelar el diseño de la aplicación. Finalmente se usó MS SQL Server para la implementación de la base de datos, mientras que para la aplicación se usó ASP.

El objetivo se ha cumplido en su totalidad y se espera poner en marcha el sistema a la brevedad posible en alguna empresa o institución interesada en llevar de manera adecuada el seguimiento de órdenes de servicio y la evidencia de su ejecución.

Al momento, el sistema cuenta con las funciones “Altas”, “Bajas”, “Consultas” y “Modificaciones” sobre las clases siguientes:

- Clientes
- Órdenes de servicio
- Empleados
- Áreas
- Tipos de servicio
- Asignación de órdenes de servicio
- Seguimiento de órdenes de servicio
- Envío de mensajes

Adicionalmente, el sistema permite realizar envío de correos electrónicos a la empresa. Esta función fue implantada bajo el uso de un componente ASP llamado “CDOSYS.DLL”, el cual hace uso del servicio SMTP del servidor.

Algunas de las funciones que podrían a este sistema son las de creación automática de órdenes de servicio. Podría agregarse una orden vía internet y el sistema podría enviar a imprimir las órdenes de alta prioridad a incluso podría mandar mensajes de aviso (por correo electrónico) a los responsables que tengan algún retardo en la ejecución de una orden de servicio agilizando la ejecución de órdenes de servicio que se hubiesen detenido en un punto del proceso.

Se pueden obtener mayores beneficios mediante la explotación de la información para:

- Generar indicadores de desempeño de los empleados
- Establecer una base de conocimiento para la resolución de órdenes posteriores similares a las ya resueltas.
- Contar con información que permita la toma de decisiones en tiempo real.
- Establecer un proceso de mejora continua en la empresa que adopte el sistema a partir de la información arrojada por el sistema.
- Indicadores de consumo de los clientes.
- Facturación automatizada.
- Estado de cuenta en línea.

Referencias

Internet

[1L] http://www.htmlweb.net

[2L] Lynch and Horton
http://www.webstyleguide.com

[3L] Andrés Lewin / Programación en Castellano
http://www.programacion.com/asp

[4L] Instituto Seguridad Internet
http://www.instisec.com/publico/descargas/criptoasp.asp

[5L] W3C
http://www.w3.org

Material bibliográfico

[1] Jim Buyens
Aprenda Desarrollo de Bases de Datos WEB Ya.
McGraw-Hill División Profesional, 1a edición 2001.

[2] Nolan Hester
Dreamweaver UltraDev 4
Anaya Multimedia, 1a edición 2002

[3] Martin Rinehart
Desarrollo de Bases de Datos en Java
McGraw-Hill, 1a edición 1998

[4] Shelley Powers
Desarrollo de Componentes ASP
Anaya Multimedia, 1a edición 2001

[5] Charles Richter
Designing Flexible Object-Oriented System with UML
McMillan Technical Publishing U.S.A., 1999

[6] Lázaro Issi Camy
Dreamweaver UltraDev 4
Anaya Multimedia, 2001

[7] Jorge Serrano Pérez
Programación con ASP .NET
Anaya Multimedia, 2002

[8] Abbey Corey
Oracle 8 Guia de Aprendizaje
McGraw-Hill, Oracle Press, 1998

[9] James Rumbaugh, Ivar Jacobson, Grady Booch
El lenguaje Unificado de Modelado, Manual de referencia
Addison Wesley, 2000

[10] Craig Larman
UML y patrones
Prentice Hall, Pearson, 1999

[11] Roger S. Pressman
Ingeniería del Software, Un Enfoque Práctico
McGraw Hill, 1994

[12] Autor Andrew S . Tanenbaum
Redes de computadoras
Prentice Hall PTR, 2003

[13] Ron Soukup
A fondo Microsoft SQL Server
Mc Graw Hill, 2002