BERRIKUNTZA TEKNOLOGIKOA
INNOVACIÓN EN LA TECNOLOGÍA

EUSKO JAURLARITZA GOBIERNO VASCO

Tknika

Actividad 6:

OPC. Conceptos.

1.- Objetivos de la actividad.

► **Conocer la necesidad de la comunicación estandarizada OPC en la automatización industrial.**

► **Conocer la estructura de comunicación OPC.**

► **Conocer las funciones definidas por la Fundación OPC.**

2.- Origen de OPC:

► En los equipos de automatización, como PLC´s, variadores, controladores,…, ha ido creciendo el número de componentes de comunicación con sus buses y protocolos.

► Nos situamos en los niveles altos de la pirámide de automatización como son la *Gestión de Proceso* (Datos sobre el proceso productivo adquiridos y procesados por sistemas SCADA) y la *Gestión de Negocio* (Integración de la información de planta en los sistemas que gestionan los aspectos productivos y financieros de la fabricación MES y ERP)

2.- Origen de OPC:

► **Los fabricantes de software de estos niveles (scadas, etc.), tenían el problema de mantener y actualizar la gran variedad de *drivers* que comunicaban los distintos equipos de planta con sus productos.**

► **En cooperación con Microsoft, un grupo constituido por cinco empresas, Intellution, Opto-22, Fisher-Rosemount, Rockwell Software e Intuitiv Software, colaboraron para solucionar este problema y dieron origen a la especificación técnica no propietaria definida por la OPC Foundation en Mayo de 1995.**
Puede consultarse en la dirección:
http://www.opcfoundation.org

3.- OPC *(OLE for Process Control)* (Tecnología OLE para el control de procesos).

► **Microsoft estaba trabajando en el desarrollo del OLE 2.0 (Object linking and enbedding) (objetos enlazados e incrustados). Aparentemente esta nueva tecnología podría reemplazar al DDE (Dynamic Data Exchange) (Intercambio dinámico de datos) que hasta ese momento había sido usada extensivamente para el intercambio de datos en sistemas SCADA diseñados para Windows. La nueva tecnología de OLE era más flexible, robusta y eficiente para el entorno industrial que la proporcionada por DDE.**

► **Los apéndices 1, 2 y 3 que se encuentran al final de esta actividad, muestran los conceptos básicos de DDE, OLE, COM y DCOM.**

► **Este grupo de empresas definieron una serie de especificaciones para el control de procesos, basadas en OLE/COM y DCOM de Microsoft y el primer borrador de las mismas fue completado al final de1995, gracias a la colaboración de otras 90 compañías a lo largo del mundo. El primer conjunto oficial de especificaciones de la Fundación OPC, se completó en Agosto de 1996: Data Access Specification 1.0a. (Actualmente se usa al menos la 2.02)**

3.- OPC *(OLE for Process Control)* (Tecnología OLE para el control de procesos).

► **Data Access Specification define como construir las interfaces entre cliente y servidor. El correcto seguimiento de las especificaciones en el desarrollo de clientes garantiza la conectividad con cualquier servidor OPC existente en el mercado. El objetivo es crear una arquitectura genérica cliente/servidor, con la robustez y rapidez requerida en entornos industriales, la cual seria ofrecida a cualquier desarrollador para acabar con los sistemas propietarios.**

► **El método definido por OPC, facilita el intercambio de datos en forma estandarizada y simple en aplicaciones de control y automatización, entre los dispositivos y sistemas de campo y las aplicaciones de supervisión, administrativas y de oficina. Es decir, OPC simplifica la interfaz entre componentes de automatización de distintos fabricantes, con programas y aplicaciones tales como sistemas administrativos y de visualización.**

► **Con estas especificaciones, el diseño de un paquete SCADA, cuya comunicación se realizará con servidores OPC, no necesita disponer de drivers para los numerosos equipos industriales posibles.**

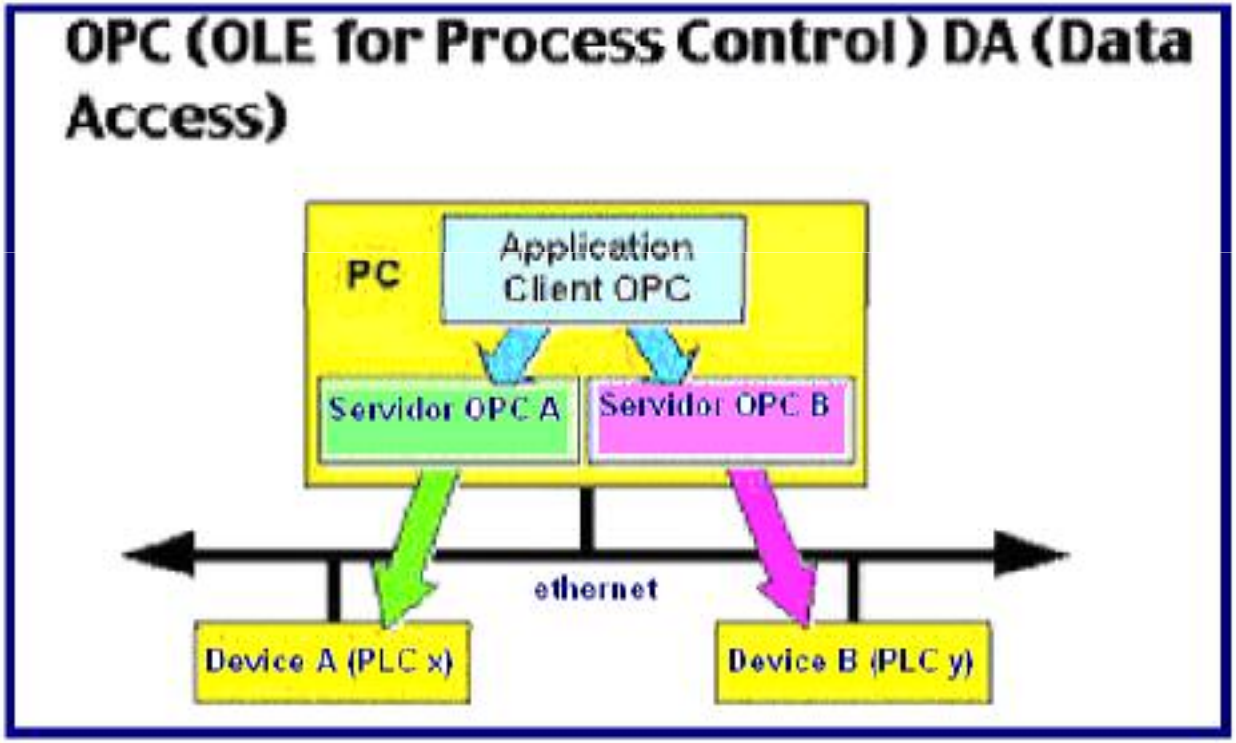
3.- OPC *(OLE for Process Control)* (Tecnología OLE para el control de procesos).

► El software se ha estandarizado y para una aplicación concreta solamente será necesario disponer en el servidor OPC, de los drivers que conviertan los elementos de campo al formato OPC. El cliente OPC, como puede ser un SCADA, Visual Basic,.., siempre se comunica en el mismo formato.

► La estandarización permite que los desarrolladores de software no tengan que reescribir drivers debido a cambios de características o modificaciones de hardware. Además los fabricantes de hardware solamente tendrán que hacer un conjunto de componentes de software para los que los clientes los utilicen en sus aplicaciones.

► Otra gran ventaja de las especificaciones "abiertas" OPC, es la utilización de lenguajes de programación como C++ o Visual Basic como clientes OPC, para la realización de aplicaciones a medida. Las próximas unidades didácticas tratarán sobre ello. El condicionante es que hay que hacerlo bajo Windows.

4.- Arquitectura de automatización industrial basada en OPC



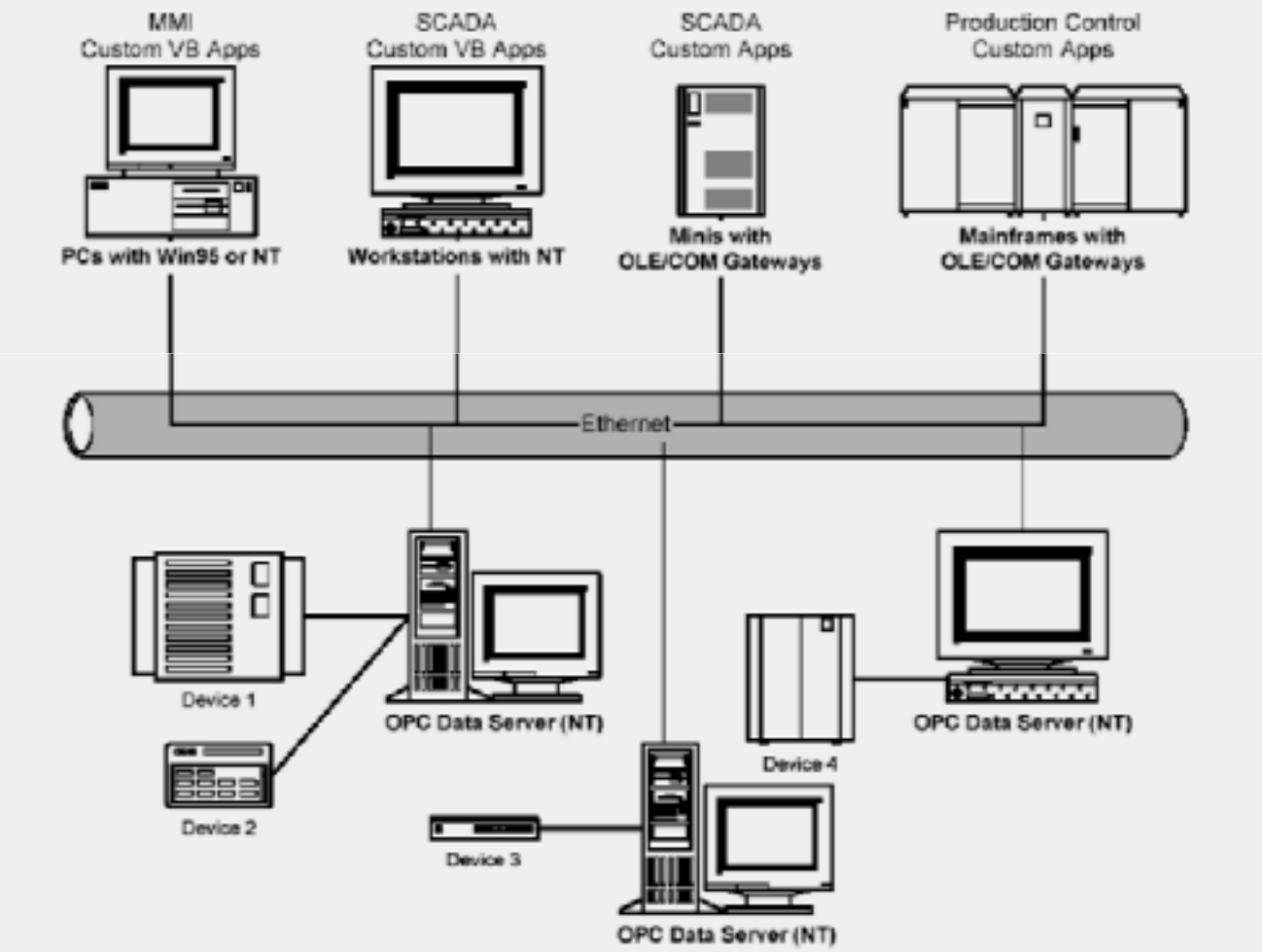
4.- Arquitectura de automatización industrial basada en OPC

► En esta figura se observa una sencilla estructura en la que un PC tiene instalados los servidores de datos OPC de los dos equipos con los que comunica. En el mismo PC también está instalado el cliente OPC que puede ser una aplicación Visual Basic, Scada, etc. La aplicación cliente OPC puede intercambiar datos con equipos de fabricantes diferentes.

► Los drivers para los servidores de datos OPC los facilita el fabricante del equipo conectado pero también están disponibles de forma gratuita numerosos drivers así como un software servidor de datos OPC-DA (Data Access) y un software de interconexión entre equipos de campo, OPC-DX (Data Exchange) en la dirección http://www.kepware.com (Versión DEMO de 2 horas de duración)

► En cada servidor OPC hay una memoria caché de datos relacionada con el dispositivo (PLC) con quien comunica.

4.- Arquitectura de automatización industrial basada en OPC



4.- Arquitectura de automatización industrial basada en OPC

► En esta figura se observa una estructura más compleja. OPC permite utilizar simultáneamente varios servidores para una aplicación cliente y ejecutar varios clientes al mismo tiempo con un servidor OPC.

► En este caso, tres servidores OPC comunican con cuatro equipos de campo. Los clientes OPC en este caso no están en los mismos PC´s que contienen los servidores si no en equipos remotos. Cada una de las aplicaciones de estos clientes podrá tener acceso a los tres servidores.

► Este tipo de comunicación entre varios servidores, situados en equipos diferentes, se desarrollará en la última unidad didáctica en la que se utiliza la configuración DCOM.

5.-Funciones definidas por la Fundación OPC.

► **OPC define varias interfaces desarrolladas para un determinado campo de aplicación:**

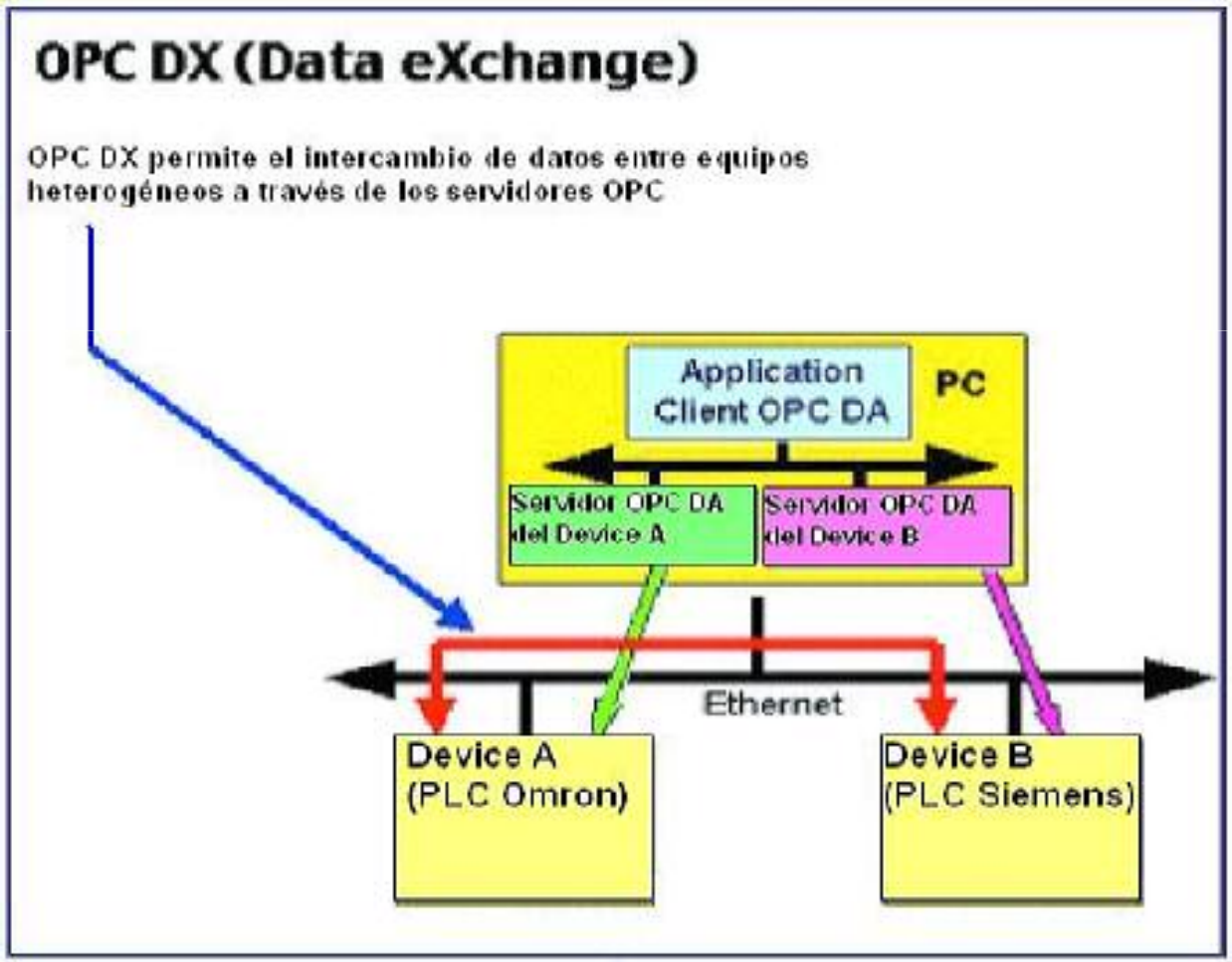
OPC DA (Data Access) Es la función más usada. Está disponible de forma gratuita con numerosos drivers en http://www.kepware.com Permite leer, modificar y monitorizar variables del proceso. La Fundación OPC ofrece una herramienta con la que se puede probar la conformidad de los servidores DA. OPC DA se basa en la tecnología COM/DCOM de Microsoft y sólo está disponible para PC´s con un sistema operativo de Microsoft; la comunicación está limitada a las estaciones de una LAN.

OPC XML-DA utiliza el protocolo XML (eXtensible Merked Lenguage) basado en el método de transporte vía http. Permite establecer la comunicación entre estaciones con distintos sistemas operativos y superar los límites de una LAN, por ejemplo, vía Internet.

OPC A&E (Alarm&Event) se utiliza para transmitir de forma inmediata, alarmas o eventos programados a cualquier cliente OPC. Por una parte estará el servidor OPC-DA comunicando datos normales del proceso, y por otra el OPC-AE para la comunicación de datos críticos. Esto permite que sistemas de información remotos puedan reaccionar y lanzar rutinas de evaluación.

5.-Funciones definidas por la Fundación OPC.

OPC DX (Data Exchange)



5.-Funciones definidas por la Fundación OPC.

OPC DX (Data Exchange)

► La función OPC DX ha sido desarrollada para intercambiar datos entre equipos de forma horizontal. Esta especificación proporciona un recurso para conectar equipos de planta entre sí a través del servidor OPC DX. De esta manera se pueden intercambiar datos de equipos heterogéneos de fabricantes distintos con protocolos de comunicación diferentes a través de sus servidores de datos OPC que sí entienden un lenguaje común. La diferencia con otras comunicaciones de campo habituales es que estos datos no deben ser críticos en el tiempo.

► Es importante recordar en este aspecto, que tanto la comunicación OPC-DX, como la comunicación que hagamos con OPC-DA entre nuestro cliente (VB) y los equipos de campo, no son en tiempo real. Además no tienen la seguridad, fiabilidad y robustez que aportan las comunicaciones de los buses de control y de campo diseñados para otros fines. La comunicación OPC está pensada para los niveles de supervisión, control de la producción y superiores donde no son críticos esos aspectos.

► La comunicación entre equipos con OPC-DX se hará cuando estos aspectos no sean relevantes.

5.-Funciones definidas por la Fundación OPC.

OPC HDA (Historical Data Access) sirve para acceder a todos los valores históricos del proceso (que ya han pasado) contenidos en una base de datos. Con el se realizan informes, gráficos o estadísticas.

OPC-Batch Los productos desarrollados para la industria de proceso tipo Batch se basan en el *IEC 61512-1 Batch Control-Part1*:

La definición de Modelos y Terminología estándar es una necesidad creciente para el intercambio de datos entre estos productos y otros sistemas.

Las interfaces existen en todos los niveles:
-Con los dispositivos de campo (estaciones de monitorización, estaciones de control,…)
-Con los sistemas de Process Management (sistemas de control batch, carga, descarga,…)
-Con los sistemas de Business Management (ERPs y MES)
El intercambio de datos necesita cubrir cuatro tipos básicos de información: Capacidades de equipos, Condiciones de operación actuales, Históricos, y Recetas.

5.-Funciones definidas por la Fundación OPC.

OPC Batch Custom Inteface Specification 2.0 **cubre las definiciones de las *interfaces* y los *'espacios de nombres'* para los siguientes tipos de datos batch:**

-Información sobre el *Current Runtime Batch*
-Información de los equipos necesarios para entender el contexto de la información *del Runtime Batch*
-Registros históricos **de la ejecución Batch, y**
-Contenido de *Recetas Maestras*

5.-Funciones definidas por la Fundación OPC.

OPC-Complex Data. ***Complex Data*** **es un término que describe los items de OPC Data Access, cuyo tipo de datos es una estructura.**

OPC-Complex Data ofrece un mecanismo a los clientes OPC DA para utilizar la estructura de los datos definida precisamente por Complex Data.

OPC DA trabaja con tipos de datos simples. Los datos con estructura compleja habrá que convertirlos utilizando el OPC Complex Data.

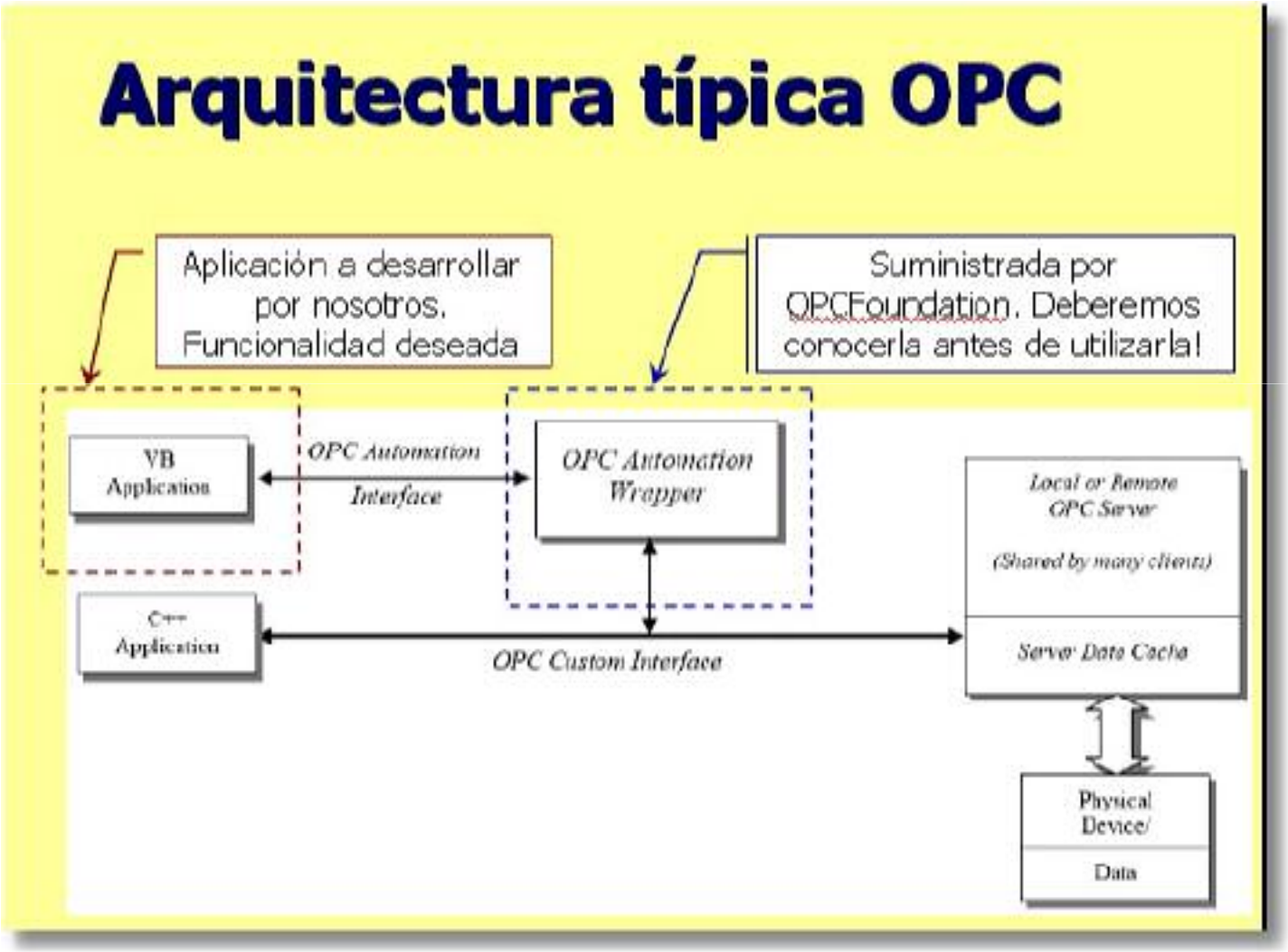
OPC Security Interface específica la manera en que los Servidores OPC deben implementar el sistema de seguridad utilizando las facilidades ofrecidas por el Sistema Operativo.

6.-Requerimientos de Funcionalidad OPC.

► **La siguiente lista de requerimientos de funcionalidad fue tomada del documento Data Access Automation Interface Standard Versión 2.01, publicado por OPC Foundation:**

-OPC es soportado completamente por VC++, Visual Basic y Delphi.
-También por cualquier cliente con interfaz OLE con ciertas limitaciones.
-No soporta el uso con VBScript o JavaScript.
-La especificación OPC requiere como Sistema Operativo al menos Windows 95/98 (con DCOM), Windows XP Windows NT 4.0 o Superior. En todos los casos es recomendable instalar la última versión de Services Pack correspondiente.

7.-Funcionamiento de OPC.



7.-Funcionamiento de OPC.

► **En esta figura se muestra la estructura típica OPC:**

- **Los datos de un Physical Device (PLC) pueden ser leídos o escritos desde el servidor OPC. Como hemos comentado anteriormente nos proveeremos de este servidor OPC a través del fabricante del equipo físico o de Kepware (software gratuito).**

- **A partir de este servidor se puede construir un cliente con una interface personalizada, (OPC Custom Interface) para lo cual se usa el lenguaje de alto nivel C++. Los datos del servidor OPC vienen originalmente para ser utilizados en lenguaje C++. Ello requiere utilizar las numerosas herramientas de programación suministradas por compañías de automatización, usadas para crear clientes y servidores OPC. Sin embargo, estos juegos de herramientas a menudo exigen que el usuario tenga unos detallados conocimientos de C++ y componentes de programación basados en (COM).**

7.-Funcionamiento de OPC.

► La opción más sencilla y habitual, es utilizar OPC Automation Wrapper. Se trata de un interface de automatización usado para conectarse desde un cliente (Visual Basic, Excel y otros lenguajes de programación) a servidores OPC.

► Se llama wrapper (envoltura) porque enmascara o envuelve el interface de programación original "OPC Custom Interface Server", que sirve los datos en C++.
Así, las llamadas desde el programa cliente al OPC Automation Wrapper, son convertidas automáticamente al interface de comunicación original OPC (Custom C++).

► La ventaja de OPC Automation Wrapper, es que proporciona una comunicación con los servidores OPC usando un lenguaje más sencillo.

► Muchos programadores están familiarizados con los servidores y los clientes de OPC creados por los miembros de la fundación de OPC. Por ejemplo, un SCADA es un cliente OPC de fácil programación, que intercambia datos con un servidor OPC (alojado en un PC) el cual a su vez se comunica con los equipos físicos (PLC, etc.). Para el programador, la comunicación en el estándar OPC es transparente, y solamente utiliza las herramientas que proporciona el scada.

7.-Funcionamiento de OPC.

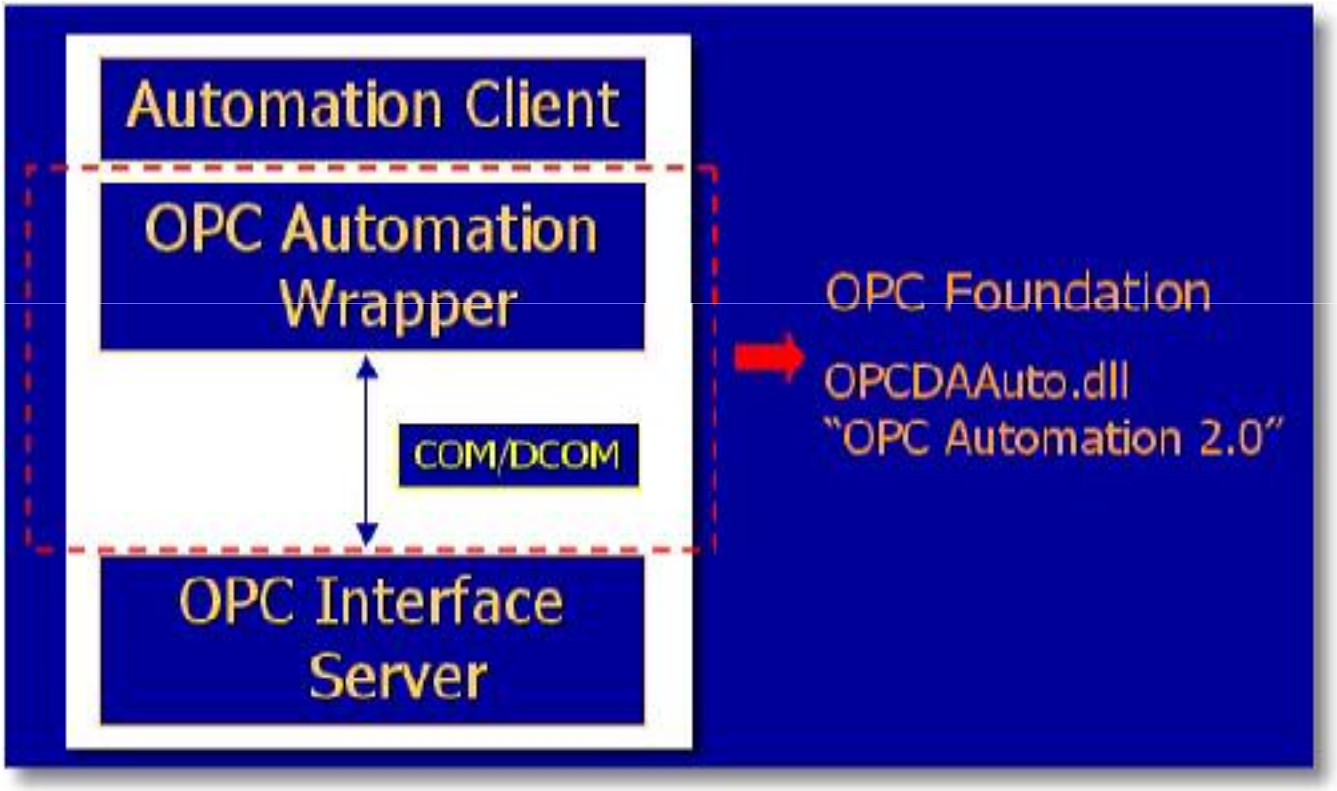
► **Sin embargo, hay pocos desarrolladores de aplicaciones SCADA que estén familiarizados con OPC Automation Wrapper. Una vez conocida su estructura podremos desarrollar aplicaciones cliente en Visual Basic de tamaño medio para la supervisión de un proceso industrial.**

► **Más concretamente, OPC Automation Wrapper proporciona una librería de objetos de programación (COM) que facilita la búsqueda de los servidores OPC creando grupos e items y permitiendo su utilización.**

► **El código fuente de OPC Automation Wrapper inicialmente fue escrito por la fundación OPC. Su especificación está definida en el OPC Data Access Automation v2.0 (hay posteriores).**

► **Para desarrollar nuestra aplicación en Visual Basic, deberemos tener cargada la librería OPCDAAuto.dll y habilitada la referencia OPC Automation 2.0 u OPC Automation Wrapper2.0.**

7.-Funcionamiento de OPC.



7.-Funcionamiento de OPC.

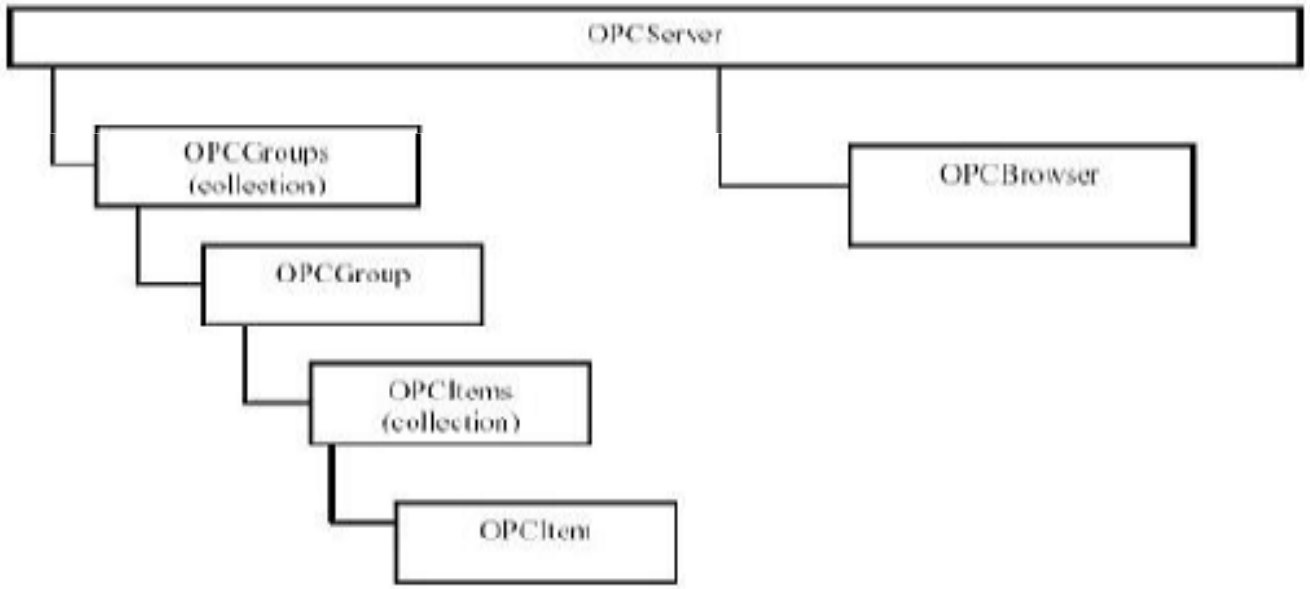
► **Si está instalado KepServerEx de Kepware, la librería estará instalada y la habilitaremos en "referencias". (este proceso se desarrolla en la unidad didáctica correspondiente).**

► **Los miembros de la fundación OPC (como Omron) están autorizados para modificar y distribuir esta librería. En este caso OMRON proporciona los objetos COM que la comprenden. Si hemos instalado Cx-Server OPC la librería de enlaces dinámicos OmronDAAuto.dll estará instalada.**

► **Así mismo, debemos recordar que OPC Automation Wrapper permite la conexión a servidores de OPC local, por ejemplo un ejecutable (.exe) que funciona en la misma computadora, o remotos (en una red podremos acceder a datos servidos por equipos remotos). En este caso será necesario usar DCOM.**

8.- Modelo de Objetos de la Especificación OPC DA Automation Wrapper.

► **El modelo jerárquico de objetos definido por OPC Foundation para DA Automation Wrapper se representa en la siguiente figura:**



8.- Modelo de Objetos de la Especificación OPC DA Automation Wrapper.

► **Debemos diferenciar que por una parte estará el software servidor OPC que tendremos instalado en un PC y que se comunicará con los equipos físicos del proceso (PLCs habitualmente).Será el servidor de acceso a datos OPC. Este software (KepServerEx de Kepware, Cx-Server-OPC de Omron, etc.) tendrá una sencilla estructura de configuración y programación para acceder a los elementos (tags o variables) de los equipos (como una entrada de un PLC). Esta estructura suele estar definida por el software en *equipos* (PLC1,..) *grupos* (grupo1,..) y *elementos* (sensor1-IN 00,..). En las siguientes unidades didácticas se explicará la forma de hacerlo.**

► **Por otra parte tendremos el OPC Automation Wrapper. Será un servidor de objetos OPC. Es aquí donde hay que definir la estructura de objetos del modelo especificado por la Fundacion OPC. Estos objetos no sólo indican el valor (p.ej. 0 ó 1), de una variable sino el estado de conexión (bueno/malo) y el tiempo (fecha y hora) de lectura real. De esta forma se estandarizan las llamadas a los objetos creados para que puedan ser usados por el cliente (Visual Basic).**

8.- Modelo de Objetos de la Especificación OPC DA Automation Wrapper.

► **Por ejemplo, para leer desde VB la entrada *IN00* del *PLC1* es necesario:**

a) configurar el servidor de datos OPC que tenga el driver del modelo del PLC. En este servidor, definiremos el elemento o tag (IN00) del grupo1 del PLC1.

b) definir en el servidor de objetos OPC Automation Wrapper (se hace en Visual Basic) toda la estructura de:
-OPCServer (asociado a nuestro servidor de equipo)
-OPCGroups, OPCGroup, OPCItems (definidos jerárquicamente)
-OPCItem (a quien le asociaremos el elemento o tag que le enviará el servidor OPC)

c) Realizar en Visual Basic un programa que lea el Item deseado.

► **La descripción de cada uno de los objetos del modelo OPC Automation Wrapper anterior, se presenta en la siguiente tabla.**

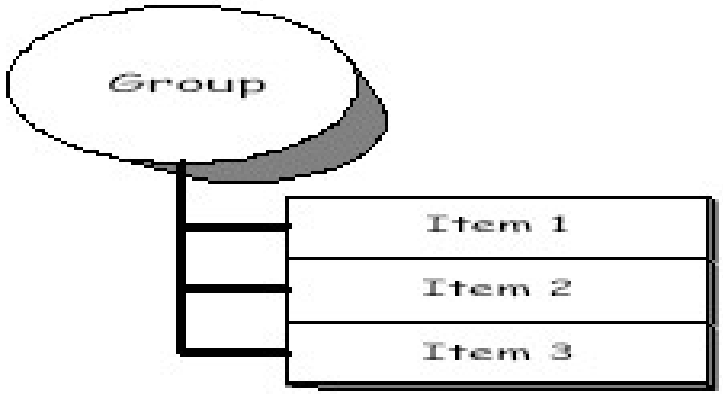
OBJETO	DESCRIPCIÓN
OPCServer	Es una instancia (referencia o solicitud) a un servidor OPC. Mantiene información sobre el servidor escogido y los servicios que ofrece. Se debe crear un objeto OPCServer antes de poder referenciar los otros objetos. Este contiene la colección OPCGroups y el objeto OPCBrowser.
OPCGroups	Es una colección de los objetos OPCGroup que el cliente ha creado.
OPCGroup	El propósito de este objeto es mantener la información de estado y proveer el mecanismo para ofrecer los servicios de adquisición de datos por la colección de objetos OPCItem.
OPCItems	Es una colección que contiene todos los objetos OPCItem que el cliente ha creado.
OPCItem	Es un objeto que mantiene la definición de los items, sus valores, estados y datos de la última actualización. Los Item representan conexiones a las fuentes de datos requeridas del servidor OPC instalado.(NO son fuentes de datos)
OPCBrowser	Es un objeto que permite (hojear) buscar nombres de items en un servidor configurado.

8.- Modelo de Objetos de la Especificación OPC DA Automation Wrapper.

► **El servidor de objetos OPC (OPCServer) ofrece información sobre el servidor y sirve como un contenedor de grupos de objetos OPC. El grupo de objetos OPC (OPCGroups) mantiene información acerca de sí mismo y proporciona los mecanismos para contener y organizar lógicamente los elementos OPC; los grupos OPC (OPCGroup) proporcionan una forma para organizar los datos de los clientes, por ejemplo, el grupo podría representar los elementos en un pantalla particular del operador o a través de un informe; los datos pueden ser leídos y escritos, y las conexiones basadas en excepciones, pueden ser creadas entre el cliente y los elementos en el grupo y pueden ser activadas y desactivadas según sea necesario; un cliente OPC puede configurar qué porcentaje de los datos deben ser cambiados antes de la actualización.**

8.- Modelo de Objetos de la Especificación OPC DA Automation Wrapper.

► **Hay dos tipos de grupos, públicos y locales (o privados); los públicos se realizan para ser compartidos entre varios clientes, mientras que los locales son privados para el cliente en cuestión. Existen interfaces específicas opcionales para los grupos públicos; dentro de cada grupo, el cliente puede definir uno o más elementos OPC, la siguiente imagen ilustra esta relación:**

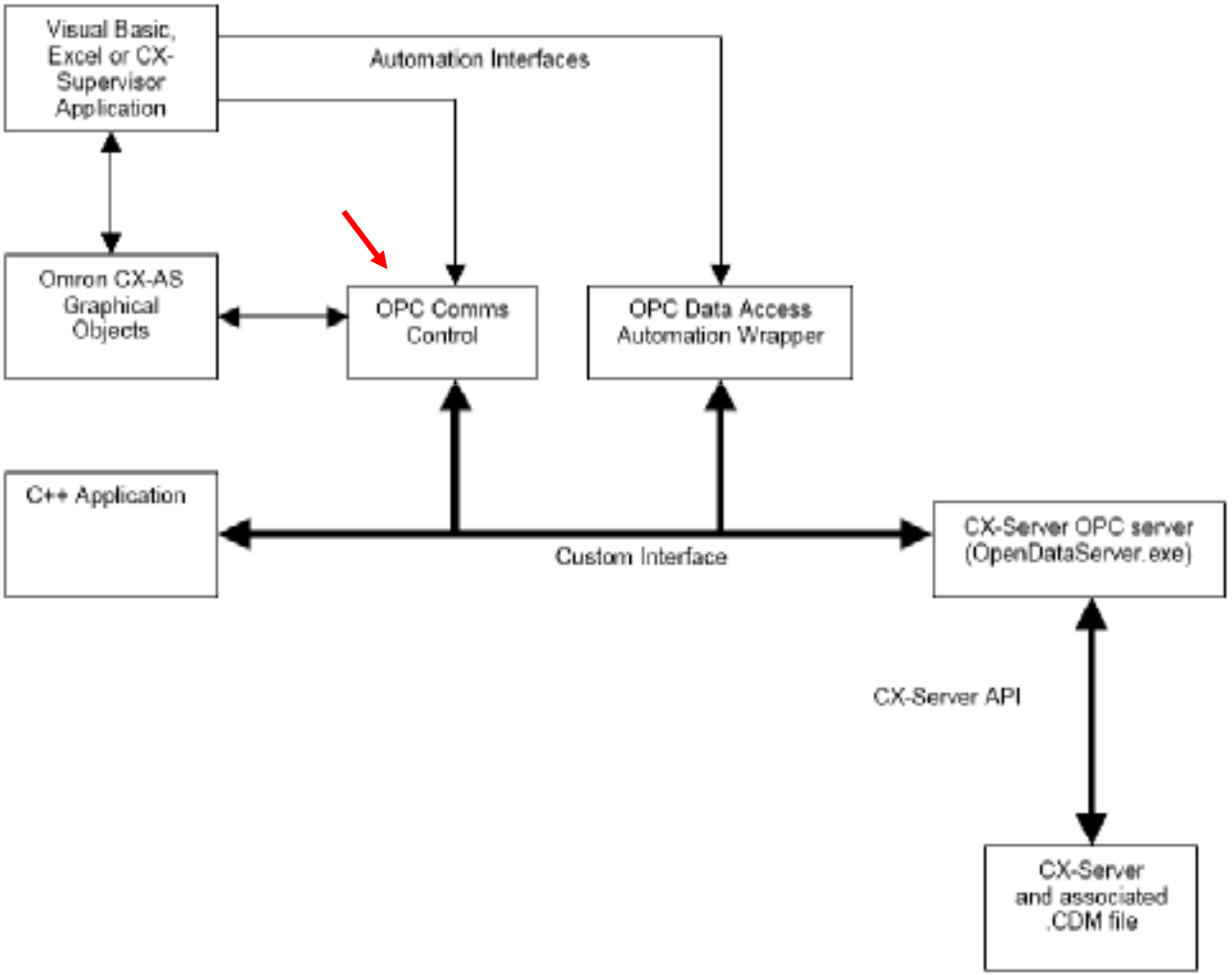


8.- Modelo de Objetos de la Especificación OPC DA Automation Wrapper.

► **Los elementos OPC (Item) representan conexiones a fuentes de datos dentro del servidor ; un elemento OPC, no es accesible por el cliente como un objeto. Así pues, no hay una interfaz externa definida para un elemento OPC; todos los accesos al elemento OPC se realizan a través del objeto grupo OPC que contiene el elemento OPC, o simplemente el grupo en el que el elemento ha sido definido.**

► **Asociado a cada elemento existe un valor, calidad y valor temporal. "Los elementos no son las fuentes de datos, sólo son conexiones a ellas; el elemento OPC debe ser entendido como la dirección de los datos, no como la fuente física actual de los datos a los que la dirección referencia", puesto que la fuente real de los datos es el dispositivo controlador, normalmente un PLC.**

9.- Control de Comunicaciones OPC de Omron.



9.- Control de Comunicaciones OPC de Omron.

► **A la hora de realizar una aplicación cliente, debe decidirse, según sean las necesidades y complejidad de la misma, si el desarrollo será con:**

- **OPC Custom Interface (Aplicaciones grandes y medianas en C++).**
- **OPC DA Automacion Grapper (Aplicaciones grandes y medianas en VB).**
- **Omron OPC Comunication Control (Aplicaciones pequeñas y medianas con o sin VB)**

► **La programación en C++ se sale del propósito de las actividades propuestas ya que está orientada para desarrolladores de software en grandes aplicaciones.**

► **Programar el código en VB de la estructura jerárquica para OPC Automation Wrapper precisa:**

-Rastrear los Servidores instalados en nuestro ordenador.

-Escoger el servidor deseado y establecer la conexión.

-Establecer la jerarquía de grupos e items deseados, creando los grupos y los links a los items que se encuentran en el servidor.

-Utilizar los links creados para acceder (leer, modificar,…) a los items (variables) y establecer así la comunicación.

Pueden verse en los ejemplos de KepServer y de Cx-Server de Omron algunos programas realizados.

► **Omron, como miembro de la fundación OPC, está autorizado para modificar y distribuir este interface. Así, proporciona los objetos COM que comprenden la librería contenida en OmronDAAuto.dll. Esta librería se basa en las especificaciones OPC Data Access Automation v2.05.**

Para muchas aplicaciones es suficiente utilizar Omron OPC Comunication Control. Se trata de evitar el tener que generar toda la estructura jerárquica en código VB. Dispone de objetos que con sencillos menús desplegables, nos permiten definir los servidores, grupos y elementos (Item). También permite, si se desea, programar código en VB con instrucciones de acceso a los datos OPC servidos.

9.- Control de Comunicaciones OPC de Omron.

► **Omron OPC Comunication Control también permite la conexión a servidores OPC locales (por ejemplo un ejecutable (.exe) o una hoja de Excel que funcionan en el mismo ordenador que los servidores OPC) y remotos (los servidores están en ordenadores remotos). En este caso en la red hay que usar DCOM.**

► **En nuestro caso, y para las próximas unidades didácticas, usaremos Omron OPC Comunication Control que se incluye con Cx-Server OPC DEMO de Omron. Al instalar este software, no sólo dispondremos del control de comunicaciones sino que también se incluyen objetos (displays, led, etc.) que se pueden insertar en nuestra aplicación y que fácilmente se configuran para relacionarse con los datos del servidor OPC.**

► **El control de comunicaciones Omron OPC Comunication Control y los objetos mencionados, pueden usarse con servidores OPC de cualquier proveedor.**
Se espera, puesto que OPC es una especificación abierta, que se encuentren disponibles diversas librerías de objetos OPC creadas por distintos programadores para el desarrollo de aplicaciones.

APENDICE 1: Intercambio dinámico de datos DDE

► **Supongamos que hemos copiado a un control TextBox un texto explicativo sobre la versión del programa que hemos escrito con un procesador de textos. Si posteriormente modificamos el texto original, la copia que hemos realizado no reflejará los cambios a menos que volvamos a efectuar la operación de copiar y pegar de nuevo. Además deberemos repetir esta operación cada vez que se modifiquen los datos originales. A este modo de operar se le denomina intercambio estático de datos para reflejar que los datos pegados son constantes y corresponde al usuario actualizarlos cuando sea necesario.**

► **Windows permite establecer otro tipo de intercambio en el cual toman más protagonismo las aplicaciones restándoselo al usuario. Distinguiremos siempre entre estas dos aplicaciones:**

1. La aplicación origen de datos: es la que suministra los datos que van a ser copiados. En el ejemplo anterior sería el proceso de textos que hemos usado para introducir el texto explicativo.

2. La aplicación destino de los datos: es la que recibe los datos que van a ser pegados. En el ejemplo anterior sería nuestra aplicación Visual Basic que recibe el texto en un control TextBox.

APENDICE 1: Intercambio dinámico de datos DDE

► **En la operación conocida como DDE (Dynamic Data Exchange: intercambio dinámico de datos) la aplicación origen envía los datos a la aplicación destino de forma automática sin que el usuario haya de preocuparse de efectuar las operaciones de copiar y pegar.**

Siguiendo el ejemplo anterior, para actualizar el contenido del TextBox se seguirían los siguientes pasos:

1. Nuestro programa Visual Basic establecería una conexión DDE con la aplicación origen de datos: el proceso de textos. Si esta aplicación estuviera inactiva se generaría un error en tiempo de ejecución. A este enlace automático entre aplicaciones se le denomina conversación DDE.

2. Una vez establecida la conversación DDE el programa origen de datos pasa la información al destino de datos: el control TextBox.

3. Al terminar la transferencia, se cierra la conversación y se liberan los recursos asignados.

► **En definitiva, el intercambio DDE permite tanto enviar como recibir datos e instrucciones entre aplicaciones bajo Windows.**

APENDICE 1: Intercambio dinámico de datos DDE

► **El protocolo DDE está basado en un sistema de mensajería construido por Windows.**

Así, dos programas de aplicación bajo Windows realizan una "conversación DDE." enviándose mensajes entre ellos.

Una conversación DDE se inicia con el programa que actúa como diente. Este transfiere un mensaje a todos los programas que se están ejecutando en ese momento en Windows. Dicho mensaje indica una categoría general de datos que el cliente necesita.

Un servidor DDE que posee dichos datos puede responder a este mensaje. En este instante comienza la conversación. Un programa puede ser cliente de otro programa, y servidor para otro, pero esto requiere dos conversaciones- DDE distintas. Un servidor puede entregar datos a varios clientes y un cliente puede obtener datos desde múltiples servidores, pero esto requiere varias comunicaciones DDE.

Un programa implicado en una comunicación DDE no necesita codificarse específicamente para trabajar con otro programa DDE. Generalmente el diseñador de un servidor DDE hace público cómo se identifican los datos. Como DDE utiliza el sistema de mensajería incluido en Windows, el programa se integra perfectamente en este entorno.

APENDICE 1: Intercambio dinámico de datos DDE

► **En DDE ambas aplicaciones deben estar ejecutándose y las dos deben dar a Windows una dirección a sus funciones de llamada antes de que la comunicación de DDE pueda comenzar. La función de llamada acepta cualquier mensaje de DDE que Windows envía a la aplicación.**

Un cliente de DDE comienza una conversación con otra aplicación (un servidor de DDE) enviando un mensaje de conexión. Después de establecer una conexión, el cliente puede enviar órdenes o datos al servidor y a cambio puede pedir el valor de datos que el servidor maneja.

Cuando la comunicación DDE para una conversación es completada, el cliente envía un mensaje de cerrar la conversación al servidor.

Un cliente estándar DDE soporta cinco operaciones básicas:

Abrir un enlace, Enviar comandos, Leer un elemento de datos, Enviar un elemento de datos y cerrar el enlace.

Las diferentes aplicaciones clientes pueden tener nombres diferentes para estas funciones. Consultar en sus manuales y ayudas.

APENDICE 1: Intercambio dinámico de datos DDE

► **El enlace DDE se configura a través de una serie de propiedades de los controles enlazables cuyo nombre empieza por "Link", como por ejemplo LinkItem y LinkTopic que proporcionan el nombre de la aplicación origen de datos y la localización de los datos, y se gestiona a través de métodos de los controles, Como por ejemplo LinkRequest para iniciar una conversación. Históricamente DDE es el tipo de enlace automático más antiguo, presente desde las primeras versiones de Windows. Desde la versión 3.1 del sistema operativo ya quedó superado por el nuevo estándar de intercambio de información OLE a partir del cual se desarrolló la moderna tecnología ActiveX.**

APENDICE 2: Enlace e inserción de objetos OLE

► **Desde la perspectiva de OLE las aplicaciones no son un bloque indivisible y homogéneo sino que por el contrario están formadas por componentes que se comunican unos con otros transfiriéndose información. Algunos de estos componentes pueden ser públicos en el sentido de ser utilizables por cualquier aplicación. Supongamos por ejemplo que estamos programando una aplicación Visual Basic que gestiona datos financieros y en un determinado momento necesitamos representar gráficamente la evolución de un capital invertido. Sabemos que un programa de hoja de cálculo como por ejemplo Microsoft Excel tiene funciones gráficas incorporadas. Pues bien, la idea sería utilizar en nuestro programa la parte de código de Excel encargada de las representaciones gráficas. Esto sólo será posible si Excel ofrece como público ese código. Este concepto que acabamos de definir se conoce con el nombre de automatización OLE: la posibilidad de utilizar en nuestro código objetos programables definidos por otra aplicación.**

OLE, Object Linked and Embedded (objetos enlazados e incrustados) o (vinculados e incluidos) fue introducido en 1991 como una extensión del protocolo DDE. En una aplicación es posible incluir (incrustar) objetos que quedearán totalmente contenidos en ella. Los objetos vinculados (enlazados) tienen una conexión en la aplicación cliente y solo son accesibles a través de la aplicación que contiene los datos originales.

APENDICE 2: Enlace e inserción de objetos OLE

► **Señalemos claramente las ventajas de la definición OLE :**

1. Podemos intercambiar información tal como permitía hacer la tecnología DDE.

2. También podemos compartir código a través de objetos públicos definidos por cualquier aplicación que use el estándar OLE.

3. Para utilizar los recursos de otra aplicación (datos o código) no siendo necesario que esa aplicación esté activa, cosa que sí ocurría con el enlace DDE.

4. Con DDE, para editar los datos proporcionados por otra aplicación se le pasaba el control de la ejecución quedando nuestro programa en segundo plano. Con OLE nuestro programa siempre está en primer plano controlando la ejecución. Esto nos da un mayor control de la situación además de ser más eficiente en la ejecución

Las aplicaciones que proporcionan objetos OLE públicos se denominan servidores OLE y las que los utilizan clientes OLE. Una aplicación visual Basic puede actuar como cliente o como servidor.

Sabemos que un objeto contiene propiedades (datos) y métodos (código) y que un cliente OLE puede utilizar cualquiera de los dos. Cuando un cliente muestra los datos proporcionados por un objeto OLE se dice que actúa como un contenedor OLE.

APENDICE 2: Enlace e inserción de objetos OLE

► **Hay dos formas de mostrar esos datos: datos insertados (también se suelen llamar incrustados o embebidos) o bien datos enlazados (o vinculados). De ahí las siglas OLE: Object Linked and Embedded (objetos enlazados y incrustados). Al insertar datos realizamos una copia de los originales en nuestra aplicación que queda desligada del original. Por ejemplo si insertamos un gráfico de Excel en un formulario Visual Basic y después modificamos el original la copia que mantiene nuestro programa no quedará afectada por los cambios. Si queremos editar el gráfico insertado llamaremos a Excel para hacerlo pero una vez guardados los cambios el gráfico quedará inaccesible para cualquier aplicación que no sea la nuestra.**

Con los datos enlazados en cambio nuestra aplicación no tiene ninguna copia independiente sino que guarda una referencia al objeto OLE que queda como externo a nuestra aplicación. Cuando lo editamos también se llama a la aplicación servidora.

En el ejemplo del gráfico de Excel si estuviera enlazado sería un objeto independiente que por tanto podría ser modificado no solo por nuestra aplicación sino por el propio Excel o cualquier otra aplicación a través de un enlace OLE.

APENDICE 3: Introducción al COM y DCOM.

► **Para poder compatibilizar entre sí objetos implementados en diferentes plataformas o arquitecturas de ordenadores es necesario definir la forma en que dichas plataformas interpretan un objeto. Para ello se requiere el denominado modelo del objeto.**
OLE utiliza el modelo COM (Component Object Model). Éste define el estándar para la interrelación de los componentes. COM permite llamadas dentro de un proceso, llamadas a otro proceso e incluso llamadas a otro ordenador.

► **Microsoft describe DCOM como "Distributed Component Object Model" (DCOM) es un protocolo que permite a los Componentes de software comunicarse directamente sobre una red de un modo seguro, y eficiente. Previamente llamado "Network OLE", DCOM es diseñado para el uso a través múltiples medios de red, incluyendo los protocolos internet como el HTTP. DCOM está basado en las especificaciones DCE - RPC de la Fundación de Software Abierto y puede funcionar con applets de Java y componentes de ActiveX® mediante el uso del Component Object Model (COM)."**

APENDICE 3: Introducción al COM y DCOM.

► **En otras palabras, DCOM es un modelo de objeto de programación para la puesta en funcionamiento de aplicaciones distribuidas que usan un patrón cliente-servidor. Un cliente puede usar varios servidores al mismo tiempo y un servidor puede suministrar la funcionalidad a clientes múltiples simultáneamente.**

► **Traduciendo eso a lenguaje coloquial, COM básicamente permite que componentes de software sean escritos de modo que puedan ser usados por aplicaciones de COM-aware (por ejemplo. C++, últimas versiones del Visual Basic) sin que estas aplicaciones necesiten tener conocimiento sobre los "Interiores" del objeto. DCOM es sólo la versión distribuida de COM. Los objetivos pueden ser difundidos a través de una red. Es un sistema muy potente, aunque sea necesaria la configuración a nivel de equipo y de seguridad para permitir que trabaje correctamente y fiablemente.**

10.-Bibliografía.

► **Se adjuntan ficheros con información OPC que en su mayoría se han obtenido de páginas de internet.**

- **Après DA, voici DX.pdf**
- **Cliente OPC realizado en Visual Basic.doc**
- **cours_dde.pdf**
- **cours_opc.pdf**
- **Cours_Supervision_uniLille.pdf**
- **Dcom_Configuration.pdf**
- **Excel_Super.pdf**
- **Fundación OPC.pdf**
- **leame - SOBRE OPC.doc**
- **OPC Chile.ppt**
- **OPC Siemens 2004.pdf**
- **opc web ehu.pdf**
- **OPC_Server_for_DDE_Manual.pdf**
- **opc_tutorial_printable_version.pdf**
- **OPC-DCOM contra XML.pdf**
- **opcguide.pdf**
- **TranspasCurso.ppt**
- **UsingVisualBasicwithOPC[1].ppt**

Los programas de KepServer (Kepware) y Cx-ServerOPC (Omron) contienen ayudas importantes con conceptos y ejemplos..