ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN

"SISTEMA DE INFORMACIÓN GEOGRÁFICO PARA UNA EMPRESA ELÉCTRICA: SISTEMA DE INVENTARIO"

PROYECTO DEL TÓPICO DE GRADUACIÓN
"Sistemas AM FM GIS"
Previo a la obtención del Título en:

INGENIERÍA EN ELECTRICIDAD
Especialización: COMPUTACIÓN

Presentado por:

NELSON GONZÁLEZ ALVARADO
KARINA ASTUDILLO BARAHONA
MARIO NARANJO FALQUEZ
DIANA VILLACIS PONCE

GUAYAQUIL - ECUADOR
1997

AGRADECIMIENTO

A Dios, a nuestros padres, a nuestra familia en general, a nuestros profesores y amigos, por haber colaborado de una u otra forma en nuestra formación.

Un agradecimiento especial al Ing. Guido Caicedo R., al Dr. Enrique Peláez J. , a la Magister Ruth Alvarez, al Ing. Tommy Topic y al Ing. Jefferson Paredes por su comprensión y colaboración en la culminación de este proyecto.

Quisiéramos agradecer también a Nyree Torres, a Carlos Luis Aráuz y a Albert Espinal por habernos apoyado durante las largas jornadas de desarrollo del proyecto.

DEDICATORIA

A NUESTRA FAMILIA

Ing. Javier Urquizo

Profesor de Tópico

Ing . Raúl Rivas

Miembro del Tribunal

Ing. Leonardo Reyes

Miembro del Tribunal

DECLARACIÓN EXPRESA

"La responsabilidad por los hechos, ideas y doctrinas expuestos en este proyecto, nos corresponden exclusivamente; y, el patrimonio intelectual de la misma, a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL".

(Reglamento de Exámenes y Títulos profesionales de la ESPOL)

Jéssica Karina Astudillo Barahona

Nelson Javier González Alvarado

Diana Maureen Villacís Ponce

Mario Alberto Naranjo Falquéz

RESUMEN

El proyecto realizado es un Sistema de Información Geográfico (GIS) para la Empresa Eléctrica del Ecuador, y abarca el sistema de inventario de esta empresa para la zona de Urdesa en Guayaquil.

El sistema comprende dos partes:

- La representación gráfica de la zona de Urdesa con sus respectivos componentes del Sistema Nacional Interconectado. Los componentes que hemos considerado para este diseño son: subestaciones, transformadores, postes, circuitos primarios y circuitos secundarios.
- Una aplicación desarrollada para un ambiente operativo *Windows 95*, que abarca un sistema de consultas, mantenimiento y reportes sobre una base de datos de inventario.

Ambas partes están íntimamente relacionadas, puesto que desde la parte gráfica se puede ingresar y consultar información sobre los objetos representados, pertenecientes al sistema eléctrico. Dicha información está almacenada en una base de datos que contiene además, datos sobre entidades que no están representadas gráficamente, como es el caso de los clientes y medidores.

INDICE GENERAL

INDICE DE FIGURAS

INTRODUCCIÓN

Este documento tiene como objetivo presentarle a usted el sistema desarrollado "EMELEC_SIG".

El documento para una mejor comprensión ha sido dividido en tres partes:

Capítulo I

Este capítulo comprende el "Manual del Sistema", el cual describe en forma general el sistema desarrollado, sus objetivos, funcionalidad, los módulos que lo componen, la metodología utilizada y las bases teóricas del mismo.

Capítulo II

Corresponde al "Manual del Usuario", en el cual se describe el uso del sistema y su interacción con el usuario.

Capítulo III

En esta parte se describe el "Diseño de la Base de Datos", información técnica sobre el modelamiento de los datos, herramientas de diseño utilizadas y cómo está dividido el sistema.

CAPÍTULO I

MANUAL DEL SISTEMA

1.1 ANTECEDENTES

El proyecto de Modernización del Estado ha impulsado tanto a compañías estatales como privadas a la búsqueda de la excelencia en su trabajo, ayudadas cada vez más por un creciente uso de sistemas computacionales. En el caso de empresas de servicios como una empresa eléctrica, que cubren áreas grandes como una ciudad y que pueden llegar a cubrir todo un país, la inversión que pueda hacerse en un sistema computacional que ayude a la empresa a tener un mayor control sobre los dispositivos que utiliza para brindar servicio y que a su vez permita mejorar dicho servicio a sus usuarios, es mínima comparada con las ventajas que esto representaría para la empresa.

En ocasiones el mayor de los gastos dentro de una empresa de este tipo no corresponde a sueldos de los empleados sino a gastos de operación, tales como reparaciones, cortes y reconexiones, reemplazo de cables, ubicación de nuevos postes, etc. Y los gastos no son sólo en la compra de

dispositivos nuevos sino también en el pago del personal involucrado en estas tareas. Con un Sistema de Información Geográfico se puede guardar información gráfica georeferenciada de una ciudad o de todo un país, que contenga no sólo lo referente a cuadras, calles y solares, sino además elementos del tendido eléctrico como: postes, transformadores, alta tensión, baja tensión, puentes o conectores, etc.. Dichos elementos gráficos pueden ser enlazados con información almacenada en una Base de Datos Relacional, facilitando de este modo la obtención de datos de un elemento específico dentro un mapa. Esto último sería de gran ayuda al momento de determinar la causa de un problema, por ejemplo la falta de corriente en un sector.

A veces una reparación puede tomar desde unas pocas horas hasta días, el alumbrado público si bien es bastante bueno no está operativo en un cien por ciento, hay sectores que aún no cuentan con el servicio, en fin. Problemas que no necesariamente se dan por falta de atención, sino más bien por la falta de un sistema que le permita a la empresa saber en forma rápida y eficiente los recursos con los que cuenta, el estado en que estos se encuentran, sus características, los sectores que cubren, los abonados a los

que sirven, el número de conexiones ilegales que existen dentro de la ciudad, y un largo etcétera.

Si un abonado hace una llamada telefónica al departamento de reparaciones, indicando que su casa está sin energía eléctrica, hay varias preguntas que el dependiente al otro lado de la línea tendrá que formular para poder deducir en base a las respuestas la probable causa del problema. La mayoría de nosotros, por no decir todos, nos hemos topado con la desagradable sorpresa de regresar a la casa y encontrar que no tenemos corriente , y la causa va desde un olvido involuntario de pagar la planilla correspondiente, enfrentándonos al odioso tiempo de espera por la reconexión, hasta la explosión del transformador de la esquina.

Las eventualidades mencionadas anteriormente pueden resolverse de una manera más rápida, beneficiando tanto a la empresa como al abonado, con la ayuda de un sistema de inventario. Dicho sistema de inventario guardaría información sobre los recursos de la empresa como postes, transformadores, circuitos primarios, circuitos secundarios, subestaciones y medidores, además de brindar información básica sobre los abonados y la

posible existencia de conexiones ilegales. Con esto la empresa ahorraría tiempo y dinero, brindando un mejor servicio a sus abonados.

1.2. OBJETIVOS DEL SISTEMA

- Desarrollar un sistema que permita mejorar el control de inventario de una empresa eléctrica.
- Permitir la determinación de fallas de la red eléctrica de manera precisa.
- Generar informes del estado de la red eléctrica.
- Registrar los clientes y sus respectivos medidores.
- Llevar un registro automatizado de conexiones ilegales.
- Facilidades para prever crecimiento futuro o cambios de infraestructura.

1.3. FUNCIONALIDAD DEL SISTEMA

El sistema desarrollado cumple con los siguientes requisitos funcionales:

- Visualización gráfica de los elementos del SNI considerados en el diseño.
- Consultas visuales sobre dichos elementos.
- Provisión de formas de ingreso de datos para los elementos gráficos.
- Programa de inventario usando tecnología cliente - servidor.
- Consultas generales y específicas sobre los elementos representados visualmente y sobre los demás elementos del sistema como subestaciones, medidores, clientes y conexiones ilegales.
- Reportes en pantalla, generados en un archivo o impresos sobre los elementos del sistema.
- Sistema de mantenimiento de la Base de Datos con opciones de búsqueda y de ingreso de elementos nuevos que no son añadidos desde la parte visual, como es el caso de las subestaciones, clientes, medidores, estados de los dispositivos, sectores y conexiones ilegales.

1.4. METODOLOGÍA

Debido a que se trata de un anteproyecto del Sistema de Inventario para una empresa eléctrica y por tratarse de un trabajo de tópico, se limitó la información tanto gráfica como de datos a la zona de Urdesa en la ciudad de Guayaquil. Los datos fueron provistos por el Dpto. de Inventario de EMELEC. No obstante, el análisis y el diseño de la Base de Datos, como la implementación de la aplicación se hizo teniendo en cuenta los requerimientos globales de un sistema que abarca una ciudad.

La parte visual del sistema fue realizada en ***MicroStation*** , que es un graficador que brinda facilidades de digitalización, dibujo y enlace de objetos gráficos a una base de datos que puede ser Oracle, Xbase o cualquiera que soporte ODBC o RIS. Por otro lado el sistema de consultas, reportes y mantenimiento de la base de datos fue desarrollado en ***Visual Basic*** bajo una plataforma *Windows*. El modelamiento de la Base de Datos y la generación de las tablas e índices que la componen se realizó usando ***Erwin*** como herramienta de diseño.

A continuación se detalla el esquema de hardware y software usado para el desarrollo del sistema:

Plataforma de Hardware:

- *Servidor:*

  IBM Compatible

  Procesador Pentium 133 Mhz

  32 Mb RAM

  Disco duro de 2 Gb

  Tarjeta de red ethernet 10 Bt NE-2000

  Monitor SVGA

- *Cliente:*

  IBM PC compatible

  Procesador Pentium 90 Mhz

  16 Mb RAM

  Disco duro de 300 Mb

  Tarjeta de red ethernet 10 Bt NE-2000

  Monitor SVGA

Plataforma de software:

- *Servidor:*

  Sistema Operativo Windows NT 4.0

- *Cliente:*

  Sistema Operativo Windows 3.1 o Windows 95

1.5. BREVE DESCRIPCIÓN DE LOS MÓDULOS IMPLEMENTADOS

Por ser una aplicación desarrollada en *Visual Basic* todos los eventos son módulos, pero en este caso nos referimos a la división de la aplicación en forma lógica. En este sentido el sistema puede decirse que está dividido en tres partes:

- Consultas
- Mantenimientos y
- Reportes

CONSULTAS

Las consultas como su nombre lo indica permiten al usuario examinar o ver los datos que se encuentran almacenados en la Base de Datos para cada uno de los elementos representados en ella. Hay consultas generales en las que se puede buscar los registros que cumplan con un determinado criterio, y hay consultas especificas que han sido elaboradas con el fin de brindar mayor información y rapidez en sus operaciones al usuario.

MANTENIMIENTOS

Los mantenimientos permiten al usuario modificar valores o datos en la base por cada uno de los elementos representados. La información de los

objetos que son representados gráficamente debe ingresarse directamente, usando las formas de ingreso que hemos elaborado, desde *Microstation*. Esto debe hacerse así para guardar concordancia, puesto que no debe existir información de un elemento gráfico que no existe en un mapa, del mismo modo la eliminación del registro asociado a un objeto gráfico debe realizarse en conjunto con la eliminación de dicho objeto del mapa correspondiente.

Para los elementos que no han sido representados gráficamente, como es el caso de medidores, subestaciones, sectores, estados de los dispositivos, clientes y conexiones ilegales, hay opciones para agregar y eliminar registros, además de las opciones para modificar información que no sea parte de la clave primaria. Sin embargo, la eliminación de registros se hace conservando la integridad referencial de la base, es decir no puede eliminarse un registro que esté asociado con otras entidades, antes deben modificarse dichas relaciones con ayuda de las opciones de los mantenimientos.

REPORTES

Los reportes pueden presentar infromación por pantalla o por impresora. La información que presentan son básicamente consultas estáticas, es decir que el ususario sólo podrá ver los reportes ya elaborados y no podrá modificarlos.

CAPÍTULO II

MANUAL DEL USUARIO

2.1 GUÍA DE CONFIGURACIÓN DE MICROSTATION

Para poder enlazar por primera vez MicroStation con una base de dBase debemos realizar los siguientes pasos.

En primer lugar escogemos la opción *Configuration* del menú *Workspace*. Ante nosotros se presentará una ventana como la siguiente.

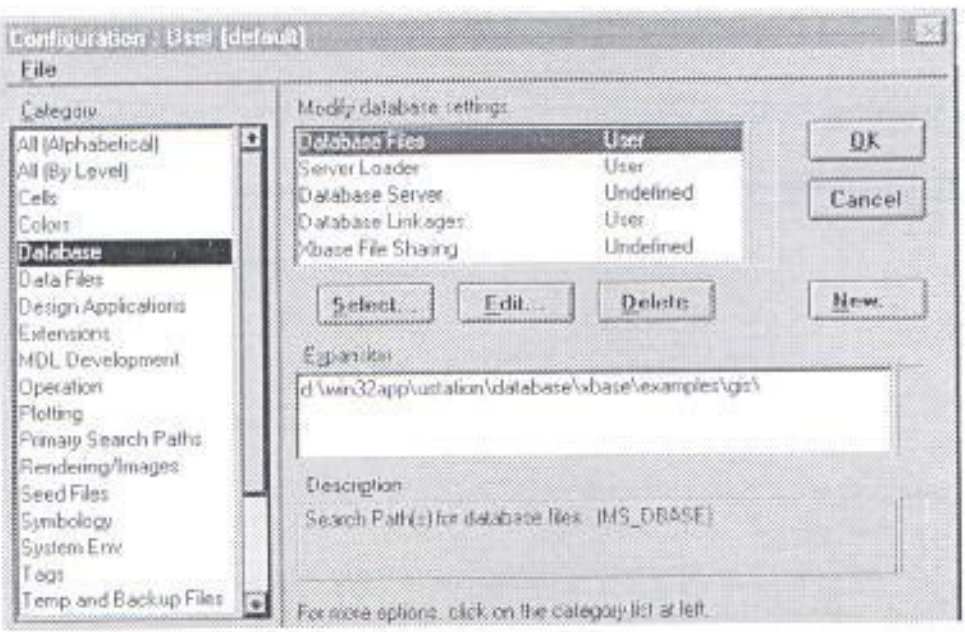


Figura 1: MicroStation's Workspace Configuration, Database

En la lista de categorias (*Category*), seleccionamos *Database*. En la lista superior derecha debemos configurar tres opciones. La primera de ellas es la opción *Database Files*. Con esta opción elegida, damos un click en el botón *Select* y en la caja de diálogo que aparece seleccionamos la ruta completa del directorio de la base de datos y lo añadimos (botón *Add*). Finalizamos la operación con el botón *Done*. La segunda opción que debemos configurar es *Server Loader*, para esto hacemos click en el botón *Select* y bajo el directorio *asneeded* de MicroStation (c:\win32app\ustation\mdlsys\asneeded\) escogemos como aplicación a cargar *dbload.ma* . La última opción para esta categoria es *Database Linkages*, para configurarla la seleccionamos y haciendo un click en el botón *Edit* nos cercioramos de que la opción *xbase* figure en la lista, si no la añadimos nosotros manualmente. Ahora deberemos seleccionar otra categoria (*Category*), en este caso *Design Applications*.

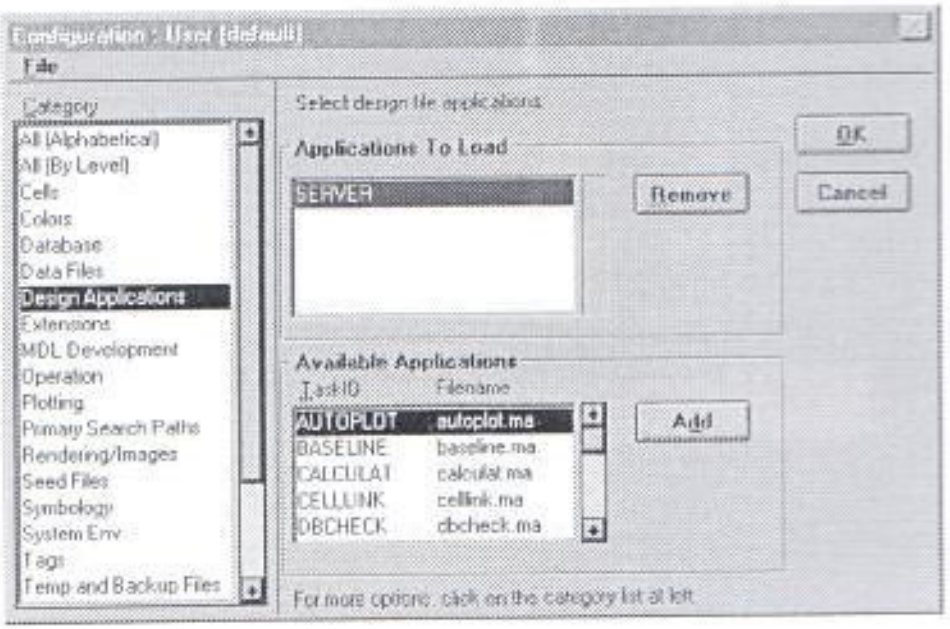


Figura 2: MicroStation's Workspace Configuration , Design Applications

En la lista superior derecha deberia aparecer marcada en azul la aplicación *SERVER*. Si no aparece, la forma de remediar esto es seleccionando *All* como categoria y buscar la variable *MS_DGNAPPS* como se muestra en la figura. Una vez seleccionada la variable escogemos el botón *Select* y cargamos la aplicación *server.ma* que se halla en el directorio *asneeded* (*c:\win32app\ustation\mdlsys\asneeded*) .

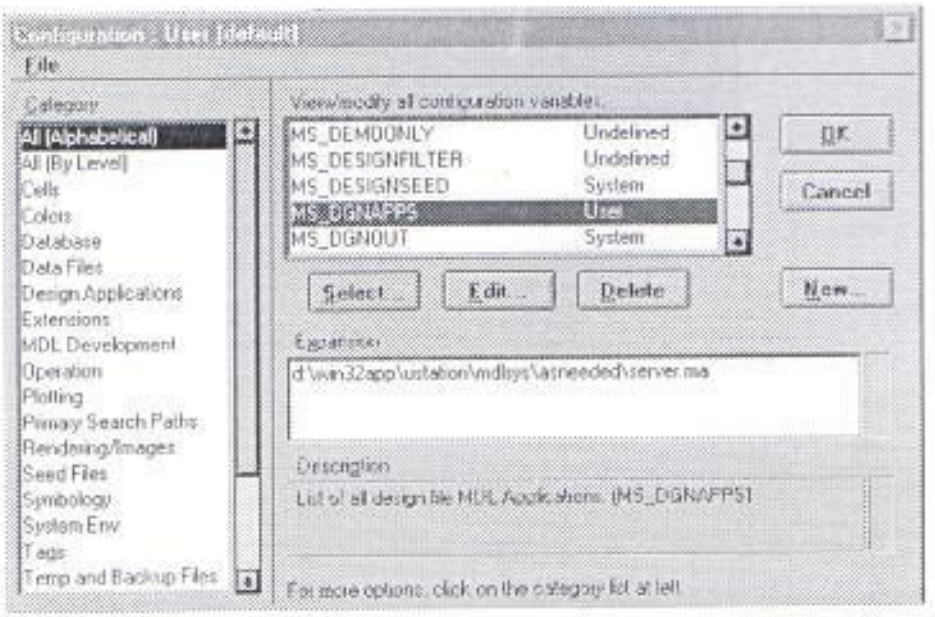


Figura 3: MicroStation Workspace Configuration, All

Finalmente, terminada la configuración aceptamos los cambios haciendo un click en el botón *OK*.

De aquí en adelante cada vez que queramos conectarnos a una base de dBase sólo tendremos que escoger la opción *Connect* del submenú *Database* del menú *Settings*, tal como se muestra en la figura.

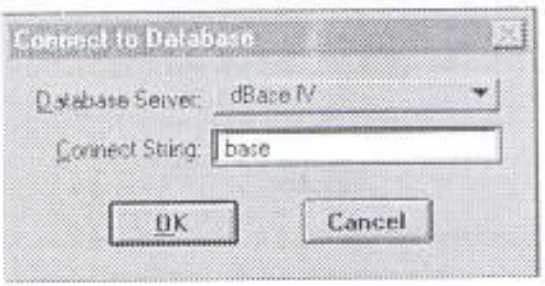


Figura 4: MicroStation's Database Connection

En la opción *Database Server* deberemos elegir *dBase IV* y en *Connect String* debe ir el nombre del último subdirectorio en donde se encuentre la base. Por ejemplo si la base de dBase se encuentra en c:\Applications\Emelsig\Base, entonces el *Connect String* es *Base* que es el nombre del último subdirectorio. Finalmente pulsamos el botón *OK* y esperamos a ver el mensaje de "*Database: Base*" en la esquina inferior derecha de MicroStation. Esto indica que la conexión fue exitosa.

2.2. GUÍA DE INSTALACIÓN DE LA APLICACIÓN

WINDOWS 95

Esta primera parte del manual cubre la instalación del programa "Sistema de Inventario para EMELEC" bajo un ambiente Windows 95. Los pasos a seguir son:

1. Coloque el diskette con la etiqueta **"Instalador Windows 95 Disco 1/3"** en su unidad de discos flexibles (por ejemplo la unidad **A**:).
2. Seleccione la opción **Ejecutar** del **Menú Inicio** y elija el programa **setup.exe** que se encuentra en el disco que acaba de colocar. Dé un click en el botón **Aceptar.**

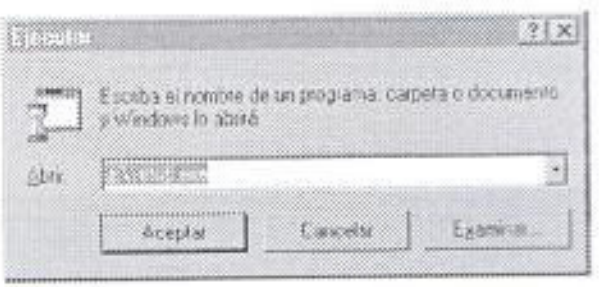


Figura 5: Instalación de la Aplicación bajo Windows 95, Setup.exe

3. A continuación se ejecutará el programa de instalación de la aplicación. Siga las instrucciones que le sean dadas en pantalla.

4. Una vez finalizada la instalación será necesario crear dos orígenes de datos en forma manual. Para esto hacemos doble click sobre el ícono **ODBC** del **Panel de Control**.

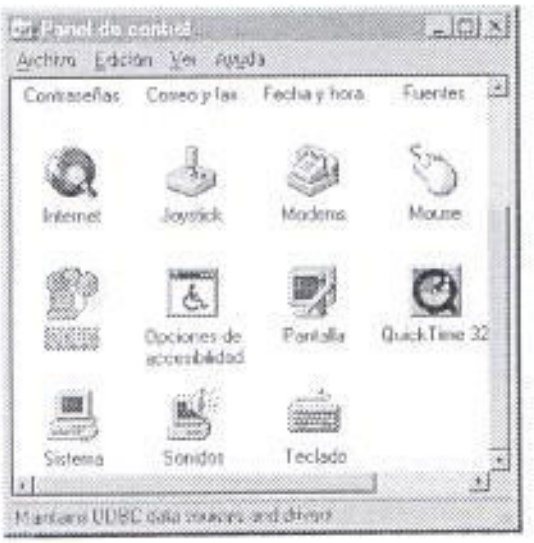


Figura 6: Instalación de la Aplicación bajo Windows 95, ODBC

5. En la pantalla que aparece seleccionamos la opción **Add** (Agregar).

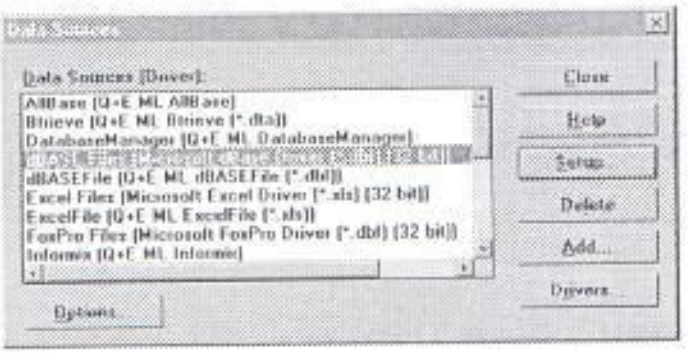


Figura 7: Instalación de la Aplicación bajo Windows 95, Agregar Origen de Datos

6. El primer origen de datos a crear lo hacemos usando el driver para dBase de Microsoft.

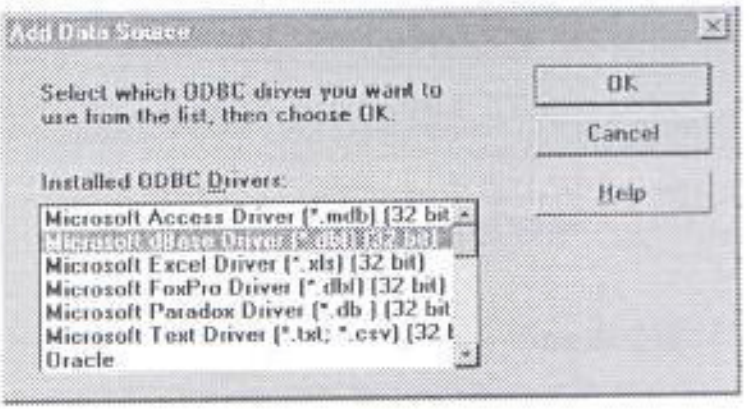


Figura 8: Instalación de la Aplicación bajo Windows 95, Microsoft Driver

7. Colocamos los datos como se muestra a continuación y antes de aceptar seleccionamos el directorio de la base (opción **Seleccionar directorio...**), el cual debe estar bajo el directorio de la aplicación que acabamos de instalar. Por ejemplo si la aplicación fue instalada en el directorio *emelsig* , en la opción **Seleccionar directorio...** debemos escoger c:\emelsig.

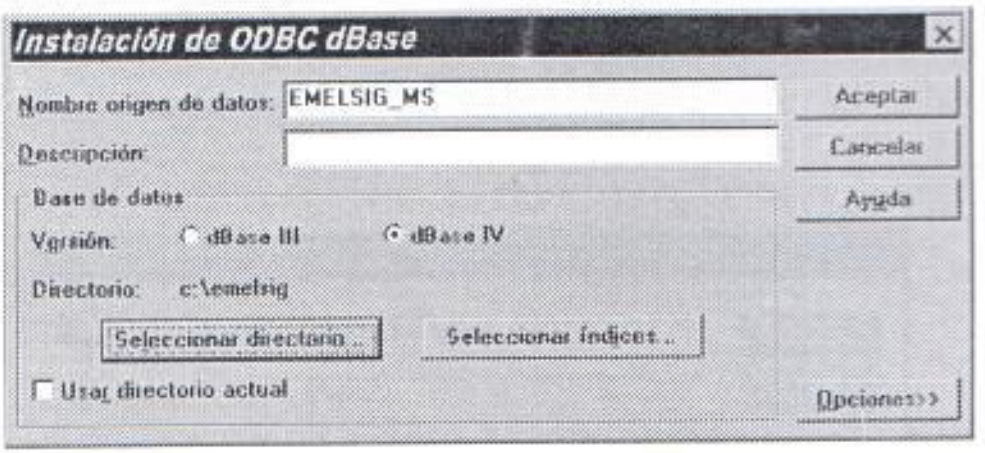


Figura 9: Uso de MicroStation, dBase Driver

8. El segundo origen de datos usa el driver para dBase de Q+E.

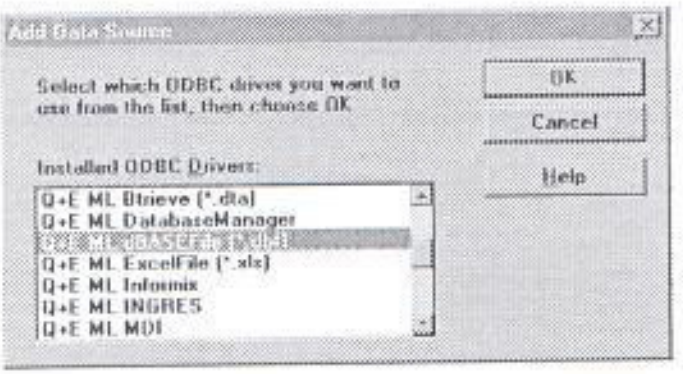


Figura 10: Instalación de la Aplicación bajo Windows 95, Q+E Driver

9. Los datos se colocan exactamente como se muestra en la figura. El comentario en este caso es muy importante. Una vez más el directorio de la base se encuentra en el directorio de la aplicación, por ejemplo

c:\emelsig Con esto se finalizan las configuraciones necesarias para poder ejecutar la aplicación.

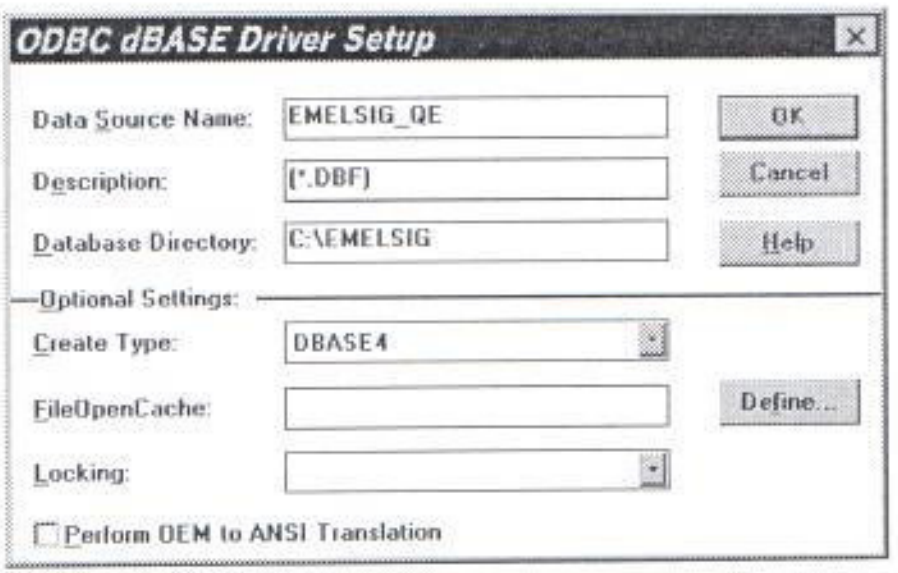


Figura 11: Instalación de la Aplicación bajo Windows 95, Q+E Driver Configuration

WINDOWS NT

Esta última parte del manual cubre la instalación del programa "Sistema de Inventario para EMELEC" bajo un ambiente Windows NT. Los pasos a seguir son:

1. Coloque el diskette con la etiqueta **"Instalación WINDOWS NT Disco 1/3"** en su unidad de discos flexibles (por ejemplo la unidad **A**:).
2. Seleccione la opción **Ejecutar** del **Menú Inicio** y elija el programa **setup.exe** que se encuentra en el disco que acaba de colocar. Dé un click en el botón **Aceptar.**

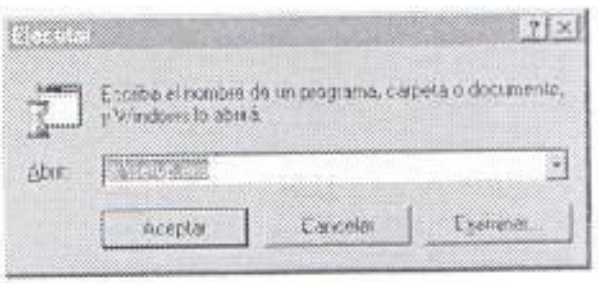


Figura 12: Instalación de la Aplicación bajo Windows NT, Setup.exe

3. A continuación se ejecutará el programa de instalación de la aplicación. Siga las instrucciones que le sean dadas en pantalla.
4. Para definir el origen de los datos existen dos archivos de texto ODBC.TXT y ODBCINST.TXT que se encuentran en el directorio

ODBC del diskette "**Instalación WINDOWS NT Disco 3/3**" ; estos archivos de texto contienen unas entradas ODBC que deben ser respectivamente añadidas de forma manual en los archivos ODBC.INI Y ODBCINST.INI del directorio WINNT

2.3. GUÍA DE USO DE MICROSTATION

Esta sección comprende el ingreso de elementos gráficos como transformadores, postes, circuitos de alta y de baja tensión, con sus respectivos datos.

Antes de comenzar a ingresar cualquier elemento gráfico, se deben conocer los siguiientes conceptos:

CAPAS: Son los diferentes niveles en los que opera microestación, por ejemplo si queremos definir postes debemos crear una capa para ellos.

CELDAS: Son los gráficos que representan a los elementos gráficos.

2.3.1. INGRESO DE ELEMENTOS GRÁFICOS

A continuación se detallan los pasos para ingresar elementos gráficos:

1. En el menú Settings, escoger Locks y seleccionar Full.

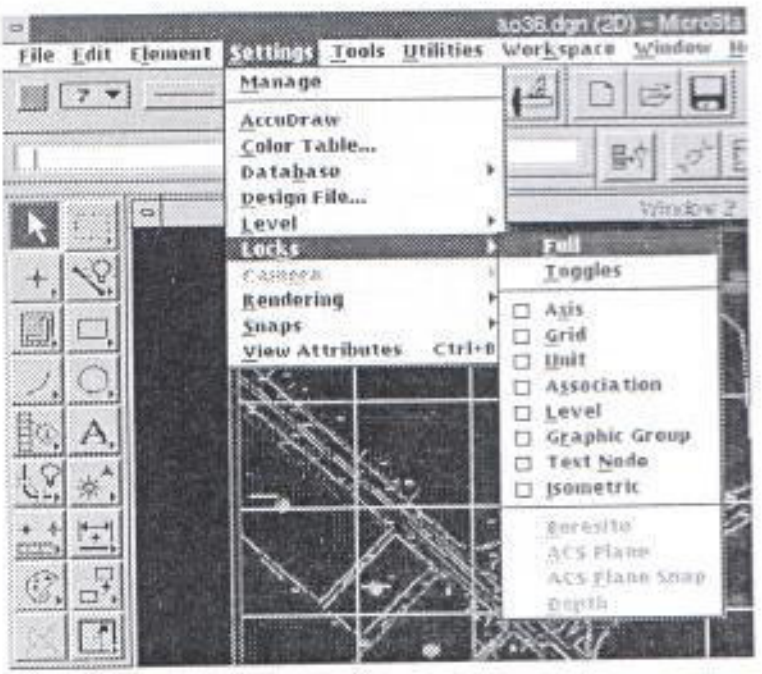


Figura 13: Uso de MicroStation, Menú Settings - Locks

2. A continuación se presenta la pantalla Locks dentro de la cual en la sección *Snap* debe setear Mode en *Origin*.

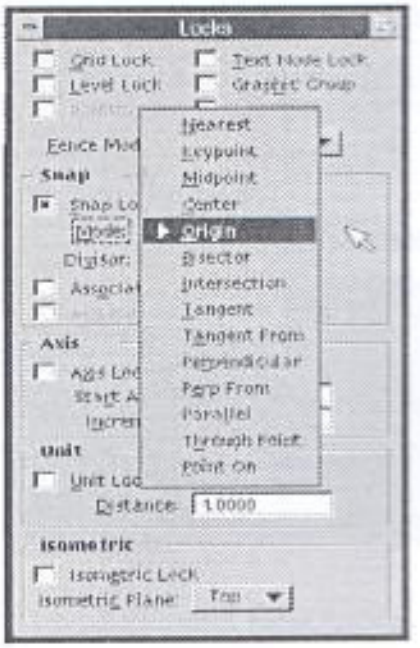


Figura 14: Uso de MicroStation, Locks

3. En el menú Settings escoja la opción Level y seleccione Names.., posteriormente se presentará la panttalla Level Names.

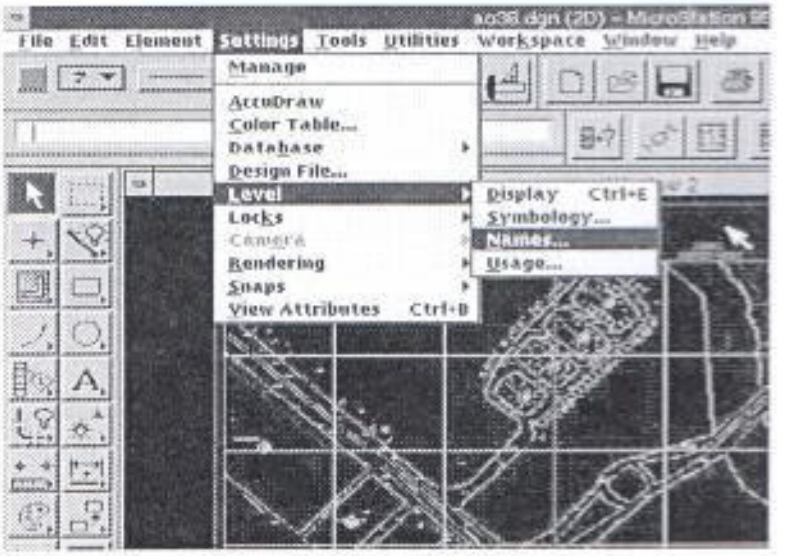


Figura 15: Uso de MicroStation, Menú Settings - Level

4. En esta pantalla en el menú File escoja la opción open, a continuación se presentará una caja de diálogo, donde deberá escoger un archivo con extensión *.lvl*. Este archivo debe contener todas las definiciones de capas sobre las cuales se va a graficar los elementos.

Figura 16: Uso de MicroStation, Level Names

2.3.2. GRAFICACIÓN DE CELDAS

1. El elemento a graficarse debe pertenecer a una capa, la cual debe ser puesta como activa. Para activar la capa, posicione el mouse en la esquina superior izquierda y seleccione Active Level y luego escoja la capa.

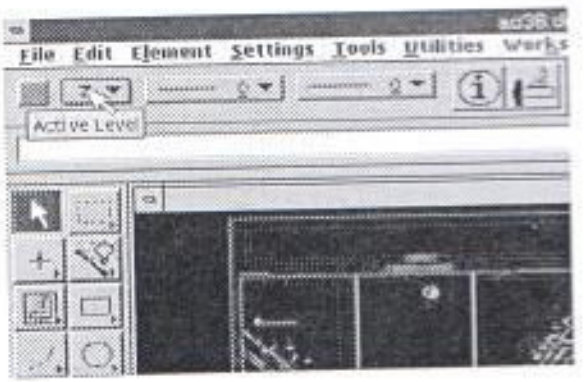


Figura 17: Uso de MicroStation, Active Level

2. La capa debe tener un color. Para darle color a la capa, posicione el mouse en la esquina superior izquierda y seleccione Active Color y luego escoja el color.

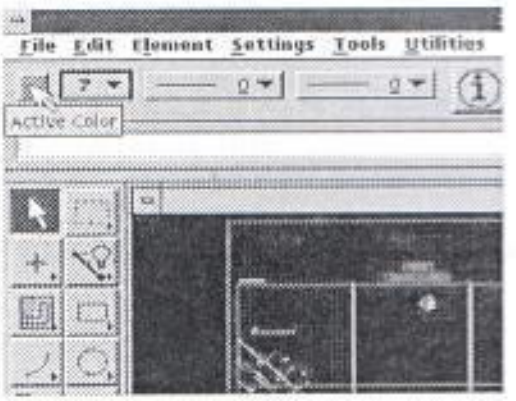


Figura 18: Uso de MicroStation, Active Color

3. En el menú Element escoger la opción Cells.

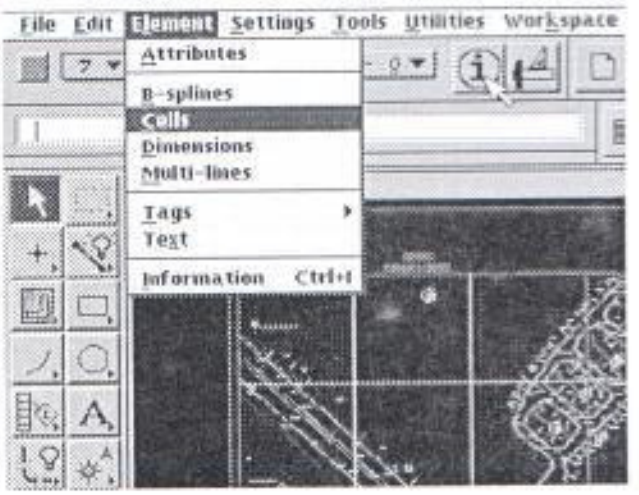


Figura 19: Uso de MicroStation, Menú Element - Cells

4. A continuación se presenta la pantalla Cell Library. En el menú File escoja Attach, se presentará una caja de diálogo donde deberá escoger un archivo con extensión .cel. Este archivo contiene todas las celdas que serán utilizadas para representar a los elementos.

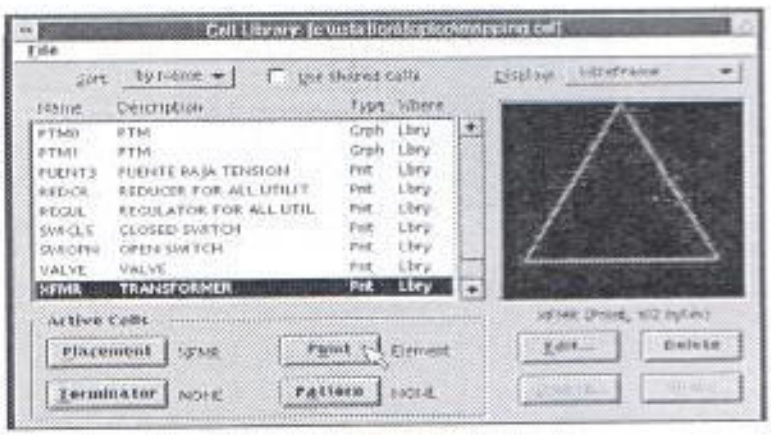


Figura 20: Uso de MicroStation, Cells

5. Escoja la celda que desee utilizar en ese momento luego haga click en Point.
6. En La barra de herramientas escoger *Place Active Point*, de esta forma queda activada la celda seleccionada en el paso anterior, para poder graficar el elemeto, es decir la celda, debe hacer click en el lugar

donde quiere que aparezca, por ejemplo en el caso del transformador, postes y puentes.

Figura 21: Uso de MicroStation, Place Active Point

2.3.3. GRAFICACIÓN DE CIRCUITOS

Debe realizar los pasos 1 y 2 de Graficación de celdas.

1. Debe selecionar un estilo de linea, para ello, posicione el mouse en la parte superior izquierda y escoja Active Line Style , luego seleccione la linea cero, para el caso de circuitos secundadios y linea 4 en el caso de ciruitos primarios.

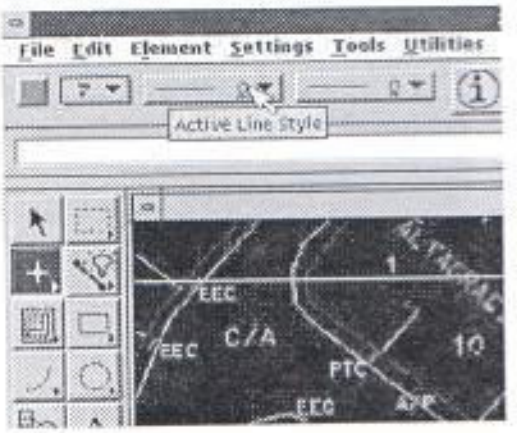


Figura 22: Uso de MicroStation, Active Line Style

2. En la barra de herramienta escoja Place Smartline y posiciónese en el lugar del mapa desde donde quiera empezar la linea. Si se desea empezar la linea desde el

origen de una celda, posiciónese sobre la celda y haga click con los dos botones del mouse al mismo tiempo; al hacer esto la celda se pondrá en color plomo y luego con el botón izquierdo acepte, si quiere rechazar haga click con el botón derecho.

El mismo procedimiento se emplea para determinar el fin de una línea.

Figura 23: Uso de MicroStation, Place Smartline

2.3.4. INGRESO DE DATOS

Cuando se va a ingresar información perteneciente a un elemento gráfico se debe primero conectar a la base de datos, lo cual está especificado en la Guia de configuración de Microstation. Posteriormente se realizan los siguientes pasos:

1. En el menú Tools seleccionar Database, de esta forma se presenta el database toolbar.

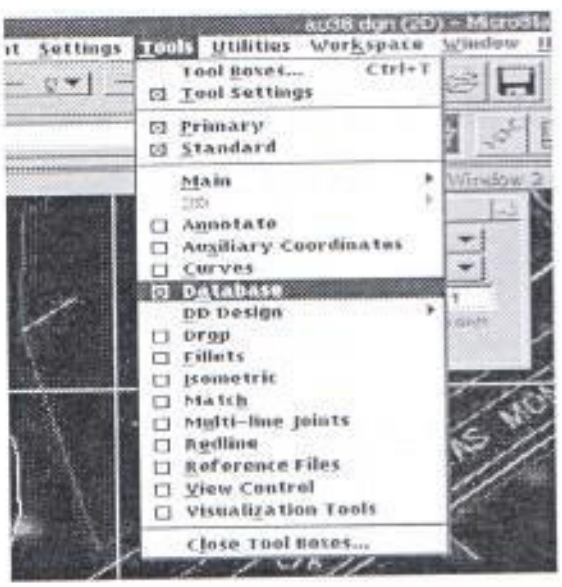


Figura 24: Uso de MicroStation, Menú Tools - Database

2. En el menú Settings escoja la opción Database y seleccione Dialog

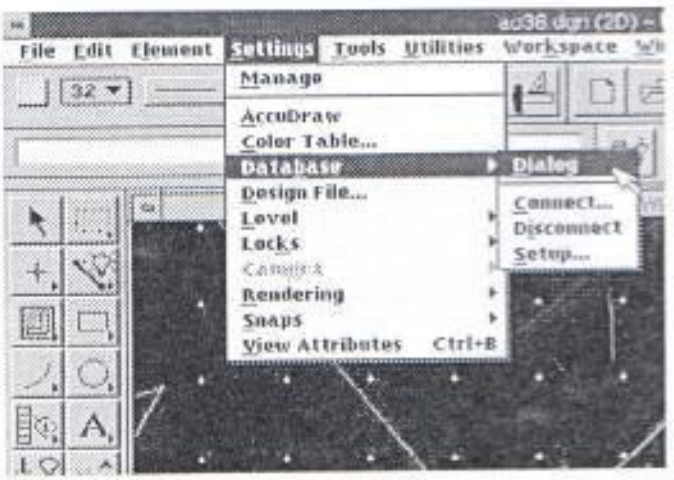


Figura 25: Uso de MicroStation, Menú Settings - Database Dialog

3. Para ingresar información de un elemento nuevo, debe setear dentro de la caja de Dialog el *Linkage Mode* en *New* y Form en Text screen.

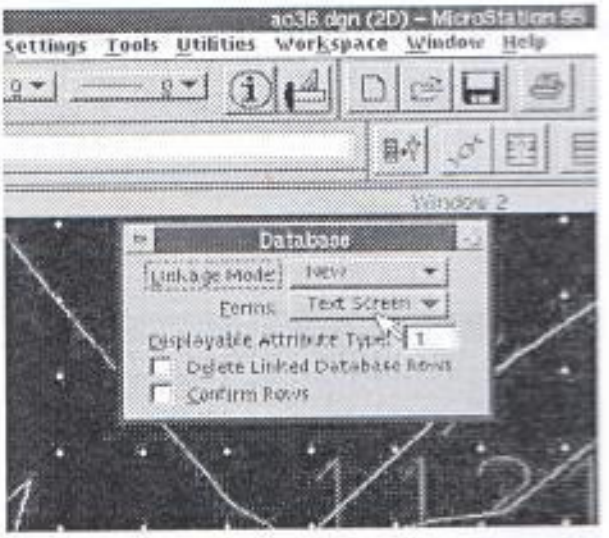


Figura 26: Uso de MicroStation, Database Linkage New

4. Seleccionar la entidad activa con la instrucción *find select * from (nombre de la tabla)* donde desea guardar la información. Esta instrucción es ejecutada desde el Key in (el mismo que es encontrado en el menú. Utilities)

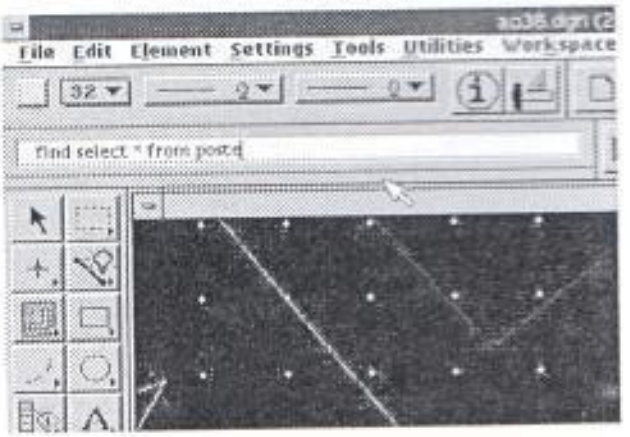


Figura 27: Uso de MicroStation, Selección de Entidad Activa

5. La información que va a ingresar se lo hace por medio de una forma o ventana la misma que es editada al ejecutar la instrucción *edit ae* en el Key in.

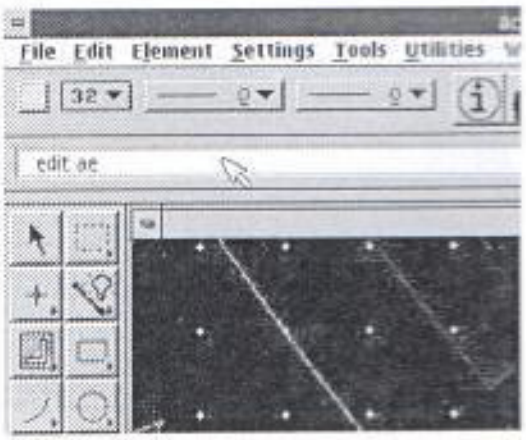


Figura 28: Uso de MicroStation, Edición de la Entidad Activa

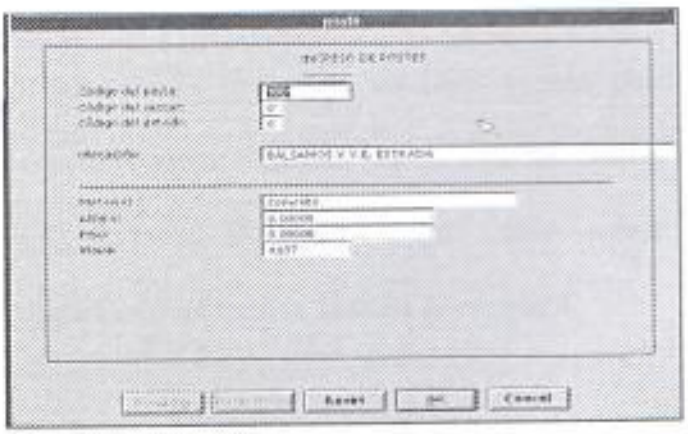


Figura 29: Uso de MicroStation, Ingresando nuevos datos

6. Seleccione Attach Active Entity del Database toolbar, luego con el mouse posiciónese en el elemento al que le desea atachar la información y haga click sobre él.

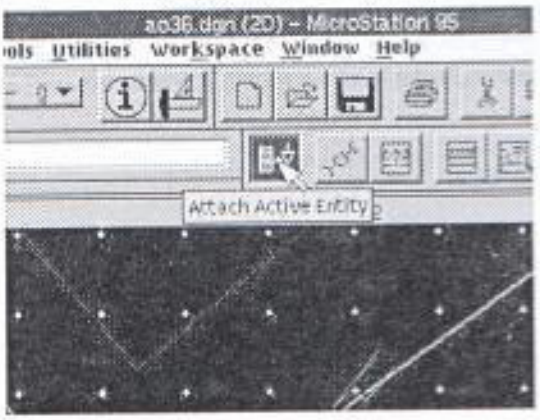


Figura 30: Uso de MicroStation, Attach Active Entity

7. Si la información ha sido relacionada a un objeto y esta misma información desea ser atachada a otro objeto, entonces debe setear dentro de la caja de Dialog el *Linkage Mode* en *Duplicate*, luego debe volver a seleccionar la entidad activa descrita en el paso 4.

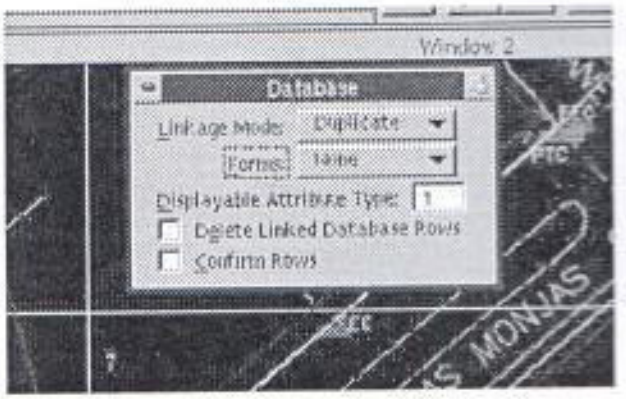


Figura 31: Uso de MicroStation, Database Linkage Duplicate

8. Otra forma como puede seleccionar la entidad activa es escogiendo del database toolbar el icono Define Active Entity Graphically y luego haciendo click sobre el elemento del cual su registro se deseasetear como activo.

Los pasos anteriores de seteo de una capa activa, color activo, celda activa, o entidad activa se los puede realizar por medio de short cut keys. Para ello seleccione la opción de menú Workspace y escoja Function Keys...

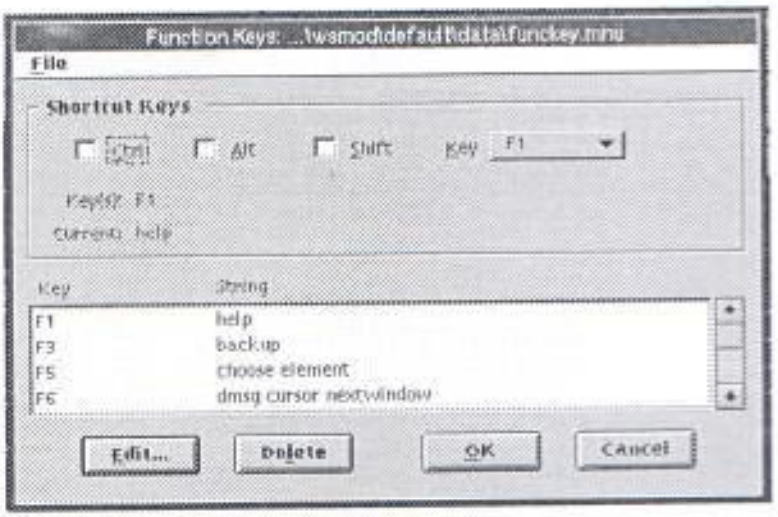


Figura 32: Uso de MicroStation, Function Keys

Luego en la pantalla Funtcion Keys se pueden añadir , modificar o eliminar short cut keys, luego presione OK. Estos cambios se guardan en un archivo .mnu, que puede luego ser usado.

2.4. GUÍA DE USO DE LA APLICACIÓN

2.4.1. MANTENIMIENTO

Las opciones de mantenimiento están agrupadas bajo la opción de menú del mismo nombre en la aplicación, como se muestra en la figura.

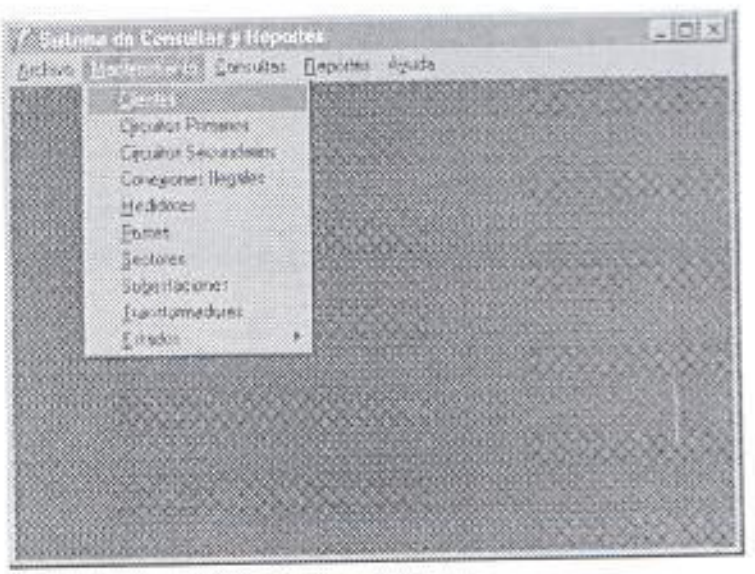


Figura 33: Aplicación, Menú de Mantenimiento

Como ya se dijo anteriormente, estas opciones permiten actualizar datos de las tablas, agregar y eliminar registros de entidades que no son enlazadas gráficamente como en el caso

de subestaciones, medidores, sectores, conexiones ilegales y estados de los dispositivos. Para hacer la actualización más rápida se han añadido opciones de búsqueda en la mayoría de los casos.

Veamos una ventana típica de mantenimiento:

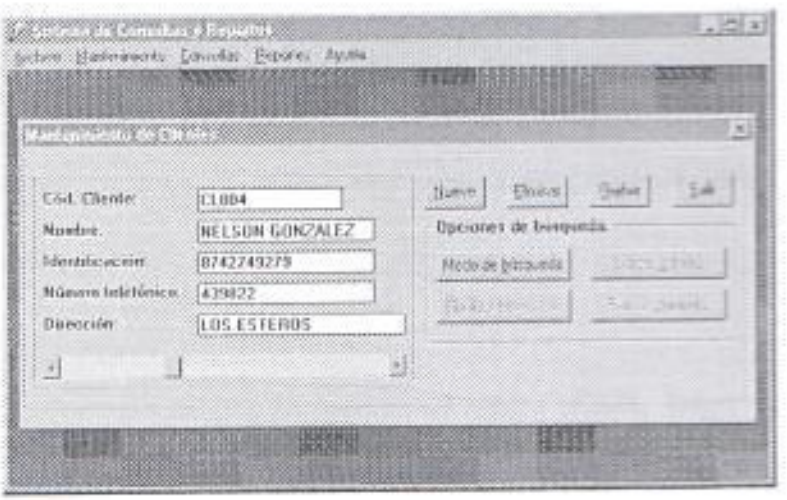


Figura 34: Aplicación, Mantenimiento de Clientes

En una ventana típica de mantenimiento hay un *scrollbar* que permite al usuario desplazarse por los registros, uno a uno. Hay varios botones, dos de ellos comunes a todos los mantenimientos: es el caso de *Grabar* y *Salir*. En cualquier caso podremos modificar el contenido de cualquier campo no clave de un registro, para hacer efectivo el cambio basta con

pulsar el botón *Grabar*. El botón *Salir* como su nombre indica sale del mantenimiento actual.

Otros botones que pueden o no estar presentes son los botones *Nuevo* y *Eliminar*. Estarán presentes en todos los mantenimientos de entidades que no tienen enlace gráfico como se indicó anteriormente. Revisemos el caso de agregar un nuevo registro, para esto deberemos pulsar el botón *Nuevo*.

Veamos un ejemplo de agregar un nuevo cliente, luego de pulsar el botón *Nuevo* obtendremos algo como lo siguiente:

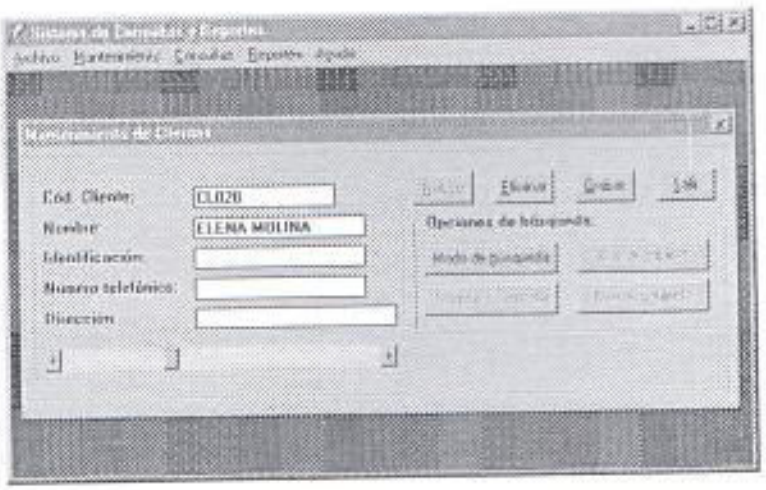


Figura 35: Aplicación, Mantenimiento - Agregar un Cliente

Cuando se ingresa un nuevo registro se blanquean todos los campos para que podamos escribir los nuevos datos y se deshabilita el botón *Nuevo*. Luego que terminemos de ingresar los datos deberemos usar la opción de *Grabar* para ingresar el nuevo registro.

En ocasiones se cometen errores y se ingresa un registro equivocado o en el caso de los clientes puede ser que alguien deje de serlo, para estos casos está el botón *Eliminar*. Si queremos eliminar un registro basta con posicionarnos sobre él con el *scrollbar* y pulsar el botón *Eliminar*.

Cuando realicemos esto obtendremos una pantalla como la siguiente:

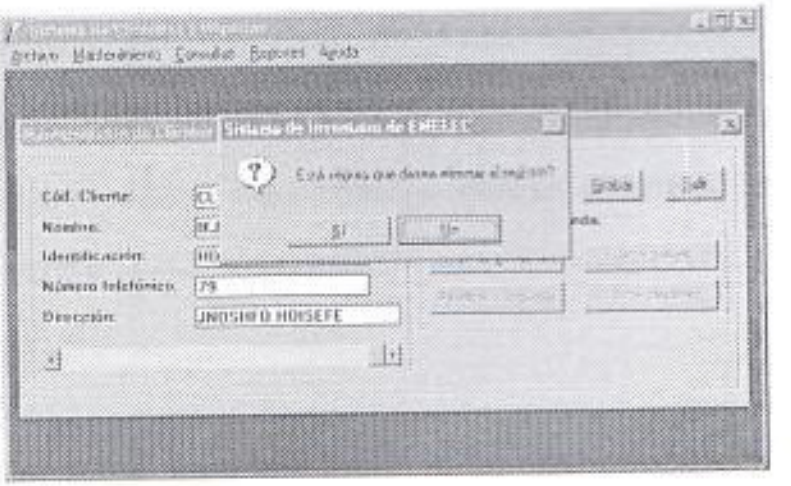


Figura 36: Aplicación, Mantenimiento - Eliminar un Cliente

Se nos preguntará si estamos seguros de querer eliminar el registro, si la respuesta es *Si* el registro es eliminado de la tabla, si la respuesta es *No* entonces todo queda inalterado.

En todos los mantenimientos a excepción de *sectores* y *estados* existen *Opciones de Búsqueda*. La razón por la cual en los mantenimientos mencionados no existen estas opciones es porque en estas entidades el número de registros tiende a ser pequeño y es mucho más rápido navegar con el *scrollbar* que realizar una búsqueda.

Para entrar en *Modo de Búsqueda* se debe pulsar el botón del mismo nombre, ante lo cual obtendremos una pantalla en la que los campos están en blanco como se muestra en la siguiente figura. Aqui podemos escribir el criterio de búsqueda en uno o más campos.

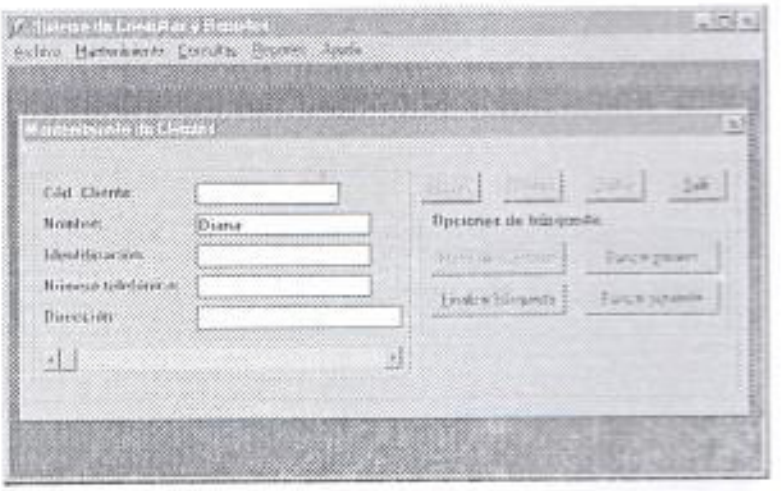


Figura 37: Aplicación, Mantenimiento - Búsqueda de un string en el campo Nombre de la tabla Clientes

Luego de ingresar el criterio de búsqueda para obtener el resultado tenemos dos opciones: *Buscar primero* y *Buscar siguiente*. Como es fácil de deducir la opción *Buscar primero* busca desde el principio de la tabla y devuelve el primer registro que cumple con el criterio especificado. La opción *Buscar Siguiente* busca a partir del registro actual el primero que cumpla con el criterio. Normalmente se pulsa primero la opción *Buscar primero* y luego se procede a pulsar cada vez el botón *Buscar siguiente*, hasta encontrar el registro que queremos revisar. Para finalizar el modo de búsqueda pulsamos el botón *Finalizar búsqueda*. A continuación se

muestra el resultado de pulsar el botón *Buscar primero* para el criterio anterior.

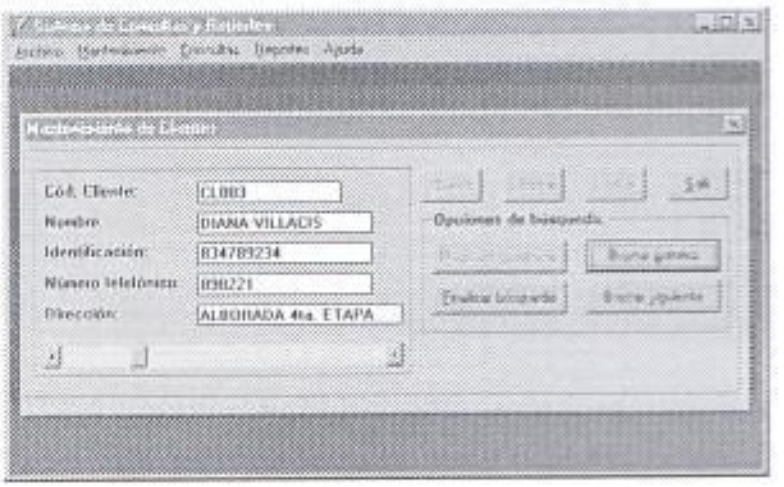


Figura 38: Aplicación, Mantenimiento - Primer registro resultado de la búsqueda

Lo hasta aquí explicado se cumple para todos los mantenimientos que cuenten con las opciones explicadas. Falta por indicar que existen mantenimientos en que el valor de un determinado campo está determinado por el valor de un *combo*, si se quiere cambiar el valor de un campo de este tipo basta con seleccionar el valor correspondiente del *combo* si este existiere, o con escribir el nuevo valor que se quiera darle al campo en la caja de texto del *combo*. En la figura siguiente se muestra un ejemplo de este tipo.

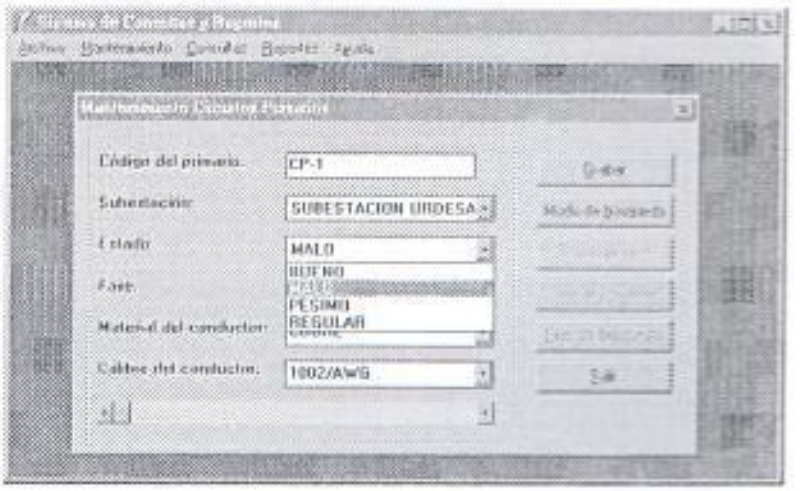


Figura 39: Aplicación, Mantenimiento - Valor de un campo en un combo

2.4.2. CONSULTAS

Las opciones de consulta están agrupadas bajo el menú del mismo nombre en la aplicación, tal como se muestra en la figura.

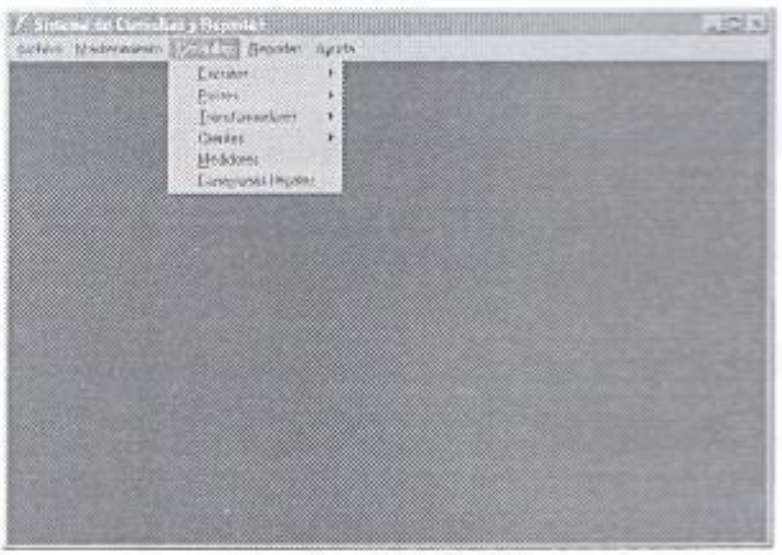


Figura 40: Aplicación, Menú Consultas

En este menú encontramos opciones para realizar consultas generales y especificas sobre los diferentes elementos representados en el modelo. Veamos un ejemplo de consulta, para obtener una *Consulta de Dispositivos por Poste* seleccionamos dicha opción del submenú *Postes* del menú *Consultas*.

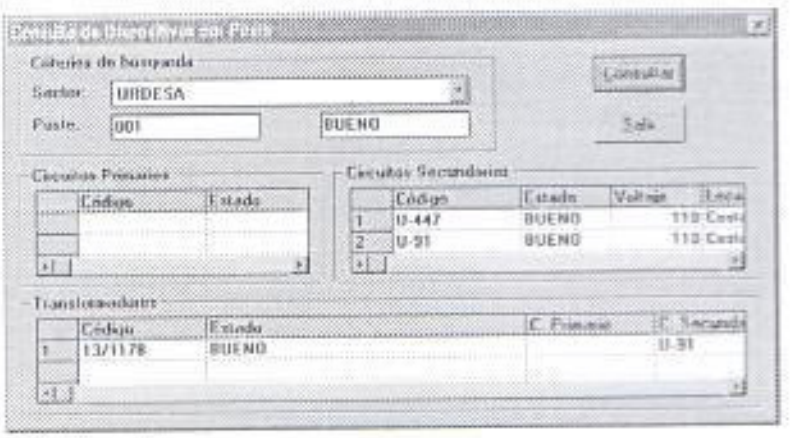


Figura 41: Aplicación, Consultas - Consulta de Dispositivos por Poste

En una consulta tipica existen varios criterios de búsqueda por los cuales se puede preguntar. En el ejemplo anterior se hizo una consulta por sector y código de poste. Veamos otro ejemplo de consulta, esta vez una consulta general de postes

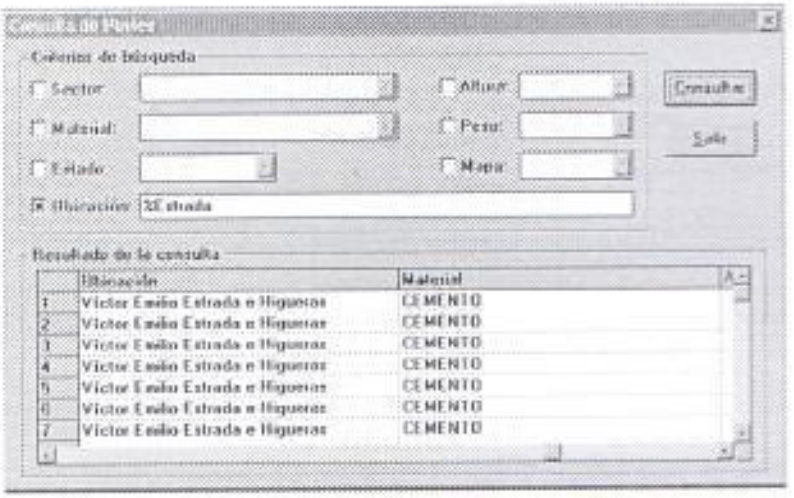


Figura 42: Aplicación, Consultas - Consulta general de postes

En este ejemplo se hizo una búsqueda de los postes que tuviesen en su campo ubicación la cadena de caracteres *Estrada*, para hacer una búsqueda aproximada se debe colocar al inicio de la cadena el carácter %.

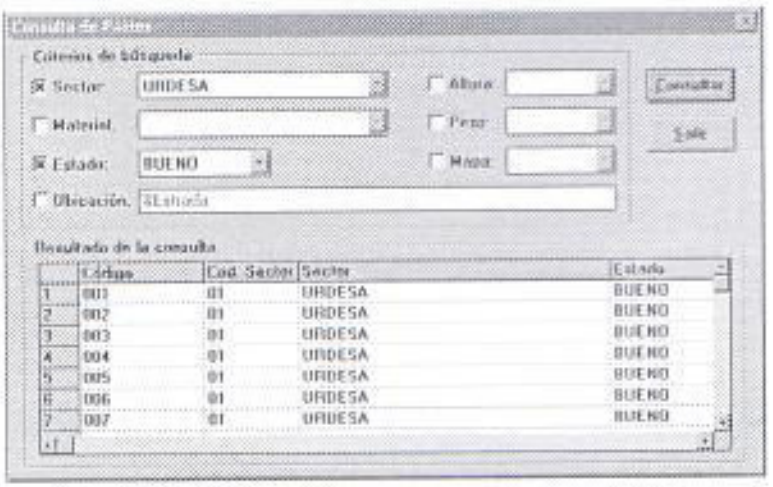


Figura 43: Aplicación, Consultas - Consulta de postes

Veamos otro ejemplo para la misma consulta de postes, esta vez se ha realizado una consulta por *sector* y por *estado*. Como siempre los registros resultantes se muestran en un grid en la parte inferior de la ventana.

Todas las consultas funcionan en la misma forma que ha sido descrita para los ejemplos anteriores.

2.4.3. REPORTES

Los reportes pueden presentar infromación por pantalla o por impresora. La información que presentan son básicamente consultas estáticas, es decir que el ususario sólo podrá ver los reportes ya elaborados y no podrá modificarlos.

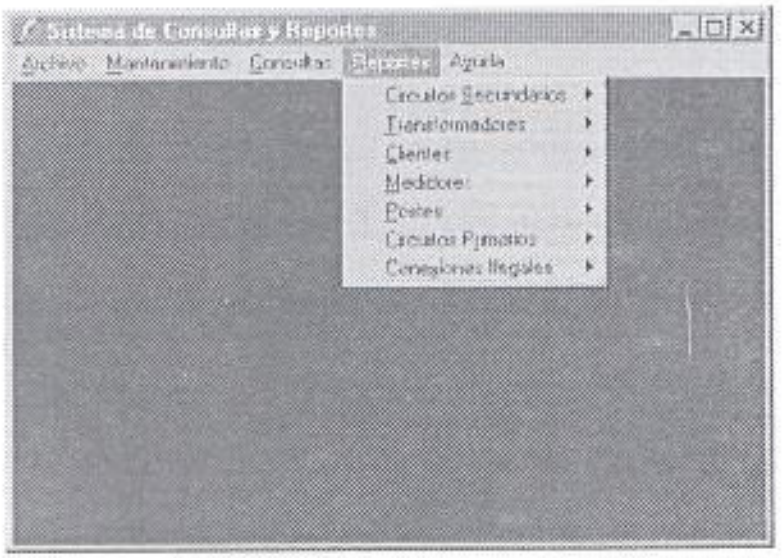


Figura 44: Aplicación, Menú Reportes

Los reportes realizados están basados en las necesidades del usuario y cada uno de ellos está agrupado o clasificado por factores como el sector y estado de manera general además

de otros factores que dependen del reporte que se esté observando.

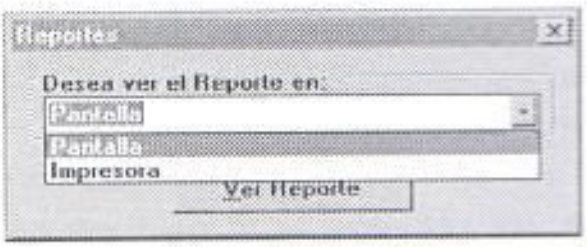


Figura 45: Aplicación, Reportes - Ventana de Reportes

Los reportes permiten obtener además de datos de inventario actualizados, información útil para determinar si es necesario reemplazar elementos o realizar expansiones. Un ejemplo de la utilidad de los reportes es en el caso del reporte de medidores por transformador, en este caso se obtiene la capacidad del transformador y se ve la carga que tiene cada medidor, con esto se puede determinar si el transformador está sobrecargado y decidir si hay que colocar uno de mayor capacidad.

A continuación se muestra el reporte de los Medidores por Transformadores:

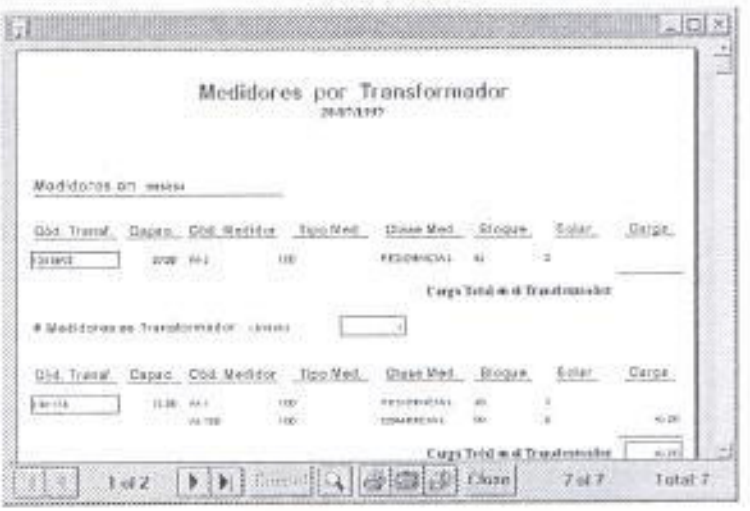


Figura 46: Aplicación, Reportes - Reporte de Medidores por Transformador

CAPÍTULO III

DISEÑO DE LA BASE DE DATOS

3.1 DIAGRAMA DE FLUJO DE DATOS

Figura 2: Diseño de la Base de Datos, Diagrama de Flujo de Datos

3.2. DIAGRAMA MODULAR DEL SISTEMA

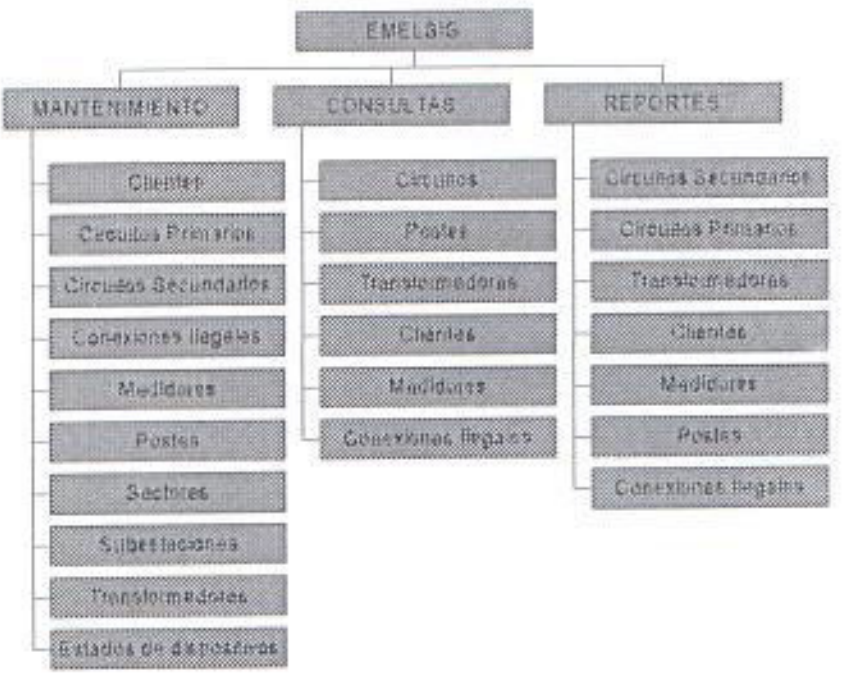
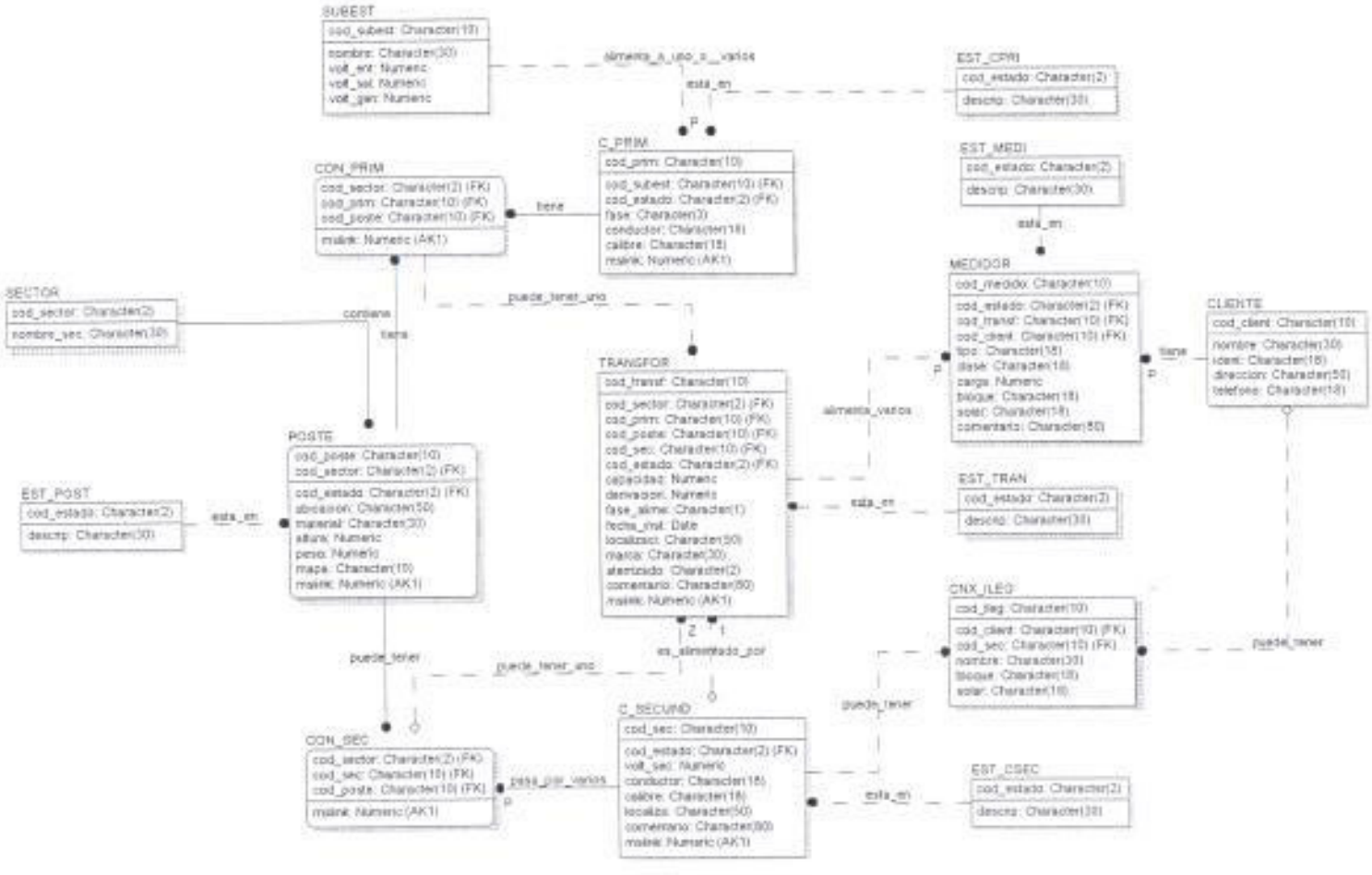


Figura 48: Diseño de la Base de Datos, Diagrama Modular del Sistema

3.3 MODELO ENTIDAD - RELACIÓN



3.4. DICCIONARIO DE DATOS

ENTIDAD: POSTE NOMBRE TABLA: POSTE

NOMBRE	DESCRIPCIÓN	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_POSTE	código del poste	CHARACTER	10	NOT NULL	PK
COD_SECTOR	sector en que se encuentra el poste	CHARACTER	2	NOT NULL	PK
COD_ESTADO	estado del poste	CHARACTER	2	NOT NULL 'B','M','R' *	FK
UBICACION	dirección	CHARACTER	50	NOT NULL	
MATERIAL	material de que está hecho el poste	CHARACTER	30	NOT NULL 'metal','madera','cemento'	
ALTURA	altura del poste en metros	NUMERIC		NOT NULL 9,12,15,18	
PESO	peso del poste en kilogramos	NUMERIC		NOT NULL 250, 500 Kg.	
MAPA	código del mapa	CHARACTER	10	NOT NULL	
MSLINK	clave para poder enlazar un registro desde MicroStation	NUMERIC		NOT NULL	

ENTIDAD: TRANSFORMADOR NOMBRE TABLA: TRANSFOR

NOMBRE	DESCRIPCION	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_TRANSF	código del transformador	CHARACTER	10	NOT NULL	PK
COD_SECTOR	código del sector	CHARACTER	2	NOT NULL	FK
COD_POSTE	poste en el que está colocado el transformador	CHARACTER	10	NOT NULL	FK
COD_SEC	código del circuito secundario al que puede alimentar el transformador	CHARACTER	10		FK
COD_PRIM	código del circuito	CHARACTER	10	NOT NULL	FK

	primario que alimenta al transformador				
COD_ESTADO	estado del transformador	CHARACTER	2	NOT NULL 'B','M','R' *	FK
CAPACIDAD	capacidad en KVA del transformador	NUMERIC		NOT NULL	
DERIVACION	porcentaje de derivación	NUMERIC		NOT NULL	
FASE_ALIME	fase de alimentación	CHARACTER	1	'A','B','C'	
FECHA_INST	fecha de instalación del transformador	DATE		NOT NULL	
LOCALIZA	dirección	CHARACTER	50	NOT NULL	
MARCA	marca del transformador	CHARACTER	30	NOT NULL	
ATERRIZADO	indica si el transformador está conectado a tierra	CHARACTER	2	SI NO	
COMENTARIO	comentarios adicionales	CHARACTER	80		

ENTIDAD: MEDIDOR **TABLA:MEDIDOR**

NOMBRE	DESCRIPCION	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_MEDIDO	código del medidor	CHARACTER	10	NOT NULL	PK
COD_ESTADO	estado del medidor	CHARACTER	2	NOT NULL 'B','M','R' *	FK
COD_TRANSF	código del transformador al que pertenece el medidor	CHARACTER	10	NOT NULL	FK
COD_CLIENT	código del dueño del medidor (cliente)	CHARACTER	10	NOT NULL	FK
TIPO		CHARACTER	18	NOT NULL	
CLASE		CHARACTER	18	NOT NULL	
BLOQUE	nombre del bloque o manzana del	CHARACTER	18	NOT NULL	

	medidor				
SOLAR	número del solar	CHARACTER	18	NOT NULL	
CARGA	carga del transformador	NUMERIC		NOT NULL	

ENTIDAD: CIRCUITO SECUNDARIO TABLA: C_SECUND

NOMBRE	DESCRIPCIÓN	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_SEC	Codigo del Circuito Secundario	CHARACTER	10	NOT NULL	PK
COD_ESTADO	Código del Estado del Circuito Secundario	CHARACTER	2	NOT NULL 'B','M','R' *	FK
VOLT_SEC	Voltaje que pasa por el circuito secundario (en Voltios)	NUMERIC		NOT NULL	
LOCALIZACI	Calles por donde pasa el circuito	CHARACTER	50	NOT NULL	
CONDUCTOR	tipo de conductor usado en el circuito	CHARACTER	18	NOT NULL	
CALIBRE	calibre del conductor	CHARACTER	18	NOT NULL	
COMENTARIO					
MSLINK	Clave para poder enlazar el registro en MicroStation	NUMERIC		NOT NULL	

ENTIDAD: CIRCUITO PRIMARIO TABLA: C_PRIM

NOMBRE	DESCRIPCION	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_PRIM	código del circuito primario	CHARACTER	10	NOT NULL	PK
COD_ESTADO	código del estado del circuito primario	CHARACTER	2	NOT NULL "B","M","R" *	FK
COD_SUBEST	código de la subestación	CHARACTER	10	NOT NULL	FK

FASE	fase del circuito primario	CHARACTER	1		
CONDUCTOR	tipo de conductor usado en el circuito	CHARACTER	18	NOT NULL	
CALIBRE	calibre del conductor	CHARACTER	18	NOT NULL	
MSLINK	clave para poder enlazar un registro desde Microstation	NUMERIC		NOT NULL	

ENTIDAD: SUBESTACIÓN TABLA: SUBEST

NOMBRE	DESCRIPCION	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_SUBEST	código de la subestación	CHARACTER	10	NOT NULL	PK
NOMBRE	nombre de la alimentadora	CHARACTER	30		
VOLT_ENT	voltaje de entrada	NUMERIC		NOT NULL	
VOLT_SAL	voltaje de salida	NUMERIC		NOT NULL	
VOLT_GEN	voltaje del generador	NUMERIC		NOT NULL	

ENTIDAD: CLIENTE TABLA: CLIENTE

NOMBRE	DESCRIPCIÓN	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_CLIENT	código del cliente	CHARACTER	10	NOT NULL	PK
NOMBRE	nombre del cliente	CHARACTER	30	NOT NULL	
IDENT	identificación del cliente	CHARACTER	18	NOT NULL no. cédula, pasaporte o ruc	
DIRECCION	dirección del cliente	CHARACTER	30	NOT NULL	
TELEFONO	número telefónico del cliente	CHARACTER	18		

ENTIDAD: SECTOR TABLA: SECTOR

NOMBRE	DESCRIPCION	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_SECTOR	código del sector	CHARACTER	2	NOT NULL	PK
NOMBRE	nombre del sector	CHARACTER	30	NOT NULL	

ENTIDAD: ESTADO DEL CIRCUITO PRIMARIO TABLA: EST_CPRI

NOMBRE	DESCRIPCIÓN	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_ESTADO	código del estado del circuito primario	CHARACTER	2	NOT NULL "B", "M", "R" *	PK
DESCRIP	descripción del estado	CHARACTER	30	NOT NULL	

ENTIDAD: ESTADO DEL CIRCUITO SECUNDARIO TABLA: EST_CSEC

NOMBRE	DESCRIPCIÓN	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_ESTADO	código del estado del circuito secundario	CHARACTER	2	NOT NULL "B", "M", "R" *	PK
DESCRIP	descripción del estado	CHARACTER	30	NOT NULL	

ENTIDAD: ESTADO DEL TRANSFORMADOR TABLA: EST_TRAN

NOMBRE	DESCRIPCIÓN	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_ESTADO	código del estado del transformador	CHARACTER	2	NOT NULL "B", "M", "R" *	PK
DESCRIP	descripción del estado	CHARACTER	30	NOT NULL	

ENTIDAD: ESTADO DEL POSTE TABLA: EST_POST

NOMBRE	DESCRIPCION	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE

COD_ESTADO	código del estado del poste	CHARACTER	2	NOT NULL "I", "M", "R" *	PK
DESCRIP	descripcion del estado	CHARACTER	30	NOT NULL	

ENTIDAD: CONECTOR PRIMARIO TABLA: CON_PRIM

NOMBRE	DESCRIPCIÓN	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_SECTOR	código del sector	CHARACTER	2	NOT NULL	FK
COD_PRIM	código del circuito primario	CHARACTER	10	NOT NULL	FK
COD_POSTE	código del poste	CHARACTER	10	NOT NULL	FK
MSLINK	clave para enlazar un registro desde Microstation	NUMERIC		NOT NULL	

ENTIDAD: CONECTOR SECUNDARIO TABLA: CON_SEC

NOMBRE	DESCRIPCION	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_SECTOR	código del sector	CHARACTER	2	NOT NULL	FK
COD_SEC	código del circuito secundario	CHARACTER	10	NOT NULL	FK
COD_POSTE	código del poste	CHARACTER	10	NOT NULL	FK
MSLINK	clave para enlazar un registro desde Microstation	NUMERIC		NOT NULL	

ENTIDAD: CONEXIÓN ILEGAL TABLA: CNX_ILEG

NOMBRE	DESCRIPCIÓN	TIPO DE DATO	LONGITUD	VALORES PERMITIDOS	CLAVE
COD_ILEG	código de la conexión ilegal	CHARACTER	10	NOT NULL	PK
COD_CLIENT	código del cliente	CHARACTER	10		FK
COD_SEC	código del circuito secundario	CHARACTER	10	NOT NULL	FK

NOMBRE	nombre de quien tiene la conexión ilegal	CHARACTER	30		
BLOQUE	nombre del bloque	CHARACTER	18	NOT NULL	
SOLAR	número del solar	CHARACTER	18	NOT NULL	

NOTA: * : "B" = BUENO , "M" = MALO , "R" = REGULAR

APÉNDICES

APÉNDICE A

GUÍA DE ODBC

Qué es ODBC?

Esencialmente ODBC es un programa de interfaz que permite a aplicaciones clientes de una PC corriendo bajo plataforma Windows 3.1 o Windows NT accesar a datos de diversos tipos de fuentes de datos. Estas fuentes de datos pueden encontrarse en una máquina cliente, o pueden estar localizadas en un servidor remoto que se comunica con la máquina cliente por medio de la red apropiada.

Las fuentes de datos pueden variar en complejidad desde bases de datos corriendo bajo Windows en una máquina cliente; hasta poderosos y sofisticados sistemas de administración de bases de datos relacionales (RDBMS) residentes en un servidor UNIX, comunicándose con las estaciones clientes por medio protocolos como TCP/IP .

Para que una aplicación cliente sea capaz de accesar a datos de una fuente de datos, un driver DLL (Dynamic Link Library) debe existir para cada tipo de fuente de datos a ser accesada.

Teóricamente toda aplicación cliente que haya sido implementada usando el Standard ODBC API puede accesar a cualquier fuente de datos, siempre que el Standard ODBC API provea el driver apropiado que soporte la fuente de datos respectiva.

Esencialmente, solo el driver sabe como "hablar" a su respectiva fuente de datos.

SINGLE-TIER DRIVERS

Los single-tier drivers hacen que el acceso a la base de datos sea más lento que usar las herramientas nativas del DBMS (Sistema de administración de bases de datos) respectivo, debido a que los single-tier drivers tienen que analizar la sintaxis de las sentencias SQL y transformarlas en operaciones básicas de archivos. De la optimización de este proceso depende el tiempo de respuesta de los single-tier drivers.

La tipica situación es donde la fuente de datos reside en el mismo computador donde se encuentran los otros componentes de ODBC.

Con el uso de single-tier drivers no se restringe a que la fuente de datos se encuentre el mismo computador. Estos drivers pueden ser usados en configuraciones FILE/SERVER.

MULTIPLE-TIER DRIVERS

En una configuración **multiple-tier** el driver envía requerimientos a un servidor que procesa estos requerimientos. Estos requerimientos pueden ser sentencias SQL o sentencias en el formato específico de la base de datos respectiva. Aunque la instalación total puede residir en un simple sistema es más a menudo dividido a través de plataformas; comúnmente la aplicación, el driver y el Driver Manager se encuentran en la máquina cliente, pero la base de datos con el software que controla el acceso a dicha base de datos residen en el servidor.

Hay dos tipos de **multiple-tier drivers: two-tier** and **three-tier** (o gateway)

TWO-TIER DRIVERS

Hay dos variaciones de two-tier drivers definidas en términos de funcionalidad de SQL:

1. two-tier drivers para DBMS basados en sentencias SQL, y
2. two-tier drivers para DBMS no basados en sentencias SQL.

Los two-tier drivers para DBMS basados en sentencias SQL como Oracle o Sybase envian directamente las sentencias SQL al DBMS que se encuentra en el servidor. El DBMS maneja todo el procesamiento de las sentencias SQL.

Los two-tier drivers par DBMS no basados en sentencias SQL requieren código extra en el lado del servidor para analizar y transformar las sentencias SQL al formato nativo de la base de datos. Hay dos implementaciones significativas de esto:

1. La sentencia SQL es completamente analizada y transformada en operaciones básicas de entrada y salida a nivel de archivo. Estas operaciones se realizan en el lado del cliente.
2. La sentencia SQL es analizada y transformada al formato nativo de la base de datos en el lado del servidor. La estación cliente también puede hacer un análisis parcial de la sentencia SQL.

THREE-TIER DRIVERS (GATEWAY)

El three-tier driver pasa las sentencias SQL al gateway, el cual las envía a la fuente de datos residentes en un servidor.

Estos drivers pueden soportar sentencias SQL y no SQL.

A menudo el gateway es simplemente una proceso que trabaja a nivel de comunicaciones de red. En este caso, la sentencia SQL es pasada de todas las maneras al servidor.

En otras implementaciones el gateway analiza y transforma las sentencias SQL en sentencias SQL especificas del DBMS; o si el DBMS no soporta SQL, el gateway se encarga de transformarlas en el formato específico correspondiente para es DBMS.

La codificación del gateway en el servidor es también requerida.

APÉNDICE B

CÓDIGO DE LA APLICACIÓN

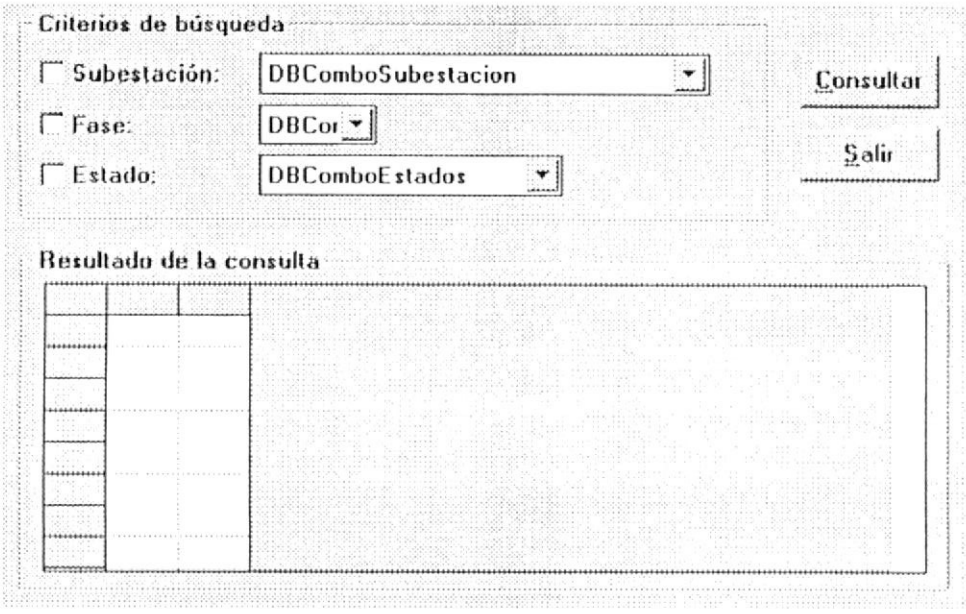
Sistema de Inventario para EMELEC

Aceptar

Versión 1.0

ACERCA.FRM - 1

```
Sub Command3D1_Click ()
    Unload formAcercaDe
End Sub

Sub DB_Command1_Click ()
    Unload Me
End Sub

Sub Form_Load ()
    centrarPantalla Me
End Sub
```

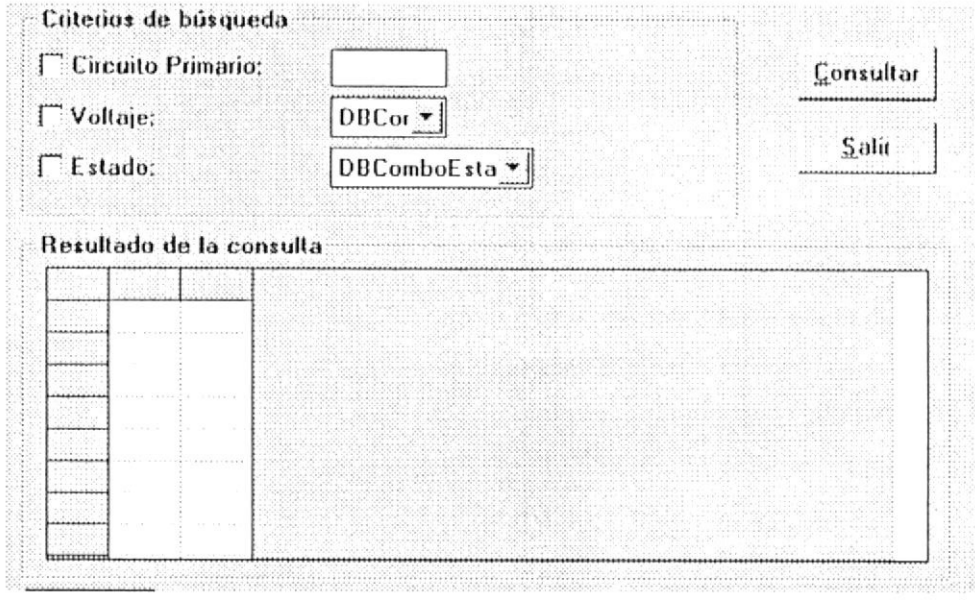

Criterios de búsqueda
Subestación:
DBComboSubestacion
Fase:
DBCor
Estado:
DBComboEstados
Consultar
Salir
Resultado de la consulta

FC_CIRCP.FRM - 1

```
Dim pWhere As String
Dim whereSubestacion As String
Dim whereFase As String
Dim whereEstado As String


Sub Command3D2_Click ()
    Unload formConsultaCircPrim
End Sub

Sub DBComboEstados_Change ()
    whereEstado = " AND DESCRIP = ?DBComboEstados"
    TextWhere.Text = pWhere + whereSubestacion + whereFase
+ whereEstado

End Sub

Sub DBComboEstados_Click ()

    whereEstado = " AND DESCRIP = ?DBComboEstados"
    TextWhere.Text = pWhere + whereSubestacion + whereFase
+ whereEstado

End Sub

Sub DBComboFase_Change ()
    whereFase = " AND FASE = ?DBComboFase"
    TextWhere.Text = pWhere + whereSubestacion + whereFase
+ whereEstado

End Sub

Sub DBComboFase_Click ()
    whereFase = " AND FASE = ?DBComboFase"
    TextWhere.Text = pWhere + whereSubestacion + whereFase
+ whereEstado

End Sub

Sub DBComboSubestacion_Change ()

    whereSubestacion = " AND NOMBRE = ?DBComboSubestacion"
    TextWhere.Text = pWhere + whereSubestacion + whereFase
+ whereEstado
End Sub
```

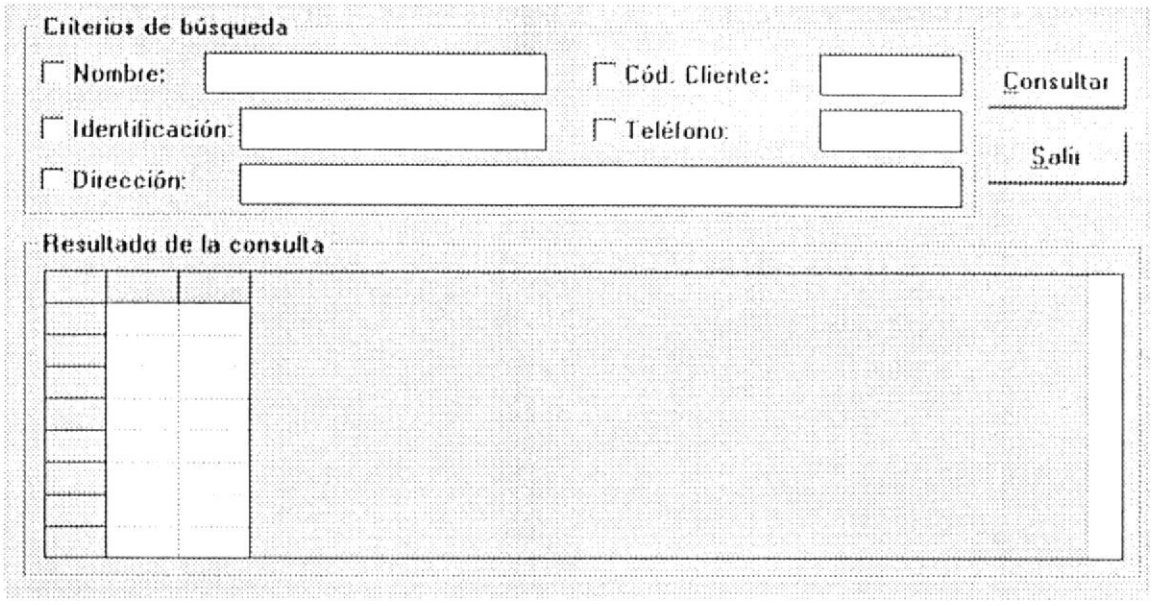
FC_CIRCP.FRM - 2

```
Sub DBComboSubestacion_Click ()

    whereSubestacion = " AND NOMBRE = ?DBComboSubestacion"
    TextWhere.Text = pWhere + whereSubestacion + whereFase
+ whereEstado

End Sub

Sub DBCommandConsultar_Click ()
    sentenciaWhere = pWhere
    If SSCheckSubestacion.Value = True Then sentenciaWhere
= sentenciaWhere + whereSubestacion
    If SSCheckFase.Value = True Then sentenciaWhere = sente
nciaWhere + whereFase
    If SSCheckEstado.Value = True Then sentenciaWhere = sen
tenciaWhere + whereEstado
    TextWhere.Text = sentenciaWhere
    QGridCircuitosPrimarios.pWhere = sentenciaWhere
    res% = fDoQuery(QGridCircuitosPrimarios)
End Sub

Sub DBCommandSalir_Click ()
    Unload Me
End Sub

Sub Form_Load ()
    centrarPantalla Me
    pWhere = QGridCircuitosPrimarios.pWhere
    TextWhere.Text = pWhere + whereSubestacion + whereFase
+ whereEstado
End Sub

Sub Form_Unload (Cancel As Integer)
    pWhere = ""
    whereSubestacion = ""
    whereFase = ""
    whereEstado = ""
    If QGridCircuitosPrimarios.pMode = 1 Then
        res% = fEndQuery(QGridCircuitosPrimarios)
    End If
End Sub

Sub SSCheckEstado_Click (Value As Integer)
    If Value = True Then DBComboEstados.Enabled = True Else
 DBComboEstados.Enabled = False
End Sub
```


Criterios de búsqueda
Circuito Primario:
Voltaje:
DBCor
Estado:
DBComboEsta
Consultar
Salir
Resultado de la consulta

FC_CIRCS.FRM - 1

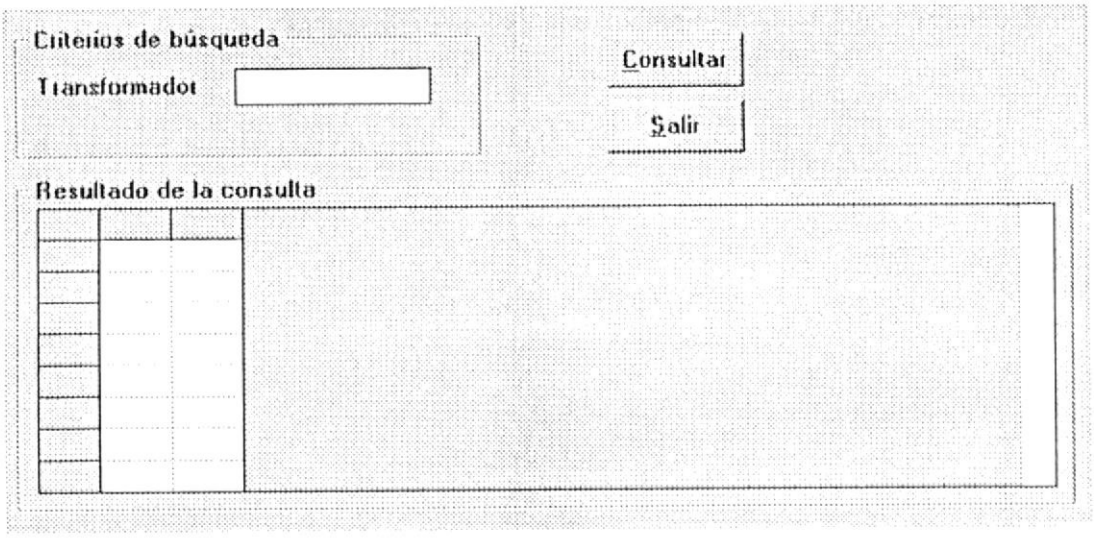
```
Dim pWhere                    As String
Dim whereCircuitoPrimario     As String
Dim whereVoltaje              As String
Dim whereEstado               As String


Sub DBComboEstados_Click ()
    whereEstado = " AND DESCRIP = ?DBComboEstados"
End Sub

Sub DBComboVoltaje_Click ()
    whereVoltaje = " AND VOLT_SEC = ?DBComboVoltaje"
End Sub

Sub DBCommandConsultar_Click ()
    sentenciaWhere = pWhere
    If SSCheckCircuitoPrimario.Value = True Then sentenciaW
here = sentenciaWhere + whereCircuitoPrimario
    If SSCheckVoltaje.Value = True Then sentenciaWhere = se
ntenciaWhere + whereVoltaje
    If SSCheckEstado.Value = True Then sentenciaWhere = sen
tenciaWhere + whereEstado
    QGridCircuitosSecundarios.pWhere = sentenciaWhere
    res% = fDoQuery(QGridCircuitosSecundarios)
End Sub

Sub DBCommandSalir_Click ()
    Unload Me
End Sub

Sub DBTextCircuitoPrimario_Change ()
    whereCircuitoPrimario = " AND COD_PRIM LIKE '%" + DBTex
tCircuitoPrimario.Text + "%'"
End Sub

Sub Form_Load ()
    centrarPantalla Me
    pWhere = QGridCircuitosSecundarios.pWhere
End Sub

Sub Form_Unload (Cancel As Integer)
    pWhere = ""
    whereCircuitoPrimario = ""
    whereVoltaje = ""
    whereEstado = ""
    If QGridCircuitosSecundarios.pMode = 1 Then
```


Criterios de búsqueda
Nombre:
Cód. Cliente:
Consultar
Identificación:
Teléfono:
Salir
Dirección:
Resultado de la consulta

FC_CLIEN.FRM - 1

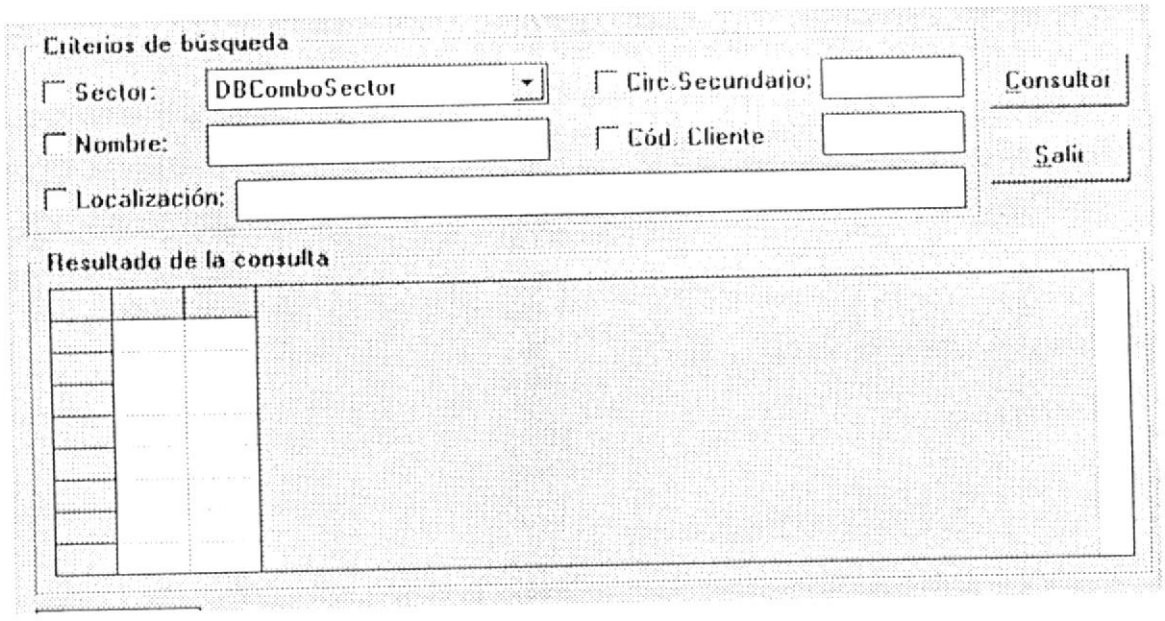
```
Dim pWhere As String
Dim whereIdentificacion As String
Dim whereNombre As String
Dim whereUbicacion As String
Dim whereCodCliente As String
Dim whereTelefono As String


Sub DBCommandConsultar_Click ()
    sentenciaWhere = pWhere
    If SSCheckIdentificacion.Value = True Then sentenciaWhe
re = sentenciaWhere + whereIdentificacion
    If SSCheckUbicacion.Value = True Then sentenciaWhere =
sentenciaWhere + whereUbicacion
    If SSCheckNombre.Value = True Then sentenciaWhere = sen
tenciaWhere + whereNombre
    If SSCheckCodCliente.Value = True Then sentenciaWhere =
 sentenciaWhere + whereCodCliente
    If SSCheckTelefono.Value = True Then sentenciaWhere = s
entenciaWhere + whereTelefono
    QGridClientes.pWhere = sentenciaWhere
    res% = fDoQuery(QGridClientes)

End Sub

Sub DBCommandSalir_Click ()
    Unload Me
End Sub

Sub DBTextCodCliente_Change ()
    whereCodCliente = " AND CLIENTE.COD_CLIENT LIKE '" + Tr
im(DBTextCodCliente.Text) + "%'"
End Sub

Sub DBTextIdentificacion_Change ()
    whereIdentificacion = " AND CLIENTE.IDENT LIKE '" + Tri
m(DBTextIdentificacion.Text) + "%'"
End Sub

Sub DBTextNombre_Change ()
    whereNombre = " AND UPPER(CLIENTE.NOMBRE) LIKE UPPER('"
 + Trim(DBTextNombre.Text) + "%')"
End Sub

Sub DBTextTelefono_Change ()
    whereTelefono = " AND CLIENTE.TELEFONO LIKE '" + Trim(D
```

```
BTextTelefono.Text) + "%'"
End Sub

Sub DBTextUbicacion_Change ()
    whereUbicacion = " AND UPPER(DIRECCION) LIKE UPPER('" +
 DBTextUbicacion.Text + "%')"
End Sub

Sub Form_Load ()
    centrarPantalla Me
    pWhere = QGridClientes.pWhere
End Sub

Sub Form_Unload (Cancel As Integer)
    pWhere = ""
    whereIdentificacion = ""
    whereNombre = ""
    whereUbicacion = ""
    whereCapacidad = ""
    whereTelefono = ""
    whereFase = ""
    whereMarca = ""
    whereAterrizado = ""
    If QGridClientes.pMode = 1 Then
        res% = fEndQuery(QGridClientes)
    End If
End Sub

Sub SSCheckCodCliente_Click (Value As Integer)
    If Value = True Then DBTextCodCliente.Enabled = True El
se DBTextCodCliente.Enabled = False
End Sub

Sub SSCheckIdentificacion_Click (Value As Integer)
    If Value = True Then DBTextIdentificacion.Enabled = Tru
e Else DBTextIdentificacion.Enabled = False
End Sub

Sub SSCheckNombre_Click (Value As Integer)
    If Value = True Then DBTextNombre.Enabled = True Else D
BTextNombre.Enabled = False
End Sub

Sub SSCheckTelefono_Click (Value As Integer)
    If Value = True Then DBTextTelefono.Enabled = True Else
 DBTextTelefono.Enabled = False
```


Criterios de búsqueda
Transformador
Consultar
Salir
Resultado de la consulta

FC_CLXTR.FRM - 1

```
Dim pWhere As String
Dim whereSecundario As String


Sub DBCommandConsultar_Click ()
    sentenciaWhere = pWhere
    sentenciaWhere = sentenciaWhere + whereSecundario
    orderBy = " ORDER BY COD_TRANSF, CLIENTE.COD_CLIENT, CO
D_MEDIDO "
    QGridClientes.pWhere = sentenciaWhere + orderBy
    res% = fDoQuery(QGridClientes)

End Sub

Sub DBCommandSalir_Click ()
    Unload Me
End Sub

Sub DBTextCircSecund_Change ()
    whereSecundario = " AND COD_TRANSF LIKE '" + DBTextCirc
Secund.Text + "%'"
End Sub

Sub Form_Load ()
    centrarPantalla Me
    pWhere = QGridClientes.pWhere
End Sub

Sub Form_Unload (Cancel As Integer)
    pWhere = ""
    whereSecundario = ""
    If QGridClientes.pMode = 1 Then
        res% = fEndQuery(QGridClientes)
    End If
End Sub
```

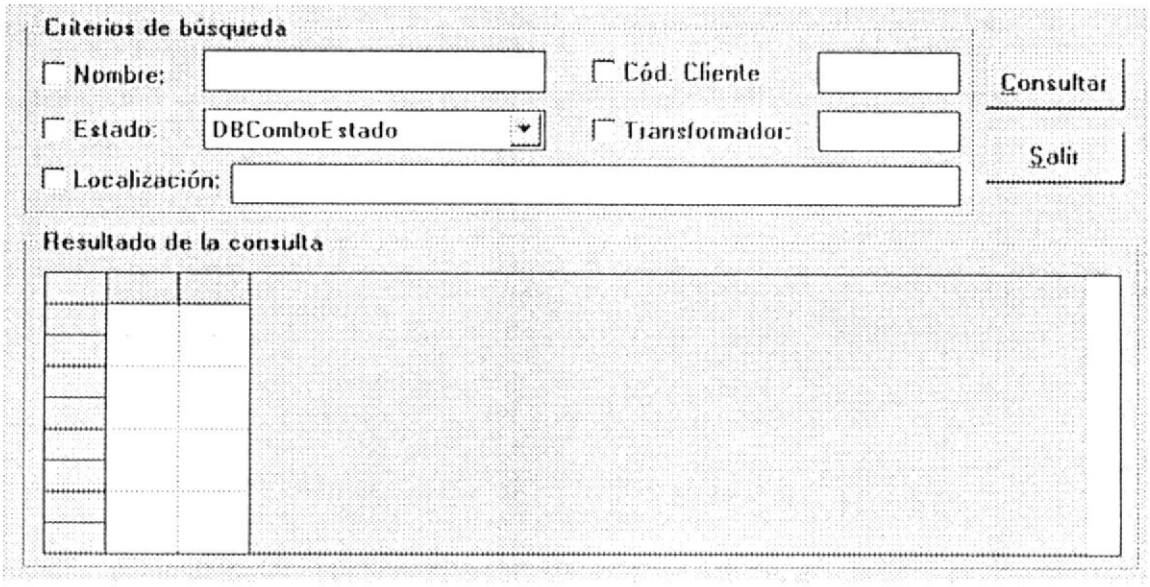

Criterios de búsqueda
Sector:
DBComboSector
Circ.Secundario:
Consultar
Nombre:
Cód. Cliente
Salir
Localización:
Resultado de la consulta

FC_CNILG.FRM - 1

```
Dim pWhere As String
Dim whereSector As String
Dim whereNombre As String
Dim whereUbicacion As String
Dim whereCodCliente As String
Dim whereCircSecundario As String


Sub DBComboSector_Change ()
    whereSector = " AND NOMBRE_SEC = ?DBComboSector "
End Sub

Sub DBComboSector_Click ()
    whereSector = " AND NOMBRE_SEC = ?DBComboSector "
End Sub

Sub DBCommandConsultar_Click ()
    sentenciaWhere = pWhere
    If SSCheckSector.Value = True Then sentenciaWhere = sen
tenciaWhere + whereSector
    If SSCheckUbicacion.Value = True Then sentenciaWhere =
sentenciaWhere + whereUbicacion
    If SSCheckNombre.Value = True Then sentenciaWhere = sen
tenciaWhere + whereNombre
    If SSCheckCodCliente.Value = True Then sentenciaWhere =
 sentenciaWhere + whereCodCliente
    If SSCheckCircSecundario.Value = True Then sentenciaWhe
re = sentenciaWhere + whereCircSecundario
    QGridConexionesIlegales.pWhere = sentenciaWhere
    res% = fDoQuery(QGridConexionesIlegales)

End Sub

Sub DBCommandSalir_Click ()
    Unload Me
End Sub

Sub DBTextCircSecundario_Change ()
    whereCircSecundario = " AND CNX_ILEG.COD_SEC LIKE '" +
Trim(DBTextCircSecundario.Text) + "%'"
End Sub

Sub DBTextCodCliente_Change ()
    whereCodCliente = " AND CNX_ILEG.COD_CLIENT LIKE '" + T
rim(DBTextCodCliente.Text) + "%'"
End Sub
```

FC_CNILG.FRM - 2

```
Sub DBTextNombre_Change ()
    whereNombre = " AND UPPER(CNX_ILEG.NOMBRE) LIKE UPPER('
" + Trim(DBTextNombre.Text) + "%')"
End Sub

Sub DBTextUbicacion_Change ()
    whereUbicacion = " AND UPPER(LOCALIZACI) LIKE UPPER('"
+ DBTextUbicacion.Text + "%')"
End Sub

Sub Form_Load ()
    centrarPantalla Me
    pWhere = QGridConexionesIlegales.pWhere
End Sub

Sub Form_Unload (Cancel As Integer)
    pWhere = ""
    whereSector = ""
    whereNombre = ""
    whereUbicacion = ""
    whereCapacidad = ""
    whereCircSecundario = ""
    whereFase = ""
    whereMarca = ""
    whereAterrizado = ""
    If QGridConexionesIlegales.pMode = 1 Then
        res% = fEndQuery(QGridConexionesIlegales)
    End If
End Sub

Sub SSCheckCircSecundario_Click (Value As Integer)
    If Value = True Then DBTextCircSecundario.Enabled = Tru
e Else DBTextCircSecundario.Enabled = False
End Sub

Sub SSCheckCodCliente_Click (Value As Integer)
    If Value = True Then DBTextCodCliente.Enabled = True El
se DBTextCodCliente.Enabled = False
End Sub

Sub SSCheckNombre_Click (Value As Integer)
    If Value = True Then DBTextNombre.Enabled = True Else D
BTextNombre.Enabled = False
End Sub
```

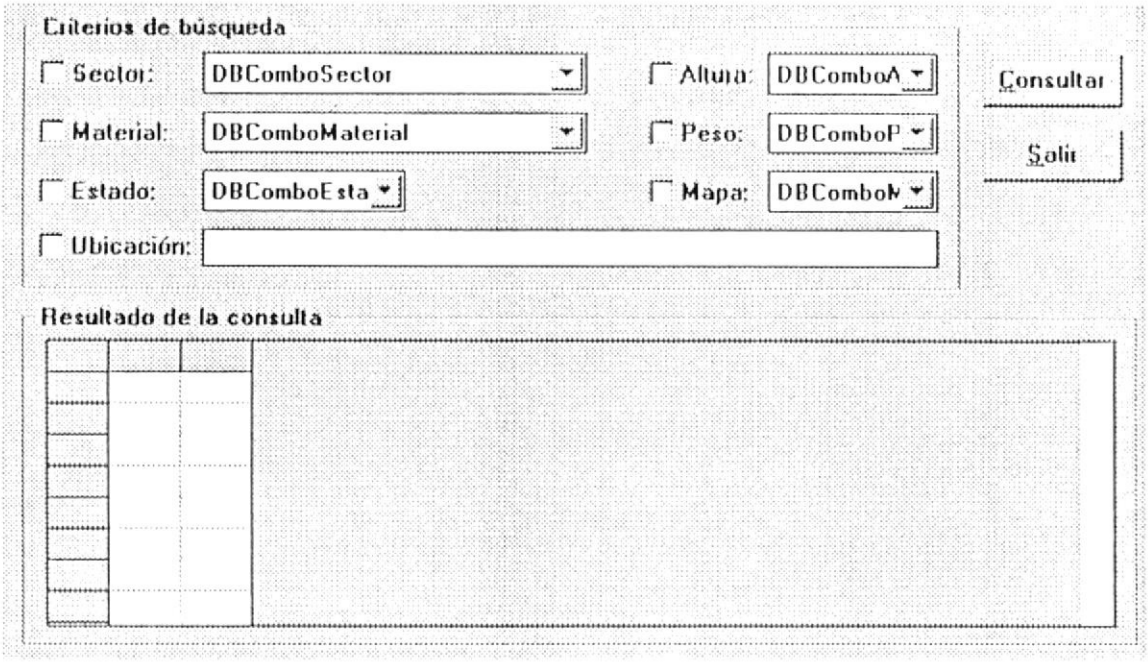

Criterios de búsqueda
Nombre:
Cód. Cliente
Consultar
Estado:
DBComboEstado
Transformador:
Salir
Localización:
Resultado de la consulta

FC_MEDID.FRM - 1

```
Dim pWhere As String
Dim whereEstado As String
Dim whereNombre As String
Dim whereUbicacion As String
Dim whereCodCliente As String
Dim whereTransformador As String


Sub DBComboSector_Change ()
    whereEstado = " AND DESCRIP = ?DBComboEstado "
End Sub

Sub DBComboSector_Click ()
    whereEstado = " AND DESCRIP = ?DBComboEstado "
End Sub

Sub DBTextCircSecundario_Change ()
    whereTransformador = " AND COD_TRANSF LIKE '" + Trim(DB
TextTransformador.Text) + "%'"
End Sub

Sub SSCheckCircSecundario_Click (Value As Integer)
    If Value = True Then DBTextTransformador.Enabled = True
 Else DBTextTransformador.Enabled = False
End Sub

Sub SSCheckSector_Click (Value As Integer)
    If Value = True Then DBComboEstado.Enabled = True Else
DBComboEstado.Enabled = False
End Sub

Sub DBComboEstado_Click ()
    whereEstado = " AND DESCRIP = ?DBComboEstado "
End Sub

Sub DBCommandConsultar_Click ()
    sentenciaWhere = pWhere
    If SSCheckEstado.Value = True Then sentenciaWhere = sen
tenciaWhere + whereEstado
    If SSCheckUbicacion.Value = True Then sentenciaWhere =
sentenciaWhere + whereUbicacion
    If SSCheckNombre.Value = True Then sentenciaWhere = sen
tenciaWhere + whereNombre
    If SSCheckCodCliente.Value = True Then sentenciaWhere =
 sentenciaWhere + whereCodCliente
    If SSCheckTransformador.Value = True Then sentenciaWher
```

FC_MEDID.FRM - 2

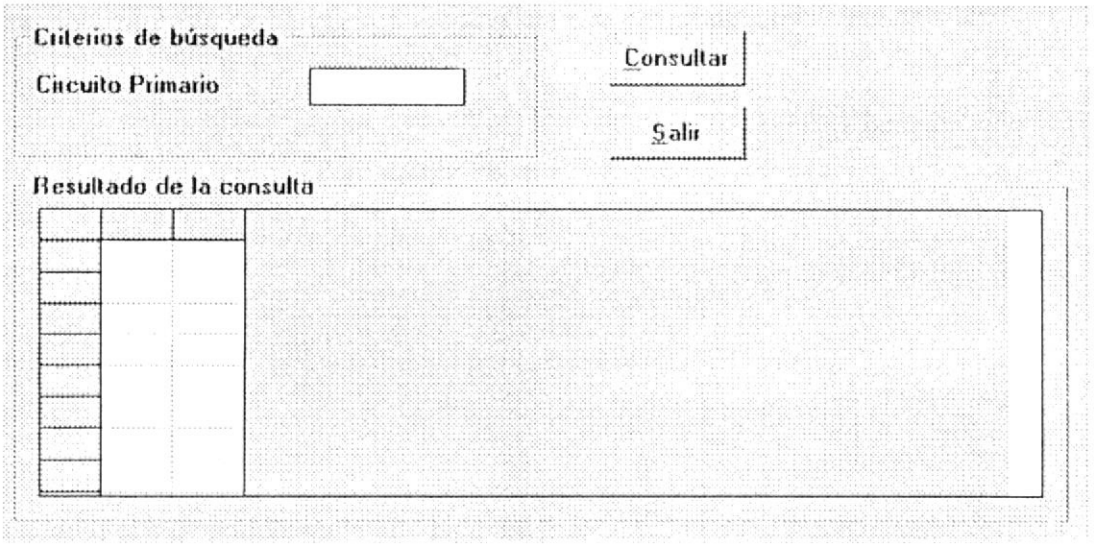
```
e = sentenciaWhere + whereTransformador
    QGridMedidores.pWhere = sentenciaWhere
    res% = fDoQuery(QGridMedidores)

End Sub

Sub DBCommandSalir_Click ()
    Unload Me
End Sub

Sub DBTextCodCliente_Change ()
    whereCodCliente = " AND CLIENTE.COD_CLIENT LIKE '" + Tr
im(DBTextCodCliente.Text) + "%'"
End Sub

Sub DBTextNombre_Change ()
    whereNombre = " AND UPPER(CLIENTE.NOMBRE) LIKE UPPER('"
 + Trim(DBTextNombre.Text) + "%')"
End Sub

Sub DBTextTransformador_Change ()
    whereTransformador = " AND MEDIDOR.COD_TRANSF LIKE '" +
 Trim(DBTextTransformador.Text) + "%'"
End Sub

Sub DBTextUbicacion_Change ()
    whereUbicacion = " AND UPPER(LOCALIZACI) LIKE UPPER('"
+ DBTextUbicacion.Text + "%')"
End Sub

Sub Form_Load ()
    centrarPantalla Me
    pWhere = QGridMedidores.pWhere
End Sub

Sub Form_Unload (Cancel As Integer)
    pWhere = ""
    whereEstado = ""
    whereNombre = ""
    whereUbicacion = ""
    whereCapacidad = ""
    whereTransformador = ""
    whereFase = ""
    whereMarca = ""
    whereAterrizado = ""
    If QGridMedidores.pMode = 1 Then
```


Criterios de búsqueda
Sector:
DBComboSector
Altura:
DBComboA
Material:
DBComboMaterial
Peso:
DBComboP
Estado:
DBComboEsta
Mapa:
DBComboM
Ubicación:
Consultar
Salir
Resultado de la consulta

FC_POSTE.FRM - 1

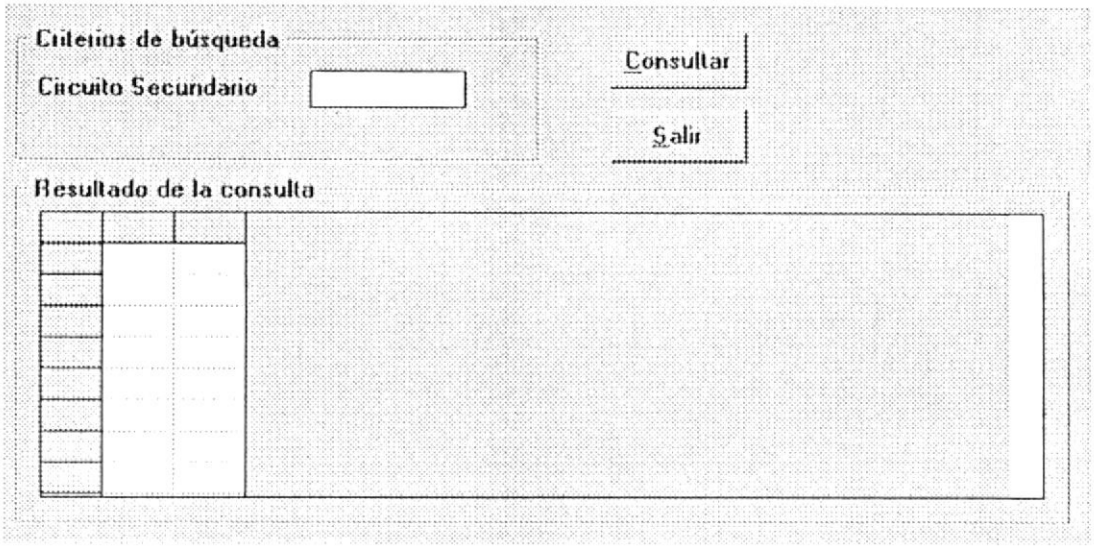
```
Dim pWhere As String
Dim whereSector As String
Dim whereEstado As String
Dim whereUbicacion As String
Dim whereMaterial As String
Dim whereAltura As String
Dim wherePeso As String
Dim whereMapa As String


Sub DBComboAltura_Change ()
    If DBComboAltura.Text = "" Then                'postes s
in altura ingresada
        whereAltura = " AND ALTURA IS NULL"
    Else
        whereAltura = " AND ALTURA = ?DBComboAltura "
    End If
End Sub

Sub DBComboAltura_Click ()
    If DBComboAltura.Text = "" Then    'postes sin altura i
ngresada
        whereAltura = " AND ALTURA IS NULL"
    Else
        whereAltura = " AND ALTURA = ?DBComboAltura "
    End If
End Sub

Sub DBComboEstados_Click ()
    whereEstado = " AND DESCRIP = ?DBComboEstados "
End Sub

Sub DBComboMapa_Change ()
    whereMapa = " AND MAPA = ?DBComboMapa "
End Sub

Sub DBComboMapa_Click ()
    whereMapa = " AND MAPA = ?DBComboMapa "
End Sub

Sub DBComboMaterial_Change ()
    whereMaterial = " AND MATERIAL = ?DBComboMaterial "
End Sub

Sub DBComboMaterial_Click ()
    whereMaterial = " AND MATERIAL = ?DBComboMaterial "
```

```
End Sub

Sub DBComboPeso_Change ()
    If DBComboPeso.Text = "" Then    'postes sin peso ingre
sado
        wherePeso = " AND PESO IS NULL"
    Else
        wherePeso = " AND PESO = ?DBComboPeso"
    End If
End Sub

Sub DBComboPeso_Click ()
    If DBComboPeso.Text = "" Then    'postes sin peso ingre
sado
        wherePeso = " AND PESO IS NULL"
    Else
        wherePeso = " AND PESO = ?DBComboPeso"
    End If
End Sub

Sub DBComboSector_Change ()
    whereSector = " AND NOMBRE_SEC = ?DBComboSector "
End Sub

Sub DBComboSector_Click ()
    whereSector = " AND NOMBRE_SEC = ?DBComboSector "
End Sub

Sub DBCommandConsultar_Click ()
    sentenciaWhere = pWhere
    If SSCheckSector.Value = True Then sentenciaWhere = sen
tenciaWhere + whereSector
    If SSCheckMaterial.Value = True Then sentenciaWhere = s
entenciaWhere + whereMaterial
    If SSCheckEstado.Value = True Then sentenciaWhere = sen
tenciaWhere + whereEstado
    If SSCheckUbicacion.Value = True Then sentenciaWhere =
sentenciaWhere + whereUbicacion
    If SSCheckAltura.Value = True Then sentenciaWhere = sen
tenciaWhere + whereAltura
    If SSCheckPeso.Value = True Then sentenciaWhere = sente
nciaWhere + wherePeso
    If SSCheckMapa.Value = True Then sentenciaWhere = sente
nciaWhere + whereMapa
    QGridPostes.pWhere = sentenciaWhere
    res% = fDoQuery(QGridPostes)
```

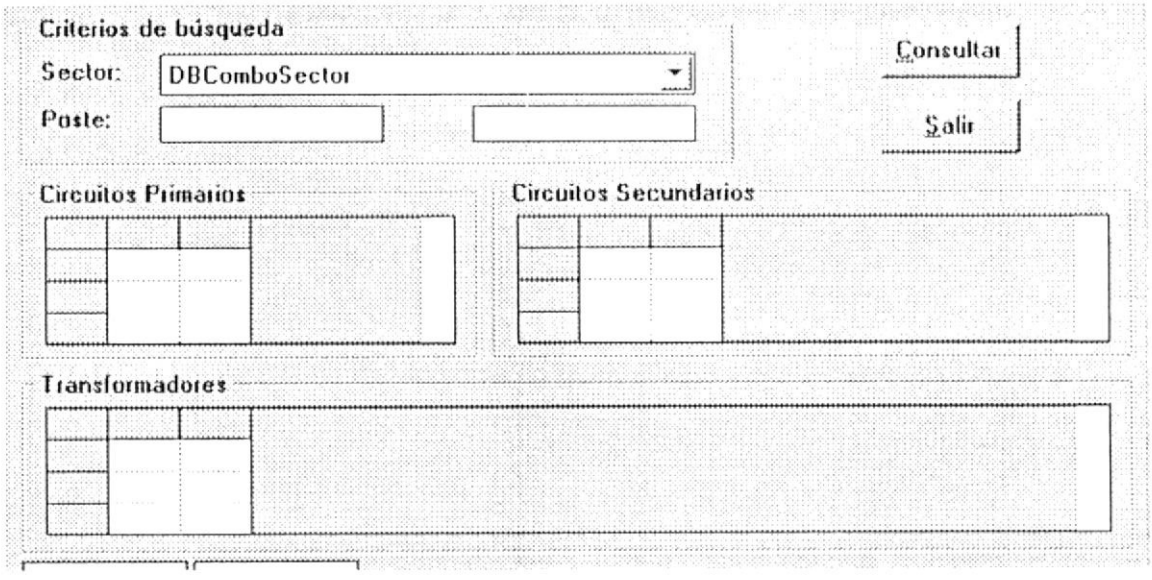

Criterios de búsqueda
Circuito Primario
Consultar
Salir
Resultado de la consulta

FC_PXCP.FRM - 1

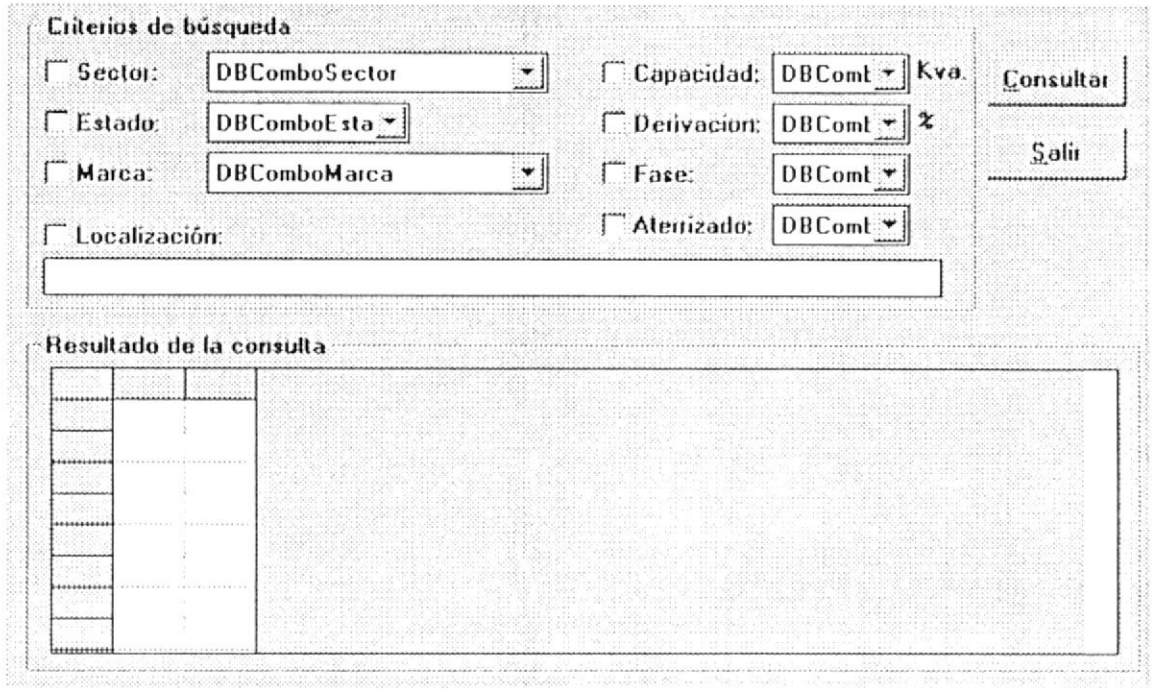
```
Dim pWhere As String
Dim wherePrimario As String


Sub DBCommandConsultar_Click ()
    sentenciaWhere = pWhere
    sentenciaWhere = sentenciaWhere + wherePrimario
    orderBy = " ORDER BY COD_PRIM, POSTE.COD_SECTOR, POSTE.
COD_POSTE "
    QGridPostes.pWhere = sentenciaWhere + orderBy
    res% = fDoQuery(QGridPostes)

End Sub

Sub DBCommandSalir_Click ()
    Unload Me
End Sub

Sub DBTextCircPrimario_Change ()
    wherePrimario = " AND COD_PRIM LIKE '" + DBTextCircPrim
ario.Text + "%'"
End Sub

Sub Form_Load ()
    centrarPantalla Me
    pWhere = QGridPostes.pWhere
End Sub

Sub Form_Unload (Cancel As Integer)
    pWhere = ""
    wherePrimario = ""
    If QGridPostes.pMode = 1 Then
        res% = fEndQuery(QGridPostes)
    End If
End Sub
```


Criterios de búsqueda
Circuito Secundario
Consultar
Salir
Resultado de la consulta

```
FC_PXCS.FRM - 1

Dim pWhere As String
Dim whereSecundario As String


Sub DBCommandConsultar_Click ()
    sentenciaWhere = pWhere
    sentenciaWhere = sentenciaWhere + whereSecundario
    orderBy = " ORDER BY COD_SEC, POSTE.COD_SECTOR, POSTE.C
OD_POSTE "
    QGridPostes.pWhere = sentenciaWhere + orderBy
    res% = fDoQuery(QGridPostes)

End Sub

Sub DBCommandSalir_Click ()
    Unload Me
End Sub

Sub DBTextCircSecund_Change ()
    whereSecundario = " AND COD_SEC LIKE '" + DBTextCircSec
und.Text + "%'"
End Sub

Sub Form_Load ()
    centrarPantalla Me
    pWhere = QGridPostes.pWhere
End Sub

Sub Form_Unload (Cancel As Integer)
    pWhere = ""
    whereSecundario = ""
    If QGridPostes.pMode = 1 Then
        res% = fEndQuery(QGridPostes)
    End If
End Sub
```


Criterios de búsqueda
Sector:
DBComboSector
Poste:
Consultar
Salir
Circuitos Primarios
Circuitos Secundarios
Transformadores

```
FC_TDXPS.FRM - 1

Dim pWhere                      As String
Dim pWhereCP                    As String
Dim pWhereTR                    As String
Dim pWherePoste                 As String
Dim whereSector                 As String
Dim wherePoste                  As String


Sub DBComboSector_Change ()
    whereSector = " AND NOMBRE_SEC = ?DBComboSector "
End Sub

Sub DBComboSector_Click ()
    whereSector = " AND NOMBRE_SEC = ?DBComboSector "
End Sub

Sub DBCommandConsultar_Click ()
    If whereSector = "" Then
        DBComboSector.SetFocus
    ElseIf wherePoste = "" Then
        DBTextPoste.SelStart = 0
        DBTextPoste.SelLength = Len(DBTextPoste.Text)
        DBTextPoste.SetFocus
    Else
        sentenciaWhere = whereSector + wherePoste

        qPoste.pWhere = pWherePoste + sentenciaWhere
        res% = fDoQuery(qPoste)
        res% = fEndQuery(qPoste)

        QGridCircuitosSecundarios.pWhere = pWhere + sentenc
iaWhere
        res% = fDoQuery(QGridCircuitosSecundarios)

        QGridCircuitosPrimarios.pWhere = pWhereCP + sentenc
iaWhere
        res% = fDoQuery(QGridCircuitosPrimarios)

        QGridTransformadores.pWhere = pWhereTR + sentenciaW
here
        res% = fDoQuery(QGridTransformadores)

    End If
End Sub

Sub DBCommandSalir_Click ()
```


Criterios de búsqueda
Sector:
DBComboSector
Estado:
DBComboEsta
Marca:
DBComboMarca
Localización:
Capacidad:
DBComt
Kva.
Derivación:
DBComt
%
Fase:
DBComt
Aterrizado:
DBComt
Consultar
Salir
Resultado de la consulta

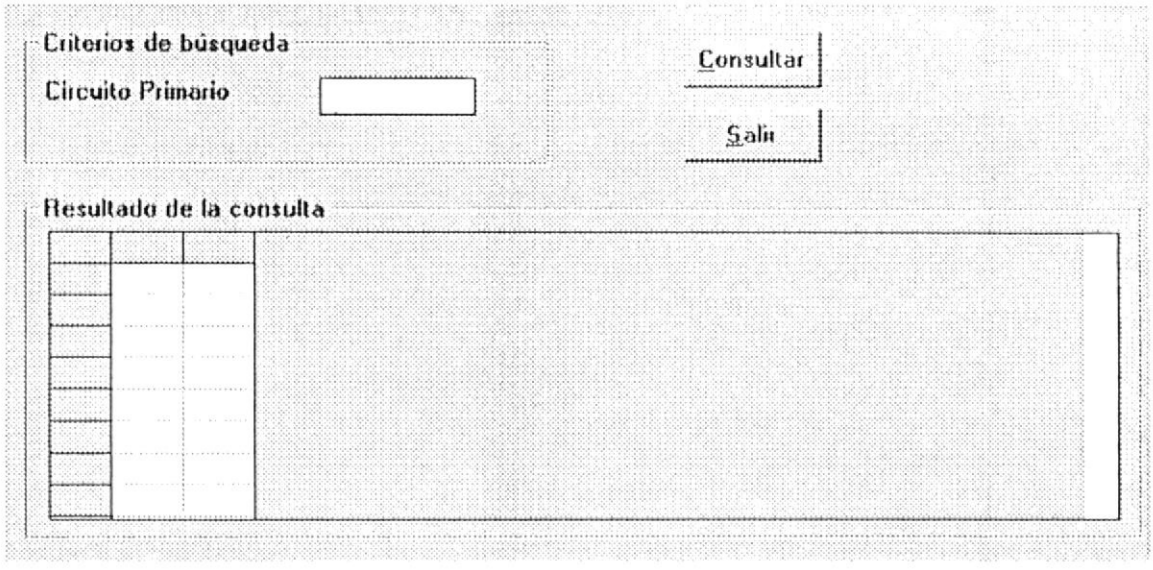
FC_TRANS.FRM - 1

```
Dim pWhere As String
Dim whereSector As String
Dim whereEstado As String
Dim whereUbicacion As String
Dim whereCapacidad As String
Dim whereDerivacion As String
Dim whereFase As String
Dim whereMarca As String
Dim whereAterrizado As String


Sub DBComboAterrizado_Change ()
    whereAterrizado = " AND ATERRIZADO = ?DBComboAterrizado
"
End Sub

Sub DBComboAterrizado_Click ()
    whereAterrizado = " AND ATERRIZADO = ?DBComboAterrizado
"
End Sub

Sub DBComboCapacidad_Change ()
    If DBComboCapacidad.Text = "" Then    'transformadores
sin Capacidad ingresado
        whereCapacidad = " AND CAPACIDAD IS NULL"
    Else
        whereCapacidad = " AND CAPACIDAD = ?DBComboCapacida
d"
    End If

End Sub

Sub DBComboCapacidad_Click ()
    If DBComboCapacidad.Text = "" Then    'transformadores
sin Capacidad ingresado
        whereCapacidad = " AND CAPACIDAD IS NULL"
    Else
        whereCapacidad = " AND CAPACIDAD = ?DBComboCapacida
d"
    End If

End Sub

Sub DBComboDerivacion_Change ()
    If DBComboDerivacion.Text = "" Then    'transformadores
 sin Derivacion ingresada
```

```
        whereDerivacion = " AND Derivacion IS NULL"
    Else
        whereDerivacion = " AND Derivacion = ?DBComboDeriva
cion "
    End If

End Sub

Sub DBComboDerivacion_Click ()
    If DBComboDerivacion.Text = "" Then    'transformadores
 sin Derivacion ingresada
        whereDerivacion = " AND Derivacion IS NULL"
    Else
        whereDerivacion = " AND Derivacion = ?DBComboDeriva
cion "
    End If

End Sub

Sub DBComboEstados_Click ()
    whereEstado = " AND DESCRIP = ?DBComboEstados "
End Sub

Sub DBComboFase_Change ()
    whereFase = " AND FASE_ALIME = ?DBComboFase"
End Sub

Sub DBComboFase_Click ()
        whereFase = " AND FASE_ALIME = ?DBComboFase"
End Sub

Sub DBComboMarca_Change ()
    whereMarca = " AND Marca = ?DBComboMarca "
End Sub

Sub DBComboMarca_Click ()
    whereMarca = " AND Marca = ?DBComboMarca "
End Sub

Sub DBComboSector_Change ()
    whereSector = " AND NOMBRE_SEC = ?DBComboSector "
End Sub

Sub DBComboSector_Click ()
    whereSector = " AND NOMBRE_SEC = ?DBComboSector "
End Sub
```

FC_TRANS.FRM - 3

```
Sub DBCommandConsultar_Click ()
    sentenciaWhere = pWhere
    If SSCheckSector.Value = True Then sentenciaWhere = sen
tenciaWhere + whereSector
    If SSCheckCapacidad.Value = True Then sentenciaWhere =
sentenciaWhere + whereCapacidad
    If SSCheckEstado.Value = True Then sentenciaWhere = sen
tenciaWhere + whereEstado
    If SSCheckUbicacion.Value = True Then sentenciaWhere =
sentenciaWhere + whereUbicacion
    If SSCheckDerivacion.Value = True Then sentenciaWhere =
 sentenciaWhere + whereDerivacion
    If SSCheckFase.Value = True Then sentenciaWhere = sente
nciaWhere + whereFase
    If SSCheckMarca.Value = True Then sentenciaWhere = sent
enciaWhere + whereMarca
    If SSCheckAterrizado.Value = True Then sentenciaWhere =
 sentenciaWhere + whereAterrizado
    QGridTransformadores.pWhere = sentenciaWhere
    res% = fDoQuery(QGridTransformadores)

End Sub

Sub DBCommandSalir_Click ()
    Unload Me
End Sub

Sub DBTextUbicacion_Change ()
    whereUbicacion = " AND UPPER(LOCALIZACI) LIKE UPPER('"
+ DBTextUbicacion.Text + "%')"
End Sub

Sub Form_Load ()
    centrarPantalla Me
    pWhere = QGridTransformadores.pWhere
End Sub

Sub Form_Unload (Cancel As Integer)
    pWhere = ""
    whereSector = ""
    whereEstado = ""
    whereUbicacion = ""
    whereCapacidad = ""
    whereDerivacion = ""
    whereFase = ""
```


Criterios de búsqueda
Circuito Primario
Consultar
Salir
Resultado de la consulta

FC_TRXCP.FRM - 1

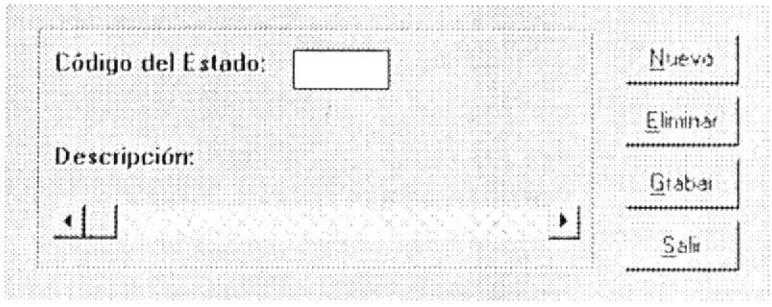
```
Dim pWhere As String
Dim wherePrimario As String


Sub DBCommandConsultar_Click ()
    sentenciaWhere = pWhere
    sentenciaWhere = sentenciaWhere + wherePrimario
    orderBy = " ORDER BY COD_PRIM, COD_TRANSF "
    QGridTransformadores.pWhere = sentenciaWhere
    res% = fDoQuery(QGridTransformadores)

End Sub

Sub DBCommandSalir_Click ()
    Unload Me
End Sub

Sub DBTextCircPrimario_Change ()
    wherePrimario = " AND COD_PRIM LIKE '" + DBTextCircPrim
ario.Text + "%'"
End Sub

Sub Form_Load ()
    centrarPantalla Me
    pWhere = QGridTransformadores.pWhere
End Sub

Sub Form_Unload (Cancel As Integer)
    pWhere = ""
    wherePrimario = ""
    If QGridTransformadores.pMode = 1 Then
        res% = fEndQuery(QGridTransformadores)
    End If
End Sub
```

```
FM_CIRCP.FRM - 1

Dim BUSCAR_PRIMERO As Integer


Sub Command3D3_Click ()
    Unload formMCircPrim
End Sub

Sub Text3_GotFocus ()

End Sub

Sub bBuscar_Click ()
    r% = fFind(qCircPrim, 1)'buscar primer registro coincid
ente
    'regresar campos a sólo lectura
    cod_prim.ReadOnly = True
    BUSCAR_PRIMERO = True
End Sub

Sub bBuscarSig_Click ()
    'regresar campos a sólo lectura
    cod_prim.ReadOnly = True
End Sub

Sub bFinBuscar_Click ()
    r% = fClearFind(qCircPrim)
    'habilitar y deshabilitar botones
    bGrabar.Enabled = True
    bModoBuscar.Enabled = True
    bBuscar.Enabled = False
    bBuscarSig.Enabled = False
    bFinBuscar.Enabled = False
    If BUSCAR_PRIMERO = False Then
        r% = fDoQuery(qCircPrim)
    End If
    BUSCAR_PRIMERO = False
    cod_prim.ReadOnly = True

End Sub

Sub bGrabar_Click ()
    If Trim$(fase.Text) <> "" And Trim$(conductor.Text) <>
"" And Trim$(calibre.Text) <> "" And IsFase(UCase$(fase.Tex
t)) Then
        res% = fUpdate(qCircPrim, 0)
    Else
```

FM_CIRCP.FRM - 2

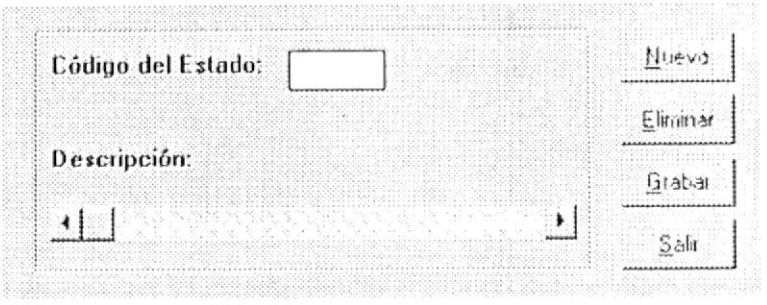
```
        If IsFase(UCase$(fase.Text)) = 0 Then
            MsgBox "Valor no válido para la fase del Circui
to Primario", 48, TITULO
        Else
            MsgBox "Faltan datos por ingresar.", 64, TITULO
        End If
    End If

    res% = fDoQuery(qCircPrimF)
    res% = fDoQuery(qCircPrimMatCon)
    res% = fDoQuery(qCircPrimCalCon)
End Sub

Sub bModoBuscar_Click ()
    bGrabar.Enabled = False 'deshabilitar botón grabar
    'permitir escribir en los campos
    cod_prim.ReadOnly = False
    'habilitar y deshabilitar botones
    bModoBuscar.Enabled = False
    bBuscar.Enabled = True
    bBuscarSig.Enabled = True
    bFinBuscar.Enabled = True
End Sub

Sub bSalir_Click ()
    Unload Me
End Sub

Sub Form_Load ()

    centrarPantalla Me

    qCircPrim.pTables = pathCircPrim
    qEstado.pTables = pathEstCircPrim
    qSubest.pTables = pathSubestacion
    qCircPrimF.pTables = pathCircPrim
    qCircPrimMatCon.pTables = pathCircPrim
    qCircPrimCalCon.pTables = pathCircPrim

    r% = fDoQuery(qEstado)
    r% = fDoQuery(qSubest)
    r% = fDoQuery(qCircPrimF)
    r% = fDoQuery(qCircPrimMatCon)
    r% = fDoQuery(qCircPrimCalCon)
    r% = fDoQuery(qCircPrim)
```

```
    r% = fEndQuery(qEstado)
    r% = fEndQuery(qSubest)
    r% = fEndQuery(qCircPrimF)
    r% = fEndQuery(qCircPrimMatCon)
    r% = fEndQuery(qCircPrimCalCon)

End Sub
```


Código del Estado:
Descripción:
Nuevo
Eliminar
Grabar
Salir

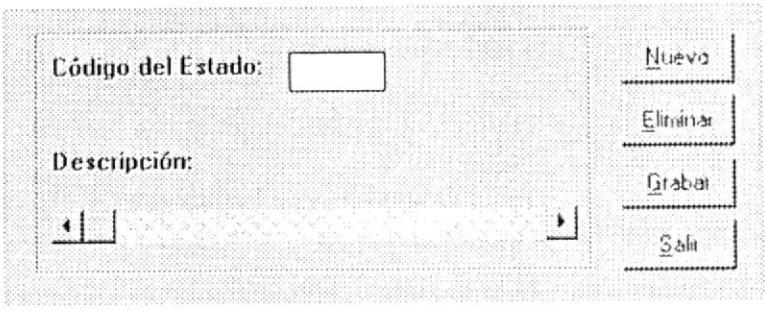
```
FM_ESTCP.FRM - 1

Dim ESTADO_NUEVO As Integer


Sub bEliminar_Click ()

    respuesta = MsgBox(MSGELREG, DEFELREG, TITULO)

    If respuesta = IDSI Then
        qTablaRel.pWhere = " cod_estado = '" + cod_estado.T
ext + "'"
        res% = fDoQuery(qTablaRel)

        If qTablaRel.pRecCount > 0 Then
            MsgBox "No puede eliminarse el registro. Exiten
 circuitos primarios con ese estado", 48, TITULO
            Me.cod_estado.SetFocus
        Else
            res% = fDelete(qEst, 0)
        End If

        res% = fEndQuery(qTablaRel)
        Me.cod_estado.ReadOnly = True
        Me.bNuevo.Enabled = True

    End If

End Sub

Sub bGrabar_Click ()

If ESTADO_NUEVO Then 'para verificar si voy a insertar un r
egistro

    qCodEst.pWhere = " COD_ESTADO = '" + cod_estado.Text +
"'"
    res% = fDoQuery(qCodEst)

    If qCodEst.pRecCount > 0 Then
        MsgBox "Ya existe un registro con ese código ", 48,
 TITULO
        Me.cod_estado.SetFocus
    Else
        res% = fUpdate(qEst, 0)
        Me.cod_estado.ReadOnly = True
        Me.bNuevo.Enabled = True
        'para ordenar el registro recién ingresado
```

```
        ESTADO_NUEVO = False
        res% = fEndQuery(qEst)
        res% = fDoQuery(qEst)
    End If

    res% = fEndQuery(qCodEst)

Else
    res% = fUpdate(qEst, 0)

End If

End Sub

Sub bNuevo_Click ()
    Me.cod_estado.ReadOnly = False
    Me.cod_estado.SetFocus
    Me.bNuevo.Enabled = False
    ESTADO_NUEVO = True

End Sub

Sub bSalir_Click ()
    Unload Me
End Sub

Sub DB_HScroll1_GotFocus ()
    Me.cod_estado.ReadOnly = True
    Me.bNuevo.Enabled = True


End Sub

Sub Form_Load ()
    centrarPantalla Me

    qEst.pTables = pathEstCircPrim
    qCodEst.pTables = pathEstCircPrim
    qTablaRel.pTables = pathCircPrim

    res% = fDoQuery(qEst)

End Sub
```


Código del Estado:
Descripción:
Nuevo
Eliminar
Grabar
Salir

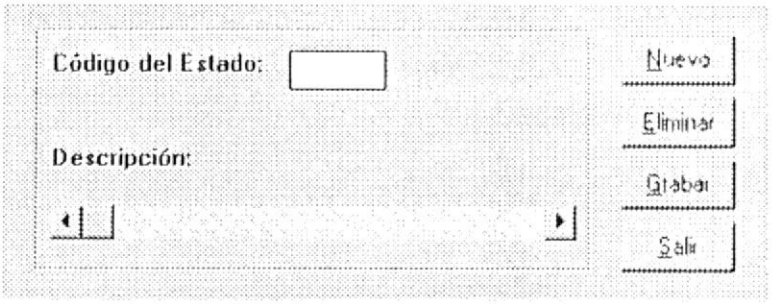
```
FM_ESTCS.FRM - 1

Dim ESTADO_NUEVO As Integer


Sub bEliminar_Click ()

    respuesta = MsgBox(MSGELREG, DEFELREG, TITULO)

    If respuesta = IDSI Then
        qTablaRel.pWhere = " cod_estado = '" + cod_estado.T
ext + "'"
        res% = fDoQuery(qTablaRel)

        If qTablaRel.pRecCount > 0 Then
            MsgBox "No puede eliminarse el registro. Exiten
 circuitos secundarios con ese estado", 48, TITULO
            Me.cod_estado.SetFocus
        Else
            res% = fDelete(qEst, 0)
        End If

        res% = fEndQuery(qTablaRel)
        Me.cod_estado.ReadOnly = True
        Me.bNuevo.Enabled = True

    End If

End Sub

Sub bGrabar_Click ()

If ESTADO_NUEVO Then 'para verificar si voy a insertar un r
egistro

    qCodEst.pWhere = " COD_ESTADO = '" + cod_estado.Text +
"'"
    res% = fDoQuery(qCodEst)

    If qCodEst.pRecCount > 0 Then
        MsgBox "Ya existe un registro con ese código ", 48,
 TITULO
        Me.cod_estado.SetFocus
    Else
        res% = fUpdate(qEst, 0)
        Me.cod_estado.ReadOnly = True
        Me.bNuevo.Enabled = True
        'para ordenar el registro recién ingresado
```

```
        ESTADO_NUEVO = False
        res% = fEndQuery(qEst)
        res% = fDoQuery(qEst)
    End If

    res% = fEndQuery(qCodEst)

Else
    res% = fUpdate(qEst, 0)

End If


End Sub

Sub bNuevo_Click ()
    Me.cod_estado.ReadOnly = False
    Me.cod_estado.SetFocus
    Me.bNuevo.Enabled = False
    ESTADO_NUEVO = True

End Sub

Sub bSalir_Click ()
     Unload Me
End Sub

Sub DB_HScroll1_GotFocus ()
    Me.cod_estado.ReadOnly = True
    Me.bNuevo.Enabled = True

End Sub

Sub Form_Load ()
    centrarPantalla Me

    qEst.pTables = pathEstCircSec
    qCodEst.pTables = pathEstCircSec
    qTablaRel.pTables = pathCircSec

    res% = fDoQuery(qEst)

End Sub
```


Código del Estado:
Descripción:
Nuevo
Eliminar
Grabar
Salir

```
FM_ESTM.FRM - 1

Dim ESTADO_NUEVO As Integer


Sub bEliminar_Click ()

    respuesta = MsgBox(MSGELREG, DEFELREG, TITULO)

    If respuesta = IDSI Then
        qTablaRel.pWhere = " cod_estado = '" + cod_estado.T
ext + "'"
        res% = fDoQuery(qTablaRel)

        If qTablaRel.pRecCount > 0 Then
            MsgBox "No puede eliminarse el registro. Exiten
 circuitos medidores con ese estado", 48, TITULO
            Me.cod_estado.SetFocus
        Else
            res% = fDelete(qEst, 0)
        End If

        res% = fEndQuery(qTablaRel)
        Me.cod_estado.ReadOnly = True
        Me.bNuevo.Enabled = True

    End If

End Sub

Sub bGrabar_Click ()

If ESTADO_NUEVO Then 'para verificar si voy a insertar un r
egistro

    qCodEst.pWhere = " COD_ESTADO = '" + cod_estado.Text +
"'"
    res% = fDoQuery(qCodEst)

    If qCodEst.pRecCount > 0 Then
        MsgBox "Ya existe un registro con ese código ", 48,
 TITULO
        Me.cod_estado.SetFocus
    Else
        res% = fUpdate(qEst, 0)
        Me.cod_estado.ReadOnly = True
        Me.bNuevo.Enabled = True
        'para ordenar el registro recién ingresado
```

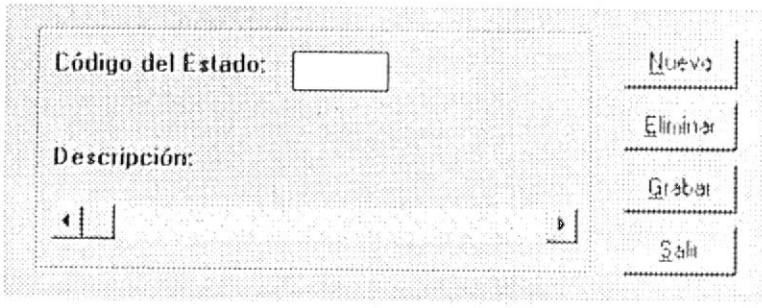
```
        ESTADO_NUEVO = False
        res% = fEndQuery(qEst)
        res% = fDoQuery(qEst)
    End If

    res% = fEndQuery(qCodEst)

Else
    res% = fUpdate(qEst, 0)

End If


End Sub

Sub bNuevo_Click ()
    Me.cod_estado.ReadOnly = False
    Me.cod_estado.SetFocus
    Me.bNuevo.Enabled = False
    ESTADO_NUEVO = True
End Sub

Sub bSalir_Click ()
    Unload Me
End Sub

Sub DB_HScroll1_GotFocus ()
    Me.cod_estado.ReadOnly = True
    Me.bNuevo.Enabled = True

End Sub

Sub Form_Load ()
    centrarPantalla Me

    qEst.pTables = pathEstMedidor
    qCodEst.pTables = pathEstMedidor
    qTablaRel.pTables = pathMedidor

    res% = fDoQuery(qEst)

End Sub
```


Código del Estado:
Descripción:
Nuevo
Eliminar
Grabar
Salir

```
FM_ESTP.FRM - 1

Dim ESTADO_NUEVO As Integer


Sub bBuscar_Click ()
    Me.cod_estado.ReadOnly = True
    Me.bNuevo.Enabled = True
End Sub

Sub bEliminar_Click ()

    respuesta = MsgBox(MSGELREG, DEFELREG, TITULO)

    If respuesta = IDSI Then
        qPoste.pWhere = " cod_estado = '" + cod_estado.Text
 + "'"
        res% = fDoQuery(qPoste)

        If qPoste.pRecCount > 0 Then
            MsgBox "No puede eliminarse el registro. Exiten
 postes con ese estado", 48, TITULO
            Me.cod_estado.SetFocus
        Else
            res% = fDelete(qEstPoste, 0)
        End If

        res% = fEndQuery(qPoste)
        Me.cod_estado.ReadOnly = True
        Me.bNuevo.Enabled = True

    End If
End Sub

Sub bGrabar_Click ()

If ESTADO_NUEVO Then 'para verificar si voy a insertar un r
egistro

    qCodEstPoste.pWhere = " COD_ESTADO = '" + cod_estado.Te
xt + "'"
    res% = fDoQuery(qCodEstPoste)

    If qCodEstPoste.pRecCount > 0 Then
        MsgBox "Ya existe un registro con ese código", 48,
TITULO
        Me.cod_estado.SetFocus
    Else
```

FM_ESTP.FRM - 2

```
        res% = fUpdate(qEstPoste, 0)
        Me.cod_estado.ReadOnly = True
        Me.bNuevo.Enabled = True
        'para ordenar el registro recién ingresado
        ESTADO_NUEVO = False
        res% = fEndQuery(qEstPoste)
        res% = fDoQuery(qEstPoste)
    End If

    res% = fEndQuery(qCodEstPoste)

Else
    res% = fUpdate(qEstPoste, 0)

End If

End Sub

Sub bNuevo_Click ()
    Me.cod_estado.ReadOnly = False
    Me.cod_estado.SetFocus
    Me.bNuevo.Enabled = False
    ESTADO_NUEVO = True

End Sub

Sub bSalir_Click ()
    Unload Me
End Sub

Sub DB_HScroll1_GotFocus ()
    Me.cod_estado.ReadOnly = True
    Me.bNuevo.Enabled = True

End Sub

Sub Form_Load ()
    centrarPantalla Me

    qEstPoste.pTables = pathEstPoste
    qCodEstPoste.pTables = pathEstPoste
    qPoste.pTables = pathPoste

    res% = fDoQuery(qEstPoste)

End Sub
```


Código del Estado:
Descripción:
Nuevo
Eliminar
Grabar
Salir

```
FM_ESTT.FRM - 1

Dim ESTADO_NUEVO As Integer


Sub bEliminar_Click ()

    respuesta = MsgBox(MSGELREG, DEFELREG, TITULO)

    If respuesta = IDSI Then
        qTablaRel.pWhere = " cod_estado = '" + cod_estado.T
ext + "'"
        res% = fDoQuery(qTablaRel)

        If qTablaRel.pRecCount > 0 Then
            MsgBox "No puede eliminarse el registro. Exiten
 postes con ese estado", 48, TITULO
            Me.cod_estado.SetFocus
        Else
            res% = fDelete(qEst, 0)
        End If

        res% = fEndQuery(qTablaRel)
        Me.cod_estado.ReadOnly = True
        Me.bNuevo.Enabled = True

    End If

End Sub

Sub bGrabar_Click ()

If ESTADO_NUEVO Then 'para verificar si voy a insertar un r
egistro

    qCodEst.pWhere = " COD_ESTADO = '" + cod_estado.Text +
"'"
    res% = fDoQuery(qCodEst)

    If qCodEst.pRecCount > 0 Then
        MsgBox "Ya existe un registro con ese código ", 48,
 TITULO
        Me.cod_estado.SetFocus
    Else
        res% = fUpdate(qEst, 0)
        Me.cod_estado.ReadOnly = True
        Me.bNuevo.Enabled = True
        'para ordenar el registro recién ingresado
```

FM_ESTT.FRM - 2

```
        ESTADO_NUEVO = False
        res% = fEndQuery(qEst)
        res% = fDoQuery(qEst)
    End If

    res% = fEndQuery(qCodEst)

Else
    res% = fUpdate(qEst, 0)

End If


End Sub

Sub bNuevo_Click ()
    Me.cod_estado.ReadOnly = False
    Me.cod_estado.SetFocus
    Me.bNuevo.Enabled = False
    ESTADO_NUEVO = True

End Sub

Sub bSalir_Click ()
    Unload Me
End Sub

Sub DB_HScroll1_GotFocus ()
    Me.cod_estado.ReadOnly = True
    Me.bNuevo.Enabled = True


End Sub

Sub Form_Load ()

    centrarPantalla Me

    qEst.pTables = pathEstTransformador
    qCodEst.pTables = pathEstTransformador
    qTablaRel.pTables = pathTransformador

    res% = fDoQuery(qEst)

End Sub
```

```
FM_MEDID.FRM - 1

Dim MEDIDOR_NUEVO As Integer, MODO_BUSQUEDA As Integer, BUS
CAR_PRIMERO As Integer


Sub bBuscar_Click ()
    r% = fFind(qMedidor, 1)
    'buscar primer registro coincidente
    'regresar campos a sólo lectura
    cod_medido.ReadOnly = True
    'cod_transf.ReadOnly = True
    BUSCAR_PRIMERO = True
End Sub

Sub bBuscarSig_Click ()
    'regresar campos a sólo lectura
    cod_medido.ReadOnly = True
    'cod_transf.ReadOnly = True
End Sub

Sub bEliminar_Click ()

    respuesta = MsgBox(MSGELREG, DEFELREG, TITULO)

    If respuesta = IDSI Then
        res% = fDelete(qMedidor, 0)
        bNuevo.Enabled = True
        cod_medido.ReadOnly = True
    End If

End Sub

Sub bFinBuscar_Click ()
    r% = fClearFind(qMedidor)
    'habilitar y deshabilitar botones
    bGrabar.Enabled = True
    bNuevo.Enabled = True
    bEliminar.Enabled = True
    bModoBuscar.Enabled = True
    bBuscar.Enabled = False
    bBuscarSig.Enabled = False
    bFinBuscar.Enabled = False
    MODO_BUSCAR = False

    If BUSCAR_PRIMERO = False Then
        r% = fDoQuery(qMedidor)
    End If
```

FM_MEDID.FRM - 2

```
    BUSCAR_PRIMERO = False
    cod_medido.ReadOnly = True

End Sub

Sub bGrabar_Click ()
    qCodTransf.pWhere = " COD_TRANSF = '" + cod_transf.Text
 + "'"
    res% = fDoQuery(qCodTransf)
    qCodCliente.pWhere = " COD_CLIENT = '" + cod_client.Tex
t + "'"
    res% = fDoQuery(qCodCliente)
    qCodMedidor.pWhere = " COD_MEDIDO = '" + cod_medido.Tex
t + "'"
    res% = fDoQuery(qCodMedidor)

    If MEDIDOR_NUEVO Then
        If qCodMedidor.pRecCount > 0 Then
            MsgBox "Ya existe un medidor con el código ingr
esado", 48, TITULO
            cod_medido.SetFocus
            GoTo fin
        End If
    End If

    If qCodTransf.pRecCount = 0 Then
        MsgBox "No existe ningún transformador con el códig
o ingresado"
        cod_transf.SetFocus
    ElseIf qCodCliente.pRecCount = 0 Then
        MsgBox "No existe ningún cliente con el código ingr
esado"
        cod_client.SetFocus
    Else
        If Trim$(bloque.Text) <> "" And Trim$(carga.Text) <
> "" And IsNumeric(carga.Text) = True And Trim$(solar.Text)
 <> "" And Trim$(cod_estado.Text) <> "" And Trim$(clase.Tex
t) <> "" And Trim$(tipo.Text) <> "" Then
            cod_medido.ReadOnly = True
            bNuevo.Enabled = True
            res% = fUpdate(qMedidor, 0)
            res% = fDoQuery(qMedidC)
            res% = fDoQuery(qMedidT)
            If MEDIDOR_NUEVO = True Then
                res% = fDoQuery(qMedidor)
            End If
```

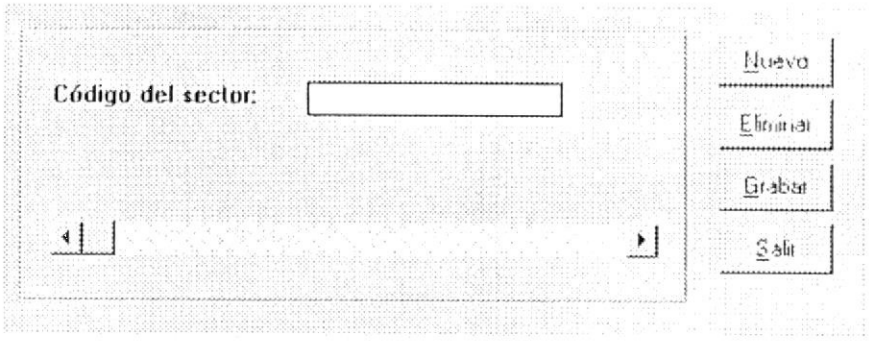
```
            MEDIDOR_NUEVO = False
        Else
            If IsNumeric(carga.Text) = False Then
                carga.SetFocus
                MsgBox "Valor no válido para la carga", 64,
 "Sistema de Inventario para EMELEC"
            Else
                MsgBox "Faltan datos por ingresar", 64, "Si
stema de Inventario para EMELEC"
            End If
        End If
    End If


fin:
    res% = fEndQuery(qCodTransf)
    res% = fEndQuery(qCodCliente)
    res% = fEndQuery(qCodMedidor)
End Sub

Sub bModoBuscar_Click ()
    'deshabilitar botones grabar, nuevo y eliminar
    bGrabar.Enabled = False
    bNuevo.Enabled = False
    bEliminar.Enabled = False
    'permitir escribir en los campos
    cod_medido.ReadOnly = False
    'cod_transf.ReadOnly = False
    'habilitar y deshabilitar botones de búsqueda
    bModoBuscar.Enabled = False
    bBuscar.Enabled = True
    bBuscarSig.Enabled = True
    bFinBuscar.Enabled = True
    MODO_BUSCAR = True
End Sub

Sub bNuevo_Click ()
    MEDIDOR_NUEVO = True
    cod_medido.ReadOnly = False
    cod_medido.SetFocus
    bNuevo.Enabled = False
End Sub

Sub bSalir_Click ()
    Unload Me
End Sub
```

```
Sub DB_HScroll1_GotFocus ()
    If MODO_BUSCAR = False Then
        cod_medido.ReadOnly = True
        bNuevo.Enabled = True
    End If
    MEDIDOR_NUEVO = False
End Sub

Sub Form_Load ()
    centrarPantalla Me

    qMedidor.pWhere = "ORDER BY cod_medido"
    qMedidor.pTables = pathMedidor
    qMedidC.pTables = pathMedidor
    qMedidT.pTables = pathMedidor
    qCodMedidor.pTables = pathMedidor
    qEstMedid.pTables = pathEstMedidor
    qCodTransf.pTables = pathTransformador
    qCodCliente.pTables = pathCliente


    r% = fDoQuery(qMedidC)
    r% = fDoQuery(qMedidT)
    r% = fDoQuery(qEstMedid)
    r% = fDoQuery(qMedidor)

    r% = fEndQuery(qMedidC)
    r% = fEndQuery(qMedidT)
    r% = fEndQuery(qEstMedid)

    MEDIDOR_NUEVO = False
End Sub
```

```
FM_POSTE.FRM - 1

Dim BUSCAR_PRIMERO As Integer


Sub DB_Command1_Click ()
    sector.Text = cod_sector.Text
End Sub

Sub DB_Command2_Click ()
    r% = fClearFind(qPoste)
    'regresar campos a sólo lectura
    cod_poste.ReadOnly = True
    mapa.ReadOnly = True
    ubicacion.ReadOnly = True       'los dbtext no permiten
tildes (-:
    'habilitar botón grabar
    bGrabar.Enabled = True
    bModoBuscar.Enabled = True
End Sub

Sub DB_HScroll1_Change ()
    sector.Text = cod_sector.Text
End Sub

Sub bBuscar_Click ()
    r% = fFind(qPoste, 1)'buscar primer registro coincident
e
    'regresar campos a sólo lectura
    cod_poste.ReadOnly = True
    mapa.ReadOnly = True
    sector.ReadOnly = True
    'los dbtext no permiten tildes (-:
    'actualizar sector
    sector.Text = cod_sector.Text
    BUSCAR_PRIMERO = True
End Sub

Sub bBuscarSig_Click ()
    'regresar campos a sólo lectura
    cod_poste.ReadOnly = True
    mapa.ReadOnly = True
    sector.ReadOnly = True
    'los dbtext no permiten tildes (-:
    'actualizar sector
    sector.Text = cod_sector.Text
End Sub
```

FM_POSTE.FRM - 2

```
Sub bFinBuscar_Click ()
    r% = fClearFind(qPoste)
    'habilitar y deshabilitar botones
    bGrabar.Enabled = True
    bModoBuscar.Enabled = True
    bBuscar.Enabled = False
    bBuscarSig.Enabled = False
    bFinBuscar.Enabled = False
    If BUSCAR_PRIMERO = False Then
        r% = fDoQuery(qPoste)
    End If
    BUSCAR_PRIMERO = False
    cod_poste.ReadOnly = True
    mapa.ReadOnly = True
    sector.ReadOnly = True


End Sub

Sub bGrabar_Click ()
    If estado <> "" And altura <> "" And material <> "" And
 peso <> "" Then
        res% = fUpdate(qPoste, 0)
    Else
        MsgBox "Faltan datos por ingresar.", 64, "Sistema d
e Inventario para EMELEC"
    End If
    res% = fDoQuery(qPosteM)
    res% = fDoQuery(qPosteP)
    res% = fDoQuery(qPosteA)
End Sub

Sub bModoBuscar_Click ()
    bGrabar.Enabled = False 'deshabilitar botón grabar
    'borrar sector
    sector.ReadOnly = False
    sector.Text = ""
    sector.ReadOnly = True
    'permitir escribir en los campos
    cod_poste.ReadOnly = False
    mapa.ReadOnly = False
    sector.ReadOnly = False
    'los dbtext no permiten tildes (-;
    bModoBuscar.Enabled = False
    bBuscar.Enabled = True
    bBuscarSig.Enabled = True
```


Código del sector:
Nuevo
Eliminar
Grabar
Salir

```
FM_SECT.FRM - 1

Dim SECTOR_NUEVO As Integer


Sub bBuscar_Click ()
    Me.cod_sector.ReadOnly = True
    Me.bNuevo.Enabled = True
End Sub

Sub validarDatos ()

End Sub

Sub bEliminar_Click ()

    respuesta = MsgBox(MSGELREG, DEFELREG, TITULO)

    If respuesta = IDSI Then
        qPoste.pWhere = " cod_sector = '" + cod_sector.Text
 + "'"
        res% = fDoQuery(qPoste)

        If qPoste.pRecCount > 0 Then
            MsgBox "No puede eliminarse el registro. Exiten
 postes con ese sector", 48, TITULO
            Me.cod_sector.SetFocus
        Else
            res% = fDelete(qSector, 0)
        End If

        res% = fEndQuery(qPoste)
        Me.cod_sector.ReadOnly = True
        Me.bNuevo.Enabled = True
    End If

End Sub

Sub bGrabar_Click ()


If SECTOR_NUEVO Then 'para verificar si voy a insertar un r
egistro

    qCodSector.pWhere = " COD_SECTOR = '" + cod_sector.Text
 + "'"
    res% = fDoQuery(qCodSector)
```

FM_SECT.FRM - 2

```
    If qCodSector.pRecCount > 0 Then
        MsgBox "Ya existe un registro con ese código de sec
tor", 48, TITULO
        Me.cod_sector.SetFocus
    Else
        res% = fUpdate(qSector, 0)
        Me.cod_sector.ReadOnly = True
        Me.bNuevo.Enabled = True
        'para ordenar el registro recién ingresado
        SECTOR_NUEVO = False
        res% = fEndQuery(qSector)
        res% = fDoQuery(qSector)
    End If

    res% = fEndQuery(qCodSector)

Else
    res% = fUpdate(qSector, 0)

End If

End Sub

Sub bNuevo_Click ()
    Me.cod_sector.ReadOnly = False
    Me.cod_sector.SetFocus
    Me.bNuevo.Enabled = False
    SECTOR_NUEVO = True
End Sub

Sub bSalir_Click ()
    Unload Me
End Sub

Sub DB_HScroll1_GotFocus ()
     Me.cod_sector.ReadOnly = True
     Me.bNuevo.Enabled = True

End Sub

Sub Form_Load ()
    centrarPantalla Me
    qSector.pTables = pathSector
    qCodSector.pTables = pathSector
    qPoste.pTables = pathPoste
    r% = fDoQuery(qSector)
```

```
End Sub
```

```
FM_SUBES.FRM - 1

Dim SUBESTACION_NUEVA As Integer, MODO_BUSCAR As Integer, B
USCAR_PRIMERO As Integer


Sub bBuscar_Click ()
    r% = fFind(qSubest, 1)'buscar primer registro coinciden
te
    'regresar campos a sólo lectura
    COD_SUBEST.ReadOnly = True
    BUSCAR_PRIMERO = True
End Sub

Sub bBuscarSig_Click ()
    'regresar campos a sólo lectura
    COD_SUBEST.ReadOnly = True
End Sub

Sub bEliminar_Click ()
    respuesta = MsgBox(MSGELREG, DEFELREG, TITULO)

    If respuesta = IDSI Then
        qCircPrim.pWhere = " cod_subest = '" + COD_SUBEST.T
ext + "'"
        res% = fDoQuery(qCircPrim)

        If qCircPrim.pRecCount > 0 Then
            MsgBox "No puede eliminarse el registro. Exiten
 circuitos primarios alimentados por esa subestación.", 48,
 TITULO
            COD_SUBEST.SetFocus
        Else
            res% = fDelete(qSubest, 0)
        End If

        res% = fEndQuery(qCircPrim)
        COD_SUBEST.ReadOnly = True
        bNuevo.Enabled = True
    End If
End Sub

Sub bFinBuscar_Click ()
    r% = fClearFind(qSubest)
    'habilitar y deshabilitar botones
    bGrabar.Enabled = True
    bNuevo.Enabled = True
    bEliminar.Enabled = True
```

FM_SUBES.FRM - 2

```
    bModoBuscar.Enabled = True
    bBuscar.Enabled = False
    bBuscarSig.Enabled = False
    bFinBuscar.Enabled = False
    MODO_BUSCAR = False
    If BUSCAR_PRIMERO = False Then
        r% = fDoQuery(qSubest)
    End If
    BUSCAR_PRIMERO = False
    COD_SUBEST.ReadOnly = True

End Sub

Sub bGrabar_Click ()
If SUBESTACION_NUEVA Then 'para verificar si voy a insertar
 un registro

    qCodSubest.pWhere = " COD_SUBEST = '" + COD_SUBEST.Text
 + "'"
    res% = fDoQuery(qCodSubest)

    If qCodSubest.pRecCount > 0 Then
        MsgBox "Ya existe un registro con ese código .", 48
, TITULO
        COD_SUBEST.SetFocus
    Else
        If Trim$(COD_SUBEST.Text) <> "" And Trim$(nombre.Te
xt) <> "" And IsNumeric(volt_ent.Text) And IsNumeric(volt_s
al.Text) And IsNumeric(volt_gen.Text) Then
            res% = fUpdate(qSubest, 0)
            SUBESTACION_NUEVA = False
            COD_SUBEST.ReadOnly = True
            bNuevo.Enabled = True
            'para ordenar el registro recién ingresado
            res% = fEndQuery(qSubest)
            res% = fDoQuery(qSubest)
        Else
            MsgBox "Faltan datos o datos incorrectos .", 48
, TITULO
            COD_SUBEST.SetFocus
        End If
    End If

    res% = fEndQuery(qCodSubest)

Else
```

```
    If Trim$(nombre.Text) <> "" And IsNumeric(volt_ent.Text
) And IsNumeric(volt_sal.Text) And IsNumeric(volt_gen.Text)
 Then
        res% = fUpdate(qSubest, 0)
    Else
        MsgBox "Faltan datos o datos incorrectos .", 48, TI
TULO
    End If
End If
End Sub

Sub bModoBuscar_Click ()
    'deshabilitar botones grabar, nuevo y eliminar
    bGrabar.Enabled = False
    bNuevo.Enabled = False
    bEliminar.Enabled = False
    'permitir escribir en los campos
    COD_SUBEST.ReadOnly = False
    'habilitar y deshabilitar botones de búsqueda
    bModoBuscar.Enabled = False
    bBuscar.Enabled = True
    bBuscarSig.Enabled = True
    bFinBuscar.Enabled = True
    MODO_BUSCAR = True
    COD_SUBEST.SetFocus

End Sub

Sub bNuevo_Click ()
    COD_SUBEST.ReadOnly = False
    COD_SUBEST.SetFocus
    bNuevo.Enabled = False
    SUBESTACION_NUEVA = True
End Sub

Sub bSalir_Click ()
    Unload Me
End Sub

Sub DB_HScroll1_GotFocus ()
    If MODO_BUSCAR = False Then
        COD_SUBEST.ReadOnly = True
        bNuevo.Enabled = True
    End If
    SUBESTACION_NUEVA = False
End Sub
```

FM_SUBES.FRM - 4

```
Sub Form_Load ()
    centrarPantalla Me
    SUBESTACION_NUEVA = False
End Sub
```

FM_TRANS.FRM - 1

```
Dim BUSCO_PRIMERO As Integer


Sub bBuscar_Click ()
    r% = fFind(qTran, 1)'buscar primer registro coincidente
    'regresar campos a sólo lectura
    BUSCO_PRIMERO = True
    cod_transf.ReadOnly = True
    cod_sec.ReadOnly = True
    cod_prima.ReadOnly = True
    cod_poste.ReadOnly = True

End Sub

Sub bBuscarSig_Click ()
    'regresar campos a sólo lectura
    cod_transf.ReadOnly = True
    cod_sec.ReadOnly = True
    cod_prima.ReadOnly = True
    cod_poste.ReadOnly = True



End Sub

Sub bFinBuscar_Click ()
    r% = fClearFind(qTran)
    'habilitar y deshabilitar botones
    bGrabar.Enabled = True
    bModoBuscar.Enabled = True
    bBuscar.Enabled = False
    bBuscarSig.Enabled = False
    bFinBuscar.Enabled = False

    If BUSCAR_PRIMERO = False Then
        r% = fDoQuery(qTran)
    End If
    BUSCAR_PRIMERO = False
    cod_transf.ReadOnly = True
    cod_sec.ReadOnly = True
    cod_prima.ReadOnly = True
    cod_poste.ReadOnly = True


End Sub
```

FM_TRANS.FRM - 2

```
Sub bModoBuscar_Click ()
    bGrabar.Enabled = False 'deshabilitar botón grabar
    'permitir escribir en los campos
    cod_transf.ReadOnly = False
    cod_sec.ReadOnly = False
    cod_prima.ReadOnly = False
    cod_poste.ReadOnly = False
    sector.ReadOnly = False
    sector.Text = ""
    sector.ReadOnly = True
    'habilitar y deshabilitar botones
    bModoBuscar.Enabled = False
    bBuscar.Enabled = True
    bBuscarSig.Enabled = True
    bFinBuscar.Enabled = True
    cod_transf.SetFocus


End Sub

Sub bSalir_Click ()
    Unload Me
End Sub

Sub DB_HScroll1_Change ()
    sector.Text = cod_sector.Text
End Sub

Sub Form_Load ()

    centrarPantalla Me

    qTran.pTables = pathTransformador
    qSector.pTables = pathSector
    qEstTran.pTables = pathEstTransformador
    qTranF.pTables = pathTransformador

    r% = fDoQuery(qSector)
    r% = fDoQuery(qEstTran)
    r% = fDoQuery(qTranF)
    r% = fDoQuery(qTran)


    r% = fEndQuery(qSector)
    r% = fEndQuery(qEstTran)
```

FM_TRANS.FRM - 3

```
    r% = fEndQuery(qTranF)

    sector.Text = cod_sector.Text
    'capacidad.SetFocus

End Sub
```

```
Sub archivoSalir_Click ()
    End
End Sub

Sub ayudaAcercaDe_Click ()
    formAcercaDe.Show
End Sub

Sub consConexIleg_Click ()
    FormCConexionesIlegales.Show
End Sub

Sub consDispPorPoste_Click ()
    FormConsultaDispositivosPorPoste.Show
End Sub

Sub consultasCircuitosPrimarios_Click ()
    FormConsultaCircPrim.Show
End Sub

Sub consultasCircuitosSecundarios_Click ()
    FormConsultaCircSec.Show
End Sub

Sub consultasClientesGeneral_Click ()
    FormCClientes.Show
End Sub

Sub consultasClientesPorTransf_Click ()
    FormConsultaClientesPorTransformador.Show
End Sub

Sub consultasMedidores_Click ()
    FormCMedidores.Show
End Sub

Sub consultasPostesGeneral_Click ()
    FormConsultaPoste.Show
End Sub

Sub consultasPostesPorCircPrim_Click ()
    FormConsultaPostesPorCircuitoPrimario.Show
End Sub
```

MAIN.FRM - 2

```
Sub consultasPostesPorCircSec_Click ()
    FormConsultaPostesPorCircuitoSecundario.Show
End Sub

Sub consultasTransfGeneral_Click ()
    FormConsultaTransformador.Show
End Sub

Sub consultasTransfPorCircPrim_Click ()
    FormConsultaTransformadorPorCircPrim.Show
End Sub

Sub mantenimientoCircPrim_Click ()
    FormMCircPrim.Show
End Sub

Sub mantenimientoCircSec_Click ()
    FormMCircSec.Show
End Sub

Sub mantenimientoClientes_Click ()
    FormMClientes.Show
End Sub

Sub mantenimientoConexionesIlegales_Click ()
    FormMConexIleg.Show
End Sub

Sub mantenimientoMedidores_Click ()
    formMMedidor.Show
End Sub

Sub mantenimientoPostes_Click ()
    FormMPoste.Show
End Sub

Sub mantenimientoSectores_Click ()
    FormMSectores.Show
End Sub

Sub mantenimientoSubestaciones_Click ()
    FormMSubestaciones.Show
End Sub

Sub mantenimientoTransformadores_Click ()
    FormMTransfor.Show
```

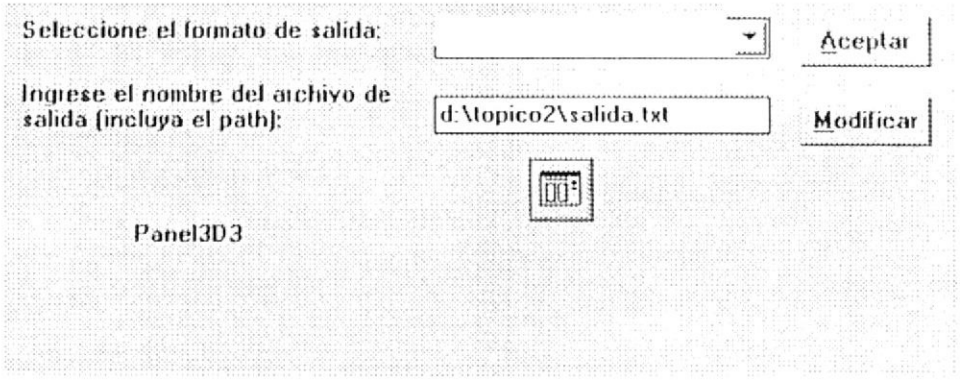
```
End Sub

Sub mantEstCircPrim_Click ()
    FormMEstCircPrim.Show
End Sub

Sub mantEstCircSec_Click ()
    FormMEstCircSec.Show
End Sub

Sub mantEstMedidores_Click ()
    FormMEstMed.Show
End Sub

Sub mantEstPostes_Click ()
    FormMEstPoste.Show
End Sub

Sub mantEstTrans_Click ()
    FormMEstTran.Show
End Sub

Sub MDIForm_Load ()

    pathPoste = "poste.dbf"
    pathEstPoste = "est_post.dbf"
    pathSector = "sector.dbf"
    pathCircPrim = "c_prim.dbf"
    pathEstCircPrim = "est_cpri.dbf"
    pathSubestacion = "subest.dbf"
    pathTransformador = "transfor.dbf"
    pathEstTransformador = "est_tran.dbf"
    pathCircSec = "c_secund.dbf"
    pathEstCircSec = "est_csec.dbf"
    pathMedidor = "medidor.dbf"
    pathEstMedidor = "est_medi.dbf"
    pathCliente = "cliente.dbf"
    pathCnxIleg = "cnx_ileg.dbf"



End Sub

Sub repCircPrimPorEstado_Click ()
    nombre_reporte = "primxest.rpt"
    Reportes.Show
```

```
End Sub

Sub repCircPrimPorPoste_Click ()
    nombre_reporte = "primxpos.rpt"
    Reportes.Show
End Sub

Sub repCircPrimPorSubest_Click ()
    nombre_reporte = "primxsub.rpt"
    Reportes.Show
End Sub

Sub repClientesGeneral_Click ()
    nombre_reporte = "cliegene.rpt"
    Reportes.Show
End Sub

Sub repConexIlegPorCS_Click ()
    nombre_reporte = "ilegxsec.rpt"
    Reportes.Show
End Sub

Sub reportesCircuitosPorEstado_Click ()
   nombre_reporte = "secxesse.rpt"
   Reportes.Show
End Sub

Sub reportesCircuitosPorPoste_Click ()
   nombre_reporte = "secxpost.rpt"
   Reportes.Show
End Sub

Sub reportesClientesPorTransforadores_Click ()
   nombre_reporte = "cliextra.rpt"
   Reportes.Show
End Sub

Sub reportesMedidoresPorCliente_Click ()
   nombre_reporte = "medxclie.rpt"
   Reportes.Show
End Sub

Sub reportesMedidoresPorTrans_Click ()
   nombre_reporte = "medxtran.rpt"
   Reportes.Show
End Sub
```

```
Sub reportesPostesPorCircuitoPrimario_Click ()
   nombre_reporte = "postxpri.rpt"
   Reportes.Show
End Sub

Sub reportesPostesPorEstado_Click ()
   nombre_reporte = "postxest.rpt"
   Reportes.Show
End Sub

Sub reportesPostesPorSecundario_Click ()
    nombre_reporte = "postxsec.rpt"
    Reportes.Show
End Sub

Sub reportesTransformadoresPorEstado_Click ()
   nombre_reporte = "traxesse.rpt"
   Reportes.Show
End Sub

Sub reportesTransformadoresPorPrimario_Click ()
   nombre_reporte = "tranxpri.rpt"
   Reportes.Show
End Sub

Sub reportesTransformmadoresDañados_Click ()
   nombre_reporte = "transdan.rpt"
   Reportes.Show
End Sub

Sub repPostesDanados_Click ()
    nombre_reporte = "postdana.rpt"
    Reportes.Show
End Sub

Sub repPostesDanadosConTrans_Click ()
    nombre_reporte = "posdatra.rpt"
    Reportes.Show
End Sub

Sub repPostesGeneral_Click ()
    nombre_reporte = "postgene.rpt"
    Reportes.Show
End Sub
```

MAIN.FRM - 6

```
Sub repTransGeneral_Click ()
    nombre_reporte = "trangene.rpt"
    Reportes.Show
End Sub
```

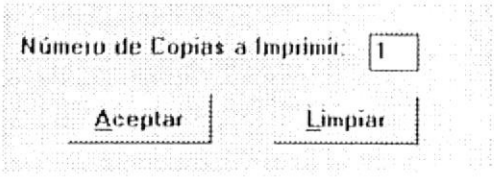

Seleccione el formato de salida:
Aceptar
Ingrese el nombre del archivo de salida (incluya el path):
d:\topico2\salida.txt
Modificar
Panel3D3

```
Sub Command1_Click ()
'    Output_File_Name = OutputFileName.Text
    'Output_File_Name = cmd_savereporte.Filename
    Rem - Set PrinterFileType property
'     If OutputFiletype = "Record Format" Then
'         Output_File_Type = 0
'     Else
'         If OutputFiletype = "Tab Seperated" Then
'             Output_File_Type = 1
'         Else
'             If OutputFiletype = "Text Format" Then
'                 Output_File_Type = 2
'             Else
'                 If OutputFiletype = "DIF Format" Then
'                     Output_File_Type = 3
'                 Else
'                     If OutputFiletype = "Comma Seperated V
alue" Then
'                         Output_File_Type = 4
'                     Else
'                        If OutputFiletype = "Tab Seperated
Text" Then
'                             Output_File_Type = 6
'                         End If
'                    End If
'                 End If
'             End If
'         End If
'     End If

'Output_File_Type = 1
Rem Close form
    Unload ReporteImprimir
End Sub

Sub Command2_Click ()
'     OutputFiletype.SetFocus
End Sub

Sub DB_Command1_Click ()
'    Output_File_Name = OutputFileName.Text

'    Rem - Set PrinterFileType property
'     If OutputFiletype = "Record Format" Then
```

```
'         Output_File_Type = 0
'     Else
'         If OutputFiletype = "Tab Seperated" Then
'             Output_File_Type = 1
'         Else
'             If OutputFiletype = "Text Format" Then
'                 Output_File_Type = 2
'             Else
'                 If OutputFiletype = "DIF Format" Then
'                     Output_File_Type = 3
'                 Else
'                     If OutputFiletype = "Comma Seperated V
alue" Then
'                         Output_File_Type = 4
'                     Else
'                         If OutputFiletype = "Tab Seperated
 Text" Then
'                             Output_File_Type = 6
'                         End If
'                     End If
'                 End If
'             End If
'         End If
'     End If
'
'Rem Close form
'     Unload ReporteImprimir



End Sub

Sub DB_Command2_Click ()
'     OutputFiletype.SetFocus
End Sub

Sub Form_Load ()
     'OutputFiletype.AddItem "Comma Seperated"
     'OutputFiletype.AddItem "Tab Seperated"
     'OutputFiletype.AddItem "Record Format"
     'OutputFiletype.AddItem "Text Format"
     'OutputFiletype.AddItem "DIF Format"
     'OutputFiletype.AddItem "Tab Seperated Text"
     'OutputFiletype.SelText = "Comma Seperated"

    Const OFN_CREATEPROMPT = &H2000&
```

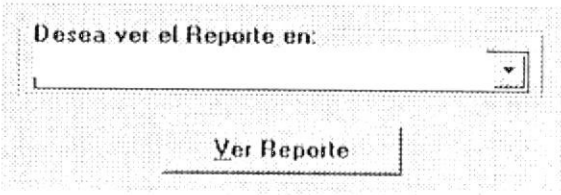
REP_ARCH.FRM - 3

```
    Const OFN_EXTENSIONDIFFERENT = &H400&
    Const OFN_OVERWRITEPROMPT = &H2&

    cmd_savereporte.Flags = &H2000& Or &H400& Or &H2&
    'cmd_savereporte.Filter = "DBase (*.dbf)|*.dbf| Access (
*.mdb)|*.mdb"
    'cmd_savereporte.Filter = "Comma Seperated|*.txt| Tab Se
perated|*.txt| Record Format|*.txt| Text Format|*.txt| DIF
Format|*.txt| Tab Seperated Text|*.txt"
    cmd_savereporte.Filter = "DBase (*.txt)|*.txt"
    cmd_savereporte.FilterIndex = 1
    cmd_savereporte.Action = 2

    'Output_File_Name = cmd_savereporte.Filename
    Nombre = cmd_savereporte.Filename
    Output_File_Type = 0
    'ls_path_name_tabla = cmd_savereporte.Filename
   ' Unload ReporteArchivo

End Sub
```


Número de Copias a Imprimir:
1
Aceptar
Limpiar

REP_IMP.FRM - 1

```
Sub Command1_Click ()
    Number_Of_Copies = Num_Copies
    Unload ReporteArchivo
End Sub

Sub Command2_Click ()
    Num_Copies.SetFocus
    Num_Copies.Text = "1"
End Sub

Sub DB_Command1_Click ()
    Number_Of_Copies = Num_Copies
    Unload ReporteArchivo
End Sub

Sub DB_Command2_Click ()
    Num_Copies.SetFocus
    Num_Copies.Text = "1"
End Sub
```


Desea ver el Reporte en:
Ver Reporte

REPORTES.FRM - 1

```
Dim StrBuffer As String * 250


Sub Command1_Click ()
    Unload Me
End Sub

Sub RangeEnd_GotFocus ()

End Sub

Sub RangeStart_GotFocus ()

End Sub

Sub Form_Load ()
    centrarPantalla Me
    OutputList.AddItem "Pantalla"
    OutputList.AddItem "Impresora"
'    OutputList.AddItem "Archivo"
    OutputList.Text = "Pantalla"
End Sub

Sub OutputList_Click ()
    If OutputList.Text = "Impresora" Then
        ReporteImprimir.Show
    Else
        If OutputList.Text = "Archivo" Then
        ReporteArchivo.Show
        End If
    End If
End Sub

Sub Print_Report_Click ()

    If OutputList.Text = "Pantalla" Then
        OutputDestination = 0
    Else
        If OutputList.Text = "Impresora" Then
            OutputDestination = 1
            reporte2.Report1.CopiesToPrinter = Number_Of_Co
pies
        Else
            If OutputList.Text = "Archivo" Then
                OutputDestination = 2
                'reporte2.Report1.PrintFileName = Output_Fi
```

```
le_Name
                reporte2.Report1.PrintFileName = "C:\diana\
topico\mario\hola.txt"
                reporte2.Report1.PrintFileType = 2
                'reporte2.Report1.PrintFileType = Output_Fi
le_Type
            End If
        End If
    End If
    reporte2.Report1.Destination = OutputDestination


'   Set location of the database file to the location from
'   which the user ran the app

    LocText$ = LCase(app.Path)
    If Right$(app.Path, 1) <> "\" Then LocText$ = LocText$
+ "\"  'handles the root

    reporte2.Report1.Connect = "DSN=EMELSIG_MS;UID=;PWD=;DS
Q=BASE"
    reporte2.Report1.ReportFileName = LocText$ + nombre_rep
orte


'  EXECUTE PRINT CALL
    On Error GoTo ErrorHandler
        reporte2.Report1.Action = 1
        Unload reportes
    Exit Sub



ErrorHandler:
    MsgBox Error$
   Exit Sub



End Sub
```

GLOBAL.BAS - 1

```
 'Definiciones para Mantenimiento
Global pathPoste As String, pathEstPoste As String, pathSec
tor As String
Global pathCircPrim As String, pathEstCircPrim As String, p
athSubestacion As String
Global pathTransformador As String, pathEstTransformador As
 String
Global pathCircSec As String, pathEstCircSec As String
Global pathMedidor As String, pathEstMedidor As String
Global pathCliente As String, pathCnxIleg As String

 'Definiciones para Reportes
Global nombre_reporte As String
Global Output_File_Type As Integer
Global Output_File_Name As String

 'Definiciones para Message Box
Global respuesta As Integer
Global Const MB_SINO = 4
Global Const MB_PREGUNTA = 32
Global Const MB_DEFBOTON2 = 256
Global Const IDSI = 6
Global Const IDNO = 7
Global Const TITULO = "Sistema de Inventario de EMELEC"
          'Título de los MsgBox de esta aplicación
Global Const MSGELREG = "Está seguro que desea eliminar el
registro?" 'Mensaje cuando se desea eliminar un registro
Global Const DEFELREG = MB_SINO + MB_PREGUNTA + MB_DEFBOTON
2         'Definición del diálogo para eliminar un registro


Sub centrarPantalla (pantalla As Form)
    pantalla.Left = (screen.Width - pantalla.Width) / 2
    pantalla.Top = (screen.Height * .8 - pantalla.Height) /
 2

End Sub

Function IsFase (valor As String) As Integer
Static ARRAY(15) As String, i As Integer, flag As Integer

ARRAY(0) = "A"
ARRAY(1) = "B"
ARRAY(2) = "C"
ARRAY(3) = "AB"
ARRAY(4) = "AC"
```

GLOBAL.BAS - 2

```
ARRAY(5) = "BC"
ARRAY(6) = "BA"
ARRAY(7) = "CA"
ARRAY(8) = "CB"
ARRAY(9) = "ABC"
ARRAY(10) = "ACB"
ARRAY(11) = "BAC"
ARRAY(12) = "BCA"
ARRAY(13) = "CAB"
ARRAY(14) = "CBA"

flag = 0
For i = 0 To 15
    If Trim$(valor) = ARRAY(i) Then
        flag = 1
        Exit For
    End If
Next i

IsFase = flag

End Function
```

CONCLUSIONES Y RECOMENDACIONES

Un Sistema de Información Geográfico es útil desde todo punto de vista, y proporciona especial ayuda para el caso de llevar el inventario de una empresa de servicios como EMELEC. El sistema diseñado de llegar a utilizarse, proporcionaría a la Empresa Eléctrica beneficios como:

- Ayuda en la determinación de problemas
- Menor tiempo de respuesta en reparaciones y reconexiones
- Facilidad para prever crecimiento futuro y cambios de infraestructura
- Determinación rápida de multas para el caso de conexiones ilegales
- Disminución en costos de operación
- Todo esto traería como resultado brindar un mejor servicio a sus abonados

El hecho de que la aplicación utilice un origen de datos para conectarse a la Base de Datos hace transparente para ella el tipo de base con que se está conectando, por lo tanto es posible cambiar el tipo de base en el origen de datos permitiendo así futuras migraciones a otro tipo de basé como Oracle, Sybase, DB2, etc..

Finalmente, esperamos que este trabajo sirva de guía para aquellos que quieran adentrarse un poco más en el amplio mundo de los Sistemas de Información Geográficos.