UNIVERSIDAD NACIONAL DE INGENIERIA

FACULTAD DE INGENIERIA INDUSTRIAL Y DE SISTEMAS

UN SISTEMA COMERCIAL EN CLIENTE - SERVIDOR CON AS-400

INFORME DE INGENIERIA

PARA OPTAR EL TITULO PROFESIONAL de:

INGENIERO DE SISTEMAS

PRESENTADO POR

LUIS ANTONIO MENDOZA MARTINEZ

LIMA - PERU

1997

Un sistema comercial en Cliente/Servidor con AS/400

Luis Antonio Mendoza Martínez

Febrero de 1997

Agradecimientos a:

mi madre,

mi hermano,

mis compañeros de trabajo.

Descriptores Temáticos

Cliente/Servidor.

GUI.

AS/400.

DB2/400.

Programación ODBC.

Sistema de Distribución.

Metodología de Desarrollo.

Introducción

Actualmente los sistemas de información tienen los mismos componentes en base de datos, además de tener productos conocidos en el mercado.

Por esta razón, antes que mostrar un sistema de información, el objetivo principal del presente trabajo es mostrar una experiencia de desarrollo de Cliente/Servidor en AS/400.

La experiencia es relativamente nueva y la configuración con la que se ha trabajado es única en nuestro país.

A simple vista es la descripción de como desarrollar una aplicación utilizando como servidor el AS/400 y como cliente el Visual Basic 3.0 con interface de programación ODBC.

La experiencia fue desarrollada desde setiembre de 1995 a la fecha. Actualmente se tiene en producción 4 sistemas bajo esta plataforma. Uno de estos sistemas es el sistema de Distribución.

No se han aprovechado todos los recursos que puede dar en Cliente/Servidor la plataforma elegida porque el Departamento de Sistemas de **Nissan Maquinarias S.A.** va descubriendo nuevas formas de trabajo para aprovechar los recursos y mejorar la performance del sistema, los cuales recién se están implementando en los nuevos sistemas.

Resumen Gerencial

Los principales componentes del trabajo son:

Cliente/Servidor. Se define conceptos y se muestra la aplicación de Cliente/Servidor.

GUI (Graphical User Interface) . Se define un modo de trabajo para hacer que los diálogos de usuario sean comprensibles y fáciles de manejar.

AS/400. Se describe el trabajo en Cliente/Servidor, comunicaciones remotas vía TCP, el Client/Access.

DB2/400. La base de datos nativa del AS/400. Se describe Los conceptos de base de datos como: Triggers, Integridad Referencial, Restricciones, Stored Procedures.

Programación ODBC. Se describe el estilo de programación que se ha usado en el cliente. Las funciones que se deben ejecutar para pasar parámetros entre el cliente y el servidor.

Sistema de Distribución. Se describe el núcleo central del sistema de distribución. Se presenta el modelo de datos y los ejemplos de programación utilizados en el desarrollo del sistema.

Metodología de Desarrollo. Se aplica la metodología de desarrollo por fases del proyecto: Planeamiento/definición, Diseño Conceptual, Diseño Computacional.

Descripción de la empresa

Nissan Maquinarias pertenece a un grupo económico que cuenta con las siguientes empresas:

- Maquinarias Comercial S.A. Concesionario de vehículos.
- ARMAQ S.A. Comercializa armas.
- Maquimar S.A. Empresa que comercializa conservas de pescado.
- LIA S.A. Empresa que se encarga de fabricar carrocería para autos y camiones.
- IE S.A. Ingenieros Ejecutores de proyectos de construcción.
- Pre-Entrega S.A. Empresa encargada de la entrega del vehículo en óptimas condiciones a los concesionarios.

Nissan Maquinarias S.A. es esencialmente una empresa dedicada a la importación y distribución de vehículos de marca NISSAN. Hasta agosto del presente año la empresa se llamó Maquinarias S.A. y precisamente ese mes se fusionó con Nissan del Perú S.A. dando origen a Nissan Maquinarias S.A.

Anteriormente, Maquinarias ya había ampliado sus operaciones con Comercialización de Camiones y la División de Minería y Construcción.

Debido a los constantes cambios de políticas que se están produciendo, se considera a Sistemas como un Departamento vital dentro de la empresa. Sistemas no sólo es considerado un departamento proveedor de información ordenada sino como un punto de inicio para todos los cambios administrativos que se están produciendo.

Estas observaciones deben ser tomadas como un límite al desarrollo del presente trabajo, pues se considera el Sistema de Distribución, como parte del rubro principal de la empresa.

Sección I. Conceptos Generales

Capítulo 1. Cliente/Servidor

1.1. ¿Qué es Cliente/Servidor?

Cliente/Servidor es un modelo para construir sistemas de información partiendo de una diversidad de tecnologías que distribuyen aplicaciones, datos o servicios a través de múltiples procesadores.

El término Cliente/Servidor implica la relación entre unos procesos que inician solicitudes de servicios (clientes) y otros procesos que responden a ellas (servidores), que se ejecutan en el mismo procesador o en otro distinto de la red.

El modelo Cliente/Servidor permite una clara separación de funciones basada en el concepto de servicio, posibilitando situar aplicaciones en la plataforma más adecuada, dentro de un entorno de sistemas abiertos y heterogéneos, aumentando así los beneficios de la empresa.

1.2. Características de una aplicación Cliente/Servidor

Una aplicación Cliente/Servidor se compone de varios procesos clientes y servidores que se pueden distribuir en una red. El Middleware Cliente/Servidor conecta los procesos que componen la aplicación. Este término, ya muy extendido, se utiliza para describir aquellas funciones que son necesarias para proporcionar la transparencia de localización.

Es a través del Middleware que estos procesos se conectan e interactúan constituyendo aplicaciones. El Middleware agrupa una serie de funciones y servicios (directorios, seguridad, tiempo... etc.) no incluidos en las aplicaciones ni en el sistema operativo, que aíslan al programador de tener que ocuparse de los aspectos de bajo nivel propios de sistemas distribuidos, comunicaciones, directorios, integridad, etc.

1.3. Diferencia con un desarrollo sin Cliente/Servidor

La diferencia fundamental está en que con Cliente/Servidor nos independizamos del hardware y del software, pudiendo comunicarse a computadoras con sistemas operativos diferentes y sistemas de manejo de base de datos diferentes a través de un estándar (ODBC por ejemplo) de interface de comunicación de software.

1.4 Ventajas de la Tecnología Cliente/Servidor

Algunas razones porque se debe elegir Cliente/Servidor como plataforma de desarrollo para el nuevo sistema:

a. Es una tecnología que recién está introduciendo IBM para AS/400 en el Perú, con lo que estaría dentro del Objetivo Estratégico de la Empresa.

b. A nivel de software es portable e independiente, ya que los clientes no dependen de la plataforma del servidor.

c. Se pueden aprovechar fácilmente las nuevas herramientas de desarrollo para adaptar el desarrollo a los requerimientos de la empresa.

d. Cuando se programa convenientemente (en nuestro caso con interface ODBC) se independiza las aplicaciones inclusive del servidor.

Capítulo 2. ODBC

El componente ODBC utiliza una arquitectura de interface de acceso a Base de Datos que permite que las aplicaciones utilicen datos usando SQL como standard.

2.1. Funciones ODBC

El ODBC provee un conjunto de API´s (Aplication Program Interface) que permiten a una simple aplicación utilizar datos de diferentes sistemas de manejo de datos para lo cual define una librería de funciones de llamada que permite a una aplicación:

- Conectar a un Sistema de Manejo de Base de Datos,
- Ejecutar sentencias SQL,
- Recibir resultados.
- La interface ODBC además provee:
    - Sintaxis de SQL,
    - Un estándar de códigos de error,
    - Una forma estándar para conectarse a una DBMS,
    - Un estándar de representación para tipos de datos.

En resumen, todos los lenguajes de programación y bases de datos que cuenten con la interface ODBC pueden comunicarse enviando y recibiendo datos.

Por ejemplo:

El Visual Basic 3.0 trabaja como nativo con la base de datos Access 1.0.

El Visual Basic 4.0 trabaja como nativo con la base de datos Access 2.0.

Entonces El Visual Basic 3.0 no puede trabajar con la base de datos del Access 2.0. por un problema de versión.

Si instalamos los drivers ODBC que viene con el lenguaje Visual Basic 3.0 e instalamos los drivers que vienen con el Access 2.0 entonces podemos crear un Data

Source (o modo de acceso) direccionado a una base de Datos definida en Access 2.0 y así el Visual Basic 3.0 podrá comunicarse con la Base de Datos Access 2.0 a través de ODBC. (Ver figura 1)

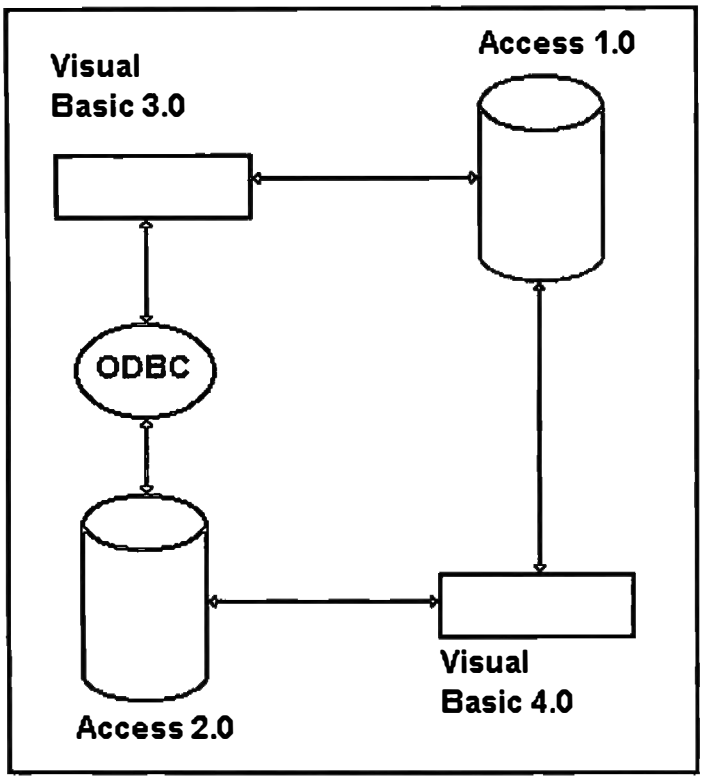

Figura 1. Acceso a base de datos a través de ODBC

Si generalizamos, el Sistema Manejador de Base de Datos puede ser cualquiera y el lenguaje de programación también puede ser cualquiera, con el único requisito que ambos tengan los drivers ODBC indicados.

Como conclusión podemos decir que si tenemos un servidor AS/400 con el DB2/400, cualquier lenguaje de programación (en cualquier plataforma o sistema operativo) puede acceder a los datos siempre que trabaje bajo la lista de Sistemas Operativos para los cuales tiene Drivers ODBC el AS/400. (Figura 2)

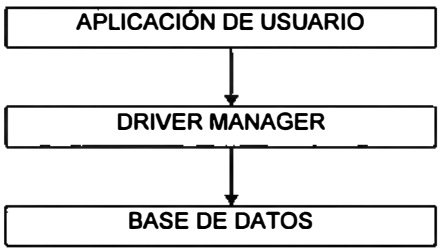

Figura 2. Esquema de ODBC

Capítulo 3. Conceptos GUI (Graphical user interface)

3.1. ¿Que es GUI?

GUI es un conjunto de técnicas que hacen fácil el desarrollo de aplicaciones con un lenguaje visual. Estas técnicas están basadas en un estándar de percepción del usuario que tiene que ver con el uso de colores y símbolos conocidos en el manejo de ventanas (uso común por ejemplo en MS Windows) y eventos.

3.2. Características de un diseño GUI

Al desarrollar bajo ambiente visual se debe tener en consideración que las ventanas de diálogo:

- Deben ser predecibles,
- Deben ser atractivas,
- Deben ser fáciles de leer,
- Deben poder ser usadas en una gran variedad de monitores,
- Deben seguir un patrón general,
- Deben hacer que los usuarios sean más productivos,
- Deben hacer mucho más fácil el trabajo de los usuarios.

3.3. Diferencia con un desarrollo en modo texto

En modo texto estamos limitados a utilizar 80 caracteres por línea con 24 líneas por pantalla (en algunos monitores se puede trabajar con 132 caracteres x 50 líneas).

En modo texto no podemos incluir gráficos o símbolos que son reconocidos por los usuarios que trabajan en ambiente gráfico (por ejemplo impresora, disco, etc.).

Esto nos obliga a diseñar un menú describiendo todas las opciones que se van a utilizar usando solo texto. Si un panel no puede contener todas las opciones tenemos que establecer una jerarquía para navegar a través del sistema. Esto nos obliga a usar una mayor cantidad de teclas y provoca más puntos de espera.

En modo gráfico se pueden poner infinidad de opciones en un solo panel. Teóricamente es posible definir el núcleo central del sistema en un solo panel, de tal manera que el usuario se pueda movilizar en una sola opción de menú que es fácil de aprender y como consecuencia puede agilizar su trabajo.

3.4 Ventajas del desarrollo en ambiente gráfico

Al desarrollar en ambiente gráfico se tienen las siguientes ventajas:

a. Se puede incluir una variedad de íconos y estilos de texto que permiten tener las opciones más frecuentes del sistema en un solo panel.

b. Es fácil la capacitación del usuario porque se tiene la ayuda de gráficos para simbolizar las opciones.

c. No es necesario tener varios niveles de menú porque se pueden resolver varias opciones en un solo panel.

d. La programación por ventanas y eventos permite ver el efecto inmediato de los cambios sin necesidad de tener una tecla de comando (la tecla Enter, Intro o Return o teclas de función).

3.5 Elementos GUI

Los elementos a tener en cuenta en ambiente gráfico son:

3.5.1. Windows o ventanas

Son las franjas básicas para alojar datos y comandos en una aplicación. Es equivalente al panel de un programa en modo texto. (Figura 3)

Ejemplo de un Window

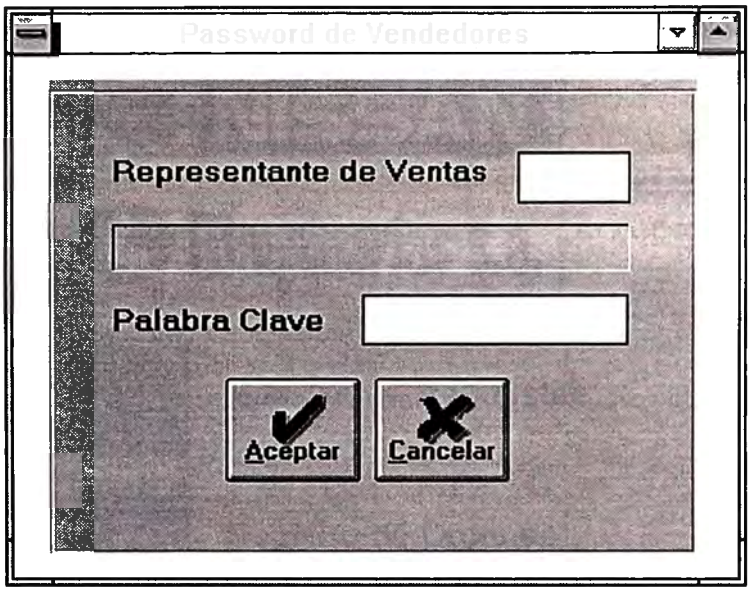

Figura 3. Ejemplo de un Window.

3.5.2. Iconos, Dibujos y Bitmaps

Los íconos, dibujos y bitmaps ayudan a representar fácilmente las acciones del sistema.

3.5.3. Menús

Es una parte esencial que hace atractivo GUI. Si todas las características de un programa se muestran en un menú, entonces incentiva a buscar exactamente lo que se quiere hacer. (Figura 4)

Figura 4. Ejemplo de un Menú

3.5.4. Mensajes y cajas de diálogo

Los programas en ventanas pueden mostrar otras ventanas cortas que solicitan o proveen información. Estos windows se llaman Cajas de Diálogo. (Figura 5)

Figura 5. Ejemplo de caja de dialogo

3.5.5. Barra de herramientas

Es una colección de íconos que representan y activan ítems del menú. (Figura 6)

Figura 6. Ejemplo de caja de diálogo.

3.5.6. Controles

Los contenidos de cajas de diálogo son usualmente ítems gráficos que proveen alternativas de entrada. Pueden ser botones, combos. A estos ítems de entrada se llaman controles. (Figura 7)

Ejemplo de algunos controles:

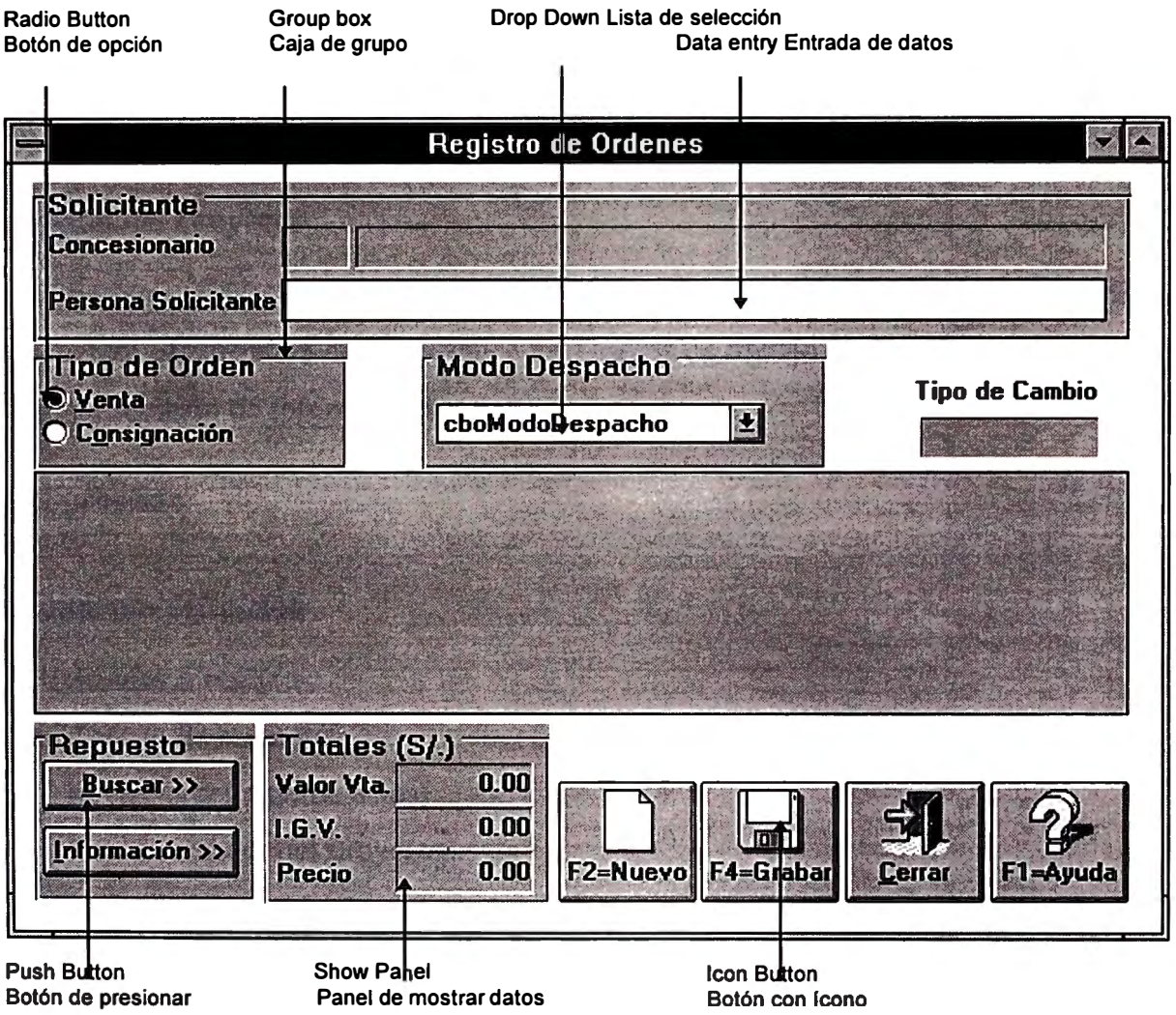

Figura 7. Ejemplo de controles.

Radio Button. Se puede elegir entre las opciones que estén amarradas por el mismo nombre de radio button, son opciones excluyentes.

Group box. Dibujo para agrupar visualmente varios controles.

Drop Down. Control que tiene la opción de seleccionar un valor mediante un botón (flecha abajo) que al ser presionado muestra una lista para elegir un dato.

Data entry. Campo para ingreso de datos.

Push Button. Botón con contenido texto que al ser presionado ejecuta una acción.

Show Panel. Control en el que se puede mostrar datos que no van ha ser actualizados.

Icon Button. Botón con contenido texto + ícono que al ser presionado ejecuta una acción.

Sección II. Metodología a utilizar

Para el desarrollo del sistema se ha adoptado una metodología propia de la instalación:

Organización del Proyecto

Definir la organización del proyecto y la asignación de responsabilidades. Se define:
Estructura organizativa del proyecto y los Grupos de trabajo.
Plan de revisión de información.

Fases del proyecto

Planeamiento/definición

Actividades a realizar:

- Agenda de reuniones.
- Revisión del sistema actual.
- Plan de trabajo del proyecto.

Producto: Memorando de alcances

- Introducción descriptiva sobre el sistema.
- Antecedentes, objetivos y alcances del sistema.
- Diagnóstico del sistema actual.
- Definición de etapas y entregas.
- Organigrama del equipo de trabajo.
- Análisis beneficio/Costo.

Diseño Conceptual

Es la presentación global del sistema, identificando sus principales componentes y las relaciones entre ellos. Se hace la identificación jerarquizada de los módulos, submódulos y funciones.

Actividades a realizar:

- Identificar requerimientos funcionales.
- Analizar el flujo de documentos del sistema.
- Identificar entradas y salidas de datos del sistema.
- Diseño preliminar de datos.
- Requerimiento y definición de recursos.
- Interfaces con otros sistemas.
- Análisis de otras alternativas de solución.

Producto: Manual de diseño funcional

- Funciones del sistema.
- Estructura del sistema.
- Modelo de datos.
- Flujograma de información del área.
- Relaciones con otras áreas.
- Alternativas de solución.

Diseño computacional

Definir conceptos funcionales y técnicos que componen el sistema.

Actividades:

- Análisis detallado de funciones.
- Definir base de datos.
- Definir entradas y salidas de datos del sistema.
- Definir programas.
- definir utilización de recursos.
- Analizar contingencias versus backups y recuperaciones.

Producto: Manual técnico del sistema

- Arquitectura del sistema.
- Entidades de Base de Datos
- Diagrama Entidad-Relación

- Definición de módulos programables.
- Interfaces de Entrada/Salida.
- Rutinas de conversión de archivos del sistema anterior.
- Plan de contingencias versus backups y recuperaciones.

Programación y pruebas

Actividades:

- Definir esquemas de programación.
- Construcción de programas.
- Pruebas de programas.
- Pruebas del sistema.
- Ajuste de errores.

Producto: Manual de programación Y Programas:

- Indice de carpetas de programas.

Implantación

Actividades

- Conversión de archivos del sistema anterior.
- Entrega del sistema.
- Capacitación del usuario.
- Evaluación final.
- Definir plan de producción.

Producto: Manual de usuario

- Indicaciones sobre el uso de software.
- Relación de funciones y procesos del sistema.
- Navegación a través del sistema organizada por prioridades de proceso.
- Manual de producción
- Relación y descripción de procesos diarios, mensuales y anuales.
- Descripción del proceso de resguardo de información.
- Normalización del rotulado de cintas, discos y documentación del sistema.

Sección III. Planeamiento/Definición

Capítulo 4. Memorando de Alcance

4.1. Introducción descriptiva

El sistema de Distribución significa para Maquinarias el proceso de distribución de unidades vigilando la ubicación y el estado físico del vehículo hasta la entrega en perfectas condiciones a los Concesionarios autorizados (los clientes de la marca Nissan).

Para hacer posible el manejo de este concepto, en Maquinarias se tienen dos áreas:

Distribución: La administración de la venta de un vehículo.

Pre-Entrega: La ubicación física y revisión para que los vehículos salgan en perfectas condiciones a los clientes.

4.2. Antecedentes, objetivos y alcances del sistema

Antecedentes

El sistema de distribución actual está diseñado en FoxPlus y trabaja en el sistema operativo UNIX con las siguientes características:

- Sistema Operativo: UNIX
- Servidor: 486 DX2, 32 Mb RAM, 3 GB HD.
- Terminales: PC/XT con 1 Mb RAM como mínimo. Todos emulando sesión UNIX.
- El sistema estaba desarrollado en FOXPLUS para UNIX.
- Los terminales remotos se comunicaban vía modem con un módem local e ingresaban emulando un terminal UNIX.
- Conexión a puertos superseriales.

- El esquema de la red era el siguiente: Todas las actualizaciones se hacían en línea, también había consultas de usuarios remotos.

Objetivo

Se busca reemplazar por un sistema que esté diseñado en ambiente visual bajo Cliente/servidor preparado para consultas de nivel gerencial.

Límites

El sistema está limitado por las funciones de las áreas de Distribución y Pre-Entrega. Comienza después de la importación de los vehículos y termina con la entrega del vehículo al cliente (concesionario Nissan).

4.3. Diagnóstico del sistema actual

El sistema adolecía de un diseño consistente de base de datos. En el caso de caída de un terminal no se llegaba a completar el proceso de indexación y se perdían registros de la base datos.

Entre las funciones que cumplía falta la de controlar la ubicación del vehículo.

En algunos casos se tenían demasiados índices de acceso y se indexaba por partes (el Foxplus maneja como máximo 7 índices).

Como el sistema ha sufrido varias migraciones faltaba depurar los datos y prepararlos para el trabajo controlado.

La navegación a través de las opciones del sistema no sigue ningún estándar. Para poder ejecutar un proceso es necesario pasar por varios paneles de ingreso de datos.

En varias oportunidades se presentaban caídas de tensión que dañaban el disco duro y/o los datos.

4.4. Definición etapas y entregas

Las etapas para el desarrollo del sistema son las siguientes:

Diseño Conceptual. Entrega de manual de diseño conceptual.

Diseño computacional. Entrega de manual de diseño computacional.

Diseño computacional. Entrega de manual de diseño computacional.

Programación. Programación del software.

Implantación. Entrega del software.

4.5. Análisis Beneficio/Costo

Por ser información de la empresa, no se permite establecer el análisis Beneficio/Costo en cifras. Haremos la comparación cualitativa de la solución Cliente/Servidor frente a los costos de la solución existente.

Asignamos puntajes como sigue: 0=No tiene valor, 1=Solución de valor bajo, 2=Solución de valor medio, 3=Solución de valor alto.

Item	*Cliente/Servidor(C/S)*	*Solución Anterior(SA)*	*C/S*	*SA*
Servidor Datos	AS/400	486 DX4	**3**	**1**
Administrador de red	Windows NT	UNIX	**3**	**3**
Tipo de red	LAN	Red estrella	**3**	**2**
	Cableado estructurado	Cableado asíncrono		
	Concentradores ethernet	Concentradores asíncronos		
Base de Datos	DB2/400	Archivos DBF	**3**	**0**
Software base	ILE RPG/400	Foxplus	**2**	**1**
	Visual Basic			
Interface con usuario	Ambiente visual	Modo texto	**3**	**2**
Tipo programación	Interface ODBC	Programas simples	**1**	**3**
Requerimiento PC's	486 con 12 MB RAM	PC´s 286 con 2 MB RAM	**3**	**1**
Puntaje Promedio			**2,625**	**1,625**

Según la comparación cualitativa, Cliente/Servidor es mucho mayor la solución anterior, y los beneficios que se obtienen son mayores porque los costos en el tiempo van bajando.

Sección IV. Diseño conceptual

Capítulo 5. Análisis.

5.1. Funciones del sistema de Distribución

El sistema cumple con las siguientes funciones:

a. Ingreso de Vehículos al sistema. Creación de los registro que identifican a cada vehículo.

b. Control de Pagos de los vehículos. Se registra los pagos y se maneja los saldos por vehículo.

c. Control de la ubicación física del vehículo. Se hace un seguimiento de el lugar exacto donde se tiene el vehículo.

d. Resumen de información. Se hace un conjunto de reportes que tienen dos propósitos: Reportes para el trabajo cotidiano, Reportes de información gerencial.

e. Funciones de Pre-Entrega. Dar el estado exacto del vehículo, emitir la guía de remisión a la entrega del vehículo.

f. Consultas de Concesionarios. Los clientes de Maquinarias S.A. son los concesionarios, por lo que se creó un módulo de acceso a Concesionarios con las opciones de consulta de vehículos del concesionario y control de sus pagos.

5.2. Estructura del sistema.

El sistema está dividido en tres módulos:

Distribución

Involucra las siguientes funciones:

a. Ingreso de Vehículos al sistema.

b. Control de Pagos de los vehículos.

c. Control de la ubicación física del vehículo.

Tiene las siguientes características:

- Está programado en Visual Basic 3.0.
- Los datos están guardados en DB2/400 (en AS/400).
- Utiliza interface de programación ODBC.
- Accesa a los datos en forma local a través de Client/Access.

Pre-Entrega

Este módulo contiene las funciones de Pre-Entrega. Tiene las siguientes características:

- Está programado en Visual Basic 3.0.
- Los datos están guardados en DB2/400 (en AS/400).
- Utiliza interface de programación ODBC.
- Accesa a los datos en forma remota vía TCP con Client/Access.

Concesionarios

Absuelve las consultas de los concesionarios. Debido a que los concesionarios tienen limitados recursos de hardware y software, se ha desarrollado este módulo con las siguientes características:

- Está programado en ILE-RPG.
- Los datos están guardados en DB2/400 (en AS/400).
- Accesa vía módem con el protocolo TCP e ingresando con emulación de terminal 5250 (de AS/400).

5.3. Modelo de datos

Las principales entidades del modelo de datos son las siguientes:

Vehículo. Los vehículos entidad principal del sistema.

Pago. Pagos que hace un concesionario por un vehículo.

Modelo. Modelo y características de un vehículo.

Concesionario. Clientes que compran los vehículos.

Ubicación. El lugar donde se encuentra el vehículo. (Figura 8)

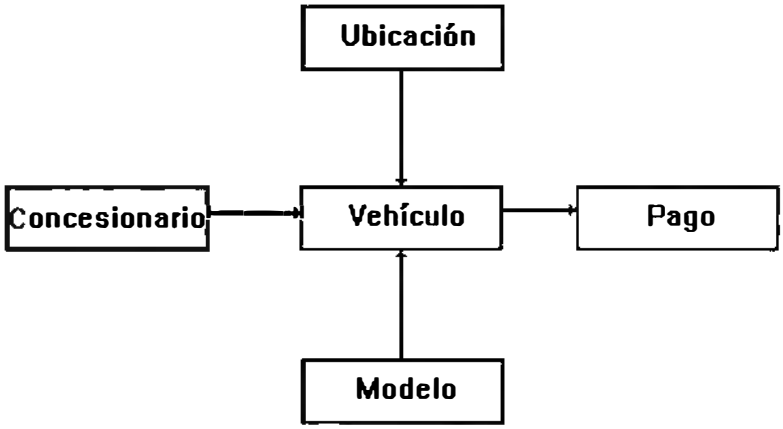

Figura 8. Modelo de datos.

5.4. Flujograma de información.

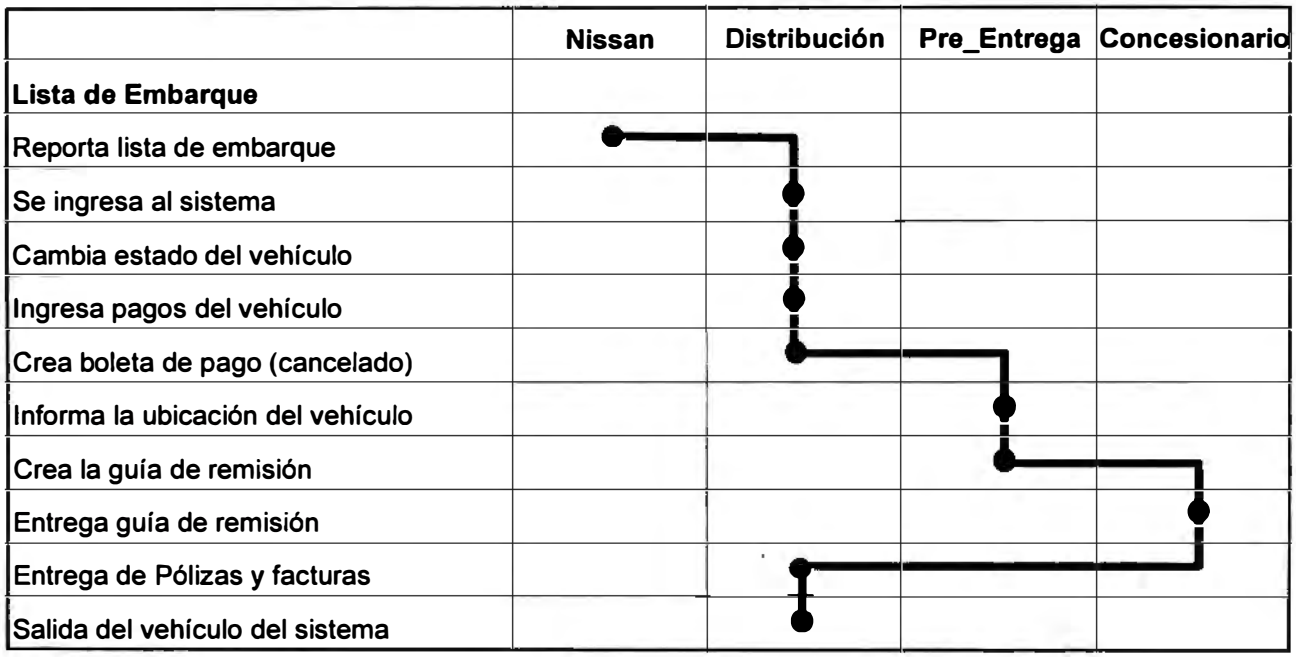

Figura 9. Flujograma de información.

5.5. Relaciones con otras áreas.

Nissan Motors S.A. Lista de embarque de los vehículos.

Concesionarios. Consulta de vehículos separados por un concesionario.

Servicios-Taller. Consulta de vehículos vendidos.

Relaciones Públicas. Consulta de vehículos vendidos.

Pre-Entrega S.A. Ubicación de los vehículos. (Figura 10)

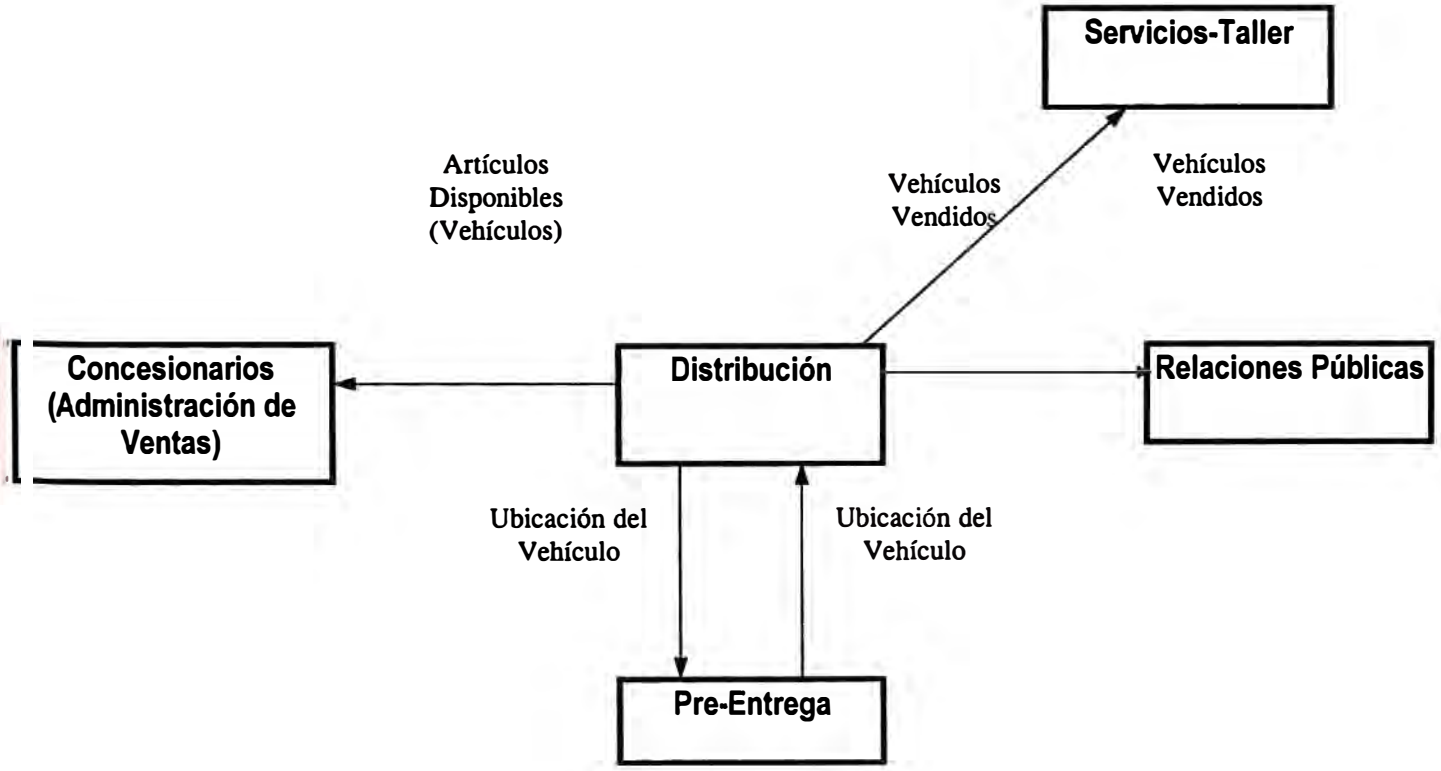

Figura 10. Relaciones del sistema con otras áreas

5.6. Alternativas de solución.

Se presentan varias alternativas de solución:

a. Rediseñar el sistema en ***UNIX*** y trabajarlo con un lenguaje como FOXPRO para mejorar la interface con el usuario.

Esta alternativa representaba un trabajo ordenado en programación con la inseguridad en los datos que ocasionaba.

b. Trabajar con ***AS/400 nativo***.

Se requiere un computador Advanced Series para poder soportar las sesiones interactivas en AS/400.

La performance de los sistemas es muy buena.

Se pierde todas las posibilidades de independencia en la manipulación de datos.

Los sistemas son demasiado rígidos en su presentación.

c. Trabajar con ***Cliente/Servidor con Programación visual con interface ODBC***.

Se requiere mayores recursos de Hardware y software para poder trabajar esta posibilidad.

Los sistemas son más lentos que en AS/400 nativo.

La programación está más ordenada por la utilización de la base de datos DB2/400.

Los sistemas son fáciles de aprender por que su presentación se puede manejar en un ambiente gráfico similar al de MS-Windows.

Se logra independencia en la programación de los productos AS/400. El mismo estilo de programación puede trabajarse para otros servidores con características similares de interface ODBC.

d. Utilizar cliente servidor con ***LANSA/400***

El uso de todo case hace más fácil la programación.

Los sistemas son sencillos pues para todas los diálogos y reportes se tiene el mismo estilo de presentación.

El LANSA/400 utiliza el SQL remoto para la programación con acceso de datos del DB2/400 y esto implica sujetarse a los estándares del AS/400.

Como todo case requiere gran cantidad de recursos para poder ejecutarse una aplicación con lo cual degrada la performance del sistema.

5.7. Tiempos.

Los tiempos para el desarrollo de las aplicaciones son:

Módulo	Sem.
Distribución	
Análisis:	8
Diseño:	6
Programación:	7
Instalación y pruebas:	2
Pre-Entrega	
Análisis:	8
Diseño:	2
Programación:	3
Instalación y pruebas:	2
Concesionarios	
Análisis:	8
Diseño:	2
Programación:	3
Instalación y pruebas:	2

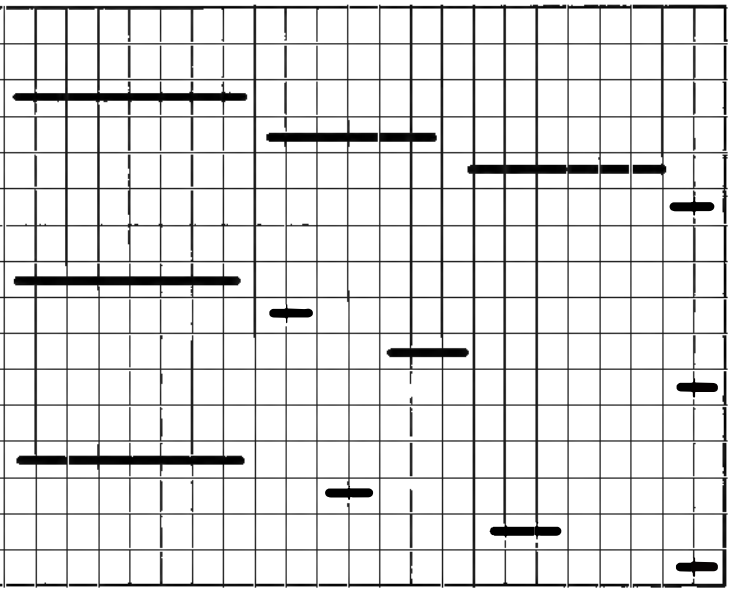

Figura 11. Diagrama de tiempos para el sistema de distribución.

El total de semanas del sistema se calculó en 23 (6 meses). La ruta crítica es el módulo de distribución.

Sección V. Diseño Computacional

¿Porqué se decide usar Cliente/Servidor con interface Gráfica?

Se decide usar Cliente/Servidor por las siguientes razones:

Debido al liderazgo que se tiene al utilizar una tecnología que se está desarrollando recientemente.

Se decide por la solución Visual Basic - AS/400 para ingresar a aplicaciones en ambiente gráfico y por la seguridad que brinda la base de datos DB2/400 (para IBM AS/400).

Los datos y procedimientos vitales del sistema estarán en la base de datos DB2/400 en el AS/400 y los programas estarán en ambiente visual programados con Visual Basic 3.0. Con lo cual las reglas del negocio se pueden controlar en el servidor y el cliente sólo tiene una máscara de información.

El manejo de las funciones del sistema y el proceso de la información se hace a través de herramientas de conectividad (ODBC) que permiten utilizar los datos del AS/400 procesándolos bajo Windows en un entorno gráfico y de fácil manejo.

Capítulo 6. El DB2/400

Se ha optado por una base de datos relacional por la seguridad de los datos.

La elección del AS/400 nos llevó a utilizar el DB2/400 base de datos relacional para AS/400.

El DB2/400 nos permite manejar:

- Restricciones,
- Integridad referencial,
- Triggers o activadores,
- Stored Procedures,

6.1. Restricciones

Las restricciones que soporta el DB2/400 son las siguientes:

- Restricción de Unicidad
- Restricción de Clave Primaria

6.2. Integridad Referencial y Restricciones Referenciales

La integridad referencial es un término amplio que abarca todos los mecanismos y técnicas que aseguran la validez de los datos referenciados en una base de datos relacional.

Reglas de las Restricciones

Reglas de Supresión

La reglas de supresión especifican la acción a llevar a cabo cuando se suprime un valor de clave padre.

*CASCADE

*SETNULL

*SETDFT

*RESTRICT

Reglas de Actualización

La regla de actualización especifica la acción llevada a cabo cuando se intenta la actualización del archivo padre.

*NOACTION

*RESTRICT

6.3. Activadores o Triggers en DB2/400

Un activador es un conjunto de acciones que se ejecutan automáticamente cuando se efectúa una operación de cambio especificada en un archivo físico de base de datos especificado (inserción, actualización o supresión) en un programa de aplicación.

- Asociación de un trigger a un archivo

  Puede asociar un máximo de seis activadores a un archivo físico, uno de cada uno de los siguientes:

  Antes de una inserción

  Después de una inserción

  Antes de una supresión

  Después de una supresión

  Antes de una actualización

  Después de una actualización

6.3.2. Funciones en el Trigger

Actualizar o consultar otras tablas si queremos.

Dentro de un trigger se puede declarar archivos hacer operaciones de cálculo y todo lo que está permitido a un trabajo en batch.

Una vez que el trigger se ha ejecutado, no envía ningún mensaje al programa que hizo la actualización de la tabla. Si el trigger tuvo un error solamente se detecta el error como si se hubiera realizado mal la actualización de la tabla asociada.

6.4. ¿Que son Stored Procedures?

Los Stored Procedures o Procedimientos Almacenados son programas que realizan actualizaciones a las tablas de la Base de Datos y/o cálculos específicos con la posibilidad de manejar parámetros de entrada y salida.

Estos Stored Procedures se ubican en el servidor y pueden ser ejecutados desde cualquier cliente en cualquier punto. El error se puede controlar por los valores de los parámetros de retorno que se tiene en el programa Cliente.

6.4.1. Funciones del Stored Procedure

Cuando se ejecutan sentencias SQL desde un leguaje cliente a través de ODBC o SQL remoto, el sistema se encarga de preparar la sentencia enviarla y luego recibir los resultados, esto tiene un tiempo de demora bastante alto (para nuestra aplicación aprox. 5 segundos) por cada actualización de una tabla.

Para rutinas en que se actualizan varias tablas se puede reemplazar todas estas actualizaciones con un Stored Procedure que representa una sola preparación y ejecución de una sentencia SQL (la de llamada al Stored Procedure).

Otra razón es porque tenemos el Stored Procedure ubicado en servidor y si queremos alterar cualquier secuencia de actualización y o de proceso de cálculo complejo, sólo tenemos que actualizar el Stored Procedure en el servidor y automáticamente lo utilizan los clientes sin necesitar de actualizar el software instalado en cada cliente.

Capítulo 7. Componentes de Hardware y software

7.1 La Red

Nuestra solución Cliente/Servidor está compuesta de lo siguiente:

- Una red LAN con Windows NT como administrador,
- Un AS/400 40S como Servidor de Datos,
- Computadoras personales con Windows for Workgroups 3.11 como Clientes.

Si revisamos el siguiente gráfico (Figura 12) tenemos la estructura básica de la red:

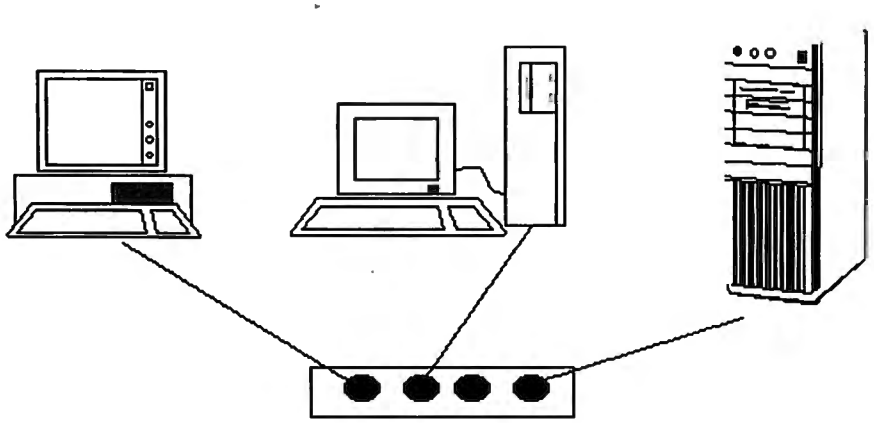

Figura 12. Diagrama de la red.

Cliente	*Red*	*Servidor*
Hardware		
486 DX2 66 Mhz	**460 DX2 100 Mhz**	**Power PC**
8 MB RAM	**80 Mb RAM**	**96 MB RAM**
Monitor VGA	**Monitor VGA**	
100 MB HD Libres	**2 GB HD**	**7,6 GB HD**
10 Base-T Red	**10 Base-T**	**Ethernet**
Salida UTP	**Salida UTP**	**Salida UTP**
Conexión a la red	**Conexión a la red**	**Conexión a la red**

Software		
Windows 3.1	**Windows NT 3.0**	**OS/400**
Windows for Workgroups 3.11	**Microsoft DLC**	**DB2/400**
Client Access/400 3.1		**SQL/400**
Microsoft DLC		**Client Access 3.1**
ODBC 1.0		**ILE-RPG/400**
Aplicación		

Software necesario en un cliente de Desarrollo:

Windows 3.1 (Sistema Operativo)
Windows for Workgroups 3.11 (Acceso a Red)
Microsoft DLC (Protocolo)
ODBC 1.0 (Interface datos)
Visual Basic 3.0 (Lenguaje programación)
Crystal Report 4.0 (Generador consultas)
Client Access/400 3.1 (Administrador de comunicaciones PC-AS/400)
Erwin 2.0 (Case para generear la base de datos)
Rumba 400 (Emulación terminal AS/400)
Aplicación (Software desarrollado por Sistemas)

7.2. Esquema de comunicación remota (Adicional a la red)

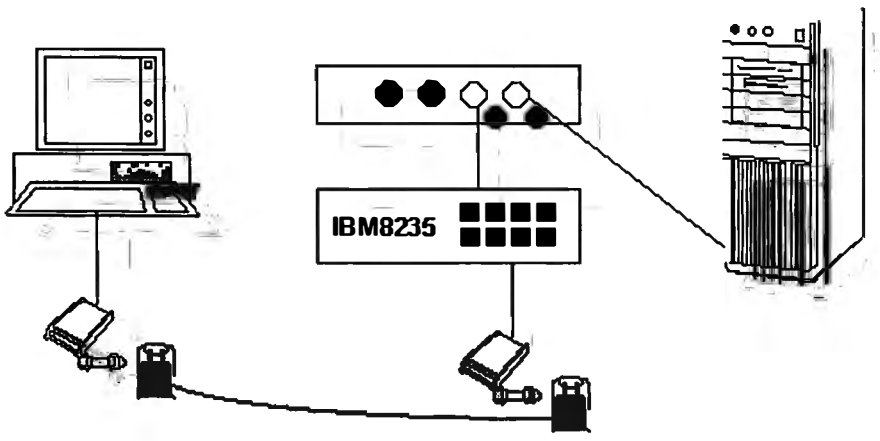

Figura 13. Comunicación remota

Cliente	*Red*	*Servidor*
Hardware		
PC XT 2 Mb RAM	**IBM-8235**	**Ninguno**
Modem	**Modem**	
Línea telefónica	**Línea Telefónica**	
Conexión Línea	**Conexión Línea**	
Software		
(*) Ver 7.3	**[1] IBM-8235**	**TCP/IP**

[1] El IBM-8235 Es un negociador de comunicaciones que además administra el acceso de los módem de la red al AS/400.

7.3. Software de un Cliente remoto con TCP/IP(*)

Con el sistema en Cliente/Servidor

Windows 3.1
Windows for Workgroups 3.11
Microsoft DLC
TCP/IP Controlador
Dial UP (IBM8235)
ODBC 1.0
Client Access/400 3.1

Con el sistema en emulación de terminal AS/400 (5250)

Windows 3.1
Windows for Workgroups 3.11
Microsoft DLC
TCP/IP Controlador y Emulador
Dial UP (IBM8235)

Capítulo 8. El Client Access/400

El Client Access/400 está instalado tanto en el AS/400 como en la computadora que va ha los datos desde un terminal en modo Cliente/Servidor.

En el AS/400 el Client Access/400 Trabaja contra el microcódigo (Núcleo del sistema Operativo OS/400) a través del Host Server (Componente del sistema operativo).

En el Cliente se instala el Client Access/400 para la plataforma requerida (Windows 3.1, DOS u OS/2).

Al instalar el Client Access/400 en el cliente también se toma parte del software que está localizado en AS/400 con lo que completa la compatibilidad de acceso.

8.1. Componentes del Client Access/400

Messaging Application Program Interface (MAPI)

Open Database Connectivity (ODBC)

Direccionador de comunicaciones APPC

Servidor de colas de datos DDE

Transformación de datos

Especificación de interfaz de controlador de red (NDIS)

Unidades de red (Carpetas compartidas).

Redireccionador de red.

Servicio de operador de nodos (NOF).

SQL remoto.

Transferencia de archivos.

Someter mandato remoto.

Impresora virtual.

Capítulo 9. Comunicación con puntos remotos

El sistema tiene usuarios remotos los cuales ingresan al sistema a través de:

Sesión Cliente/Servidor y

Emulación de terminal.

9.1. Sesión Cliente/Servidor en AS/400

Para que un cliente remoto ingrese a una sesión Cliente/Servidor es necesario:

9.1.1. En el servidor

Definir la dirección IP del usuario en TCP/IP en AS/400.

Se ejecuta el comando ADDTCPHTE (añadir entrada de tabla de sistema principal) el cual pide la dirección IP y el nombre del sistema remoto.

Se cambia la lista de configuración (Figura 14):

```
CHGCFGL TYPE(*APPNRMT)

Cambiar Lista de Configuración                MAQUI02
                                                        18/11/96  09:37:32
 Lista de configuración  :   QAPPNRMT
 Tipo de lista de
   configuración . . . . :   *APPNRMT
 Texto . . . . . . . . . :
 Teclee cambios, pulse Intro.

 -------------------------Ubicaciones Remotas APPN-------------------------------
 Ubica-     ID         Ubica-     Punto      ID Red
 ción       Red        ción       Control    Punto      Contraseña         Ubicac
 Remota     Remota     Local      Remoto     Control    Ubicación          Proteg
 PREENTRE   APPN       MAQUI02               APPN                           *NO
 CAUTOS     APPN       MAQUI02               APPN                           *NO
            *NETATR    *NETATR               *NETATR                        *NO
            *NETATR    *NETATR               *NETATR                        *NO
Más...
 F3=Salir         F11=Visualizar información de sesión   F12=Cancelar
 F17=Principio    F18=Final
-------------------------------------------------------------------------------
```

Figura 14. Cambiar listas de configuración

Una vez creada la dirección IP se crea el controlador manualmente (CRTCTLAPPC) que pide el nombre del controlador, nombre del sistema remoto, el nombre del recurso de red:

```
CRTCTLAPPC
CTLD(REMOTO)          Descripción controlador
LINKTYPE(APPN)        Tipo de enlace
ONLINE(*YES)          En línea al dar IPL
```

Se activa el controlador (VRYCFG) y se ingresa el nombre del controlador (creado con CRTCTLAPPC y ONLINE(*YES)).

9.1.2. En el cliente

Al definir el Modo de acceso en Client Access se debe especificar TCP/IP (opciones comunes de configuración de sistemas en Client Access/400).

9.1.3. Comunicación con el servidor

Para comunicarse con el servidor se deben seguir los siguientes pasos:

- Ejecutar el Dial Up en el cliente, lo que equivale a llamar al IBM-8235. Internamente el IBM-8235 hace un telnet al AS/400. El usuario que hace su ingreso al IBM-8235 debe estar previamente registrado.
- Al conectarse al AS/400 el controlador está activo (con el IPL se crean las sesiones de conexión con modo de descripción TCP)
- Abrir la aplicación (Se activa el dispositivo y se crean las sesiones con modo de descripción QSERVER)

9.2. Emulación de terminal AS/400

Para que un usuario remoto emule un terminal de AS/400 se deben tener definidas:

- Una dirección IP en AS/400.
- Una cola de impresión remota en AS/400 (Comando CRTOUTQ) conectada a la dirección IP respectiva.

CRTOUTQ	
OUTQ(CCCC)	Cola de salida
RMTSYS(CALEJANDRI)	Nombre del sistema remoto
RMTPRTQ(CALEJANDRI)	Nombre de cola en el sistema remoto
AUTOSTRWTR(1)	Número de sesiones de impresión automática
CNNTYPE(*IP)	Tipo de conexión
DESTTYPE(*OTHER)	Tipo de destino (cualquier otra computadora)
MFRTYPMDL(*EPLX810)	Tipo de impresora remota

Para emular un terminal AS/400 debe seguir los siguientes pasos:

- Ejecutar el Dial Up en el cliente, lo que equivale a llamar al IBM-8235. Internamente el IBM-8235 hace un telnet al AS/400, que ya se ha configurado previamente. El usuario que hace su ingreso al IBM-8235 debe estar previamente registrado por este.
- Activar el LPD para la cola de impresión.
- Ejecutar el software de emulación de terminal.
- Abrir la sesión de emulación. Abre una sesión de usuario AS/400.

Capítulo 10. Diseño del Sistema De Distribución

10.1. Arquitectura del sistema.

El sistema de Distribución trabaja bajo dos plataformas:

a. El módulo central de Distribución

- La base de Datos está en DB2/400
- Triggers y Stored Procedures codificados en ILE RPG/400
- Programas del cliente codificados en Visual Basic 3.0.
- Acceso al sistema a través de Client Access/400
- Acceso a los datos a través de ODBC nivel 2
- El manejo de la base de datos desde el cliente se ha programado usando Visual Basic 3.0 con interface ODBC.
- Se ha previsto que los clientes trabajen bajo Windows for workgroups 3.11.

b. El módulo de Pre-Entrega

- Base de datos en DB2/400.
- Los programas están codificados en Visual Basic 3.0
- El acceso a los datos se hace vía TCP.

c. El módulo para concesionarios

- Todos los programas están codificados en ILE RPG/400
- El acceso al sistema se hace a través de TCP como emulación de terminal.

10.2. Diseño de la Base de Datos.

Lista de Entidades de Base de Datos (Archivos)

Modelo. Tabla donde se encuentran todos lo modelos de vehículos disponibles.

Color. Tabla de colores de vehículos.

Banco. Tabla de Bancos.

Modelo_Dato_Tecnico. Especificaciones técnicas de cada modelo.

Lista_Embarque. Identificación del lote de vehículos que vienen juntos.

Vehiculo. Identificación de cada unidad (vehículo).

Pago. Archivo donde se guardan todos los pagos que se realizan para cancelar un vehículo.

Ubicacion_Vehiculo. El lugar físico donde se encuentra un vehículo.

Concesionario. El registro de los Concesionarios (clientes de Nissan Maquinarias).

10.3. Diagrama Entidad Relación

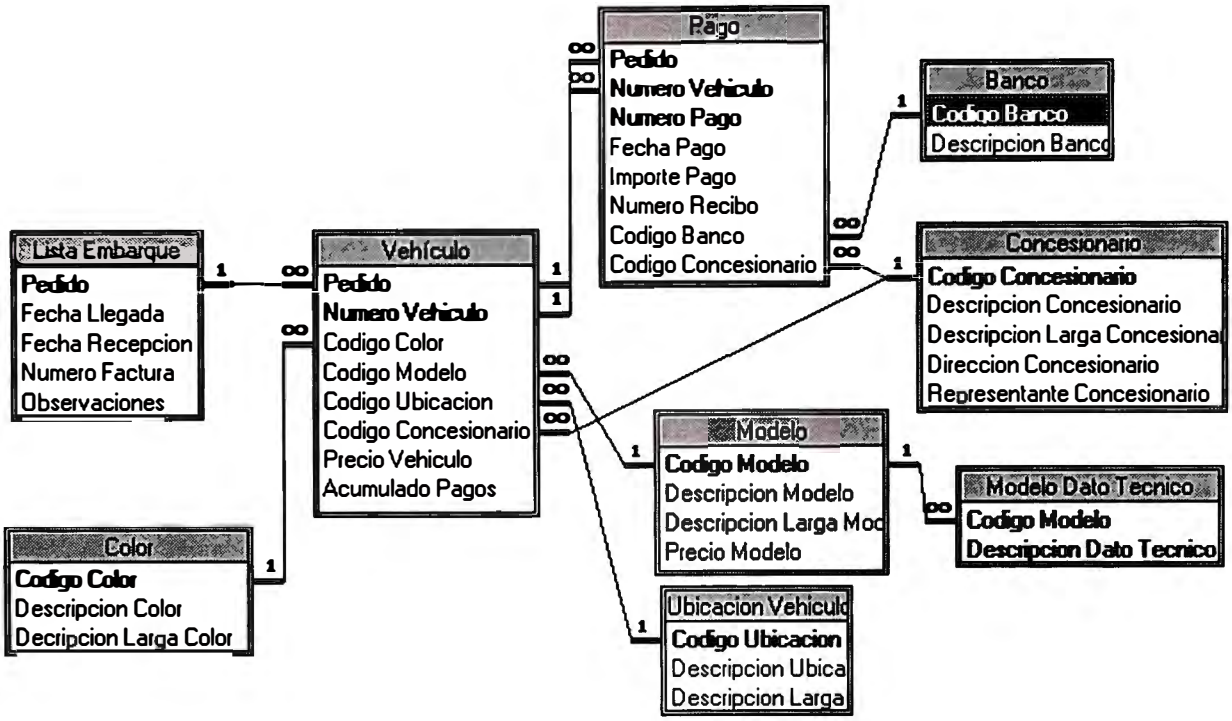

Figura 15. Diagrama Entidad-Relación.

10.4. Módulos programables.

10.4.1. Módulo central de Distribución

a. De actualización de información:

a.1. Mantenimiento de Tablas:

Modelo. Actualiza tabla de modelos.

Color. Actualiza tabla de colores.

Banco. Actualiza tabla de bancos.

Concesionario. Actualiza tabla de concesionarios.

Ubicación Vehiculo. Actualiza tabla de ubicaciones de vehículo.

a.2. Ingreso tipo cabecera detalle:

Actualización de Datos Técnicos del Modelo. Es llamado por la actualización del Modelo para poder ingresar el detalle de sus Datos técnicos.

Actualización de Lista de Embarque. Actualiza como cabecera Lista_Embarque y como detalle Vehículo.

Actualización de Pagos. Registra cada pago que se hace por un vehículo. Al final de los pagos emite automáticamente la boleta de pago.

a.3. Cambios de datos:

Separación de Unidades.

Cambio de Ubicación del vehículo.

b. Consultas y Reportes

Lista de Precios.

Unidades Libres.

Unidades Separadas.

Unidades Canceladas.

Cuentas Pendientes por Concesionario.

Histórico de Ventas.

c. Triggers y Stored Procedures

Trigger de actualización de saldos de pagos.

Actualización de Pagos.

10.4.2. Del módulo de Pre-Entrega

a. Seguimiento de ubicación de la unidad

Unidades ingresadas

Unidades retiradas

Entrega de unidades a los concesionarios

b. Vigilancia del estado de la unidad

Unidades en óptimo estado.

Dictado de la unidad.

10.4.3. Del módulo de Concesionarios

a. Consultas

Consulta de Vehículos ordenado por Lista de Embarque.

Consulta de Vehículos ordenado por Modelo.

Consulta de Pagos por Vehículos.

b. Reportes

Reporte de Unidades Libres.

Reporte de Unidades Separadas.

Reporte de Unidades Canceladas.

10.5. Interfaces de entrada salida .

10.5.1. Módulo central de Distribución

Menú del Sistema.

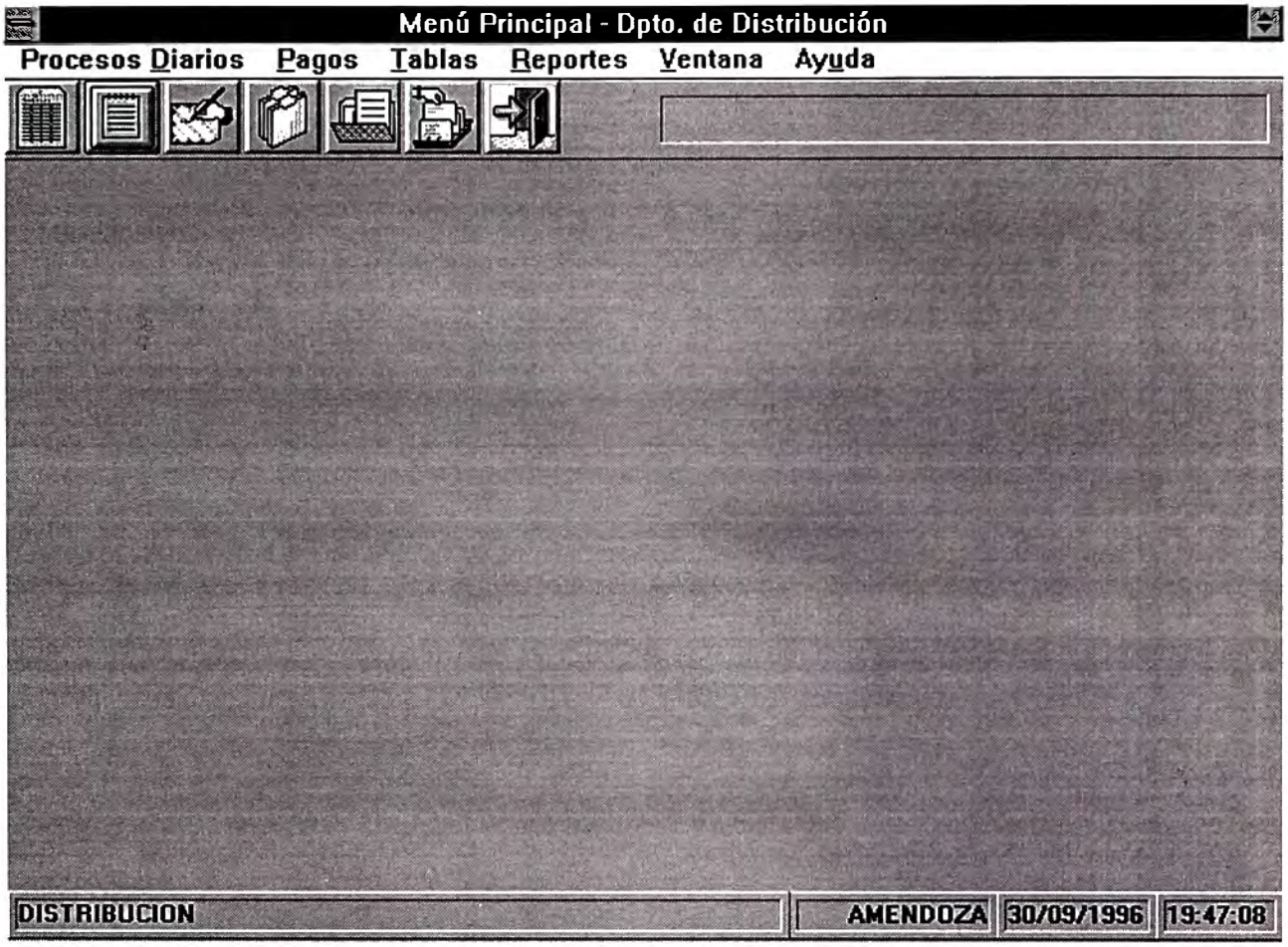

Figura 16. Menú del sistema

De Actualización de tablas.

Los programas de actualización tienen una primera ventana (window) que muestra:

a. Las opciones de proceso, representados por Botones con iconos estándares para las diferentes opciones de actualización.

Nuevo = Ingresar registros a la tabla.

Consulta = Consultar registros de la tabla.

Modificar = Modificar registros de la tabla.

Eliminar = Eliminar registros de la tabla.

Imprimir = Imprimir lista de registros de la tabla.

Salir = Dejar el programa.

b. Una franja donde aparecen todos los registros de la tabla. (Figura 17)

Ejemplo para tabla de colores

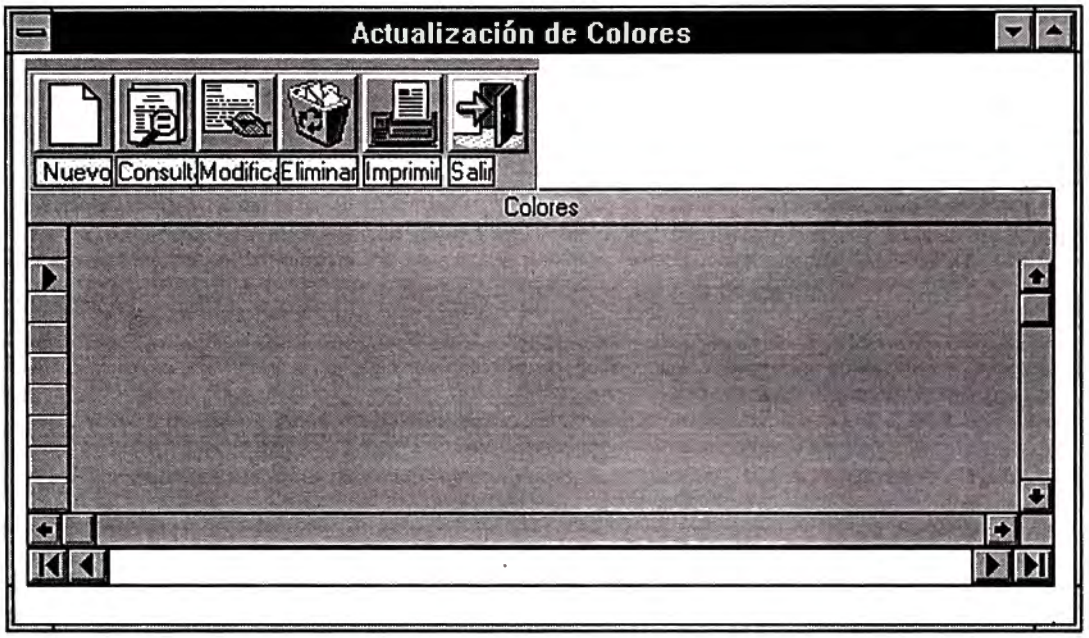

Figura 17. Ejemplo de programa de actualización de tabla.

Al presionar los botones de ícono aparece la ventana de actualización donde tenemos:

a. Los campos de la tabla.

b. Los botones de icono con la confirmación o cancelación de la actualización.

Por ejemplo:

Para la opción de ingresar aparece el panel de actualización con los datos en blanco, listos para recibir datos. (Figura 18)

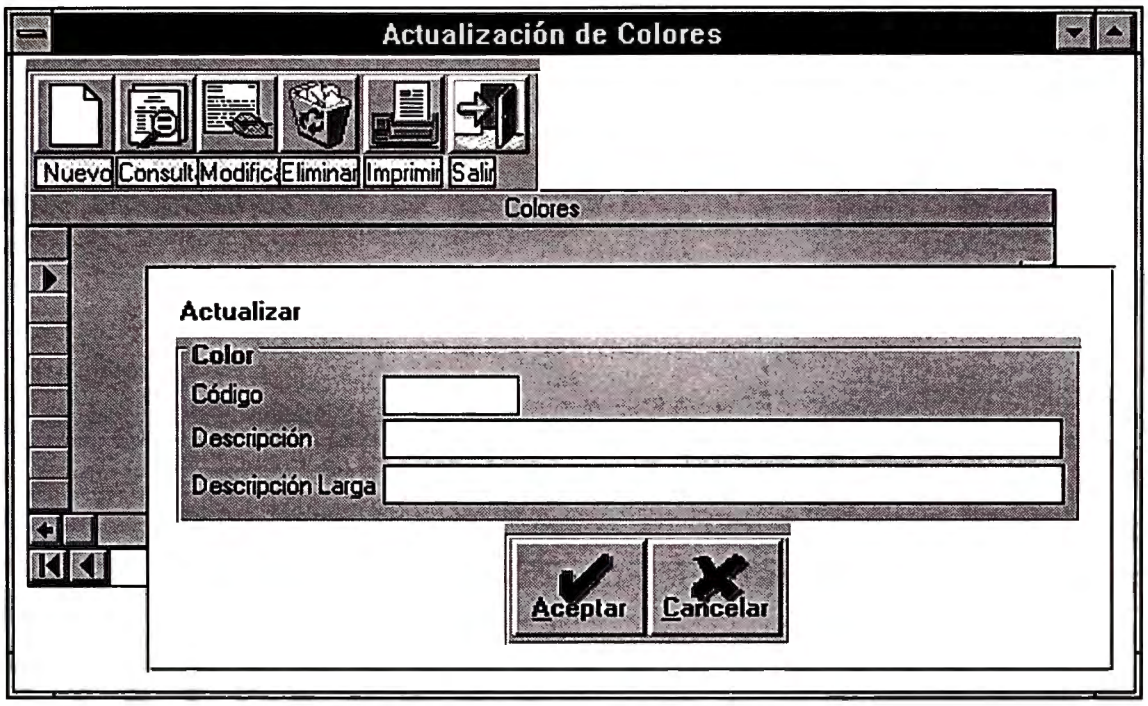

Figura 18.

Para confirmar la actualización se presiona el botón Aceptar, en caso contrario se presiona el botón Cancelar. En ambos casos el control retorna a la ventana anterior.

De Ingreso tipo Cabecera-Detalle.

Los programas de ingreso tipo Cabecera-Detalle tienen las siguientes áreas:

a. Un grupo de campos de ingreso (en la parte superior de la ventana) que es necesario para identificar a la cabecera.

b. Una franja que corresponde a la lista de los ítems de detalle asociados a la cabecera.

c. Un grupo de botones con las opciones de la ventana:

Nuevo = Ingresar nuevo vehículo a la lista.

Modificar = Modificar vehículo a la lista.

Eliminar = Eliminar vehículo a la lista.

Imprimir = Imprimir vehículos de la lista.

Aceptar = Grabar los cambios hechos (actualización de Lista de Embarque y los vehículos)

Cancelar = Salir del programa sin grabar cambios. (Figura 19)

Como ejemplo tenemos el ingreso de Lista de Embarque.

Este programa ingresa la Lista de Embarque como cabecera y los vehículos como detalle.

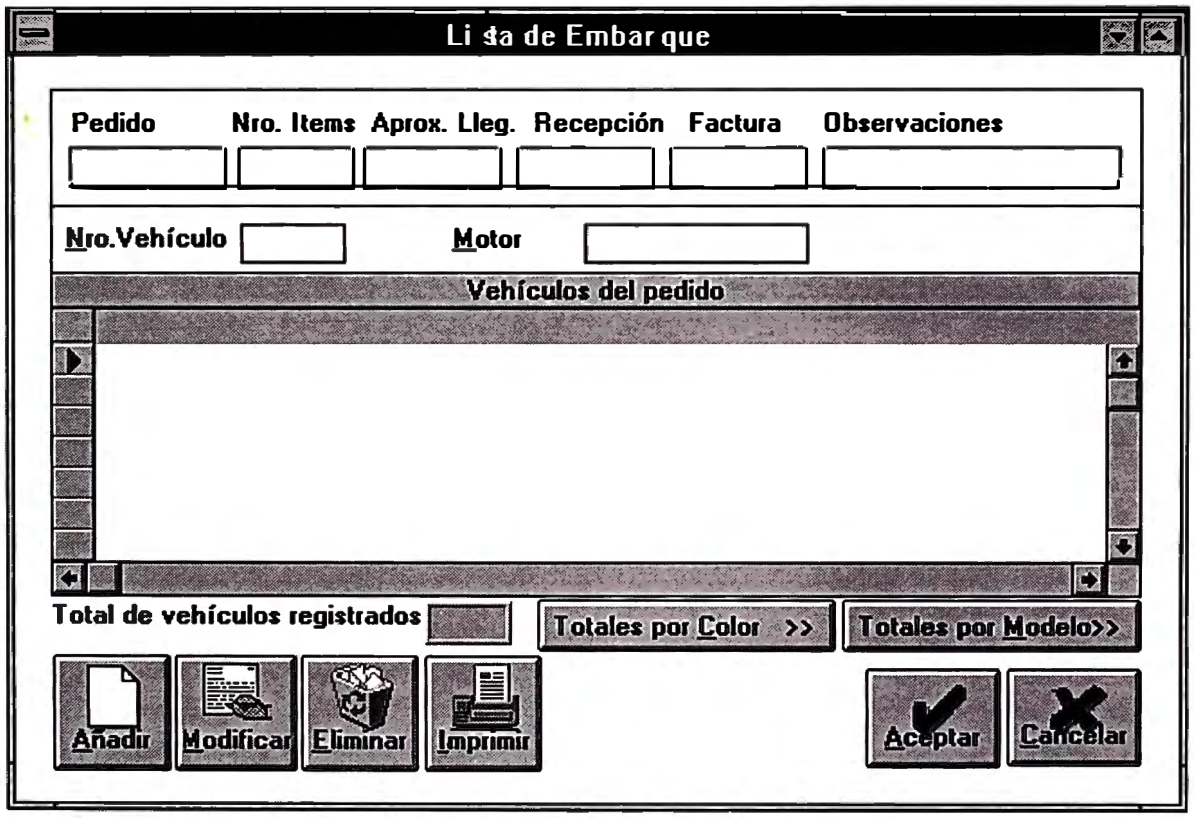

Figura 19. Ejemplo de Cabecera-Detalle.

Al presionar los botones Añadir, Modificar, Eliminar se presenta el siguiente panel que permite actualizar los datos (Figura 20):

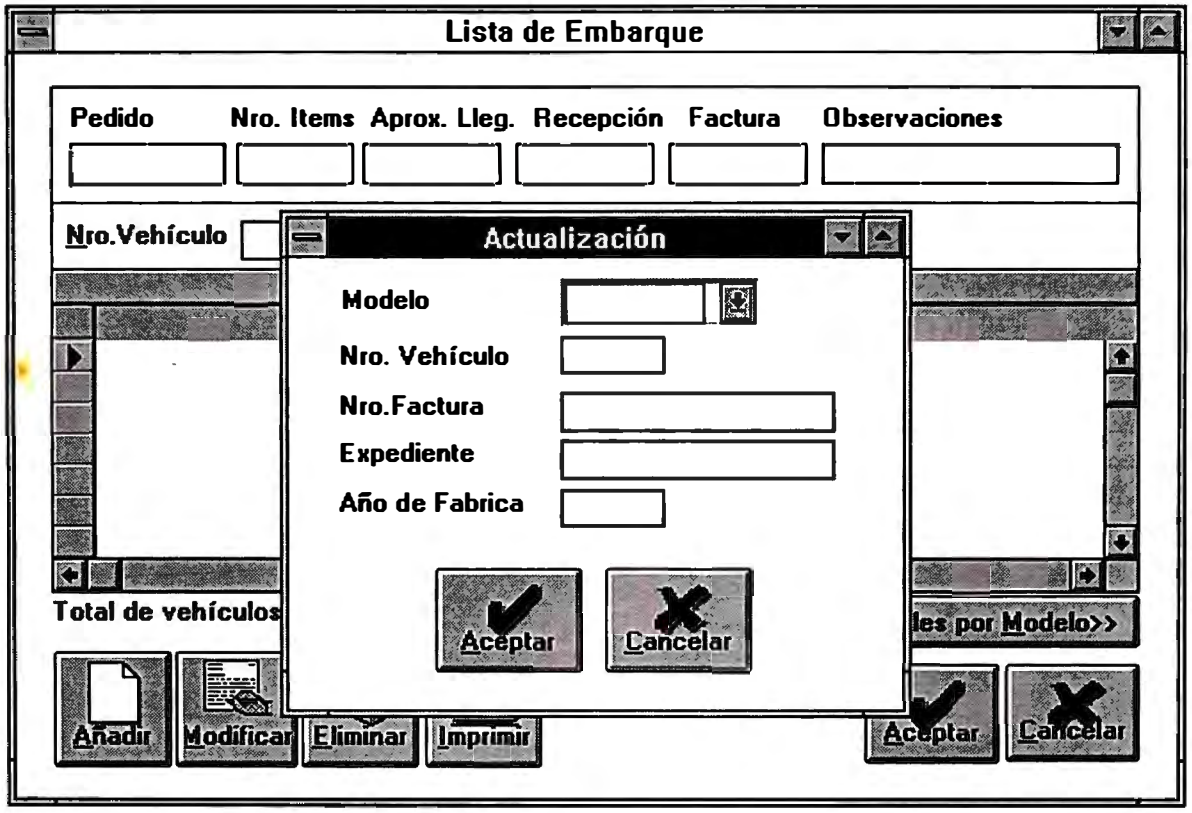

Figura 20.

Para confirmar la actualización se presiona el botón Aceptar, en otro caso se presiona el botón Cancelar. En ambos casos el control retorna a la ventana anterior.

10.5.2. Para el módulo de Concesionarios

Todos los programas están desarrollados en ILE RPG/400. Los programas tipo son:

Para el Menú

```
CONPRU                          Menú de Concesionarios

 Seleccione una de las siguientes opciones

      1.  Consulta de Vehículos
      2.  Consulta de Vehículos por Clave
      3.
      4.
      5.
      6.  Reporte de Unidades Libres
      7.  Reporte de Pendientes de Pago
      8.  Reporte de Unidades Canceladas
      9.
     10.  Salir del Sistema

 Selección o mandato

 F3=Salir    F12=Cancelar
```

Figura 21. Ejemplo e menú en ILE RPG/400

Para una consulta:

```
Sistema de Distribución                                          96/09/30
Atención a Concesionarios                                        19:51:03
                      Consulta el Maestro de Vehículos

 Buscar

Opc: 5=Detalle Vehículo    8=Ver Pagos
 Opc  Modelo    Pedido  #      Color  Año  Saldo Pag
      SENTRA    103     00002 BLK01  0000 14,010.00
      SENTAU    105     00096 KN04   1996
      SENTAU    105     00097 QM101  1996
      SENTAU    105     00099 KN04   1996
      SENTAU    105     00100 FN04   1996
      SENTAU    105     00101 ALO01  1996
      SENTAU    105     00102 QM10   1996
      SENTAU    105     00103 FN10   1996
      SENTAU    105     00104 ALO01  1996
                                                                 Más...
 Enter=Procesar, F3=Salir, F12=Cancelar, Av Pág, Re Pág
```

Figura 22. Ejemplo e menú en ILE RPG/400

10.6. Rutinas de conversión de archivos del sistema anterior.

Para convertir los archivos del sistema anterior se estableció el siguiente proceso:

Pasar los archivos de UNIX a DOS a través de FTP.

Crear una imagen de las bases de datos de AS/400 en DOS.

Procesar con FOXPRO la información que viene de UNIX hacia las imágenes de las bases de datos en PC.

Creación de listas de embarque.

Creación de Artículos.

Creación de pagos de cada unidad.

Transferir las bases de datos al AS/400.

10.7. Plan de contingencias, backups y recuperaciones.

10.7.1. BACKUPS

El Backup correspondiente a la información del servidor se realizará *diario*, para prevenir los riesgos de pérdida de información, en casos de:

Caída de tensión.

Fallas en el disco.

Fallas de hardware.

Se tiene una cinta rotulada con el nombre del día de la semana:

```
Lunes
Martes
Miércoles
Jueves
Viernes
Sábado
Domingo
```

Al final del día de trabajo (7:30 pm) se pedirá a todos los usuarios que abandonen sus terminales.

Se ubicará la cinta del día en la unidad de Tape Backup.

Se dará formato a la cinta con el siguiente comando:

```
INZTAP DEV(TAP01)
```

El backup se debe realizar con el comando:

```
SAVLIB
LIB(MDISDAT)
DEV(TAP01)
```

el comando pedirá cuantas cintas necesite.

Se procederá a rotular la cinta de la siguiente manera:

```
Nombre:      Lunes
Fecha:       02/12/96
Librería:    MDISDAT
```

El backup del día sábado se hará por duplicado. Una copia se queda en la oficina y la segunda va a un local externo.

El último día del mes se efectúan tres backups. La primera copia se queda en la oficina, la segunda va a un local externo y la tercera se almacena como la cinta del mes.

10.7.2 Recuperación

El proceso de recuperación de datos se hará de la siguiente manera:

Se buscará la cinta que tenga la información a ser recuperada.

Se ubicara la cinta en la unidad y se ejecuta el comando:

```
RSTLIB
SAVLIB (MCOMDES)
DEV(TAP01)
```

Una vez terminado el proceso se guardará la cinta.

Sección VI. Programación

Capítulo 11. Esquemas de programación

11.1. Apertura de sesiones AS/400

En cada aplicación que se desee usar un Data Source se debe abrir una sesión para este Data Source.

Esta apertura de sesión se debe hacer al empezar la ejecución el sistema.

11.1.1 A través una aplicación Visual Basic 3.0.

En Visual Basic el comando es:

```
SET GBASEDATOS = OPENDATABASE(" ", False, False,
                              ODBC;DSN=MCOMDAT;UID=VMACOMSA;PWD=PWDMACOMSA")
```

Donde se crea un acceso llamado GBASEDATOS (Con el que se reconocerá dentro de Visual Basic).

Este a su vez abre la base de datos (OPENDATABASE)

La base de datos que abre tiene la ruta

"ODBC;DSN=MCOMDAT;UID=VMACOMSA;PWD=PWDMACOMSA"

Se reconoce que es una ruta de ODBC,

El nombre del Data Source (DSN) es MCOMDAT,

El nombre de usuario de acceso (UID) es VMACOMSA, usuario creado en AS/400,

El password de usuario (PWD) es PWDMACOMSA, password de VMACOMSA en AS/400.

11.1.2. A través de una aplicación Visual Basic 3.0 con ODBC.

A través de funciones ODBC (Preparada para interface de programación ODBC) el comando es:

```
IRC = LOGONVIAODBC("MCOMDAT", "VMACOMSA", "PWDMACOMSA")
```

Se crea un acceso a través de interfaces de programación ODBC.

Donde MCOMDAT es el nombre del Data Source,

El nombre de usuario de acceso (UID) es VMACOMSA, usuario creado en AS/400,

El password de usuario (PWD) es PWDMACOMSA, password de VMACOMSA en AS/400.

11.2. Manejo de base de datos

11.2.1. Manejo de base de datos con sentencias Visual Basic.

Para manejar la base de datos a través de sentencias Visual Basic es necesario abrir una sesión:

```
SET GBASEDATOS = OPENDATABASE(" ", False, False,
                                    "ODBC;DSN=MCOMDAT;UID=VMACOMSA;PWD=PWDMACOMSA")
```

11.2.1.1. Recuperar datos del DB2/400

Las sentencias SQL de definen como una cadena de caracteres que interpreta el SQL del AS-400.

Por ejemplo:

```
"SELECT * FROM MODELO"
```

Luego se ejecutan dichas sentencias con la siguiente sentencia:

```
set GSNP = GBASEDATOS.CREATESNAPSHOT("SELECT * FROM MODELO", 64)
```

Donde GSNP es el arreglo de variables que va a contener los datos recuperados.

La manipulación de los campos que se recibe después del comando se hace con:

```
CMODELO = GSNP.("CODIGO_MODELO")
```

Donde CMODELO es una variable de visual basic y CODIGO MODELO es el nombre del campo en DB2/400.

La manipulación de las variables entonces es sencilla.

11.2.1.2. Actualizar datos en DB2/400

Cuando se trata de actualizar datos en DB2/400 la sentencia se trata como una cadena de caracteres:

Por ejemplo:

```
"INSERT INTO MODELO CODIGO_MODELO, DESCRIPCION_MODELO VALUES ('SENTRA', 'SENTRA AUSTERO')"
```

La ejecución de la sentencia se realiza con el comando:

```
RECSINS = GBASEDATOS.EXECUTESQL("INSERT INTO MODELO CODIGO_MODELO, DESCRIPCION_MODELO
                                 VALUES ('SENTRA', 'SENTRA AUSTERO')")
```

Donde RECINS es una variable numérica que contiene la cantidad de registros procesados.

el manejo de variables es también sencillo pero no hay la posibilidad de manejar parámetros.

Para ejecutar un Stored Procedure, la sentencia es la siguiente:

```
RECSINS = GBASEDATOS.EXECUTESQL("CALL SP_INS ('SENTRA','SENTRA AUSTERO')")
```

No existe la posibilidad de saber como ejecutó el Stored Procedure ni en qué punto se presentó un problema.

Este manejo de la base de datos es sencillo si se tiene bien controlado, la programación debe ser muy precisa, pero no se tiene el control total del sistema.

Además se tiene en contra la lentitud del proceso en tiempo de ejecución, tarda entre 5 y 10 segundos el tiempo respuesta por ejecución de sentencia.

11.2.2. Manejo de base de datos con interface ODBC.

Para manejar la base de datos con interface de programación ODBC es necesario abrir la sesión ODBC:

```
IRC = LOGONVIAODBC("MCOMDAT", "VMACOMSA", "PWDMACOMSA")
```

11.2.2.1. Recuperar datos de DB2/400

Primero tenemos que crear la cadena de la sentencia:

Por ejemplo:

```
Dim stmtModelo As String
stmtModelo = "SELECT CODIGO_MODELO, DESCRIPCION_MODELO"
stmtModelo = stmtModelo + " FROM MODELO"
stmtModelo = stmtModelo + " WHERE  CODIGO_MODELO = ?"
```

Luego creamos la dirección de memoria para los parámetros de la sentencia:

```
IRC = SQLAllocStmt(ghdbc, hstmtModelo)
```

Donde hstmtModelo almacenará la dirección

Preparamos la sentencia, asignamos la sentencia a la dirección de memoria:

```
IRC = SQLPrepare(hstmtModelo, stmtModelo, SQL_NTS)
```

11.2.2.2. Actualizar datos en DB2/400

Declaramos primero el procedimiento y creamos la estructura de los parámetros del Stored Procedure:

```
Dim WCODIGO_MODELO AS STRING *6
Dim WDESRIPCION_MODELO AS STRING *20
Dim WESTADO AS STRING *1
WCODIGO_MODELO = "SENTRA"
WDESCRIPCION_MODELO = "SENTRA AUSTERO      "
stmt = "DECLARE SP_INS PROCEDURE "
stmt = stmt + "(:CODMOD IN CHAR(6),"
stmt = stmt + " :DESMOD IN CHAR(20),"
stmt = stmt + " :ESTADO INOUT CHAR(1) "
stmt = stmt + " (EXTERNAL NAME MCOMDES.SP_INS SIMPLE CALL) "
```

Preparamos la sentencia, asignamos el nombre SP_INS con el nombre físico del Stored Procedure en ILE RPG:

```
IRC = SQLPrepare(ghstmt, stmt, SQL_NTS)
```

Creamos la cadena de llamada al Stored Procedure:

```
stmtStored = "CALL SP_INS (?, ?, ?)"
```

Direccionamos en la memoria los parámetros del Stored Procedure:

```
IRC = SQLAllocStmt(ghdbc, hstmtStored)
```

Preparamos la sentencia, asignamos los parámetros a la dirección de memoria:

```
IRC = SQLPrepare(hstmtStored, stmtStored, SQL_NTS)
```

Enviamos los parámetros al stored procedure:

```
IRC = SQLBindParameter(hstmtStored, 1, SQL_PARAM_INPUT,
                                    SQL_C_CHAR, SQL_CHAR, 6, 0, ByVal WCODIGO_MODELO, 0, SQL_NTS)
```

```
IRC = SQLBindParameter(hstmtStored, 2, SQL_PARAM_INPUT,
                                    SQL_C_CHAR, SQL_CHAR, 20, 0, ByVal WDESCRIPCION_MODELO, 0,
SQL_NTS)
IRC = SQLBindParameter(hstmtStored, 3, SQL_PARAM_INPUT_OUTPUT,
                                    SQL_C_CHAR, SQL_CHAR, 1, 0, ByVal WESTADO, 0, SQL_NTS)
```

Ejecutamos el Stored Procedure:

```
IRC = SQLExecute(hstmtStored)
```

Tenemos un valor de devolución en la variable WESTADO (valor que devuelve el Stored Procedure).

Finalmente liberamos la dirección de memoria:

```
IRC = SQLFreeStmt(hstmtStored, SQL_DROP)
```

La programación es difícil y engorrosa pero el tiempo de ejecución es una fracción de segundo.

Además tenemos total control sobre los parámetros de input y output con respecto al Stored Procedure.

Conclusiones y Recomendaciones

De las propuestas iniciales que se tomaron en cuenta podemos decir lo siguiente:

Del uso de GUI (Graphical user interface)

a. El período de capacitación del sistema fue muy corto debido a la fácil comprensión de los paneles de trabajo.

b. Se pudo condensar el mayor flujo de trabajo en pocos paneles con lo cual se simplificó el trabajo de los usuarios.

c. Debido al manejo particular de rejillas del Visual Basic, se puede actualizar en grupos con lo que se hace más fácil el trabajo.

d. El fácil manejo de íconos y el formato libre para agrupar datos, hizo que se cometan muchos errores al diseñar los primeros paneles de actualización de datos. Para definir el primer formato definitivo de un panel de actualización Cabecera-detalle se demoró aproximadamente un mes.

De la decisión de usar cliente servidor:

a. A nivel de software los clientes no dependen de la plataforma del servidor. Actualmente tenemos aplicaciones en software de varios fabricantes. Por Ejemplo: tenemos las aplicaciones en Visual Basic, sistemas de información gerencial en COGNOS (Power Play), y estamos probando algunas consultas Power Builder.

b. El tener la posibilidad de traer datos a la PC ha permitido utilizar los datos que están en el servidor usando el software y el hardware que esté instalado en la PC.

c. El tener dos ambientes de programación: uno en el cliente y otro en la PC ha permitido administrar mejor el software. Los procesos principales y las reglas del negocio se han ubicado en el servidor y el diseño de los paneles y la interface con el usuario se han ubicado en el cliente. Si tenemos cálculos complejos que solo podemos manejar por programa y estos varían, solo necesitamos variar el programa que realiza estos cálculos en el servidor.

De la elección del AS/400 y el DB2/400:

a. El AS/400 tiene todo integrado: la programación, la Base de Datos, compiladores, ambiente de trabajo. El trabajar todo bajo un mismo ambiente o estándar nos facilita el aprendizaje y el manejo del recurso.

b. El DB2/400, cuenta con los recursos de seguridad (manejo de compromiso y control de journal) del AS/400 que han servido para controlar errores en el servidor desde el punto del cliente.

Del Beneficio/Costo:

a. El más importante beneficio ha sido la construcción de la red, con un mantenimiento fácil para poder intercambiar de sitio las computadoras sin ningún problema. Se utiliza tarjetas de red Ethernet de bajo costo.

Los servidores AS/400 tienen una base de datos segura que casi nunca ha traído problemas.

b. El soporte de hardware que se tiene de IBM es mejor que el que se tenía de UNIX.

c. A pesar de tener pocas experiencias con Cliente/Servidor en AS/400 estas se han incrementado rápidamente.

d. El departamento de sistemas de Nissan Maquinarias se ha convertido en un laboratorio en el que conviven varias plataformas y sistemas operativos, con lo que se puede decir que se tiene el dominio de buena parte de la tecnología.

Recomendaciones:

a. Al empezar un trabajo en Cliente/Servidor se tiene que pensar que se van a utilizar diferentes servidores, con diferentes sistemas operativos y protocolos de comunicación.

b. Antes de empezar el desarrollo se debe tener bien definida la tecnología a usar, probando de antemano que todo funcione, de lo contrario se amplía

innecesariamente el tiempo de pruebas de comunicaciones y compatibilidad de hardware y software.

c. Al optar por un desarrollo en una solución tecnológica similar se debe tomar en cuenta que los tiempos de programación se multiplican por cinco, debido a las funciones ODBC que se tienen que preparar para el envío de parámetros al servidor.

d. Al programar en ambiente visual, es muy fácil llenar de íconos y botones un panel de usuario. Se debe considerar que a mayor cantidad de íconos y gráficos la ejecución de los programas se hace más lenta. Además, cuando se recarga mucho los paneles, se pierde el objetivo de optar por un software visual: simplificar el trabajo al usuario y hacer intuitivas las funciones.

e. En líneas generales es necesario desarrollar en Cliente/Servidor, por las facilidades y el software que se está trabajando con esta plataforma.

Anexos

Anexo A. ODBC.

A.1. Componentes ODBC.

a. **Aplicación o programas en el lenguaje del cliente.**

La aplicación realizará las siguientes funciones:

Requiere conexión o sesión con un Data Source,

Envía requerimientos SQL al Data Source,

Define áreas de almacenamiento y formato de datos para los resultados de requerimientos SQL,

Requiere los resultados,

Recupera columna de datos de resultado,

Procesa errores,

Retorna los resultados al usuario si es necesario,

Requiere compromiso o retrotracción para el control de la transacción,

Termina la conexión al Data Source.

b. **Driver Manager o manejador de ODBC**

Usa un archivo de control (ODBC.INI para windows 3.1) para mapear un nombre de Data Source para especificar un driver DLL,

Inicializa la llamada al ODBC tanto del lenguaje como del Manejador de Base de Datos,

Provee puntos de entrada a las funciones ODBC para cada función,

Provee validación de parámetros y secuencia de validación para llamadas ODBC.

c. **Driver**

Establece una conexión al Data Source,

Submite requerimientos al Data Source,

Traslada datos de o a otros formatos, si es requerido por la aplicación,

Retorna resultados a la aplicación,

Formatea errores en los códigos estándar de error y los retorna a la aplicación,

Declara y manipula cursores si es necesario.

Inicia transacciones si el Data Source requiere iniciación explícita de la transacción,

d. **Data Source**

Las funciones generales proveídas por SQL DBMS,

Una instancia específica de combinación de productos de DBMS, operación remota de sistemas, red necesaria para acceso.

A.2. ODBC para AS/400

En el AS/400 el ODBC es un componente del Client Access/400.

El Client Access/400 es el software que permite trabajar AS/400 emulando terminales de AS/400 y utilizando los datos del DB2/400 desde cualquier plataforma.

Para el Client Access/400 V3R1 el ODBC tiene la siguiente secuencia de trabajo desde el punto de vista del Cliente

DB2/400* es un sistema de gestión de bases de datos relacionales multiusuario que se ejecuta en el sistema AS/400. Para acceder a los datos de DB2/400 se utiliza Lenguaje de Consulta Estructurada (SQL). Los ordenadores personales IBM y compatibles que funcionan con Microsoft. (Figura 23)

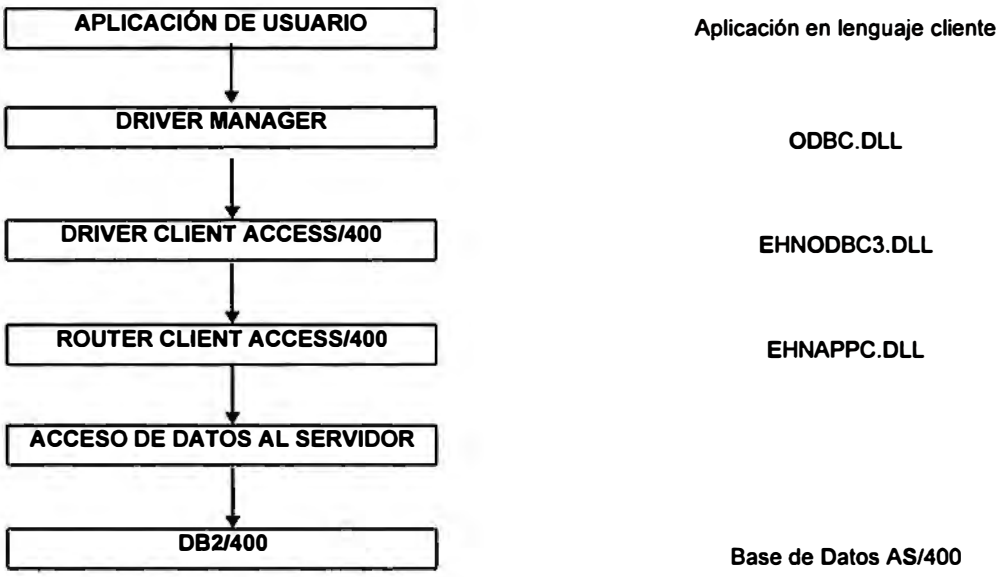

Figura 23. Como funciona ODBC para Client Access/400

Anexo B. DB2/400

El DB2/400 nos permite manejar:

- Restricciones,
- Integridad referencial,
- Triggers o activadores,
- Stored Procedures,

B.1. Restricciones

Restricción de Unicidad

Una restricción de unicidad identifica un campo o un conjunto de campos de un archivo de base de datos que cumpla las siguientes condiciones:

- es único dentro del archivo
- está en orden ascendente
- puede tener posibilidad de nulos

Restricción de Clave Primaria

Una restricción de clave primaria identifica un campo o un conjunto de campos de un archivo de base de datos que cumpla las siguientes condiciones:

- es único dentro del archivo
- está en orden ascendente
- no puede tener posibilidad de nulos

B.2. Integridad Referencial y Restricciones Referenciales

La integridad referencial es un término amplio que abarca todos los mecanismos y técnicas que aseguran la validez de los datos referenciados en una base de datos relacional. Las restricciones referenciales son un mecanismo para imponer el nivel de validez deseado en la base de datos.

Los usuarios de bases de datos desearán implantar la integridad referencial en su sistema de gestión de bases de datos por varias razones:

- Para asegurar que los valores de datos entre archivos se conservan en un estado que cumpla las normas de la empresa. Por ejemplo, si una empresa tiene una lista de clientes en un archivo y una lista de sus cuentas en otro archivo, no tiene sentido permitir que se añada una cuenta si no existe un cliente asociado a ella. Por la misma razón, no es razonable suprimir un cliente hasta que se hayan suprimido sus cuentas.
- Para poder definir las relaciones entre valores de datos.
- Para hacer que el sistema imponga las relaciones de datos, sin tener en cuenta qué aplicación efectúa cambios.
- Para mejorar el rendimiento de las comprobaciones de integridad realizadas a nivel HLL o SQL trasladando la comprobación a la base de datos.

Terminología de la Integridad Referencial

Para poder explicar qué es la integridad referencial es necesario comprender diversos términos.

- Clave Primaria. Un campo o conjunto de campos de un archivo de base de datos que debe ser exclusivo, ascendente y no puede contener valores nulos. La clave primaria es la vía de acceso primaria del archivo. La clave primaria puede convertirse en la clave padre en el archivo padre.
- Clave Exclusiva. Un campo o conjunto de campos de un archivo de base de datos que debe ser exclusivo, ascendente y puede contener valores nulos.
- Clave Padre. Un campo o conjunto de campos de un archivo de base de datos que debe ser exclusivo, ascendente y puede contener o no valores nulos.

- Clave Foránea. Un campo o conjunto de campos en el que cada valor no nulo debe coincidir con un valor de clave del archivo padre relacionado.
- Integridad referencial. El estado de una base de datos en la que los valores de las claves foráneas son válidas. La imposición de la integridad referencial impide la violación de la regla "la clave foránea no nula debe tener una clave padre coincidente".
- Restricción referencial. La restricción define una relación entre los registros identificados por claves padre y los registros identificados por claves foráneas. Dado que un archivo dependiente siempre depende del archivo padre, la restricción referencial se define desde la perspectiva del archivo dependiente. La restricción no puede definirse desde la perspectiva del archivo padre.
- Archivo padre. El archivo de una relación que contiene la clave padre o clave primaria.
- Archivo dependiente. El archivo de una relación que contiene la clave foránea.
- Pendiente de comprobación. El estado que se produce cuando la base de datos no sabe con certeza que un archivo dependiente concreto sólo contiene datos válidos relativos a su clave padre asociada.
- Regla de supresión. Una definición de la acción que la base de datos deberá llevar a cabo cuando se intente suprimir un registro padre.
- Regla de actualización. Una definición de la acción que la base de datos deberá llevar a cabo cuando se intente actualizar un registro padre.

Ejemplo Simple de Integridad Referencial

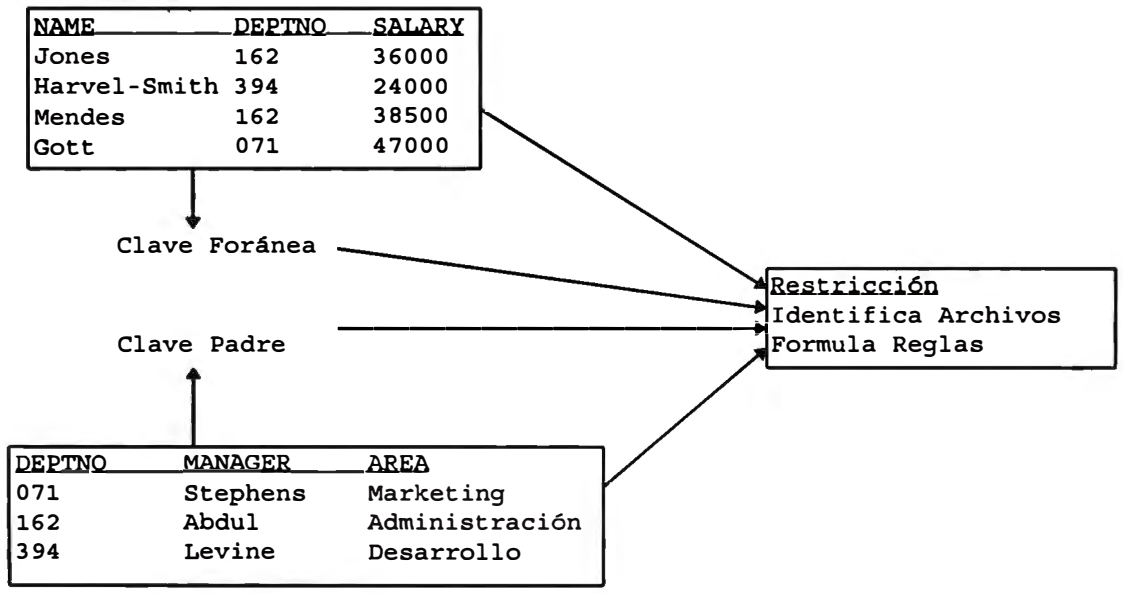

Figura 24. Integridad referencial.

Una base de datos contiene un archivo de empleados y un archivo de departamentos. Ambos archivos tienen un campo de número de departamento denominado DEPTNO. Los registros relacionados de estos archivos de base de datos son aquellos para los que empleado.DEPTNO sea igual a departamento.DEPTNO.

El archivo de departamento es el padre y departamento.DEPTNO es la clave padre. También es la clave primaria para el archivo. El archivo de empleados es el archivo dependiente y empleado.DEPTNO es la clave foránea. Cada registro del archivo de empleados que tiene un valor no nulo para DEPTNO está relacionado con un registro del archivo de departamentos y sólo uno.

La restricción sobre los valores de DEPTNO en el archivo de empleados es la restricción referencial. La restricción identifica los archivos, las claves y especifica las reglas que deben seguirse cuando un usuario o una aplicación realiza cambios (supresiones o actualizaciones) en el archivo padre.

Reglas de las Restricciones

Las restricciones referenciales le permiten especificar ciertas reglas que el sistema impondrá. Las reglas de las restricciones referenciales son aplicables a las supresiones y actualizaciones de claves padre.

Reglas de Supresión

La reglas de supresión especifican la acción a llevar a cabo cuando se suprime un valor de clave padre. La regla de supresión no afecta a los valores de clave padre nulos.

*NOACTION (Este es el valor por omisión.)

No se producirá la supresión de registros de un archivo padre si el valor de clave padre tiene un valor de clave foránea coincidente.

*CASCADE

La supresión de registros de un archivo padre provoca que se supriman los registros que tengan valores de clave foránea coincidentes en el archivo dependiente cuando el valor de clave padre coincida con el valor de clave foránea.

*SETNULL

La supresión de registros de un archivo padre actualiza los registros del archivo dependiente en los que el valor de clave padre no nulo coincida con el valor de clave foránea. Para los registros dependientes que coincidan con los criterios precedentes, se establecerán como nulos todos los campos con posibilidad de nulos de la clave foránea. No se actualizan los campos de clave foránea que tengan el atributo de no nulos.

*SETDFT

La supresión de registros de un archivo padre actualiza los registros del archivo dependiente en los que el valor de clave padre no nulo coincida con el valor de clave foránea. Para los registros dependientes que coincidan con los criterios precedentes, se establecerá el campo o campos de clave foránea a sus valores por omisión correspondientes.

*RESTRICT

No se producirá la supresión de registros de un archivo padre si el valor de clave padre tiene un valor de clave foránea coincidente.

Reglas de Actualización

La regla de actualización especifica la acción llevada a cabo cuando se intenta la actualización del archivo padre.

*NOACTION (Este es el valor por omisión.)

No se produce la actualización de registros de un archivo padre si hay un valor de clave foránea coincidente en el archivo dependiente.

*RESTRICT

No se produce la actualización de registros de un archivo padre si hay un valor de clave padre no nulo que coincida con un valor de clave foránea.

Si está efectuando operaciones de insertar, actualizar o suprimir en un archivo asociado con una restricción referencial y la regla de supresión, la regla de actualización o ninguna de ambas es *RESTRICT, deberá utilizar el registro por diario. El archivo padre y el dependiente deben registrarse por diario en el mismo receptor de diario.

Si inserta, actualiza o suprime registros en un archivo que esté asociado con una restricción referencial que tenga una regla de supresión, una regla de actualización o ambas y no sean *RESTRICT, será necesario el control de compromiso. Si no ha arrancado el control de compromiso, se arrancará y finalizará atuomáticamente.

B.3. Activadores o Triggers en DB2/400

Un activador es un conjunto de acciones que se ejecutan automáticamente cuando se efectúa una operación de cambio especificada en un archivo físico de base de datos especificado. La operación de cambio puede ser una sentencia de lenguaje de alto nivel de inserción, actualización o supresión en un programa de aplicación.

Los usuarios de bases de datos pueden utilizar activadores para:

- Imponer reglas de gestión

- Validar datos de entrada
- Generar un valor exclusivo para una fila recién insertada en un archivo distinto (función de subrogación)
- Grabar en otros archivos con fin de realizar un seguimiento de comprobación.
- Consultar otros archivos para referencias cruzadas
- Acceder a funciones del sistema (por ejemplo, imprimir un mensaje de excepción cuando se viola una regla)
- Reproducir datos en distintos archivos para conseguir coherencia de los datos.

Pueden conseguirse los siguientes beneficios en el entorno de gestión de los clientes:

- Desarrollo de aplicaciones más rápido: debido a que los activadores de almacenan en la base de datos, las acciones que llevan a cabo no tienen que codificarse en cada aplicación de la base de datos.
- Imposición global de reglas de gestión: un activador puede definirse una vez y luego utilizarse varias veces para cada aplicación que utilice la base de datos.
- Mantenimiento más fácil: si cambia la política de gestión, sólo será necesario cambiar el programa activador correspondiente en lugar de cambiar cada programa de aplicación.
- Mejora del rendimiento en el entorno cliente/servidor: todas las reglas se ejecutan en el servidor antes de devolver el resultado.

Para utilizar el soporte de activador AS/400 debe crear un programa activador y añadirlo a un archivo físico. Para añadir un activador a un archivo, debe:

- Identificar el archivo físico a supervisar
- Identificar la clase de operación a supervisar en el archivo.
- Crear un programa de lenguaje de alto nivel o CL que realice las acciones deseadas.

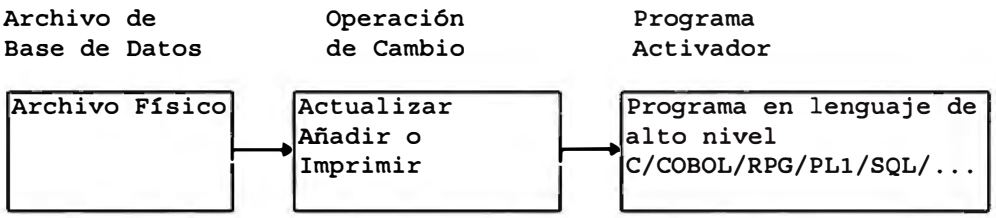

Figura 25. Secuencia de un trigger

Asociación de un trigger a un archivo

Puede asociar un máximo de seis activadores a un archivo físico, uno de cada uno de los siguientes:

- Antes de una inserción
- Después de una inserción
- Antes de una supresión
- Después de una supresión
- Antes de una actualización
- Después de una actualización

Un trigger se asocia a una tabla en DB2/400 indicando el tiempo y el evento al que pertenecen:

```
ADDPFTRG            Comando de asociación
FILE                Archivo al cual se asocia el programa
PGM                 Programa que se asocia
TRGTIME             Tiempo en que se ejecutará el programa
TRGEVENT            Evento por el que se ejecutará el programa
```

El tiempo de ejecución del trigger (TRGTIME) puede tener los siguientes valores:

```
*BEFORE             Se ejecuta antes de la actualización.
*AFTER              Se ejecuta después de la actualización.
```

El evento de ejecución del trigger (TRGEVENT) puede tener los siguientes valores:

```
*INSERT     Se ejecuta al insertar un registro en la tabla.
*UPDATE     Se ejecuta al actualizar un registro de la tabla.
*DELETE     Se ejecuta al borrar un registro de la tabla.
```

Las combinaciones posibles nos dan la siguiente tabla:

```
TRGTIME             TRGEVENT            Se ejecuta
*BEFORE             *INSERT             Antes de insertar registro.
```

	*UPDATE	Antes de actualizar registro
	*DELETE	Antes de eliminar registro
*AFTER	*INSERT	Después de insertar registro.
	*UPDATE	Después de actualizar registro.
	*DELETE	Después de eliminar registro.

Cada inserción, supresión o actualización puede llamar a un activador antes de que se produzca la operación de cambio y también después. Por ejemplo, en la Figura 17-2, en que se actualiza un registro del archivo EMP, la operación de actualización llama a un programa activador antes y después de llevarse a cabo la actualización.

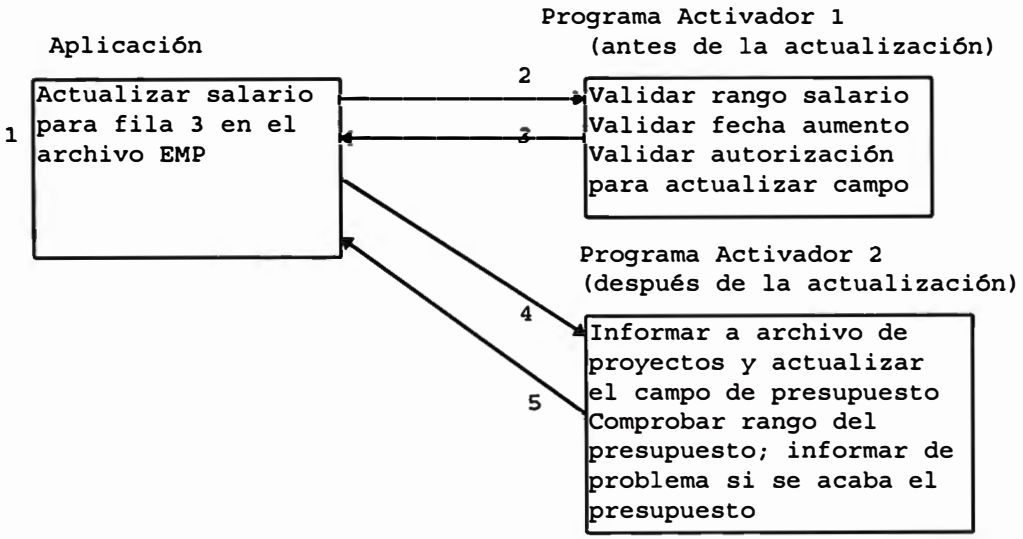

Figura 26. Activadores Antes y Después de una Operación de Cambio

Supuestos para la Figura

Todos los archivos se abren bajo la misma definición de compromiso.

El archivo EMP tiene un activador de actualización anterior (Programa Activador 1) y un activador de actualización posterior (Programa Activador 2).

Notas para la Figura

1. La aplicación intenta actualizar el campo de salario de un empleado en el archivo EMP.

2. El sistema llama al activador de actualización anterior antes de que se actualice el registro. El activador de actualización anterior valida todas las reglas.

3. Si la validación resulta anómala, el activador indica una excepción informando al sistema que se ha producido un error en el programa activador. El sistema informa entonces a la aplicación de que la operación de actualización es anómala y retrotrae los cambios efectuados por el activador anterior. En esta situación no se llama al activador de actualización posterior.

4. Si se han validado todas las reglas satisfactoriamente, el programa activador vuelve normalmente. El sistema efectúa entonces la operación de actualización. Si la operación de actualización es satisfactoria, el sistema llama al activador de actualización posterior para que efectúe las acciones posteriores a la actualización.

5. El activador de actualización posterior realiza todas las acciones necesarias y vuelve. Si se produce algún error en el programa activador de actualización posterior, el programa indica una excepción al sistema. El sistema informa a la aplicación de que la operación de actualización ha resultado anómala y se retrotraen todos los cambios efectuados por ambos activadores más la operación de actualización.

Sección del Almacenamiento Intermedio del Activador

El almacenamiento intermedio del activador, (vea la Figura 17-3, tiene dos secciones lógicas, una estática y una variable:

La sección estática contiene:

- Una plantilla de activador que contiene el nombre del archivo físico, el nombre de miembro, el evento activador, la hora de activador, el nivel de bloqueo de compromiso y el CCSID del registro de cambio actual.
- Desplazamiento y longitudes de las áreas de registro y correlaciones de bytes nulos.
- Esta sección ocupa (en decimales) los desplazamientos del 0 al 95.

La sección variable contiene:

- Areas para el registro antiguo, la antigua correlación de bytes nulos, el nuevo registro y la nueva correlación de bytes nulos.

Desplz			
Dec	Hex	Tipo	Campo
0	0	CHAR(10)	Nombre del archivo físico
10	A	CHAR(10)	Nombre de biblioteca donde está el archivo físico
20	14	CHAR(10)	Nombre de miembro del archivo físico
30	1E	CHAR(1)	Evento activador
31	1F	CHAR(1)	Hora de activador
32	20	CHAR(1)	Nivel de bloqueo de compromiso
33	21	CHAR(3)	Reservado
36	24	BINARY(4)	CCSID de los datos
40	28	CHAR(8)	Reservado
48	30	BINARY(4)	Desplazamiento del registro original
52	34	BINARY(4)	Longitud del registro original
56	38	BINARY(4)	Desplazamiento de correlación de bytes nulos del registro original
60	3C	BINARY(4)	Longitud de correlación de bytes nulos del registro original
64	40	BINARY(4)	Desplazamiento del registro nuevo
68	44	BINARY(4)	Longitud del registro nuevo
72	48	BINARY(4)	Desplazamiento de correlación de bytes nulos del registro nuevo
76	4C	BINARY(4)	Longitud de correlación de bytes nulos del registro nuevo
80	50	CHAR(16)	Reservado
*	*	CHAR(*)	Registro original
*	*	CHAR(*)	Correlación de bytes nulos del registro original
*	*	CHAR(*)	Registro nuevo
*	*	CHAR(*)	Correlación de bytes nulos del registro nuevo

Figura 27. Almacenamiento Intermedio del Activador.

Funciones en el Trigger

El trigger recibe como únicos parámetros los valores de los registros antiguo y nuevo de la base de datos asociada. Con estos parámetros nosotros podemos actualizar o consultar otras tablas si queremos.

Dentro de un trigger se puede declarar archivos hacer operaciones de cálculo y todo lo que está permitido a un trabajo en batch.

Una vez que el trigger se ha ejecutado, no envía ningún mensaje al programa que hizo la actualización de la tabla. Si el trigger tuvo un error solamente se detecta el error como si se hubiera realizado mal la actualización de la tabla asociada.

B.4. Stored Procedures

Los Stored Procedures o Procedimientos Almacenados son programas que realizan actualizaciones a las tablas de la Base de Datos y/o cálculos específicos con la posibilidad de manejar parámetros de entrada y salida.

Estos Stored Procedures se ubican en el servidor y pueden ser ejecutados desde cualquier cliente en cualquier punto. El error se puede controlar por el valor de parámetros de retorno que se tiene en el programa Cliente.

Funciones del Stored Procedure

Cuando se ejecutan sentencias SQL desde un leguaje cliente a través de ODBC o SQL remoto, el sistema se encarga de preparar la sentencia enviarla y luego recibir los resultados, esto tiene un tiempo de demora bastante alto (para nuestra aplicación aprox. 5 segundos) por cada actualización de una tabla.

Para rutinas en que se actualizan varias tablas se puede reemplazar todas estas actualizaciones con un Stored Procedure que representa una sola preparación y ejecución de una sentencia SQL (la de llamada al Stored Procedure).

Otra razón es porque tenemos el Stored Procedure ubicado en servidor y si queremos alterar cualquier secuencia de actualización y o de proceso de cálculo complejo, sólo

tenemos que actualizar el Stored Procedure en el servidor y automáticamente lo utilizan los clientes sin necesitar de actualizar el software instalado en cada cliente.

Por ejemplo.

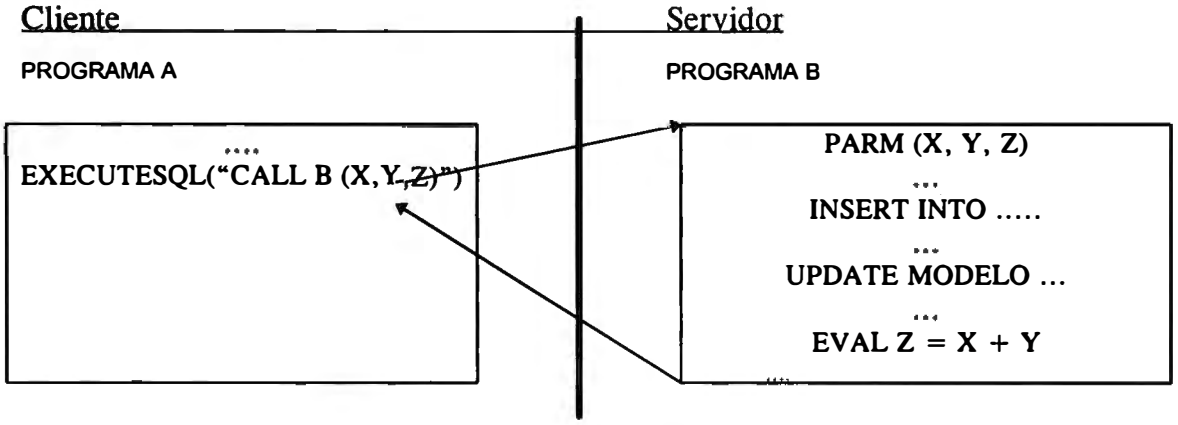

Figura 28. Ejemplo de ejecución de Stored Procedure.

El programa A (en el cliente escrito en un lenguaje visual) invoca al Programa B (Stored Procedure escrito un lenguaje del servidor) enviándole los parámetros X, Y, Z.

El Stored Procedure después de realizar varias actualizaciones, realiza un cálculo y devuelve el nuevo valor de las variables al programa A.

Es decir como si los dos programas estuvieran en el mismo computador.

Anexo C. Client Access/400

C.1. Messaging Application Program Interface (MAPI).

La especificación Messaging Application Program Interface (MAPI) de Microsoft** proporciona a los programadores de aplicaciones acceso a los sistemas de mensajes para clientes de Windows** 3.1.

C.2. Open Database Connectivity (ODBC).

ODBC es una interfaz abierta destinada a Windows Open Services Architecture (WOSA) de Microsoft** para Windows** 3.x, Windows para Trabajo en Grupo**, Windows NT**. ODBC proporciona una interfaz SQL para aplicaciones Windows.

C.3. Direccionador de comunicaciones APPC.

La API del direcccionador de comunicaciones APPC proporciona un método para escribir aplicaciones de proceso cooperativo entre ordenadores personales y sistemas AS/400. La API del direccionador aísla al programador de la programación de comunicaciones de bajo nivel y de los tipos de conectividad de hardware.

C.4. Servidor de colas de datos DDE.

Las colas de datos son un tipo de objeto el sistema AS/400. Son espacios de espera para diferentes grupos de datos a los que se denomina mensajes.

Múltiples aplicaciones pueden enviar mensajes a las colas de datos, y los mensajes se quedan allí hasta que otras aplicaciones los recuperan.

Para arrancar el Servidor de Colas de Datos DDE se utiliza el icono Colas de Datos DDE. El servidor DDE deben utilizarlo aplicaciones que utilicen la interfaz

Intercambio Dinámico de Datos (DDE) de Microsoft Windows** para comunicarse con las colas de datos de AS/400. El servidor DDE no es necesario cuando se utilizan APIs de colas de datos para comunicarse con las colas de datos AS/400.

C.5. Transformación de datos.

La API de transformación de datos se utiliza para convertir datos entre formatos AS/400 y PC. La API de transformación de datos soporta conversión de texto y numerosos formatos numéricos.

C.6. Especificación de interfaz de controlador de red (NDIS).

Microsoft** Corporation y 3Com** Corporation han desarrollado conjuntamente la Especificación de Interfaz de Controlador de Red (NDIS).

NDIS es una interfaz de estándar industrial, común y abierta, que:

- Proporciona una interfaz de control de acceso medio (MAC) para controladores de adaptador de red de área local (LAN) y controladores de protocolo.
- Habilita la comunicación entre sí de adaptadores de red y software LAN de distintos fabricantes.

C.7. Unidades de red (Carpetas compartidas).

La API de carpetas compartidas proporciona un conjunto de rutinas para ayudar a gestionar la asignación y liberación de unidades de red (unidades de carpetas compartidas). Es específico para carpetas compartidas y no interfiere en la asignación de otras unidades de red. Esta API permite hasta 26 asignaciones de unidad y no limita en forma alguna el rango de letras de unidad.

C.8. Redireccionador de red.

La API de redireccionador de red proporciona una interfaz a las funciones de redirección de red DOS. Utilizando esta API, puede escribir una aplicación para acceder a unidades de red e impresoras de Client Access/400 así como a dispositivos conectados a otras redes.

C.9. Servicio de operador de nodos (NOF).

El soporte de servicios de red proporciona una interfaz de programa de aplicación (API) de servicio de operador de nodos que permite a un programa de aplicación:

- Cambiar determinada información de configuración después de arrancar servicios de red.
- Activar y desactivar unidades lógicas (LUs).
- Conectar y desconectar enlaces.

C.10. SQL remoto.

La API de SQL remoto proporciona acceso a archivos de base de datos AS/400. La API de SQL remoto también proporciona un mecanismo de comunicaciones entre aplicaciones PC y AS/400.

C.11. Transferencia.

La API de función de transferencia proporciona acceso a bases de datos de AS/400 utilizando sintaxis SQL. Puede recuperar registros individuales; sin embargo, sólo opera en archivos de base de datos completos cuando se realizan actualizaciones. Esta API resulta útil para aplicaciones que necesitan la posibilidad de consulta del sistema principal o que desean ejecución ampliada para actualizar o transferir archivos enteros.

C.12. Someter mandato remoto.

La API para someter mandato remoto permite a las estaciones de trabajo arrancar programas no interactivos y procedimientos en el sistema AS/400 y recibir mensajes de terminación. Puede ejecutarse cualquier mandato o programa por lotes AS/400. Pueden enviarse hasta diez mensajes de respuesta desde el mandato o programa AS/400. Los datos de programa no pueden retornarse a través de esta interfaz.

C.13. Impresora virtual.

La API de impresora virtual proporciona acceso a impresoras AS/400. Las impresoras virtuales pueden asignarse y liberarse utilizando esta API. La API soporta hasta nueve asignaciones de impresora.

C.14. Instalación del Client Access/400

Se crea el sistema (servidor AS/400) en la lista de acceso a AS/400 (Pueden ser más de uno)

Se configura el servidor de datos.

En las opciones comunes el cliente pide que se ingrese el Modo (QPCSUPP, APPN o TCP/IP).

Se ingresa además el nombre con el que será reconocida el cliente en el servidor.

Se establece la conexión con el AS/400 y se termina la instalación del Client Access/400.

Se instalan los componentes requeridos para el cliente (Por ejemplo: ODBC, Rumba/400, Transferencia de datos, etc.).

C.15. Creación de un Data Source para Client Access/400 (Windows 3.1)

Para hacer uso del driver ODBC del Client Access/400 debemos ir al panel de control e ingresar con el icono del ODBC (o en el icono ODBC del Client Access).

Elegir la opción ADD de la ventana que muestra los Data Sources.

Muestra la Ventana de Drivers ODBC Instalados.

Elegir el driver Client Access/400 ODBC Drivers y presionar OK.

Nos muestra la ventana para definir un Data Source (se pueden definir, varios Data Source de acuerdo a los sistemas que se quieran acceder).

Este Data Source tiene las características de poder accesar a los datos en AS/400.

Nos pide los siguientes Valores:

Data Source Name	Nombre del Data Source para nuestras aplicaciones
System	Nombre del sistema.
USER ID	Un nombre de usuario AS/400 (puede estar en blanco).
Default Libraries	Las librerías que se van a poder trabajar.
Commit Mode	El nivel de control de compromiso que se va ha tener.

Y luego una serie de opciones que nos piden los estándares de fecha, hora, y separador decimal que se va ha usar.

Una vez que se da OK se actualiza el archivo ODBC.INI en el cual aparece el nuevo Data Source con su descripción y sus parámetros ingresados.

Se debe mencionar que para cada driver de ODBC (Por ejemplo: Access 2.0, FOXPRO) los parámetros son distintos.

Anexo D. Como trabaja AS/400 en Cliente/Servidor

D.1. Controladores y Dispositivos APPC

Controladores

Al instalar el Client Access/400 en un cliente se le da un nombre de PC.

Al conectarse al AS/400 por primera vez (durante la instalación) este reconoce el nombre de PC y lo asocia con el Número interno de tarjeta de red (hardware) y crea un controlador para este cliente (PC).

Cada vez que aparece un nuevo cliente el sistema crea un controlador para el con su nombre y su número de tarjeta de red.

Los controladores deben ser únicos por cada combinación Nombre de Cliente-Tarjeta de red

Los controladores se activan en la automáticamente cuando un cliente se conecta al AS/400.

Si se hace un cambio en el nombre del cliente o un cambio de tarjeta de red, el AS/400 no reconoce la asociación Nombre de Cliente-Tarjeta de red y no permite su conexión.

Al finalizar la conexión de un cliente con el AS/400 automáticamente se desactiva el controlador correspondiente.

Por ejemplo para el usuario EESCUDERO se ha conectado a AS/400 tenemos:

Estado de los controladores.

Para ver los controladores: comando WRKCFGSTS *CTL

```
                    Trabajar con Estado de Configuración            MAQUI02
                                                          96/09/30  19:17:33
Situar en . . . . . .              Caracteres iniciales

Teclee opciones, pulse Intro.
  1=Activar   2=Desactivar     5=Trabajar con trabajo
  8=Trabajar con descripción   9=Ver estado modalidad ...

Opc  Descripción       Estado               ----------Trabajo-----------
     AGAVIDIA          PENDIENTE ACTIVACION
       AGAVIDIA        PENDIENTE ACTIVACION
     ALMAC01           DESACTIVADO
       ALMAC01         DESACTIVADO
     EESCUDERO         ACTIVO
       EESCUDERO       ACTIVO
                                                                      Más...
Parámetros o mandato
===>
F3=Salir   F4=Solicitud   F12=Cancelar   F23=Más opciones   F24=Más teclas
```

Figura 29. Controladores.

Estado de la línea: comando WRKCFGSTS *LIN

```
                    Trabajar con Estado de Configuración            MAQUI02
                                                          96/09/30  19:15:58
Situar en . . . . . .              Caracteres iniciales

Teclee opciones, pulse Intro.
  1=Activar   2=Desactivar     5=Trabajar con trabajo
  8=Trabajar con descripción   9=Ver estado modalidad ...

Opc  Descripción       Estado               ----------Trabajo-----------
     ETHERNET          ACTIVO
       EESCUDER        ACTIVO
         EESCUDER      ACTIVO
           QPCSUPP     ACTIVO/DESTINO       EESCUDER    PCS         076179
           QPCSUPP     ACTIVO/DESTINO       QPWFSERV    PCS         076178
       JDIAZ           ACTIVO
         JDIAZ         ACTIVO
                                                                      Más...
Parámetros o mandato
===>
F3=Salir   F4=Solicitud   F12=Cancelar   F23=Más opciones   F24=Más teclas
```

Figura 30. Línea.

Dispositivos

Los dispositivos son creados automáticamente por el sistema y corresponden al acceso de una aplicación Cliente/Servidor.

Al cerrar una aplicación automáticamente se desactiva el dispositivo correspondiente.

Para el mismo ejemplo veamos que tenemos al analizar el usuario EESCUDERO:

Sin abrir sesión Cliente/Servidor: Comando: WRKCFGSTS *DEV

```
                    Trabajar con Estado de Configuración            MAQUI02
                                                          96/09/30  19:25:20
 Situar en . . . . . .              Caracteres iniciales

 Teclee opciones, pulse Intro.
   1=Activar   2=Desactivar     5=Trabajar con trabajo
   8=Trabajar con descripción   9=Ver estado modalidad ...

Opc  Descripción       Estado                ----------Trabajo------------
      EESCUDER          ACTIVO
        QPCSUPP         ACTIVO/DESTINO        EESCUDER     PCS          076179
        QPCSUPP         ACTIVO/DESTINO        QPWFSERV     PCS          076178
      EESCUDERS1        ACTIVO                EESCUDERS1   AMENDOZA     076180
      ETHERTCP          ACTIVO                QTCPIP       QTCP         075674
      GCASTELL          DESACTIVADO
      GCASTELLS1        PENDIENTE ACTIVACION
                                                                          Más...
 Parámetros o mandato
 ===>
 F3=Salir   F4=Solicitud   F12=Cancelar   F23=Más opciones   F24=Más teclas
```

Figura 31. Dispositivos.

D.2. Modalidad de Descripción

Las Modalidades de APPC son nombres que Client Access/400 para Windows utiliza para solicitar un conjunto específico de conectividades de red de una sesión que Client Access/400 para Windows desea utilizar para una conversación.

Estas conectividades incluyen, por ejemplo, el nivel de sincronización más alto para las conversaciones de las sesiones, la clase de servicio para las sesiones y las características de direccionamiento y retardo de la sesión.

Cuando se instala el Client Access/400 se pide la Modalidad (QPCSUPP, APPN o TCP/IP) esta Modalidad existe en AS/400 y representa la Modalidad de acceso al sistema que en AS/400 se llama Modalidad de Descripción APPC.

Estas Modalidades de descripción se activan cuando se abre una aplicación y se empiezan a abrir la bases de datos.

Esta sesión tendrá la Modalidad de Descripción que se ingresó al instalar el Client Access en el cliente (el defecto es QPCSUPP).

Si esta Modalidad de Descripción no existe, no podrá ingresar la PC al sistema.

Cuando se conecta al AS/400 por ejemplo abre la Modalidad de Descripción QPCSUPP:

```
                    Trabajar con Estado de Configuración            MAQUI02
                                                          96/09/30  19:25:20
 Situar en . . . . . .              Caracteres iniciales

 Teclee opciones, pulse Intro.
   1=Activar   2=Desactivar     5=Trabajar con trabajo
   8=Trabajar con descripción   9=Ver estado modalidad ...

Opc  Descripción       Estado               ----------Trabajo------------
      EESCUDER          ACTIVO
        QPCSUPP         ACTIVO/DESTINO        EESCUDER     PCS          076179
        QPCSUPP         ACTIVO/DESTINO        QPWFSERV     PCS          076178
      EESCUDERS1        ACTIVO                EESCUDERS1   AMENDOZA     076180
      ETHERTCP          ACTIVO                QTCPIP       QTCP         075674
      GCASTELL          DESACTIVADO
      GCASTELLS1        PENDIENTE ACTIVACION
                                                                        Más...
 Parámetros o mandato
 ===>
 F3=Salir   F4=Solicitud   F12=Cancelar   F23=Más opciones   F24=Más teclas
```

Figura 32. Conexión al Client/Access.

Por cada OPENDATABASE o LOGONVIAODBC se abre una sesión Cliente/Servidor en el subsistema QSERVER con la Modalidad de Descripción QSERVER.

Para nuestro ejemplo EESCUDERO abre la aplicación que a su vez abre dos sesiones (OPENDATABASE y LOGONVIAODBC) en Cliente/Servidor entonces se se observa que adicionalmente la Modalidad de Descripción QPCSUPP tenemos dos accesos abiertos con Modalidad de Descripción QSERVER

```
                    Trabajar con Estado de Configuración            MAQUI02
                                                          96/09/30  19:35:06
 Situar en . . . . . .              Caracteres iniciales

 Teclee opciones, pulse Intro.
```

```
  1=Activar   2=Desactivar     5=Trabajar con trabajo
  8=Trabajar con descripción   9=Ver estado modalidad

Opc  Descripción       Estado              ----------Trabajo-----------
     EESCUDER          ACTIVO
       QPCSUPP         ACTIVO/DESTINO      EESCUDER    PCS         076179
       QPCSUPP         ACTIVO/DESTINO      QPWFSERV    PCS         076178
       QSERVER         ACTIVO/DESTINO      QZDAINIT    QUSER       076185
       QSERVER         ACTIVO/DESTINO      QZDAINIT    QUSER       076186
     EESCUDERS1        ACTIVO              EESCUDERS1  AMENDOZA    076180
     ETHERTCP          ACTIVO              QTCPIP      QTCP        075674
                                                                     Más...
Parámetros o mandato
===>
F3=Salir   F4=Solicitud   F12=Cancelar   F23=Más opciones   F24=Más teclas
```

Figura 33. Sesiones Cliente/Servidor con dos accesos a la base de datos.

Al cerrar la base de datos en la aplicación (LogoffViaODBC o gbasedatos.Close) se elimina automáticamente la sesión correspondiente en el subsistema QSERVER.

Bibliografía

AS/400 Client/Server Performance Using Aplication Develoment Tools, Agosto 1995, Inglés
IBM International Technical Support Organization
Rochester Center

News/400 N° 188 Julio de 1996, Inglés
Step-by-Step Setup for TCP/IP, pp 109-117
Duke Communications International

Puesta a punto rápida de TCP/IP Versión 3, Setiembre de 1995, Español
AS/400 Advanced Series
IBM S.A. National Solutions Center

Secret of effective GUI design, Mark Minasi, 1994, Inglés
SYBEX Inc.

Client/Server Business Solutions: 10 key issues to consider Before you buy, 1995, Inglés
Peoplesoft Inc.

ODBC 2.0 Programmer's Reference and SDK Guide, 1994, Inglés
Microsoft Press

News/400, March 1996, Inglés,
Duke communications International. Pp 42-65

Visual Basic Controls desk reference CD, 1995, Inglés
Mark Pruett, C.Woody Butler, Greg Irwin
The Waite Group Inc.

Fundamentos Cliente/Servidor, 1996, Español
Peter Bohnhoff, Rob Janssen, Rossana Martín
Documentos Computerworld
Madrid