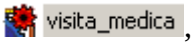
MANUAL TÉCNICO

PharmaMovil

APICACION MOVIL PARA EL PROCESO DE VISITA MEDICA

La aplicación de visita médica está desarrollada sobre la herramienta de programación PocketBuilder 2.0, que permite el desarrollo de aplicaciones orientadas a objetos.

1. Objeto Aplicación

El objeto principal que representa todo el sistema se denomina Visita_Médica 

, los eventos programados en este objeto son:

Open (string comandline) returns (none)

Descripción: Inicialmente se crea una instancia del objeto n_visita_medica_conectar que tiene la información de conexión a la base de datos, para posteriormente llamar a la función of_ConnectDb() si el resultado es igual a 0 la conexión se estableció satisfactoriamente y se procede a validar contra el registro del dispositivo si existen sincronizaciones previas RegistryGet(regpath, "MLUserName", RegString!, ls_work), sino existen sincronizaciones se invoca a la función gf_visita_medica_configure_ulsync(SQLCA) la cual realiza la sincronización de la información. Una vez realizado este procedimiento se procede a la apertura de la ventana w_login.

Código:

```
n_visita_medica_conectar lnv_connectserv
lnv_connectserv = Create n_visita_medica_conectar
If lnv_connectserv.of_ConnectDB ( ) = 0 Then
        nvo_visita_medica_ulsync uosync
        String ls_work
        String regpath
        integer rc
        integer registros

        // obtiene la cadena de registro para esta aplicación
        uosync = CREATE nvo_visita_medica_ulsync
        regpath = uosync.APP_REGPATH
        destroy uosync

        // Chequea que exista el id de usuario en el registro
        rc = RegistryGet(regpath, "MLUserName", RegString!, ls_work)

        //g_rep_codigo = ls_work

        if rc <> 1 then
                // Responde por id de usuario si no existe
                gf_visita_medica_configure_ulsync(SQLCA)

                // Si el empleado no existe cierra la sincronización
                rc = RegistryGet(regpath, "MLUserName", RegString!, ls_work)
                        if rc <> 1 then
                                MessageBox("Error Sincronización", "Opciones en el menú y establezca el
                                usuario de MobilLink.")
                        end if
        else
                //Se verifica si existen registros en los parámetros
                select count(*) into :registros
                from cvm_usuarios;

                if (registros < 1) then
                        gf_visita_medica_configure_ulsync(SQLCA)
```

```
                    else
                            Open ( w_login)
                    end if
            end if
end If
Destroy lnv_connectserv
```

2. Objeto n_visita_medica_conectar

Permite la conexión a la base de datos ultralite tiene tres funciones principales: of_getconnection_info(), of_connectdb() y of_disconnectdb()

Función of_getConnection_info()

integer of_getconnectioninfo (ref string as_dbms, ref string as_database, ref string as_userid, ref string as_dbpass, ref string as_logid, ref string as_logpass, ref string as_server, ref string as_dbparm, ref string as_lock, ref string as_autocommit)

Descripción: Recibe como parámetros usuario, contraseña, base de datos, servidor, tipo de dbms y el tipo de método que va a utilizar para validar la información.
El tipo puede ser un archivo .ini entonces se hace un llamado a la función ProfileString(), si el tipo de método es de registro se llama a la funcion RegistryGet() y si el tipo es un script se compara directamente con los valores ingresados. El origen de la información de conexión puede cambiarse alterando el valor de la variable "is_connectfrom".

Código:

```
Choose Case is_connectfrom

        Case IS_USE_INIFILE         /*  Conexión a la base de datos a través de un archive INI  */
                string ls_inifile = ""
                as_dbms             = ProfileString ( ls_inifile, "Database", "DBMS", "UL9")
                as_database         = ProfileString ( ls_inifile, "Database", "Database", "")
                as_userid           = ProfileString ( ls_inifile, "Database", "UserID", "")
                as_dbpass           = ProfileString ( ls_inifile, "Database", "DBPass", "")
                as_logid            = ProfileString ( ls_inifile, "Database", "LogID", "")
                as_logpass          = ProfileString ( ls_inifile, "Database", "LogPassword","")
                as_server           = ProfileString ( ls_inifile, "Database", "Servername", "")
                as_dbparm           = ProfileString ( ls_inifile, "Database", "DBParm",
                                      "ConnectString='DBF=D:\AppTesis\udb\bdd_ult_cvm.udb;UID=dba;PWD=sql'")
               as_dbparm            = ProfileString (ls_inifile, "Database", "DBParm",
                                      "ConnectString='DBF=\Program   Files\bdd_ult_cvm.udb;UID=dba;PWD=sql'")
                as_lock             = ProfileString ( ls_inifile, "Database", "Lock", "")
                as_autocommit       = ProfileString ( ls_inifile, "Database", "AutoCommit",          "false")

        Case IS_USE_REGISTRY    /*  Conexión a la base de datos a través de Registro  */
                string ls_registrykey = "" + "\DataBase"

                If RegistryGet ( ls_registrykey, "DBMS",                    RegString!, as_dbms ) <> 1 Then
                        RegistrySet ( ls_registrykey, "DBMS",               RegString!, "UL9" )
                        RegistryGet ( ls_registrykey, "DBMS",               RegString!, as_dbms )
                End If
                If RegistryGet ( ls_registrykey, "Database",        RegString!, as_database ) <> 1 Then
                        RegistrySet ( ls_registrykey, "Database",           RegString!, "" )
```

```
                RegistryGet ( ls_registrykey, "Database",          RegString!, as_database )
            End If
            If RegistryGet ( ls_registrykey, "UserID",                  RegString!, as_userid ) <> 1 Then
                RegistrySet ( ls_registrykey, "UserID",             RegString!, "" )
                RegistryGet ( ls_registrykey, "UserID",             RegString!, as_userid )
            End If
            If RegistryGet ( ls_registrykey, "DBPass",                  RegString!, as_dbpass ) <> 1 Then
                RegistrySet ( ls_registrykey, "DBPass",             RegString!, "" )
                RegistryGet ( ls_registrykey, "DBPass",                     RegString!, as_dbpass )
            End If
            If RegistryGet ( ls_registrykey, "LogID",           RegString!, as_logid ) <> 1 Then
                RegistrySet ( ls_registrykey, "LogID",              RegString!, "" )
                RegistryGet ( ls_registrykey, "LogID",              RegString!, as_logid )
            End If
            If RegistryGet ( ls_registrykey, "LogPassword", RegString!, as_logpass ) <> 1 Then
                RegistrySet ( ls_registrykey, "LogPassword", RegString!, "" )
                RegistryGet ( ls_registrykey, "LogPassword", RegString!, as_logpass )
            End If
            If RegistryGet ( ls_registrykey, "Servername",      RegString!, as_server ) <> 1 Then
                RegistrySet ( ls_registrykey, "Servername",         RegString!, "" )
                RegistryGet ( ls_registrykey, "Servername",         RegString!, as_server )
            End If
            If RegistryGet ( ls_registrykey, "DBParm",                  RegString!, as_dbparm ) <> 1 Then
                RegistrySet ( ls_registrykey, "DBParm",                     RegString!,
                "ConnectString='DBF=D:\AppTesis\udb\bdd_ult_cvm.udb;UID=dba;PWD=sql'" )
                RegistrySet ( ls_registrykey, "DBParm",                     RegString!,
                "ConnectString='DBF=\Program Files\bdd_ult_cvm.udb;UID=dba;PWD=sql'")
                RegistryGet ( ls_registrykey, "DBParm",                     RegString!, as_dbparm )
            End If
            If RegistryGet ( ls_registrykey, "Lock",                    RegString!, as_lock ) <> 1 Then
                RegistrySet ( ls_registrykey, "Lock",                       RegString!, "" )
                RegistryGet ( ls_registrykey, "Lock",                       RegString!, as_lock )
            End If
            If RegistryGet ( ls_registrykey, "AutoCommit",      RegString!, as_autocommit ) <> 1 Then
                RegistrySet ( ls_registrykey, "AutoCommit",         RegString!, "false" )
                RegistryGet ( ls_registrykey, "AutoCommit",         RegString!, as_autocommit )
            End If

        Case IS_USE_SCRIPT      /*  Conexión a la base de datos a través de un Script  */
            as_dbms         = "UL9"
            as_database     = ""
            as_userid       = ""
            as_dbpass       = ""
            as_logid        = ""
            as_logpass      = ""
            as_server       = ""
            as_dbparm       = "ConnectString='DBF=D:\AppTesis\udb\bdd_ult_cvm.udb;UID=dba;PWD=sql'"
            as_dbparm       = "ConnectString='DBF=\Program Files\bdd_ult_cvm.udb;UID=dba;PWD=sql'"
            as_lock         = ""
            as_autocommit   = "false"

        Case Else

            Return -1
End Choose
Return 1
```

Función of_Connectdb()

public function integer of_connectdb ()

Descripción: Invoca a la función of_getconnection_info() y con esta información procede a realizar la conexión a la base de datos, si es satisfactoria devuelve el valor de conexión caso contrario devuelve (-1).

Código:

```
Connect using SQLCA;
If SQLCA.SQLCode <> 0 Then
        MessageBox ("No se puede conectar a la base de datos", SQLCA.SQLErrText)
End If
Return SQLCA.SQLCode
```

Función of_disconnectdb()

public function integer of_disconnectdb ()

Descripción: Desconecta de la base de datos

Código:

```
Disconnect using SQLCA;
If SQLCA.SQLCode <> 0 Then
        MessageBox ("No se puede desconectar de la BDD", SQLCA.SQLErrText )
End If

Return SQLCA.SQLCode
```

3. Objeto w_login

Permite el ingreso a la aplicación mediante el ingreso del nombre de usuario y contraseña. La ventana contiene el botón cbaceptar cuyo evento clicked realiza lo siguiente:

Evento Clicked del botón cbaceptar:

Descripción: Utiliza los cuadros de texto sleusr y slepsw, los cuales contienen la información de usuario y contraseña, verifica que no estén en blanco, si lo están emite mensaje de error, caso contrario, consulta a la base de datos si el usuario y contraseña existen, si no existen emite mensaje de error, caso contrario admite el acceso al sistema, y llama a la ventana w_principal.

Código:

```
string verifica_usr
if trim(sleusr.text) = "" or trim(slepsw.text) = "" then
        MessageBox("Datos incompletos", "No ingresó datos requeridos para continuar.")
else
  SELECT "cvm_usuarios"."usr_cod_rep"
    INTO :g_rep_codigo
    FROM "cvm_usuarios"
  WHERE ( "cvm_usuarios"."usr_cod_rep" = :sleusr.text ) AND
       ( "cvm_usuarios"."usr_password" = :slepsw.text ) using sqlca;
        if trim(g_rep_codigo) = ""      then
                MessageBox("Usuarios", "Usuario no existe")
        else

          SELECT "cvm_cab_boleta"."cli_codigo"
                INTO :verifica_usr
                FROM "cvm_cab_boleta"
```

```
                        WHERE ( "cvm_cab_boleta"."rep_codigo" = :g_rep_codigo ) using sqlca;

                        if trim(verifica_usr) = "" or isnull(verifica_usr) then
                                MessageBox("Usuarios", "Usuario no autorizado")
                        else
                          open(w_principal)
                                close(w_login)
                        end if
                end if
        end if
```

4. Objeto w_principal

Contiene las opciones principales del sistema de visita médica, los principales botones que presenta esta pantalla son cb_1, cb_2, cb_3, cb_4.

Evento clicked del botón cb_1

Llama a evento open de la ventana w_registro_visita

Evento clicked del botón cb_2

Llama a evento open de la ventana w_mantenimiento_clientes

Evento clicked del botón cb_3

Llama a evento open de la ventana w_pedidos_cliente

Evento clicked del botón cb_4

Llama a evento open de la ventana w_reportes

5. Objeto w_registro_visita

Descripción: El evento Open de esta ventana permite utilizar un objeto de transacción, en este caso sqlca el cual suministra la información necesaria para comunicarse con la base de datos. Luego, realiza el retrieve de los datos tomando como base el parámetro g_rep_codigo, a continuación hace referencia a la función fsub_verifica_visita().

Código:

```
tab_1.tabpage_1.dw_1.settransobject(sqlca)
tab_1.tabpage_2.dw_2.settransobject(sqlca)
tab_1.tabpage_3.dw_3.settransobject(sqlca)
tab_1.tabpage_1.dw_1.retrieve(g_rep_codigo)

if tab_1.tabpage_1.dw_1.rowcount( ) <> 0 then
      CabBoleta=tab_1.tabpage_1.dw_1.getitemstring(tab_1.tabpage_1.dw_1.getrow(
                                                                                ),"cvm_cab_boleta_cab_bol_codigo")
end if

tab_1.tabpage_1.text = "Datos"
tab_1.tabpage_2.text = "Registro"
tab_1.tabpage_3.text = "Productos"

fsub_verifica_visita()
```

Variable: g_rep_codigo

Variable global que contiene el código del representante que ha ingresado al sistema.

Función fsub_verifica_visita()

Descripción: Función que permite verificar si en la boleta actual se encuentra registrada la visita y la entrega de productos. Bandera, es una variable global que indica si la boleta ha sido visitada, no visitada o no registrada. Para saber si se los productos se encuentran registrados se realiza una consulta a la base de datos tomando como parámetro el código de la boleta y consultando a la tabla cvm_det_boleta.

Código:

```
string TStr_boleta
int Tint_Cant_Real = -1 //se registraron todos los productos

tab_1.tabpage_2.dw_2.accepttext( ) //acepto todos los cambios en el datawindow de registro de visita

if tab_1.tabpage_1.dw_1.rowcount( ) <> 0 then //si existen boletas
        tab_1.triggerevent(selectionchanged!)
        //almaceno codigo de boleta en variable TStr_boleta
        TStr_boleta=tab_1.tabpage_1.dw_1.getitemstring(tab_1.tabpage_1.dw_1.getrow(),"cvm_cab_boleta_cab_b
                ol_codigo")

        if bandera = "S" then //si se visitó
                tab_1.tabpage_1.rb_visita.checked = true
                tab_1.tabpage_3.cb_1.enabled = true
        elseif bandera = "N" then //no se visitó
                tab_1.tabpage_1.rb_visita.checked = true
                tab_1.tabpage_3.cb_1.enabled = false
        elseif trim(bandera) = "" or isnull(bandera) or trim(bandera) = "G" then //falta por registrar visita
                tab_1.tabpage_1.rb_visita.checked = false
                tab_1.tabpage_3.cb_1.enabled = false
        end if

        //consulto si cantidad real de producto se encuentra vacío significa que aún no se registra los productos
          SELECT cvm_det_boleta.det_bol_cant_real
                 INTO :TInt_Cant_Real
                 FROM cvm_cab_boleta,
                                cvm_det_boleta
                WHERE ( cvm_cab_boleta.cab_bol_codigo = cvm_det_boleta.cab_bol_codigo ) and
                                ( cvm_cab_boleta.cab_bol_codigo = :TStr_boleta )  and
                                ( (cvm_det_boleta.det_bol_cant_real = 0)  )  ;

        if TInt_Cant_Real = 0 and tab_1.tabpage_3.cb_1.enabled = true and bandera = "S" then
                tab_1.tabpage_1.rb_productos.checked = false
        elseif TInt_Cant_Real = 0 and tab_1.tabpage_3.cb_1.enabled = false and bandera = "N" then
                tab_1.tabpage_1.rb_productos.checked = true
        elseif TInt_Cant_Real = -1 and tab_1.tabpage_3.cb_1.enabled = true and bandera = "S" then
                tab_1.tabpage_1.rb_productos.checked = true
        elseif TInt_Cant_Real = -1 and tab_1.tabpage_3.cb_1.enabled = true and bandera = "S" then
                tab_1.tabpage_1.rb_productos.checked = false
        elseif TInt_Cant_Real = 0 and tab_1.tabpage_3.cb_1.enabled = false and (trim(bandera) = "" or
isnull(bandera) or trim(bandera) = "G") then
                tab_1.tabpage_1.rb_productos.checked = false
        elseif TInt_Cant_Real = -1 and tab_1.tabpage_3.cb_1.enabled = false and (trim(bandera) = "" or
isnull(bandera) or trim(bandera) = "G") then
                tab_1.tabpage_1.rb_productos.checked = false
        end if
end if
```

Evento selectionchanged del objeto tab_1

Descripción: Verifica si se ha cambiado entre Pages, tomando como base el código de la boleta, consulta si la boleta ha sido visitada (S), no visitada (N) o generada (G), de acuerdo a esto habilita o deshabilita los controles grabar, rb_visita, rb_no_visita, o el datawindows para el ingreso de productos. Cuando el campo registro de visita se encuentra vacío tanto el datawindow de productos como los controles drop down listbox de visita y no visita se encuentran deshabilitados.

Código:

```
string fecha, ori, cau,nom,nom1
string oricodigo,orinombre

if tab_1.tabpage_1.dw_1.rowcount( ) <> 0 then  //si existen boletas

tab_1.tabpage_2.ddlb_origen.reset( ) //limpia el control drop down list box de origen de visita

////lleno ddlb_causas
DECLARE ori_cur CURSOR FOR
SELECT cvm_origen_causas_no_visita.ori_cau_codigo, ori_cau_nombre FROM cvm_origen_causas_no_visita;
OPEN ori_cur;
        DO WHILE sqlca.sqlcode = 0
                FETCH ori_cur INTO :oricodigo,:orinombre;
                if SQLCA.sqlcode=0 then
                                tab_1.tabpage_2.ddlb_origen.AddItem(orinombre)
                end if
        LOOP
CLOSE ori_cur;

tab_1.tabpage_2.ddlb_origen.TriggerEvent(selectionchanged!)

if tab_1.tabpage_1.dw_1.rowcount( ) <> 0 then //asigna a cabboleta el código de boleta
        CabBoleta               =               tab_1.tabpage_1.dw_1.getitemstring(tab_1.tabpage_1.dw_1.getrow(
),"cvm_cab_boleta_cab_bol_codigo")
        tab_1.tabpage_2.dw_2.settransobject(sqlca)
        tab_1.tabpage_2.dw_2.retrieve(CabBoleta) //llena los datos dependiendo del código de boleta a la visita
        //almacena los valores de S (si visita), N (no visita), o G (generada)  a la variable bandera
        bandera = tab_1.tabpage_2.dw_2.getitemstring( tab_1.tabpage_2.dw_2.getrow( ), "cab_bol_bandera")
        tab_1.tabpage_3.dw_3.settransobject(sqlca)
        tab_1.tabpage_3.dw_3.retrieve(CabBoleta) //llena el datawindow de productos

        if tab_1.selectedtab = 3 then //si estoy en produtos
                if bandera = "N" then //si no visito
                        tab_1.tabpage_3.dw_3.enabled = false
                        tab_1.tabpage_3.cb_1.enabled = false

                elseif trim(bandera) = "" or isnull(bandera) then //si es generada
                        tab_1.tabpage_3.dw_3.enabled = false
                        tab_1.tabpage_3.cb_1.enabled = false
                        tab_1.tabpage_2.cb_2.enabled = false

                elseif bandera = "S" then //si es una visita
                        tab_1.tabpage_3.dw_3.enabled = true
                        tab_1.tabpage_3.cb_1.enabled = true
                end if
        elseif tab_1.selectedtab = 2 then

        if bandera = "N" then //si no se ha registrado la visita
                tab_1.tabpage_2.dw_2.Object.cab_bol_fecha.TabSequence = 20
                tab_1.tabpage_2.dw_2.Object.cab_bol_observaciones.TabSequence = 30
                fecha = string(tab_1.tabpage_2.dw_2.object.cab_bol_fecha[tab_1.tabpage_2.dw_2.getrow()])
                ori = tab_1.tabpage_2.dw_2.object.ori_cau_codigo[tab_1.tabpage_2.dw_2.getrow()]
                cau = tab_1.tabpage_2.dw_2.object.cau_no_vis_codigo[tab_1.tabpage_2.dw_2.getrow()]

                //SACO LA DESCRIPCION EN BASE A LOS CODIGOS

                SELECT ori_cau_nombre INTO :nom FROM cvm_origen_causas_no_visita
                WHERE ori_cau_codigo = :ori;

                SELECT cau_no_vis_descripcion INTO :nom1 FROM cvm_causas_no_visita
                WHERE cau_no_vis_codigo = :cau;

                tab_1.tabpage_2.ddlb_origen.selectitem(nom, 1)

                tab_1.tabpage_2.ddlb_origen.TriggerEvent(selectionchanged!)

                tab_1.tabpage_2.ddlb_causas.selectitem(nom1,1)

                tab_1.tabpage_2.ddlb_causas.enabled = true
                tab_1.tabpage_2.ddlb_origen.enabled = true
                if isnull(ori) or isnull(cau) or isnull(fecha) then
```

```
                        tab_1.tabpage_2.cb_2.enabled = false
                else
                        tab_1.tabpage_2.cb_2.enabled = true
                end if

        elseif trim(bandera) = "" or isnull(bandera) then
                tab_1.tabpage_2.ddlb_causas.enabled = false
                tab_1.tabpage_2.ddlb_origen.enabled = false
                tab_1.tabpage_2.dw_2.Object.cab_bol_fecha.TabSequence = 0
                tab_1.tabpage_2.dw_2.Object.cab_bol_observaciones.TabSequence = 0
                tab_1.tabpage_2.cb_2.enabled = false
        elseif bandera = "S" then
                tab_1.tabpage_2.dw_2.Object.cab_bol_fecha.TabSequence = 20
                tab_1.tabpage_2.dw_2.Object.cab_bol_observaciones.TabSequence = 30
                tab_1.tabpage_2.ddlb_causas.enabled = false
                tab_1.tabpage_2.ddlb_origen.enabled = false
                if isnull(fecha) then
                tab_1.tabpage_2.cb_2.enabled  = false
        else
                tab_1.tabpage_2.cb_2.enabled = true
        end if
        end if
elseif tab_1.selectedtab = 1 then
        if bandera = "S" or bandera = "N" then
                tab_1.tabpage_1.rb_visita.checked = true
        else
                tab_1.tabpage_1.rb_visita.checked = false
        end if

end if
End if
else
        tab_1.tabpage_2.ddlb_causas.enabled = false
        tab_1.tabpage_2.ddlb_origen.enabled = false
        tab_1.tabpage_2.cb_2.enabled = false
end if
```

Evento clicked del botón cb_2 del tab registro_visita

Descripción: El botón cb_2, permite grabar los cambios realizados al datawindow dw_2 que representa el registro de la visita o de la no visita. Realiza consultas directas a la base de datos para conoce los códigos de los dropdown listbox de la visita y no visita en base a la descripción. Se utiliza la función interna del datawindow Update() para realizar la grabación a la base de datos. Si update() = a 1 entonces la grabación es exitosa, caso contrario emite un mensaje de error.

Código:

```
string origen, causas

//código de origen no visita
SELECT ori_cau_codigo INTO :origen FROM cvm_origen_causas_no_visita
WHERE ori_cau_nombre = :tab_1.tabpage_2.ddlb_origen.text;

//código de causas no visita
SELECT cvm_causas_no_visita.cau_no_vis_codigo INTO :causas FROM cvm_causas_no_visita
WHERE cau_no_vis_descripcion = :tab_1.tabpage_2.ddlb_causas.text;

//insercion en datawindow de la cabecera de boleta
tab_1.tabpage_2.dw_2.object.ori_cau_codigo[tab_1.tabpage_2.dw_2.getrow()]=origen
tab_1.tabpage_2.dw_2.object.cau_no_vis_codigo[tab_1.tabpage_2.dw_2.getrow()]=causas
tab_1.tabpage_2.dw_2.accepttext( ) //datos cabecera

//manejo de la transaccion
if tab_1.tabpage_2.dw_2.update()=1 then //se hizo update
        commit using sqlca; //entonces commit
        tab_1.tabpage_2.dw_2.ResetUpdate()
```

```
		Messagebox("Registro de visita","Grabación exitosa")
		FSub_Valor_Cero()
		fsub_verifica_visita() //habilita o deshabilita los cheked del registro de  visita y registro de productos
else
		Messagebox("Error al Grabar","Registro Visita")
		rollback using sqlca;
end if
```

Evento clicked del botón cb_1 del tab registro_productos

Descripción: El botón cb_1, permite grabar los cambios realizados al datawindow dw_3 que representa el registro de los productos entregados al cliente. Por cada registro de productos se verifica que el valor ingresado sea diferente de cero o valor ingresado sea menor que el planificado si es así hace update() caso contrario emite mensaje de error.

Código:

```
integer cantreal, cantplan, reg_total, i, verifica = 0

tab_1.tabpage_3.dw_3.accepttext( ) //acepto cambios al datawindow de productos

reg_total = tab_1.tabpage_3.dw_3.rowcount( )

if reg_total  <> 0 then //si existen productos
	for i = 1 to reg_total //para cada registro de productos
		cantreal = tab_1.tabpage_3.dw_3.getitemnumber(i,"cvm_det_boleta_det_bol_cant_real_1")
		cantplan = tab_1.tabpage_3.dw_3.getitemnumber(i,"cvm_det_boleta_det_bol_cant_plan")
		//verifico que se haga ingresado un valor o valor ingresado sea menor al planificado
		if cantreal > cantplan or cantreal = 0.00 then
			verifica=1
		end if
	next

	if verifica = 0 then
				tab_1.tabpage_3.dw_3.update()
				commit using sqlca;
				MessageBox("Productos", "Grabación exitosa")
				fsub_verifica_visita()
	else
				MessageBox("Error", "Cant real > Cant plan; ó igual 0")

	end if
else
	MessageBox("Error", "No existe Datos")
end if
```

Evento itemchanged del control datawindow dw_2 del registro de visita

Descripción: Este evento se dispara cuando el control dropdown listbox de la columna cab_bol_bandera se ha cambiado, si el valor seleccionado es N (no visita), entonces se activan o los controles de origen y causa de no visita y el botón grabar; si el valor seleccionado es S (si visita), entonces se desactivan o los controles de origen y causa de no visita y se activa el botón grabar, si el valor seleccionado es Null o vacío (generada), entonces se desactivan todos los controles. Los principales controles de esta ventana son:

dw_1: datawindow de datos de boleta (numero de boleta, cliente, lugar de visita)
dw_2: datawindow del registro de visita
dw_3: datawindow del registro de productos
cb_1: botón Grabar de registro de productos

cb_2: botón Grabar de registro de visita

Código:

```
string fecha, origen, causas
tab_1.tabpage_2.dw_2.accepttext( ) //acepto todos los cambios del registro de visita
//asigno a variable bandera el código de la boleta
bandera = tab_1.tabpage_2.dw_2.getitemstring(tab_1.tabpage_2.dw_2.getrow( ),"cab_bol_bandera")

if bandera = "N" then //si no se visita
	tab_1.tabpage_2.dw_2.object.cab_bol_fecha[tab_1.tabpage_2.dw_2.getrow()] = today()
	tab_1.tabpage_2.dw_2.Object.cab_bol_fecha.TabSequence = 20
	tab_1.tabpage_2.dw_2.Object.cab_bol_observaciones.TabSequence = 30
	tab_1.tabpage_2.ddlb_causas.enabled = true
	tab_1.tabpage_2.ddlb_origen.enabled = true
	tab_1.tabpage_3.dw_3.enabled = false
	tab_1.tabpage_3.cb_1.enabled = false

	fecha = string(tab_1.tabpage_2.dw_2.object.cab_bol_fecha[tab_1.tabpage_2.dw_2.getrow()])

	if isnull(origen) or isnull(causas) or isnull(fecha) then
		tab_1.tabpage_2.cb_2.enabled = false
	else
		tab_1.tabpage_2.cb_2.enabled = true
	end if
	//falta el boton grabar
elseif trim(bandera) = "" or isnull(bandera) then //si boleta es generada
	tab_1.tabpage_2.ddlb_causas.enabled = false
	tab_1.tabpage_2.ddlb_origen.enabled = false
	tab_1.tabpage_2.dw_2.Object.cab_bol_fecha.TabSequence = 0
	tab_1.tabpage_2.dw_2.Object.cab_bol_observaciones.TabSequence = 0
	tab_1.tabpage_2.dw_2.Object.cab_bol_fecha[tab_1.tabpage_2.dw_2.getrow()] = today()
	tab_1.tabpage_3.dw_3.enabled = false
	tab_1.tabpage_3.cb_1.enabled = false
	tab_1.tabpage_2.cb_2.enabled  = false
elseif bandera = "S" then //si se visita
	tab_1.tabpage_2.dw_2.object.cab_bol_fecha[tab_1.tabpage_2.dw_2.getrow()] = today()
	tab_1.tabpage_2.dw_2.Object.cab_bol_fecha.TabSequence = 20
	tab_1.tabpage_2.dw_2.Object.cab_bol_observaciones.TabSequence = 30
	tab_1.tabpage_2.ddlb_causas.text = ""
	tab_1.tabpage_2.ddlb_origen.text = ""
	tab_1.tabpage_2.ddlb_causas.enabled = false
	tab_1.tabpage_2.ddlb_origen.enabled = false
	tab_1.tabpage_3.dw_3.enabled = true
	tab_1.tabpage_3.cb_1.enabled = true

	if isnull(fecha) then
		tab_1.tabpage_2.cb_2.enabled  = false
	else
		tab_1.tabpage_2.cb_2.enabled = true
	end if

end if
```

6. Objeto w_mantenimiento_clientes

Descripción: El evento Open de esta ventana permite utilizar un objeto de transacción, en este caso sqlca el cual suministra la información necesaria para comunicarse con la base de datos. Luego, realiza el retrieve de los datos tomando como base el parámetro cliente.
Los principales elementos que conforman esta ventana son:
dw_1: datawindow de maestro de clientes
dw_2: datawindow de los datos personales del cliente
dw_3: datawindow de los datos del lugar de visita del cliente
cb_1: botón Grabar

Código:

```
dw_1.settransobject(sqlca)
tab_1.tabpage_1.dw_2.settransobject(sqlca)
tab_1.tabpage_2.dw_3.settransobject(sqlca)
tab_1.tabpage_1.text = "Clientes"
tab_1.tabpage_2.text = "Lugar Trabajo"

if dw_1.retrieve() > 0 then //si existen datos
        //almaceno el código de cliente activo en ese momento en variable cliente
        cliente = dw_1.getitemstring( dw_1.getrow(),"cli_codigo")
        tab_1.tabpage_1.dw_2.retrieve(cliente) //hago retrieve de datos utilizando el código de cliente.
end if
```

Esta ventana contiene 3 funciones principales: ue_actualiza_cli(), ue_buscar(), ue_elimina_cli()

Función ue_actualiza_cli()

Descripción: Realiza un retrieve de datos

Código: dw_1.retrieve( )

Función ue_buscar(string cli, string apellido)

Descripción: Permite realizar la búsqueda de un cliente ya sea por cédula de identidad o por apellido, para lo cual utiliza los parámetros enviados a la función cli y apellido, si estos son diferentes de vacío hace una consulta a la base de datos para encontrar el cliente, caso contrario emite un mensaje de error.

Código:

```
int codigo_encontrado
string ruc_encontrado

close(w_buscar) //cierro ventana buscar
if dw_1.rowcount( ) > 0 then //si maestro de clientes tiene registros

        if trim(cli) <> "" and trim(apellido) = "" then          //busca por ruc
                SELECT count(cvm_cliente.cli_codigo)  //busca en la base de datos
                 INTO :codigo_encontrado
                 FROM cvm_cliente
                WHERE cvm_cliente.cli_codigo = :cli;

                if codigo_encontrado = 1 then //si encuentra ubica el cliente
                        dw_1.SetFilter("cli_codigo = '"+ cli +" '")
                        dw_1.filter()
                        tab_1.tabpage_1.dw_2.retrieve(cli)
                        tab_1.tabpage_2.dw_3.retrieve(cli)
                else
                        MessageBox("Error", "Código no existe")
                end if
        end if

        if trim(cli) = "" and trim(apellido) <> "" then //busca por apellido

                        SELECT cvm_cliente.cli_codigo
                         INTO :ruc_encontrado
                         FROM cvm_cliente
                        WHERE ( cvm_cliente.cli_apellido1= :apellido );

                        /*SELECT cvm_cliente.cli_codigo
                         INTO :ruc_encontrado
                         FROM cvm_cliente
```

```
                        WHERE ( cvm_cliente.cli_apellido1 like "%' + :apellido + '%" + );
                        */

                if trim(ruc_encontrado) <> "" then //si ruc no es vacío

                        dw_1.SetFilter("cli_codigo = "'+ ruc_encontrado + " "")
                        dw_1.filter()
                        tab_1.tabpage_1.dw_2.retrieve(ruc_encontrado)
                        tab_1.tabpage_2.dw_3.retrieve(ruc_encontrado)
                else
                        MessageBox("Error", "Código no existe")
                end if
        end if

        if trim(cli) <> "" and trim(apellido) <> "" then //busca por ruc y apellido

                SELECT cvm_cliente.cli_codigo
                 INTO :ruc_encontrado
                 FROM cvm_cliente
                WHERE ( cvm_cliente.cli_codigo = :cli ) AND
                                ( cvm_cliente.cli_apellido1 = :apellido )   ;

                if trim(ruc_encontrado) <> "" then //busca por ruc
                        dw_1.SetFilter("cli_codigo = "'+ ruc_encontrado +" "")
                        dw_1.filter()
                        tab_1.tabpage_1.dw_2.retrieve(ruc_encontrado)
                        tab_1.tabpage_2.dw_3.retrieve(ruc_encontrado)
                else
                        MessageBox("Error", "Código no existe")
                end if
        end if
else
        MessageBox("Error", "No existe registros")
end if
```

Función ue_eliminar_cli ()

Descripción: Permite eliminar un cliente que no se encuentre asignado a una boleta, si este se encuentra asignado entonces emite mensaje de error, caso contrario, visualiza cuadro de diálogo "Está seguro de borrar el registro", si selecciona Si el registro será eliminado permanentemente de las tablas "cvm_cliente", "cvm_lug_vis_cli", "cvm_lugar_visita", "cvm_especialidad_cliente", "cvm_orden_ruta", "cvm_cab_pedidos", "cvm_det_pedidos"; caso contrario el registro no se eliminará.

Código:

```
int b, cabpedi, contador
string CodCliente, BolNumero, LugVisitaCli

if dw_1.rowcount( ) > 0 then //si existen registros
//asigno el código de cliente actual a la variable codcliente
CodCliente= dw_1.getitemstring(dw_1.getrow(),"cli_codigo")

//verifico si cliente se encuentra en boletas
  SELECT cvm_cab_boleta.cab_bol_numero
    INTO :BolNumero
    FROM cvm_cab_boleta
   WHERE cvm_cab_boleta.cli_codigo = :CodCliente
         ;

if trim(BolNumero) <> "" then //si cliente se encuentra en boletas
        messagebox("No se puede eliminar Clientes","Cliente asignado a una boleta..!",stopsign!)
        return
else //si no está en boletas
        b=Messagebox("Advertencia","Está seguro de borrar el registro",Exclamation!,YesNo!)
        if (b=1) then

//consulto el lugar de visita
```

```
   SELECT cvm_orden_ruta.lug_vis_codigo
    INTO :LugVisitaCli
    FROM cvm_orden_ruta
   WHERE cvm_orden_ruta.cli_codigo = :CodCliente
         ;

//consulto el lugviscli
SELECT count(cvm_lug_vis_cli.lug_vis_codigo)
    INTO :contador
    FROM cvm_lug_vis_cli
   WHERE cvm_lug_vis_cli.lug_vis_codigo = :LugVisitaCli
         ;

//elimino de tabla de juntura lugar_cliente
DELETE FROM cvm_lug_vis_cli
   WHERE cvm_lug_vis_cli.cli_codigo = :CodCliente   ;

//elimino de la ruta
  DELETE FROM cvm_orden_ruta
   WHERE cvm_orden_ruta.cli_codigo = :CodCliente
         ;

//si es el único lugar de visita asignado a ese cliente lo borro
if contador = 1 then
DELETE FROM cvm_lugar_visita
   WHERE cvm_lugar_visita.lug_vis_codigo = :LugVisitaCli
                                          ;
end if

//elimino especialidad_cliente
DELETE FROM r019
   WHERE r019.cli_codigo = :CodCliente
         ;

//elimino pedidos detalle para ese cliente
DELETE FROM cvm_det_pedidos
            where cvm_det_pedidos.ped_det_codigo in (select cvm_det_pedidos.ped_det_codigo
                                                            from   cvm_det_pedidos, cvm_cab_pedidos
                                                  WHERE                (cvm_det_pedidos.ped_cab_codigo                =
cvm_cab_pedidos.ped_cab_codigo)
                                                    and (cvm_cab_pedidos.ped_cli_codigo = :CodCliente)  )
                                          ;
//elimino pedidos cabecera para ese cliente
DELETE FROM cvm_cab_pedidos
   WHERE cvm_cab_pedidos.ped_cli_codigo = :CodCliente
         ;

//elimino cliente
DELETE FROM cvm_cliente
   WHERE cvm_cliente.cli_codigo = :CodCliente
         ;

            commit using sqlca;

            dw_1.retrieve( )
            tab_1.tabpage_1.dw_2.retrieve(cliente)
            tab_1.tabpage_2.dw_3.retrieve(cliente)
            MessageBox("Clientes", "Grabación exitosa")

            end if
end if
else
            MessageBox("Error", "No existe registros")
end if
```

Evento selectionchanged del tab_1 de mantenimiento de clientes

Descripción: Verifica el page que se encuentra activo, de acuerdo a esto activa o desactiva controles. Y almacena en variable Cliente el cliente activo en ese momento.

Código:

```
if tab_1.selectedtab = 1 then //page activo es datos de cliente
	m_mant_cli.m_clientes.m_nuevo.enabled = true
elseif tab_1.selectedtab = 2 then //page activo es lugar de visita
	if dw_1.rowcount( ) <> 0 then //si existen clients
		//asigno a variable cliente el cliente activo
		Cliente = dw_1.getitemstring(dw_1.getrow( ),"cli_codigo")
		tab_1.tabpage_2.dw_3.settransobject(sqlca)
		if tab_1.tabpage_2.dw_3.retrieve(Cliente ) <> 0 then
			horario_ant = tab_1.tabpage_2.dw_3.getitemstring(tab_1.tabpage_2.dw_3.getrow(),
"cvm_horario_atencion_hor_ate_codigo")
		end if
		m_mant_cli.m_clientes.m_nuevo.enabled = false
	end if
end if
```

Evento clicked del botón cb_1 de la ventana w_mantenimiento_clientes

Descripción: Permite grabar los cambios realizados a los datawindows de datos de cliente y datos de lugares de visita dependiendo de cuál se encuentre activo en ese momento. En ambos casos hace uso de la función interna Update(), si esta es igual a 1 emite mensaje de "Grabación Exitosa", caso contrario emite mensaje de error.

Código:

```
string lugar, callep, numero, calles, lugtelf
string clinom, cliapell, clitelf
if dw_1.rowcount( ) <> 0 then //si existen clientes

	if tab_1.selectedtab = 1 then //si estoy en datos de cliente
		tab_1.tabpage_1.dw_2.accepttext( ) //acepta los cambios realizados
		//almacena en variable el apellido
		cliapell= tab_1.tabpage_1.dw_2.getitemstring(tab_1.tabpage_1.dw_2.getrow(),"cli_apellido1")
		//almacena en variable el nombre
		clinom = tab_1.tabpage_1.dw_2.getitemstring(tab_1.tabpage_1.dw_2.getrow(),"cli_nombre1")
		//almacena en variable el teléfono
		clitelf = tab_1.tabpage_1.dw_2.getitemstring(tab_1.tabpage_1.dw_2.getrow(),"cli_telefono_personal")

		if trim(cliapell) <> "" and trim(clinom) <> "" and trim(clitelf) <> "" then
			if len(trim(clitelf)) = 9 then
				if tab_1.tabpage_1.dw_2.update( ) = 1 and actualiza_especialidad () = 1 then
					commit using sqlca;
					dw_1.retrieve()
					cliente = dw_1.getitemstring( dw_1.getrow(),"cli_codigo")
					tab_1.tabpage_1.dw_2.retrieve(cliente)
					MessageBox("Clientes", "Grabación exitosa")
				else
					MessageBox("Error al grabar ", sqlca.sqlerrtext)
					rollback using sqlca;
				end if
			else
				MessageBox("Error", "Ingrese 9 dígitos en Teléfono")

			end if
		else
			MessageBox("Error", "Datos incompletos")
		end if

	elseif tab_1.selectedtab = 2 then //si estoy en lugar de visita

		tab_1.tabpage_2.dw_3.accepttext( ) //acepto los cambios
		lugar=tab_1.tabpage_2.dw_3.getitemstring(tab_1.tabpage_2.dw_3.getrow(
				),"cvm_lugar_visita_lug_vis_nombre")
```

```
                callep          =          tab_1.tabpage_2.dw_3.getitemstring(tab_1.tabpage_2.dw_3.getrow(
),"cvm_lugar_visita_lug_vis_calle_p")
                numero = tab_1.tabpage_2.dw_3.getitemstring(tab_1.tabpage_2.dw_3.getrow( ),"lug_vis_numero")
                calles          =          tab_1.tabpage_2.dw_3.getitemstring(tab_1.tabpage_2.dw_3.getrow(
),"cvm_lugar_visita_lug_vis_calle_s")
                lugtelf                    =tab_1.tabpage_2.dw_3.getitemstring(tab_1.tabpage_2.dw_3.getrow(
),"cvm_lugar_visita_lug_vis_telefono")

                if trim(lugar) <> "" and trim(numero) <> "" and trim(calles) <> "" and trim(lugtelf) <> "" then
                        if len(trim(lugtelf)) = 9 then
                                if tab_1.tabpage_2.dw_3.update( ) = 1 and actualiza_horario( ) = 1 then
                                        commit using sqlca;
                                        dw_1.retrieve()
                                        cliente = dw_1.getitemstring( dw_1.getrow(),"cli_codigo")
                                        tab_1.tabpage_2.dw_3.retrieve(cliente)
                                        MessageBox("Lugares Visita", "Grabación exitosa")
                                elseIF SQLCA.SQLCode = -1 THEN
                                        MessageBox("SQL error", SQLCA.SQLErrText)
                                        rollback using sqlca;
                                end if
                        else
                                MessageBox ("Error","Ingrese 9 dígitos en Teléfono")

                        end if
                else
                        MessageBox ("Error", "Datos incompletos!")
                end if
        end if
else
        MessageBox ("Error", "No existen registros")
end if
```

Menú Opciones de ventana w_mantenimiento_clientes

Descripción: El menú Opciones contiene 3 submenús descritos a continuación:
Nuevo: Llama al evento Open de la ventana w_nuevo_cli
Eliminar: Llama a la función ue_eliminar() de la ventana w_mantenimiento_clientes descrita anteriormente.
Buscar: Llama al evento Open de la ventana w_buscar.

Evento Open de la ventana w_nuevo_cli

Descripción: Inserta nuevas filas a los datawindows de cliente, especialidad, lugar_visita, orden_ruta y llena los dropdown listbox de tipo_cliente, especialidad, lugar_visita, horario y ruta con datos necesarios para seleccionarlos.

Código:

```
long NuevaFila
string item,cod

dw_1.settransobject(sqlca)
dw_3.settransobject(sqlca)
dw_4.settransobject(sqlca)
dw_5.settransobject(sqlca)

NuevaFila = dw_1.InsertRow(0)   //inserta nuevas filas a los datawindows de cliente, especialidad, lugar_visita, orden_ruta
NF_R019 = dw_3.InsertRow(0)
NF_lug_vis_cli = dw_4.InsertRow(0)
NF_orden_ruta = dw_5.InsertRow(0)

ddlb_tipo_cli.reset( )  //limpia los dropdown listbox de tipo_cliente, especialidad, lugar_visita, horario y ruta
ddlb_especialidad.reset()
ddlb_lugar.reset( )
```

```
ddlb_horario.reset( )
ddlb_ruta.reset( )


// Retrieve tipo_cliente
DECLARE cust_cur CURSOR FOR
SELECT tip_cli_codigo, tip_cli_descripcion FROM cvm_tipo_cliente;
OPEN cust_cur;
DO WHILE sqlca.sqlcode = 0
	FETCH cust_cur INTO :cod, :item;
	if SQLCA.sqlcode=0 then
		ddlb_tipo_cli.AddItem(item)
	end if
LOOP
CLOSE cust_cur;

// Retrieve especialidad
DECLARE prod_cur CURSOR FOR
SELECT esp_codigo, esp_descripcion FROM cvm_especialidad;
OPEN prod_cur;
DO WHILE sqlca.sqlcode = 0
	FETCH prod_cur INTO :cod, :item;
	if SQLCA.sqlcode=0 then
		ddlb_especialidad.AddItem(item)
	end if
LOOP
CLOSE prod_cur;

// Retrieve lugar_visita
DECLARE lug_cur CURSOR FOR
SELECT lug_vis_codigo, lug_vis_nombre FROM cvm_lugar_visita;
OPEN lug_cur;
DO WHILE sqlca.sqlcode = 0
	FETCH lug_cur INTO :cod, :item;
	if SQLCA.sqlcode=0 then
		ddlb_lugar.AddItem(item)
	end if
LOOP
CLOSE lug_cur;

// Retrieve horario
DECLARE hor_cur CURSOR FOR
SELECT hor_ate_codigo, rtrim(hor_ate_dia) + ' ' + hor_ate_hinicio + ' ' + hor_ate_hfin as horario FROM cvm_horario_atencion;
OPEN hor_cur;
DO WHILE sqlca.sqlcode = 0
	FETCH hor_cur INTO :cod, :item;
	if SQLCA.sqlcode=0 then
		ddlb_horario.AddItem(item)
	end if
LOOP
CLOSE hor_cur;

// Retrieve ruta
DECLARE rut_cur CURSOR FOR
SELECT cvm_ruta.rut_codigo, rut_descripcion FROM cvm_ruta, cvm_cab_boleta
WHERE cvm_ruta.rut_codigo = cvm_cab_boleta.rut_codigo;
OPEN rut_cur;
DO WHILE sqlca.sqlcode = 0
	FETCH rut_cur INTO :cod, :item;
	if SQLCA.sqlcode=0 then
		ddlb_ruta.AddItem(item)
	end if
LOOP
CLOSE rut_cur;
```

Evento clicked del botón cb_1 de la ventana w_mantenimiento_clientes

Descripción: Inserta los nuevos valores en los datawindows y hace uso de la función interna Update() para poder actualizar los nuevos valores. Si update() es igual a 1 la actualización fue exitosa, caso contrario se emite mensaje de error.

Código:

```
string telf_per
dw_1.accepttext( ) //datos clientes
//asigno código de cliente en variable código
codigo = dw_1.getitemstring(dw_1.getrow(),"cli_codigo")
telf_per = string(dw_1.getitemstring(dw_1.getrow(),"cli_telefono_personal"))

if isnumber(codigo) and len(codigo) = 10 then  //verifico que cédula tenga 10 dígitos y sean números
        if len(telf_per) = 9 then

                apellido1 = dw_1.getitemstring(dw_1.getrow(),"cli_apellido1")
                nombre1 = dw_1.getitemstring(dw_1.getrow(),"cli_nombre1")

                //busco los código de los dropdown listbox en la base de datos
                //tip_cli
                SELECT tip_cli_codigo INTO :tip_cli FROM cvm_tipo_cliente
                WHERE tip_cli_descripcion = :ddlb_tipo_cli.text;
                sexo = ddlb_sexo.text

                //especialidad
                SELECT esp_codigo INTO :especialidad FROM cvm_especialidad
                WHERE esp_descripcion = :ddlb_especialidad.text;

                //lug_vis
                SELECT lug_vis_codigo INTO :lug_vis FROM cvm_lugar_visita
                WHERE lug_vis_nombre = :ddlb_lugar.text;

                //horario
                SELECT hor_ate_codigo INTO :horario FROM cvm_horario_atencion
                where (RTRIM(hor_ate_dia) + ' ' + hor_ate_hinicio + ' ' + hor_ate_hfin) = :ddlb_horario.text;

                //ruta
                SELECT rut_codigo INTO :ruta FROM cvm_ruta
                WHERE rut_descripcion = :ddlb_ruta.text;

                if len(trim(codigo)) <> 0 AND len(trim(apellido1)) <> 0 AND len(trim(nombre1)) <>0 AND len(trim(tip_cli)) <> 0 AND len(trim(sexo)) <> 0 AND len(trim(especialidad)) <> 0 AND len(trim(lug_vis))<> 0 AND len(trim(horario)) <> 0 AND len(trim(ruta)) <> 0 then

                        //insercion en cliente
                        dw_1.object.tip_cli_codigo[dw_1.getrow()]=tip_cli
                        dw_1.object.cli_sexo[dw_1.getrow()]=sexo
                        dw_1.object.pot_codigo[dw_1.getrow()]="0001"

                        //insercion en lug_vis_cli
                        dw_4.object.cli_codigo[dw_4.getrow()]=codigo
                        dw_4.object.lug_vis_codigo[dw_4.getrow()]=lug_vis
                        dw_4.object.hor_ate_codigo[dw_4.getrow()]=horario
                        dw_4.accepttext( )

                        //insercion en Tabla R019  Cliente_especialidad
                        dw_3.object.esp_codigo[dw_1.getrow()]=especialidad
                        dw_3.object.cli_codigo[dw_1.getrow()]=codigo
                        dw_3.accepttext( )

                        //Insercion en la tabla orden ruta
                        dw_5.object.cli_codigo[dw_5.getrow()]=codigo
                        dw_5.object.lug_vis_codigo[dw_5.getrow()]=lug_vis
                        dw_5.object.rut_codigo[dw_5.getrow()]=ruta
                        dw_5.object.ord_rut_orden[dw_5.getrow()]=1
                        dw_5.object.esp_codigo[dw_5.getrow()]=especialidad
                        dw_5.accepttext( )

                        //Manejo de transaccion
                        if dw_1.update(true,false) = 1 then
```

```
                                        if dw_3.update(true,false)=1 then
                                                if dw_4.update(true,false)=1 then
                                                        if dw_5.update()=1 then
                                                                commit using sqlca;
                                                                dw_1.ResetUpdate()
                                                                dw_3.ResetUpdate()
                                                                dw_4.ResetUpdate()
                                                                dw_5.ResetUpdate()
                                                                Messagebox("Nuevo          cliente","Grabación
exitosa")
                                                                close(parent)
                                                         else
                                                                error_nuevo = true
                                                                Messagebox("Error al Grabar","Ruta")
                                                                rollback using sqlca;
                                                          end if
                                                else
                                                                error_nuevo = true
                                                                Messagebox("Error al Grabar","Lugar Visita")
                                                                rollback using sqlca;
                                                end if
                                        else
                                                        error_nuevo = true
                                                        Messagebox("Error al Grabar","Especialidad")
                                                        rollback using sqlca;
                                        end if
                                else
                                                error_nuevo = true
                                                Messagebox("Error al Grabar","Cliente")
                                                rollback using sqlca;
                                        end if
                        else
                                error_nuevo = true
                                MessageBox("Error", "Datos incompletos.!")
                        end if
                else
                        error_nuevo = true
                        MessageBox("Error", "Ingrese 9 dígitos para Teléfono.!")
                end if
else
                error_nuevo = true
                MessageBox("Error", "Código incorrecto.!")
end if
```

Función Insert_client(string codcli, string apellido, string nombre, string tipo, string sex)

Descripción: Recibe como parámetros al código de cliente, apellido, nombre, tipo, sex y los almacena en la base de datos a través de una función SQL Insert. Si sqlca.sqlcode es igual a 0 significa que la inserción fue exitosa entonces realiza un commit a la transacción, caso contrario realiza un rollback a la transacción.

Código:

```
if len(CodCli) <> 10 then //si no son 10 dígitos en la cédula
        MessageBox("Error", "Cédula incorrecta")
        return -1
end if
matricula = dw_1.getitemstring(dw_1.getrow(),"cli_matricula")
ruc = dw_1.getitemstring(dw_1.getrow(),"cli_ruc")
direccion = dw_1.getitemstring(dw_1.getrow(),"cli_direccion_domicilio")
telefono = dw_1.getitemstring(dw_1.getrow(),"cli_telefono_personal")
potencialidad = dw_1.getitemstring(dw_1.getrow(),"pot_codigo")

  INSERT INTO "cvm_cliente"
      ( "cli_codigo",
        "tip_cli_codigo",
        "cli_apellido1",
        "cli_nombre1",
```

```
        "cli_matricula",
        "cli_sexo",
        "cli_ruc",
        "cli_direccion_domicilio",
        "cli_telefono_personal",
        "cli_email",
        "pot_codigo" )
  VALUES ( :CodCli, :Tipo, :Apellido, :Nombre, :matricula, :Sex, :ruc,:direccion, :telefono, :email,  '0001' )  using sqlca;

if sqlca.sqlcode = 0 then //no existen errores
        commit using sqlca;
        return 1
else
        MessageBox("SQL error", SQLCA.SQLErrText)
        rollback using sqlca;
        return 0
end if
```

Evento Open de la ventana w_buscar

Descripción: Permite utilizar la función ue_buscar que recibe como parámetro el ruc del cliente y el apellido almacenados en los controles sle_1 y sle_2. La función ue_buscar se la describió anteriormente.

Código:

```
w_mantenimiento_clientes.event ue_buscar(trim(sle_1.text),trim(sle_2.text))
```

7. Objeto w_pedidos_cliente

Descripción: El evento Open de esta ventana permite utilizar un objeto de transacción, en este caso sqlca el cual suministra la información necesaria para comunicarse con la base de datos. Luego, realiza el retrieve de los datos tomando como base el parámetro g_rep_codigo (que es el código del representante de ventas).
Contiene un menú con las siguientes opciones:
ue_insertar: w_pedidos_cliente.event ue_insertar()
ue_eliminar: w_pedidos_cliente.event ue_eliminar()
ue_grabar: w_pedidos_cliente.event ue_grabar()
cb_1: Inserta productos
cb_2: Elimina productos
cb_3: Guarda productos

Código:

```
dw_1.settransobject(sqlca)
dw_2.settransobject(sqlca)
dw_3.settransobject(sqlca)

if dw_1.retrieve() > 0 then //si existen clientes
        //almaceno código de cliente en variable cliente
        cliente = dw_1.getitemstring(dw_1.getrow(),"cli_codigo")
        insertar = false
        insertar_prd = false

        if dw_2.retrieve(cliente,g_rep_codigo) > 0  then //si cabecera de pedido contiene registros
                dw_2.accepttext( ) //acepto cambios
                this.cb_1.enabled = true
                this.cb_2.enabled = true
                this.cb_3.enabled = true
                cabpedido = dw_2.getitemnumber(dw_2.getrow(), "ped_cab_codigo")
                dw_3.retrieve(cabpedido)
                int cuenta
                cuenta = dw_3.rowcount( )
```

```
                return 0
        else
                this.cb_1.enabled = false
                this.cb_2.enabled = false
                this.cb_3.enabled = false
                dw_3.Reset()
        end if
end if
```

Evento ue_insertar de la ventana w_pedidos_cliente

Descripción: Permite ingresar un nuevo registro en la tabla cvm_cab_pedidos, que es la cabecera de los pedidos. Deshabilita controles como insertar, eliminar y guardar productos.

Código:

```
//selecciona el mayor valor de código de la tabla cvm_cab_pedidos
  SELECT max("cvm_cab_pedidos"."ped_cab_codigo"), count("cvm_cab_pedidos"."ped_cab_codigo")
    INTO :cab, :num
    FROM "cvm_cab_pedidos";
long ll_nuevafila
if isnull(cab) then
        cab =0
end if

cab = cab+100 + integer(g_rep_codigo) //aumenta código en 1
dw_1.acceptText() //acepta cambios
dw_2.reset()
fila_ant = dw_2.getrow( )
ll_nuevafila=dw_2.insertrow(0) //inserta una nueva fila en datawindow de pedidos
dw_2.setfocus()
dw_2.object.ped_fecha[ll_nuevafila] = today() //asigna fecha actual al pedido
dw_2.object.ped_cab_codigo[ll_nuevafila] = cab
dw_2.object.ped_numero[ll_nuevafila] = num + 1
insertar = true
cb_1.enabled = false
cb_2.enabled = false
cb_3.enabled = false
m_pedidos.m_opciones.m_insertar.enabled = false
```

Evento ue_eliminar de la ventana w_pedidos_cliente

Descripción: Permite eliminar un registro en la tabla cvm_cab_pedidos, que es la cabecera de los pedidos, verifica primeramente si ese pedido tiene asignado productos si es así, emite mensaje que indica que se debe eliminar todos los productos para continuar, si la cabecera de pedido está sin productos, emite un cuadro de diálogo preguntando si está seguro eliminar el registro, si es así borra permanentemente ese registro.

Código:

```
int b
string str
int cabpedi
str=dw_3.object.datawindow.firstrowonpage //verifica si existen productos
if str <> "0" then
        messagebox("Pedidos","Favor elimine productos..!",stopsign!)
        return
else //si no hay productos
        b=Messagebox("Advertencia","Está seguro de borrar el registro",Exclamation!,YesNo!)
        if (b=1) then

                cabpedi = dw_2.getitemnumber(dw_2.getrow(), "ped_cab_codigo")
               dw_2.deleterow(dw_2.getrow())

               //elimina permanentemente el pedido de la tabla cm_cab_pedidos
               DELETE FROM cvm_cab_pedidos
```

```
                    WHERE cvm_cab_pedidos.ped_cab_codigo = :cabpedi;

                    this.cb_1.enabled = false
                    this.cb_2.enabled = false
                    this.cb_3.enabled = false
                insertar = false

                if dw_2.rowcount( ) <> 0 then
                        cabpedi = dw_2.getitemnumber(dw_2.getrow(), "ped_cab_codigo")
                        dw_3.retrieve(cabpedi)
                end if
        end if
end if
```

Evento ue_grabar de la ventana w_pedidos_cliente

Descripción: Permite grabar los cambios realizados en la cabecera de pedido, verifica que un pedido posea al menos un producto si no es así, se emite un mensaje de error. Se considera al contacto y la fecha y productos como datos obligatorios. Se considera un pedido duplicado cuando se ingresa un pedido para el mismo cliente en la misma fecha.

Código:

```
string contacto, CodCliente,cabecera
date fecha
int cabped, faltadx
faltadx = 0
dw_1.acceptText() //acepta cambios
dw_2.acceptText()
cabecera = ""

if dw_1.rowcount( ) <> 0 then //si existe clientes
        CodCliente = dw_1.object.cli_codigo[dw_1.getrow()]
        dw_2.object.ped_cli_codigo[dw_2.getrow( )]= cliente //asigna variables a datawindow
        dw_2.object.ped_cod_rep[dw_2.getrow( )] = g_rep_codigo
        fecha = dw_2.getitemdate(dw_2.getrow(),"ped_fecha")
        contacto = dw_2.object.ped_contact_codigo[dw_2.getrow()]

        if isnull(contacto) or isnull(fecha) then //verifica que la fecha y contacto no sean vacíos
                MessageBox("Error", "Cabecera incompleta")
                faltadx = 1
        else
                if dw_3.rowcount() = 0 then //verifica productos
                                MessageBox("Error", "detalle incompleto")
                                faltadx = 1
                                cb_1.enabled = true
                                cb_2.enabled = true
                                cb_3.enabled = true
                else

                        //VERIFICO SI YA EXISTE UN PEDIDO PARA ESE CLIENTE EN ESA FECHA
                          SELECT "cvm_cab_pedidos"."ped_cli_codigo"
                                  INTO :cabecera
                                  FROM "cvm_cab_pedidos"
                                 WHERE ( "cvm_cab_pedidos"."ped_cli_codigo" = :CodCliente ) AND
                                                ( "cvm_cab_pedidos"."ped_fecha" = :fecha ) AND
                                                (  "cvm_cab_pedidos"."ped_cod_rep"  =  :g_rep_codigo  ) USING
SQLCA;

                        if cabecera = "" or insertar = false then
                                if dw_2.Update(true,false)=1  then
                                        if dw_3.Update()=1 then
                                                commit using sqlca;
                                                dw_2.ResetUpdate()
                                                dw_3.ResetUpdate()
                                                this.cb_1.enabled = true
                                                this.cb_2.enabled = true
                                                this.cb_3.enabled = true
                                                m_pedidos.m_opciones.m_insertar.enabled = true
```

```
                                        dw_1.enabled = true
                                        dw_2.enabled = true
                                    else
                                    Messagebox("Grabar","Error al grabar")
                                    rollback using sqlca;
                                    end if
                                else
                                    Messagebox("Grabar","Error al grabar")
                                    rollback using sqlca;
                                end if

                                else
                                    MessageBox("Error","Pedido Duplicado")
                                    if dw_2.retrieve(CodCliente,g_rep_codigo) > 0  then
                                        dw_2.accepttext( )
                                        cb_1.enabled = true
                                        cb_2.enabled = true
                                        cb_3.enabled = true
                                        dw_1.enabled = true
                                        dw_2.enabled = true
                                        m_pedidos.m_opciones.m_insertar.enabled = true
                                        m_pedidos.m_opciones.m_guardar.enabled = true
                                        m_pedidos.m_opciones.m_eliminar.enabled = true
                                        cabped = dw_2.getitemnumber(dw_2.getrow(), "ped_cab_codigo")
                                        dw_3.retrieve(cabped)
                                        insertar = false
                                    else
                                        m_pedidos.m_opciones.m_insertar.enabled = true
                                        m_pedidos.m_opciones.m_guardar.enabled = true
                                        m_pedidos.m_opciones.m_eliminar.enabled = true
                                        cb_1.enabled = false
                                        cb_2.enabled = false
                                        cb_3.enabled = false
                                        dw_3.Reset()
                                    end if
                                end if
                            end if
                        end if
                if insertar = true and faltadx = 0 then
                        dw_1.enabled = false
                        dw_2.enabled = false
                        m_pedidos.m_opciones.m_insertar.enabled = false
                        m_pedidos.m_opciones.m_guardar.enabled = false
                        insertar = false
                end if
else
        MessageBox("Error", "Cliente no existe")
end if
```

Evento clicked del botón cb_1 de la ventana w_pedidos_cliente

Descripción: Permite insertar los productos para el pedido e inserta una nueva fila en el datawindow de detalle de pedidos.

Código:

```
long ll_newrow,e
ll_newrow = dw_3.insertrow(1)
dw_3.setrow(ll_newrow)
dw_2.accepttext()
e=dw_2.getrow()

dw_3.object.cvm_det_pedidos_ped_cab_codigo[ll_newrow]=dw_2.object.ped_cab_codigo[e]
dw_3.setfocus()
insertar_prd = true
```

Evento clicked del botón cb_2 de la ventana w_pedidos_cliente

Descripción: Permite eliminar los productos para el pedido.

Código:

```
if dw_3.rowcount( ) = 1 then //verifica que exista registros
		dw_3.deleterow(dw_3.getrow()) //borra el registro
		dw_1.enabled = false
		dw_2.enabled = false
		cb_3.enabled = false
		m_pedidos.m_opciones.m_insertar.enabled = false
		m_pedidos.m_opciones.m_guardar.enabled = true
		m_pedidos.m_opciones.m_eliminar.enabled = true
	else
		dw_3.deleterow(dw_3.getrow()) //borra el registro
		dw_1.enabled = true
		dw_2.enabled = true
		m_pedidos.m_opciones.m_insertar.enabled = true
		m_pedidos.m_opciones.m_guardar.enabled = true
		m_pedidos.m_opciones.m_eliminar.enabled = true
	end if

	if dw_3.Update()=1 then //actualiza el datawindow
		commit using sqlca;
		insertar_prd = false
		dw_1.enabled = true
		dw_2.enabled = true
		m_pedidos.m_opciones.m_insertar.enabled = true
		m_pedidos.m_opciones.m_guardar.enabled = true
	else
		Messagebox("Grabar","Error al grabar")
		rollback using sqlca;
	end if
```

Evento clicked del botón cb_3 de la ventana w_pedidos_cliente

Descripción: Permite grabar los datos de los pedidos. Verifica que la cantidad del producto sea diferente de 0, si no es así emite mensaje de error.

Código:

```
string prd
int cant
prd = ""

dw_3.accepttext( ) //acepta cambios
dw_2.accepttext( )

if dw_3.rowcount( ) <> 0 then //existen datos
	prd = dw_3.object.cvm_det_pedidos_prd_codigo[dw_3.getrow()]
	cant = dw_3.object.cvm_det_pedidos_cantidad[dw_3.getrow()]
end if

if prd = "" or isnull(cant) then //verifica que cantidad haya sido ingresada
	MessageBox("Error", "Datos incompletos")
else
	if dw_2.Update(true,false)=1  then
		if dw_3.Update()=1 then
			commit using sqlca;
			dw_2.ResetUpdate()
			insertar_prd = false
			dw_1.enabled = true
			dw_2.enabled = true
			m_pedidos.m_opciones.m_insertar.enabled = true
			m_pedidos.m_opciones.m_guardar.enabled = true
			Messagebox("Grabar","Grabación exitosa")
		else
		Messagebox("Grabar","Error al grabar")
		rollback using sqlca;
	end if
	else
```

```
                Messagebox("Grabar","Error al grabar")
                rollback using sqlca;
        end if
end if
```

8. Objeto w_reportes

Descripción: El evento Open de esta ventana permite visualizar un reporte de cantidad de clientes a visitar. Esta ventana utiliza el datawindow d_rep_clientes. Contiene un menú de reportes descrito a continuación.

Notas de Pedido: Hace referencia al evento open de la ventan w_rep_notas_pedido
Registro de Visita: Hace referencia al evento open de la ventana w_estadisticas.

Código:

```
dw_1.settransobject(sqlca)
dw_1.retrieve()

st_total.text = string(dw_1.rowcount( )) //total de clientes
```

Evento Open de la ventana w_rep_notas_pedido

Descripción: El evento Open de esta ventana permite visualizar el reporte de notas de pedido por cada cliente. Esta ventana utiliza el datawindow d_rep_pedidos.

Código:

```
dw_1.settransobject(sqlca)
dw_1.retrieve()
```

Evento Open de la ventana w_estadisticas

Descripción: El evento Open de esta ventana permite visualizar el reporte gráfico del registro de las visitas, no visitas y boletas no registradas. Esta ventana utiliza el datawindow d_gr_visitas.

Código:

```
settransobject(sqlca)
long r1

r1= dw_1.retrieve(g_rep_codigo) //verifica que existan datos para ese representante

if r1 < 1 then
        dw_1.object.r_1.visible=false
        dw_1.object.r_2.visible=false
        dw_1.object.r_3.visible=false
end if.
```

8. Objeto w_visita_medica_ulsync_options

Descripción: Permite establecer los valores de configuración para el proceso de sincronización. Posee los siguientes eventos open(), close(), postopen().

Evento open() de la ventana w_visita_medica_ulsync_options

Código:

```
string   errCaption = "Usage Error"
string   badparm = "When invoking this window, use OpenWithParm and pass an instance" + &
            "of s_visita_medica_ulsync_parms."

if IsValid(Message.PowerObjectParm) then
            if Message.PowerObjectParm.TypeOf() = Structure! then
                        i_parms = Message.PowerObjectParm
            else
                        messagebox(errCaption, badparm)
                        cb_ok.enabled = FALSE
            end if
else
                        messagebox(errCaption, badparm)
                        cb_ok.enabled = FALSE
end if

i_uosync = CREATE nvo_visita_medica_ulsync
this.event POST  ue_postopen()
```

Evento close() de la ventana w_visita_medica_ulsync_options

Código:

```
wf_trysaving()
destroy i_uosync
```

Evento ue_postopen() de la ventana w_visita_medica_ulsync_options

Código:

```
string ls_regkey, ls_work
integer rc

tab_1.tabpage_subscribe.sle_pub.text = i_uosync.is_publication_name
tab_1.tabpage_subscribe.sle_version.text = i_uosync.is_version

if IsNull(i_parms.a_mluser) or i_parms.a_mluser = "" then
            ls_regkey =  "MLUserName"
            rc = RegistryGet(i_uosync.APP_REGPATH, ls_regkey, RegString!, ls_work)
            if rc = 1 then
                        tab_1.tabpage_subscribe.sle_mluser.text = ls_work
            end if
else
            tab_1.tabpage_subscribe.sle_mluser.text = i_parms.a_mluser
end if

ls_regkey =  "SavePW"
rc = RegistryGet(i_uosync.APP_REGPATH, ls_regkey, RegString!, ls_work)
if rc = 1 then
            if ls_work = "1" then
                        tab_1.tabpage_subscribe.cbx_savepw.checked = TRUE
            else
                        tab_1.tabpage_subscribe.cbx_savepw.checked = FALSE
            end if
end if

if IsNull(i_parms.a_mlpass) or i_parms.a_mlpass = ""  then
            if ls_work = "1" then
                        ls_regkey =  "MLPassword"
                        rc = RegistryGet(i_uosync.APP_REGPATH, ls_regkey, RegString!, ls_work)
                        if rc = 1 then
                                    i_uosync.uf_decrypt_pw(ls_work)
```

```
                        tab_1.tabpage_subscribe.sle_mlpass.text = ls_work
                end if
        end if
else
        tab_1.tabpage_subscribe.sle_mlpass.text = i_parms.a_mlpass
end if

ls_work = "Use these stream parameters to communicate with " + &
        "the MobiLink server via "
ls_work += i_uosync.is_stream + '.'
tab_1.tabpage_mlserver.st_mlserve.text = ls_work
tab_1.tabpage_mlserver.sle_host.text = i_uosync.is_host
tab_1.tabpage_mlserver.sle_port.text = i_uosync.is_port
tab_1.tabpage_mlserver.sle_stradd.text = i_uosync.is_stradd

if i_uosync.is_app_window = "YES" then
                tab_1.tabpage_runtime.cbx_appwin.checked = TRUE
else
                tab_1.tabpage_runtime.cbx_appwin.checked = FALSE

end if
if i_uosync.is_colnames = "YES" then
                tab_1.tabpage_runtime.cbx_colnames.checked = TRUE
else
                tab_1.tabpage_runtime.cbx_colnames.checked = FALSE

end if
tab_1.tabpage_runtime.sle_authparms.text = i_uosync.is_authparms

ib_SaveSub    = FALSE
ib_SaveRun    = FALSE
ib_SaveServer = FALSE
return TRUE
```

Evento clicked del botón cb_cancel de la ventana w_visita_medica_ulsync_options

Descripción: Retorna el valor de 100 indicando que el usuario a presionado CANCEL

Código:

```
ib_SaveSub    = FALSE
ib_SaveServer = FALSE
ib_SaveRun    = FALSE

i_parms.a_returncode = 100
closeWithReturn(parent, i_parms)
```

Evento clicked del botón cb_ok de la ventana w_visita_medica_ulsync_options

Descripción: Esta ventana pasa al servidor de sincronización el nombre de usuario y password en la estructura i_parms. También retorna 0 indicando que el usuario presionó OK.

Código:

```
i_parms.a_mluser = tab_1.tabpage_subscribe.sle_mluser.text
g_rep_codigo=trim(tab_1.tabpage_subscribe.sle_mluser.text)
i_parms.a_mlpass = tab_1.tabpage_subscribe.sle_mlpass.text
i_parms.a_returncode = 0
closeWithReturn(parent, i_parms)
```

9. Objeto w_visita_medica_ulsync

Descripción: Permite establecer realizar el proceso de sincronización. Posee los siguientes eventos open, ue_begin_download, ue_begin_sync, ue_begin_upload, ue_connecting, ue_disconnecting, ue_display_msg, ue_end_download, ue_end_sync, ue_end_upload,

ue_error_msg, ue_file_msg, ue_postopen, ue_progress_info, ue_receiven_upload_ack, ue_sroll_mle, ue_send_download_ack, ue_upload_ack, ue_wait_for_upload_ack, ue_warning_msg.

Evento open de la ventana w_visita_medica_ulsync

Código:

```
i_uosync = message.powerobjectparm
if IsValid(i_uosync) then
	this.event POST ue_postopen()
else
	cb_cancel.enabled = FALSE
	cb_OK.enabled = TRUE
	mle_status.text = 'Usage Error~r~n' + &
	'This window must be called using ~r~n' + &
	'OpenWithParm(w_ulconn_ULsync, g_uosync)~r~n'
end if
```

Evento ue_begin_download **de la ventana w_visita_medica_ulsync**

Código:

```
mle_status.SetRedraw(FALSE)
mle_status.text += 'Begin download.~r~n'
this.event TRIGGER ue_scroll_mle()
hpb_progress.position = 0
st_phasebar.text = 'Downloading...'

return 0
```

Evento ue_begin_sync **de la ventana w_visita_medica_ulsync**

Código:

```
mle_status.SetRedraw(FALSE)
mle_status.text += 'Begin synchronization  User: ' + &
	user_name + '~r~nPub Names: ' + pub_names + '~r~n'
this.event TRIGGER ue_scroll_mle()
return 0
```

Evento ue_begin_upload **de la ventana w_visita_medica_ulsync**

Código:

```
mle_status.SetRedraw(FALSE)
mle_status.text += 'Begin upload.~r~n'
this.event TRIGGER ue_scroll_mle()
hpb_progress.position = 0
st_phasebar.text = 'Uploading...'
return 0
```