BERRIKUNTZA TEKNOLOGIKOA
INNOVACIÓN EN LA TECNOLOGÍA

EUSKO JAURLARITZA GOBIERNO VASCO

Tknika

Actividad 9:

Cliente OPC Visual Basic.

1.- Listado de materiales:

- **PC con Tarjeta de red 3com o similar.**

- **1 PLC Omrom CJ1M – CPU11 – ETN**

Este autómata lleva integrada la tarjeta de comunicaciones ethernet que deberá estar previamente configurada.
Sería posible utilizar un autómata de la misma serie añadiéndole una tarjeta de comunicaciones ethernet.
Incluirá fuente de alimentación y unidades de entrada y salida.

- **Software Visual Basic v:6.0 o superior de Microsoft**

- **Software CX-Server OPC**

- **Software CX-Programmer ver 5.0**

- **Cable cruzado con conector RJ-45 para conexión Ethernet o bien Switch y cable paralelo.**

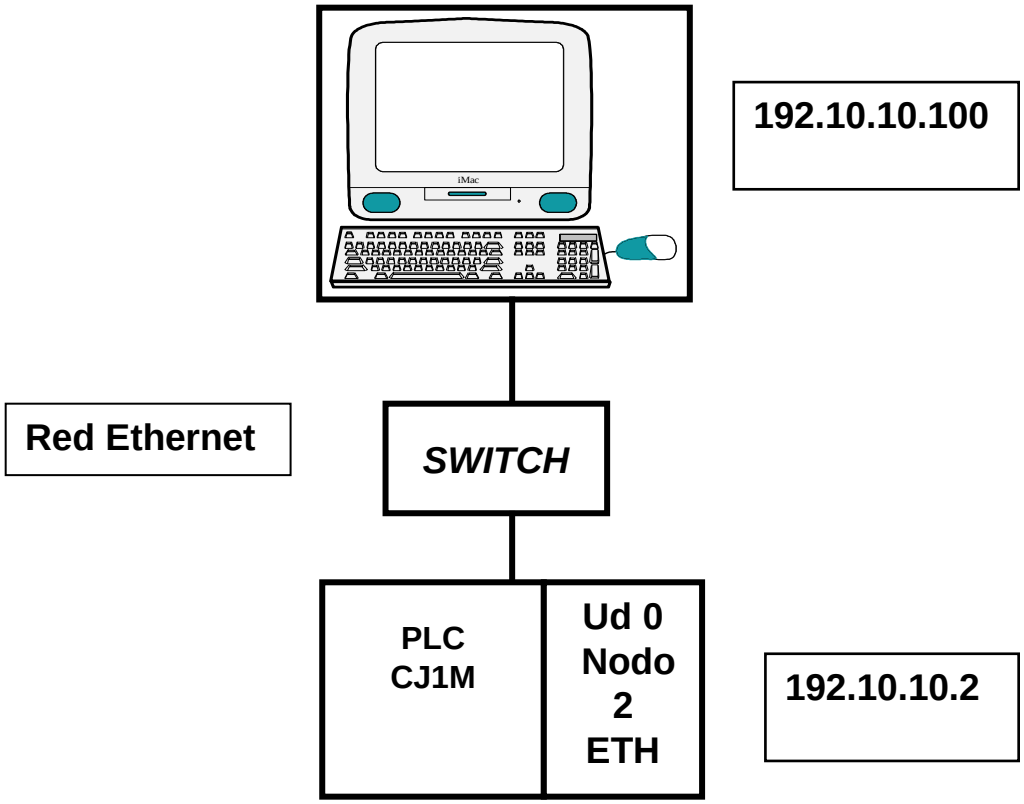
2.- Objetivos de la actividad.

Intercambiar datos entre el PLC CJ1M y Visual Basic utilizando el servidor de datos Cx-Server OPC de OMRON.

- **En este caso el programa VB se comporta como cliente OPC.**
- **Desarrollar una aplicación cliente (PARKING) utilizando los controles Active-X que proporciona Omron en la versión Demo de Cx_Server OPC.**
- **Desarrollar la misma aplicación en VB utilizando los comandos de programación que suministra el servidor OPC de Omron.**

NOTA: Esta actividad será desarrollada utilizando el servidor Cx-Server OPC. Es posible utilizar cualquier otro servidor OPC como KepServer cuya configuración puede seguirse en la actividad 7. Se definirán PLCs, Grupos e Items.
El control de comunicaciones "OMRON CX OPC Communication Control" se configurará escogiendo el servidor OPC deseado. Tanto los controles Active-X como los comandos de programación son igualmente válidos para cualquier servidor OPC. De la misma forma es posible utilizar más de un servidor OPC. En este caso será necesario insertar en la aplicación un control de comunicaciones por cada servidor usado. Al insertarlos se iran numerando (OPCComms1, OPCComms2 …). En la aplicación escogeremos el correspondiente para cada objeto.

3.- Desarrollo de la actividad con un servidor OPC. Esquema del equipo



Condiciones de funcionamiento del PLC.

► **Ejercicio de simulación de un Parking con un autómata programable con adquisición y control de datos del mismo, desde una aplicación realizada en Visual Basic.**

► **Condiciones de funcionamiento del parking:**

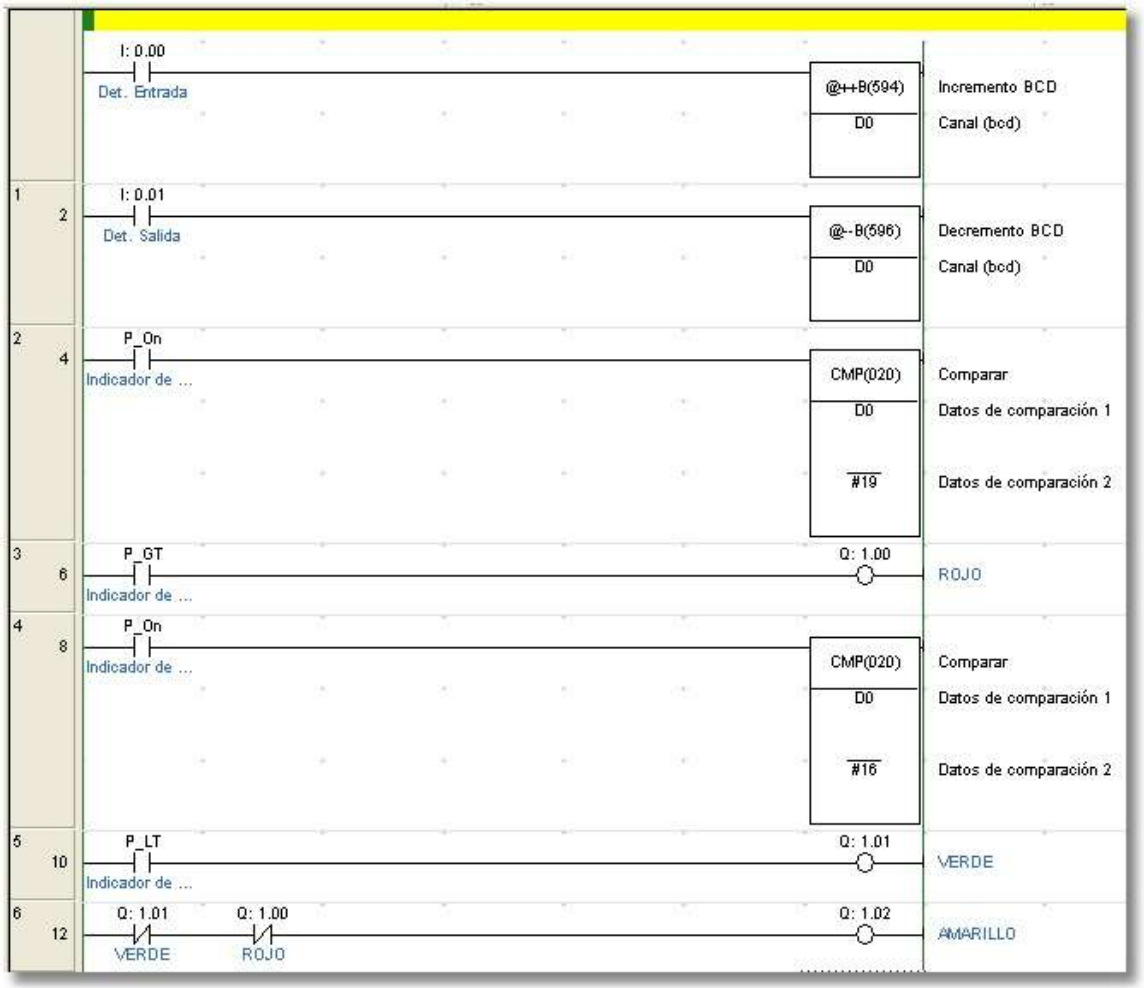
El autómata PLC1 debe programarse para controlar los vehículos que entran (CIO 000.00) y los que salen (CIO 000.01) del mismo.

- **La capacidad del parking es de 20 vehículos por lo que lucirá un semáforo:**
  - verde (CIO 001.02), si hay menos de 16 plazas ocupadas
  -amarillo (CIO 001.01), entre 16 y 19 plazas
  -rojo (CIO 001.00), cuando llega a 20

- **El número de vehículos del parking se almacena en el registro D0000.**

Programación del autómata:

► **El proceso de programación del autómata se realizará desde CX-Programmer (parking.cxp).**



Configuración del servidor Cx-Server OPC

► **Definiremos los siguientes grupos y puntos en un proyecto de Cx Server OPC tal y como se ha desarrollado en la actividad 7:**

-El PLC con el que comunicamos: PLC1

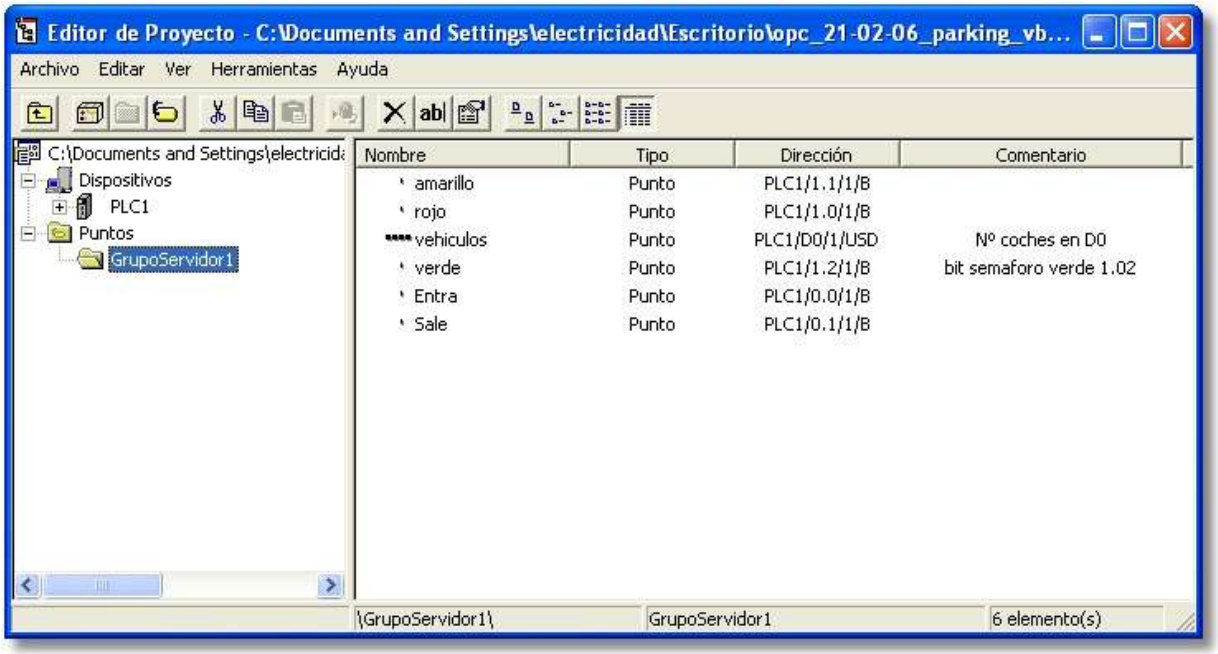
-Un grupo que contendrá todos los puntos (tag o item):

GrupoServidor1

-Seis puntos del PLC para la comunicación:

Entra (CIO 000.00)
Sale (CIO 000.01)
Verde (CIO 001.02)
Amarillo (CIO 001.01)
Rojo (CIO 001.00)
Vehiculos (D0000)

Configuración del servidor Cx-Server OPC



► **Estos pasos estan suficientemente indicados en dicha actividad.**
Puede consultarse el fichero opc_vb.cdm donde se han creado dichos puntos. Si se usa este fichero para desarrollar la actividad, es necesario seleccionarlo desde CX-Server OPC escogiéndolo en la ruta del PC. De esta forma se abrirá este proyecto cuando VB solicite la comunicación desde el Control de Comunicaciones que tendrá insertado.

Desarrollo en Visual Basic de la aplicación.

► **Las aplicaciones que vamos a realizar utilizarán el control de comunicaciones de Omron. Este control de comunicaciones y los comandos de programación disponibles con el servidor Cx-Server OPC de Omron, están realizados bajo el estándar de la fundación OPC. Su diseño nos facilitará el desarrollo de las aplicaciones porque proporcionan unos comandos y menús de sencilla utilización.**

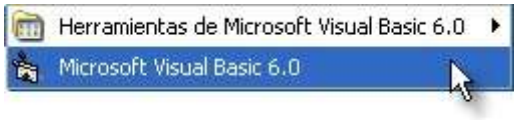
► **En primer lugar desarrollaremos un cliente, utilizando los controles y objetos proporcionados por Omron, los cuales no precisan ninguna programación. Bastará con configurarlos definiendo sus propiedades.**

► **En otro apartado realizaremos la aplicación cliente, programando controles y objetos propios de VB.**

► **Una aplicación más completa, incluiría tanto controles Active X proporcionados por Omron, o cualquier otro desarrollador, así como programación en VB que la perfeccione.**

Desarrollo de la aplicación en Visual Basic con Componentes Omron.

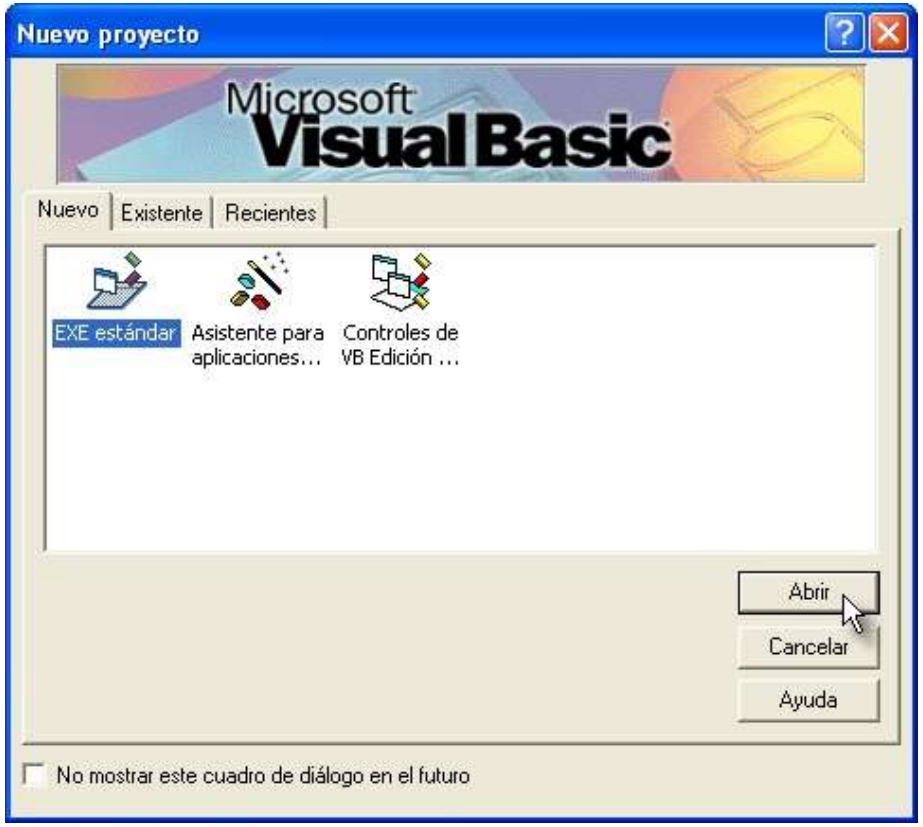
- **Iniciar Microsoft Visual Basic 6.0**



- **Iniciaremos un nuevo proyecto. Si se ha instalado Cx-Server OPC con posterioridad a VB, podremos abrir un nuevo proyecto *CX-Server OPC Proyect.***
- **En este caso se cargan las librerías de objetos de Omron.**

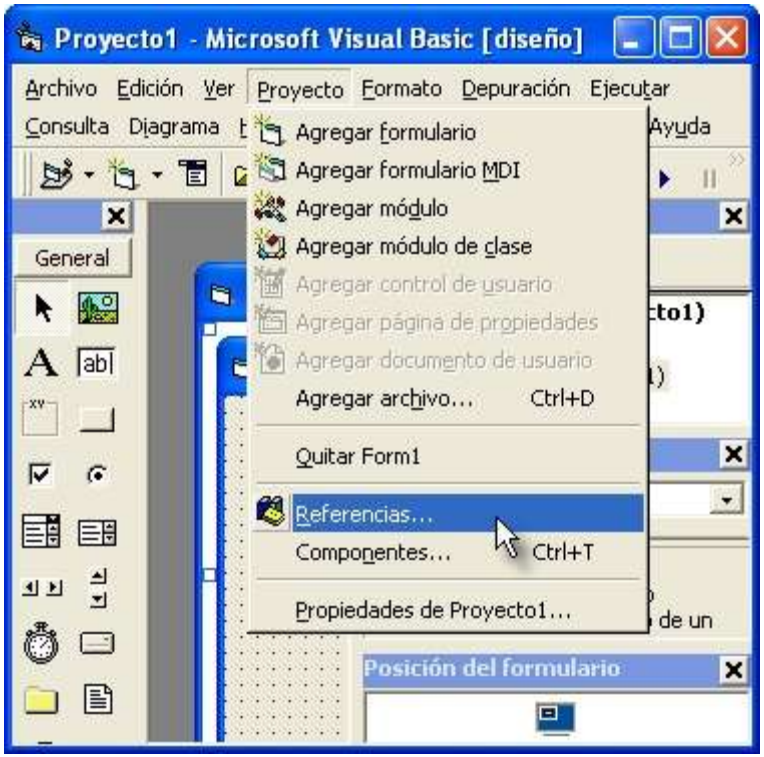
Desarrollo de la aplicación en Visual Basic con Componentes Omron.

- **En caso contrario, abriremos un proyecto EXE estándar:**



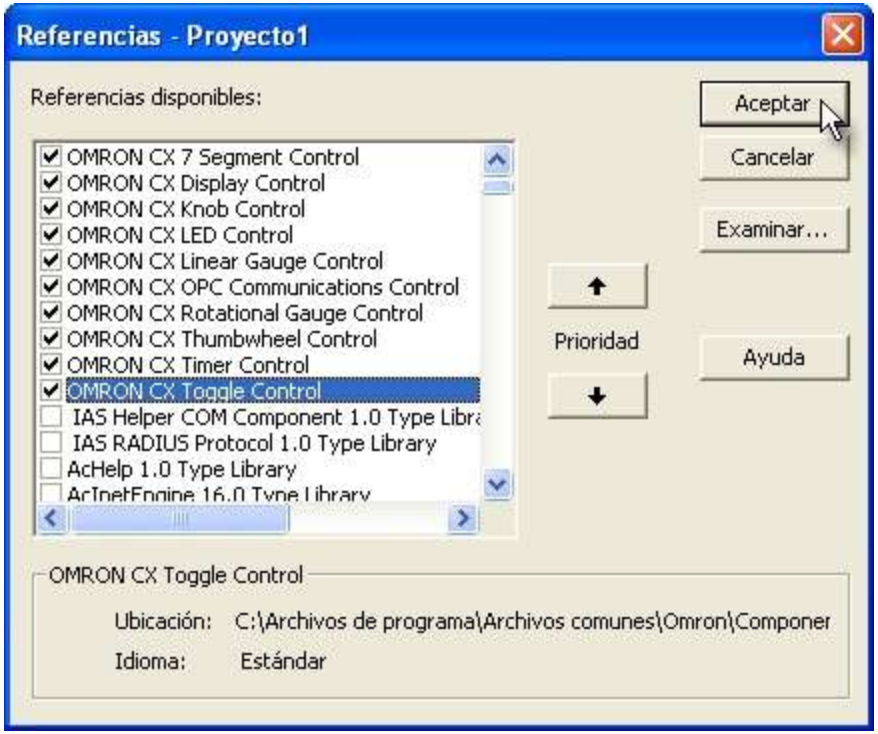
Desarrollo de la aplicación en Visual Basic con Componentes Omron.

- **al que tendremos que añadir las librerías en →*proyecto* →*referencias.***



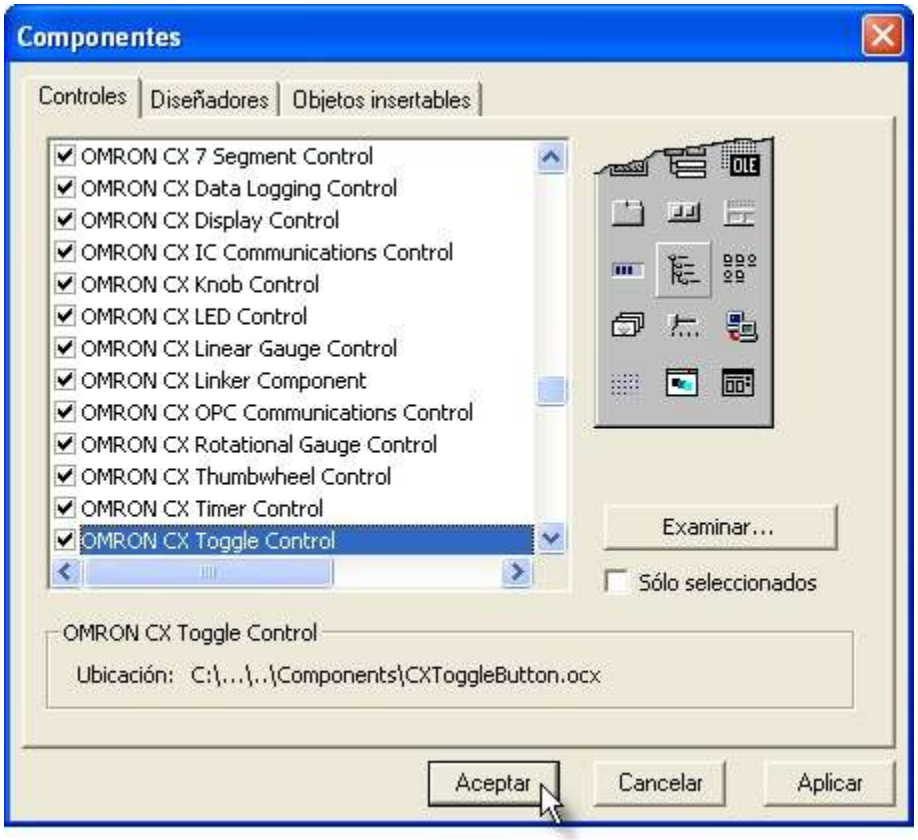
Desarrollo de la aplicación en Visual Basic con Componentes Omron.

► **En este menú seleccionaremos las librerías de Omron que vamos a utilizar. Escogeremos el control de comunicaciones y los objetos de visualización :**



Desarrollo de la aplicación en Visual Basic con Componentes Omron.

► **Para facilitar la programación situaremos los iconos de los objetos habilitados, en el cuadro de herramientas. Para ello pulsaremos con el botón derecho en dicha barra seleccionando componentes:**



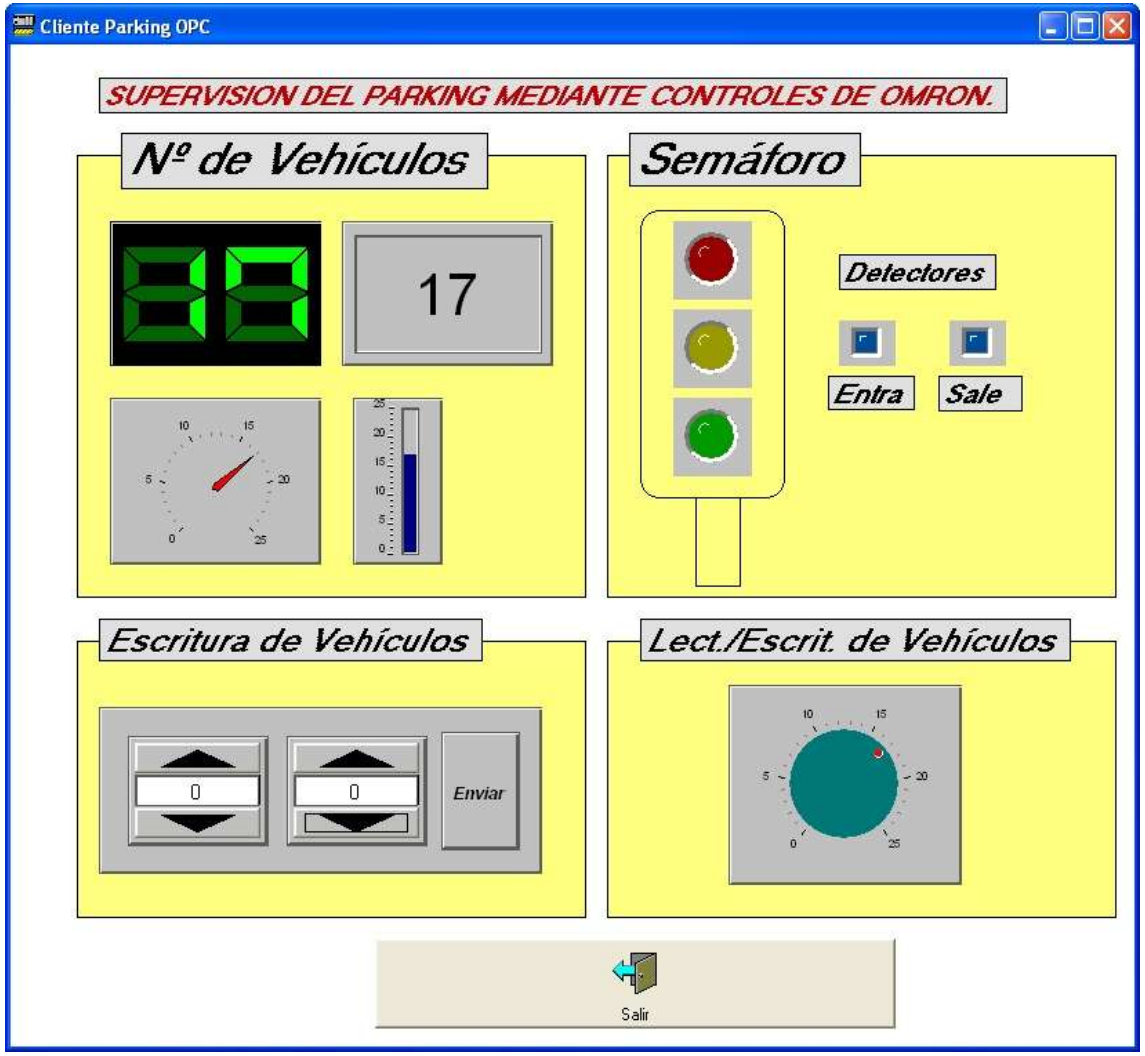
► **En el caso de que añadamos los componentes en primer lugar, se cargarán las referencias automáticamente evitándonos un paso.**

Supervisión en Visual Basic con los controles de Omron.

► **Se desarrollará la aplicación cliente para supervisar el control del Parking utilizando los controles Active-X que proporciona Omron en la versión Demo de Cx_Server OPC. Se trata de obtener un ejecutable que visualice y permita el control del parking por un operador.**

La pantalla final tendrá el siguiente aspecto:

Supervisión en Visual Basic con los controles de Omron.



Insertar el objeto Control de Comunicaciones de Omron:

► **Seleccionar del cuadro de herramientas el objeto OPCComms con el boton izquierdo:**

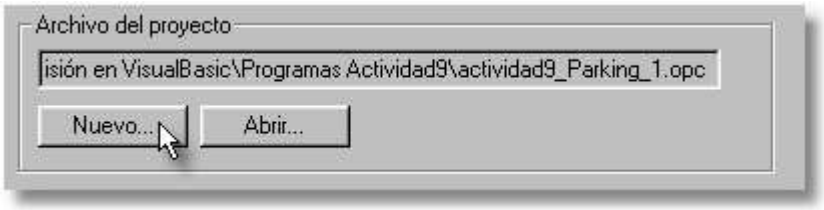
► **Y pegarlo en el formulario arrastrando con el boton izquierdo. En ejecución este objeto no será visible.**
Sobre este objeto, con el botón derecho, se definen las propiedades.

Configurar el *"control de comunicaciones OPC"*

► **En primer lugar, se escogerá el servidor OPC deseado. Aunque en esta unidad didáctica lo hagamos con Omron, es posible utilizar otros servidores. Si en un cliente se desean utilizar dos o más servidores diferentes se insertarán tantos controles ce comunicación como sean necesarios.**

Configurar el *"control de comunicaciones OPC"*

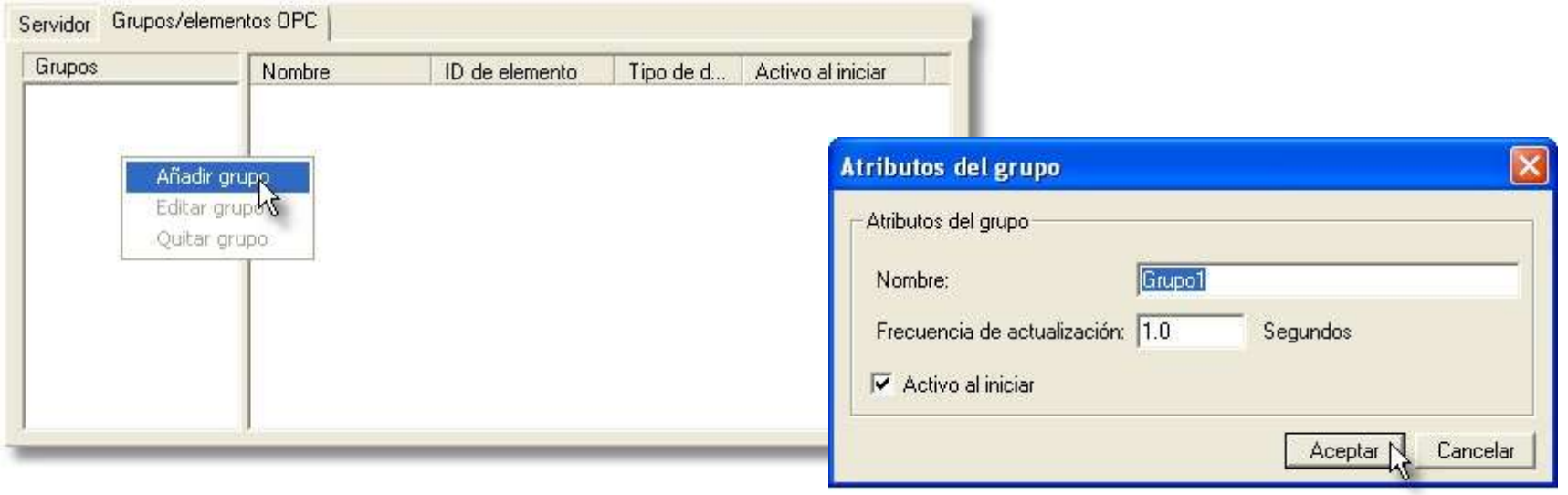
► **En esta misma pantalla, se creará un nuevo proyecto que pertenece al control de comunicaciones en el que se definirán los grupos y elementos. Estos elementos estarán asociados con los puntos definidos en el servidor OPC.**



Nota: Si utilizas el fichero adjunto (con extensión opc) debes indircarle la nueva ruta de tu PC.

Configurar el *"control de comunicaciones OPC"*

► **Seleccionaremos los puntos o elementos del servidor OPC creados anteriormente y los organizaremos en grupos. Con un click derecho en la ventana de grupos añadimos y nombramos los deseados.**



► **Dependiendo de la aplicación se puede seleccionar una actualización de datos a intervalos menores de 1 segundo además de indicar si estará activo al iniciar. Estas dos opciones (de tiempo de actualización y de activación al iniciar la aplicación) es lo que se llama suscripción.**

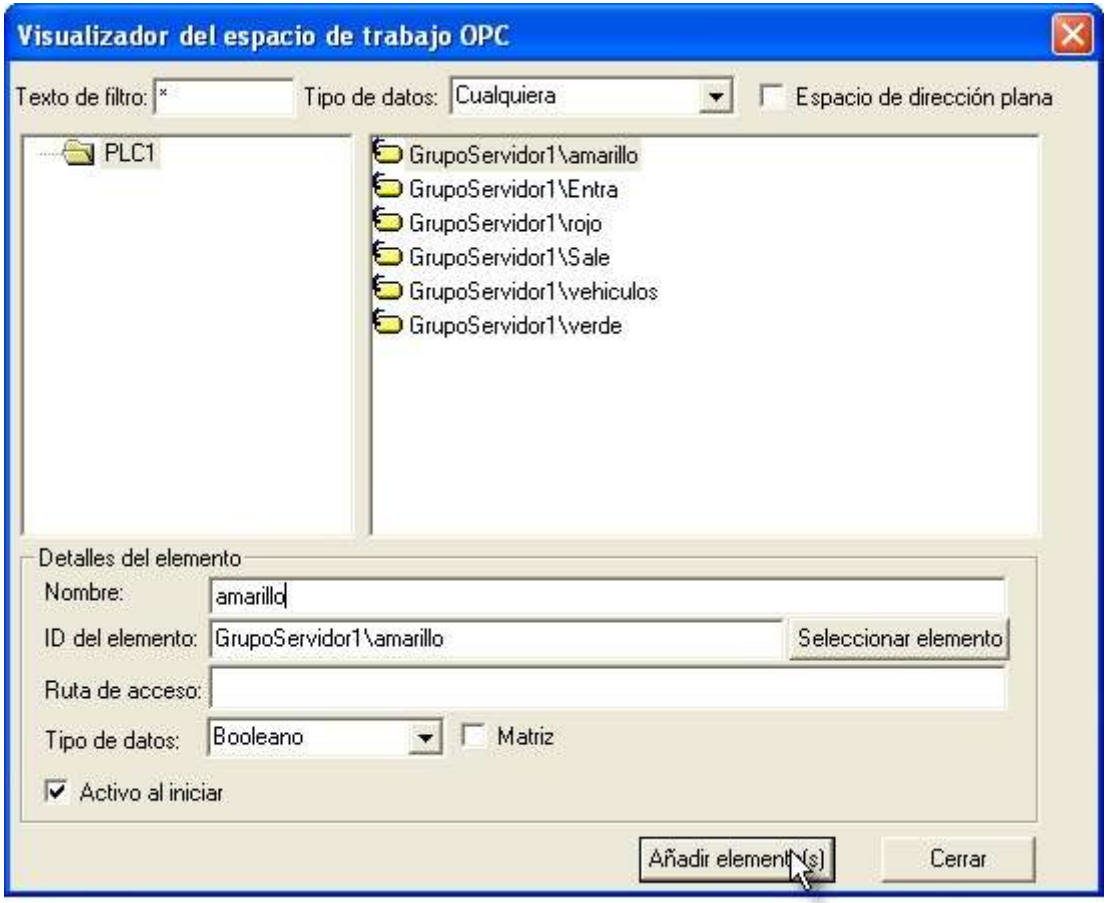
Configurar el *"control de comunicaciones OPC"*

► **Con un click derecho en la ventana de elementos añadimos y nombramos los deseados.**

► **En este momento se abrirá el servidor escogido con el último proyecto que se haya editado. En nuestro caso será el definido anteriormente.**

Configurar el *"control de comunicaciones OPC"*

► **En la figura se muestra como se añade el elemento del servidor GrupoServidor1\amarillo al que podemos cambiar el nombre (*amarillo)*.**



Configurar el *"control de comunicaciones OPC"*

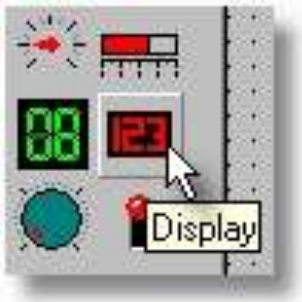
► **Para el resto de los elementos se repite el mismo proceso quedando definido el control de comunicaciones:**

Insertar objetos de Omron:

► **A continuación se muestra solamente el proceso para uno de ellos.**

► **Para ello se selecciona el objeto de Omron, situándolo y dimensionándolos en el formulario. En las propiedades de estos objetos se definen, en el origen de datos, el servidor OPC con el que están vinculados y el punto o elemento del PLC con el que se establece la comunicación. El proceso es el mismo que se realizó en la hoja de Excel pues en ambos casos se ejecuta VB.**

► **Escoger el objeto display:**



Insertar objetos de Omron:

- **Definir sus propiedades**

Insertar objetos de Omron:

► **Le adjudicamos al display el servidor OPCComms1 (sólo tenemos un control de comuniocaciones insertado en el formulario) del que leerá el dato definido por el *grupo1* y elemento *vehículos*.**

► **El resto de los objetos se insertan y definen con el mismo procedimiento hasta lograr el diseño deseado.**

► **Por último, se ejecuta la aplicación para comprobar su funcionamiento.**

SUPERVISIÓN EN VISUAL BASIC MEDIANTE PROCEDIMIENTOS. (PROGRAMACIÓN).

► **Desarrollaremos la aplicación cliente para supervisar el control del Parking utilizando los controles Active-X de Visual Basic: botones, cuadros, etiquetas, etc.**

► **Es necesario tener instalada la versión Demo de Cx_Server OPC. Ello nos permite, de la misma forma que en la actividad anterior, utilizar el Control de Comunicaciones de Omron que nos facilita la generación de la estructura para un cliente OPC y, ademas, un conjunto de instrucciones o funciones de programación en VB para la lectura, escritura, validaciones, etc. El Control de Comunicaciones de Omron, hace de interface entre el cliente VB y el servidor OPC utilizado, para interpretar dichas instrucciones.**

Conceptos de comunicación con los procedimientos del Control de Comunicaciones de Omron.

Comunicación Síncrona.

► **En la comunicación síncrona el cliente pide al servidor que realice una tarea (lectura o escritura) pasando la ejecución de esa tarea del cliente al servidor. En otras palabras, el cliente espera que el servidor atienda la tarea asignada y cuando éste finaliza, el cliente continúa con la ejecución de su programa en el punto en que hizo la llamada.**

► **De este modo el cliente espera tanto tiempo como le lleve al servidor finalizar la tarea y no se ejecuta nada en el cliente mientras está esperando que el servidor termine la operación.**

► **En la comunicación síncrona se asegura la comunicación del cliente con el dispositivo con el inconveniente de que se hace más lenta.**

Conceptos de comunicación con los procedimientos del Control de Comunicaciones de Omron.

► **Ejemplo de escritura síncrona donde se detiene la ejecución hasta que se completa la operación:**

OPCComms1.Write" Grupo", "Item", VariableX, WaitUntilComplete

► **Ejemplos de lectura síncrona donde también se detiene la ejecución hasta que se completa la operación aunque existen opciones: Lectura de *caché* o de *device* (PLC).**

VariableX=OPCComms1.Read("Grupo", "Item",ReadFromDevice)
VariableY=OPCComms1.Read("Grupo", "Item",ReadFromCache)
VariableZ=OPCComms1.Read("Grupo", "Item",ReadFromCacheOrDevice)

A la hora de programar una lectura síncrona, el interface del servidor OPC de Omron soporta dos tipos*:* de *caché* o de *device* (PLC).

Conceptos de comunicación con los procedimientos del Control de Comunicaciones de Omron.

► **ReadFromCache**

Si se especifica *caché*, será leído el último valor conocido del servidor OPC. No se hará ningún intento de leer el valor actual del dispositivo. El dato será válido si tanto el grupo como el item están activos (si están suscritos y leidos en intervalos regulares desde el dispositivo).

Puesto que no hay ningún motivo para que el servidor OPC lea de nuevo el valor del dispositivo, es mucho más rápido leer valores del caché, siendo recomendable siempre y cuando no sea esencial que el valor esté completamente actualizado. Es decir, cuando el valor no sea crítico.

► **ReadFromDevice**

Si se especifica *device* el estado del grupo e Item será ignorado. (Los Grupos y los Items pueden activarse o desactivarse con instrucciones). La dirección actual de memoria para el punto es siempre leida desde el PLC y sólo cuando el dato es obtenido será devuelto al cliente. Nos aseguramos de que el valor es rigurosamente actualizado aunque hacemos más lenta la ejecución de nuestro programa.

► **ReadFromCacheorDevice**

Esto supondrá que el OPC Comunication Control compruebe que el Item está activo y, si es así, pedirá al servidor que lea el dato de la caché. Si el Item no está activo (no estará en la caché) entonces el OPC server leerá el Item del PLC.

Conceptos de comunicación con los procedimientos del Control de Comunicaciones de Omron.

Comunicación Asíncrona.

► **En la comunicación asíncrona el cliente hace una petición al servidor y continua con la ejecución de su programa mientras el servidor atiende la petición.**

► **Cuando el servidor termina, interrumpe al cliente (con un evento en VB) y le devuelve el dato requerido.**

► **El tiempo entre la petición y la respuesta del servidor es indeterminado. Es decir, el cliente hace la petición y continua con sus operaciones; cuando el servidor esté listo, llamará al cliente con el resultado final de la operación.**

Conceptos de comunicación con los procedimientos del Control de Comunicaciones de Omron.

Suscripción.

► **Esta es la operación normal de un Servidor OPC. Nosotros lo hemos hecho al configurar el Control de Comunicaciones de Omron. Un cliente pide que el servidor cree un grupo con un nombre determinado y añade una lista de items con sus nombres que corresponden a los puntos accesibles (creados) en el servidor (p.e… en un fichero CDM de CX-Server).**
El cliente puede pedir que el grupo sea creado con un determinado período (frecuencia) de actualización y si está activo o no activo al iniciar la aplicación cliente.

► **Ver diapositiva 20 de esta actividad, donde hemos realizado la suscripción.**

► **Esto significa que el servidor enviará el valor, calidad y timestamp de cada item del grupo al cliente en intervalos regulares. El tiempo exacto depende de varios factores. La mayoría de los servidores OPC incluido el CX-Server OPC soportan un limitado número de frecuencias de actualización. El servidor devolverá al cliente un período de actualización lo más cercano al solicitado.**
CX-Server OPC soporta los períodos siguientes:
100, 500, 1000, 2000, 5000, 10000, 60000, 120000 milisegundos.

Conceptos de comunicación con los procedimientos del Control de Comunicaciones de Omron.

► **Los grupos e ítems también pueden activarse o desactivarse para la suscripción con las instrucciones EnableGroup y EnableItem. (true o false), además de la opción elegida en la casilla de activación inicial en la ventana donde los hemos suscrito. (Ver anexo)**

► **La suscripción supone menor carga de trabajo para el servidor y por tanto es el método usualmente empleado para recibir datos. Esto podría ocurrir con un programa SCADA cuando una página es visualizada (p.e. el grupo es activado cuando se carga la página y desactivado cuando se cierra).**

Conceptos de comunicación con los procedimientos del Control de Comunicaciones de Omron.

► **Si se desean realizar lecturas asíncronas de los grupos e items suscritos con los intervalos definidos, es necesario que anteriormente se programe la función GetData que inicia el proceso de eventos OnData:**

OPCComms1.GetData "Grupo", "Item" , OnChange
OPCComms1.GetData "Grupo", "Item" , Continous

► **El valor que devuelve el servidor se recoge en el cliente programando el evento OnData. (Ver anexo)**

► **El dato asociado al item sólo será devuelto (se generará un evento) si ha cambiado durante el período de actualización (opción OnChange). Si un grupo tiene un período de 1000ms y ningún item del grupo cambia en ese tiempo, el cliente no recibirá ningún dato. Si el grupo está inactivo, el cliente no recibirá ninguna notificación de cambios. Si el grupo está activo sólo los items activos y que cambien en el período serán devueltos al cliente. Con la opción Continuous el dato será enviado en el intervalo de tiempo suscrito incluso si su valor no ha cambiado. En la práctica esta es una diferencia muy pequeña y la mayoria de servidores OPC (incluido CX-Server OPC v1.2) solo devolverán el valor si hay cambio.**

Conceptos de comunicación con los procedimientos del Control de Comunicaciones de Omron.

► **Además de recoger los datos con el evento OnData en intervalos regulares, el valor actual, calidad y timestamp asociados a cada punto (Item) suscrito, son almacenados en la caché del servidor, y pueden ser obtenidos posteriormente mediante una lectura síncrona desde la caché.**

Conceptos de comunicación con los procedimientos del Control de Comunicaciones de Omron.

► **Si se programa una petición de lectura asíncrona por instrucción como por ejemplo:**

OPCComms1.Read "Grupo", "Item", ReadFromDeviceAsync

no se precisa suscripción. Programar una lectura asíncrona es como suscribirse para recibir el dato una sola vez. El programa sigue ejecutándose y cuando el valor indicado para su lectura está disponible en el servidor, se genera un evento OnData, como en el caso de la suscripción, para que el dato pueda recogerse por el cliente.

► **Si se programa una escritura asíncrona como por ejemplo:**

OPCComms1.Write "Grupo", "Item", VariableX, NoWaiting

se pide al servidor que escriba datos en el PLC, pero no esperará hasta que la escritura se complete. La ejecución vuelve al cliente inmediatamente, antes de que el dato haya sido escrito. Como en la escritura síncrona el estado activo del grupo e item son ignorados. Cuando se complete la escritura, la caché usada para la lectura síncrona se actualiza y el servidor devuelve una confirmación de la escritura al cliente mediante una llamada (enviando un evento al cliente). Con un OnData podríamos recoger el valor, calidad y timestamp de la variable escrita como confirmación de escritura.

Lectura y escritura con Comunicación Síncrona:

► **En el siguiente programa de VB se puede leer o escribir el valor del número de vehículos del parking. No se incluyen más opciones puesto que solamente se trata de exponer las diferencias de una comunicación síncrona.**
La pantalla final tendrá el siguiente aspecto:

Lectura y escritura con Comunicación Síncrona:

- **El código de programa es el siguiente:**

```
Dim intvehiculos As Integer  'Definición de la variable que contencrá el nº de vehículos

Private Sub CmdLee_Click()

    intvehiculos = OPCComms1.Read("Grupo1", "vehiculos", ReadFromCacheOrDevice)
    'Se lee del servidor OPC el nº de vehículos y se almacena en la variable intvehiculos
    'El programa se detendrá en este punto hasta completar la lectura
    TxtLee.Text = intvehiculos
    'Se escribe el valor en el cuadro de texto

End Sub

Private Sub CmdEscribe_Click()

    intvehiculos = CInt(TxtEscribe.Text)
    'Se almacena en la variable intvehiculos el valor del cuadro de texto
    OPCComms1.Write "Grupo1", "vehiculos", intvehiculos, WaitUntilComplete
    'Se escribe el valor de la variable intvehiculos en el Servidor OPC
    'El programa se detendrá en este punto hasta completar la escritura

End Sub

Private Sub CmdSalir_Click()
    End
End Sub
```

Lectura y escritura con Comunicación Síncrona:

► **La comunicación síncrona también puede realizarse con la orden Value. En este caso se programa con las órdenes:**

Lectura:

intvehiculos = OPCComms1.Value("Grupo1", "vehiculos")

Escritura:

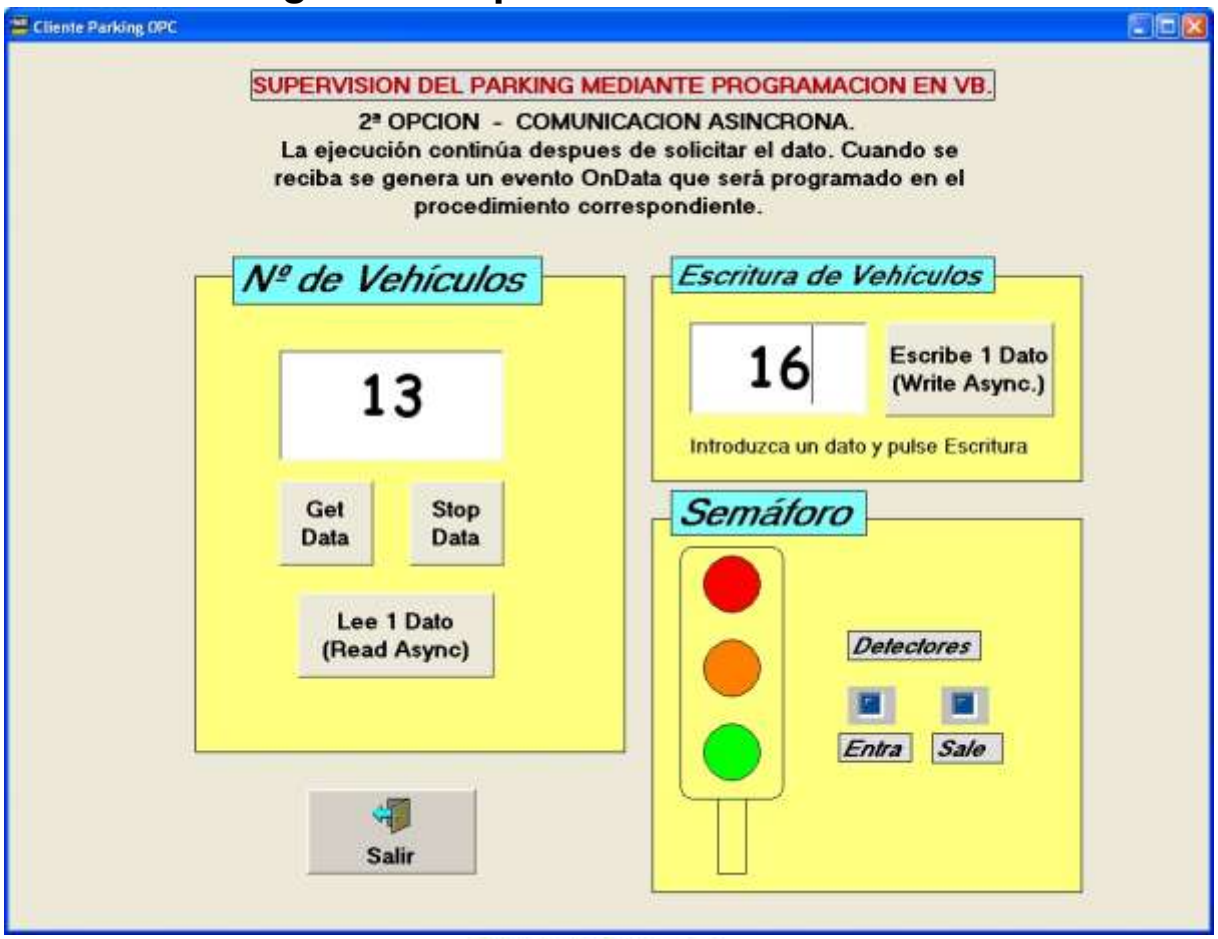
OPCComms1.Value("Grupo1", "vehiculos") = intvehiculos

Nota: Si deseas ejecutar el programa que se adjunta, debes indicar las nuevas rutas para los ficheros:
a.- Con extensión .opc definido en *propiedades* del Control de Comunicaciones.
b.- Con extensión .opt definido en el CX-ServerOPC del PC donde se ejecute.

Lectura y escritura con Comunicación Asíncrona:

► **Se trata de obtener un ejecutable que visualice y permita el control del parking por un operador.**

La pantalla final tendrá el siguiente aspecto:



Lectura y escritura con Comunicación Asíncrona:

- **El código de programa es el siguiente:**

```
Dim intvehiculos As Integer


'PROCEDIMIENTO PARA INICIAR LA ADQUISICIÓN DE DATOS (GENERA EVENTOS OnData) A INTERVALOS REGULARES,
'DEFINIDOS EN LA SUSCRIPCION.

Private Sub CmdGetData_Click()
    OPCComms1.GetData "Grupo1", "vehiculos"
End Sub
'Cada vez que se reciba el dato vehiculos el programa ejecutará el procedimiento:
'Private Sub OPCComms1 OnData


'PROCEDIMIENTO PARA DETENER LA ADQUISICIÓN DE DATOS A INTERVALOS REGULARES.
'Deja de solicitar el dato vehiculos y por tanto de ejecutar el procedimiento OnData
'cada vez que lo reciba

Private Sub CmdStopData_Click()
    OPCComms1.StopData "Grupo1", "vehiculos"
End Sub
```

Lectura y escritura con Comunicación Asíncrona:

```
' PROCEDIMIENTO PARA SER EJECUTADO CADA VEZ QUE SE ACTIVA UN EVENTO OnData

Private Sub OPCComms1_OnData(ByVal Group As String, ByVal Item As String, _
ByVal value As Variant, ByVal BadQuality As Boolean)

If Group = "Grupo1" And Item = "vehiculos" Then
    TxtDatos.Text = value
End If
If Group = "Grupo1" And Item = "amarillo" Then
    If value = True Then
        SemaforoAmarillo.FillColor = &H80FF&
    Else
        SemaforoAmarillo.FillColor = vbWhite
    End If
End If
If Group = "Grupo1" And Item = "verde" Then
    If value = True Then
        SemaforoVerde.FillColor = vbGreen
    Else
        SemaforoVerde.FillColor = vbWhite
    End If
End If
If Group = "Grupo1" And Item = "rojo" Then
    If value = True Then
        SemafoRorojo.FillColor = vbRed
    Else
        SemafoRorojo.FillColor = vbWhite
    End If
End If
If Group = "Grupo1" And Item = "Entra" Then
    If value = True Then
        DetectorEntra.LEDColour = vbRed
    Else
        DetectorEntra.LEDColour = vbWhite
    End If
End If
If Group = "Grupo1" And Item = "Sale" Then
    If value = True Then
        DetectorSale.LEDColour = vbRed
    Else
        DetectorSale.LEDColour = vbWhite
    End If
End If

End Sub
```

Lectura y escritura con Comunicación Asíncrona:

```
'PODEMOS ACTIVAR LA LECTURA DE DATOS A INTERVALOS REGULARES AL COMENZAR EL PROGRAMA
'INCLUYENDO LOS COMANDOS GetData EN LA CARGA DEL FORMULARIO.

Private Sub Form_Load()
    OPCComms1.GetData "Grupo1", "vehiculos"
    OPCComms1.GetData "Grupo1", "amarillo"
    OPCComms1.GetData "Grupo1", "rojo"
    OPCComms1.GetData "Grupo1", "verde"
    OPCComms1.GetData "Grupo1", "Entra"
    OPCComms1.GetData "Grupo1", "Sale"

End Sub
```

```
' PROCEDIMIENTO PARA LEER UN DATO MEDIANTE EL COMANDO READ
Private Sub CmdLee1DatoRead_Click()
    OPCComms1.Read "Grupo1", "vehiculos", ReadFromDeviceAsync
    'Txt1Dato.Text = intvehiculos
End Sub
```

```
'PROCEDIMIENTO PARA ESCRIBIR EL DATO VEHICULOS INTRODUCIDO EN EL CUADRO DE TEXTO TxtEscri

Private Sub CmdEscri1Dato_Click()
    intvehiculos = CInt(TxtEscri.Text)
    OPCComms1.Write "Grupo1", "vehiculos", intvehiculos, NoWaiting
End Sub
```

```
Private Sub CmdSalir_Click()
End
End Sub
```

```
Private Sub LED4_GotFocus()
TxtDatos.Text = "HTFYT"
End Sub
```

ANEXO : Interfaz de script

► **La interfaz de script de Visual Basic define los controles de comunicaciones OPC. (Nota: la aplicación de desarrollo también puede tener controles específicos adicionales aplicables a todos los objetos, por ejemplo Excel provee un método BringToFront. Consultar la ayuda para el programa de desarrollo para los detalles de estos métodos.)**

Funciones.

Value	Función para obtener y dar valor de forma síncrona a un Item OPC
Read	Función para leer el valor de un Item OPC
Write	Función escribir un valor en un Item OPC
GetData	Función para comenzar el evento OnData
StopData	Función para detener el evento OnData
OnData	Evento para recibir notificación de cambio del valor de un dato
EnableGroup	Función para activar (true o false) el estado de un Grupo OPC
EnableItem	Función para activar (true o false) el estado de un Item OPC
IsBadQuality	Verifica si un Item está indicando " mala calidad " actualmente
About	Abre el menú “acerca de”
Help	Abre el menú “ayuda”
ConnectToServer	Se conecta a un servidor OPC que, opcionalmente añade definiciones de cliente, grupo e item para el servidor. Nota: generalmente no es necesario llamar a este método, como muchos otros métodos son llamados automáticamente si es requerido.
Disconnect	Se desconecta de un servidor OPC. Nota: generalmente no es necesario llamar a este método, como muchos otros métodos son llamados automáticamente si es requerido.
ListGroups	Devuelve un listado de los Grupos OPC en el proyecto.
ListItems	Devuelve un listado de los Items OPC en el proyecto.
VarType	Devuelve el tipo de dato de un Item OPC.

Value

- Función para obtener y dar valor de forma síncrona a un Item OPC

- **Ejemplo 1:**

intVal = OPCComms1.Value("MyGroup", "BoilerTemp")

En el ejemplo se lee el valor del Item o Punto "*BoilerTemp*" del Grupo "*MyGroup*" y se almacena en la variable *intval* del script.

- **Ejemplo 2:**

OPCComms1.Value("MyGroup", "BoilerTemp") = 50

En el ejemplo se escribe el valor "50" en el Item o Punto "*BoilerTemp*" del Grupo "*MyGroup*"

NOTA: Value es la propiedad por defecto de todos los objetos, por lo tanto podrá omitirse y será lo mismo escribir:

intVal = OPCComms1.Value("MyGroup", "BoilerTemp")
intVal = OPCComms1("MyGroup", "BoilerTemp")

Read

- **Función para leer el valor de un Item OPC**

- **Ejemplo: Lectura síncrona desde Dispositivo (Device):**

intVal = OPCComms1.Read("MyGroup", "BoilerTemp", ReadFromDevice)

En este ejemplo, el item "BoilerTemp" en el grupo "MyGroup" será leído desde el dispositivo (por ejemplo. PLC) por el servidor OPC y el valor ser guardado en la variable "IntVal". El script esperará que se complete la operación de lectura antes de continuar ejecutando la siguiente línea. Esto es idéntico a la operación del método de "Value".

- **Ejemplo: Lectura síncrona desde Cache:**

intVal = OPCComms1.Read("MyGroup", "BoilerTemp", ReadFromCache)

En este ejemplo, el item "BoilerTemp" en el grupo "MyGroup" será devuelto por el servidor OPC desde su caché, y su valor será guardado en la variable "IntVal".
El script esperará a que la operación de lectura termine antes de continuar con la ejecución de la siguiente línea.
Si el valor no está disponible en la memoria caché (por ejemplo porque el punto no está activo) entonces un error (E_FAIL) será devuelto y la calidad del Item será puesta a calidad mala (bad cuality).

Read

- Función para leer el valor de un Item OPC

- Ejemplo: Lectura síncrona desde Caché o Dispositivo si la Caché no esta accesible:

intVal = OPCComms1.Read("MyGroup", "BoilerTemp", ReadFromCacheOrDevice)

En este ejemplo, el item "BoilerTemp" del grupo "MyGroup" será devuelto por el servidor OPC de su caché, y su valor será guardado en la variable "IntVal".
El script esperará a que la operación de lectura termine antes de continuar con la ejecución de la siguiente línea.
Si el valor no está disponible en la memoria caché (por ejemplo porque el punto no está activo), entonces será leído del dispositivo.

- Ejemplo: Lectura asíncrona:

OPCComms1.Read "MyGroup", "BoilerTemp", ReadFromDeviceAsync

En este ejemplo, el item "BoilerTemp" en el grupo "MyGroup" será leído desde el dispositivo (por ejemplo. PLC) por el servidor OPC. El script continuará ejecutando la siguiente línea inmediatamente y cuando el dato sea leido, se generará un evento OnData (que podrá ser programado en un procedimiento).

Write

- **Función para escribir un valor en un Item OPC**

- **Ejemplo de escritura síncrona**

OPCComms1.Write "MyGroup", "BoilerTemp", NewValue, WaitUntilComplete

En el ejemplo, el valor de la variable "NewValue" será escrito en el Item "BoilerTemp" del grupo llamado "MyGroup". El script (programa) esperará a que la operación se haya completado antes de continuar ejecutando la siguiente línea de programa. Es idéntica a la operación del método *Value*.

- **Ejemplo de escritura asíncrona**

OPCComms1.Write "MyGroup", "BoilerTemp", NewValue, NoWaiting

En el ejemplo, el valor de la variable "New Value" será escrito en el Item "BoilerTemp" del grupo llamado "MyGroup". El script continuará inmediatamente con la siguiente línea de programa.

GetData

► **Comienza la lectura asíncrona de un Item OPC según el intervalo de actualización definido para el Grupo**

► **Ejemplo**

OPCComms1.GetData "MyGroup", "MyItem"

En este ejemplo, MyItem de MyGroup será leido a cada intervalo de actualización del grupo.El dato es entonces enviado a la rutina *OnData*.
Un elemento individual de un punto, definido como un array en el fichero CDM, puede ser invocado para su lectura o escritura.

GetData

► Comienza la lectura asíncrona de un Item OPC según el intervalo de actualización definido para el Grupo

► **Ejemplo**

Comms1.GetData "MyPLC", "MyPoint[2]", nUpdateRate

En el ejemplo el tercer elemento de MyPoint (definido como plc1/DM500/10/USH) puede ser invocado como MyPoint[2]. Por defecto, si el elemento del array no se especifica, se accederá a todo el punto (todos los elementos del array).
Usando como cuarto parámetro el comando *OnChange,* el método *GetData* solo devolcerá los puntos cuando cambien.

► **Ejemplo**

Comms1.GetData "MyPLC", "MyPoint", nUpdateRate, OnChange

***OnChange* es del tipo 'UpdateSetting'. Por defecto, la 'UpdateSetting' se pone a *Continuous*, para hacerla compatible con versiones anteriores.**

StopData

- Detiene la lectura asíncrona de un Item OPC

- Ejemplo

OPCComms1.StopData "MyGroup", "MyItem"

En el ejemplo, la lectura asíncrona de MyItem de MyGroup será detenida

OnData

► **Este evento recibe la notificación del cambio de un dato.**

Esto se dará tanto cuando el dato es llamado mediante un comando GetData como cuando es llamado mediante comandos Read o Write en modo asíncrono.

► **Ejemplo**

Private Sub OPCComms1_OnData(ByVal Group As String, ByVal Item As String, ByVal Value As Variant,ByVal BadQuality as Boolean)
TextBox1 = Item
Segment1 = Value
End Sub

En el ejemplo se escribe el valor del punto en el componente *CX-Server 7 Segmentos y el Cuadro de Texto* muestra el nombre del Item.
En este ejemplo sólo debe estar suscrito este Item con GetData pues si hay más de uno activos, se recibirá el valor del Item que cambie de valor. En el siguiente ejemplo se observa la forma de recoger el valor deseado.
Si BadCuality está a True el valor puede ser incorrecto (ej. Desde la caché cuando el PLC ha sido desconectado).

OnData

► **Ejemplo**

La rutina OnData puede ser ampliada para incluir las expresiones lógicas sobre el nombre del Item entrante para leer el objeto con el dato correcto.

```
Private Sub OPCComms1_OnData(ByVal Group As String,ByVal Item As String, ByVal
Value As Variant, ByVal BadQuality as Boolean)
If Item = "MyItem" then
Segment1 = Value
Else if Item = "MyOtherItem" then
Cells(1,1) = Value
End if
End Sub
```

EnableGroup

- Cambia el estado de un Grupo específico OPC a Permitido o Inhabilitado.

- Ejemplo

OPCComms1.EnableGroup "MyGroup1", True
OPCComms1.EnableGroup "MyGroup2", False

En el primer ejemplo se activa el grupo *MyGroup1.* En el segundo el *MyGroup2* se hace inactivo (deteniendo las respuestas a las llamadas de comunicación y suscripción).

EnableItem

- Cambia el estado de un Item específico OPC a Permitido o Inhabilitado.

- Ejemplo

OPCComms1.EnableItem "MyGroup", "MyItem1", True
OPCComms1.EnableItem "MyGroup", "MyItem2", False

En el primer ejemplo se activa el *Item MyItem1.* En el segundo el *MyItem2* se hace inactivo (deteniendo las respuestas a las llamadas de comunicación y suscripción).

IsBadQuality

- **Chequea si un Item está indicando 'BadQuality'**

- **Ejemplo**

bBad = OPCComms1.IsBadQuality("MyGroup","MyItem")

La variable boleana *bBad* se pone a *True* si el Item *MyItem* indica *'BadQuality'* (por ejemplo si está asociado a un PLC desconectado).

AboutBox

- Abre la About Box.

- Ejemplo

OPCComms1.AboutBox

Help

- Abre la información de ayuda (Help)

- Ejemplo

OPCComms1.Help

ConnectToServer

► **Conecta con un Servidor OPC. Opcionalmente añade grupo cliente y definición de item al Servidor.**

Nota: normalmente no es necesario llamar a este método. Como muchos otros métodos es llamado automaticamente cuando es requerido.

► **Ejemplo**

OPCComms1.ConnectToServer(True) ‘ connects and informs server of groups and items

DisconnectTo

► **Desconecta el Servidor OPC.**

Nota: normalmente no es necesario llamar a este método. Como muchos otros métodos es llamado automaticamente cuando es requerido.

► **Ejemplo**

OPCComms1.Disconnect

VarType

► **Devuelve el tipo de dato de una variable como una variant conteniendo el valor largo que puede ser comparado con los tipos estandar de VisualBasic (p.e. vbArray, vbBoolean, vbByte, vbChar, vbCurrency, vbDate, vbDecimal, vbDouble, vbEmpty, vbInteger, vbLong, vbNull, vbObject, vbSingle, vbString, vbUserDefinedType and vbVariant).**
Vea las ayudas de Visual Basic (or VBScript) para más información de los tipos de variables.

► **Ejemplo**

```
Dim v as variant
v = OPCComms1.VarType("Group1", "Item1")
‘ el tipo de dato devuelto a v puede ser ahora comparado con los tipos de variables
```