ARQUITECTURA PARA LA INTEGRACIÓN DE PLATAFORMAS DE CONTROL BASADAS EN PC MEDIANTE OPC

ANEXOS

AMPARO DIAZ MUÑOZ

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Electrónica, Instrumentación y Control
Ingeniería en Automática Industrial

Popayán, Julio de 2011

ANEXOS

LISTA DE ILUSTRACIONES

Lista de tablas

ANEXO A. PLANTILLAS Y DIAGRAMAS PARA CASOS DE USO Y REQUERIMIENTOS BÁSICOS DE LA ARQUITECTURA

I. PLANTILLAS PARA LOS CASOS DE USO

A continuación se presentan las plantillas para la especificación de los casos de uso para: el módulo cliente OPC XML en Matlab (MCOXM), módulo Cliente OPC DCOM en Rtai-Lab (MCODR), módulo cliente OPC XML en Matlab (MCOXR) y el módulo de Monitoreo y supervisión Web (MM&SW). Cada plantilla contiene los campos estándar para los casos de uso. En ella se describen los actores, el propósito, los requisitos asociados, las precondiciones, la secuencia entre actores, entradas, salidas, estabilidad y comentarios para cada caso de uso.

La tabla A-1 presenta la plantilla para el módulo cliente OPC XML en Matlab, con la información obtenida en esta plantilla se diseña el diagrama caso de uso y los requerimientos básicos para el módulo MCOXM.

La tabla A-2 presenta la plantilla para el módulo cliente OPC DCOM en Rtai-Lab, con la información obtenida en esta plantilla se diseñó el diagrama caso de uso y los requerimientos básicos para el módulo MCODR.

La tabla A-3 presenta la plantilla para el módulo cliente OPC XML en Rtai-Lab, con la información obtenida en esta plantilla se diseñó el diagrama caso de uso y los requerimientos básicos para el módulo MCOXR.

La tabla A-4 presenta la plantilla para el módulo Servidor OPC XML, con la información obtenida en esta plantilla se diseñó el diagrama caso de uso y los requerimientos básicos para el módulo MSOX

La tabla A-5 presenta la plantilla para el módulo de Monitoreo y supervisión Web, con la información obtenida en esta plantilla se diseñó el diagrama caso de uso y los requerimientos básicos para el módulo MM&SW

Tabla A-1: Plantilla para el caso de uso del Módulo cliente OPC XML en Matlab

CASO DE USO 1	**Módulo Cliente OPC XML en Matlab**	
Actores	Operario, servidor OPC XML	
Propósito	Leer datos provenientes de Matlab y/o *Simulink* y enviarlos al Servidor OPC XML.	
Requisitos asociados	➢ Permitir la comunicación entre Matlab y uno o varios Servidores OPC XML. ➢ Tomar datos definidos por el usuario o por el entorno Matlab y enviarlos al Servidor OPC XML luego de iniciado el cliente. ➢ Recibir automáticamente los datos de notificaciones de cambio de cada uno de los ítems OPC en el servidor XML y visualizar su valor actual.	
Precondiciones	El servidor OPC XML debe tener configurados los objetos ítems	
Secuencia entre los actores y el MCOXM	**Acción de los actores**	**Respuesta del Módulo MCOXM**
	1. Este caso de uso comienza cuando un usuario ejecuta el cliente OPC XML en Matlab.	2. Se presenta un formulario con un menú a) inicializar cliente b) importar variables c) leer variables d) escribir variables e) finalizar conexión.
	Iniciar cliente: **3** El usuario selecciona inicializar cliente	4. Se presenta un formulario, para que el usuario diligencie la dirección del host y el servidor.
	5. El usuario registra la IP del host y el nombre del servidor y presiona ok	6. El sistema envía solicitud de conexión al servidor OPC XML ubicado en el equipo remoto.
	7. El servidor OPC XML acepta la solicitud de comunicación.	8. Se crea el cliente OPC XML
	Caso Importar variables 9. El usuario selecciona del menú principal importar variables.	10. Se presenta un formulario con todos los ítems, para que el usuario pueda seleccionarlos e importarlos.
	11. El usuario selecciona las variables de lectura y escritura y presiona ok.	12. Se almacenan las variables para la lectura y escritura.
	Caso Leer variables 13**.** El usuario selecciona leer variables.	14. Se solicita al servidor OPC XML las propiedades los ítems de lectura.
	15. El servidor OPC retorna las propiedades de los ítems.	16. Se presenta al usuario los ítems de lectura.
	Caso Escribir variables 17. El usuario o el entorno Matlab selecciona la opción escribir variables.	18. Se presenta un formulario de donde el usuario puede escribir el valor del ítem OPC.
	19. El usuario escribe el valor de cada ítem.	20. Se toma el valor que el usuario ha especificado y se envía al servidor.
	21. El usuario finaliza el cliente	22. El sistema finaliza la comunicación con el servidor OPC XML.
Entradas	Dirección del servidor, Nombre de servidor, Dirección de los objetos ítems, Nombre de los objetos ítems, Valor de los objetos ítems, Tipo de objetos ítems	
Salidas	Valor de los objetos ítems	
Estabilidad	Alta	
Comentarios	Ninguno	

Fuente: Propia

Tabla A-2: Plantilla para el caso de uso Módulo cliente OPC DCOM en Rtai-Lab

CASO DE USO 2	**Módulo Cliente OPC DCOM en Rtai-Lab**	
Actores	Operario, servidor OPC DCOM	
Propósito	Leer datos provenientes de Rtai-Lab y enviarlos al Servidor OPC DCOM	
Requisitos asociados	➢ Permitir la comunicación entre Rtai-Lab y uno o varios Servidores OPC DCOM. ➢ Recibir automáticamente los datos de notificaciones de cambio de cada uno de los ítems OPC y visualizar su valor actual. ➢ Tomar datos definidos por el usuario o por el entorno *Scicos* / *Scilab* y enviarlos al Servidor OPC DCOM luego de iniciado el cliente.	
Precondiciones	El servidor OPC DCOM debe tener configurados los objetos ítems	
Secuencia entre los actores y el MCOXM	**Acción de los actores**	**Respuesta del Módulo MCODR**
	1. Este caso de uso comienza cuando un usuario selecciona en *Scicos* la paleta OPC DCOM.	2. Se presentan dos bloques OPC DCOM a) bloque de lectura b) bloque de escritura.
	Caso bloque Lectura: 3. El usuario selecciona el bloque de lectura.	4. Se presenta un bloque que permitirá establecer la dirección del host, la dirección del servidor OPC DCOM, número de ítems de lectura y el nombre de los respectivos elementos del servidor.
	5. El usuario configura el bloque de lectura.	6. Las salidas del bloque se modifican según el número de ítems que el usuario establece.
	Caso bloque Escritura: 7. El usuario selecciona el bloque de escritura	8. Se presenta un bloque que permite establecer la dirección del servidor OPC DCOM, número de ítems de escritura y el nombre de los respectivos ítems del servidor.
	9. El usuario configura el bloque de escritura.	10. Las entradas del bloque se modifican según el número de ítems que el usuario establece.
	11. El usuario conecta las entradas del bloque de lectura y envía la orden para que inicie la comunicación OPC.	12. Se realiza la solicitud de comunicación con el servidor OPC DCOM.
	13. El servidor OPC DCOM envía respuesta de comunicación.	14. Comienza la lectura y escritura de las propiedades de los ítems, y se espera una orden de finalización.
	15. El usuario envía orden de finalización de la comunicación OPC DCOM.	16. Se Finaliza la Comunicación OPC.
Entradas	Dirección del servidor OPC DCOM, Nombre de servidor, Dirección de los objetos ítems, Nombre de los objetos ítems, Valor de los objetos ítems, Tipo de objetos ítems	
Salidas	Valor de los objetos ítems	
Estabilidad	Alta	
Comentarios	Ninguno	

Fuente: Propia

Tabla A-3: Plantilla para el caso de uso del Módulo cliente OPC XML en Rtai-Lab

CASO DE USO 3	**Módulo Cliente OPC XML en Rtai-Lab**	
Actores	Operario, servidor OPC XML	
Propósito	Leer datos provenientes de Rtai-Lab y enviarlos al Servidor OPC XML	
Requisitos asociados	➢ Permitir la comunicación entre Rtai-Lab y uno o varios Servidores OPC XML. ➢ Recibir automáticamente los datos de notificaciones de cambio de cada uno de los ítems OPC y visualizar su valor actual. ➢ Tomar datos definidos por el usuario o por el entorno *Seicos*/Scilab y enviarlos al Servidor OPC XML luego de iniciado el cliente.	
Precondiciones	El servidor OPC XML debe tener configurados los objetos ítems	
Secuencia entre los actores y el MCOXM	**Acción de los actores**	**Respuesta del Módulo MCOXM**
	1. Este caso de uso comienza cuando un usuario selecciona en Scicos la paleta OPC XML.	2. Se presentan dos bloques OPC XML a) bloque de lectura b) bloque de escritura.
	Caso bloque lectura 3. El usuario selecciona el bloque de lectura	4. Se presenta un bloque que permitirá establecer la dirección del host, la dirección del servidor OPC XML, número de ítems de lectura y el nombre de los respectivos elementos del servidor.
	5. El usuario configura el bloque de lectura.	6. Las salidas del bloque se modifican según el número de ítems que el usuario establece.
	Caso bloque escritura 7. El usuario selecciona el bloque de escritura	8. Se presenta un bloque que permite establecer la dirección del servidor OPC XML, número de ítems de escritura y el nombre de los respectivos ítems del servidor.
	9. El usuario configura el bloque de lectura	10. Las entradas del bloque se modifican según el número de ítems que el usuario establece.
	11. El usuario conecta las entradas del bloque de lectura, e inicia el proceso de comunicación.	12. Se realiza la solicitud de comunicación con el servidor OPC XML.
	13. El servidor OPC XML envía respuesta de comunicación.	14. Comienza la lectura y escritura de las propiedades de los ítems, y se espera una orden de finalización.
	15. El usuario envía orden de finalización de la comunicación OPC XML.	16. Finaliza la comunicación con el servidor OPC XML.
Entradas	Dirección del servidor OPC XML, Nombre de servidor, Dirección de los objetos ítems, Nombre de los objetos ítems, Valor de los objetos ítems, Tipo de objetos ítems	
Salidas	Valor de los objetos ítems	
Estabilidad	Alta	
Comentarios	Ninguno	

Fuente: Propia

Tabla A-4: Plantilla para el caso de uso del Módulo Servidor OPC XML

CASO DE USO 4	**Módulo Servidor OPC XML**	
Actores	Operario, Cliente OPC XML	
Propósito	Traducir los datos provenientes de las fuentes de datos (Matlab y Rtai-Lab) para que sea compatible con la especificación OPC XML DA.	
Requisitos asociados	➢ Configuración de parámetros del servidor: se debe permitir al usuario configurar los parámetros de inicio del servidor. ➢ Configurar ítems: el usuario debe poder crear, modificar y eliminar cada uno de ítems del servidor. ➢ Lectura y escritura dinámica de objetos OPC XML: después de inicializado el servidor se debe permitir la lectura de datos desde fuentes consumidores y la escritura de ítems desde las plataformas de control. ➢ Estado del servidor: se debe permitir a los usuarios visualizar el estado del servidor.	
Precondiciones	Ninguna	
Secuencia entre los actores y el MCOXM	**Acción de los actores**	**Respuesta del Módulo MCOXM**
	1. El operario inicia el módulo servidor OPC XML	2. presenta un formulario donde el usuario puede configurar los objetos del servidor.
	3. El operario crea los elementos y establece las propiedades.	4. Almacenar las propiedades de los objetos OPC XML
	5. El cliente OPC XML solicita el inicio de la comunicación	6. Aceptar solicitud de conexión con los clientes OPC XML.
	7. El cliente OPC XML solicita el estado de los ítems.	8. El servidor retorna las propiedades de los ítems
	9. El cliente OPC XML escribe las propiedades de los ítems.	10. El servidor captura los datos del cliente OPC XML.
	11. El cliente OPC XML finaliza la comunicación.	12. El servidor envía mensaje de finalización de la comunicación.
Entradas	Valor de los objetos ítems	
Salidas	Dirección del servidor, Nombre de servidor, Dirección de los objetos ítems, Nombre de los objetos ítems, Valor de los objetos ítems, Tipo de objetos ítems	
Estabilidad	Alta	
Comentarios	Ninguno	

Fuente: Propia

Tabla A-5: Plantilla para el caso de uso del Módulo de Monitoreo y supervisión Web

CASO DE USO 5	**Módulo de Monitoreo y supervisión Web**	
Actores	Operario, Administrador del sitio Web, servidor OPC XML, Servidor OPC DCOM.	
Propósito	Rrepresentar gráficamente la información proveniente de los Servidores OPC DCOM/XML	
Requisitos asociados	➢ Posibilitar la actualización, mantenimiento y ampliación de la web con la colaboración de múltiples usuarios mediante un gestor de contenido. ➢ Permitir adquirir un conjunto de datos provenientes de los Servidores OPC DCOM y XML. ➢ Permitir el diseño de las pantallas a visualizar tomando información de los objeto ítems de los servidores OPC, manejando las alarmas, eventos y tendencias. ➢ Alertar al operador través de las señales de alarma frente a una falla o la presencia de una condición perjudicial o fuera de lo aceptable. ➢ Brindar imágenes en movimiento que representen el comportamiento del proceso, dándole al operador la impresión de estar presente dentro de una planta real. ➢ Representar variables mediante curvas de señales en el tiempo.	
Precondiciones	Los servidores OPC XML/DCOM deben tener configurados los objetos ítems	
Secuencia entre los actores y el MCOXM	**Acción de los actores**	**Respuesta del MM&SW**
	1. Este caso de uso comienza cuando el usuario ingresa el nombre de la página HMI en un explorador.	2. Se presenta la página de inicio, que brinda al usuario la posibilidad de identificación, actualización, mantenimiento y ampliación de la web mediante un gestor de contenido.
	3. El usuario se identifica	4. Presenta un menú principal, con un enlace Sistema HMI.
	5. El usuario da clic el menú Sistema HMI.	6. Presenta un nuevo menú de a) diseño, b) monitoreo.
	Opción diseño de procesos 7. El usuario selecciona diseño	8. Se presenta un menú: importar variables OPC, diseño de pantallas, configuración de alarmas, configuración eventos, Ayuda.
	9. El usuario diseña el sistema HMI y da clic en guardar	10. Se almacenan los datos del diseño
	Opción monitoreo 11. El usuario selecciona la opción monitoreo.	12. Se crean archivos asociados al sistema HMI, y se presenta un menú con las opciones: procesos, alarmas, eventos, Ayuda.
	13. El usuario interactúa con el sistema de monitoreo.	14. El sistema recoge datos de los servidores OPC y los refleja al usuario.
	15. El usuario Finaliza el monitoreo	16. Se eliminan los archivos asociados al monitoreo.
Entradas	Dirección del servidor OPC XML, Nombre de servidor OPC DCOM, Dirección de los objetos ítems, Nombre de los objetos ítems, Valor de los objetos ítems, Tipo de objetos ítems	
Salidas	Valor de los objetos ítems	

Fuente: Propia

II. DIAGRAMAS CASOS DE USO

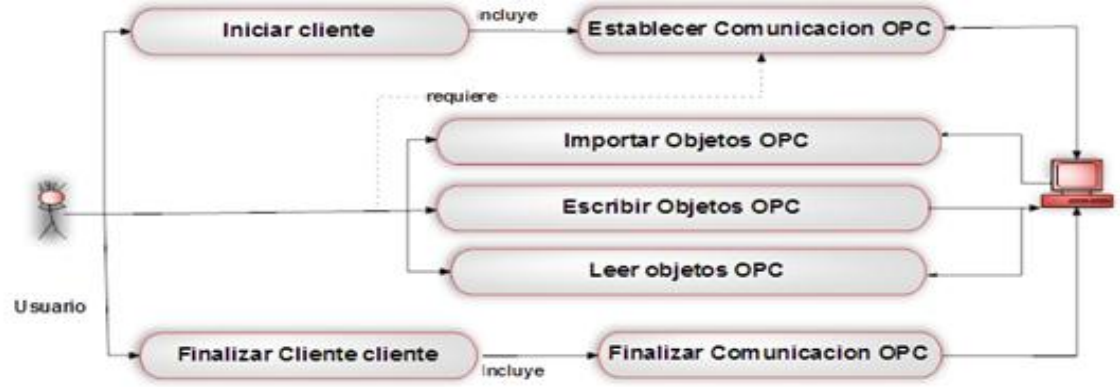
A continuación se detallan los diagramas casos de uso, los cuales se obtuvieron a partir de las plantillas de casos de uso expuestas en la sección I de este anexo.

Caso 1: Módulo cliente OPC XML en Matlab (MCOXM)

La figura A-1 presenta el diagrama caso de uso para el módulo cliente OPC XML en Matlab, el caso comienza cuando el operario ingresa a Matlab y ejecuta el cliente OPC XML, establece el nombre del servidor OPC XML, importa cada uno de los objetos ítems requeridos existentes en el servidor OPC XML, inicia la lectura/escritura activa de datos, y finaliza la comunicación entre el cliente y el servidor OPC XML.

- ***Propósito****:* permitir la comunicación entre el usuario, la plataforma Matlab y el servidor OPC XML.
- ***Actores****:* dentro de los actores involucrados en este caso de uso están: el operario que se encarga de realizar la comunicación OPC XML, y el servidor OPC XML quien se encarga de recibir los datos y convertirlos en formato OPC.
- ***Precondiciones****:* el servidor OPC XML debe estar inicializado y configurado los objetos ítems.
- ***Entradas:*** dirección del host donde está instalado el servidor OPC XML, dirección del servidor OPC XML, dirección y valores de los objetos ítems.

Ilustración A-1: Diagrama caso de uso para el MCOXM



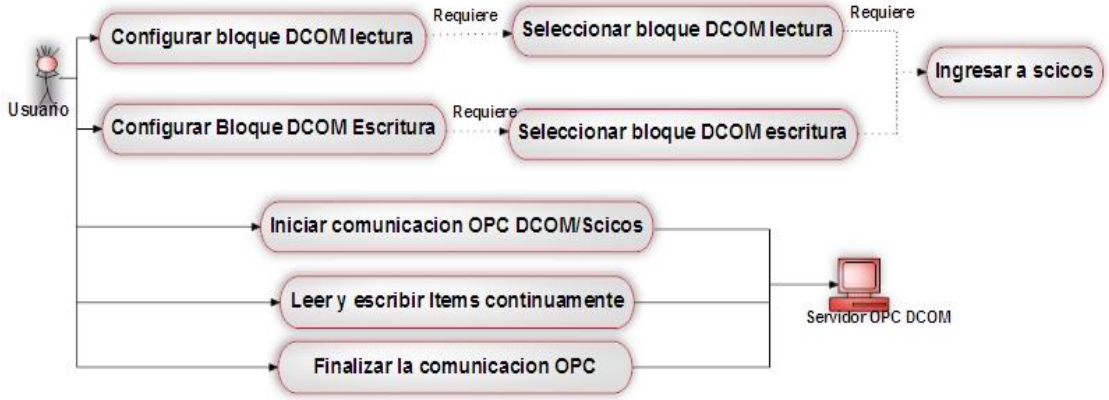
Fuente: propia

Caso 2: Módulo cliente OPC DCOM en Rtai-Lab (MCODR)

La figura A-2 presenta el diagrama caso de uso para el módulo cliente OPC DCOM en Rtai-Lab, el caso comienza cuando el usuario ingresa a Scicos, selecciona entre los bloques de escritura o lectura en servidores OPC DCOM, configura los parámetros e interconecta los bloques, y ejecuta la aplicación.

- ***Propósito:*** permitir la comunicación entre la plataforma Rtai-Lab y el servidor OPC DCOM.
- ***Actores:*** el operario que se encarga de establecer parámetros y el servidor OPC DCOM quien se encarga de recibir los datos y convertirlos en formato OPC.
- ***Entradas:*** dirección del host, dirección del servidor OPC DCOM, dirección y valores de los objetos ítems.
- ***Precondiciones:*** el servidor OPC DCOM debe estar inicializado y los objetos ítems deben estar configurados.

Ilustración A-2: Diagrama caso de uso para el MCODR



Fuente: propia

Caso 3: Módulo cliente OPC XML en Rtai-Lab (MCOXR)

La figura A-3 presenta el diagrama caso de uso para el módulo cliente OPC XML en Rtai-Lab, el caso comienza cuando el usuario ingresa a Scicos, selecciona entre los bloques de escritura o lectura en servidores OPC DCOM, configura los parámetros e interconecta los bloques, y ejecuta la aplicación.

- ***Propósito:*** permitir la comunicación entre la plataforma Rtai-Lab y el servidor OPC XML.
- ***Actores:*** el operario que se encarga de establecer parámetros y el servidor OPC XML quien se encarga de recibir los datos y convertirlos en formato OPC.
- ***Entradas:*** dirección del host, dirección del servidor OPC XML, dirección y valores de los objetos ítems.
- ***Precondiciones:*** el servidor OPC XML debe estar inicializado y los objetos ítems deben estar configurados.

Ilustración A-3: Diagrama caso de uso para el MCOXR

Fuente: propia

Caso 4: Módulo servidor OPC XML (MSOX)

La figura A-4 representa el diagrama caso de uso para el módulo servidor OPC XML, el caso comienza cuando el usuario inicializa el servidor, y configura los parámetros de los objetos ítems. El servidor está atento a las conexiones con los clientes, y simultáneamente envía y recibe información hacia/desde los clientes OPC XML. A continuación se presentan algunas propiedades del caso de uso MSOX.

- ***Propósito:*** traducir los datos provenientes de (Matlab y Rtai-Lab) para que sea compatible con la especificación OPC XML y además pueda ser visualizada por cualquier cliente que presente compatibilidad con este estándar.
- ***Actores:*** dentro de los actores involucrados en este caso de uso están: el operario y los clientes OPC XML.

- ***Entradas:*** parámetros del servidor, parámetros de los objetos ítems.

Ilustración A-4: Asistente Diagrama caso de uso para el MSOX

Fuente: propia

Caso 5: Módulo sistema de monitoreo y supervisión web (MM&SW)

La figura A-5 representa el diagrama caso de uso para el módulo de monitoreo y supervisión web, el caso comienza cuando el usuario ingresa una URL en un explorador web, donde mediante un módulo gestor de contenido se puede configurar las propiedades de la página, además el usuario puede registrarse, y cuando este correctamente autenticado se presenta el menú de diseño de proyectos, donde se puede crear pantallas HMI, configurar las variables asociadas a los servidores OPC DCOM y XML, configurar alarmas, eventos y la ayuda, otra opción que se presenta es el de monitoreo de datos, donde el usuario interactúa con los procesos de planta mediante los servidores OPC.

A continuación se presentan algunas propiedades del caso de uso del módulo de Monitoreo y supervisión web.

- ***Nombre:*** módulo de monitoreo y supervisión web.
- ***Actores:*** administrador, operario, servidores OPC DCOM, XML.
- ***Propósito:*** capturar las variables de los procesos de Matlab y Rtai-Lab mediante la comunicación con los servidores OPC DCOM y XML y representarlos gráficamente.

- ***Precondiciones:*** los servidores OPC XML y DCOM deben estar inicializados.

Ilustración A-5: Diagrama caso de uso para el MM&SW

Fuente: propia

III. REQUERIMIENTOS BASICOS DE LA ARQUITECTURA PROPUESTA

Con base a la información obtenida en las tablas A-1, A-2, A-3, A-4 y A-5 de la sección I de este anexo, se establecieron los requerimientos básicos para cada uno de los módulos los cuales se listan en las tablas A-6, A-7, A-8, A-9, A-10.

En cada una de las tablas se especifica el identificador y la descripción de los requerimientos básicos, como también los sub-módulos en los que se subdivide cada módulo de la arquitectura para la integración de plataformas de control basados en PC mediante OPC.

Tabla A-6: Requerimientos básicos para el Módulo cliente OPC XML en Matlab

ID	DESCRIPCION REQUERIMIENTO
RBMCOXM1	Se debe presentar un formulario con un menú con los siguientes elementos: inicializar cliente, importar ítems, leer ítems, escribir ítems, finalizar cliente.
	Sub-módulo inicializar cliente
RBMCOXM2	Se debe permitir al usuario ingresar la dirección del servidor OPC XML.
RBMCOXM3	Se debe tomar la dirección que ingresa el usuario y conectarse con el servidor OPC XML.
RBMCOXM4	Se debe indicar al usuario el estado del servidor OPC XML mediante una etiqueta.
	Sub-módulo importar variables
RBMCOXM5	Se debe permitir al usuario seleccionar cuales son los ítems con los que se va a trabajar.
RBMCOXM6	Se debe almacenar los ítems seleccionados por el usuario.
	leer variables
RBMCOXM7	Leer ítems constante mente: se debe solicitar al servidor, las propiedades de los ítems y reformarlos al usuario mediante un formulario.
	Sub-módulo escribir variables
RBMCOXM7	Se debe presentar un formulario donde el usuario ingrese el nombre de la variable y el valor.
RBMCOXM8	Se debe enviar al servidor la información con las propiedades de los ítems.
	Sub-módulo Finalizar servidor.
RBMCOXM9	Se debe enviar al servidor la orden de finalizar la comunicación.

Fuente: propia

Tabla A-7: Requerimientos básicos del módulo cliente OPC DCOM en Rtai-Lab

ID	DESCRIPCION REQUERIMIENTO
RBMCODR1	Se debe permitir al usuario seleccionar entre dos bloques, uno para la lectura y otro para la escritura.
	Bloque de lectura
RBMCODR2	Se debe permitir al usuario configurar los parámetros del bloque de lectura.
RBMCODR3	Se debe permitir conectar las salidas del bloque, para que la información proveniente de este se pueda visualizar.
	Bloque de escritura
RBMCODR4	El bloque de escritura debe capturar los valores provenientes de los servidores OPC DCOM. Cada valor de la salida debe tener asociado un valor proveniente de cada ítem del servidor.
RBMCODR5	Se debe permitir enviar datos a los ítems mediante el bloque de lectura, los usuarios deben poder asignar valores y especificar a qué elementos del servidor deben ser enviados.
	Inicio de la comunicación entre Rtai-Lab y el servidor OPC DCOM
RBMCODR6	Se debe iniciar la comunicación cuando el usuario lo solicite, en esta parte se debe estar leyendo y escribiendo constantemente en los elementos del servidor.
RBMCODR7	Cuando el usuario de la orden de finalizar la comunicación se debe enviar al servidor un mensaje de finalización.

Fuente: propia

Tabla A-8: Requerimientos básicos del módulo cliente OPC XML en Rtai-Lab.

ID	DESCRIPCION REQUERIMIENTO
RBMCOXR1	Se debe permitir al usuario seleccionar entre dos bloques, uno para la lectura y otro para la escritura.
	Bloque de lectura
RBMCOXR2	Se debe permitir al usuario configurar los parámetros del bloque de lectura.
RBMCOXR3	Se debe permitir conectar las salidas del bloque, para que la información proveniente de este se pueda visualizar.
	Bloque de escritura
RBMCOXR4	El bloque de escritura debe capturar los valores provenientes de los servidores OPC XML. Cada valor de la salida debe tener asociado un valor proveniente de cada ítem del servidor.
RBMCOXR5	Se debe permitir enviar datos a los ítems mediante el bloque de lectura. Los usuarios deben poder asignar valores a los objetos ítems del servidor OPC XML.
	Inicio de la comunicación entre Rtai-Lab y el servidor OPC DCOM
RBMCOXR6	Se debe iniciar la comunicación cuando el usuario lo solicite, en esta parte se debe estar leyendo y escribiendo constantemente en los elementos del servidor.
RBMCOXR7	Cuando el usuario de la orden de finalizar la comunicación se debe enviar al servidor un mensaje de finalización.

Fuente: propia

Tabla A-9: Requerimientos básicos para el módulo servidor OPC XML

ID	DESCRIPCION REQUERIMIENTO
	Configuración de parámetros iníciales
RBMSOX1	Se debe presentar un formulario donde el usuario establezca las propiedades del servidor.
	Configuración de objetos OPC
RBMSOX2	Se debe presentar un formulario donde el usuario puede crear y configurar los objetos del servidor
	Comunicación con los clientes
RBMSOX3	Se debe aceptar la solicitud de comunicación a los clientes OPC XML.
RBMSOX4	Retornar a los clientes información acerca de las propiedades de los objetos (servidor, grupo, ítems).
RBMSOX5	Se debe capturar los datos desde las plataformas de control y convertirlos en el formato estándar para que puedan ser leídos por los clientes consumidores, en este caso el sistema HMI WEB.
RBMSOX6	Finalizar la comunicación con los clientes OPC XML y retornarles un mensaje de finalización.

Fuente: propia

Tabla A-10: Requerimientos básicos para el módulo de monitoreo y supervisión Web

ID	DESCRIPCION REQUERIMIENTO
RBMM&SW1	El sistema de monitoreo y supervisión web debe contar con tres sub-módulos básicos: gestor de contenidos, diseño de proyectos y monitoreo de procesos.
	Gestor de contenido
RBMM&SW2	Posibilitar la actualización, mantenimiento y ampliación de la web con la colaboración de múltiples usuarios.
	Diseño de proyectos
RBMM&SW3	Adquirir un conjunto de datos provenientes de los servidores OPC DCOM y XML.
RBMM&SW4	El diseño de las pantallas a visualizar tomando información de los elementos de los servidores OPC
RBMM&SW5	Manejar alarmas
RBMM&SW6	Manejar eventos
RBMM&SW7	Configurar la ayuda del sistema HMI
	Monitoreo de procesos
RBMM&SW8	Alertar al operador través de las señales de alarma frente a una falla o la presencia de una condición perjudicial o fuera de lo aceptable. Estas señales deben ser visuales.
RBMM&SW9	Brindar imágenes en movimiento que representen el comportamiento del proceso, dándole al operador la impresión de estar presente dentro de una planta real.
RBMM&SW10	Representar variables mediante curvas de señales analizadas en el tiempo.

Fuente: propia

ANEXO B. MANUAL DE USUARIO DEL MÓDULO SERVIDOR OPC DCOM KEPServerEx

I. INTRODUCCION

KEPServerEx es una aplicación que proporciona un medio para llevar información de una amplia gama de dispositivos industriales y aplicaciones cliente bajo WINDOWS®. Esta aplicación cliente – servidor habilita el intercambio de datos de producción entre numerosas aplicaciones desde HMI, servidores de datos históricos hasta MES y ERP. Cualquier versión de este software soporta comunicaciones de tipo OPC Data Access Versión 1.0a, 2.05a, 3.0; interfaz PDB para iFix y formatos DDE como son *CF_Text y Advance DDE*.

El servidor KEPServerEx está compuesto de dos partes, la primera proporciona toda la conectividad OPC y DDE, así como las funciones de interfaz de usuario y la segunda parte está compuesta por los drivers de comunicación. Esta división permite al usuario utilizar múltiples opciones de comunicación con distintos canales, por medio de un único servidor.

Esta guía contiene los pasos básicos para la configuración de los elementos OPC en el servidor KEPServerEx versión 4.0. Inicialmente se presentan los pasos de la instalación y ejecución, luego se describe la configuración de los objetos ítems para la comunicación desde el PLC *Micrologix* 1500, y finalmente se detalla la configuración de objetos ítems para Matlab y Rtai-lab.

II. INSTALACION

Para instalar el Servidor OPC KEPserverEX, se debe seguir los siguientes pasos:

- Descargar el instalador de la página: http://www.kepware.com, en versión demo o comprar la licencia (los precios de las diferentes versiones del servidor KEPserverEX se encuentran también en esta página).

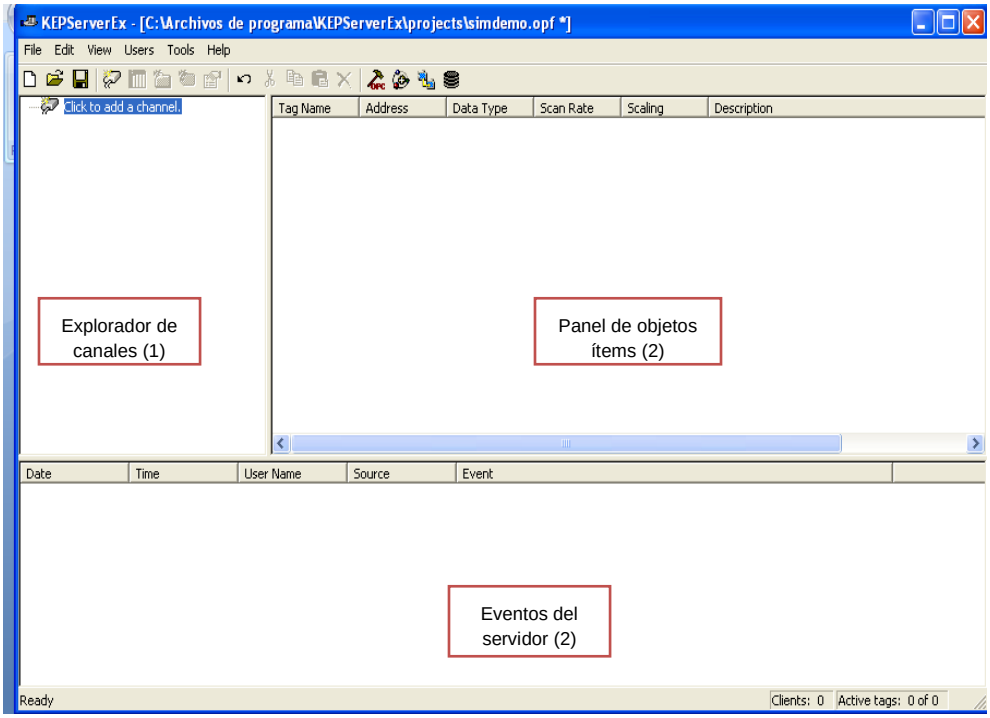
- instalarlo como cualquier aplicación que posee asistente de instalación "*InstallShield Wizard*" que guía al usuario durante este proceso.

III. EJECUCION

Una vez instalado el software KEPserverEX, se debe iniciar el servidor, haciendo clic en el icono 

La interfaz de usuario principal del servidor KEPserverEX está dividida en tres paneles: el explorador de los canales al lado izquierdo, ver ilustración B-1 recuadro 1, a la derecha el panel de vista detallada de los "ítems" del servidor, en otras palabras los objetos ítems, ver recuadro 2, y el tercer panel contiene los eventos de servidor (*event log*), ver recuadro 3.

Ilustración B-1: Interfaz de usuario - KEPserverEX V4.0



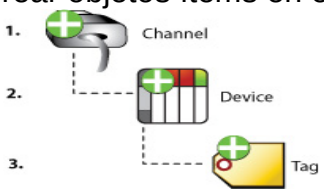
Fuente: propia

La ilustración B-2 presenta los pasos para configurar los objetos ítems (tags) en el servidor KEPserverEX los cuales se listan a continuación:

1. Seleccionar un *Driver* para crear un canal

2. Especificar un dispositivo o sistema de comunicación
3. Crear un grupo de *tags*

Ilustración B-2: Pasos para crear objetos ítems en el Servidor KEPserverEX



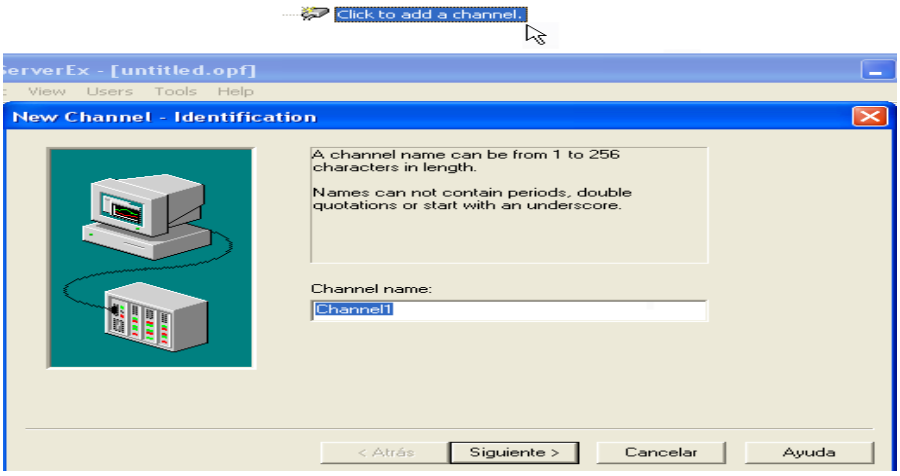
Fuente: propia

A continuación se detallan los pasos necesarios para configurar los objetos ítems en el servidor OPC DCOM KEPserverEX, primero se describe la configuración para la comunicación con el PLC Micrologix 1500, y luego se realiza una guía de configuración de ítems para Matlab y Rtai-Lab.

1. Configuración de objetos ítems para la comunicación con el PLC

Para realizar la comunicación entre el servidor y el dispositivo (PLC u otro equipo) es necesario crear un canal de comunicación, para esto se hace clic en el icono "*clic to add channel"* en el panel izquierdo, ver ilustración B-3.

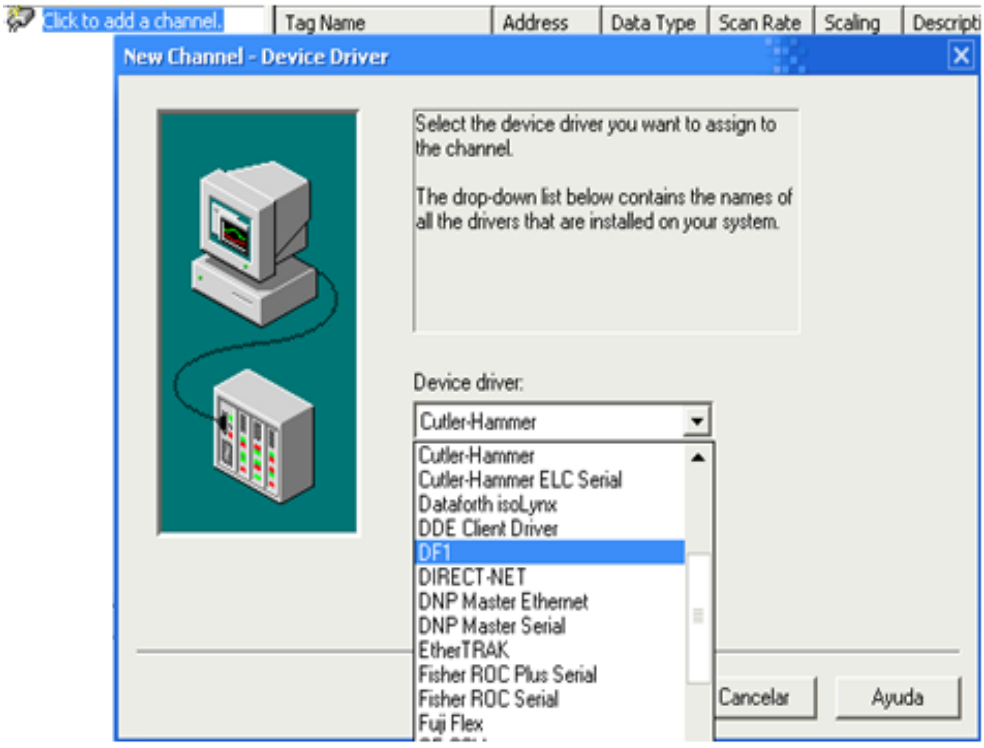
Ilustración B-3: Asistente de configuración del canal de comunicación

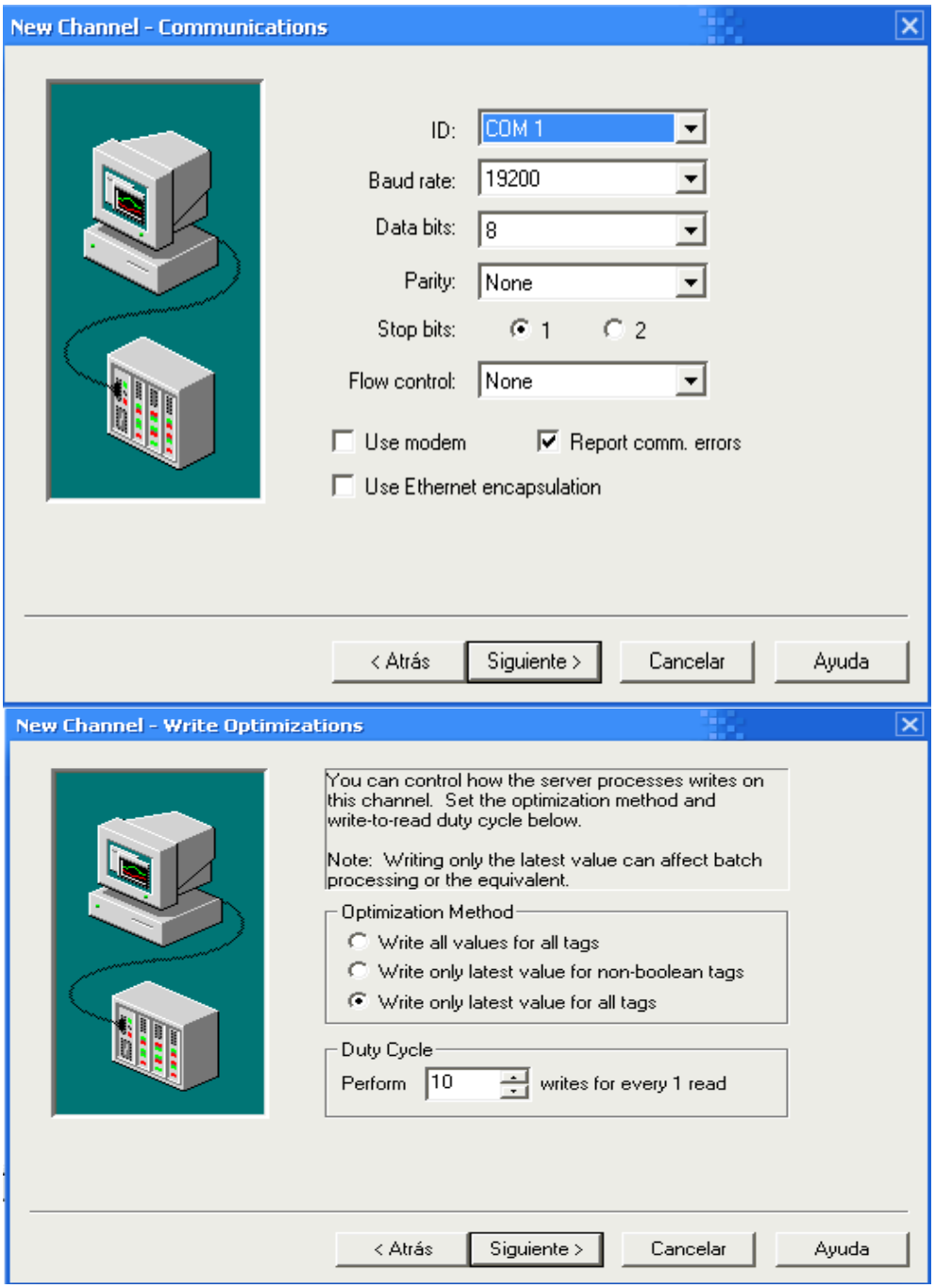


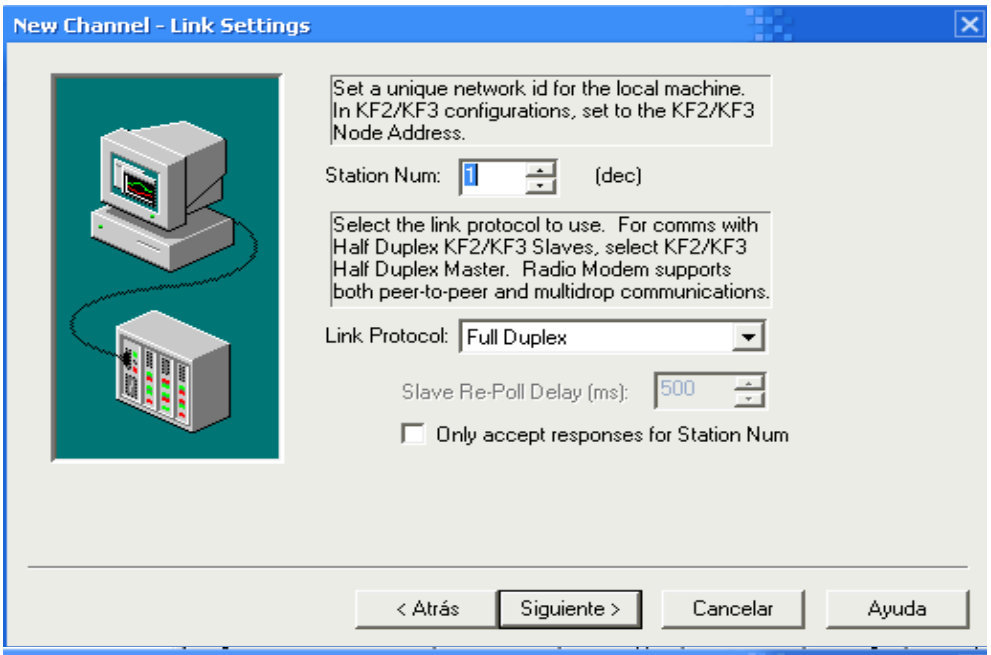
Fuente: propia

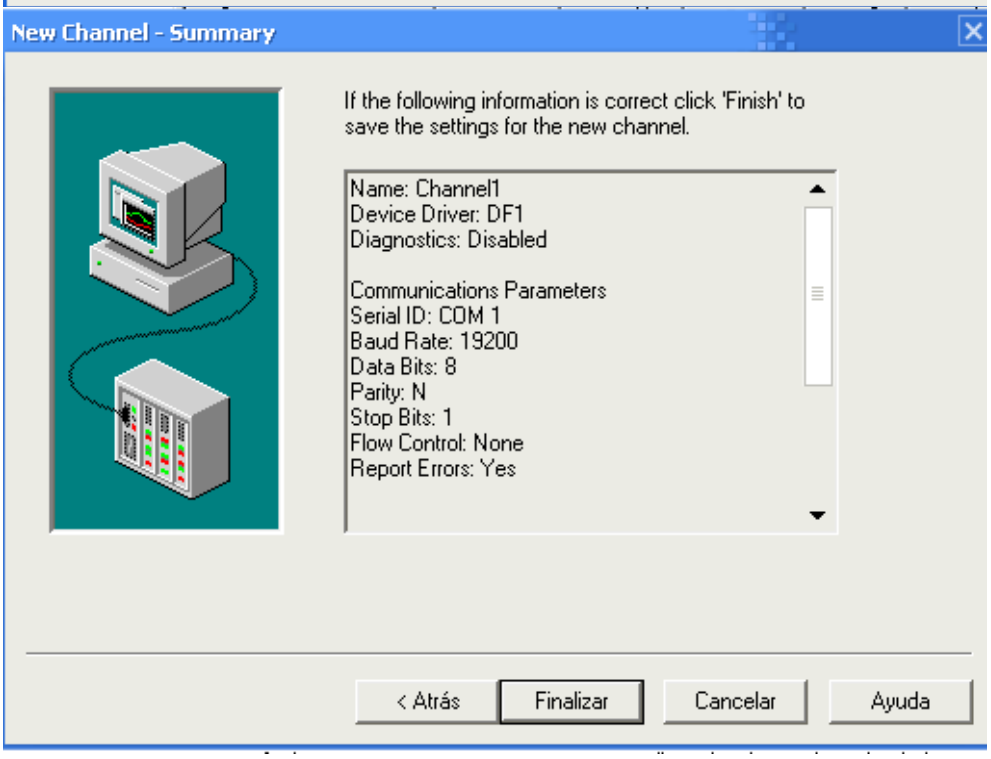
En este asistente se configuran todas las características del canal de comunicación, luego de darle un nombre al canal (Channel1) se da clic en el botón <siguiente> y se configura el driver que se va a usar, para este caso es el “DF1”, luego se configuran las comunicaciones, el ID se configura en “COM 1”, la velocidad de transmisión en “19200 Baudios”, el tamaño de la palabra en “8 bits”, sin paridad, con un bit de stop y habilitada la opción de reporte de los errores de comunicación. En la siguiente pantalla se configuran las opciones de optimización de escritura en donde no se modifican los valores por defecto. Por último se configuran las opciones del enlace, se asigna un identificador de red para la máquina local o la dirección del nodo, en este caso “*Station Num*: 1” y el protocolo de enlace debe ser “Full *Dúplex*”. Al dar clic en el botón <siguiente> se presenta un resumen de la configuración y se finaliza el asistente, ver ilustración B-4.

Ilustración B-4: Pasos en la configuración del canal de comunicación en servidor KEPserverEX




New Channel - Communications
ID:
COM 1
Baud rate:
19200
Data bits:
8
Parity:
None
Stop bits:
1
2
Flow control:
None
Use modem
Report comm. errors
Use Ethernet encapsulation
< Atrás
Siguiente >
Cancelar
Ayuda
New Channel - Write Optimizations
You can control how the server processes writes on this channel. Set the optimization method and write-to-read duty cycle below.
Note: Writing only the latest value can affect batch processing or the equivalent.
Optimization Method
Write all values for all tags
Write only latest value for non-boolean tags
Write only latest value for all tags
Duty Cycle
Perform
10
writes for every 1 read
< Atrás
Siguiente >
Cancelar
Ayuda

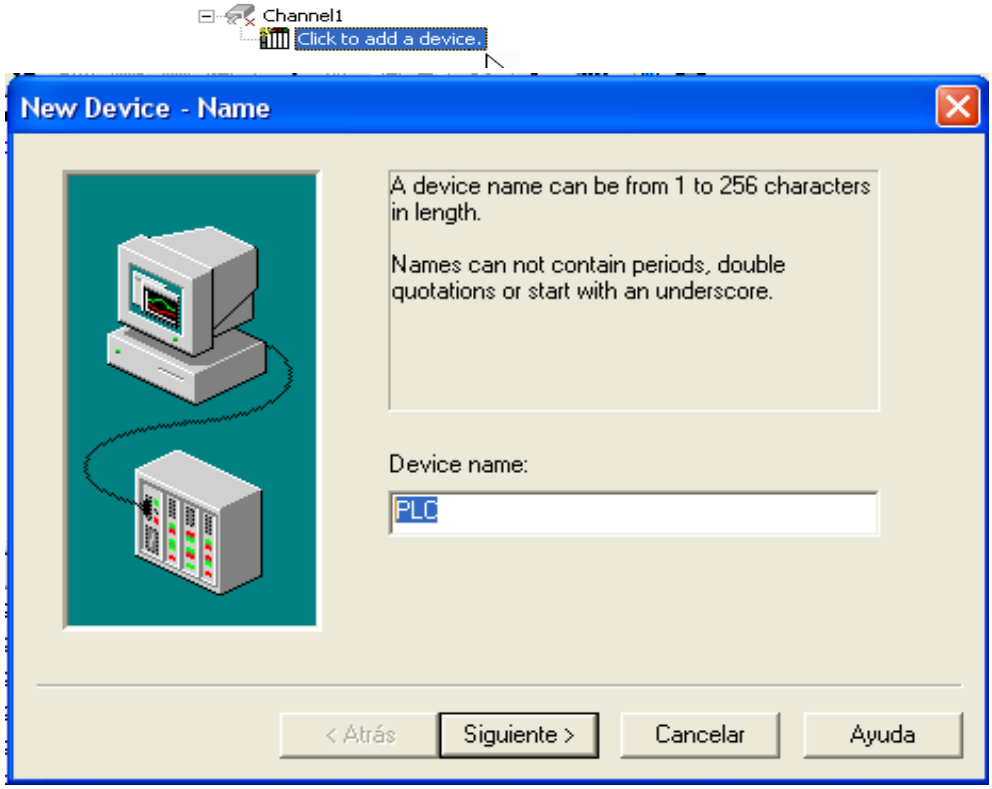




Fuente: propia

Una vez se tenga el canal creado y configurado se debe crear un "*device*" que este caso es el PLC (*MicroLogix* 1500), ver ilustración B-5.

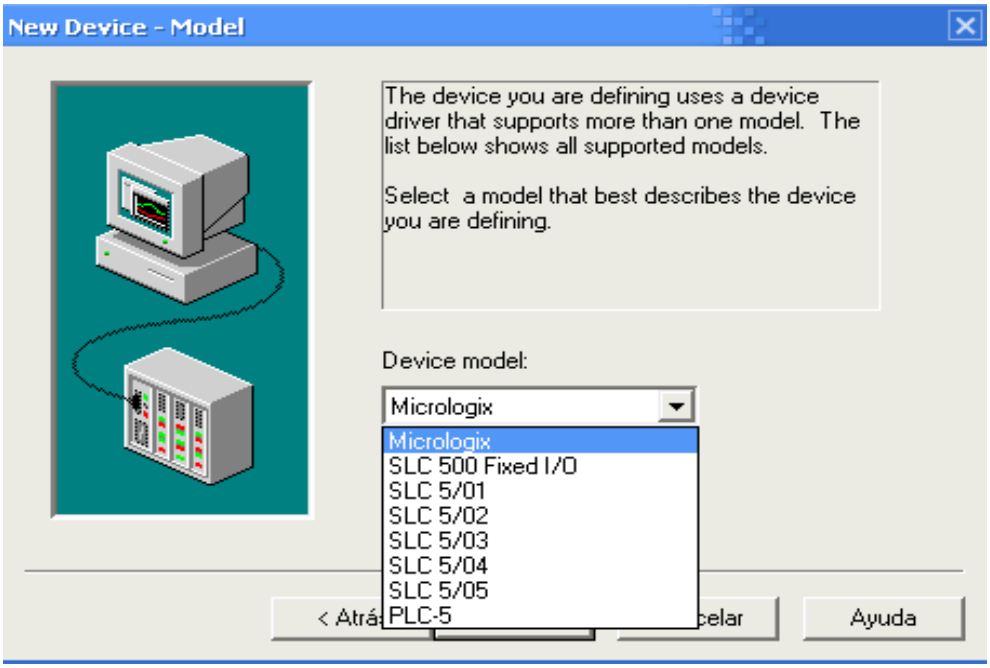
Ilustración B-5: Asistente de configuración del dispositivo en servidor KEPserverEX

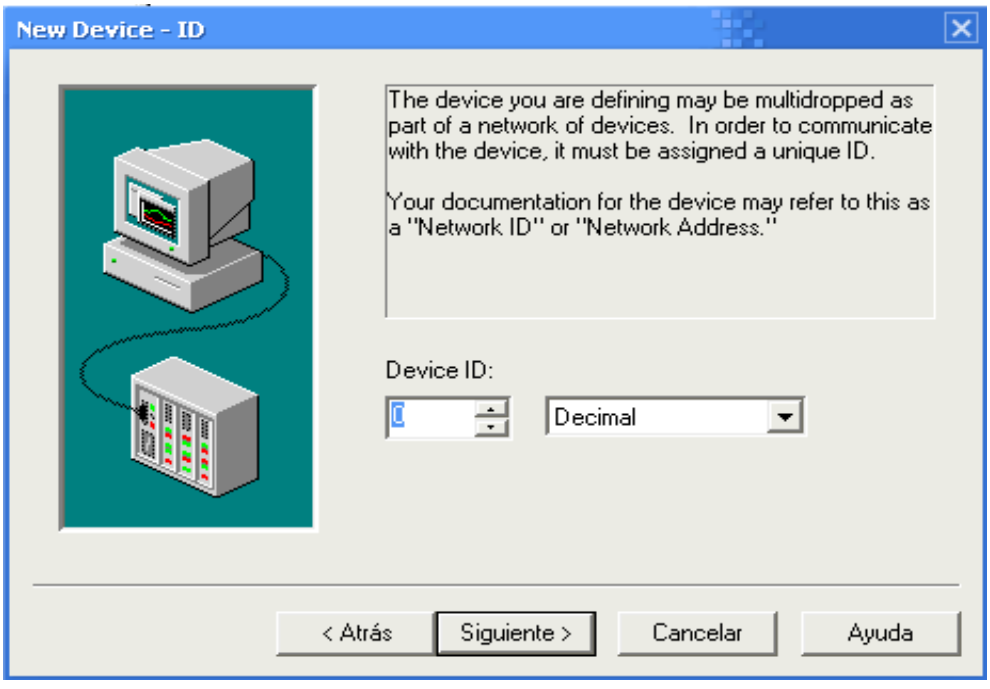


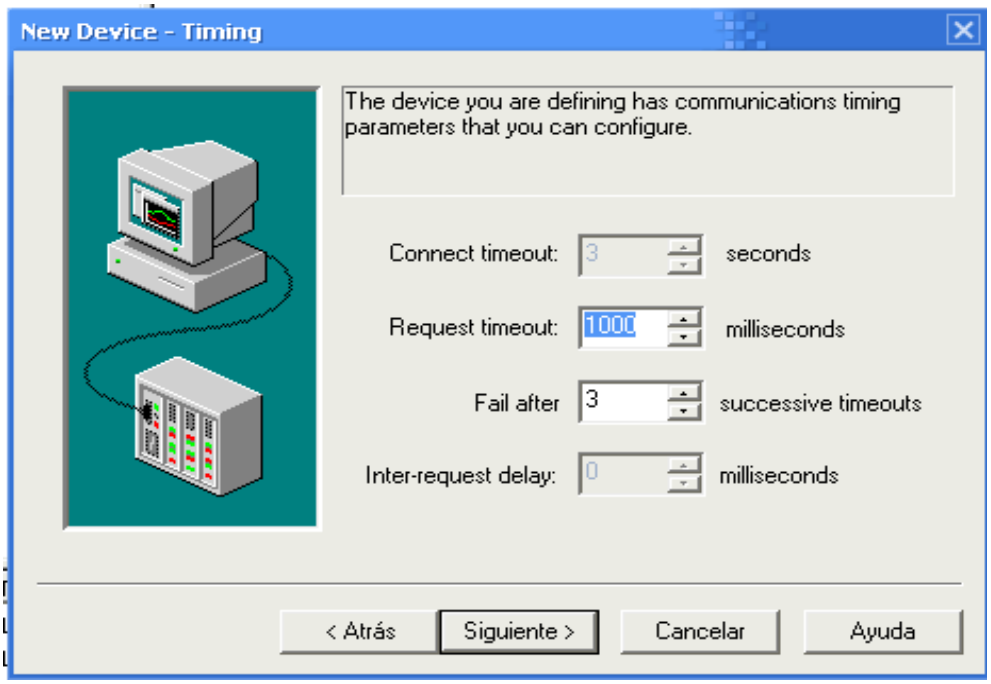
Fuente: propia

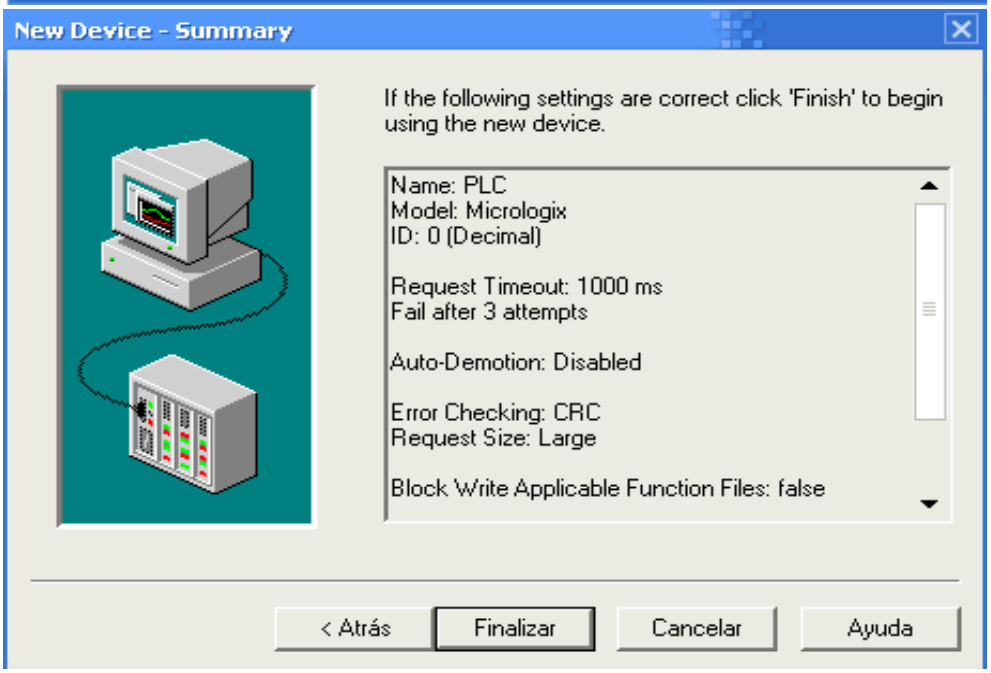
En este asistente se crea y configura un dispositivo conectado al canal que se configuró anteriormente, en la primera pantalla se asigna un nombre al dispositivo, se da clic en el botón <siguiente> y el asistente presenta los dispositivos que soportan el canal de comunicación configurado, para este caso se configura "micrologix", luego se asigna un identificador de red al dispositivo, en este caso "*Device* ID: 0", en las siguientes 4 pantallas del asistente, "*Timing", "Auto-Demotion", "Protocol Settings", "Function File Options",* no se modifican los parámetros por defecto, por último se presenta un resumen de la configuración y se finaliza el asistente, ver ilustración B-6.

Ilustración B-6: Pasos para la configuración del dispositivo para el canal seleccionado









Fuente: propia

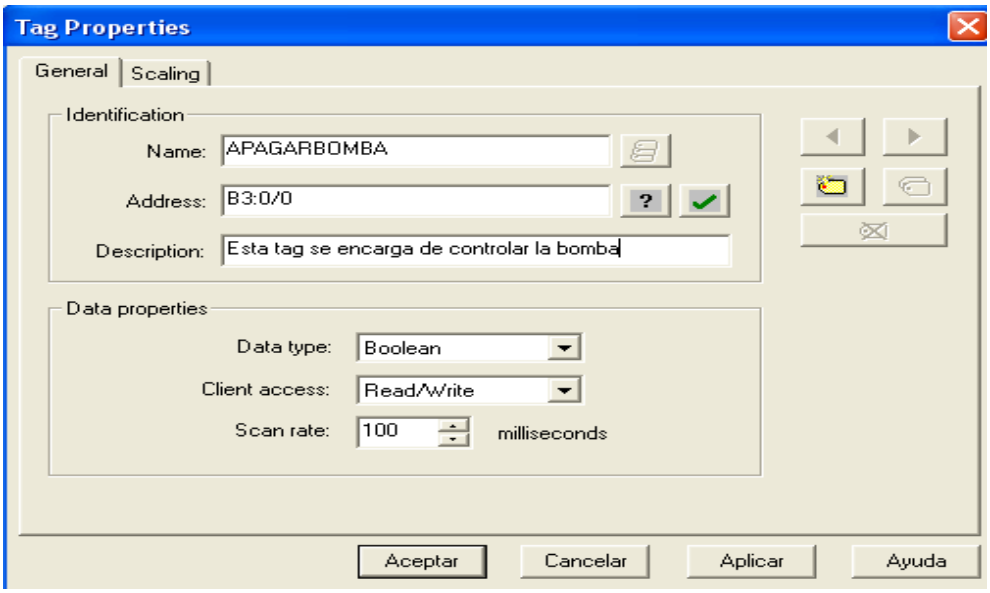
Teniendo el canal y el dispositivo configurado, es necesario crear un grupo en donde se almacenan los objetos ítems del equipo. Sobre el icono del dispositivo se hace clic derecho como se muestra en la ilustración B-7.

Ilustración B-7: Creación de un grupo de objetos ítems en el dispositivo PLC.

Fuente: propia

Se crean uno a uno los objetos ítems que maneja el PLC, dándole un nombre (preferiblemente el mismo que tiene en el PLC), se debe asignar la misma dirección que posee la tag en el PLC, una descripción de la misma y haciendo clic en el botón se confirman las propiedades de los datos, ver ilustración B-8. Con esto se tiene el canal, el dispositivo, el grupo y los ítems (objetos ítems) del servidor OPC. Luego de toda la configuración es necesario verificar si en el panel "Event Log" (panel inferior) de la interfaz se ha desplegado algún error.

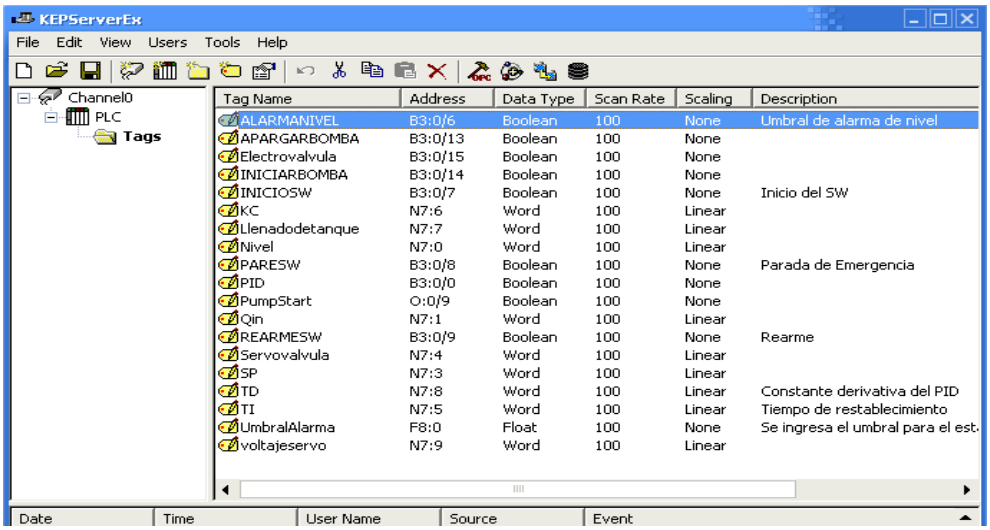
Ilustración B-8: Formulario para la creación de objetos ítems en servidor KEPserverEX



Fuente: propia

A partir del código del Ladder se deben clasificar las variables (objetos ítems) del proceso con su respectiva dirección de memoria del PLC. En el caso que la configuración sea exitosa, la interfaz del servidor se verá como en la ilustración B-9.

Ilustración B-9: ítems para la comunicación con el PLC Micrologix 1500

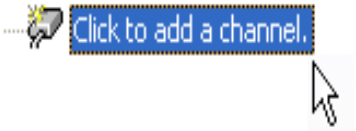


Fuente: propia

2. Configuración de objetos ítems para Matlab Y Rtai-Lab

Como en el paso anterior, también es necesario crear un canal de comunicación, un dispositivo, y un grupo de tags asociados. Para crear un canal de comunicación se hace clic en el icono "*click to add channel"* en el panel izquierdo, ver ilustración B-10.

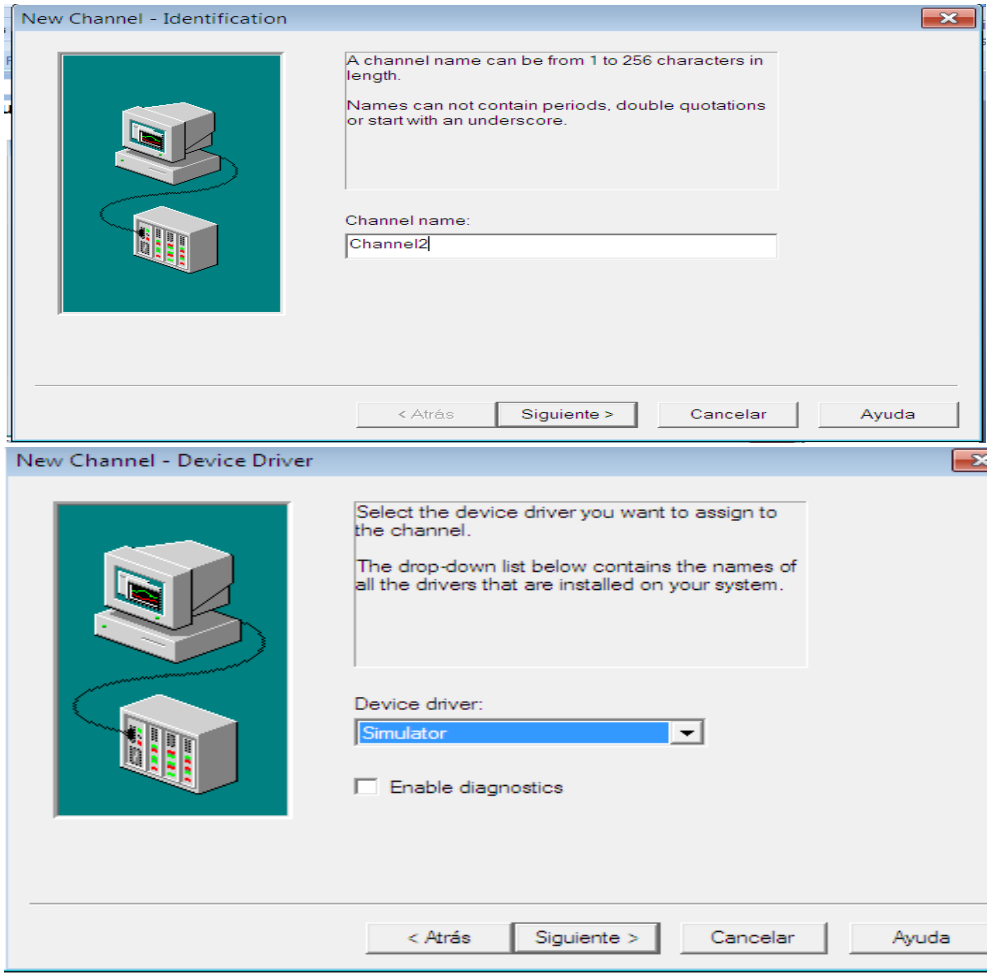
Ilustración B-10: Asistente de configuración de canal en servidor KEPserverEX

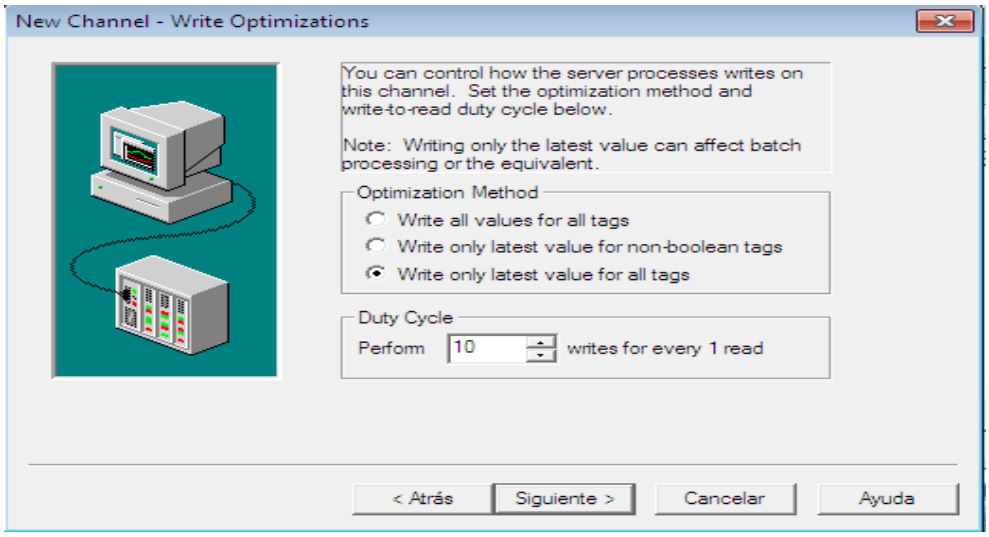


Fuente: propia

En este asistente se configuran todas las características del canal de comunicación, luego de darle un nombre al canal (Channel2) se da clic en el botón <siguiente> y se configura el driver que se va a usar, para este caso no se requiere driver por lo tanto se utiliza la opción "*simulator*", al dar clic en el botón <siguiente> se presenta una pantalla que permite la configuración de algunos parámetros y finalmente se finaliza el asistente, ver ilustración B-11.

Ilustración B-11: Pasos para la configuración de canal en el servidor KEPserverEX

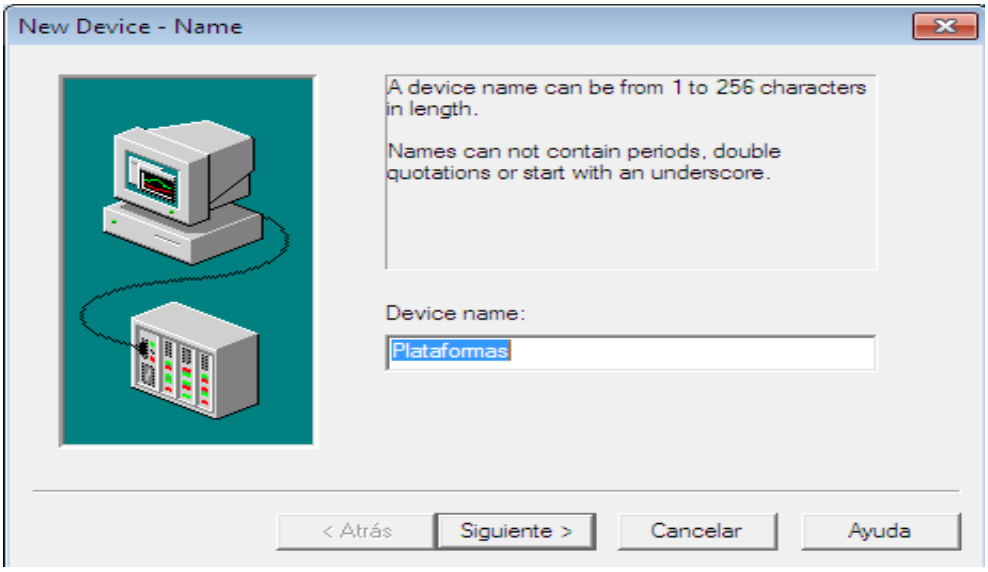


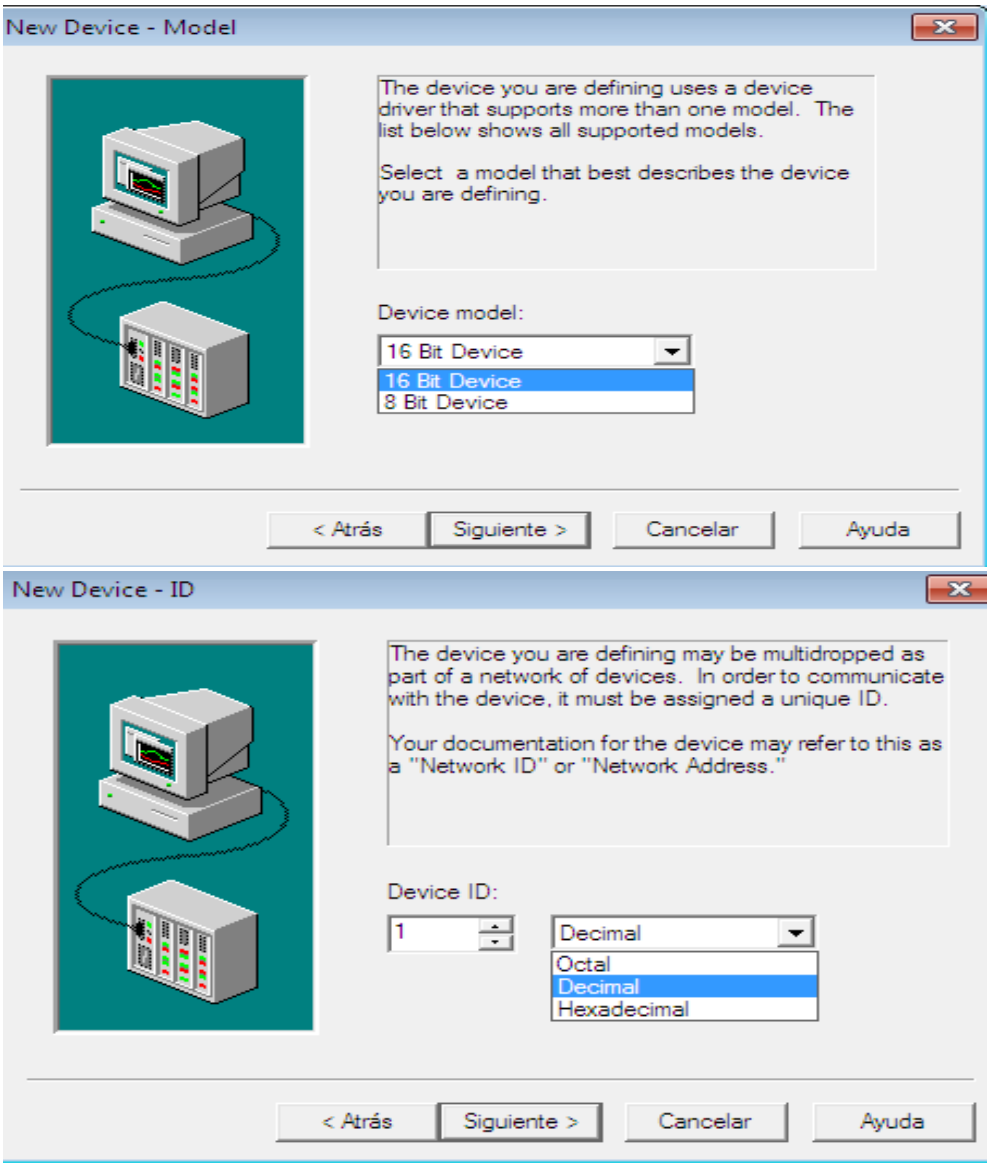


Fuente: propia

Una vez se tenga el canal creado, se debe configurar un dispositivo (device), como se indica en la ilustración B-12, en la primera pantalla del asistente se asigna un nombre al dispositivo (en este caso plataformas), al hacer clic en el botón <siguiente>, el asistente presenta los dispositivos existentes para el canal de comunicación configurado; dependiendo del tipo de variables se selecciona el modelo del dispositivo (de 8 o 16 bits), luego se asigna un identificador de red para cada dispositivo (en este caso “Device ID: 1”), y se finaliza el asistente.

Ilustración B-12: pasos para la creación de un dispositivo en KEPserverEX

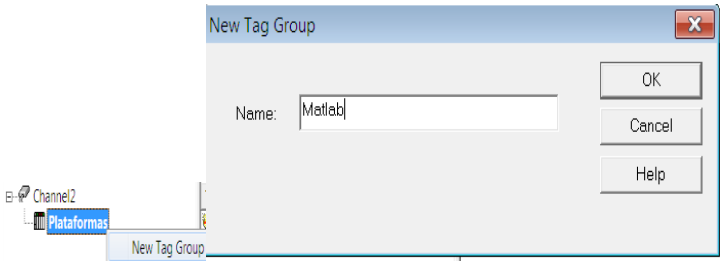




Fuente: propia

Teniendo el canal y el dispositivo creado es necesario establecer un grupo de objetos ítems para cada plataforma de control, para fijar el nombre del grupo se hace clic derecho en el nombre del dispositivo (en este caso Plataformas) y se establece el nombre como se muestra en la ilustración B-13.

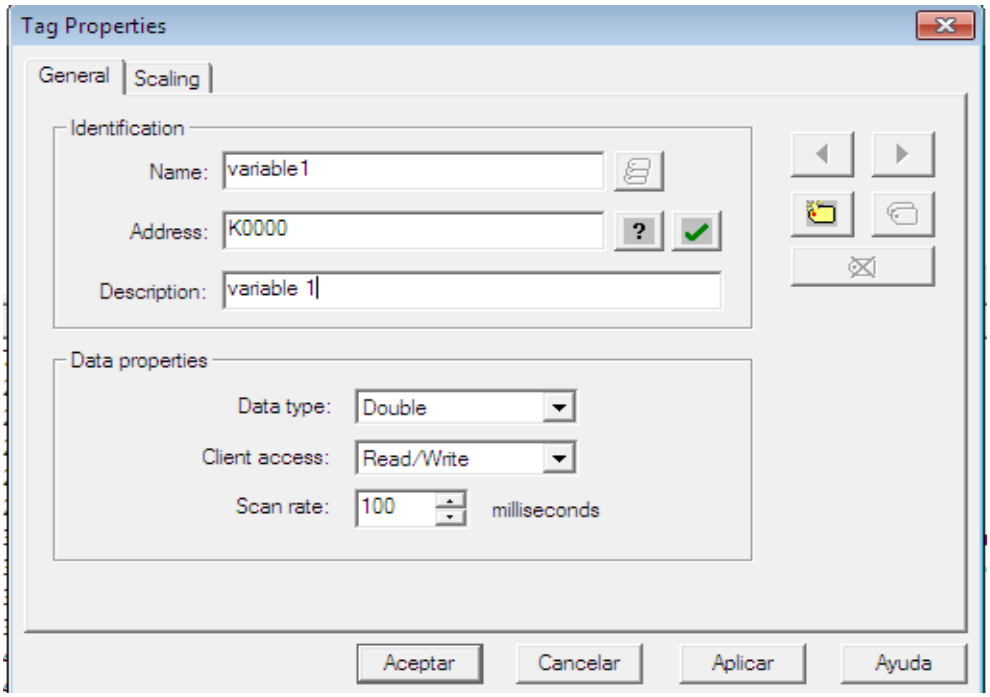
Ilustración B-13: Creación de un grupo de objetos ítems en el dispositivo



Fuente: propia

Para agregar los objetos ítems al servidor se pueden importar desde un archivo tipo CSV si se cuenta con él, en caso contrario se inicia el asistente de creación de objetos ítems. Para crear los ítems se hace clic en el grupo Matlab, donde aparece un formulario, ver ilustración B-14, que permite configurar las siguientes propiedades: nombre, dirección (presionar en ? para consultar el valor correspondiente de dirección), tipo de datos. Para finalizar se hace clic en el botón y se verifica si en el panel "Event Log" (panel inferior) de la interfaz se ha desplegado algún error.

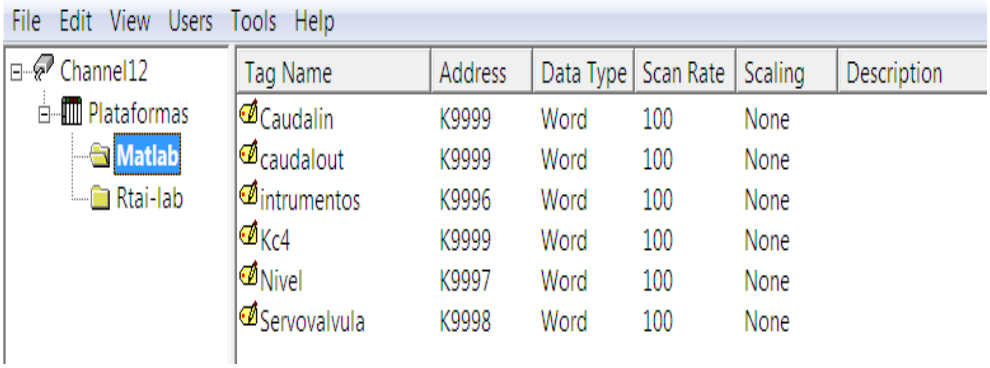
Ilustración B-14: Formulario de configuración de objetos en KEPserverEX



Fuente: propia

Luego de configurados los objetos items, el servidor OPC Kepserver estara listo para establecer la comunicación con Matlab y Rtai_lab, la ilustracion B-15 presenta una pantalla con los objetos items configurados para Matlab y Rtai_lab.

Ilustración B-15: Lista de ítems creados en KEPserverEX para Matlab y Rtai-Lab.

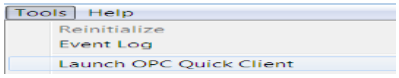


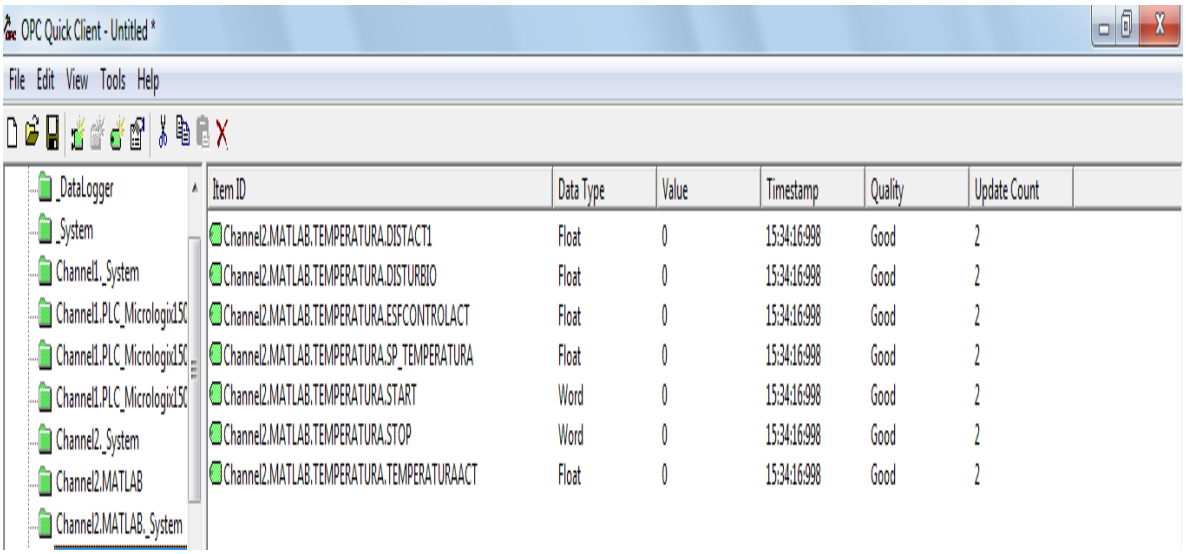
Fuente: propia

Cuando los objetos items ya estan configurados, los clientes pueden solicitar el inicio de la comunicación y las propiedades de estos objetos items.

El instalador del servidor KEPserverEX tiene tambien un cliente OPC para la lectura y escritura de los objetos ítems, para la utilizacion de este cliente se debe hacer clic en en el menu del servidor: “tools”-“Launch OPC Quick Client”, para que aparezca una pantalla con todos los objetos existentes en el servidor, como la que se presenta en la ilustracion B-16.

Ilustración B-16: Cliente OPC DCOM presente en el servidor KEPserverEX





Fuente: propia

ANEXO C. GUÍA DE INSTALACIÓN Y MANUAL DE USUARIO DEL MÓDULO SERVIDOR OPC-XML

I. INTRODUCCION

En el desarrollo de esta guía se presenta la instalación y configuración del Módulo servidor OPC XML, el cual permite el acceso de lectura y escritura de elementos desde otras aplicaciones cliente que manejen el estándar OPC XML. Inicialmente se describe la instalación del MSOX en Windows y en Linux, luego se describe la configuración y uso del MSOX, y finalmente se verifica el funcionamiento del MSOX con una prueba que consiste en comunicar el servidor OPC XML y el cliente comercial *dOPC Explorer.*

Para la utilización del Módulo servidor OPC XML se debe tener en cuenta que: los enlaces OPC XML están pensados para un intercambio de pocos datos, el tiempo de ciclo de las variables no debe configurarse por debajo de 1 segundo, en las variables de tipo "*Stirling*" sólo se soportan los caracteres ASCII válidos del 20 ex al 7F ex.

Dentro de las tecnologías utilizadas en el Módulo Servidor OPC XML se encuentran:

- ***Soap***: es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML (5).

- ***XML:*** es un lenguaje de etiquetado extensible muy simple, pero estricto que juega un papel fundamental en el intercambio de una gran variedad de datos (6).

II. *INSTALACION DEL SERVIDOR OPC XML*

El servidor OPC XML funciona en cualquier sistema operativo capaz de soportar Python, pero en este apartado se presentan los pasos de instalación en Fedora 3 y en Windows 7.

1. Instalación del MSOX en Linux (Fedora Core)

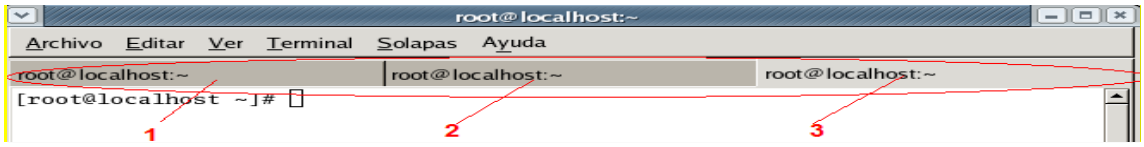
Para ejecutar el servidor OPC XML en Linux es indispensable utilizar la herramienta "Terminal", desde la cual se puede ejecutar una serie de sentencias para acceder a ciertos lugares del sistema. Para iniciar el terminal se hace clic sobre la ventana "Terminal" en la barra de herramientas que aparece en la parte inferior del escritorio, ver ilustración C-1.

Ilustración C-1: Ubicación de *Terminal* en la barra de tarea en Fedora.

Fuente: propia

Una vez esté abierta la ventana de terminal se hace clic en "archivo/abrir solapa, 1/default", este procedimiento permite abrir las ventanas para las instrucciones necesarias, ver ilustración C-2.

Ilustración C-2: Ventana Terminal abierta en Linux con tres solapas

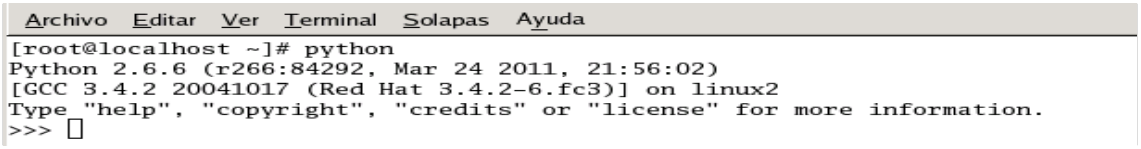


Fuente: propia

El siguiente paso consiste en ingresar los comandos correspondientes a cada solapa, se debe tener en cuenta las siguientes recomendaciones: para el ingreso de instrucciones en el terminal de comandos se debe prestar especial atención a los espacios (identificados en esta guía con el símbolo $), puntos, guiones dobles (__), y flash (/), que se presentan en las instrucciones que vienen a continuación.
La flecha hacia arriba (↑) del teclado permite ver algunas de las instrucciones que digitó anteriormente, esto le puede servir para corregir en caso de cometer errores al momento de digitar las instrucciones.

Verificar la instalación de Python: Python viene por defecto en Linux, pero si no está instalado se debe descargar el archivo "Python2.5.tgz.tar" de la siguiente página: http://www.Python.org/download/. Para saber si Python está instalado se debe escribir **Python** en la consola del terminal y se presiona entre, ver ilustración C-3.

Ilustración C-3: Ventana de verificación de versión de Python en Linux



Fuente: propia

Copiar instaladores del servidor: copiar la carpeta **código_fuente/servidor_OPC_XML** en el disco duro del computador, para el caso de ejemplo la carpeta quedara ubicada en el disco duro /home.

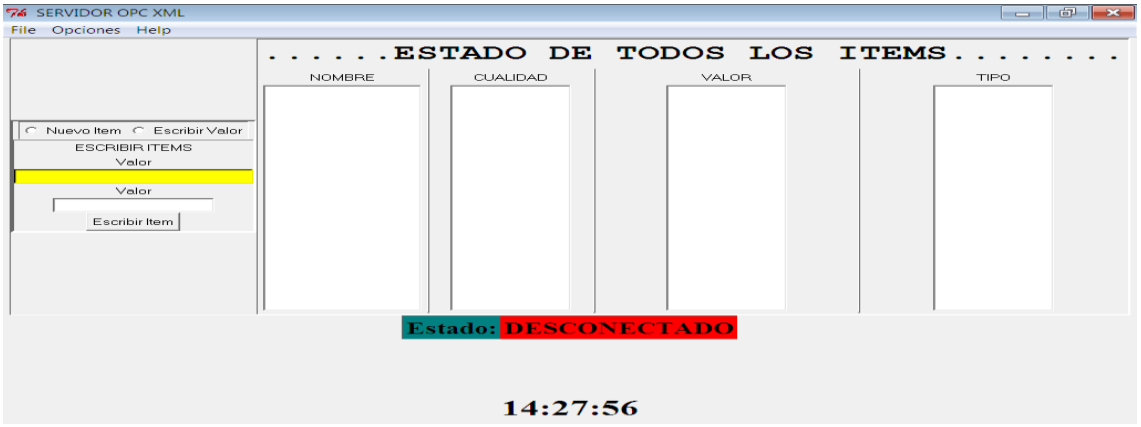
Adicionar librerías: ubicarse en /home/Servidor_OPC_XML y copiar las carpetas "site-packages" y XML en la carpeta LIB que contiene las librerías de Python, para la versión 2.5 de Python la dirección es usr/local/lib/Python2.5/, debe aparecer un mensaje que afirma que las carpetas ya existen, se debe remplazar dichas carpetas debido a que se adicionaron y actualizaron librerías.

Ejecutar el servidor: para ejecutar el servidor se debe abrir un terminal, y ubicarse en el directorio donde está la carpeta Servidor_OPC_XML, insertar en la solapa las siguientes instrucciones presionando *enter* después de cada instrucción:

cd$home/Servidor_ OPC_XML
Python$inicio.py

Si el servidor se ha instalado correctamente se debe presentar la pantalla principal del servidor OPC_XML, ver ilustración C-4.

Ilustración C-4: Pantalla principal del MSOX en estado desconectado.



Fuente: propia

2. Instalación del servidor OPC XML en Windows

Los pasos que se deben seguir para la instalación del MSOX en Windows son los siguientes:

Instalación de Python: descargar el instalador de Python 2.5 o superior de la siguiente página: http://www.Python.org/download/ e instalarlo siguiendo las instrucciones del asistente de instalación.

Copiar instaladores del servidor: copiar la carpeta que está en la ruta: código_fuente/servidor_OPC_XML, en el disco duro del computador, para el caso de ejemplo la carpeta quedara ubicada en el disco duro C.

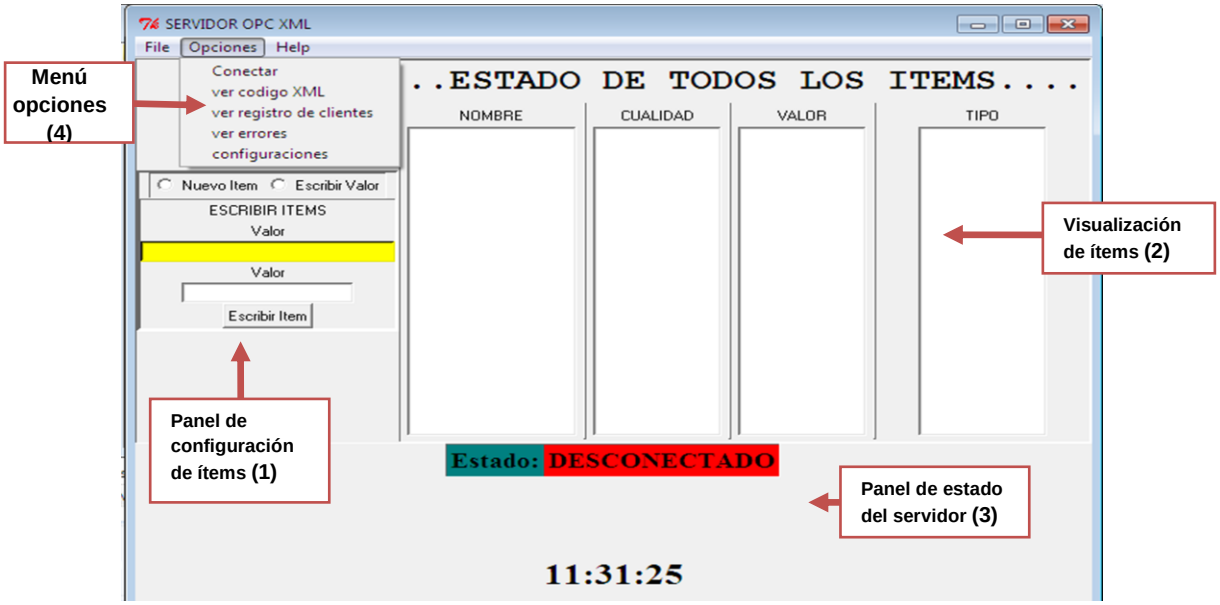
Adicionar librerías: ubicarse en C:/Servidor_OPC_XML y Copiar las carpetas "site-packages" y XML en la carpeta LIB que contiene las librerías de Python, para la versión 2.5 de Python la dirección es C:\Python25\Lib, debe aparecer un mensaje que afirma que las carpetas ya existen, se debe remplazar dichas carpetas debido a que se adicionaron y actualizaron librerías.

Ejecutar el servidor: Ubicarse en la carpeta "C:/Servidor_OPC_XML" y dar doble clic en el archivo inicio.py.

III. CONFIGURACION Y USO DEL MSOX

La pantalla principal del servidor OPC XML, está compuesta por tres paneles, ver ilustración C-5, el primer panel es de configuración de objetos ítems (recuadro 1), donde el usuario puede crear o editar las propiedades de cada objeto ítem, el segundo es el de visualización (recuadro 2), donde se presenta cada una de los objetos ítems con sus respectivas propiedades (Nombre, valor, cualidad, tipo de datos) y el tercero, ver recuadro3, indica el estado del servidor (conectado, desconectado). También se presenta el menú "File" donde se puede guardar y abrir los archivos correspondientes a los ítems, el menú "opciones" donde se establecen las propiedades del servidor, ver numeral 4, y "help" presenta un manual de usuario para el manejo del servidor.

Ilustración C-5: Pantalla principal del Servidor OPC XML.

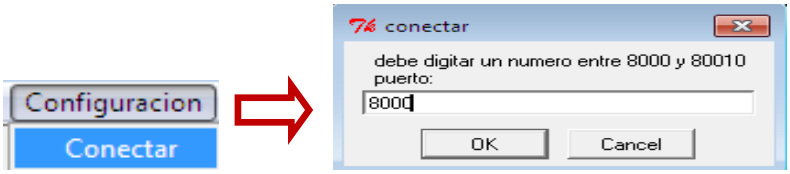


***Fuente**: propia*

Para que los clientes OPC XML puedan comunicarse con el MSOX se deben efectuar los siguientes pasos:

Configuración del puerto: el primer paso es configurar el puerto mediante el cual el servidor va a establecer la comunicación. Para esto se presiona el botón "opciones" del menú principal y se da clic en conectar, para que aparezca un formulario, ver ilustración C-6, donde se debe digitar un número entre 8000 y 80010 que representa el puerto de comunicación. Si aparece un mensaje "el puerto está ocupado", se debe intentar trabajar con otro valor numérico.

Ilustración C-6: Ventana de configuración del puerto del MSOX

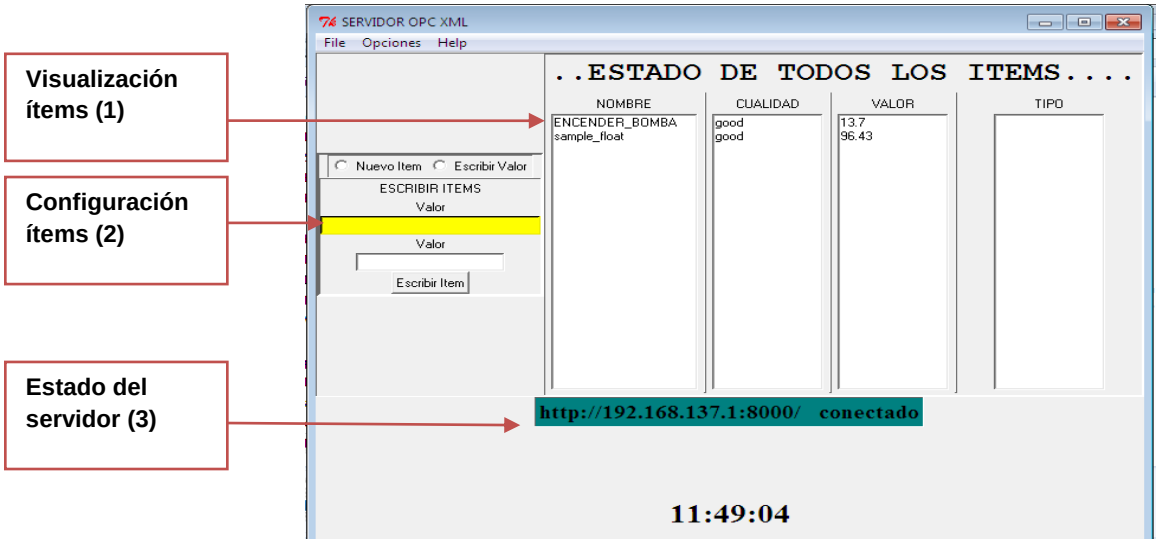


Fuente: propia

En la ilustración C-7, se indica la pantalla del Servidor OPC XML cuando el servidor se ha inicializado correctamente. En el panel de estado del formulario principal, ver recuadro 3, se presenta la dirección completa del servidor OPC XML

(en este caso es: http://192.168.137.1:8000), esta dirección es la que los clientes OPC XML deben configurar para establecer la comunicación con el MSOX. Si el estado del servidor es conectado, se puede empezar a adicionar, modificar, ver recuadro 2, y visualizar los objetos ítems del Servidor OPC, ver recuadro 1, a si mismo los clientes OPC XML pueden iniciar la solicitud de conexión.

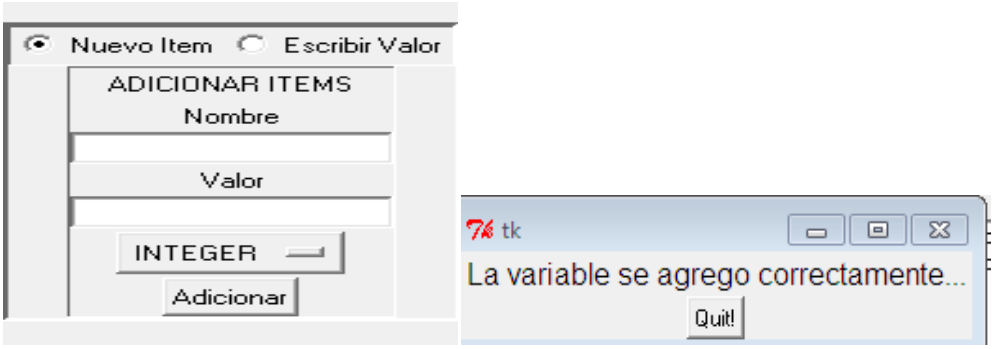
Ilustración C-7: Ventana principal del MSOX en estado conectado.



Fuente: propia

Adicionar ítems: Para adicionar los objetos ítems, se debe agregar el nombre, valor y el tipo de datos, y presionar el botón "adicionar", hasta que aparezca el mensaje "La variable se agregó correctamente", ver ilustración C-8.

Ilustración C-8: Formulario que permite la adición de ítems en el MSOX



Fuente: propia

Si la variable se ha agregado correctamente, se debe visualizar en el formulario de visualización de la pantalla principal el valor, la cualidad (*good o bad*), el nombre y

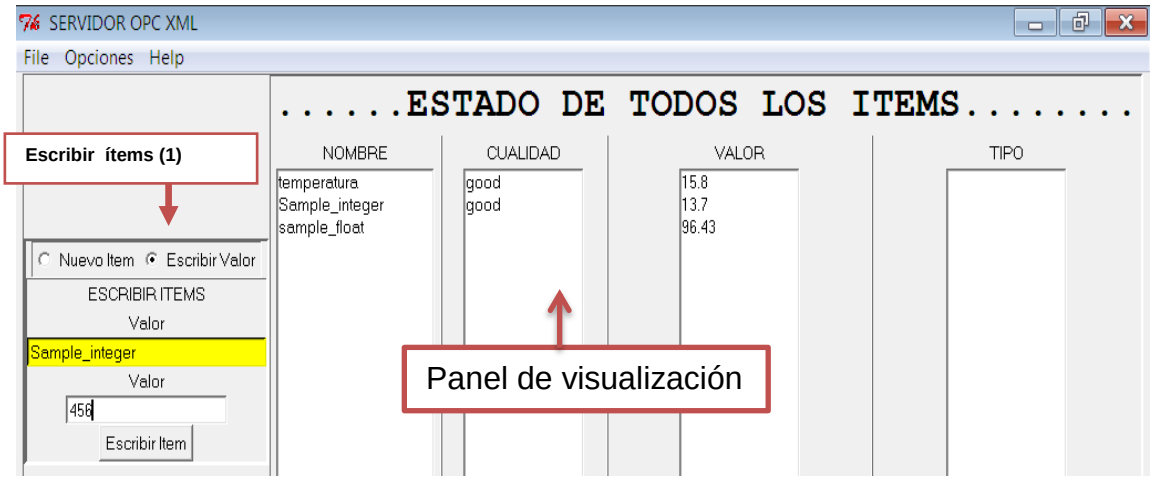
el tipo de datos. La ilustración C-9 presenta el panel de visualización del servidor OPC XML después de que se han agregado tres objetos ítems con sus respectivos valores. Se debe tener en cuenta al adicionar los ítems que: si el tipo de datos es *String* el valor debe ir entre comillas simples y si es un arreglo el valor debe ir entre corchetes, si el tipo de datos es doble o flotante se debe utilizar el punto como separador decimal.

Ilustración C-9: Panel de visualización de ítems en el MSOX

Fuente: propia

Escritura de ítems: Para modificar el valor de cada objeto ítem se debe seleccionar "escribir ítem" del menú configuración de ítems, ver ilustración C-10 recuadro 1. Al dar clic en el nombre en el objeto ítem al cual se va a cambiar el valor (en el panel de visualización), debe aparecer automáticamente el nombre en el cuadro de texto amarillo, en el cuadro de texto "valor" se debe establecer el nuevo valor de cada objeto ítem. En la ilustración C-10 se presenta una pantalla del MSOX con tres variables en el panel de visualización (temperatura, Sample_integer y *Sample_floa*t), al seleccionar *sample_integer*, este nombre aparece inmediatamente en el recuadro amarillo, lo que indica que se puede escribir el nuevo valor a esta variable. Cuando se esté seguro que es el nombre correcto del objeto ítems, se debe digitar el nuevo valor y hacer clic en "escribir ítem". Si el objeto ítem se ha escrito correctamente, el cambio se debe visualizar en el panel de visualización de ítems.

Ilustración C-10: Ventana principal del MSOX cuando se han adicionado ítems.

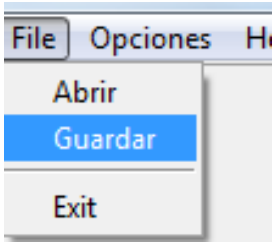


Fuente: propia

Cuando se hayan configurado todas los objetos ítems en el servidor OPC XML, Los clientes OPC XML pueden leer/escribir las propiedades de los objetos ítems.

Guardar configuración de variables: las variables se pueden almacenar en un archivo para su posterior uso, se debe ir al menú “file” - “Guardar” como lo indica la ilustración C-11, seleccionar el directorio y fijar un nombre con extensión “.xml”

Ilustración C-11: Menú para guardar los objetos ítems en el MSOX



Fuente: propia

Abrir archivo existente: para abrir un archivo que contiene la configuración de los objetos ítems, se debe ir al menú “File”- “Abrir”, seleccionar el directorio y abrir el archivo que contiene las variables existentes.

Ayuda: para mirar la información de cómo ejecutar el servidor y sus funciones básicas se debe ir a “menú”- “ayuda”.

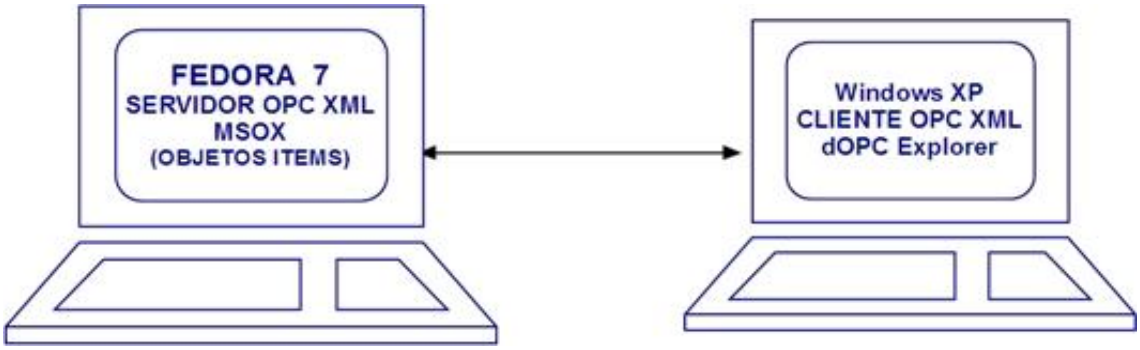
Activar puerto para la comunicación con clientes en equipos remotos: para que el servidor pueda establecer una comunicación con los clientes instalados en otro computador, se debe configurar el firewall para activar el puerto y permitir el intercambio de información, ver el anexo J - sección I (Guía para activar los puertos para la comunicación - pasos para activar puertos TCP).

Si el servidor OPC XML está bien configurado, los clientes OPC XML instalados en una red LAN pueden intercambiar información con el MSOX. En el caso de que el MSOX tenga una IP pública, cualquier cliente OPC XML con acceso a Internet puede establecer la comunicación con el servidor OPC XML.

IV. PRUEBAS DE COMUNICACIÓN

Para verificar el funcionamiento del MSOX, se realizó una prueba de comunicación entre el MSOX y el cliente comercial *dOPC Explorer. E*n un computador con sistema operativo Linux se instaló el servidor OPC XML, y en otro computador con Windows el cliente *dOPC Explorer,* la ilustración C-12 presenta el esquema de comunicación.

Ilustración C-12: Esquema de validación para el servidor OPC XML



Fuente: propia

A continuación se describe el procedimiento y los resultados obtenidos en la comunicación entre el MSOX y el cliente OPC comercial *dOPC Explorer.*

1. Procedimiento para comunicación entre el MSOX y el cliente *dOPC Explorer*

Para comunicar el MCOX y el cliente *dOPC Explorer* se efectuaron los siguientes pasos:

Instalación del Software necesario

- Se descargó la versión demo del cliente *dOPC Explorer* de la página http://www.kassl.de y se instaló siguiendo los pasos propuestos en el asistente.

- Se Instaló el MSOX según las instrucciones del ítem II de este anexo.

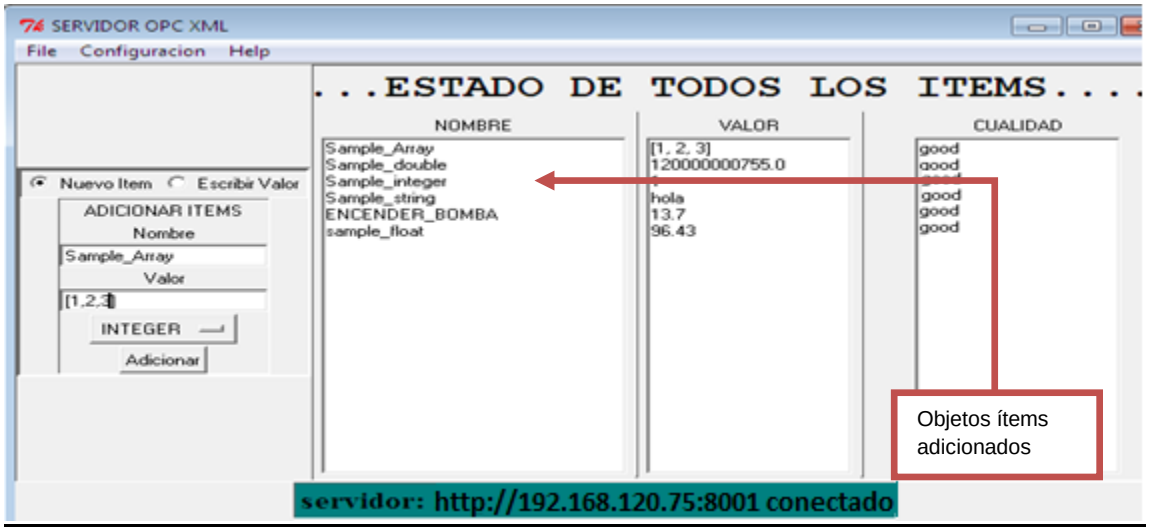
Configuración del MSOX

- Se ejecutó el MSOX como lo indica el ítem III de este anexo.
- Se configuró el puerto 8001 para la comunicación.
- Se adicionaron los siguientes objetos ítems:

Nombre	**Tipo**
Sample _string	string
Sample _integer	Integer
Sample _Float	Float
Sample _Doble	Doublé
Sample_Array	Array
encender_bomba	Float

La ilustración C-13, presenta una pantalla del servidor OPC XML con todos los objetos *ítem*s anteriormente mencionados y configurados.

Ilustración C-13: Pantalla del servidor OPC XML con los objetos ítems configurados

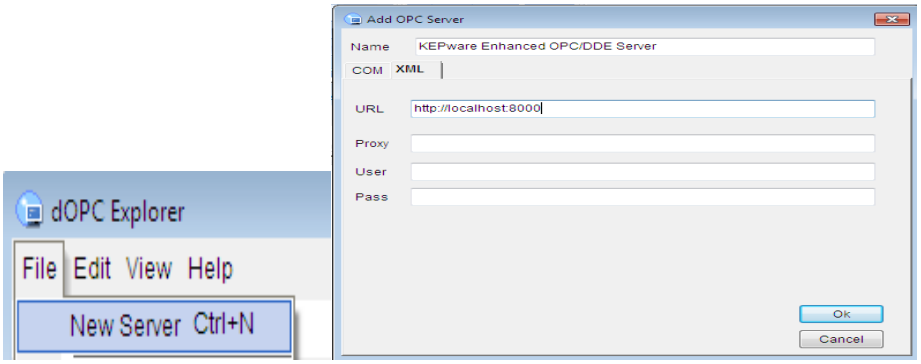


Fuente: propia

Configuración del cliente DOPCExplorer

- Para iniciar el cliente dOPCExplorer se hizo clic en icono dOPCExplorer, donde aparece una pantalla que permite configurar los parámetros del servidor OPC XML, ver ilustración C-15. Para establecer una comunicación con el servidor OPC XML se hizo clic en el menú "file" -"New Server", para que aparezca el formulario donde se puede digitar la dirección del servidor OPC XML, ver ilustración C-15.

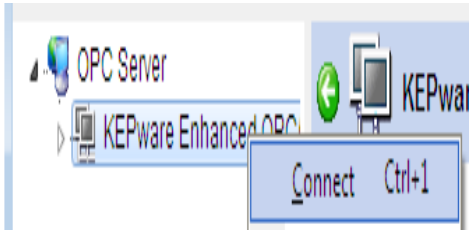
Ilustración C-14: Pasos en el cliente dOPCExplorer para la conexión con el servidor OPC XML



Fuente: propia

- El siguiente paso fue conectarse al MSOX, para ello se hizo clic derecho en el servidor y se presionó "*Connect",* ver ilustración C -16.

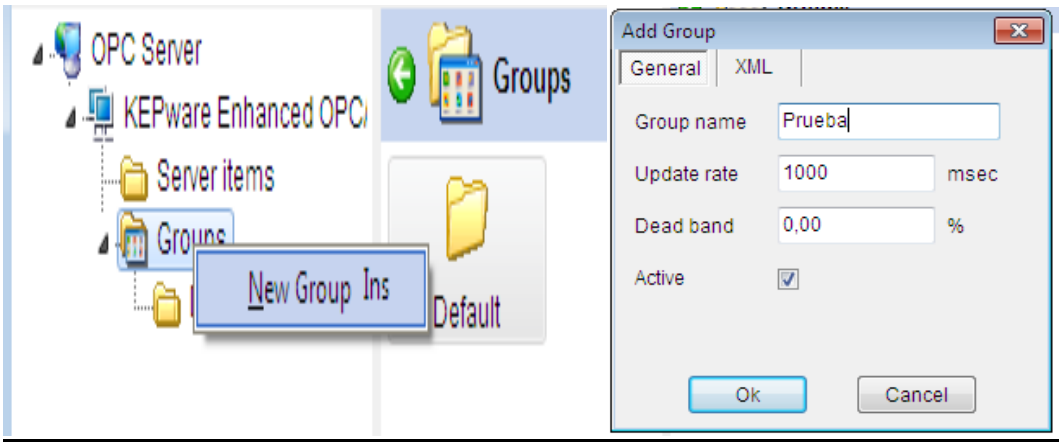
Ilustración C-15: Menú del DOPCExplorer para conectarse a los servidores OPC.



Fuente: propia

- Luego se creó un grupo para contener los objetos ítems del MSOX; para ello se hizo clic derecho en *groups*/ New *group*, y se asignó un nombre al grupo, (en este caso se lo nombro prueba) ver lustración C-16.

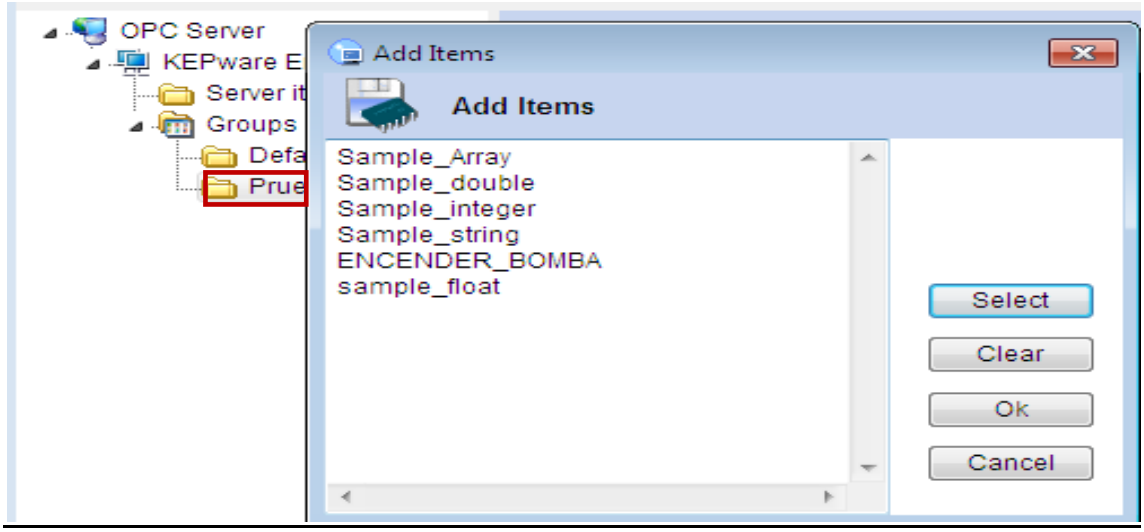
Ilustración C-16: Formulario para la creación de una carpeta en el cliente dOPC



Fuente: propia

- Para adicionar las variables se hizo clic derecho en la carpeta creada, en este caso "prueba", al presionar "*Add Item*" se presentó un formulario que permitió seleccionar los objetos ítems creados en el MSOX, ver ilustración C -18.

Ilustración C-17: Formulario para la creación de un grupo de variables en el cliente dOPCExplorer

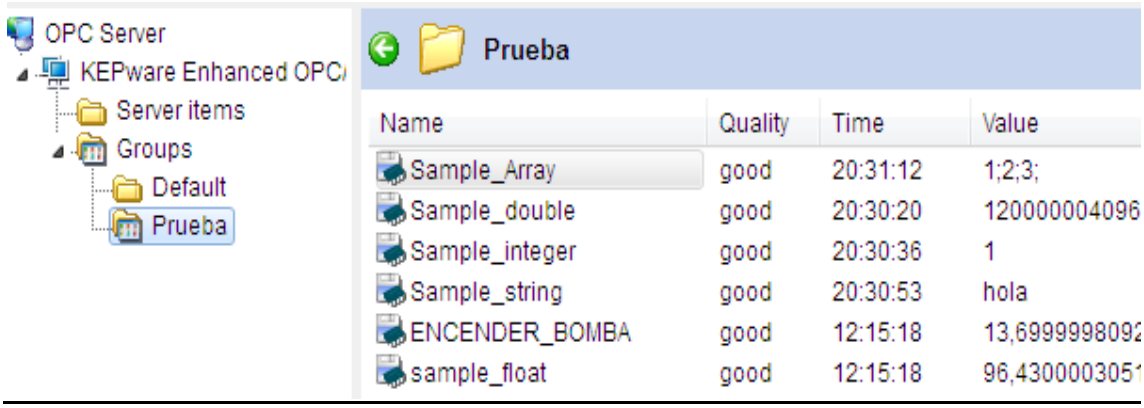


Fuente: propia

Lectura y escritura de objetos ítems

- El siguiente paso consistió en activar la lectura de las propiedades de los objetos ítems importados, se hizo clic derecho en la carpeta "prueba" y se presiona "*Actívate*". Cuando la conexión se estableció correctamente, apareció en el panel de visualización, las propiedades de los objetos ítems, ver ilustración C-18.

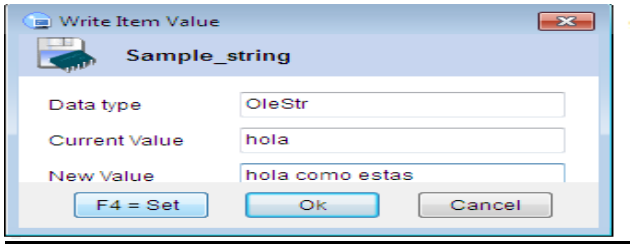
Ilustración C-18: Formulario de visualización de variables en el cliente dOPCExplorer.



Fuente: propia

Para cambiar el valor de cada uno de los objetos ítems el servidor OPC XML, se seleccionó el objeto ítem, se hizo clic derecho y se presionó "*write value*", apareció un formulario donde presenta el tipo de dato, el valor actual y cuadro de texto para ingresar el nuevo valor, ver ilustración C-19.

Ilustración C-19: Formulario de escritura de variables en el cliente dOPCExplorer



Fuente: propia

2. Resultados

- Con la realización de esta prueba se verifico que el MSOX es compatible con los clientes OPC XML.
- se realizó la escritura y lectura de objetos ítems en el MSOX desde el cliente OPC XML sin registrar inconvenientes.
- El tiempo de respuesta del MSOX fue menor a un segundo.
- El MSOX no registro problemas al enviar los tipos de datos: flotantes, enteros, arreglos y cadenas.

ANEXO D. GUÍA DE INSTALACIÓN Y MANUAL DE USUARIO DEL CLIENTE OPC-XML EN MATLAB

I. INTRODUCCION

En el desarrollo de esta guía se presenta la instalación y configuración del Módulo cliente OPC XML en Matlab, el cual permite comunicar bidireccionalmente a Matlab con los servidores OPC XML. Inicialmente se detalla la instalación, configuración y uso del MCOXM, finalmente se describe una prueba de comunicación con el MSOX y se exponen los resultados.

Para la utilización del MCOXM se debe tener en cuenta que: los enlaces OPC XML están pensados para un intercambio de pocos datos, en las variables de tipo "*String*" sólo se soportan los caracteres ASCII válidos del 20 hex al 7F hex.

II. INSTALACION DEL MÓDULO CLIENTE OPC XML EN MATLAB

Para la instalación del módulo cliente OPC XML en Matlab se deben efectuar los siguientes pasos:

- *Instalación de Python:* descargar el instalador de Python 2.5 o superior de la siguiente página: http://www.Python.org/download/ e instalarlo siguiendo las instrucciones presentes en el asistente de instalación.

- Copiar instaladores del servidor: copiar la carpeta “Cliente-OPC_XML_Matlab” que está en la ruta: código_fuente/ al el disco duro del computador, para el caso de ejemplo la carpeta quedara ubicada en el disco duro C.

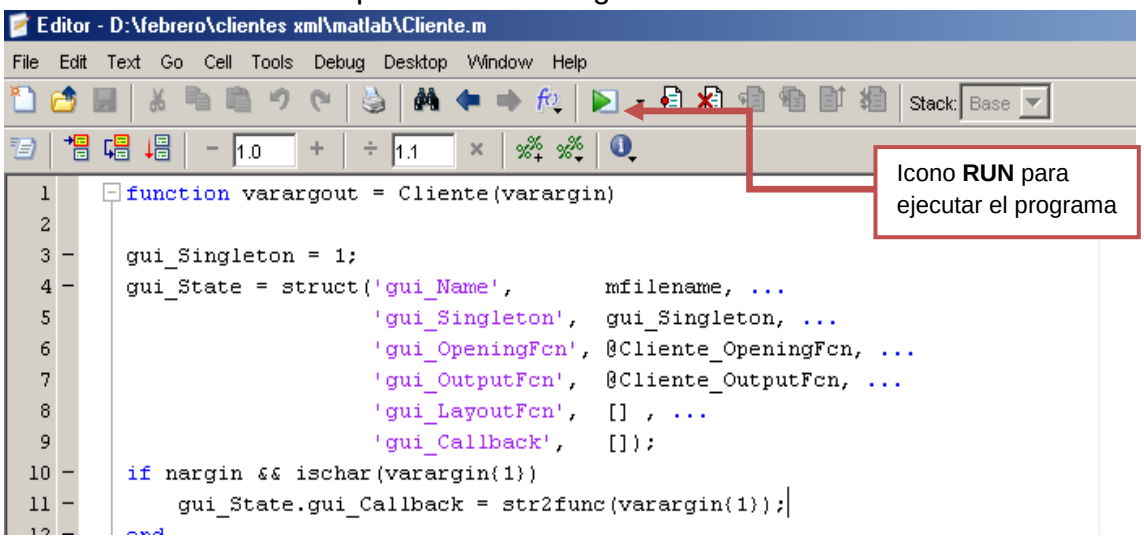
- Adicionar librerías: ubicarse en C:/Cliente_OPC_XML_Matlab/ LibreriasPython y copiar las carpetas “site-packages” y XML en la carpeta Lib que contiene las librerías de Python, para la versión 2.5 de Python la dirección es C:\Python25\Lib, debe aparecer un mensaje que afirma que las carpetas ya existen, se debe remplazar dichas carpetas debido a que se adicionaron y actualizaron librerías.

- Ejecutar el servidor: ubicarse en la carpeta “C:/ Cliente-OPC_XML_Matlab” y dar doble clic en el archivo cliente.m.

III. CONFIGURACION Y USO DEL MCOXM

Al hacer clic sobre el archivo cliente, debe aparecer una ventana del editor de Matlab que permite visualizar el código, ver ilustración D-1. Al ejecutar el programa mediante el icono RUN, aparece la ventana principal del cliente OPC XML con los menús: archivo, importar variables y comandos, ver ilustración D-2.

Ilustración D-1: ventana que indica el código del cliente en el editor de Matlab



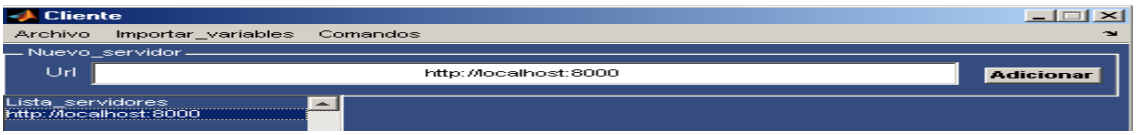
Fuente: propia

Ilustración D-2: ventana principal del cliente OPC XML en Matlab

Fuente: propia

Nuevo cliente OPC XML: para conectarse con un servidor OPC XML se debe ir a "archivo" – "Nuevo", donde aparece un cuadro de texto que permite digitar el nombre del servidor y luego adicionarlo a una lista como lo indica la ilustración D-3.

Ilustración D-3: ventana que permite conectarse a un servidor OPC XML

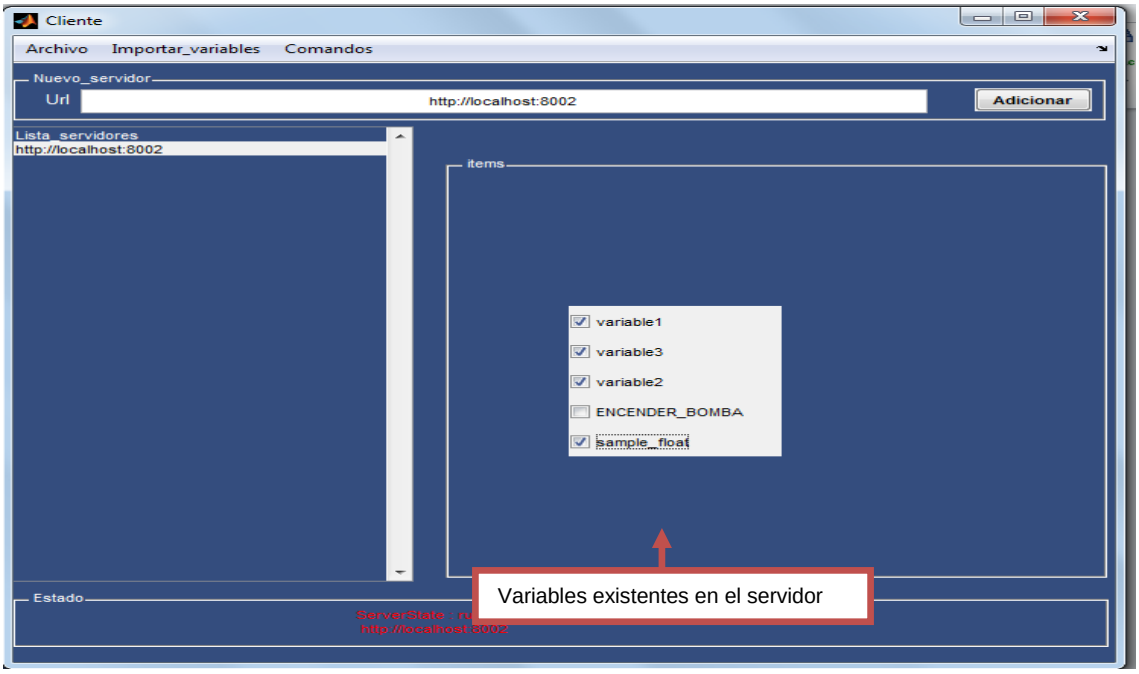


Fuente: propia

Importar variables: Se debe tener en cuenta que las variables deben estar previamente creadas en el servidor OPC XML y este debe estar ejecutándose.
Para seleccionar las variables que se van a utilizar se debe hacer clic en el menú "importar variables", para que presente un formulario con los objetos ítems disponibles en el servidor OPC XML, ver ilustración D-4, en este caso en el servidor OPC XML se han creado cuatro objetos ítems (variable1, variable2, variable3, ENCENDER_BOMBA, sample_float) de las cuales se han seleccionado cuatro.

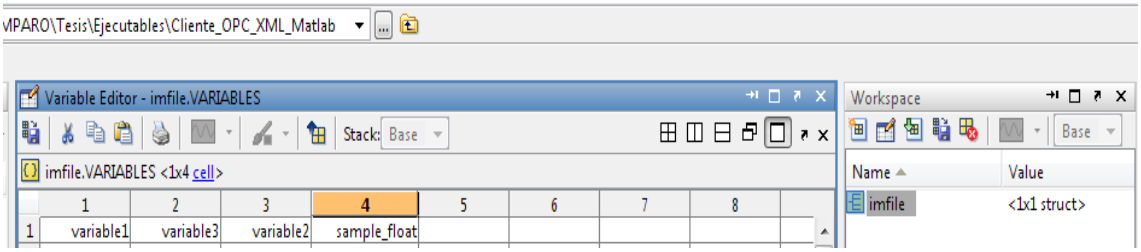
Verificar que todas las variables importadas, se almacenen en el *Workspace* para ser utilizadas en el momento de la lectura y escritura, estas variables se almacena en una estructura llamada **imfile**, ver ilustración D-5.

Ilustración D-4: ventana del cliente OPC XML que permite importar variables



Fuente: propia

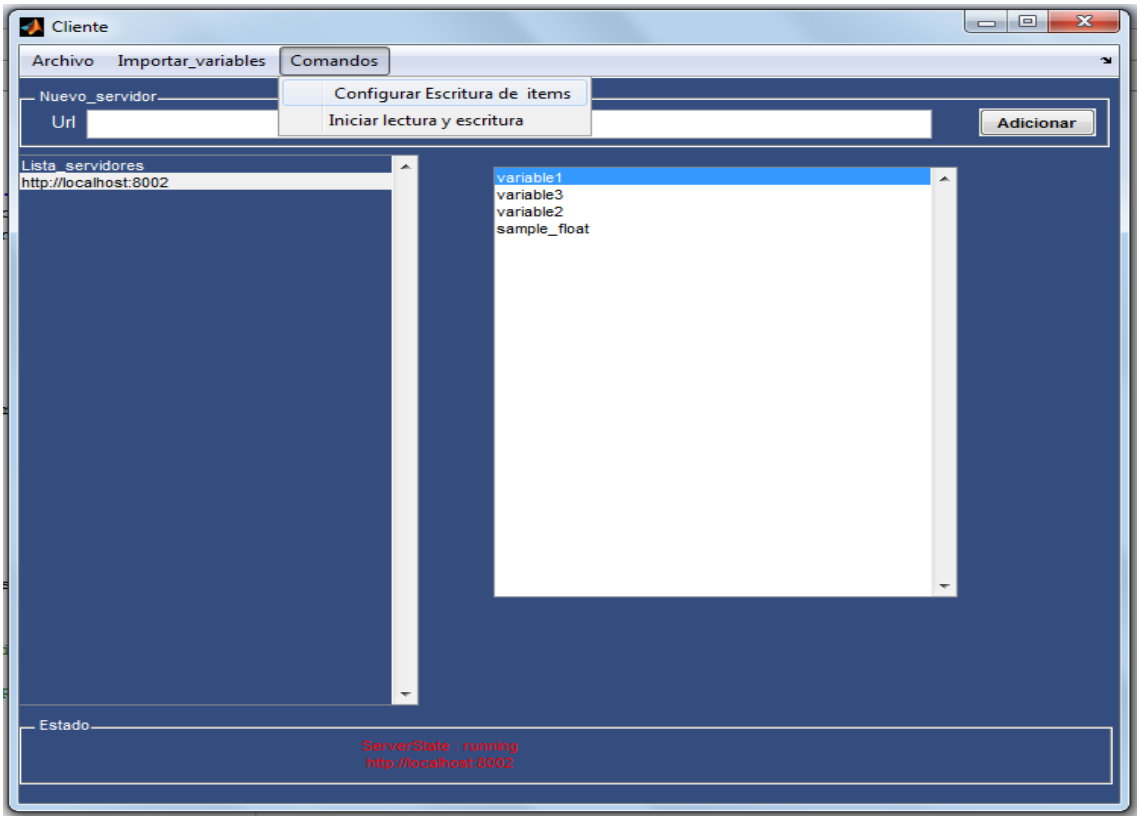
Ilustración D-5: Ventana del Workspace con la estructura de las variables importadas



Fuente: propia

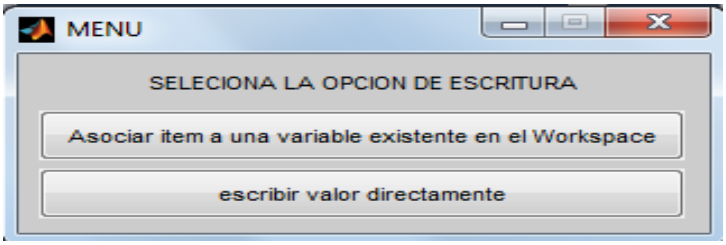
Configurar e iniciar escritura de datos: al seleccionar "configurar escritura de ítems" del menú comandos se presenta una lista con las variables importadas, ver ilustración D-6, al dar clic en cada variable aparece un formulario con dos opciones: asociar a variable existente en el Workspace y escribir valor directamente, ver ilustración D-7.

Ilustración D-6: ventana del cliente OPC XML en Matlab que permite seleccionar las variables para la escritura



Fuente: propia

Ilustración D-7: ventana del cliente OPC XML que permite escribir en los elementos del servidor.

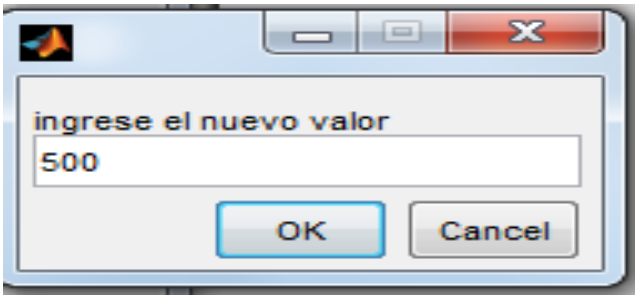


Fuente: propia

Cuando se selecciona la opción “escribir valor directamente”, se presenta un formulario como el de la figura D-8 que permite introducir el valor del objeto ítem en el servidor OPC XML, este cambio solo se realiza cada vez que se haga este procedimiento. Si se quiere escribir continuamente una variable, se debe

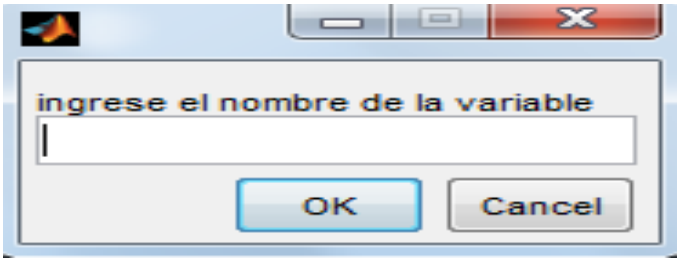
seleccionar la opción “asociar ítem a una variable existente en el Workspace”, donde se presenta un formulario que permite ingresar el nombre de una variable existente en el Workspace, ver Ilustración D-9, así cada vez que la variable en el Workspace cambia, el valor del objeto ítem asociado a esa variable cambia.

Ilustración D-8. Ventana del cliente OPC XML que permite escribir directamente un valor en los ítems del servidor.



Fuente: propia

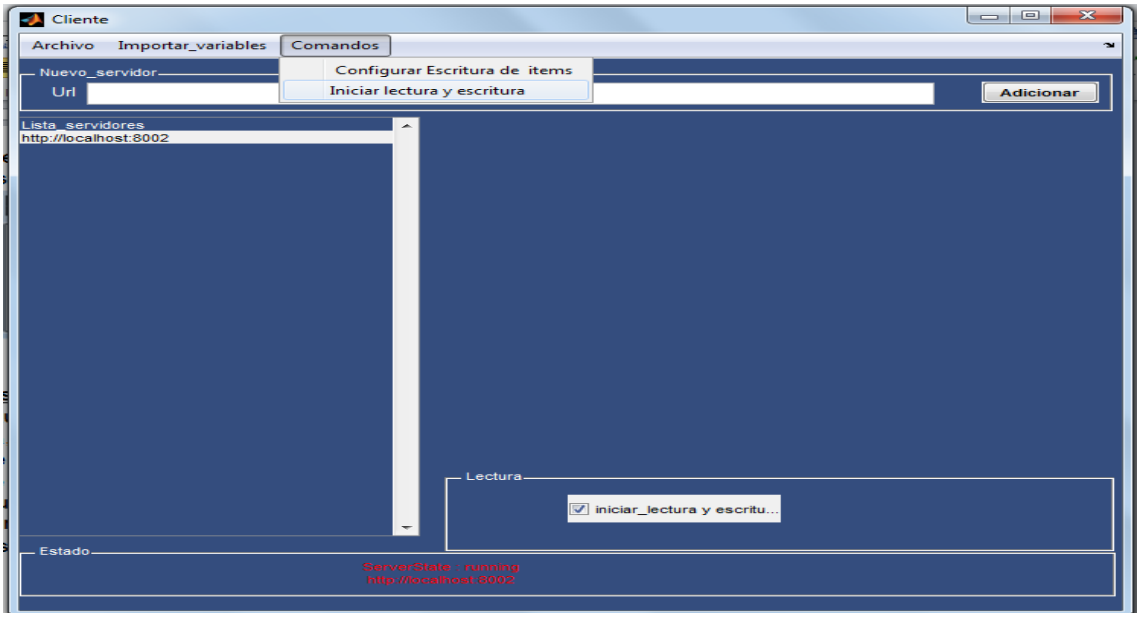
Ilustración D-9. Ventana del cliente OPC XML que permite asociar un objeto ítem a una variable existente en el Workspace.



Fuente: propia

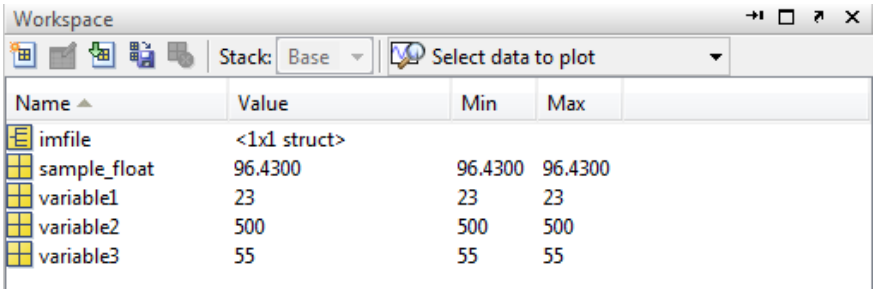
Iniciar lectura y escritura de variables: luego de que se hayan asociado las ítems de escritura con variables en el *Workspace*, se puede iniciar la lectura y escritura de los objetos ítems en forma continua, se debe dar clic en el menú “Comandos” – “Iniciar lectura y escritura” para que se presente un formulario que permite activar el inicio le lectura de los valores de los objetos ítems, ver ilustración D-10, al seleccionar “iniciar lectura”, cada segundo se realiza la toma de datos en el servidor OPC XML, estos valores se almacenan en el *workspace*, ver ilustración D-10, para que puedan ser utilizados por otros programas en Matlab o en Simulink.

Ilustración D-10. Ventana del cliente OPC XML que permite inicializar la lectura de datos.



Fuente: propia

Ilustración D-11. Ventana del *Workspace* que contiene las propiedades de las variables importadas.



Fuente: propia

Si se quiere salir del cliente se debe ir al menu comandos/iniciarlectura y escritura y desactivar la opcion iniciar lectura, hasta que aparezca en la pantalla de comandos “lectura finalizada”, ver ilustracion D-12, y luego archivo/salir. Se debe seguir siempre este procedimiento para finalizar correctamente todas las tareas del cliente que se estan ejecutando en Matlab.

Ilustración D-12: ventana en Matlab que indica que el cliente ha finalizado la lectura y escritura.

```
StopFcn event occurred at 06-Jul-2011 09:54:34
Lectura y escritura finalizada
```

Fuente: propia

IV. USO DE LAS LIBRERIAS OPC XML EN MATLAB

Tambien se puede utilizar las librerias de lectura y escritura en un programa de Matlab independiente, estas librerias se encuentran ubicadas en la carpeta Cliente_OPC_XML_Matlab/librerías_opc_xml_matlab que se copio en el disco duro C. en esta carpeta existen tres funciones implementadas en Matlab, las cuales se explican a continuación:

- La funcion “variables_opc_xml”, permite visualizar los objetos ítems existentes en el servidor, la forma de uso es **[estado, variables]= variables_opc_xml(‘direccion_servidor’).** estado contiene la informacion del servidor y variables contiene un arreglo de todoas los objetos ítems existentes en el servidor.

- La funcion “read_opc_xml” permite visualizar los valores de los objetos ítems existentes en el servidor, la forma de uso es **[estado,variables_f]=read_opc_xml(‘servidor’,variablesr).** esta funcion devueve el estado del servidor y los nombres de las variables con sus valores, se debe tener en cuenta que las variables deben ir separadas por una coma, y dentro de comillas simples, ver ejemplo:

  read_opc_xml('http://localhost:8000','temperatura,otravariable')

- La funcion “write_opc_xml” escribe las variables en el servidor OPC XML, la forma de us es **function write_opc_xml(servidor,variablesw)**, se debe tener encuenta que: el nombre del servidor debe ir entre comillas, y las variables con sus valores tambien, ver ejemplo:
  write_opc_xml('http://localhost:8000','variable1=12,variable2=12')

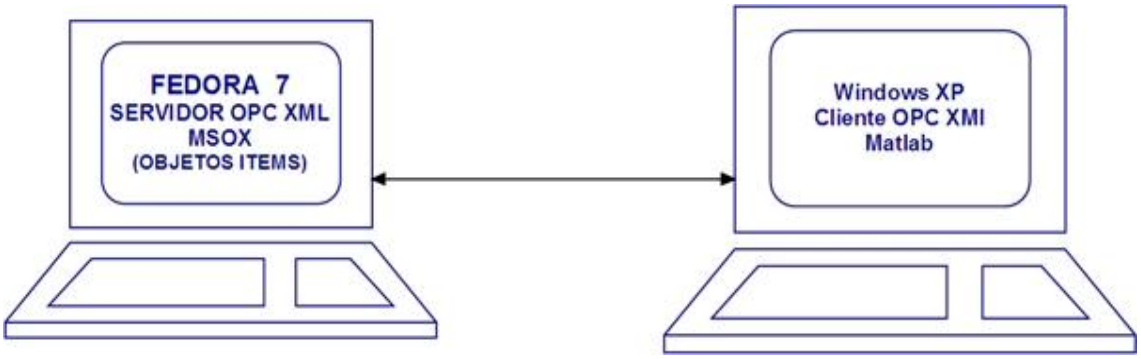
Estas librerias se las debe copiar en la carpeta desde donde se las va a ejecutar.

V. PRUEBAS DE VALIDACION DEL MCOXM

Para verificar el funcionamiento del MCOXM, se realizó una prueba de comunicación con el módulo servidor OPC XML. Se crearon ítems en el MSOX y realizaron pruebas de lectura y escritura desde el MCOXM.

En un computador con sistema operativo Linux se instaló el servidor OPC XML y en otro computador con Windows el MCOXM, el esquema de comunicación se presenta en la ilustración D -13.

Ilustración D-13: esquema de comunicación MCOXM y MSOX



Fuente: propia

Para la realización de esta prueba, se instaló el software necesario en cada computador, se crearon los objetos ítems en el servidor OPC XML, se configuro el módulo cliente OPC XML en Matlab, se iniciaron pruebas de lectura/escritura y se consignaron resultados.

1. Instalación del Software necesario

- Se Instaló el MSOX según las instrucciones del ítem II del anexo C (guía de instalación y configuración del Servidor OPC XML).

- Se Instaló el MCOXM según las instrucciones del ítem II de este anexo (instalación del módulo cliente OPC XML En Matlab).

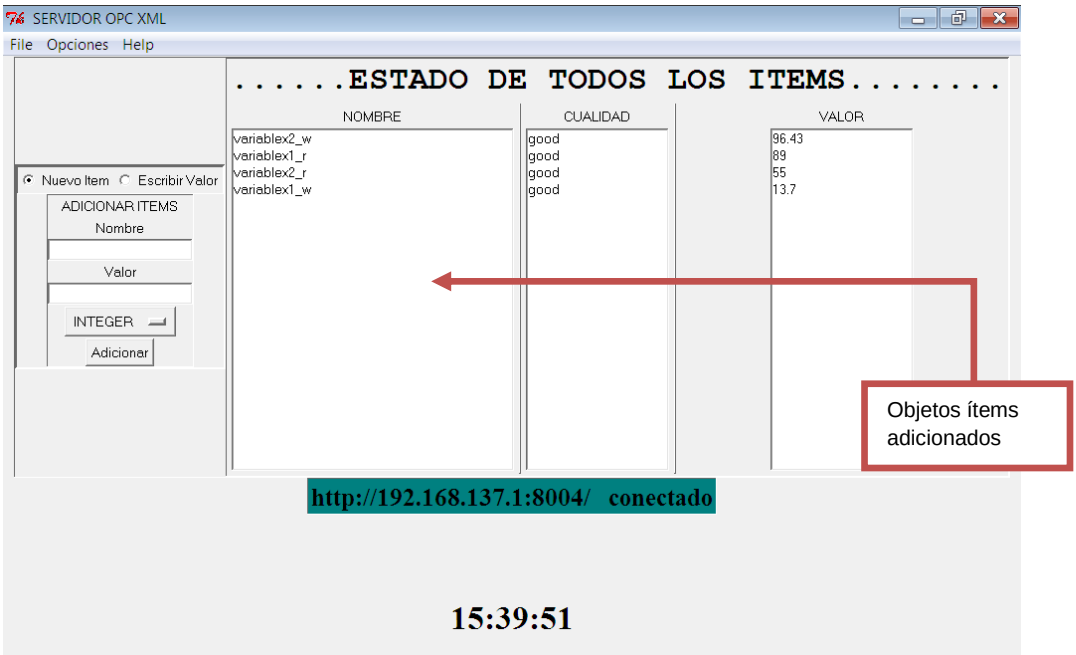
2. Configuración del Módulo Servidor OPC XML

- Según el anexo C (Guía de instalación y configuración del servidor OPC XML) numeral 4, se configuro el puerto de comunicación en servidor OPC XML (en 8004), y se adicionaron los siguientes objetos ítem:

Nombre	Tipo
Variables1_w	Float
Variables2_w	Integer
Variables1_r	Double
Variables2_r	Array

La ilustración D-14, presenta una pantalla del servidor OPC XML con los objetos *ítem*s configurados, a los cuales se les estableció valores con diferentes tipos de datos.

Ilustración D-14: pantalla del servidor OPC XML-configuración de ítems.



Fuente: propia

3. Configuración del Módulo cliente OPC XML en Matlab

Para la configuración del servidor OPC XML se siguieron los siguientes pasos:

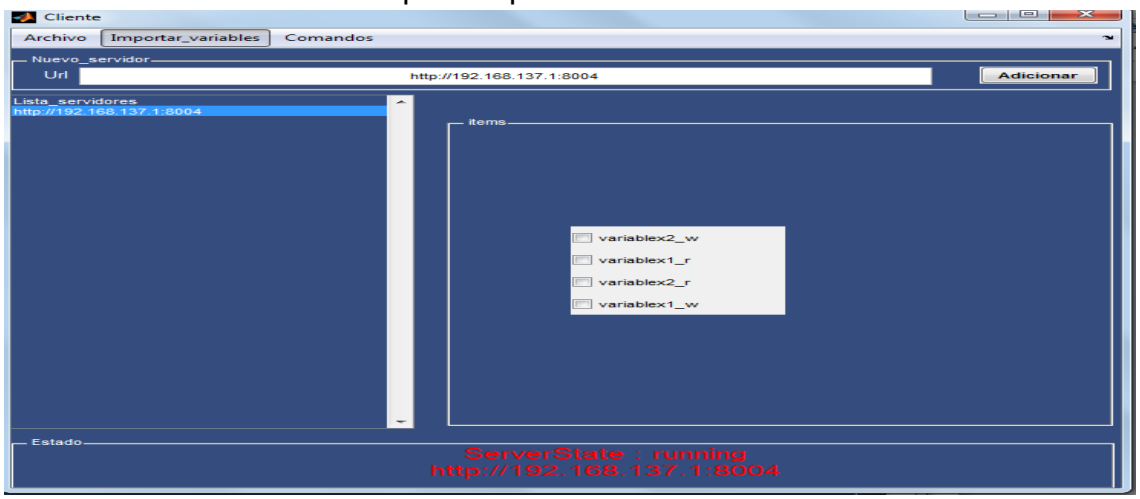
- Iniciar el MCOXM mediante la ejecución del archivo cliente.m (como lo indica la sección III de este anexo).
- Ir a archivo/Nuevo para que aparezca formulario que permita introducir la URL del Servidor.
- Ingresar la dirección del servidor OPC XML, en este caso http://192.168.137.1:8004, ver ilustración D-15.

Ilustración D-15: formulario para el ingreso de la URL

Fuente: propia

El siguiente paso consistió en importar las variables; se hizo clic en menú/importar variables, y se presentó un formulario con los objetos ítems existentes en el servidor, ver ilustración D-16, se seleccionaron las cuatro variables para realizar la prueba.

Ilustración D-16: Formulario para importar variables en el MCOX



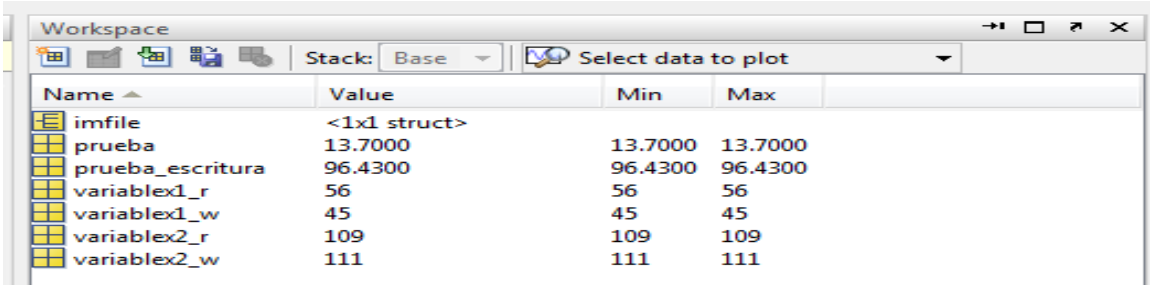
Fuente: propia

4. Lectura y escritura en el módulo servidor OPC XML

Se verifico que la escritura de objetos ítems desde el MCOXM funcionara correctamente, enviando valores fijos directamente a cada objeto ítem.

El paso siguiente fue iniciar la lectura y escritura de los objetos ítems, se cambiaron los valores de los objetos ítems en el servidor OPC, y se verifico que se realizaran los cambios en el *Workspace*, ver ilustración D-17.

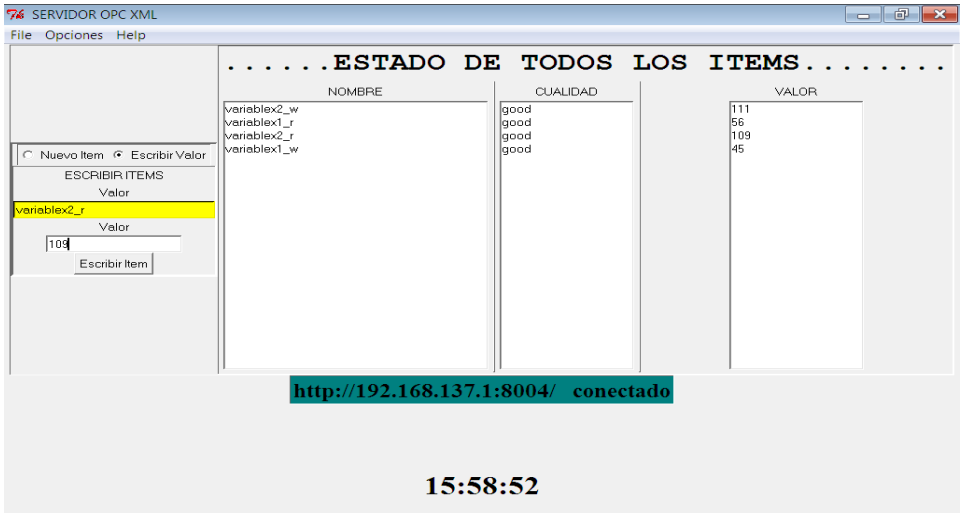
Ilustración D-17: variables en el Workspace



Fuente: propia

Se crearon variables en el *Workspace* y se enviaron datos al servidor OPC XML, en la ilustración D-18 se presenta el servidor OPC XML con los nuevos valores.

Ilustración D-18: Servidor OPC XML con nuevos valores



Fuente: propia

Para verificar que el MCOXM no presentara fallos en el transcurso del tiempo, se realizaron cambios durante una hora aproximadamente.

5. Resultados

- Con la realización de esta prueba se pudo verificar que el MCOXM, es capaz de leer y escribir en los objetos ítems de los servidores OPC XML.

- Desafortunadamente el tiempo de respuesta fue de aproximadamente 3 segundos, esto debido a que están conectados en una red.

- El MCOXM soporta tipos de datos enteros, dobles, flotantes y string (sin espacios).

ANEXO E. GUÍA DE INSTALACION Y CONFIGURACION DE LOS CLIENTES OPC DCOM/XML EN RTAI-LAB

I. INTRODUCCION

En el desarrollo de esta guía se presenta la instalación y configuración de los Módulos cliente OPC en Rtai-Lab, los cuales permiten el intercambio de información entre programas diseñados en Rtai-Lab y los servidores OPC DCOM/XML. Inicialmente se presenta una guía de instalación de los módulos cliente OPC en Fedora Core (2.6.23-rtai), luego se describe la construcción de tareas en Rtai-Lab, y finalmente se detalla una prueba de validación que consiste en intercambiar información entre servidores con diferentes tecnologías (DCOM/XML) a través de los MCODR Y MCOXR.

II. INSTALACION DE LOS MODULOS CLIENTES OPC XML Y DCOM EN RTAI-LAB

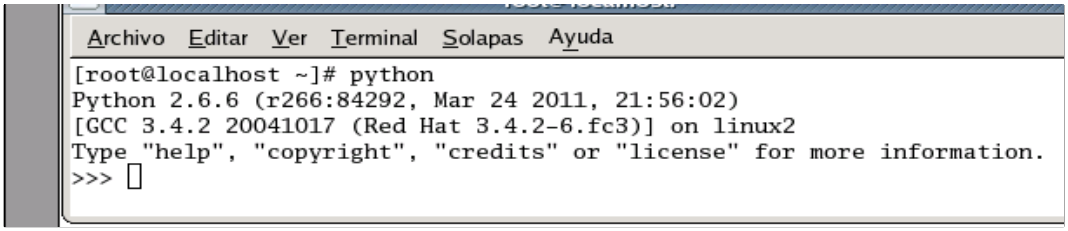
A continuación se describen las condiciones iníciales y los pasos que se deben cumplir para la instalación de los módulos OPC en Rtai-Lab.

- El computador donde se instale los MCODR y MCOXR debe tener el sistema operativo bajo RTAI LINUX, en esta guía se presenta la instalación para Fedora Core (2.6.23-rtai).

- se debe utilizar Scilab/Scicos en su versión 4.1.2, soportada por RTAI.

  Aunque python viene por defecto en todas las versiones de Linux, se debe verificar que Python esté instalado en el equipo donde se va a instalar los MCODR y MCOXR, digitando **#Python** en el terminal, ver ilustración E-1. En caso de que Python no esté instalado se debe descargar e instalar de la página: http://www.Python.org/download/, verificar que la versión sea superior a 2.5.

Ilustración E-1: Verificación de la instalación de Python



Fuente: propia

Para la instalación de los módulos OPC DCOM/XML en Rtai-Lab se deben cargar los módulos de Rtai-Lab, crear los bloques OPC y adicionarlos en una paleta en Scicos. Estos pasos se detallan a continuación:

1. Cargar módulos de Rtai-Lab

Para implementar y ejecutar tareas en Rtai-Lab se deben cargar unos módulos que contienen las librerías necesarias, el primer paso a seguir, es abrir la ventana "Terminal" en la barra de herramientas que aparece en la parte inferior del escritorio, ver ilustración E-2, y digitar la siguiente instrucción:

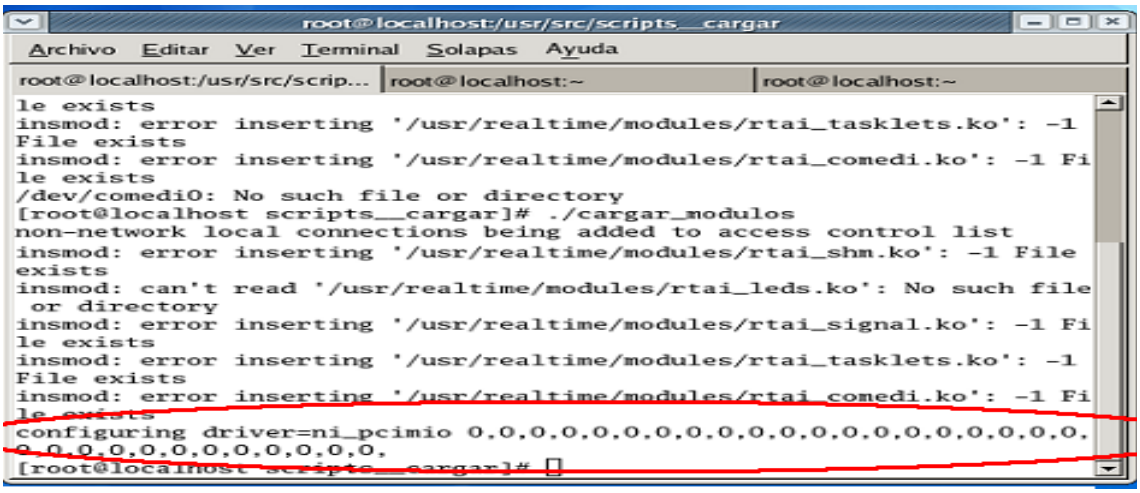
cd /usr/src/scripts__cargar/ (enter)
./cargar_módulos

Ilustración E-2: Apertura de ventana Terminal de Comandos.

Fuente: propia

Esta última instrucción se debe repetir hasta que aparezca en pantalla una fila de ceros 0,0,0…..,0,0 para que carguen los módulos completamente, ver ilustración E-3.

Ilustración E-3: Ventana que indica que los módulos han sido cargados.



Fuente: propia

2. Crear bloques OPC y adicionarlos a la paleta en Scicos

Scicos cuenta con una interfaz gráfica en el que se pueden desarrollar nuevos modelados partiendo de bloques que se encuentran en las paletas (*palettes*). En este apartado se presentan los pasos para adicionar en *Scicos*, la paleta que contiene los cuatro bloques OPC (Dcom_Read, Dcom_Write, Xml_Read, Xml_Write).

Para construir un nuevo bloque en *Scicos* se necesita de dos funciones:

- Una función interfaz que está escrita en código *Scilab*, que se encarga de la interfaz con el usuario.

- Una función computacional para especificar el comportamiento dinámico del bloque. La función computacional puede ser programada en C, Fortran y en lenguaje *Scilab.*

Por lo tanto, para adicionar la paleta OPC en Scicos se deben instalar las librerías asociadas a cada bloque como se describe a continuación:

Instalación de las librerías para cada bloque OPC

- Copiar las cuatro funciones interfaz escritas en scilab: Dcom_Read.sci, Dcom_Write.sci, Xml_Read.sci, Xml_Write.sci, estas funciones que se encuentran en la carpeta **código_fuente/clientes_rtailab/macros**, se las debe copiar en la carpeta **usr/local/scilab-4.1.2-rtailab\macros.**

- Copiar las librerías computacionales (Dcom_Read.c, Dcom_Write.c, Xml_Read.c, Xml_Write.c), que se encuentran ubicadas en la carpeta: **código_fuente/Clientes_Rtai-lab/device** en la carpeta **usr/local/scilab-4.1.2-rtailab/device.**

- Ubicarse en **código_fuente/Clientes_Rtai-lab** y copiar las carpetas **site-packages y XML** en la carpeta donde se almacenan las librerías de python, en el caso de Python2.5 la dirección es: **usr\local\lib\Python25\Lib\site-packages,** en python2.6 la dirección es: **usr\local\lib\Python26\Lib\site-packages.** Al copiarlas se debe presentar un mensaje que indica que las carpetas ya existen, se debe remplazar el contenido, ya que se adicionaron y modificaron algunas librerías.

Compilar librerías

Para que se puedan adicionar los bloques en Scicos, se debe compilar las funciones de interfaz (escritas en Scilab) y computacionales (escritas en C++). Se debe abrir un terminal, ubicarse en el directorio macros de Rtai-Lab y ejecutar "make install ", como lo indica la ilustración E-4.

Ilustración E-4: Instrucciones para compilar las librerías para los bloques OPC

Fuente: propia

Si la compilación no ha generado errores, al finalizar la instrucción se debe visualizar unas líneas de código, como se presenta en la ilustración E-5.

Ilustración E-5: Instrucciones para compilar las librerías para los bloques OPC.

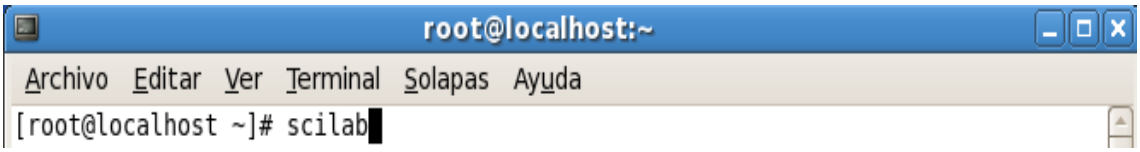
```
ar -r libsciblk.a Dcom_Read.o dcom_write.o Dcom_Write.o getstr.o mbx_ovrwr_send.
o mbx_receive.o mbx_receive_if.o mbx_send_if.o opc_write_dcom.o opc_xml_read.o r
tai_comedi_datain.o rtai_comedi_dataout.o rtai_comedi_dioin.o rtai_comedi_dioout
.o rtai_extdata.o rtai_fifoin1.o rtai_fifoin.o rtai_fifoout.o rtai_led.o rtai_me
ter.o rtai_opcdcomr1.o rtai_opcdcomr.o rtai_scope.o rtai_sem_signal.o rtai_sem_w
ait.o rtai_sinus.o rtai_square.o rtai_step.o Xml_Read.o Xml_Write.o
ar: creating libsciblk.a
cp libsciblk.a /usr/realtime/lib
```

Fuente: propia

Para crear una paleta se debe ingresar a Scilab/Scicos, adicionar los bloques, y agruparlos en una paleta, a continuación se describen estos pasos:

Ingresar a Scilab/Scicos:Para ingresar a Scilab se debe digitar #scilab en el terminal, ver ilustración E-6.

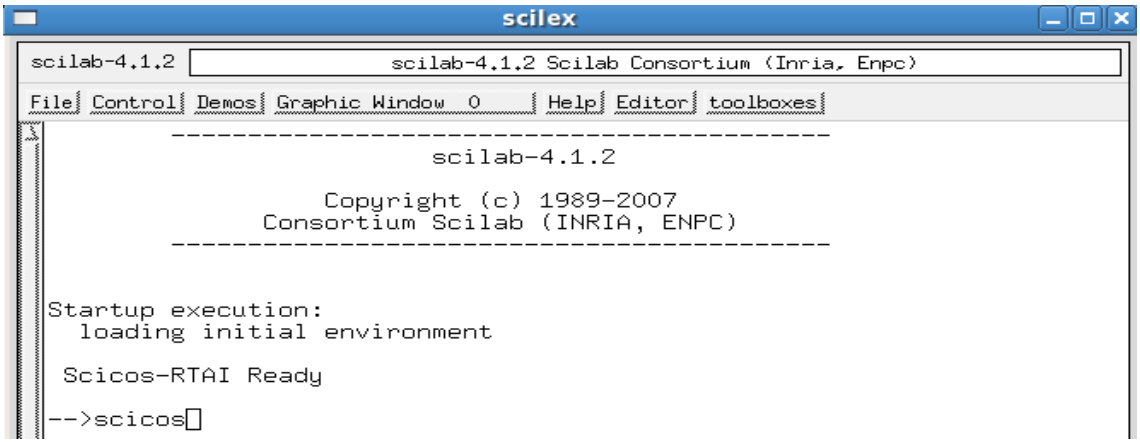
Ilustración E-6: Instrucción en el terminal para ingresar a Scilab.



Fuente: propia

En la ventana de la aplicación, en el *prompt* de *scilab* escribir →*Scicos y presionar enter,* ver ilustración E-7.

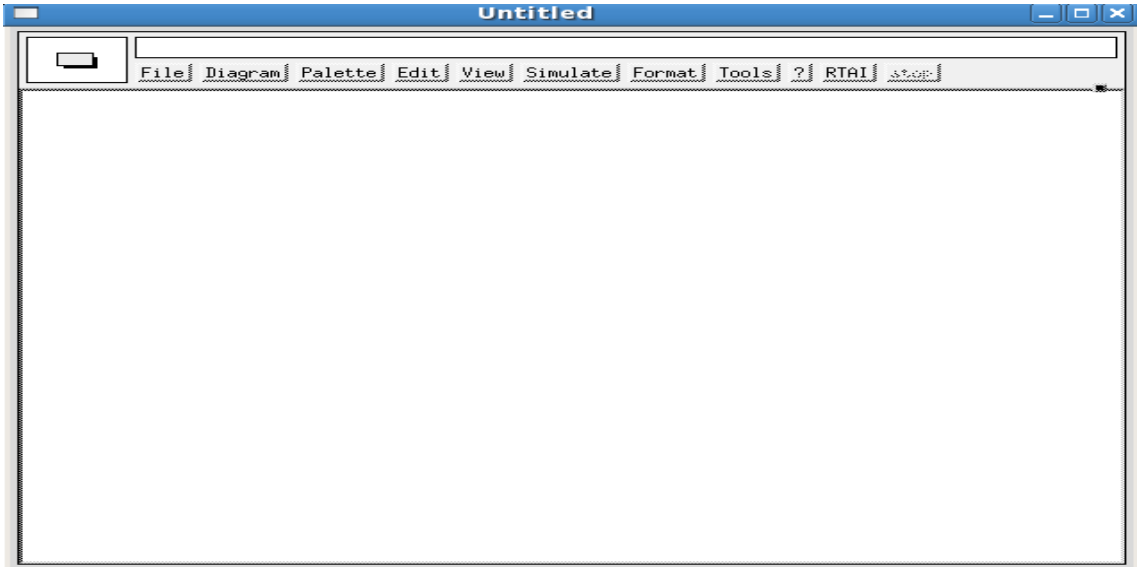
Ilustración E-7: Ingresar a scicos mediante Scilab



Fuente: propia

La ilustración E-8 presenta la ventana principal de scicos; en esta ventana se puede modelar y simular sistemas dinámicos.

Ilustración E-8: Pantalla principal de Scicos



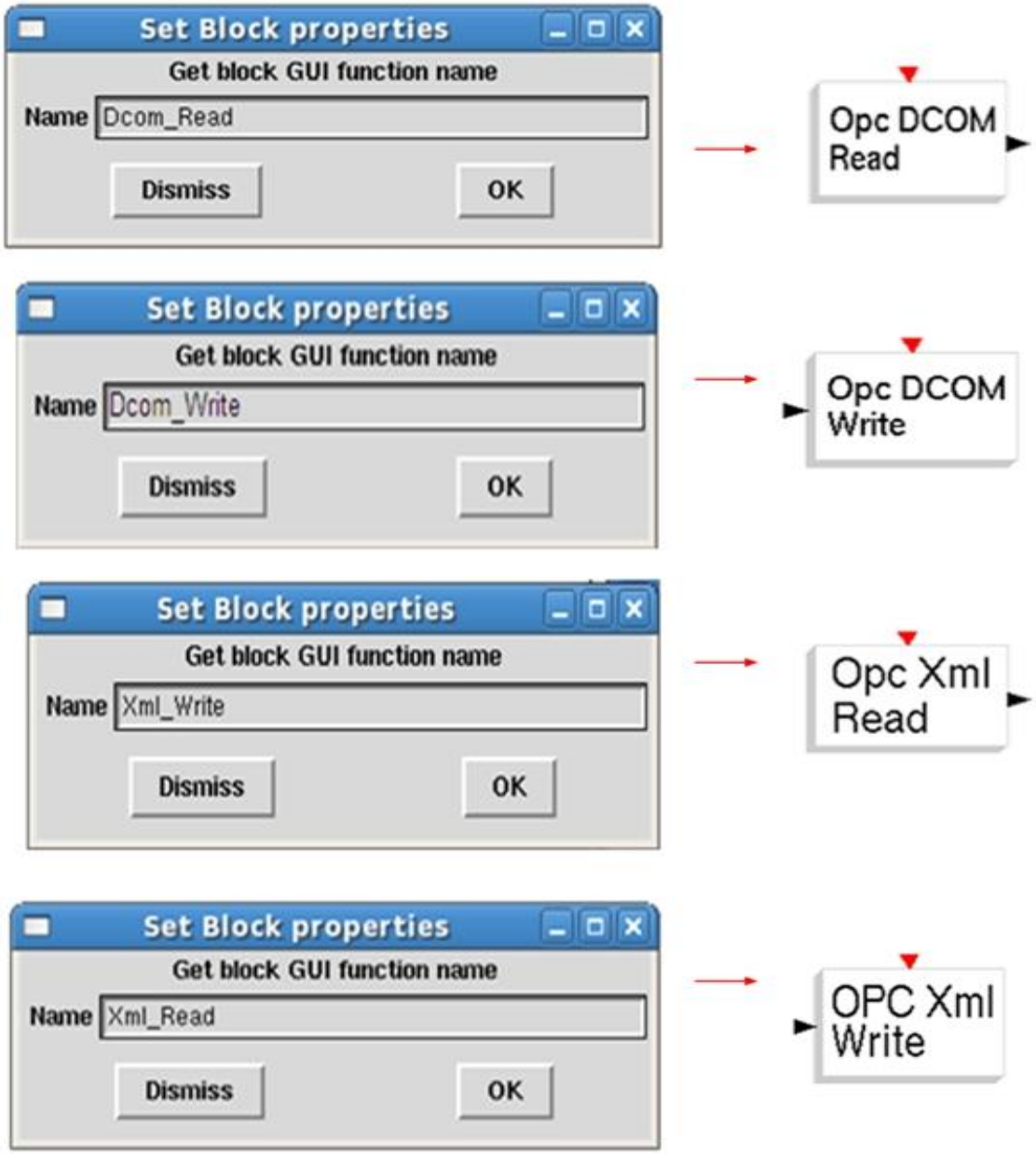
Fuente: propia

Adicionar los bloques OPC en scicos

Para adicionar los bloques OPC en Scicos, se debe hacer clic en **"edit/Add new block"** del menú principal, y digitar el nombre de cada uno de los bloques, ver ilustración E -9, como son cuatro bloques este procedimiento se ejecuta cuatro

veces, teniendo en cuenta que los nombres de los bloques son: Dcom_Read, Dcom_Write, Xml_Read, Xml_Write.

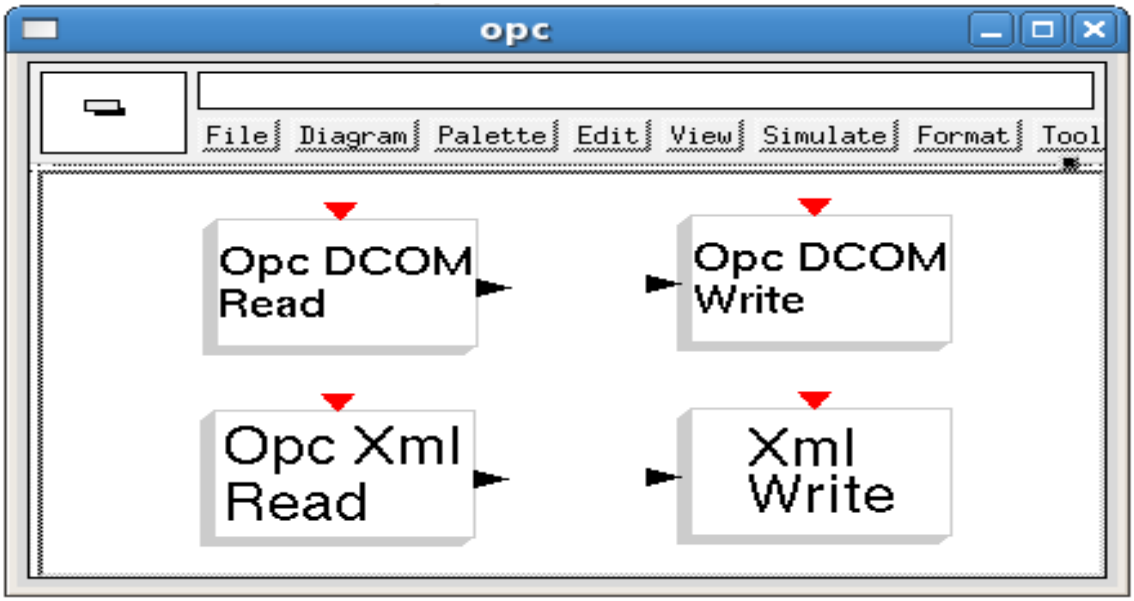
Ilustración E-9: Formularios para adicionar nuevos bloques en scicos



Fuente: propia

Cuando los cuatro bloques se hayan adicionado de forma correcta, debe aparecer una pantalla como indica la ilustración E-10.

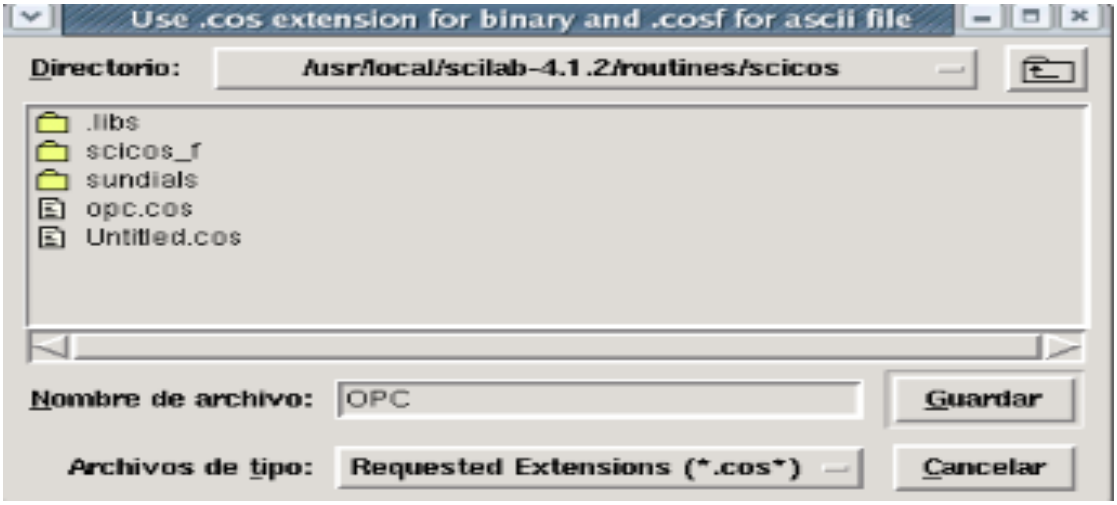
Ilustración E-10: Pantalla con los cuatro Bloques OPC en Scicos



Fuente: propia

Guardar los bloques en una paleta: ir al menú “palette/save as palette” colocar el nombre de la paleta en este caso lo llamaremos OPC, y guardar, ver ilustración E-11.

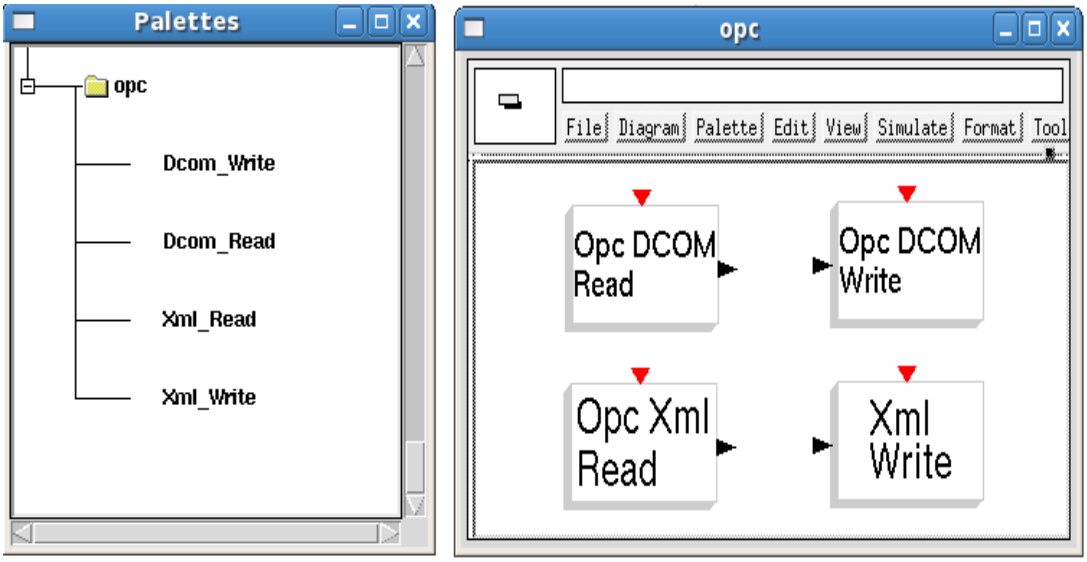
Ilustración E-11: Pasos para crear una paleta de bloques OPC.



Fuente: propia

Si la paleta se ha creado correctamente, en el menú **Paletts/Paletts/** debe aparecer la paleta con los bloques como se indica en la ilustración E-12.

Ilustración E-12: Paleta con los cuatro bloques OPC.



Fuente: propia

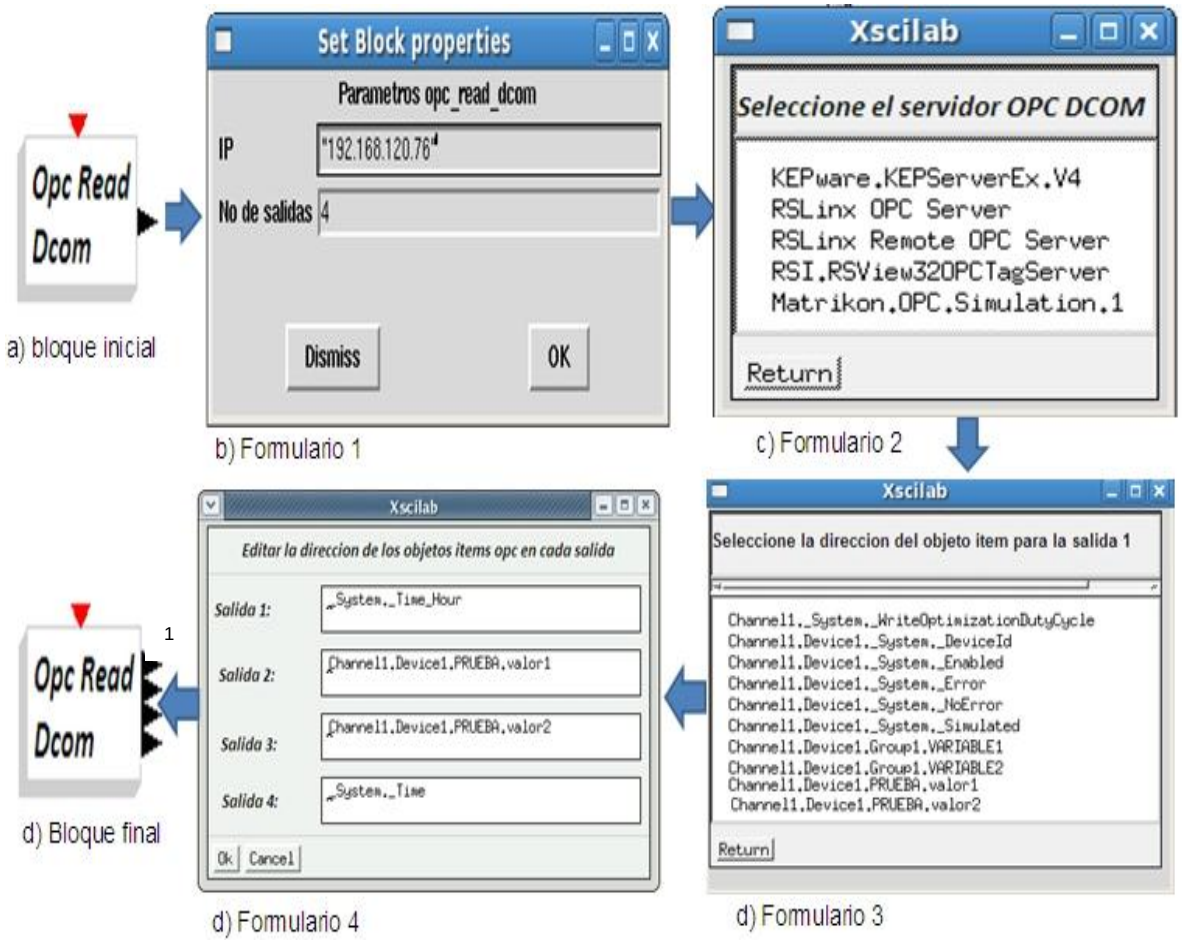
Cuando se pueda visualizar esta paleta, los bloques ya están listos para ser configurados y utilizados por Rtai-Lab. El siguiente paso es configurar los parámetros de cada bloque e interconectarlos con otros bloques del sistema.

III. CONFIGURACIÓN DE LOS PARÁMETROS A LOS BLOQUES OPC

El operario debe seleccionar el bloque de su interés, configurar los parámetros correspondientes y realizar las respectivas conexiones. Al hacer clic derecho sobre el bloque Dcom_Read, ver ilustración E-13a se presenta un formulario, ver ilustración E-13b, que permite definir la dirección del Servidor OPC DCOM y el número de salidas, al presionar ok se presenta el segundo formulario, ver ilustración E-13c, donde indica una lista de servidores disponibles, al seleccionar el servidor con el que se va a trabajar, se presenta un tercer formulario, ver ilustración E-13d, que lista los objetos ítems pertenecientes a dicho servidor, (este formulario se presenta para cada salida) y finalmente se presenta el cuarto formulario, ver ilustración E-13e, que permite editar los objetos ítems finales, se debe hacer clic en OK para obtener el bloque final, ver ilustración E-13f. En el ejemplo presentado en la ilustración 13 se ha configurado el bloque con la

dirección IP igual a 192.168.120.75, y cuatro salidas, en el equipo con esta dirección IP se encuentran instalados cinco servidores de los cuales se seleccionó el KEPServerEX, y se seleccionó la dirección de los cuatro ítems configurados en este servidor, y se generó el bloque con cuatro salidas.

Ilustración E-13: Formularios para establecer los parámetros del bloque Dcom_Read.

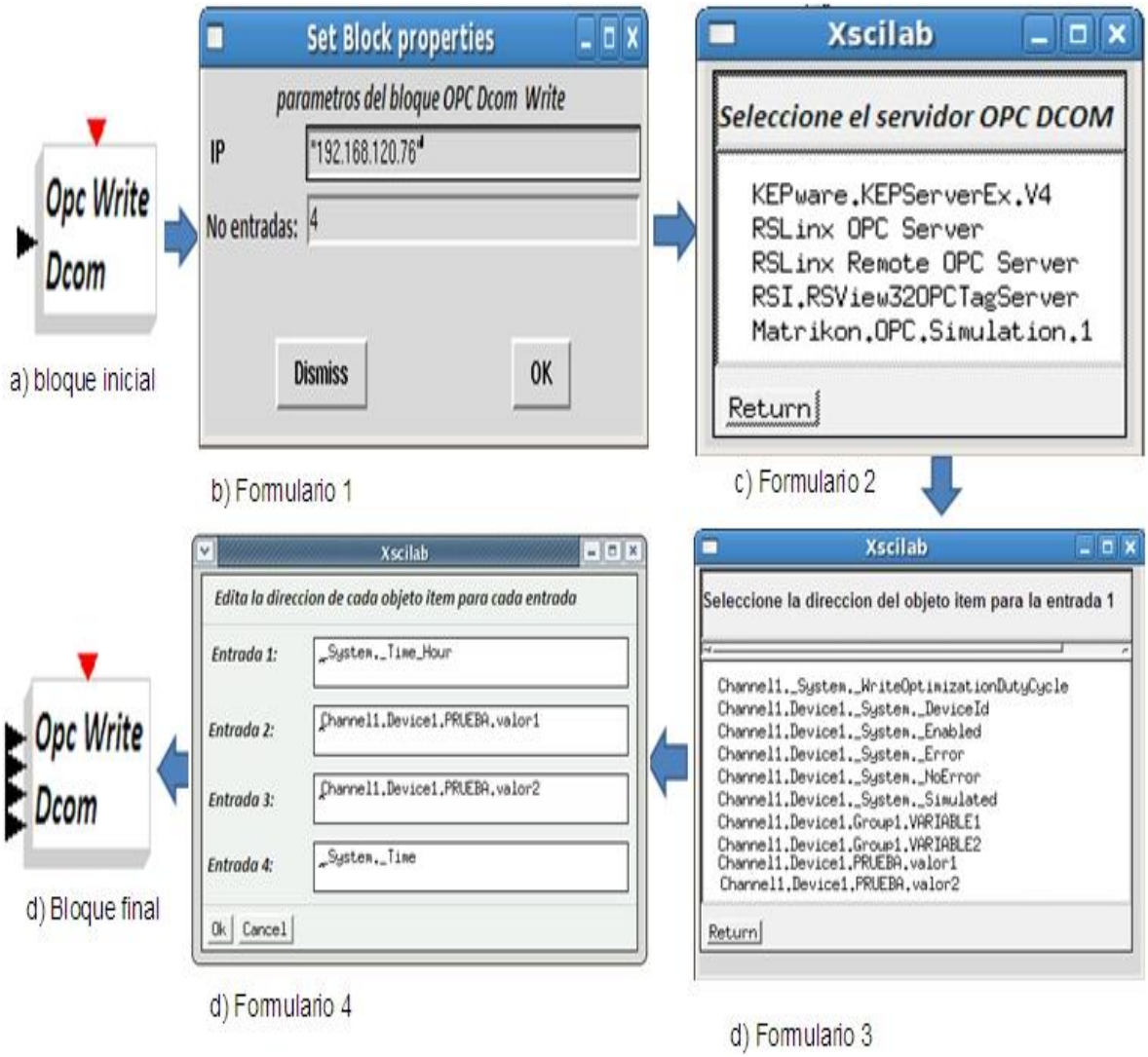


Fuente: Propia.

La ilustración E-14 presenta los formularios necesarios para la configuración de los parámetros del bloque Dcom_Write; en el primer formulario, ver La ilustración E-14b, se debe definir la dirección del Servidor OPC DCOM y el número de entradas, al presionar ok se presenta el segundo formulario, ver ilustración E-14c,

donde indica una lista de servidores disponibles, se debe seleccionar el servidor con el que se va a trabajar para que se presente un tercer formulario, ver ilustración E-14d, que lista los objetos ítems pertenecientes a dicho servidor (este formulario se presenta para cada entrada), se debe seleccionar la dirección de los objetos ítems para cada entrada, al finalizar se presenta el cuarto formulario, ver ilustración E-14e, que permite editar los objetos ítems finales para obtener el bloque final, ver ilustración E-14-f.

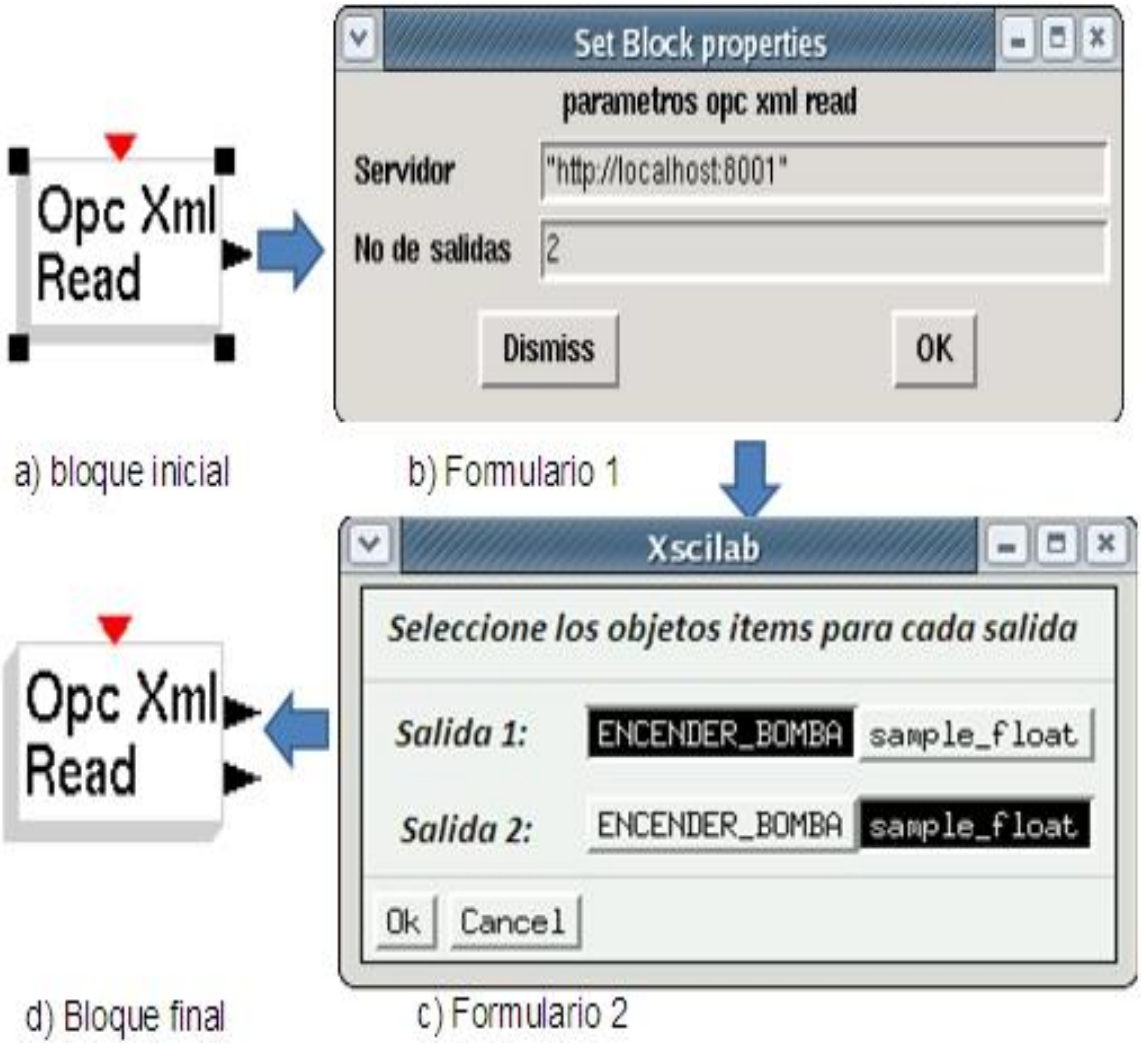
Ilustración E-14: Formularios para establecer los parámetros del bloque Dcom_Write.



Fuente: Propia.

Para la configuración del bloque Xml_Read se deben seguir los siguientes pasos básicos: seleccionar de la paleta de Scicos el bloque Xml_read, ver ilustración E-15a, hacer clic derecho en el bloque para que se presente un segundo formulario, ver ilustración E-15b, que permite ingresar la dirección del Servidor OPC XML y el número de salidas, presionar ok y seleccionar el objeto ítem correspondiente a cada salida y ok, ver ilustración E-15c, para que aparezca el bloque con los nuevos parámetros, ver La ilustración E-15 d.

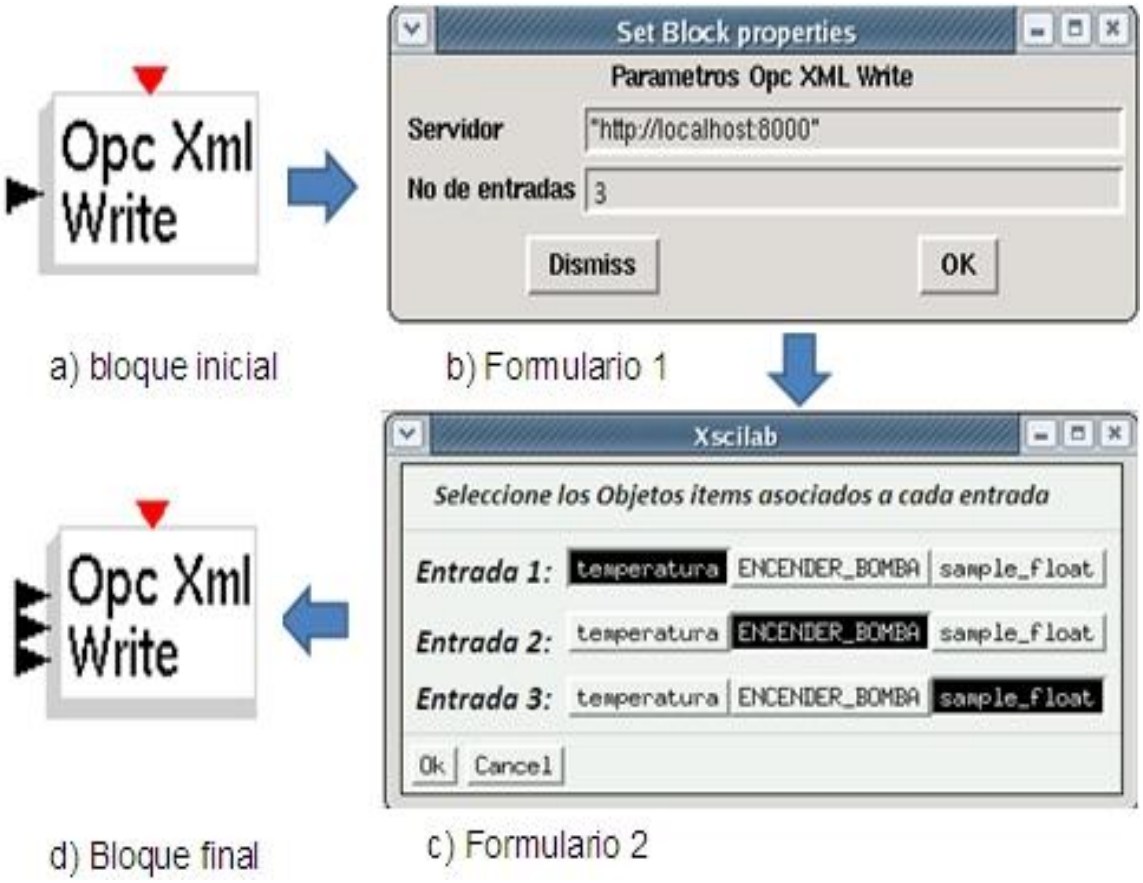
Ilustración E-15: Formulario para establecer los parámetros del bloque Xml_Read



Fuente: **Propia.**

Para la configuración del bloque Xml_Write se deben seguir los siguientes pasos básicos: seleccionar de la paleta de Scicos el bloque Xml_Write, ver ilustración E-16a, hacer clic derecho en el bloque para que se presente un segundo formulario, ver ilustración E-16b, que permite ingresar la dirección del Servidor OPC XML y el número de entradas, si el usuario presiona ok se presenta un tercer formulario, ver ilustración E-16c, con todos los objetos ítems configurados en el Servidor OPC XML, se debe seleccionar el objeto ítem correspondiente a cada entrada, finalmente se diseña y configuran los nuevos parámetros, ver ilustración E-16 d.

Ilustración E-16: Formulario para establecer los parámetros del bloque Xml_Write



Fuente: **Propia.**

IV. CONSTRUCCIÓN DE TAREAS EN TIEMPO REAL

Después de establecidos los parámetros para cada bloque, se inicia la interconexión con otros bloques de Scicos/Rtai-Lab, se debe tener en cuenta que los bloques de Xml_Read y Dcom_Read soportan como máximo cinco salidas, en caso de que se necesite más salidas, se debe adicionar más bloques. Así mismo los bloques Xml_Write y Dcom_Write también soportan como máximo cinco entradas.

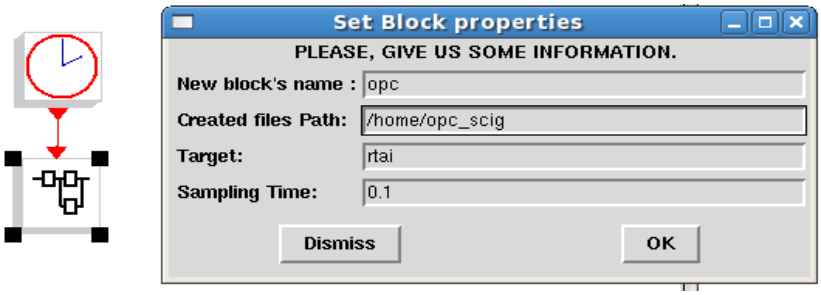
1. Construcción de un modelo en scicos

En Scicos, en el menú *Paletts* con *click* sostenido se selecciona *Palette* o *Pal Tree* y aparecen el conjunto de librerías para construir diagramas de bloques, entre algunas librerías se encuentran: *Sources, Sinks, Linear, Non Linear*, *Events*, Rtai-Lab y la paleta adicionada llamada OPC que contiene los bloques asociados a los clientes OPC.

2. Generación del código para Rtai-Lab

Una vez construido el diagrama de bloques se debe realizar su compilación y generar la tarea de tiempo real asociada al diagrama.
Se selecciona en el menú *Scicos* → *Diagram* → *Region to Super Block*, esta función permite dibujar un marco "elástico" alrededor del diagrama de bloques, pero se debe excluir el evento de reloj, de esta manera se crea un súper bloque asociado al diseño en diagrama de bloques. Se hace *click* en el menú *RTAI* → *RTAI Set Target, se debe* seleccionar el súper bloque generado y aparece la ventana mostrada en la ilustracion E-17, en donde el usuario puede modificar las propiedades del ejecutable.

Ilustración E-17: Opciones para generación de código tarea de tiempo real



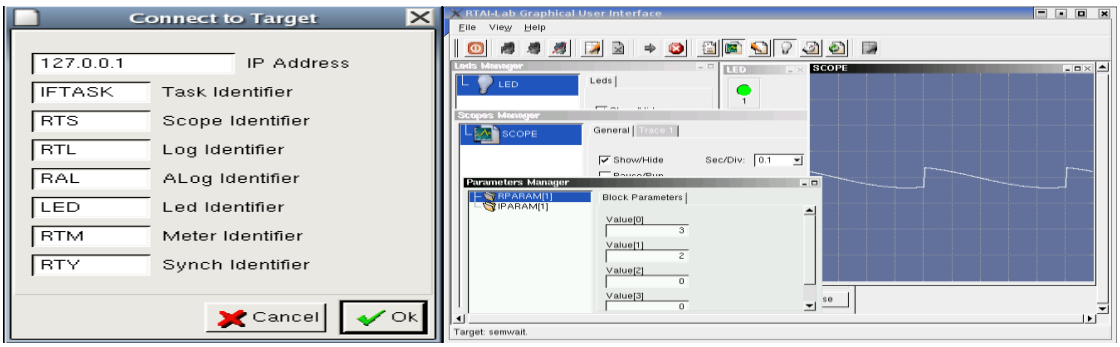
Fuente: Propia.

Para ejecutar una tarea de tiempo real es indispensable dar inicio a dos terminales: en el primer terminal se digita el nombre del ejecutable generado, para el caso general es "***./opc –u"*** o de acuerdo al nombre dado por el diseñador en la opción "*New block's name"* de la figura 3-17; la opción *"–u"* provee instrucciones de uso, como opciones de comando de línea. Para iniciar el ejecutable de tiempo real con tipo de salida "*verbose*", entonces se digita "***./opc –v"***, esta opción muestra las características de la tarea de tiempo real en el terminal. Para ejecutar el osciloscopio *xrtailab*, se debe tener en cuenta que únicamente se ejecuta cuando un *kernel* RTAI Linux está corriendo. *Xrtailab* genera una falla si inicia sobre un *kernel* estándar de Linux. En el segundo terminal se digita "*xrtailab"* para llamar al osciloscopio de Rtai-Lab.

Ahora se debe conectar la tarea de tiempo real en espacio de *kernel* con el osciloscopio del espacio de usuario. Para ello, en la ventana principal de *xrtailab* se selecciona el menú *File ->Connect* o con *Alt+C*, aparece la ventana que se aprecia en el recuadro rojo de la ilustración 46. Se selecciona *OK* y *xrtailab* desde el espacio de usuario se conecta con la tarea de tiempo real espacio del *kernel.*

Dependiendo del tipo de bloques con los que se diseña el diagrama, en la barra de herramientas que provee *xrtailab* se pueden activar *scopes* en donde se pueden ver las trazas o señales adquiridas y enviadas al proceso, también se pueden activar *leds* que indican circunstancias o eventos del proceso como por ejemplo alarmas, o activar *meters* que son indicadores de variables o señales; también se puede activar el administrador de parámetros (*Parameters Manager*) que permite modificar constantes y valores de los parámetros en tiempo real; dichas opciones se aprecian en el recuadro azul de la imagen E -18 .

Ilustración E-18: Ventana de Comunicación Rtai-Lab con una tarea de tiempo real.

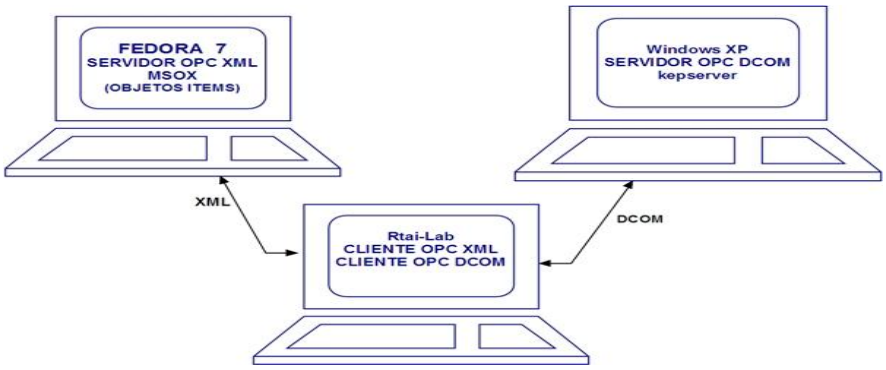


Fuente: Propia.

V. PRUEBAS DE VALIDACION DE LOS BLOQUES OPC PARA RTAI-LAB

Para verificar el correcto funcionamiento de los clientes OPC en Rtai-Lab se diseñó una prueba que consiste en intercambiar información entre los servidores OPC de diferentes estándares (KEPserverEX y el Módulo servidor OPC XML). El objetivo de esta prueba es demostrar que al cambiar un valor en un objeto ítem en cualquiera de estos servidores el cambio del valor se ve reflejado en el otro servidor, el esquema de comunicación de esta prueba se presenta en la ilustración E-19.

Ilustración E-19: Esquema de comunicación entre Rtai-Lab y los servidores OPC



Fuente: Propia.

Para realizar esta prueba se realizaron los siguientes pasos: se instaló el software necesario, se crearon los objetos ítems en cada servidor OPC, se crearon los diagramas de bloques correspondientes haciendo uso de la paleta OPC, se generó el código para la tarea en tiempo real, se ejecutó la tarea en tiempo real, y se realizaron las pruebas de lectura y escritura en los servidores OPC.

1. Instalación del software necesario

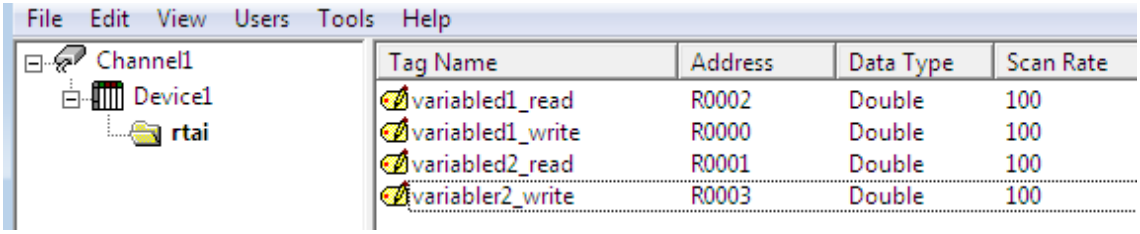
- Se instaló el servidor KEPserverEXEX según las instrucciones de la sección II del anexo B (Manual de usuario del módulo servidor OPC DCOM KEPServerEx)
- Se instaló el MSOX según las instrucciones de la sección II del anexo C (Guía de instalación y manual de usuario del módulo servidor OPC-XML).

- Se creó la paleta en OPC en scicos, según la información obtenida del numeral II de este anexo.

2. Configuración del Servidor OPC DCOM

Según el anexo B sección III-2 (manual de usuario del módulo servidor OPC DCOM KEPServerEx) se configuro en el servidor OPC DCOM KEPserverEX, cuatro variables tipo doble: variabled1_write, variabled2_write, variabled1_read, variabled1_read. La ilustración E-20 presenta una pantalla del servidor OPC KEPserverEX con la configuración de las variables descritas anteriormente.

Ilustración E-20. Ítems configurados en el Servidor OPC KEPserverEX

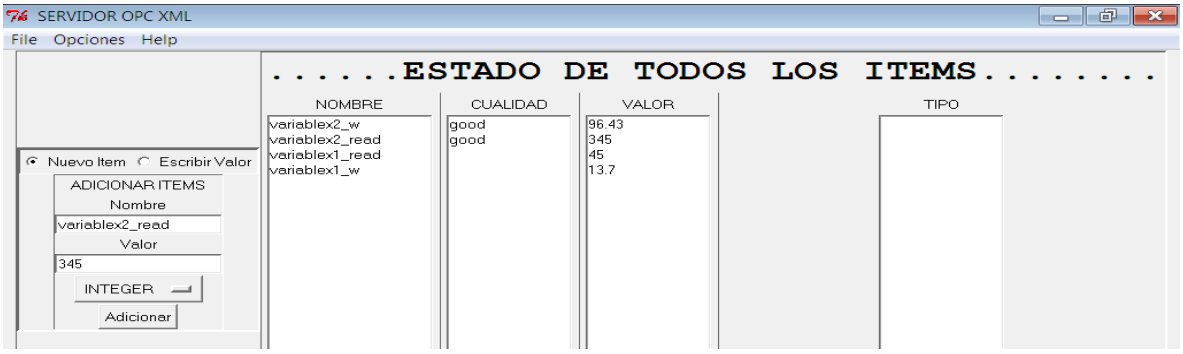


Fuente: propia

3. Configuración del Servidor OPC XML

Se configuro en el servidor OPC XML según el anexo C – III (guía de instalación y manual de usuario del módulo servidor OPC-XML) las cuatro variables de tipo doble: variablex1_w, variablex2_w, variablex1_read, variablex1_read, ver ilustración E-21.

Ilustración E-21. Ítems configurados en el MSOX



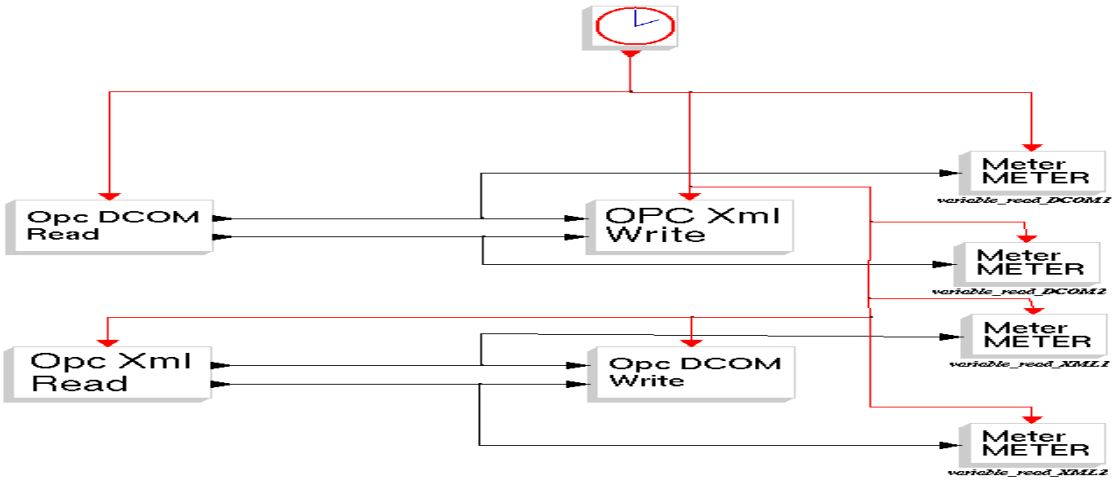
Fuente: propia

4. Diseñar los diagramas de bloques

Para crear los bloques se ingresó a Scicos siguiendo los pasos del numeral III-1 de este anexo, y se diseñó el diagrama de bloques expuesto en la ilustración E-22.

La información proveniente de las variables: variabled1_read, variabled2_read, se envían al bloque que se encarga de escribir estos valores en el MSOX (variables: variablex1_w, variables: variablex2_w), y La información proveniente de las variables: variablex1_read, variablex2_read, se envían al bloque que se encarga de escribir estos valores en el MSOX (variables: variabled1_write, variabled2_write). Así mismo los valores de las salidas se envían a un meter para la visualización en la pantalla de Rtai-Lab.

Ilustración E-22. Diagrama de bloques en Scicos



Fuente: propia

Se configuraron los parámetros en cada bloque (según el numeral III- 4 de este anexo) de la siguiente manera:

Bloque Dcom_Read:

- IP del servidor:192.168.120.76
- Servidor: KEPserverEX
- Salidas: 2
- Objetos ítems: salida1 (variabled1_read), salida2 (, variabled2_read).

Bloque Dcom_Write:

- IP del servidor: 192.168.120.76
- Servidor: KEPserverEX
- Entradas: 2
- Objetos ítems: entrada1 (variabled1_write), entrada2 (variabled2_write)

Bloque XML_Read:

- Dirección del servidor: http://192.168.120.74:8004
- Salidas: 2
- Objetos Ítems: salida1 (variablex1_read), salida2(variablex2_read)

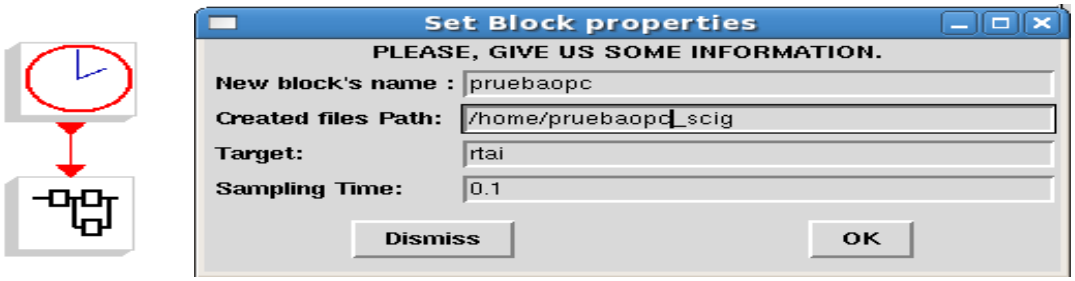
Bloque XML_Write:

- Dirección del servidor: http://192.168.120.74:8004
- Entradas: 2
- Objetos ítems: entrada1 (variablex1_w), entradas 2(variablex2_w)

5. Generar el código para Rtai-Lab

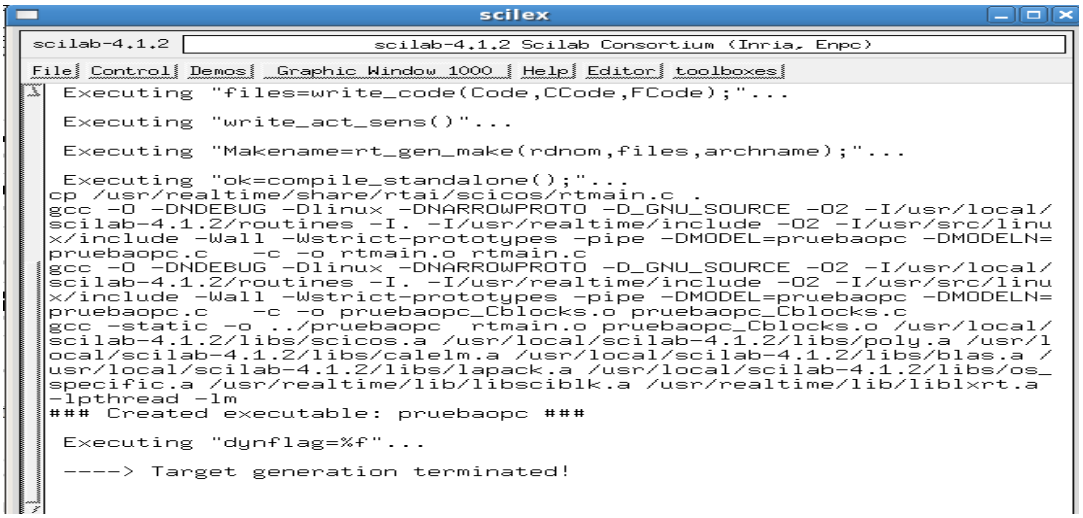
Luego de creado el diagrama de bloques, se creó un súper bloque y se generó el ejecutable con el nombre “pruebaopc”, ver ilustración E-23.

Ilustración E-23. Parámetros para generar el código para Rtai-Lab



Fuente: propia

Ilustración E-24. Pantalla que indica que el código para Rtai-Lab se ha generdo.

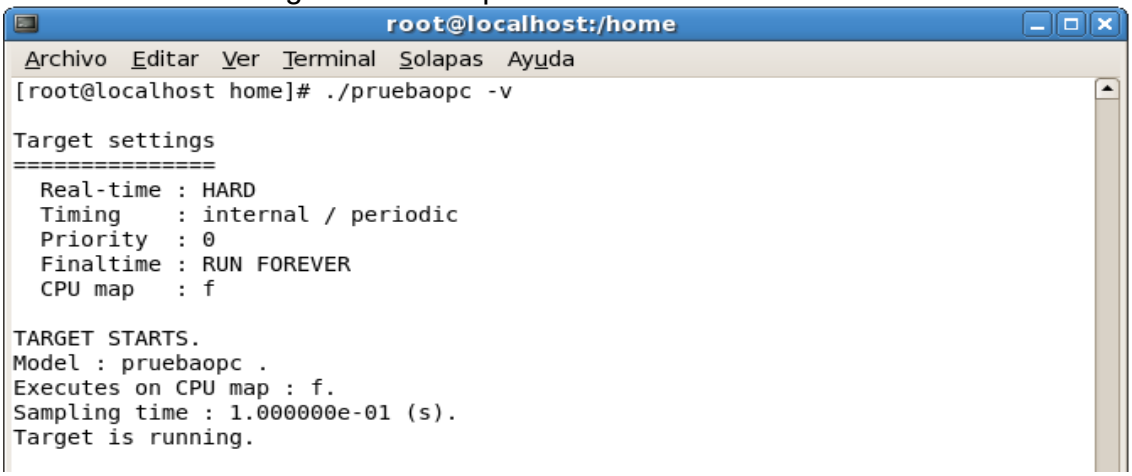


Fuente: propia

6. Ejecutar la tarea

En un terminal se ejecutó la instrucción**. /pruebaopc –v** (ver ilustración E-25).

Ilustración E-25. Diagrama de bloques en Scicos

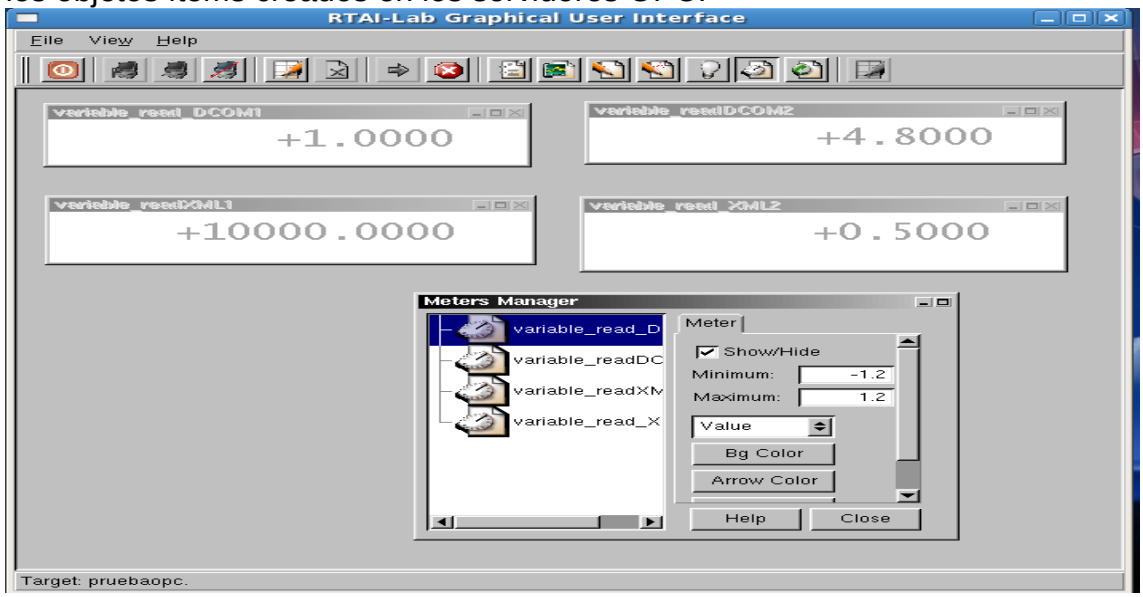


Fuente: propia

7. Leer y escribir objetos ítems en los servidores OPC

Desde los clientes de cada servidor se modificaron los datos los cuales se vieron reflejados en el osciloscopio de Rtai-Lab, ver ilustración E-26.

Ilustración E-26: osciloscopio de Rtai-Lab que permite visualizar los valores de los objetos ítems creados en los servidores OPC.



Fuente: propia

8. Resultados

- Con la realización de esta prueba se logró la comunicación entre servidores OPC de diferentes estándares.
- Se pudo comprobar el funcionamiento de los bloques Dcom_Read, Dcom_Write, Xml_Read, Xml_Write, y la interconexión de estos bloques con otros bloques existentes en Scicos.
- Los MCODR, MCOXR solo soportan datos de carácter numérico.
- Se observó que la comunicación OPC DCOM presenta mejor tiempo de respuesta que la OPC XML.

ANEXO F. GUÍA DE INSTALACIÓN Y MANUAL DE USUARIO DEL MODULO DE MONITOREO Y SUPERVISIÓN WEB (MM&SW)

En esta guía se presenta la instalación y el manual de usuario para el módulo de monitoreo y supervisión web el cual consta de tres sub-módulos básicos que permiten crear representaciones dinámicas del proceso, configurar alarmas, eventos, y tendencias.

I. INSTALACION DEL MM&SW

En el equipo servidor del laboratorio de control de procesos del PIAI, en el disco local **D**, existe una máquina virtual **hmiwebopc**, donde se encuentra instalado el MM&SW, si la máquina virtual no está disponible, se debe instalar los servidores, y librerías necesarias como se indica a continuación:

- Instalación servidor web Wampserver: descargar la última versión de Wampserver en la página: http://www.wampserver.com/download.php, hacer clic en el ejecutable, seleccionar el directorio donde se va a instalar (por defecto c: wamp) y seguir las instrucciones que presenta el asistente de instalación.

- Copiar archivos en el directorio del servidor: ir a la carpeta **código_fuente/ Modulo_monitoreo_supervision** y copiar la carpeta "**hmiwebopc"** en el directorio raíz del servidor: **c:wamp/www**

- Copiar la estructura de la base de datos: ir a la carpeta **código_fuente/ Modulo_monitoreo_supervision** y copiar la carpeta que contiene la estructura de la base de datos "**joomla"** en el directorio de Mysql que en este caso es: **c:wamp/ww/bin/mysqlmysql51.36/data**

- Verificar que el servidor web está funcionando ingresando a la siguiente dirección: http://localhost/hmiwebopc/

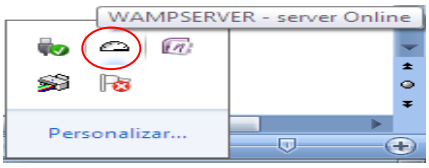
- Instalación de Python: descargar el instalador de Python 2.5 o superior de la siguiente página: http://www.Python.org/download/ e instalarlo siguiendo las instrucciones del asistente de instalación.

- Adicionar librerías para los clientes OPC: ubicarse en **código_fuente/ Modulo_monitoreo_supervision** y copiar las carpetas "site-packages" y XML en la carpeta **Lib** que contiene las librerías de Python, para la versión 2.5 la dirección es C:\Python25\Lib, debe aparecer un mensaje que afirma que las carpetas ya existen, se debe remplazar dichas carpetas debido a que se adicionaron y actualizaron librerías.

Verificar que el servidor Wampserver este inicializado: ir a la parte derecha de la barra de tareas dar clic en 

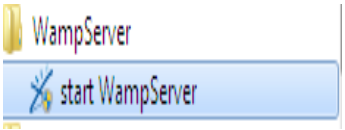
donde aparecerá el icono  , debemos confirmar que este en estado "online" para que se puedan visualizar las paginas desde un equipo remoto, ver ilustración F-1. Si el icono de wampaserver no se encuentra, se debe ir a inicio **wampserver/start wampServer**, ver ilustración F-2, y ejecutarlo.

Ilustración F-1. Icono que indica el servidor web está activado.



Fuente: Propia.

Ilustración F-2. Pantalla para iniciar el servidor WEB.

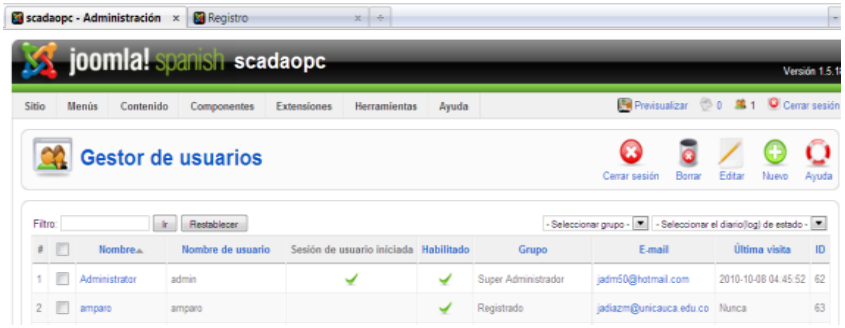


Fuente: Propia.

II. CONFIGURACION Y USO DEL MM&SW

El módulo de monitoreo y supervisión web, está conformado por el panel de control (para administradores), y las páginas de usuario. El administrador puede ingresar al panel de control, mediante la URL http://ip_servidor/hmiwebOPC/administrator y obtener los siguientes privilegios: gestionar usuarios, crear menús, editar contenido, adicionar componentes, configurar parámetros de presentación, entre otros, ver ilustración F-3.

Ilustración F-3: panel de administración del MM&SW



Fuente: propia

Los operarios, administradores e invitados pueden ingresar al módulo de monitoreo y supervisión web, digitando la URL que corresponde a**:** **http://ip/hmiwebopc/** en un navegador web. Por ejemplo si la dirección del computador es 192.164.125.1 la URL será: **http://192.164.125.1/hmiwebopc/.** Dentro de los navegadores se recomienda Mozilla, Opera, Google Croome. En internet Explorer solo funciona en las versiónes iguales o superiores a la 8.

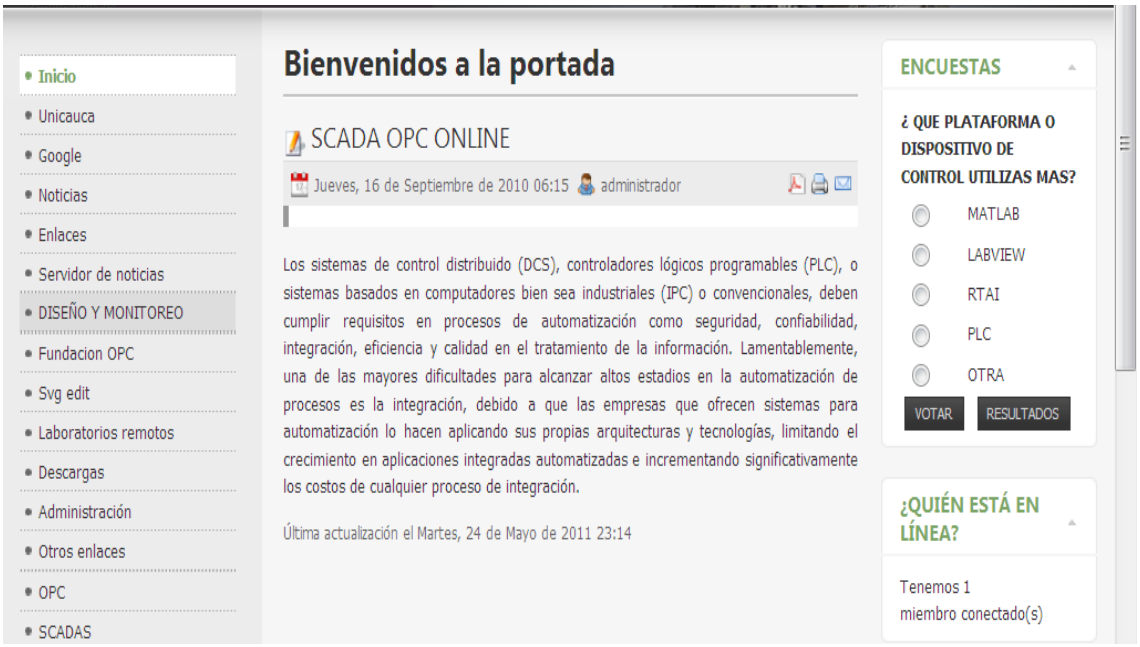
Al ingresar la URL en el explorador, se presenta una página de inicio, ver ilustración F-4, la cual está compuesta por un menú principal, y un formulario para que los usuarios registrados ingresen sus datos y tengan acceso a información y funciones restringidas para los usuarios invitados.

Ilustración F-4. Página principal del sistema HMI

Fuente: Propia.

Para que se active en el menú principal los enlaces para diseño de proyectos, encuestas, usuarios en línea entre otros se debe ingresar el nombre de usuario y la contraseña, ver ilustración F-5.

Ilustración F-5. Página del sistema HMI para usuarios registrados



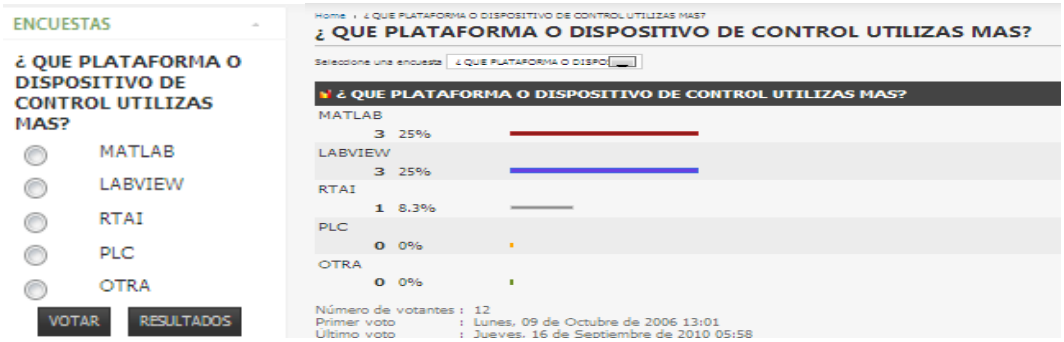
Fuente: Propia.

Los usuarios registrados pueden editar los artículos, ver ilustración F-6, diseñar encuestas ver ilustración F-7, y utilizar todas las herramientas que joomla presenta (actualización de noticias, anuncios, entre otros, ver el manual de usuario de Joomla (3).

Ilustración F-6. Edición de artículos en el sistema HMI

Fuente: Propia.

Ilustración F-7. Selección de encuestas



Fuente: Propia.

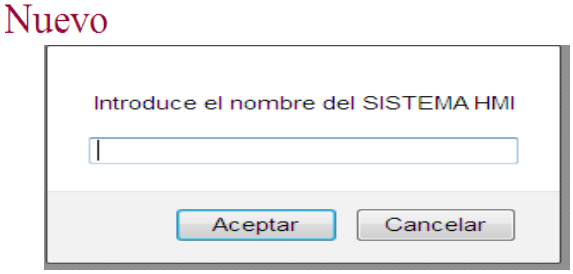
Para ingresar al sub-módulo de diseño y monitoreo, se debe hacer clic en el enlace “diseño y monitoreo” del menú principal. Al presionar este enlace aparece una página con el menú: nuevo, eliminar, editar, ver ilustración F-8.

Ilustración F-8: Menú principal de la pantalla de diseño.

Fuente: propia

El primer paso es establecer el nombre del proyecto, haciendo clic en la opción “nuevo”, para que se presente un formulario como el que se indica en la ilustración F-9.

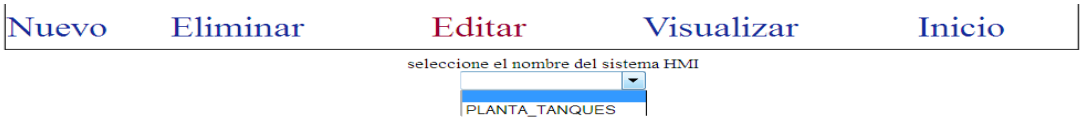
Ilustración F-9: Formulario para crear un nuevo HMI



Fuente: propia

Luego de establecido el nombre del proyecto, se puede iniciar con el diseño de esquemas, al hacer clic “editar” del menú principal, se presenta un formulario que permite seleccionar el proyecto con el que se va a trabajar, ver ilustración F-10.

Ilustración F-10: Menú “Editar” del sub-módulo de diseño en el MM&SW



Fuente: propia

Al seleccionar el nombre del proyecto, se presenta otro menú con las opciones: Fuente de Datos, Diseño gráfico, Config Eventos, Config. Alarmas y Ayuda, ver ilustración F-11.

Ilustración F-11: Menú “Editar” del sub-módulo de diseño

Fuente: propia

A continuación se detalla cada ítem del menú “editar”.

Fuente de datos: antes de empezar a diseñar los esquemas de los procesos, se debe seleccionar la fuente de los datos, importar los objetos ítems de los Servidores OPC y almacenarlos en la base de datos para su posterior uso. Al seleccionar “Fuente de datos” se presenta un formulario con las opciones: servidor OPC DCOM, servidor OPC XML u otra, ver ilustración F-12.

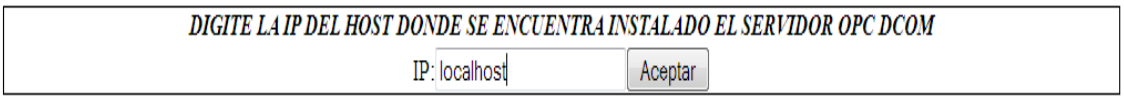
Ilustración F-12: Menú principal para la selección de la fuente de datos

Fuente: propia

Si se selecciona como fuente de datos “servidor OPC DCOM”, se presenta un formulario que permite digitar la dirección IP del host donde se encuentra el

servidor OPC DCOM, ver ilustración F-13. Luego de que se ingresa la IP se presenta un formulario que permite seleccionar el nombre servidor, ver ilustración F-14, al seleccionar el nombre del servidor se presenta un tercer formulario que permite establecer la ruta completa donde está el grupo de ítems OPC DCOM, ver ilustración F-15, Finalmente un cuarto formulario permite visualizar todos los objetos ítems existentes en el objeto grupo del servidor OPC, ver ilustración F-16, se debe elegir los objetos ítems que se van a utilizar y hacer clic en guardar.

Ilustración F-13: Formulario para ingresar la dirección IP del host donde está el servidor OPC DCOM.



Fuente: propia

Ilustración F-14: Formulario que permite seleccionar los servidores disponibles en el equipo.

Fuente: propia

Ilustración F-15: Formulario para ingresar la ruta donde está el grupo de objetos ítems

Fuente: propia

Ilustración F-16: Formulario que permite seleccionar los objetos ítems

SELECCIONE LAS VARIABLES

SERVIDOR	Matrikon.OPC.Simulation.1/			
Name	Adicionar	Cambiar nombre	unidades	Write
prueba.variable1	☑	variable1w	cm	☑
prueba.variable2	☑	variable2r	cm	☐
prueba.variable3	☑	variable4w	m	☑

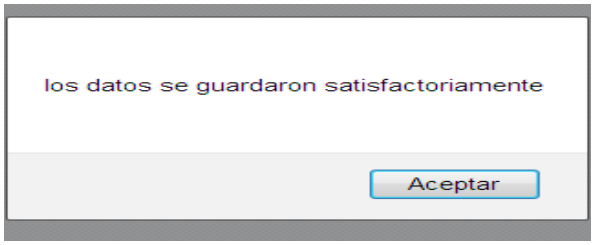
Guardar | Visualizar todos

VARIABLES EXISTENTES					
Direccion OPC	Modo	Tipo	datatype	Nombre	Eliminar
prueba.variable2	read	prueba.variable2	string	adminPLANTA_TANQUESvariable2r	☐
prueba.variable3	read	prueba.variable3	string	adminPLANTA_TANQUESvariable4w	☐
prueba.variable1	read	prueba.variable1	string	adminPLANTA_TANQUESvariable1w	☐

Fuente: propia

Al hacer clic en guardar, debe aparecer un formulario como el que se indica en la ilustración F-17.

Ilustración F-17: Formulario que afirma que los datos se guardaron correctamente.



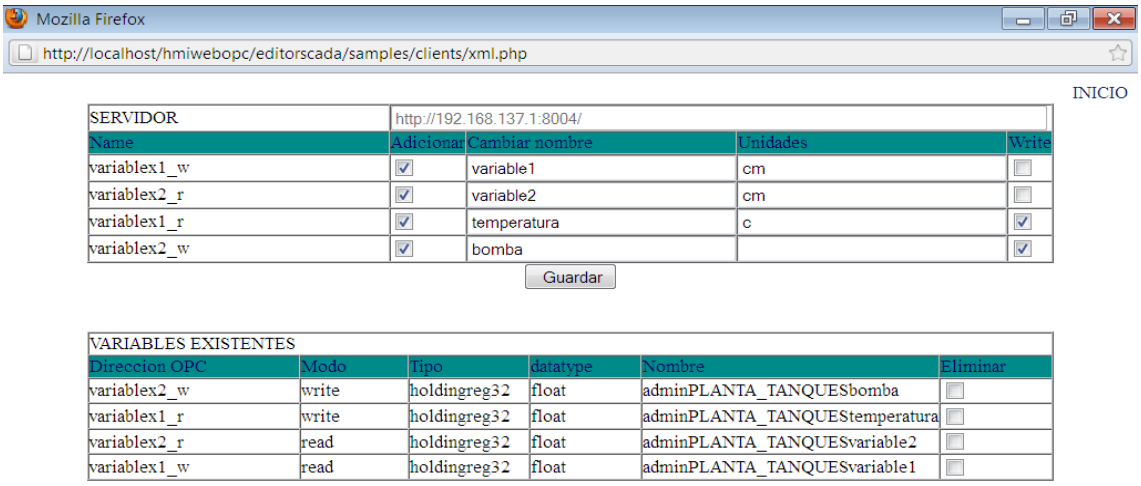
Fuente: propia

Al hacer clic "editar/fuente de datos/XML", se presenta un formulario donde se solicita el nombre del servidor OPC XML, cuando el usuario digita el nombre del servidor, ver ilustración F-18, se presenta otro formulario que permite importar las los objetos ítems creados en dicho servidor para que se almacenen en la base de datos, , ver ilustración F-19, a cada variable se le debe establecer un nuevo nombre, las unidades y el acceso (Lectura o escritura), cuando las variables estén listan se puede guardar para su posterior uso.

Ilustración F-18: Formulario permite ingresar la dirección del servidor OPC XML.

Fuente: propia

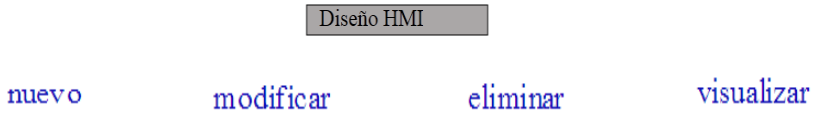
Ilustración F-19: Formulario que permite seleccionar los objetos ítems presentes en el servidor OPC XML.



Fuente: propia

Diseño de esquemas de los procesos: Al seleccionar Diseño HMI, aparece un nuevo menú con las opciones: nuevo, modificar, eliminar, visualizar, ver ilustración F-20.

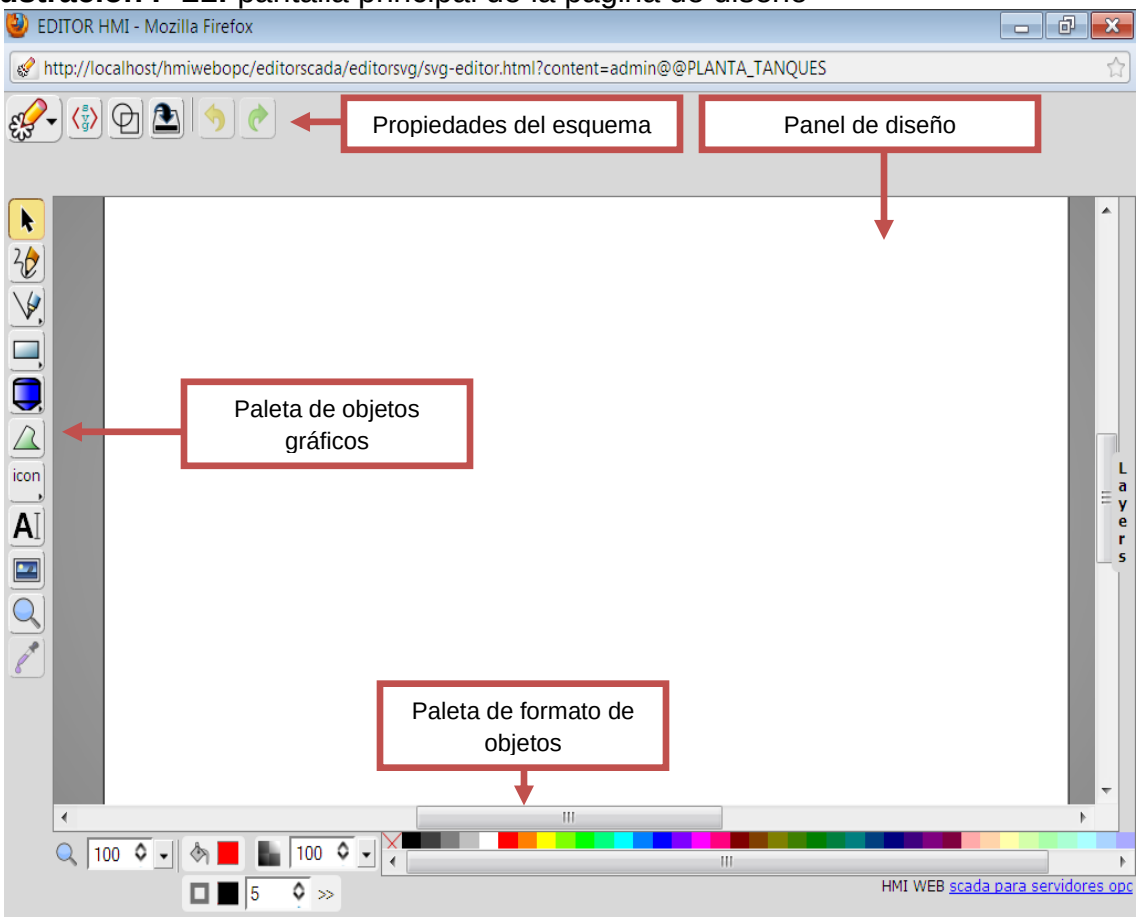
Ilustración F-20: Menú diseño HMI



Fuente: propia

El item "nuevo" permite ir a una página donde se presenta una pantalla que permite diseñar HMI, ver la ilustracion F-21, en esta página se puede crear un esquema con imágenes en formato SVG y asociarlas a funciones creadas en JavaScript para las respectivas animaciones.

Ilustración F-21: pantalla principal de la página de diseño



Fuente: propia

Esta página esta compuesta por:

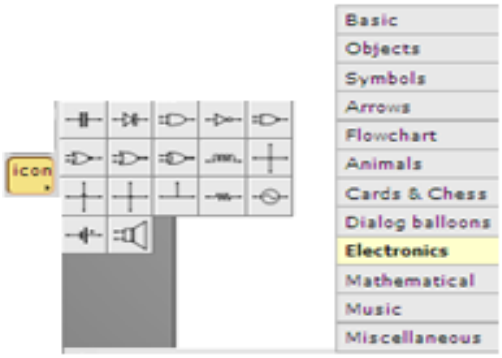
- ***Propiedades del esquema:***

  - ✓  *P*ermite importar, exportar imágenes, configurar las propiedades del esquema: nombre, tamaño, lenguaje y color.

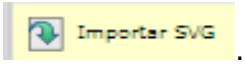
  - ✓  Permite visualizar código fuente y modificarlo
  - ✓  Muestra solo bordes de la figura
  - ✓  Clona un objeto
  - ✓  Agrupar varios elementos en uno solo elemento.

- ***Paleta de objetos gráficos:*** permite obtener objetos de diseño predeterminados y de instrumentación (círculos, cuadrados, texto, líneas, tanques, etc.).

  - ✓  permite seleccionar un objeto
  - ✓  Lápiz
  - ✓  Dibuja líneas
  - ✓ 
  - ✓  Tanques, columnas, pilotos
  - ✓  Permite crear un dibujo.

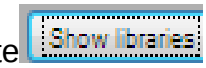


✓  Permite importar imágenes jpg, gif, png etc.

Además de la paleta de objetos existentes, también se puede importar gráficos con formato SVG, haciendo clic en el icono  y luego 

.

También existen dos librerías, que permiten importar gráficos SVG, se debe ir a  luego 

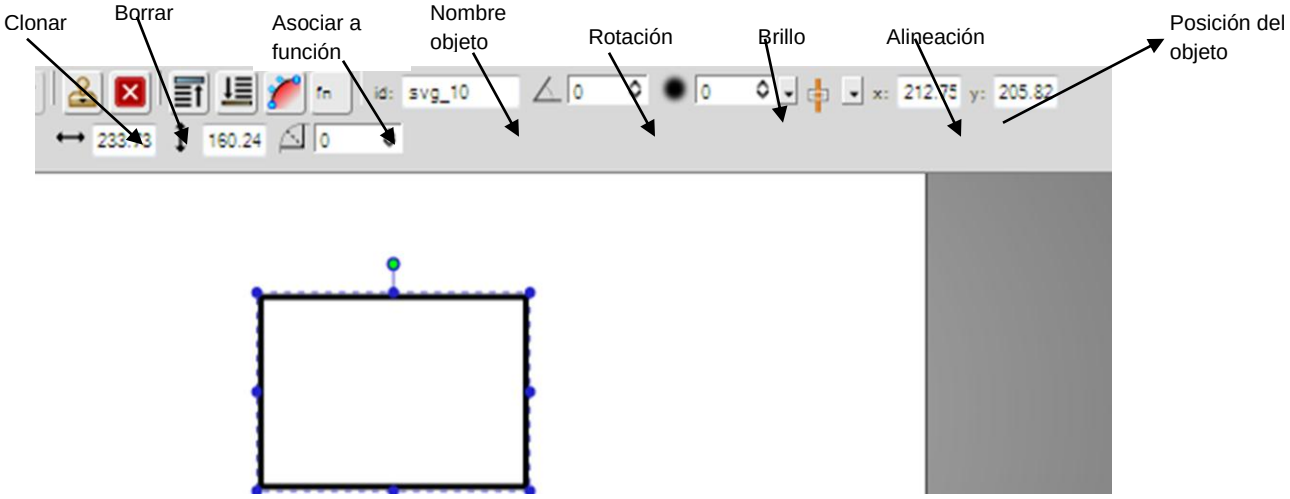
y finalmente 

.

- ***Panel de diseño:*** en este panel se puede crear los esquemas de los procesos.

Paleta de configuracion de objetos SVG: A cada uno de los objetos se les puede configurar las propiedades como lo indica la ilustración F-23. Al dar clic derecho en cada objeto se pueden configurar los siguientes parametros: la posicicion, el angulo de rotacion, el color, la intensidad entre otros.

Ilustración F-22. Formulario de configuración de objetos



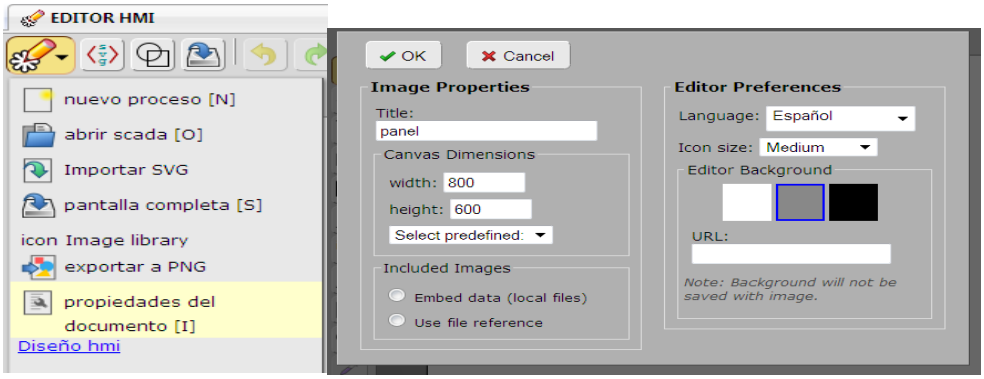
Fuente: Propia.

Para crear un esquema interactivo, que permita el monitoreo de procesos, se deben seguir los pasos que se listan a continuación:

1. Establecer el nombre del esquema
2. Diseñar el esquema
3. Asociar cada elemento del esquema a una función de Javascript
4. Guardar

Para colocarle el nombre al esquema se presiona el icono  , se hace clic en **propiedades**, colocar un título y establecer las propiedades de la ventana del editor, ver ilustración F-22.

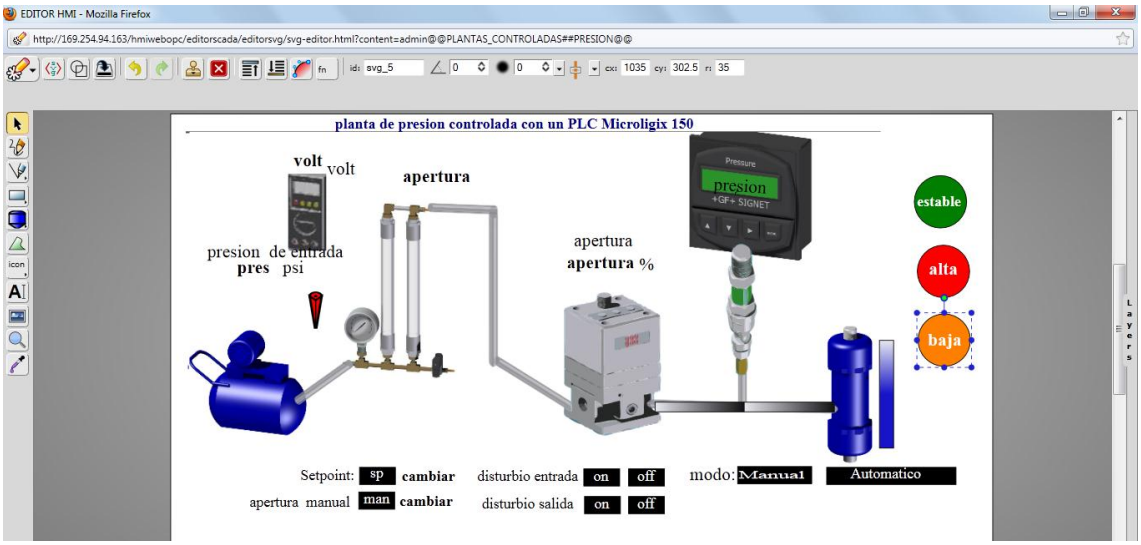
Ilustración F-23. Formulario que permite configurar las propiedades del esquema.



Fuente: propia

El segundo paso es diseñar el esquema, el cual hace uso de la paleta de objetos gráficos. La ilustración F-24 presenta un ejemplo de un esquema diseñado mediante gráficos SVG.

Ilustración F-24. Ejemplo de un esquema diseñado en el MM&SW

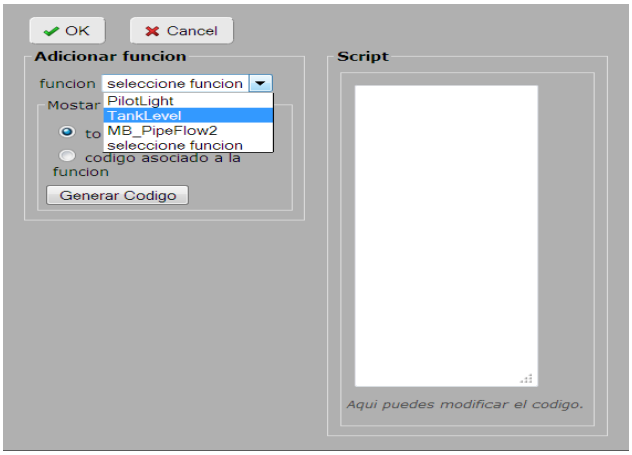


Fuente: Propia.

Las imágenes en formato SVG encadenadas con el lenguaje de programación JavaScript permiten crear animaciones. Este sistema permite asociar cada grafico SVG a una funcion en javaScript y hacer uso de las variables OPC, ya sea para la visualizacion o para el envio de datos.

Para asociar cada objeto grafico a una funcion, se debe hacer clic en el icono fn, donde dependiendo del tipo de objeto se presenta un formulario con diferentes funciones, ver ilustracion F-25.

Ilustración F-25. Formulario para asociar un gráfico a una función.



Fuente: Propia.

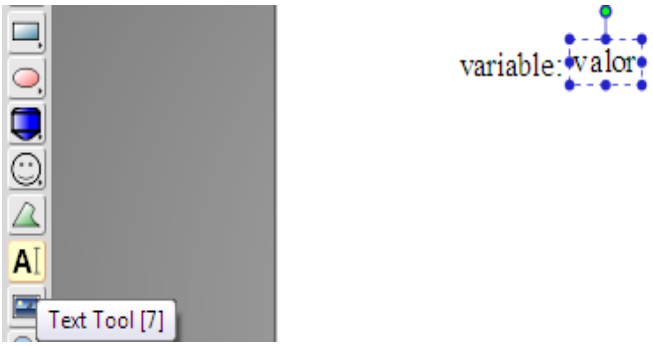
Dentro de las funciones asociadas a cada imagen tenemos:

- ***Writeinmediate():*** permite al usuario enviar un valor a un elemento del servidor OPC, haciendo clic en el grafico. Esta funcion esta dispoinble para todos los objetos graficos.
- ***stringDysplay():*** Toma el valor de una variable y lo visualiza en la pantalla. . Esta funcion esta disponible unicamente para el objeto texto.
- ***Rotate():*** rota la imagen según el valor de una variable. Esta funcion esta disponible para todos los objetos graficos.
- ***Translate():*** mueve la imagen según el valor de una variable. Esta funcion esta disponible para todos los objetos graficos.
- ***SlideDisplay():*** visualiza el cambio de una variable, cambiando de color la imagen. Se utiliza para simular que hay flujo en una tuberia, o que esta circulando una corriente. Esta funcion esta disponible para los rectangulos y cuadrados.
- ***TankLevel():*** visualiza el nivel de un tanque. Esta funcion esta disponible para los rectangulos y cuadrados.
- ***PilotLight():*** cambia el color de la imagen dependiendo del valor recibido, si es uno en rojo, si es cero en verde.

A continuación se detalla el uso de estas funciones:

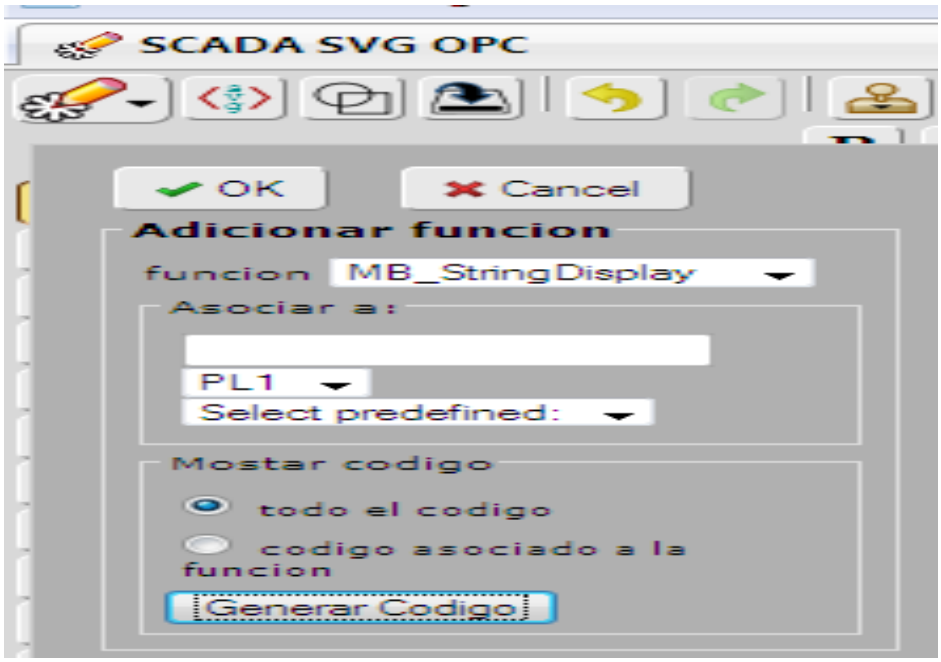
stringDisplay(): si se quiere visualizar datos de una variable se debe dar clic en el icono  ubicado en la paleta de objetos graficos, colocarlo en el panel de diseño, ver ilustración F-26, digitar un texto, en este caso "**valor**", seleccionar este objeto y presionar el botón  para que se presente un formulario como el de la ilustración F-27; seleccionar la función MB_StringDisplay, y escoger la variable OPC que se quiere visualizar, y finalmente presionar el botón "generar código" y luego OK.

Ilustración F-26. Formulario para visualizar el valor de una variable.



Fuente: Propia.

Ilustración F-27. Formulario para configurar las opciones de visualización de valores de objetos ítems.



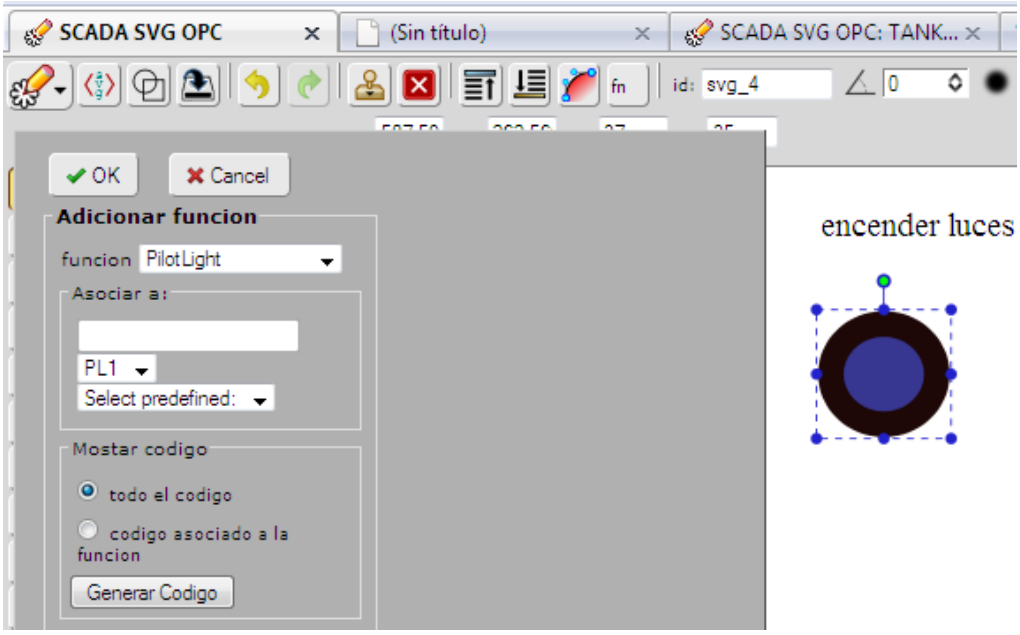
Fuente: Propia.

Writeinmediate: Para cambiar el valor de una variable, se debe seleccionar un objeto, hacer clic en el icono 

, seleccionar la función Writeinmediate, asociar a la variable OPC a la que se le cambiara el valor y hacer clic en generar código.

PilotLight: si se quiere colocar simular un piloto, se selecciona un circulo se configura el tamaño, se selecciona la función PilotLight, se asocia a una variable y se presiona generar código, ver ilustración F-26.

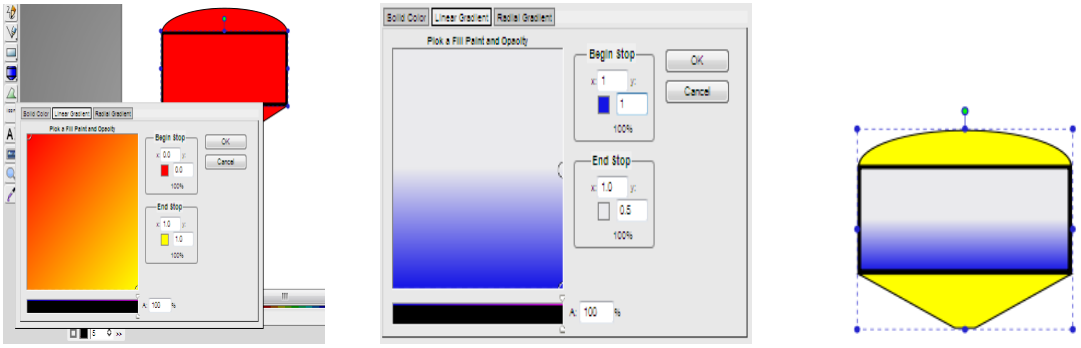
Ilustración F-28. Configuración de un círculo para que actué como piloto.



Fuente: Propia.

Si se quiere visualizar el nivel del tanque se coloca el tanque con el icono  , se sobrepone un cuadrado sobre el tanque, se selecciona este cuadrado, se presiona sobre el icono  que se encuentra en la parte inferior, para que aparezca un formulario de colores, del cual se debe seleccionar **linear gradient,** establecer los colores y configurar los parámetros **Begin stop:** X=1, Y=1 **end stop:** x=1 Y=0.5 y OK. Ver ilustración F-29.

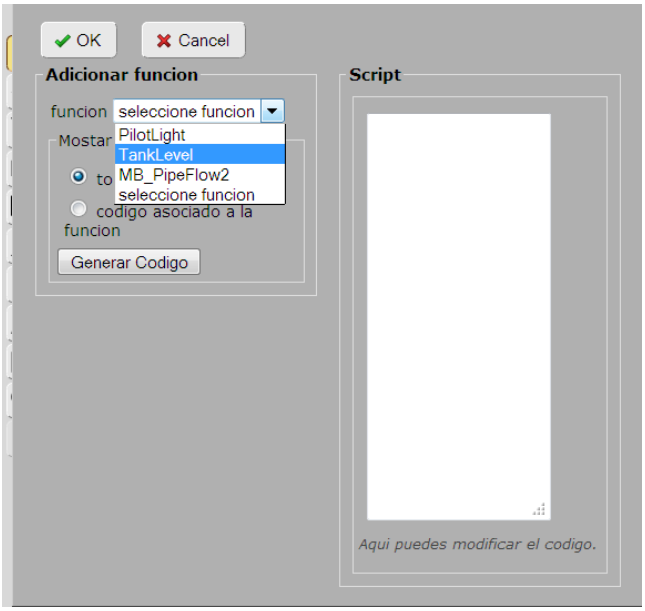
Ilustración F-29. De un rectángulo para que permita la visualización de nivel de un tanque.



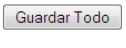
Fuente: Propia.

Seccionar el cuadrado y dar clic en el icono **asociar función**, selecciona la función **tank_level**, el rango máximo con que se llena el tanque, elegir la variable OPC y presionar generar código, ver Ilustración F-30.

Ilustración F-30. Formulario de configuración del nivel de un tanque.

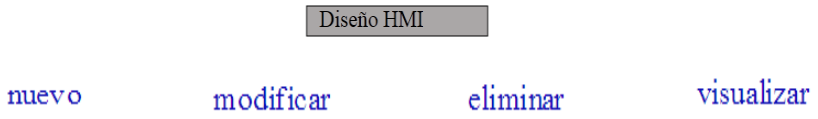


Fuente: Propia.

Cuando el diseño esté listo se debe presionar el icono  (guardar), y luego  para que se almacene la información y permita el retorno a la página principal.

Dentro de diseño gráfico también se puede modificar, visualizar, eliminar cada uno de los esquemas diseñados, para cualquiera de estas opciones se debe hacer clic en cada menú y seleccionar el esquema al que se va a modificar, eliminar o visualizar, ver ilustración F-31.

Ilustración F-31. Formulario de diseño gráfico en el MM&SW



Fuente: Propia.

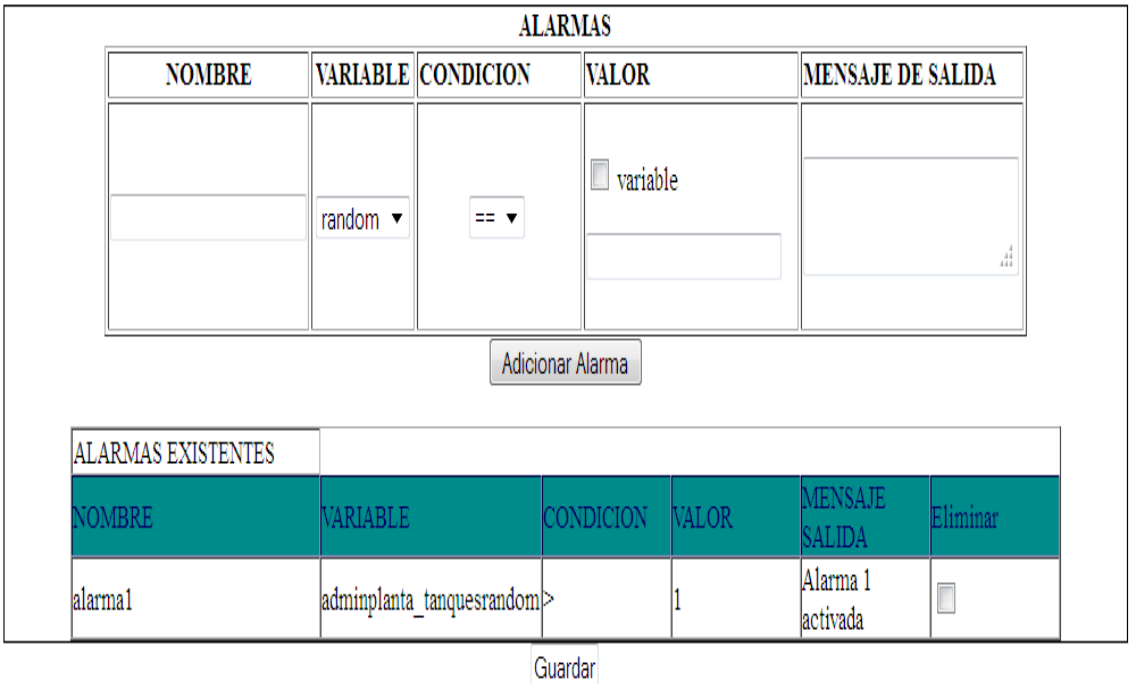
Configuración de alarmas: cuando el usuario selecciona la opción "editar/Configurar Alarmas", se presenta un formulario, ver ilustración F-32, que permite configurar las alarmas para que puedan ser visualizadas en el sub-módulo de monitoreo.

Los parámetros que se deben establecer para adicionar las alarmas son: nombre de alarma, variable, condición, y mensaje de salida.

En cuadro de texto de nombre, se debe colocar la identificación de la alarma que se va a adicionar, en la celda "variable" se debe seleccionar una de las variables existentes que generara la alarma, en la opción "condición" se debe seleccionar entre los signos existentes en la lista (menor, mayor, igual, menor o igual, mayor o igual); la variable se la puede comparar con un valor por el usuario, o compararla con otra variable, y finalmente se debe configurar el mensaje que describirá el motivo de la alarma.

En la imagen F-32, también se puede visualizar una alarma ya configurada, en este caso el nombre es **alarma1,** esta alarma presenta el mensaje de salida Alarma1 activada cuando la variable random sea mayor a uno.

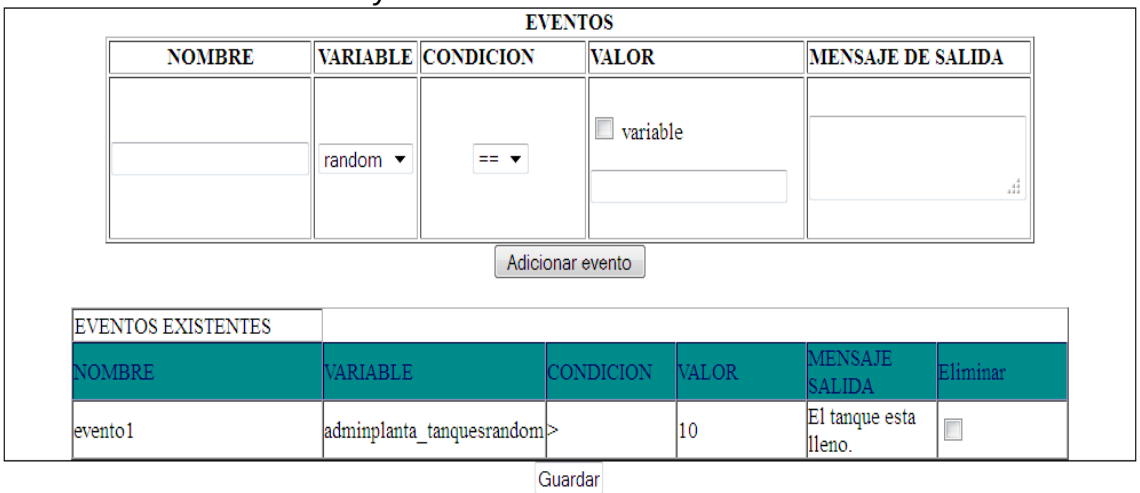
Ilustración F-32. Configuración y visualización de Alarmas



Fuente: propia

Configuración de eventos: cuando el usuario selecciona la opción "Configurar eventos", se presenta un formulario, ver ilustración F-33, con las mismas opciones de configuración de las alarmas, se debe configurar los parámetros de cada evento, hacer clic en adicionar evento, y finalmente en guardar.

Ilustración F-33. Creación y visualización de eventos

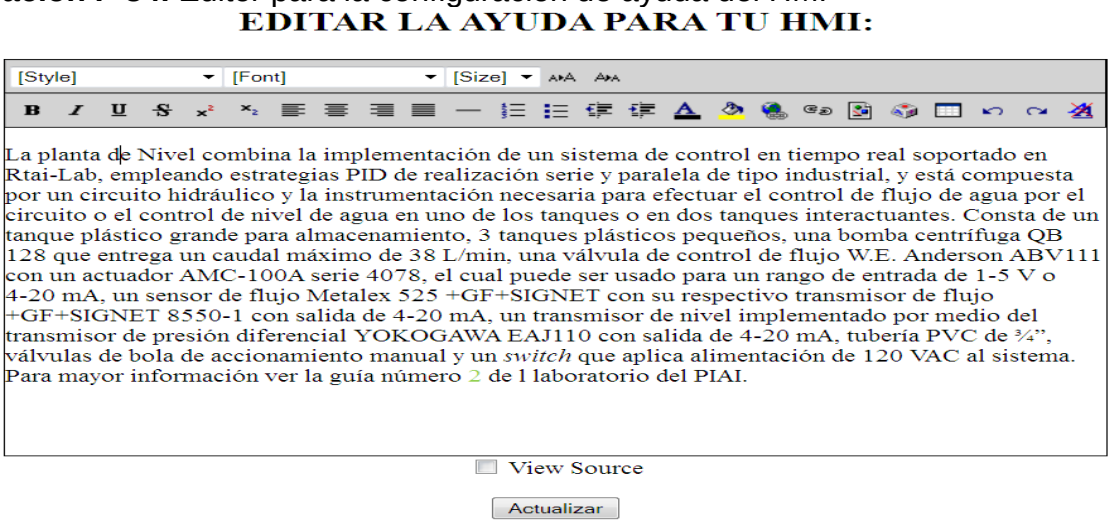


Fuente: propia

Configurar ayuda: al hacer clic en “editar“-“ayuda” se presenta un editor que permite establecer información referente al uso del HMI, ver ilustración F-34, este editor presenta funciones similares a *Word*, permite adicionar tablas, imágenes, entre otros. Al ingresar información internamente se genera el código HTML el cual puede ser modificado presionando la opción *view source.*

Se debe ingresar las instrucciones para el manejo de cada HMI, hacer clic en actualizar, y finalmente en guardar, (siempre se debe actualizar antes de guardar).

Ilustración F-34. Editor para la configuración de ayuda del HMI



Fuente: propia

Hasta aquí, se presentó la guía de diseño de un proyecto de monitoreo y supervisión, esta guía recomienda el siguiente proceso:

- Establecer un nombre al proyecto (hacer clic en “nuevo del menú principal”)
- Importar las variables desde diferentes fuentes de datos (OPC DCOM, XML, Otra fuente)
- Crear los esquemas HMI: asignarle un nombre al esquema, diseñar el esquema, asociar cada objeto grafico a una función SVG, y guardar.
- Configurar las alarmas
- Configurar los eventos
- Configurar la ayuda

Monitoreo y supervisión

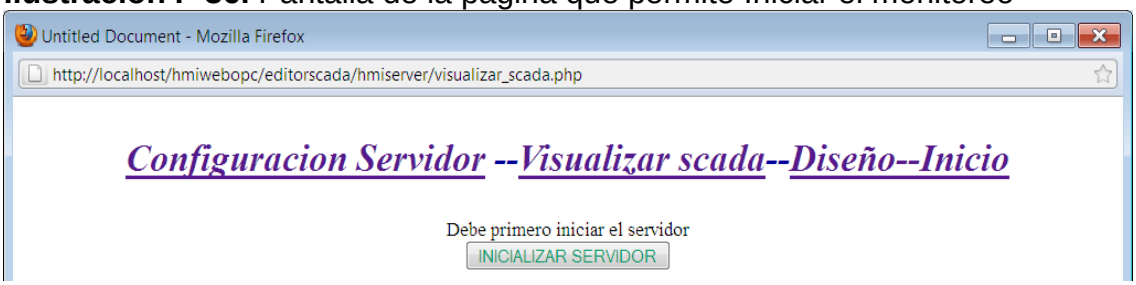
Para iniciar el monitoreo se debe seleccionar la opción “visualizar” del menú principal de la página de diseño, para que aparezca un formulario con una lista de HMI disponibles, ver ilustración F-35.

Ilustración F-35. Menú que permite seleccionar el proyecto para el monitoreo y supervisión.

Fuente: propia

Al seleccionar el nombre del proyecto, aparece una página con los menús: configuración servidor, iniciar monitoreo, visualizar scada, diseño, Inicio e inicializar servidor, ver ilustración F-36. El menú inicio lo enlaza a la página principal, el menú diseño lo retorna a la página anterior, y el botón “iniciar servidor” permite establecer la conexión con el servidor para el monitoreo y supervisión. Antes de hacer clic en Iniciar Monitoreo, se debe presionar primero el botón “iniciar servidor” y esperar cinco segundos.

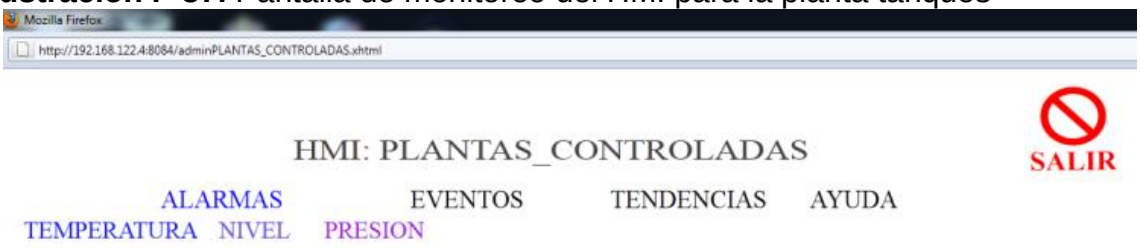
Ilustración F-36. Pantalla de la página que permite Iniciar el monitoreo



Fuente: propia

Al hacer clic en **iniciar monitoreo** se presenta una nueva página que permite la visualización de los procesos diseñados, las alarmas, tendencias y eventos., ver ilustración F-37.

Ilustración F-37. Pantalla de monitoreo del HMI para la planta tanques



Fuente: propia

Las alarmas, los eventos y tendencias presentan el mismo formato para todos los proyectos.

Menú Alarmas: en el MM&SW las alarmas son mensajes de texto que indican que una condición de falla está presente.

La ilustración F-38 presenta una pantalla que permite visualizar las alarmas que se han configurado en el módulo de diseño. Al hacer clic en el menú **alarmas** se presenta una tabla con el nombre, el estado, el tiempo en que fue activada, y el número de veces que se repite esta acción.

Ilustración F-38. Pantalla de la página que permite Iniciar el monitoreo

Alarm:	Estado de Alarma:	Tiempo:	Tiempo OK:	Contador:
Alarma 1 activada	Alarma Activa	Thu Jun 23 2011 14:20:24 GMT-0500	Not OK	1

Fuente: propia

El mensaje de alarma permanecerá visible hasta que el mensaje ha sido reconocido por el operador y la entrada en representación de la culpa se ha restablecido.

Las alarmas pueden estar en los siguientes estados:

- Inactivo: no hay ningún mensaje de alarma asociado a la espera de ser reconocido.
- Activo: una condición de falla está presente y el mensaje de alarma no ha sido reconocido por el operador.
- Reconocido: a condición de falla está presente, y el operador ya la ha visualizado el mensaje.

Las alarmas sólo en los estados: "activo", "reconocido" se presentan en pantalla, no habrá mensajes visibles en la tabla de alarma para el estado "inactivo".

Las alarmas son detectadas por el servidor HMI mediante el control de un valor booleano que se define en la configuración. Cuando ese valor booleano es *True*, se activa la alarma, y si el valor booleano es *False* se desactiva.

Cada mensaje de alarma contendrá la siguiente información:

- El texto de alarma: Esta es una descripción de la alarma.
- El estado de alarma: describe el estado actual de la alarma (por ejemplo, "inactivo", "activo", "reconocido", etc.).
- El tiempo: Presenta el tiempo en que el servidor ha detectado la alarma.
- Aceptar el tiempo: este es el momento en que el usuario acepto la falla.
- Contador: número de veces que la alarma se ha presentado (el valor medido ha cambiado de false a true). Una vez que la alarma ha desaparecido de la pantalla, este número se restablece.
- "Reconociendo" la alarma significa que el operador ha tomado una acción para indicar que ha visto los mensajes de alarma.

Historial de alarma: cuando un mensaje de alarma ha cambiado desde el activo a estado inactivo, una copia del mensaje se agrega a la historial de alarma. El historial de alarmas es un registro secuencial de alarmas.

Menú Eventos:

Los eventos son mensajes que representan a la ocurrencia de incidentes sobre los cuales debe ser el operador notificado, pero que el operador normalmente no necesita tomar ninguna acción. A diferencia de las alarmas, el operador no reconoce los eventos.

Cada vez que se genera un evento, se presenta un mensaje en la pantalla de eventos. Esto significa que el mismo evento puede aparecer varias veces en la tabla de visualización de eventos, ver la ilustración F-39.

Ilustración F-39. Formulario de visualización de eventos

Evento #:	Fecha:	Evento:	Estado:
1308856822	Thu Jun 23 2011 14:20:24 GMT-0500	El tanque esta lleno.	1

Fuente: propia

La tabla de eventos incluye la siguiente información:

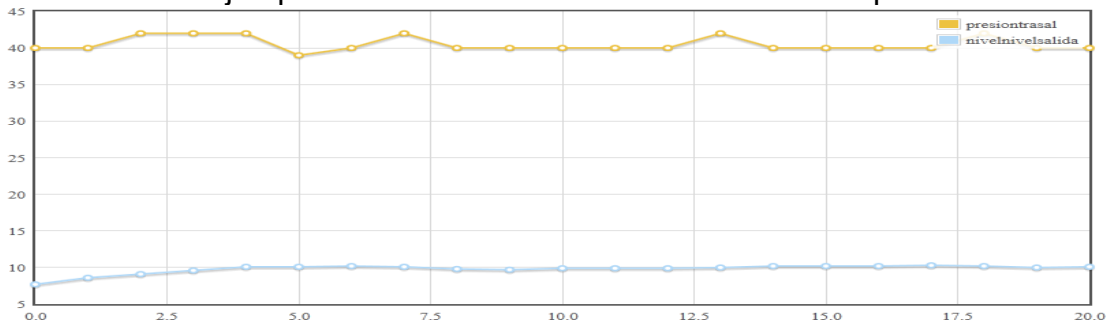
- El número de evento: este es un número secuencial generado por la pista del servidor los mensajes de eventos.
- Tiempo del evento: este es el momento en que se generó el evento.
- Evento. descripción del evento.
- Estado: un valor que indica el estado del evento (0 para apagado, 1 para el).

Menú Tendencias:

En cada HMI se presenta una gráfica que permite representar todas las variables creadas en el módulo de diseño, ver ilustración F-40, aquí se debe seleccionar las variables que se quieren representar.

La ilustración F-40 presenta una pantalla con una gráfica de cuatro variables creadas en el módulo de diseño, el eje de la abscisa representa el tiempo en minutos, y el eje Y representa el valor que tiene cada objeto ítem creado en el servidor OPC.

Ilustración F-40. Ejemplo de tendencias con variación en el tiempo



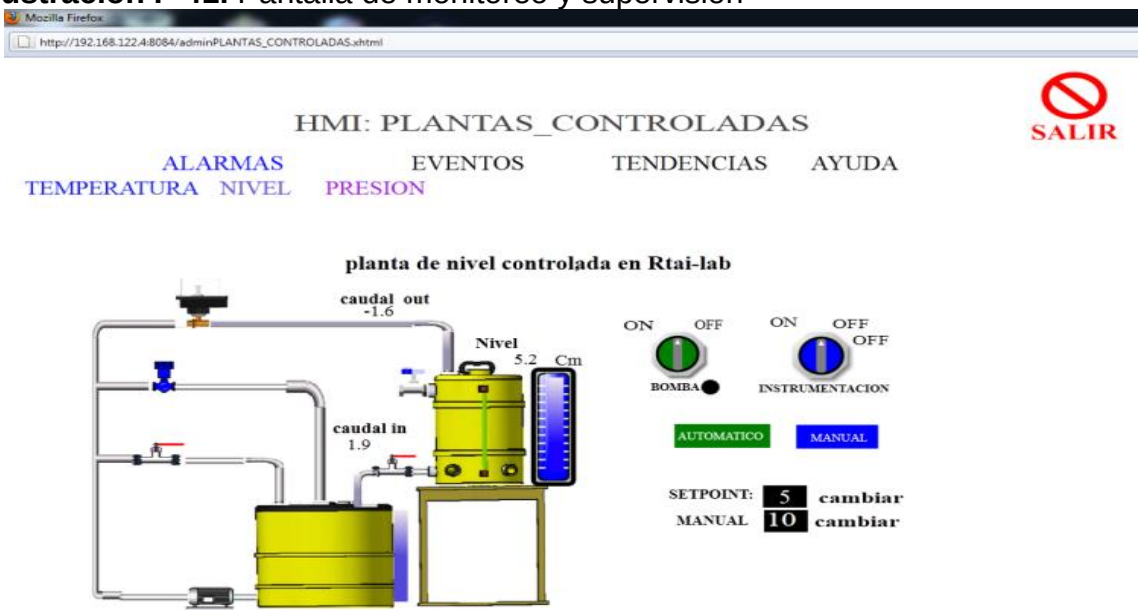
Fuente: Propia

Menú Ayuda:

Al hacer clic en ayuda, se presenta una página con un manual de usuario para cada Proyecto.

Esquemas para el monitoreo: al hacer clic en el menú correspondiente a cada esquema, se presenta un formulario con los objetos gráficos creados en el módulo de diseño. La ilustración F-41 presenta un ejemplo de un ejemplo de una página del MM&SW que permite el monitoreo de procesos de automatización.

Ilustración F-41. Pantalla de monitoreo y supervisión



Fuente: Propia

Terminar monitoreo: Para finalizar el monitoreo **siempre** se debe hacer clic en "salir" parte superior izquierda, para que aparezca la página de configuración del servidor HMIServer, ver ilustración F-42, se presenta un formulario que permite recargar las variables en el servidor, y finalizar el monitoreo.

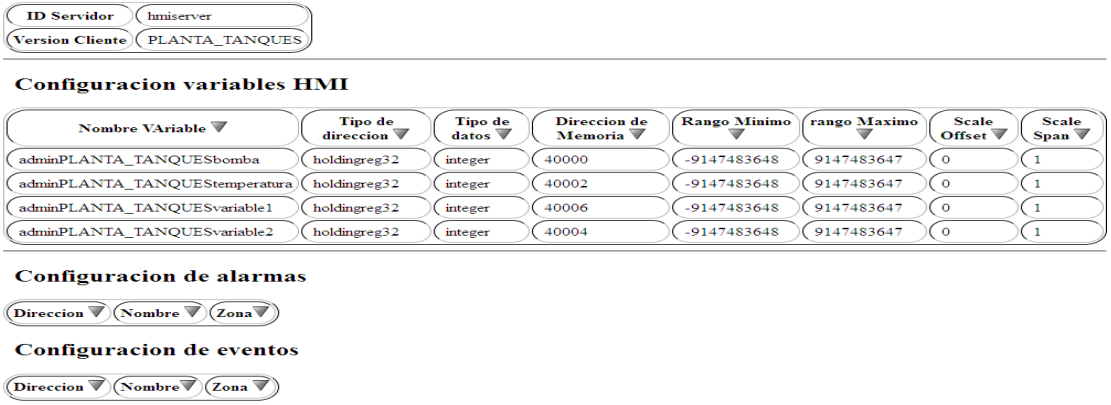
Ilustración F-42. Página que permite finalizar el monitoreo

Fuente: Propia

Esta página también contiene un menú con las opciones: Inicio, Configuración, Control, Ayuda.

Cuando se selecciona **configuración** se presenta las variables existentes y sus configuraciones, ver ilustración F-43, también se presentan las alarmas y los eventos.

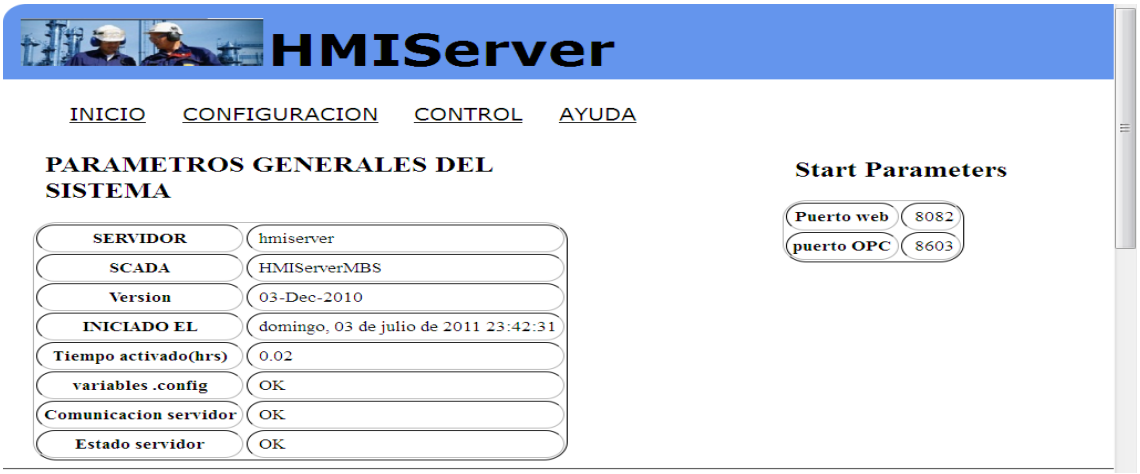
Ilustración F-43. Variables existentes en el proyecto seleccionado



Fuente: Propia

Al presionar **Control** se visualiza el estado y las propiedades del servidor HMIServer, ver ilustración F-44. Dentro de las propiedades se encuentran: puertos de comunicación, fecha y hora de inicializado, archivo de configuración, entre otros.

Ilustración F-44. Parámetros del servidor HMIServer



Fuente: Propia

Al hacer clic en **ayuda**, se presenta un manual de usuario de todo el módulo HMIWEB.

ANEXO G. VALIDACION DE LOS MODULOS DE LA ARQUITECTURA

I. REQUERIMIENTOS FUNCIONALES

Las tablas G-1, G-2, G-3, G-4 y G-5, registran la validación de los requerimientos funcionales de los módulos diseñados para la arquitectura de integración según las pruebas realizadas. En la primera columna se especifica los sub-módulos, en la segunda columna se presenta un identificador de cada requerimiento, en la tercera columna se realiza la descripción de los requerimientos, y en la cuarta columna se describe si se cumple o no el requerimiento.

Tabla G -1: requerimientos funcionales para el módulo cliente OPC XML en Matlab.

Sub Módulo	ID	Requerimientos Funcionales	
	RFMCOXM1	La aplicación debe iniciarse mediante la presentación de un formulario que contenga los siguientes componentes a) Conexión Servidor b) importar variables c) Lectura y escritura activa de datos d) finalizar servidor.	✓
Conexión con el servidor	**RFMCOXM2**	Se debe permitir al usuario ingresar la dirección del servidor OPC XML	✓
	RFMCOXM3	Con base a la información recolectada se debe establecer la comunicación con el servidor OPC XML	✓
Configuración de variables	**RFMCOXM4**	Se debe permitir al usuario seleccionar cuales son los ítems de lectura y escritura con los que se va a trabajar.	✓
	RFMCOXM5	Se debe almacenar los ítems seleccionados por el usuario.	✓
	RFMCOXM6	Las propiedades de los ítems de lectura se deben poder consultar periódicamente al servidor OPC XML.	✓
Lectura y escritura dinámica de objetos OPC	**RFMCOXM7**	Se debe poder cambiar el valor de los ítems en el servidor OPC XML con parámetros establecidos por el usuario o por el entono Matlab.	✓
	RFMCOXM8	Se debe permitir al usuario finalizar la comunicación con el servidor OPC XML.	✓
	RFMCOXM9	Se debe enviar al servidor OPC XML la orden de finalizar la comunicación.	✓

Fuente: Propia.

Tabla G -2: requerimientos funcionales para el Módulo Cliente OPC DCOM en Rtai-Lab.

Sub Módulo	ID	Requerimientos Funcionales	
	RFMCODR1	Se debe permitir al usuario seleccionar entre dos bloques en Scicos, uno para la lectura y otro para la escritura.	✓
Bloque de lectura	**RFMCODR2**	Se debe permitir al usuario configurar el bloque de lectura con los parámetros: IP, host, dirección servidor, número de salidas, dirección de las variables.	✓
	RFMCODR3	Cambiar la configuración funcional y gráfica del bloque de salida con los nuevos parámetros.	✓
	RFMCODR4	Se debe permitir conectar las salidas del bloque con otros bloques creados en Scicos.	✓
Bloque de escritura	**RFMCODR5**	Se debe permitir al usuario configurar el bloque de escritura con los parámetros: IP, host, dirección servidor, número de entradas, dirección de las variables.	✓
	RFMCODR6	Cambiar la configuración funcional y gráfica del bloque de salida con los nuevos parámetros.	✓
	RFMCODR7	Se debe permitir conectar a las entradas del bloque de escritura, las salidas provenientes de otros bloques, para que estos valores sean enviados al servidor OPC DCOM.	✓
Lectura y escritura dinámica de objetos OPC	**RFMCODR8**	Se debe iniciar la comunicación cuando el usuario lo solicite, leer y escribir constantemente en los objeto ítems del servidor.	✓
	RFMCODR9	El bloque de escritura debe capturar los valores provenientes de los Servidores OPC DCOM	✓
	RFMCODR10	Se debe permitir al usuario finalizar la comunicación con el servidor OPC DCOM.	✓
	RFMCODR11	Cuando el usuario de la orden de finalizar la comunicación se debe enviar al servidor un mensaje de finalización.	✓

Fuente: Propia.

Tabla G -3: Requerimientos funcionales para el Módulo Cliente OPC XML en Rtai-Lab.

Sub Módulo	ID	Requerimientos Funcionales	
	RFMCOXR1	Se debe permitir al usuario seleccionar entre dos bloques, uno para la lectura y otro para la escritura.	✓
Bloque de lectura	**RFMCOXR2**	Se debe permitir al usuario configurar el bloque de lectura con los parámetros: IP, host, dirección servidor, número de salidas, dirección de las variables.	✓
	RFMCOXR3	Cambiar la configuración funcional y gráfica del bloque de salida con los nuevos parámetros.	✓
	RFMCOXR4	Se debe permitir conectar las salidas del bloque con otros bloques creados en Scicos.	✓
Bloque de escritura	**RFMCOXR5**	Se debe permitir al usuario configurar el bloque de escritura con los parámetros: IP, host, dirección servidor, número de entradas, dirección de las variables.	✓
	RFMCOXR6	Cambiar la configuración funcional y gráfica del bloque de salida con los nuevos parámetros.	✓
	RFMCOXR7	Se debe permitir conectar las entradas del bloque de escritura.	✓
Lectura y escritura dinámica de objetos OPC	**RFMCOXR8**	Se debe iniciar la comunicación cuando el usuario lo solicite, en esta parte se debe estar leyendo y escribiendo constantemente en los objeto ítems del servidor.	✓
	RFMCOXR9	El bloque de escritura debe capturar los valores provenientes de los Servidores OPC XML	✓
	RFMCOXR10	Se debe permitir al usuario finalizar la comunicación con el servidor OPC XML.	✓
	RFMCOXR11	Cuando el usuario de la orden de finalizar la comunicación se debe enviar al servidor un mensaje de finalización.	✓

Fuente: Propia

Tabla G -4: requerimientos funcionales para el módulo servidor OPC XML.

Sub Módulo	ID	Requerimientos Funcionales	
	RFMSOX1	Se debe presentar al operario un formulario que contenga los siguientes Opciones: Opciones servidor, Opciones ítems, comunicación con los clientes OPC.	
Opciones del servidor	**RFMSOX2**	Se debe permitir configurar al servidor el puerto por el cual se va a establecer la comunicación.	✓
	RFMSOX3	Configuraciones: Se debe permitir al usuario establecer parámetros como: almacenamiento automático, tamaño de numero de objetos ítems, cantidad de solicitudes aceptadas.	✓
	RFMSOX4	Visualizar errores: Se debe permitir al usuario ver el registro de errores que se presentan en la comunicación.	✓
	RFMSOX5	Ver código XML: El usuario debe tener disponible el código XML, para usarlo en otras aplicaciones.	✓
	RFMSOX6	El usuario debe poder visualizar todos los accesos de los clientes al servidor, incluido IP del host, hora, y nombre del cliente.	✓
	RFMSOX7	Se debe permitir visualizar el estado del servidor.	✓
Configuración De Ítems	**RFMSOX8**	El usuario debe poder crear ítems, con propiedades: nombre, tipo.	✓
	RFMSOX9	Todos los objetos ítems, al igual que las propiedades se deben poder visualizar.	✓
	RFMSOX10	Se debe permitir establecer el valor de un objeto ítem desde el servidor OPC.	✓
	RFMSOX11	Se debe permitir, guardar en un archivo las propiedades de los ítems para su reutilización.	✓
	RFMSOX12	Se debe permitir extraer de un archivo las propiedades de los ítems, y visualizarlas en el servidor OPC.	✓
Comunicación con los clientes OPC	**RFMSOX13**	Aceptar solicitud de conexión con los clientes OPC XML.	✓
	RFMSOX14	Retornar a los clientes información acerca de las propiedades de los objetos OPC.	✓
	RFMSOX15	se debe capturar los datos del Cliente OPC y convertirlos en el formato estándar para que puedan ser leídos por los clientes consumidores	✓
	RFMSOX16	Los clientes deben poder solicitar todas las propiedades de los objetos ítems.	✓
	RFMSOX17	Finalizar la comunicación con los clientes OPC XML y retornar un mensaje de finalización.	✓

Fuente: **Propia**

Tabla G -5: requerimientos funcionales para el Módulo de Monitoreo y supervisión Web (MM&SW)

Sub Módulo	ID	Requerimientos Funcionales	
	RFMM&SW1	El sistema de monitoreo y supervisión debe contar con tres sub-módulos básicos: gestor de contenidos, de diseño y de monitoreo	✓
Gestor de contenido	**RFMM&SW2**	Ingreso al sistema: se debe permitir al usuario ingresar mediante una URL.	✓
		Gestión de usuarios: El sistema deberá permitir administrar los usuarios	✓
	RFMM&SW4	Usuarios conectados:el sistema debera indicar quien esta en línea'	✓
	RFMM&SW5	Encuestas: proporcionar un método sencillo de crear encuestas breves	✓
	RFMM&SW6	Edición de contenido: los usuarios registrados deberán poder editar los artículos publicados por cualquier otro usuario.	✓
	RFMM&SW7	Enlaces de interés: dentro de estos enlaces se deberá visualizar un menú de diseño y monitoreo	✓
Diseño de proyectos	**RFMM&SW8**	Crear: se debe permitir crear un Nuevo proyecto	✓
	RFMM&SW9	Eliminar: se debe permitir eliminar los proyectos seleccionados.	✓
	RFMM&SW10	Editar: Se debe permitir editar los proyectos seleccionados.	✓
	RFMM&SW11	Visualizar: se debe permitir visualizar los proyectos seleccionados.	✓
	RFMM&SW12	Se debe permitir importar objetos ítems de los Servidores OPC DCOM/XML y almacenarlos para su posterior uso	✓
	RFMM&SW13	La página de diseño debe utilizar objetos y formas predefinidos botones, marcos, llaves	✓
	RFMM&SW14	Se debe permitir Importar imágenes u objetos de otras aplicaciones.	✓
	RFMM&SW15	Utilizar diseños de tanques, cañerías, máquinas, Icono y equipamiento de diferentes tipos de industrias.	✓
	RFMM&SW16	Cada objeto ítem incluido en la pantalla debe poder ser animado en función de alguna variable.	✓
	RFMM&SW17	El tipo de animación debe depender del tipo de objeto	✓
	RFMM&SW18	El sistema debe permitir administrar los sistemas de alarmas y eventos.	✓
Monitoreo supervisión	**RFMM&SW20**	Ejecutar las acciones de mando pre-programadas a partir de los valores actuales de variables leídas.	✓
	RFMM&SW21	Visualizar, almacenar y controlar Alarma	✓
	RFMM&SW22	Visualizar eventos	✓
	RFMM&SW23	Graficar variables en función del tiempo	✓
	RFMM&SW24	Obtener ayuda	✓
	RFMM&SW25	Salir del sistema	✓

Fuente: Propia.

II. REQUERIMIENTOS NO FUNCIONALES

Tabla G -6: Requerimientos no funcionales generales.

Requerimiento	Descripción	Verificación
RNFG1	Cada uno de los módulos deben utilizar herramientas que le permitan un tiempo de respuesta menor a un segundo.	Aunque el servidor OPC DCOM presento un tiempo de respuesta menor que el XML, se pudo verificar que el MSOX cumple con el requerimiento **RNFG,** que dice que el tiempo de respuesta debe ser menor a un segundo.
RNFG2	Cada uno de los módulos deben estar disponible 100% o muy cercano a esta disponibilidad durante el horario hábil establecido para la comunicación OPC.	Para verificar este requerimiento, se dejó en funcionamiento las pruebas realizadas durante 2 horas, en este lapso de tiempo, ninguno de los módulos presento fallas.
RNFG3	El sistema debe estar en capacidad de permitir en el futuro el desarrollo de nuevas funcionalidades, modificar o eliminar funcionalidades después de su construcción y puesta en marcha inicial.	Todos los módulos implementados pueden ser modificados si se quiere nuevas funcionalidades, ya que está diseñado bajo el paradigma de "Software Libre".
RNFG4	Cada uno de los módulos debe: tener una interfaz atractiva y amigable, presentar mensajes de error que permitan al usuario identificar el tipo de error.	✓
RNFG5	El sistema debe ser diseñado y construido con los mayores niveles de flexibilidad en cuanto a la parametrización de los tipos de datos.	
RNFG6	Cada uno de los módulos deberá estar documentado, con los manuales de administración y de usuario.	Los Anexos B;C;D;E;F presentan la documentación correspondiente a cada módulo
RNFG7	Los módulos deben ser de fácil operación por el área técnica.	✓
RNFG8	En Cada uno de los módulos se debe validar automáticamente la información contenida en los formularios de ingreso. Teniendo en cuenta aspectos tales como obligatoriedad de campos, longitud de caracteres permitida por campo, manejo de tipos de datos, etc.	✓

Fuente: propia

Tabla G -7: Requerimientos no funcionales específicos en cada módulo

Módulo	Código	Requerimiento	Verificación
MCOXM	RNFMCOXM1	Estar en capacidad de comunicarse con todos los Servidores OPC en el estándar XML con tiempo de respuesta aceptable y uniforme en períodos de alta, media y baja demanda de uso del sistema.	✓
	RNFMCOXM2	El módulo Cliente OPC XML en Matlab debe ser fácil de instalar, y debe permitir su instalación en Windows.	✓
	RNFMCOXM3	Se debe permitir comunicarse con uno o varios Servidores OPC XML a la vez	✓
MCODR	RNFMCODR1	Estar en capacidad de comunicarse con todos los Servidores OPC en el estándar DCOM con tiempo de respuesta aceptable y uniforme en períodos de alta, media y baja demanda de uso del sistema.	✓
	RNFMCODR2	Permitir su instalación en Linux.	✓
	RNFMCODR3	Se debe permitir comunicarse con uno o varios Servidores OPC DCOM a la vez	✓
	RNFMCODR4	Las herramientas utilizadas para la implementación de este módulo deben ser de fuente libre.	✓
MCOXR	RNFMCOXR1	Estar en capacidad de comunicarse con todos los Servidores OPC XML con tiempo de respuesta aceptable y uniforme en períodos de alta, media y baja demanda de uso.	s ✓
	RNFMCOXR2	El módulo Cliente OPC XML debe permitir su instalación en Linux.	✓
	RNFMCOXR3	Las herramientas utilizadas para la implementación de este módulo deben ser de fuente libre.	✓
	RNFMCOXR4	Se debe permitir comunicarse con uno o varios Servidores OPC XML a la vez	✓
MSOX	RNFMSOX1	Las herramientas utilizadas para la implementación de este módulo deben ser de fuente libre.	✓
	RNFMSOX2	El servidor OPC XML debe poder comunicarse con varios clientes OPC XML instantáneamente.	✓
	RNFMSOX3	El sistema debe ser, capaz de ser instalado en plataformas Windows o Linux.	✓
MM&SW	RNFMM&SW1	El sistema puede ser utilizado bajo cualquier plataforma e independiente del navegador.	
	RNFMM&SW22	El sistema debe manejar acceso por roles, así como consideraciones mínimas de seguridad.	✓
	RNFMM&SW3	El sistema debe soportar que un mismo programa sea usado por dos o más usuarios distintos.	✓
	RNFMM&SW4	El sistema debe ser, capaz de ser instalado en plataformas Windows o Macintosh y navegable con diferentes exploradores de Internet.	✓
	RNFMM&SW5	Las herramientas utilizadas para la implementación de este módulo deben ser de fuente libre.	✓

Fuente: propia

ANEXO H. GUIA PARA LA INTEGRACION DE MATLAB, RTAI-LAB Y EL PLC MICROLOGIX 1500 MEDIANTE EL ESTANDAR OPC DCOM

I. INTRODUCCION

En este anexo se presenta una guía para la integración de Matlab, Rtai-Lab y el PLC Micrologix 1500 mediante el estándar OPC DCOM. El objetivo de esta práctica es que los estudiantes conozcan las ventajas que brinda OPC DCOM y afiancen los conocimientos adquiridos en el transcurso de la carrera de Ingeniería en Automática. Para la integración de estos elementos de control, se hace uso de las plantas de Nivel y Presión existentes en el laboratorio de control de procesos (LCP) del PIAI, de una planta virtual de control de temperatura implementada en Matlab y de los módulos: módulo servidor OPC DCOM (MSOD), módulo cliente OPC DCOM Matlab (MCODM), módulo cliente OPC DCOM Rtai-Lab (MCODR) y el módulo de monitoreo y supervisión MM&SW.

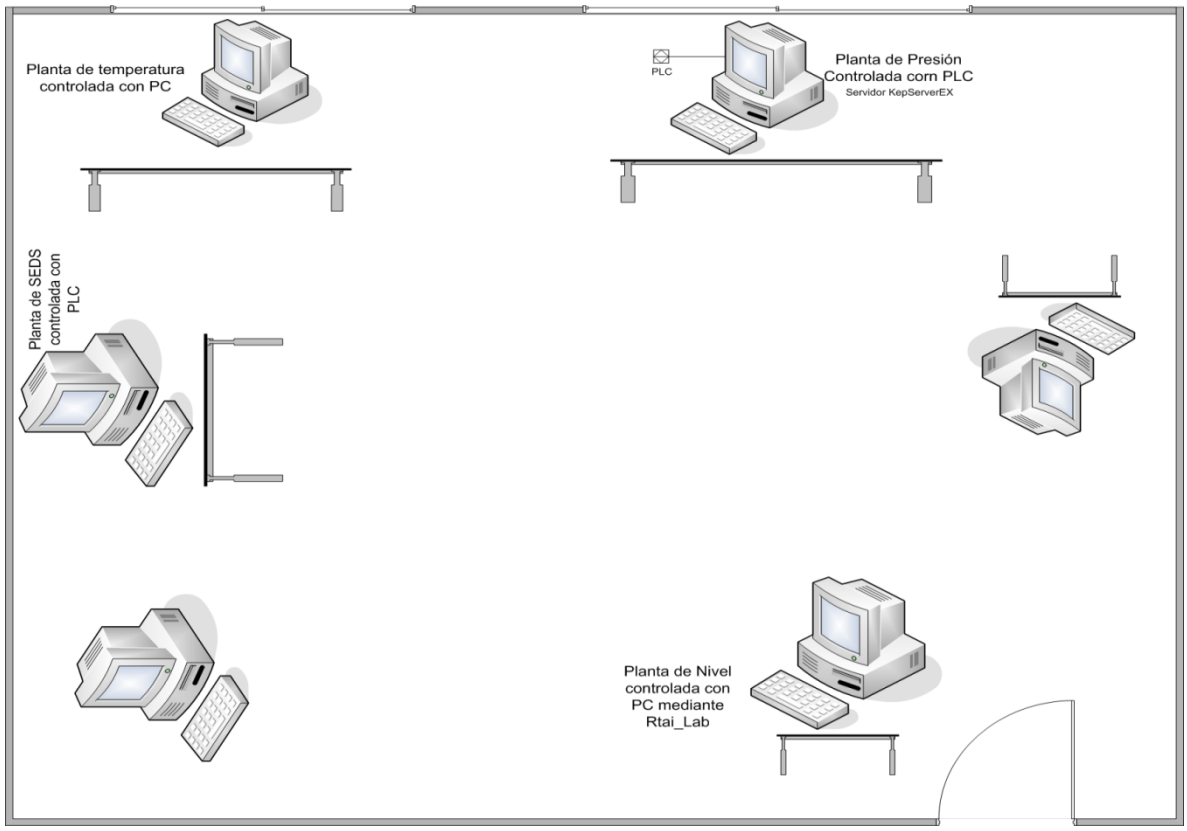
La ilustración H-1 presenta el esquema general de integración mediante el estándar OPC DCOM, el cual está conformado por las plantas, los esquemas de control, el MCODM, MCODR, el servidor OPC DCOM kepserverEX y el módulo de monitoreo y supervisión. Las plantas de Nivel, Temperatura y Presión son controladas en Rtai-Lab, Matlab y el PLC Micrologix 1500 respectivamente, y los datos obtenidos son enviados al servidor OPC KeepserverEX para que sean capturados por el MM&SW y se puedan representar gráficamente.

Ilustración H-1: Esquema general de la integración de los elementos de control mediante el estándar OPC DCOM

Fuente: Propia

La ilustración H-2 presenta la distribución física de la arquitectura de PCs del LCP; en el equipo de la planta de presión se encuentra instalado un servidor OPC KepserverEX, el cual captura información del PLC micrologix 1500 y de la planta virtual "intercambiador de calor" implementada en Matlab. En el equipo de la planta de Nivel se encuentra instalado el módulo cliente OPC Rtai-Lab, el cual le envía información al servidor OPC KepserverEX que se encuentra instalado en la máquina virtual de la planta de SEDS. EL Servidor del sistema de monitoreo y supervisión que se encuentra instalado en el equipo servidor del LCP, mediante unas librerías OPC DCOM, captura la información de los dos servidores OPC (los que se encuentran instalados en los equipos de las plantas de presión y SEDs).

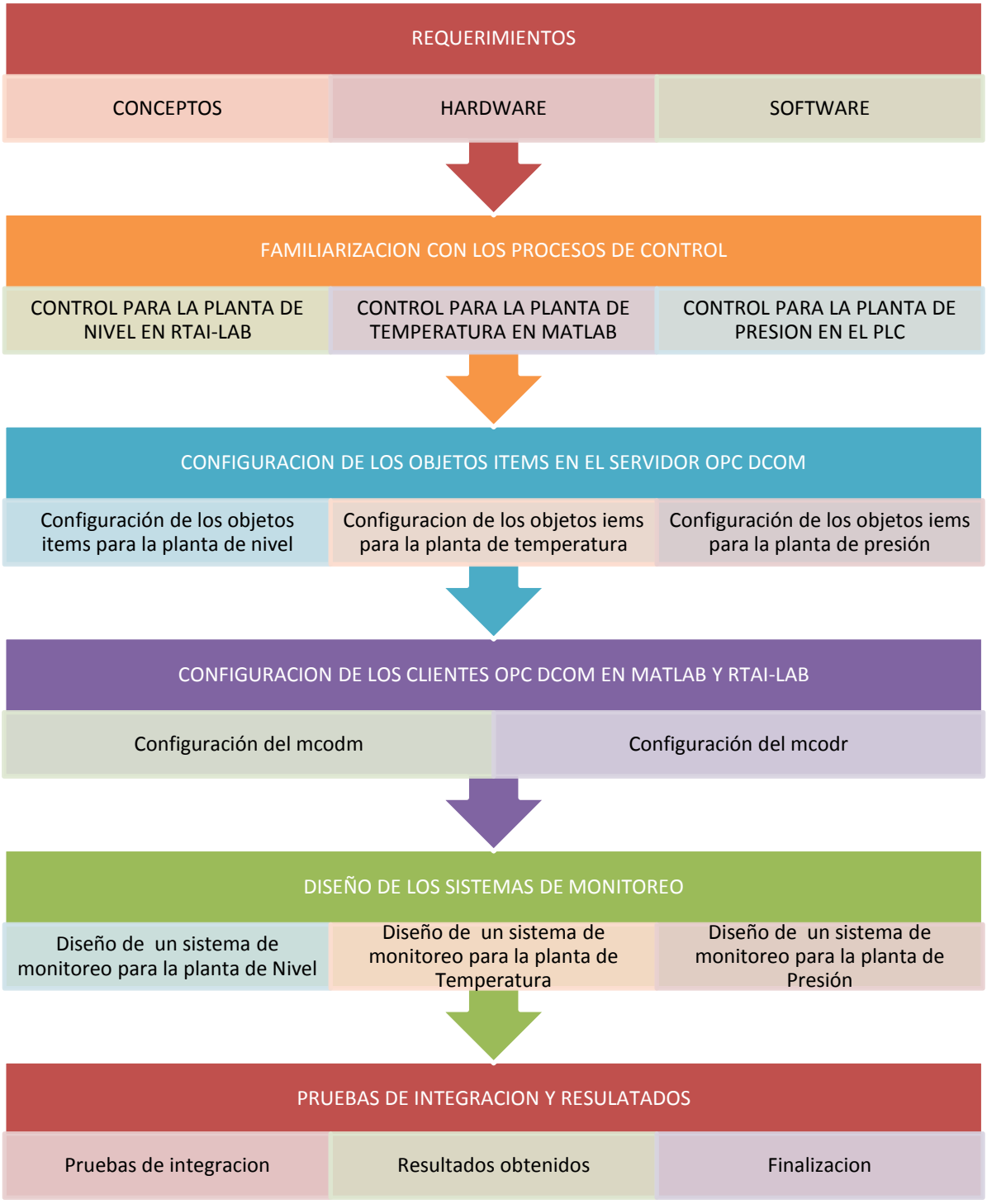
Ilustración H-2: Distribución física de la arquitectura en la red de PCs del LCP del PIAI



II. PROCEDIMIENTO

Los pasos necesarios para la integración de Matlab, Rtai-Lab y el PLC micrologix 1500 se representan en la ilustración H-3 y se explican en los numerales IV, V, VI de este anexo.

Ilustración H-3: Pasos para la integración de Matlab, Rtai-Lab y el PLC micrologix 1500.



III. REQUERIMIENTOS

Para la realización de esta práctica se deben manejar los siguientes conceptos:

- Control basado en PC.
- Plataformas de control basado en PC: Matlab y Rtai-Lab.
- Controladores lógicos programables: PLC Micrologix 1500.
- OPC estándar DCOM
- Sistemas de monitoreo y supervisión

La tabla H-1 presenta los requerimientos hardware y software necesarios en la integración de Matlab, Rtai-Lab y el PLC.

Tabla H -1: Requerimientos para la integración de plataformas académicas mediante el estándar OPC DCOM

Requerimientos					
Componentes		**Descripción**	**Hardware**	**Software**	
Procesos	**Control de en Rtai-Lab para la planta de Nivel**	Se encarga de controlar la planta de Nivel y de la comunicación remota con el servidor OPC DCOM.	Elementos de la planta de nivel. Un PC con conexión a una red LAN.	**Rtai-Lab:** Se encuentra instalado en el computador de la planta de nivel del LCP. Aquí se efectúa el control de la planta de nivel.	**Diagrama de bloques de un control PID serie para la planta de nivel.** Verificar que en la carpeta home/grupoati del computador de la planta de Nivel se encuentre el archivo Pidserieopc
					MCODR: Paleta de bloques OPC DCOM Verificar que los bloques OPC se encuentran instalados en el computador de la planta de nivel (ingresar a scilab, scicos y en el menú palets verificar que exista una paleta llamada OPC). Para mayor información dirigirse al numeral III del anexo E (guía de instalación y configuración de los clientes OPC DCOM/XML en Rtai-Lab).
	Control de la planta de temperatura en Matlab	Se encarga de controlar la planta de temperatura y de la comunicación remota con el servidor OPC DCOM.	Un PC con Matlab y conexión a una red LAN.	***Matlab:*** Se encuentra instalado en el computador de la planta de Presión del LCP	**MCODM: Módulo cliente OPC DCOM en Matlab** Se debe verificar que en Matlab se encuentren los bloques OPC DCOM: ingresar a la librería de Simulink y en la barra de búsqueda digitar OPC, donde deben aparecer tres bloques (*opc read, opc write, opc configuration*).
					Diagrama de bloques para el control de temperatura: ingresar a la carpeta **control** ubicada en el disco D del equipo ubicado en el computador de la planta de presión y verificar la presencia del archivo: heatex.m
	Control de presión en el PLC para la planta de Nivel	Se encarga de controlar la planta de presión y de la comunicación remota con el servidor OPC DCOM.	Elementos de la planta de presión incluido el PLC Micrologix 1500. Un PC con conexión a una red LAN.	**Factory Talk de Rockwell**: Este software se utiliza para la programación del PLC, verificar la instalación de este paquete en el computador de la planta de presión.	

Servidor OPC DCOM	Encargado de integrar la información de las plataformas de control y el PLC.	➢ Un PC con sistema operativo Windows y conexión a una red LAN para la adquisición de datos desde Matlab y el PLC. ➢ Un PC con sistema operativo Windows y conexión a una red LAN. (Para la adquisición de datos desde Rtai-lab)	servidor KEPServerEX V4.5 que se encuentra instalado en el computador de la planta de Presion Servidor KEPServerEX V4.5, que se encuentra instalado en la máquina virtual en el computador de la planta SEDs	➢ Verificar que en el computador de la planta de presión se encuentre instalado el servidor KepserverEX. En caso de no estar instalado seguir las instrucciones del anexo B (MANUAL DE USUARIO DEL MÓDULO SERVIDOR OPC DCOM KEPServerEx) ➢ Iniciar la máquina virtual XpOPC ubicada en el computador de la planta de SED y verificar que allí se encuentre instalado el Servidor KepserverEX. En caso de no estar instalado seguir las instrucciones del anexo B (MANUAL DE USUARIO DEL MÓDULO SERVIDOR OPC DCOM KEPServerEx)
Sistema HMI	Encargado de permitir el diseño de mímicos y el monitoreo de los procesos mediante la web.	Un PC que cumple la función de servidor WEB del MM&SW. Un PC que cumple la función de cliente WEB del MM&SW.	Módulo de Monitoreo y supervisión Web Se encuentra instalado en la máquina virtual **hmiwebopc** del computador que hace de servidor en el LCP.	Se debe verificar que en el disco D del computador servidor del laboratorio de control de procesos se encuentre la máquina virtual **hmiwebopc** Se debe elegir un computador cualquiera con acceso a Internet que haga de cliente del hmiwebopc.

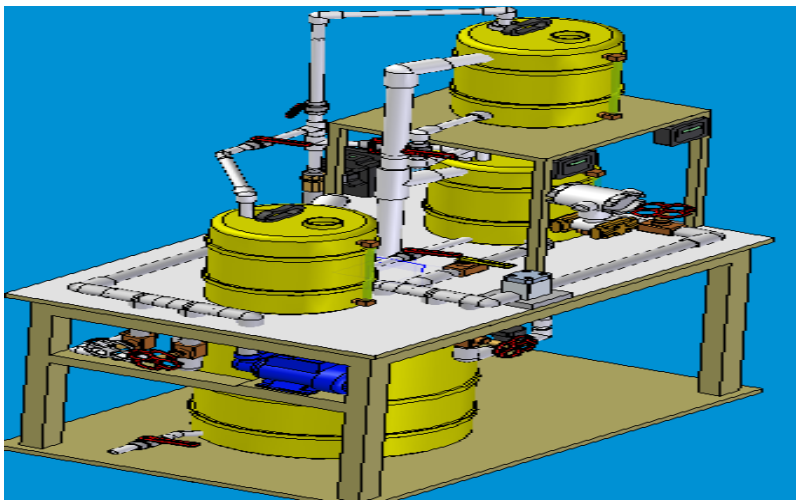
Fuente: Propia

IV. FAMILIARIZACION CON LOS PROCESOS DE CONTROL

1. Control PID implementado en Rtai-Lab para la planta de Nivel

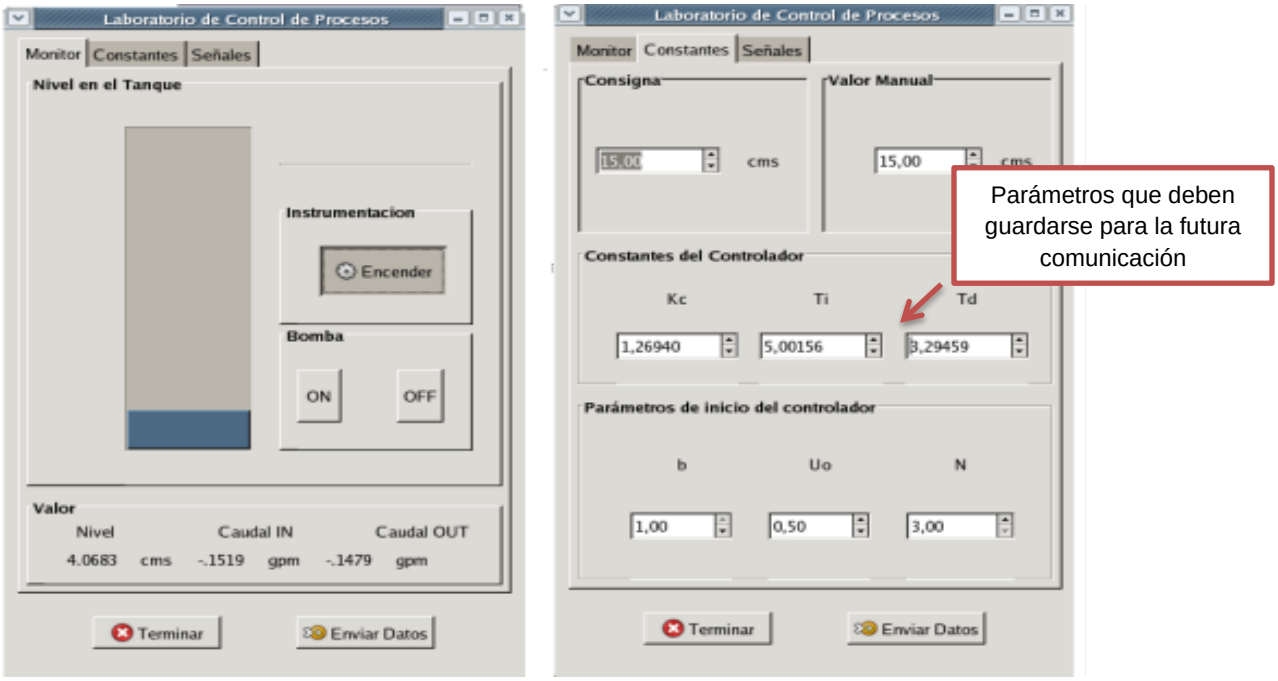
El Laboratorio de Control de Procesos de la Universidad del Cauca, del programa de Ingeniería en Automática Industrial, cuenta con una Planta de Nivel que combina la implementación de un sistema de control en tiempo real soportado en Rtai-Lab, empleando estrategias PID de realización serie y paralela de tipo industrial. El control en Rtai_Lab se encuentra instalado en el PC asociado a esta misma planta. La planta está compuesta por un circuito hidráulico y la instrumentación necesaria para efectuar el control de flujo de agua por el circuito o el control de nivel de agua en uno de los tanques. Consta de un tanque plástico grande para almacenamiento, 3 tanques plásticos pequeños, una bomba centrífuga QB 128 que entrega un caudal máximo de 38 L/min, una válvula de control de flujo W.E. Anderson ABV111 con un actuador AMC-100A serie 4078, el cual puede ser usado para un rango de entrada de 1-5 V o 4-20 mA, un sensor de flujo Metalex 525 +GF+SIGNET con su respectivo transmisor de flujo +GF+SIGNET 8550-1 con salida de 4-20 mA, un transmisor de nivel implementado por medio del transmisor de presión diferencial YOKOGAWA EAJ110 con salida de 4-20 mA, tubería PVC de ¾", válvulas de bola de accionamiento manual y un interruptor manual que aplica alimentación de 120 VAC al sistema. Ver Ilustración H-3.

Ilustración H-4: Planta de Nivel controlada en Rtai-Lab.



Antes de realizar la comunicación de la planta de nivel con el servidor OPC DCOM, se debe implementar y sintonizar un control PID serie para el nivel, para ello se debe ejecutar la guía “CONTROL REGULATORIO PID SERIE”, existente en el laboratorio de control de procesos del PIAI. Cuando se haya realizado esta práctica se deben copiar los parámetros Kc, Ti, y Td los cuáles serán utilizados más adelante en la configuración de valores del servidor KepserverEX, ver ilustración H-5.

Ilustración H-5: Parámetros para la planta de nivel.



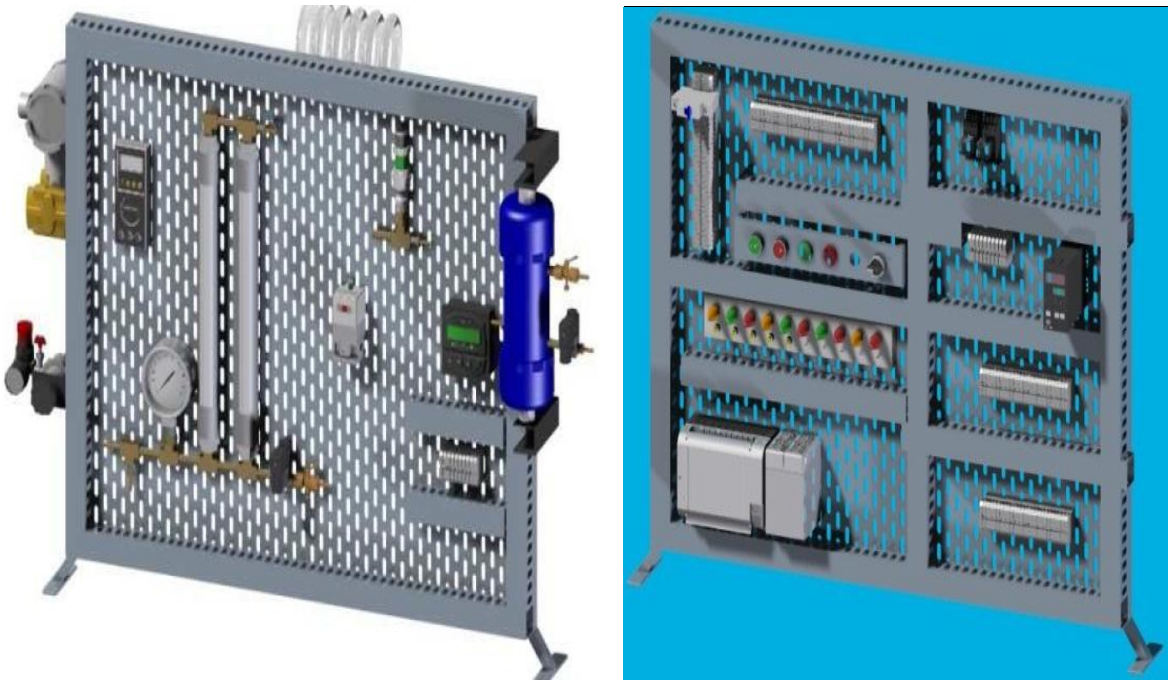
Fuente: **Propia**

2. Control regulatorio de la planta de presión del laboratorio de control de procesos

La planta de presión y flujo de aire (presión) es un sistema compuesto por un compresor, un circuito de alimentación de aire a presión, filtros, reguladores de presión, un circuito neumático, un transmisor ciego de presión, indicadores de presión, rotámetros, electroválvulas, una servo válvula, un transmisor indicador de

presión, un controlador de presión, un tanque de almacenamiento de aire a presión y muchos otros elementos. La planta de presión está diseñada para implementar un proceso de aire a presión sometido a disturbios tanto en la entrada como en la salida de aire. El propósito de la planta es poder diseñar e implementar un control regulatorio de presión por medio de un PLC micrologix 1500 de Allen Bradley. Se dispone de un computador asociado a esta planta con la herramienta Factory Talk que permite programar el PLC. La automatización de la planta permite la comunicación directa de los transmisores y actuadores a un PLC Micrologix 1500 serie C de Allen-Bradley. Ver Ilustración H-6.

Ilustración H-6: Planta de Presión del LCP



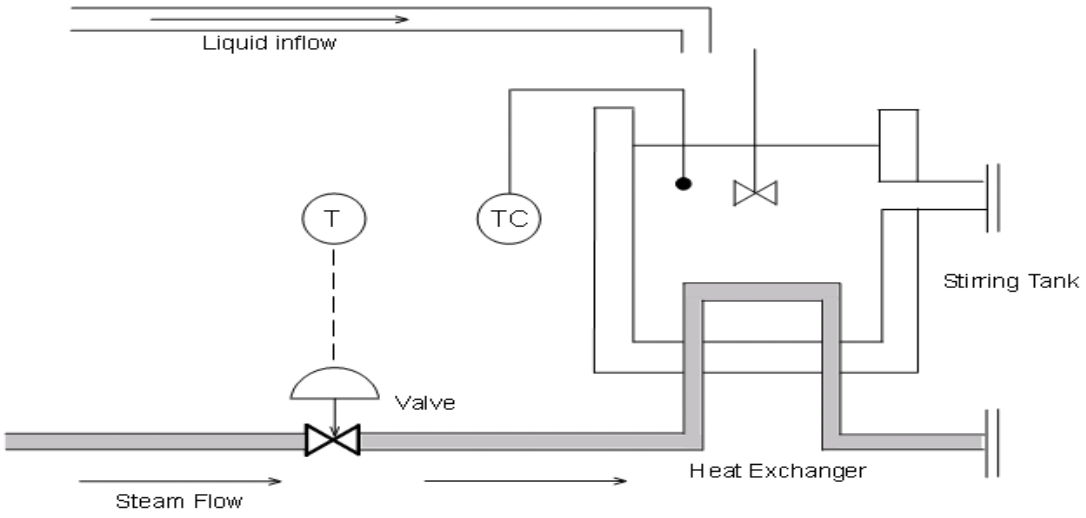
Fuente: Propia

Para poder configurar los parámetros en el servidor OPC kepserverEX, se debe realizar la práctica de control de la planta de presión mediante el **PLC micrologix 1500.** (Programar el PLC con el diagrama en escalera existente en la carpeta V2, **Presión**, ubicada en el disco D del computador de la planta de Presión).

3. Control PI implementado en Matlab para regular la temperatura de un Reactor Químico

Este proceso no pertenece a las prácticas convencionales del LCP, pero se incluyó para mostrar las funcionalidades del esquema general de integración. Este proceso se ejecuta en el computador asignado a la planta de presión del LCP. Matlab presenta una demostración de un reactor químico "tanque agitado" el cual se controla mediante un PI. La entrada superior proporciona el líquido que se mezcla en el tanque, el cual debe ser mantenido a una temperatura constante mediante la variación de la cantidad de vapor suministrado al intercambiador de calor (tubo inferior) a través de su válvula de control. Las variaciones en la temperatura del flujo de entrada son la principal fuente de las perturbaciones en este proceso, ver ilustración H-7.

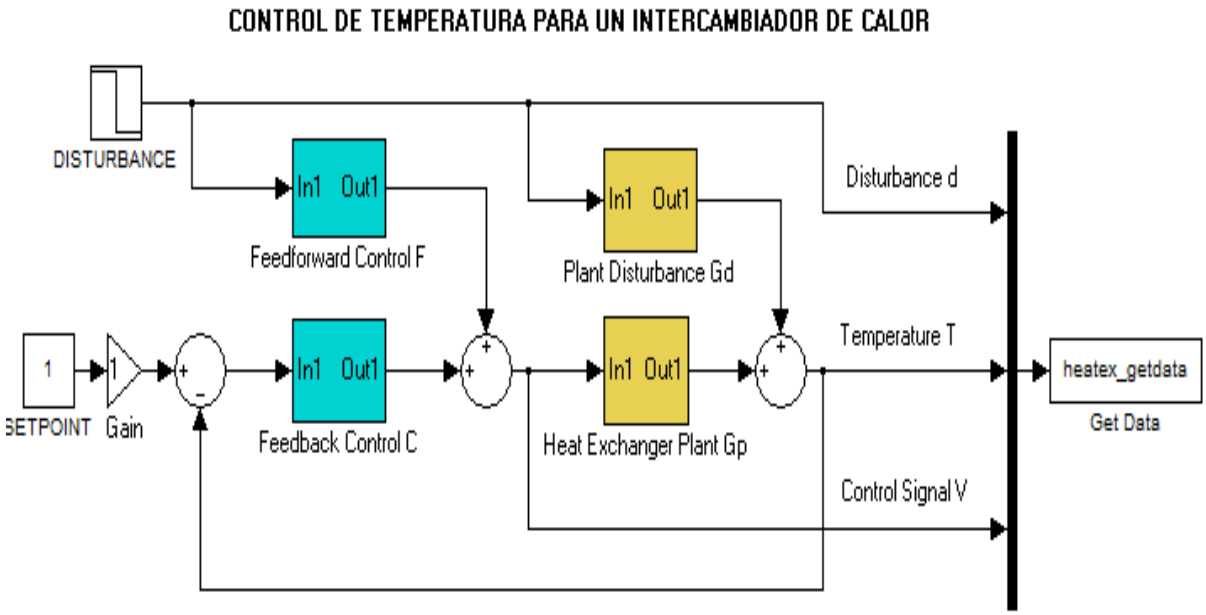
Ilustración H-7: reactores de agitación con intercambiador de calor.



Fuente: Matlab

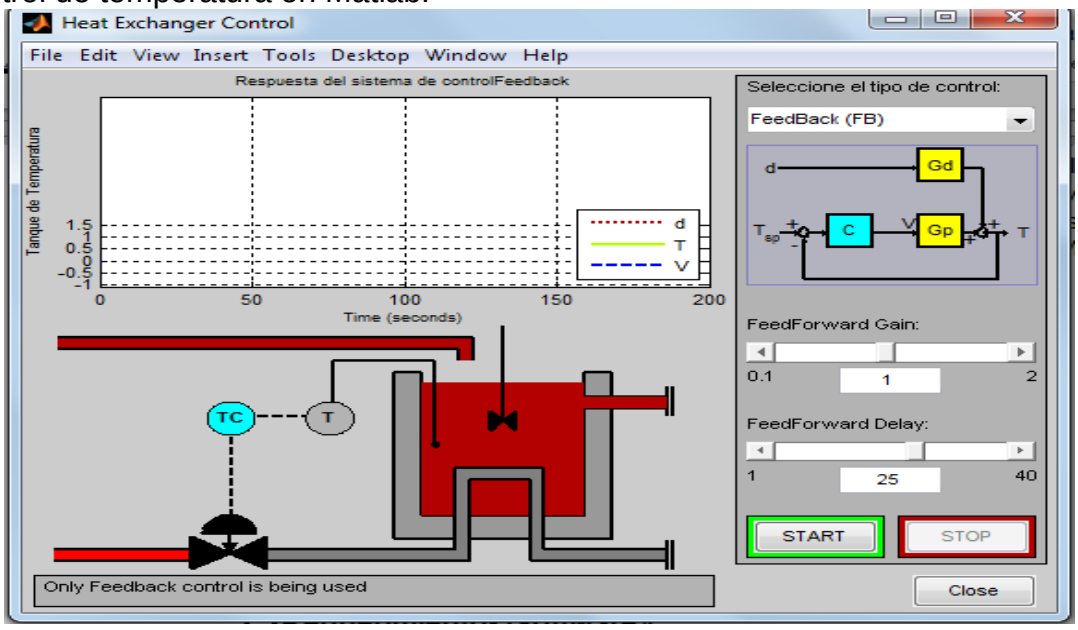
La Ilustración H-8 presenta el diagrama de bloques en Simulink para el control de este proceso y la ilustración H-9 presenta la pantalla que permite establecer y visualizar los parámetros del control de temperatura implementado en Simulink.

Ilustración H-8: Diagrama de bloques en Simulink



Fuente: Matlab

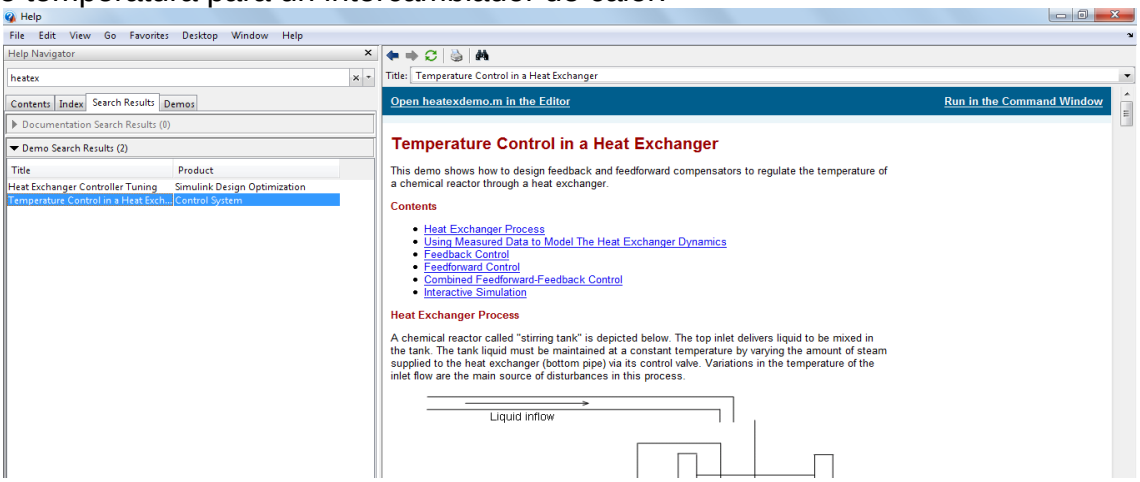
Ilustración H-9: Pantalla que permite establecer y visualizar parámetros para el control de temperatura en Matlab.



Fuente: Matlab

Para familiarizarse con esta planta se debe seguir la guía presente en la ayuda de Matlab. En el buscador de la ayuda se debe digitar **heatex**, y ejecutar el programa como lo indica la ilustración H-9.

Ilustración H-10: Ayuda de Matlab donde presenta el funcionamiento del control de temperatura para un intercambiador de calor.



Fuente: Matlab

V. CONFIGURACION DE LOS OBJETOS ÍTEMS EN EL SERVIDOR OPC DCOM

Después de la familiarización con las plantas de nivel, presión y temperatura, se inicia el proceso de interconexión de las plantas, mediante el estándar OPC DCOM. Para el desarrollo de esta práctica se requiere dos servidores OPC KeepserverEX, uno que se encuentra instalado en el computador asignado a la planta de presión (para la captura de datos tanto desde el PLC que gobierna la planta de presión, como desde Matlab que simula y controla la planta del intercambiador de calor), y el otro en la máquina virtual **XpOPC** instalado en el computador de la planta de SED encargado de la captura de datos desde Rtai_Lab.

1. Configuración de los objetos ítems en el servidor para el control de la planta de presión

1.1. Procedimiento para comunicar el servidor OPC KepserverEX con el PLC

Para comunicar el servidor OPC DCOM con el PLC que controla la planta de presión se deben realizar los siguientes pasos:

a) Ubicarse en el computador de la planta de Presión.
b) Verificar que el PLC se encuentre conectado al computador por medio del cable serial (DF1).
c) Encender el computador y la fuente que alimenta a la planta y al PLC.
d) Descargar en el PLC el diagrama en escalera **Presión** que se encuentra en la carpeta V2 del disco D.
e) Desactivar el driver del PLC en el programa Rslink.

A continuación se presentan los pasos que se deben seguir para la configuración de las tags en el servidor OPC KepeserverEX:

En la sección II del anexo B se presenta una guía para la configuración de tags desde PLC.

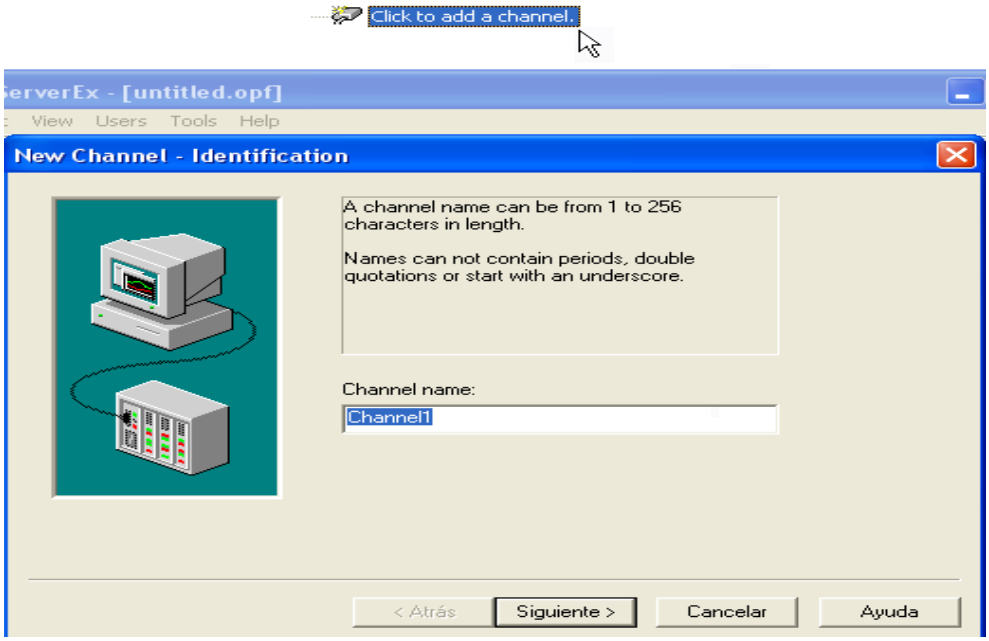
Ejecutar el servidor (hacer clic en el icono 

) ubicado en el escritorio del equipo de la planta de presión. Si no está instalado seguir las instrucciones descritas en el numeral II del anexo B (manual de usuario del módulo servidor OPC DCOM KEPServerEx).

1.1.1. Crear un canal de comunicación para el PLC

Se debe crear un canal de comunicación, para esto se hace clic en el icono "*click to add channel"* en el panel izquierdo, ver ilustración H-10.

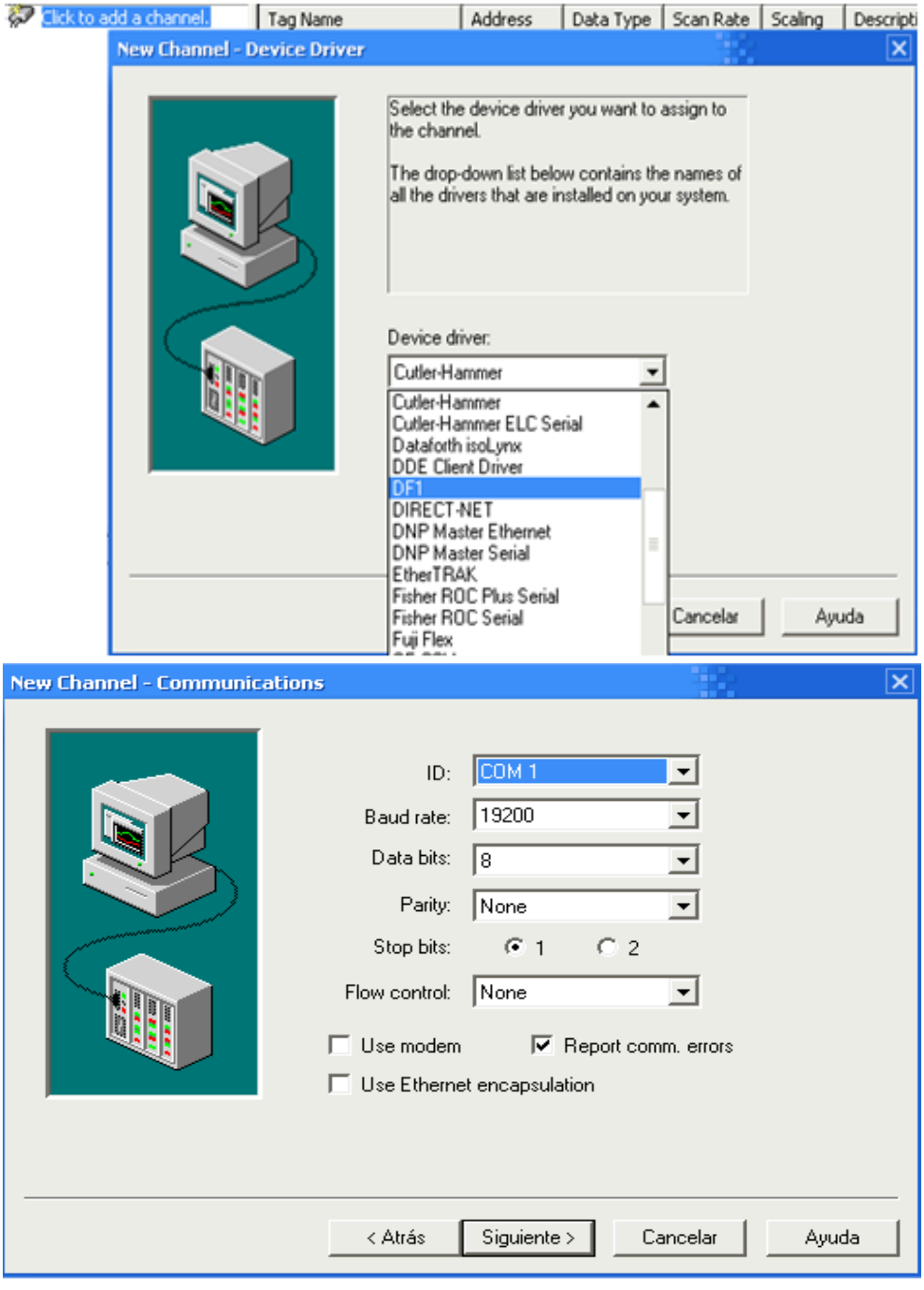
Ilustración H-11: Asistente de configuración del canal de comunicación

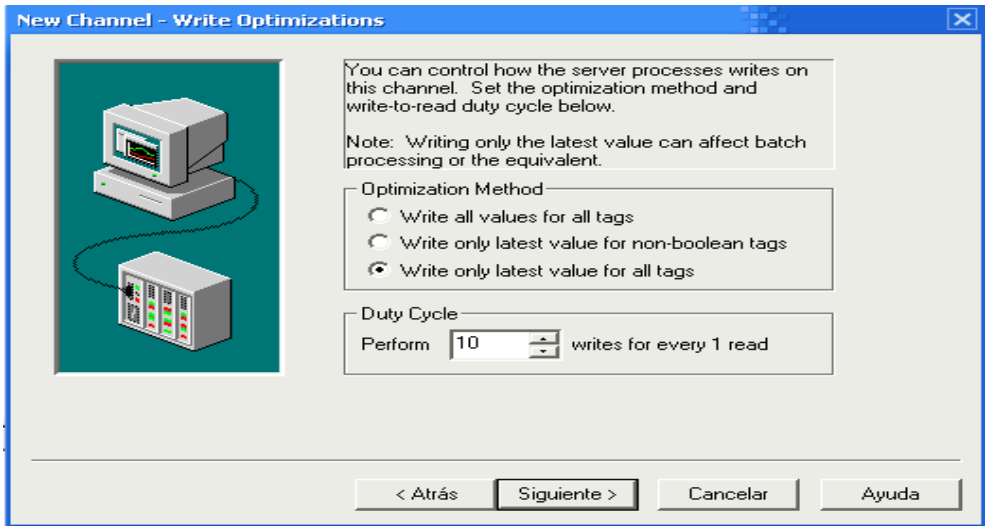


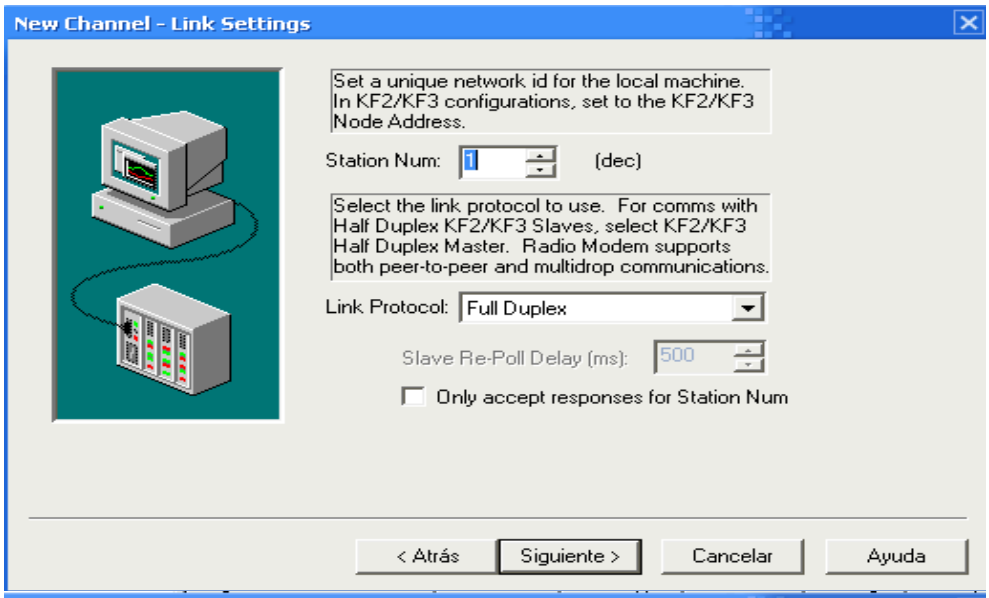
Fuente: propia

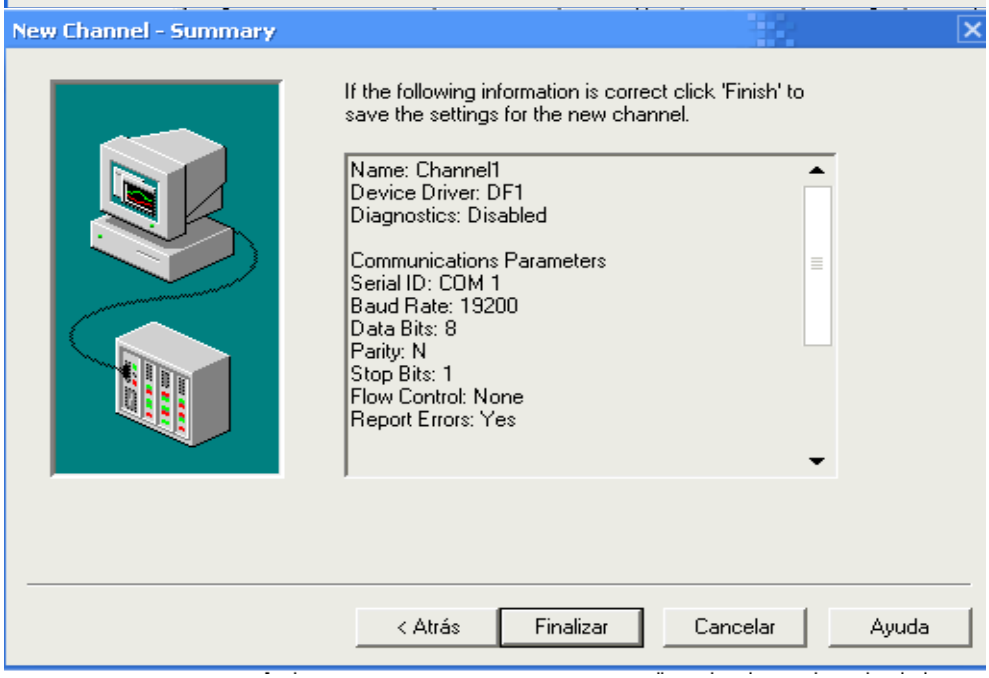
En este asistente se configuran todas las características del canal de comunicación, luego de darle un nombre al canal (Channel1) se da clic en el botón <siguiente> y se configura el driver para “DF1”, luego se configuran las comunicaciones, el ID se configura en “COM 1”, la velocidad de transmisión en “19200 Baudios”, el tamaño de la palabra en “8 bits”, sin paridad, con un bit de stop y habilitada la opción de reporte de los errores de comunicación. En la siguiente pantalla se configuran las opciones de optimización de escritura en donde no se modifican los valores por defecto. Por último se configuran las opciones del enlace, se asigna un identificador de red para la máquina local o la dirección del nodo, en este caso “*Station Num*: 1” y el protocolo de enlace debe ser “Full *Dúplex*”. Al dar clic en el botón <siguiente> se presenta un resumen de la configuración y se finaliza el asistente, ver ilustración H-11.

Ilustración H-12: Configuración del canal de comunicación en servidor KEPserverEX




New Channel - Write Optimizations
You can control how the server processes writes on this channel. Set the optimization method and write-to-read duty cycle below.
Note: Writing only the latest value can affect batch processing or the equivalent.
Optimization Method
Write all values for all tags
Write only latest value for non-boolean tags
Write only latest value for all tags
Duty Cycle
Perform 10 writes for every 1 read
< Atrás
Siguiente >
Cancelar
Ayuda



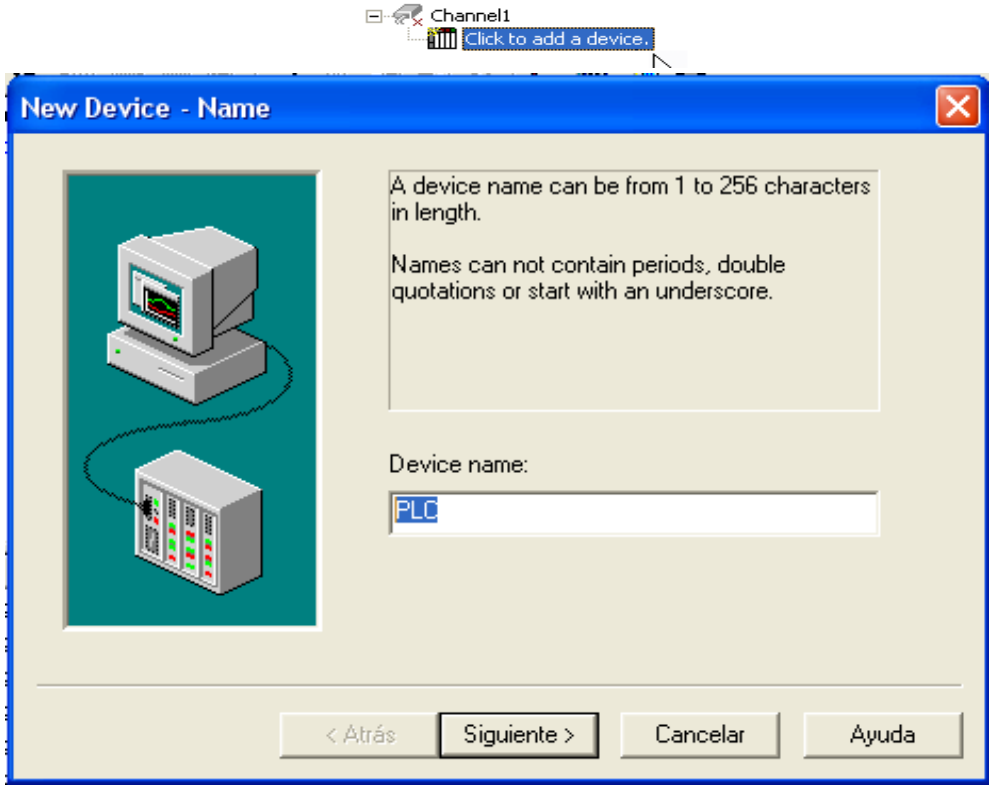


Fuente: propia

1.1.2. Adicionar el dispositivo PLC micrologix1500

Una vez se tenga el canal creado y configurado se debe crear un "*device*" que este caso es el PLC (*MicroLogix* 1500), ver ilustración H-13.

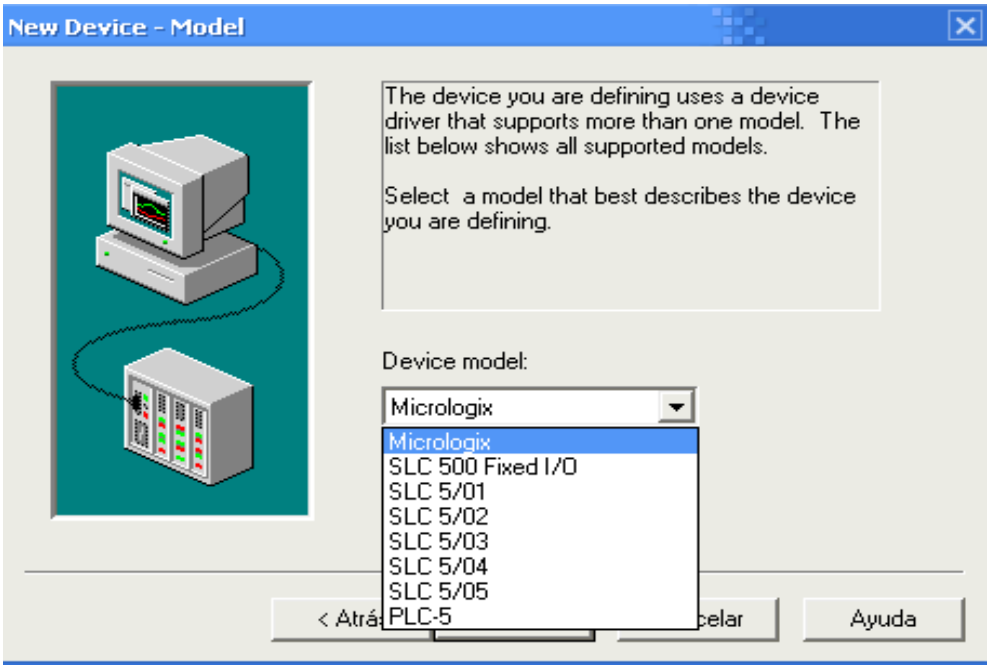
Ilustración H-13: Asistente de configuración del dispositivo en servidor KEPserverEX

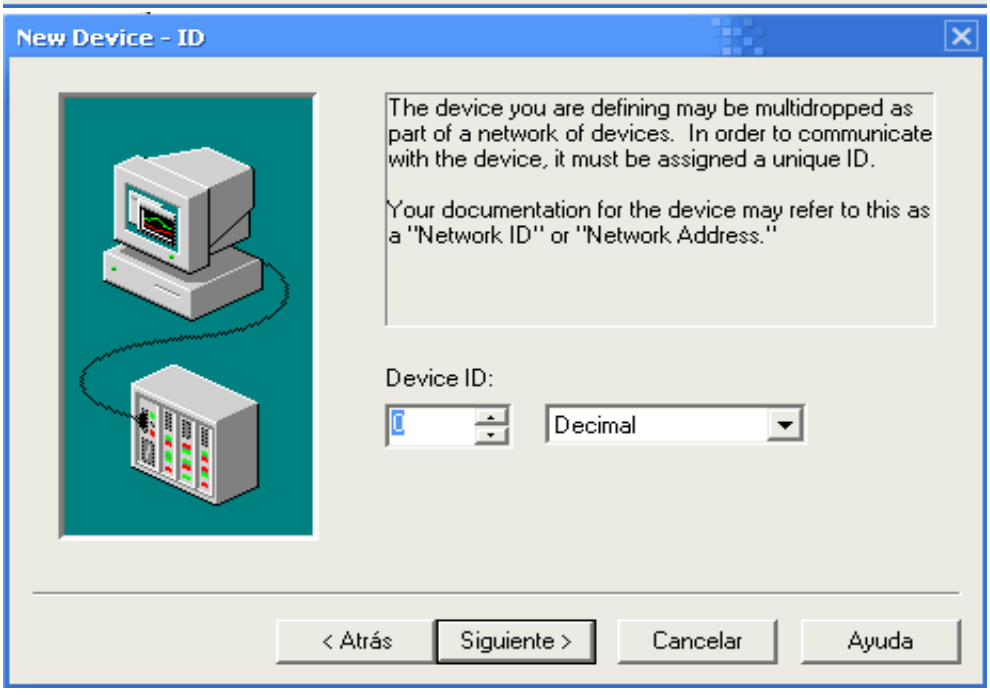


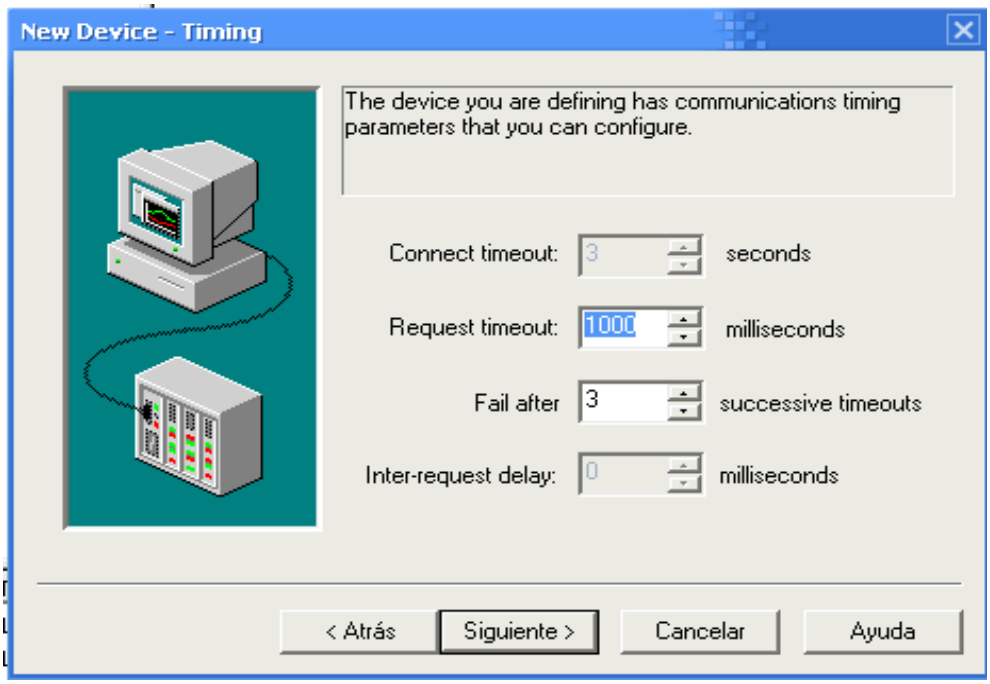
Fuente: propia

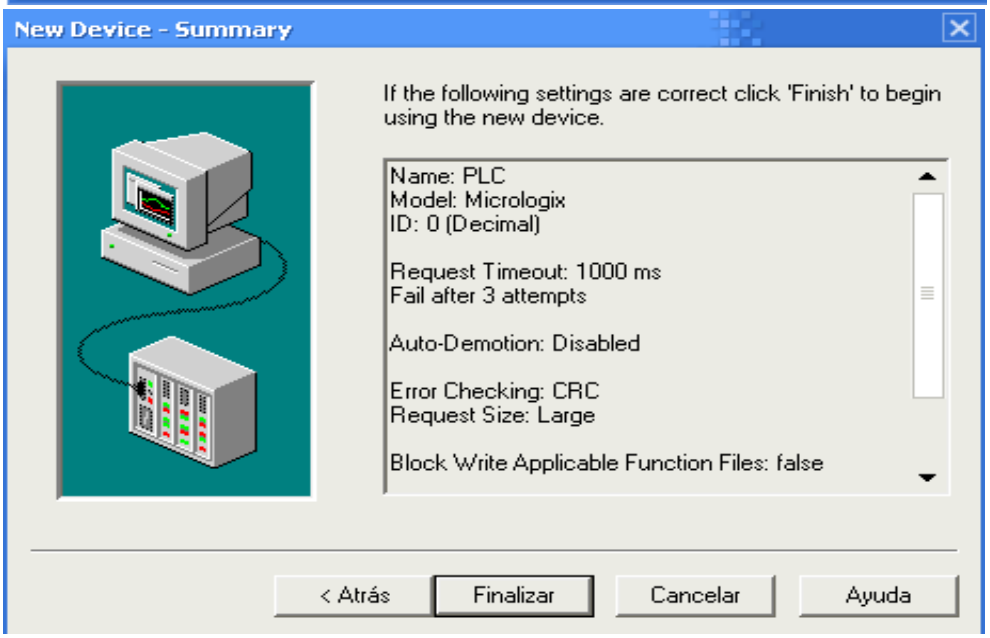
En este asistente se crea y configura un dispositivo conectado al canal que se configuró anteriormente, en la primera pantalla se asigna un nombre al dispositivo (PLC), se da clic en el botón <siguiente> y se configura "micrologix", luego se asigna un identificador de red "*Device* ID: 0", en las siguientes 4 pantallas del asistente, "*Timing", "Auto-Demotion", "Protocol Settings", "Function File Options",* no se modifican los parámetros por defecto, por último se presenta un resumen de la configuración y se finaliza el asistente, ver ilustración H-14.

Ilustración H-14: Pasos para la configuración del dispositivo para el canal seleccionado









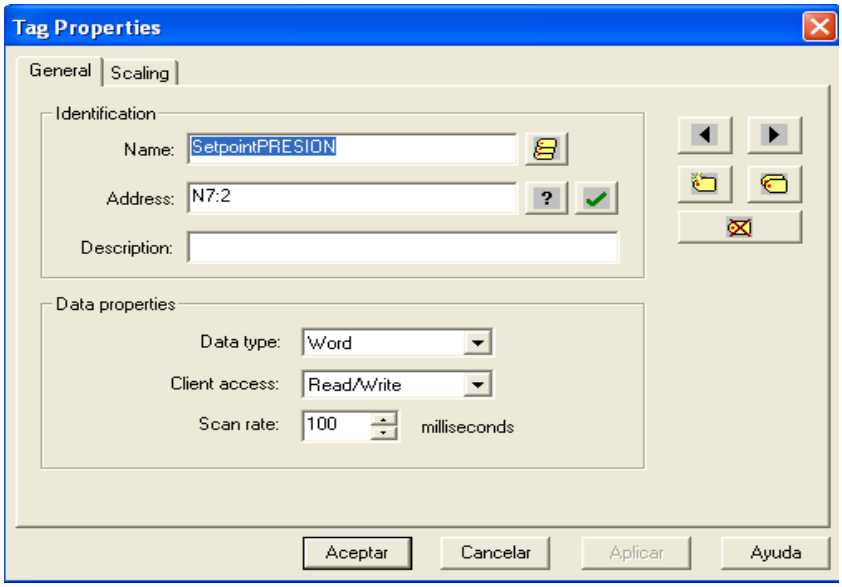
Fuente: propia

1.1.3. Crear el grupo que contiene los objetos ítems

Teniendo el canal y el dispositivo configurado, es necesario crear un grupo en donde se almacenan los objetos ítems del equipo. Sobre el icono del

dispositivo se hace clic derecho, donde se presenta un formulario, ver ilustración H-15 que permite crear uno a uno los objetos ítems que maneja el PLC. Se debe asignar la misma dirección que posee la tag en el PLC, una descripción de la misma. Para el diagrama en escalera **presión** existente el disco D, carpeta V2, se puede seguir la configuración que se presenta en la tabla H-2. En la columna CHEQUEO de la tabla H-2 se debe colocar un chulo si la dirección es correcta o la nueva dirección si esta ha cambiado. Nota al ingresar los nombres de las tags u objetos ítems en el campo **Name** del formulario de la ilustración H-15 se recomienda colocar una palabra o expresión sin espacios.

Ilustración H-15: Formulario para la creación de objetos ítems en servidor KEPserverEX



Fuente: propia

Tabla H -2: Clasificación de las variables (tags) del proceso de presión con su respectiva dirección de memoria del PLC.

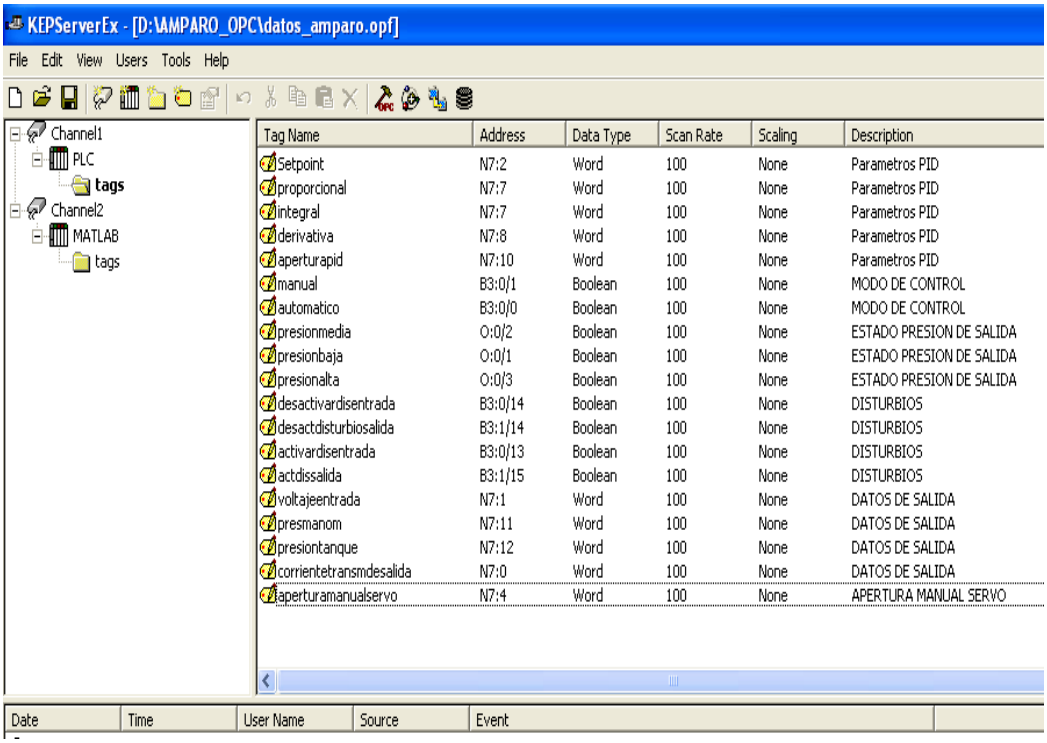
SEÑALES	DIRECCION	TIPO DESEÑAL	CHEQUEO
Tx GF SIGNET 2450	I:1.0	4 – 20 Ma	
Tx Honeywell	I:1.1	1 – 5 V	
PLC a Servo Válvula	O:2.0	0 – 10 V	
Botón Start Compresor	O:0/0	Discreta	
Botón Stop Compresor	O:0/1	Discreta	
Presión Tx GF SIGNET	N:7:0	0-25 psi	

2450			
Presión Tx Honeywell	N:7:1	0-25 psi	
Setpoint Presion	N:7:2	0-100%	
Proporcional	N:7:3	Entero	
Apertura Manual de Servoválvula	N:7:4	0-100%	
Apertura de Servoválvula	N:7:6	0-100%	
Integral	N:7:7	Entero	
Derivativa	N:7:8	Entero	
Apertura PID servoválvula	N:7:9	0-100%	
Voltaje de entrada	N:7:1	0-100%	
Presión media	O:O/2		
Presión baja	O:O/1		
Presión Alta	O:O/3		

Fuente: Propia

Al configurar todas las tags para la planta de presión, el servidor KepserverEX se debe de visualizar como en la ilustración H-16.

Ilustración H-16: servidor KepserverEX con las tags configuradas para la comunicación con el PLC Micrologix encargado de controlar la planta de presion

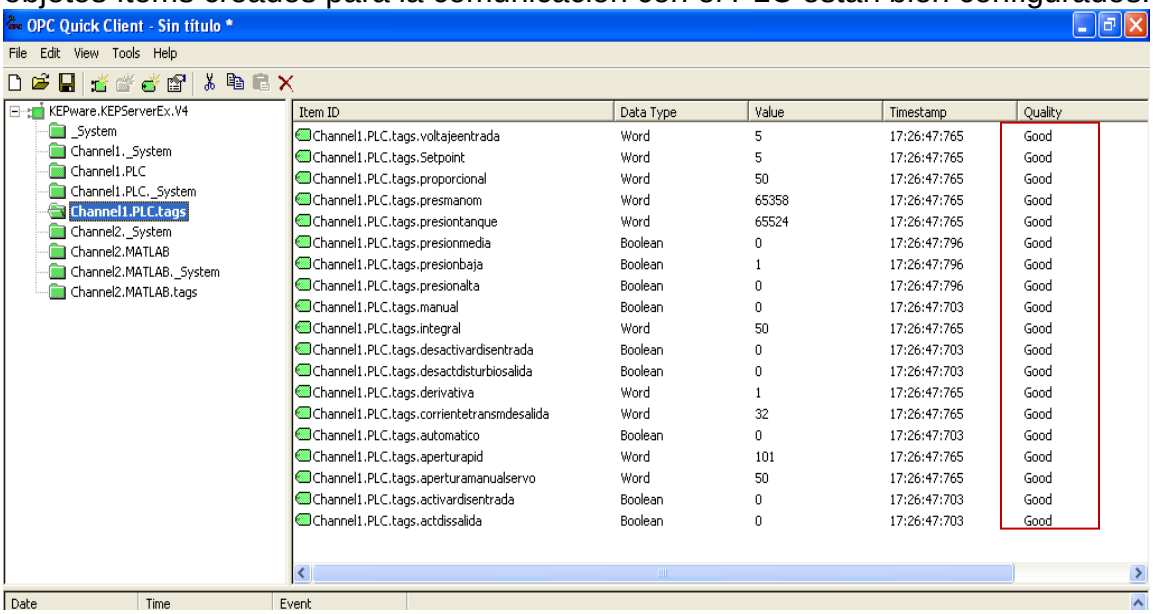


Fuente: Propia

1.2. Verificación la comunicación entre el PLC y el servidor OPC DCOM

Al hacer clic en el icono  (cliente OPC) en el servidor KepserverEX debe aparecer una pantalla con todas las variables configuradas en el servidor, se debe seleccionar la dirección donde están las tags para el PLC, ver ilustración H-17, y verificar que aparezcan en **Quality** *good*.

Ilustración H-17: pantalla del cliente OPC DCOM que permite verificar si los objetos ítems creados para la comunicación con el PLC están bien configurados.



Fuente: Propia

Si la comunicación entre el PLC Micrologix y el servidor no se establece, se debe verificar que:

- El puerto DF1 no esté ocupado con otro programa (Rslink).
- El PLC este efectuando el control.

2. **Configuración de las tags en el servidor KepserverEX para el intercambiador de calor en Matlab**

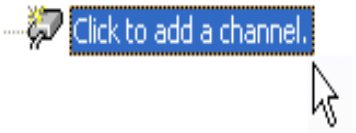
Ubicarse en el computador de la planta de presión, iniciar el servidor KepserverEX, y efectuar los pasos siguientes: crear un canal con el nombre:

Channel1, crear un dispositivo con el nombre: **Matlab, c**rear un grupo de tags con el nombre: **tags,** y crear todas las tags descritas en la tabla H-3.

2.1. Crear un canal de comunicación

Para crear un canal de comunicación se hace clic en el icono "*click to add channel"* en el panel izquierdo, ver ilustración H-18.

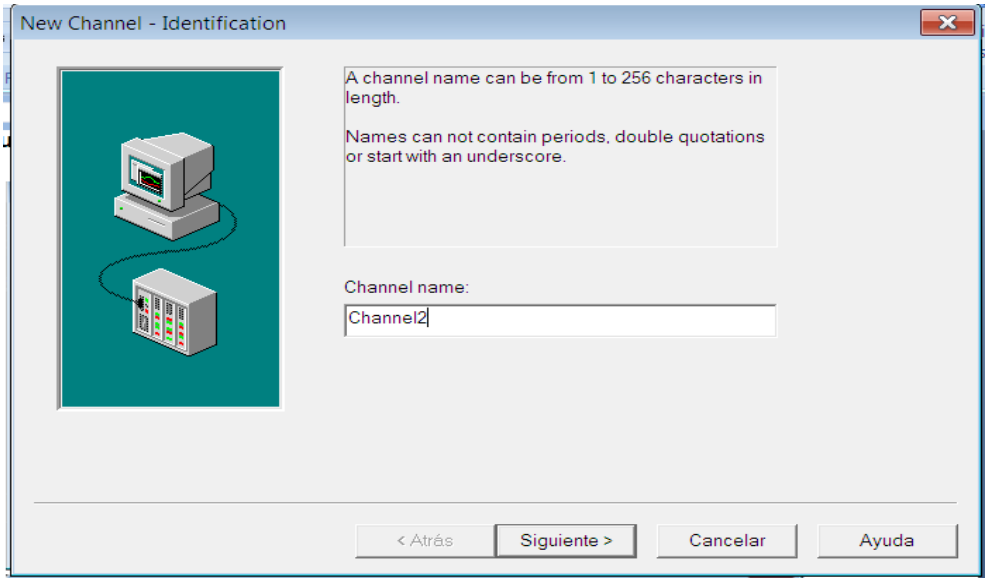
Ilustración H-18: Asistente de configuración de canal en servidor KEPserverEX

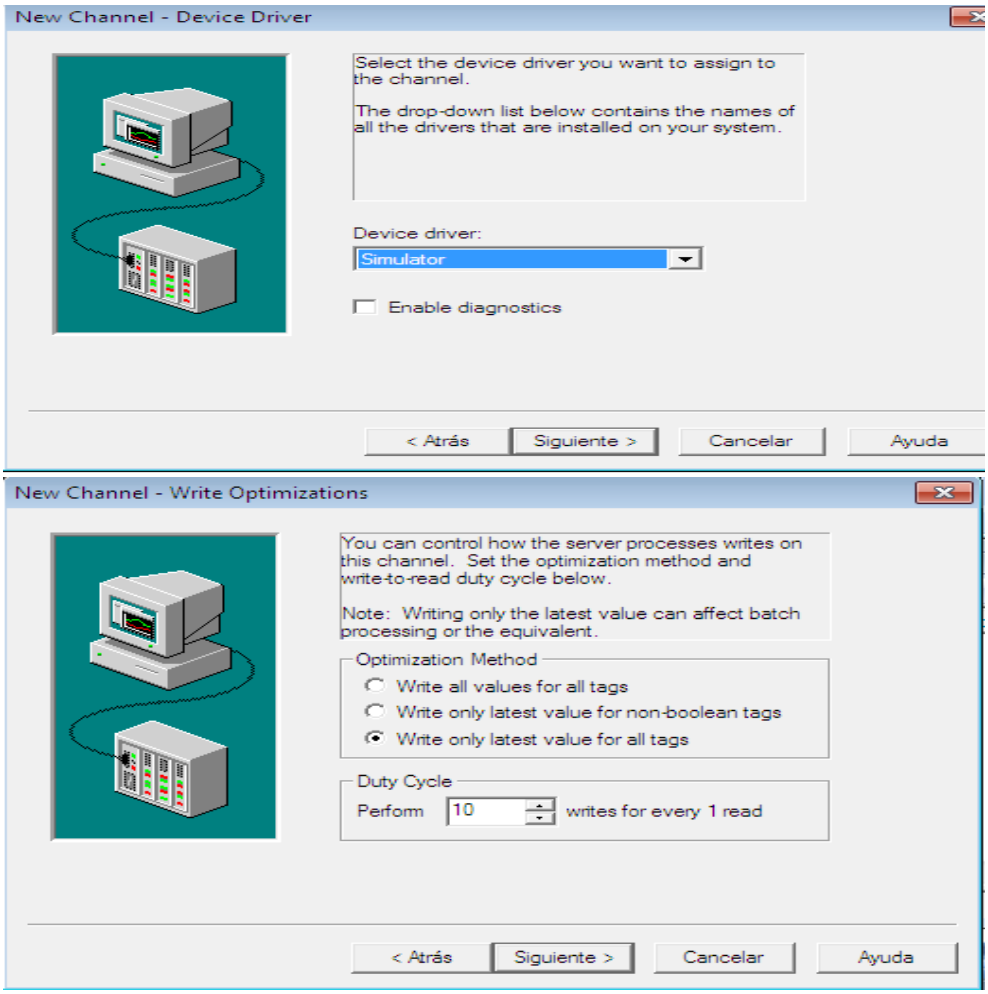


Fuente: propia

En este asistente se configuran todas las características del canal de comunicación, luego de darle un nombre al canal (Channel2) se da clic en el botón <siguiente> y se configura el driver que se va a usar, para este caso no se requiere driver por lo tanto se utiliza la opción "*simulator*", al dar clic en el botón <siguiente> se presenta una pantalla que permite la configuración de algunos parámetros y finalmente se cierra el asistente, ver ilustración H-19.

Ilustración H-19: Pasos para la configuración de canal en el servidor KEPserverEX para la planta en Matlab del intercambiador de calor



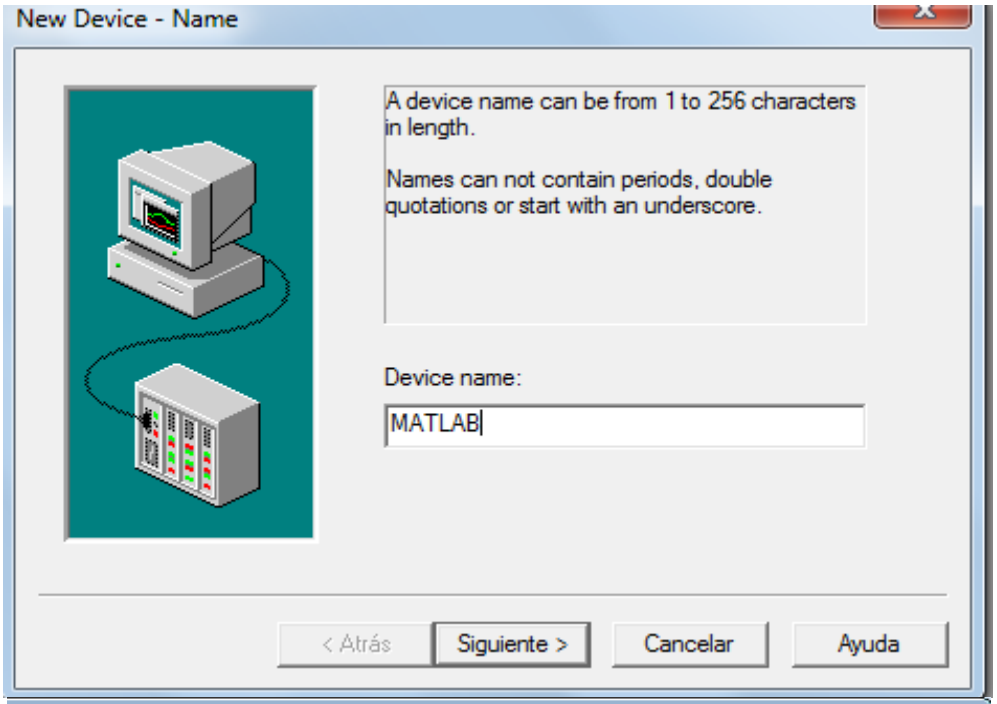


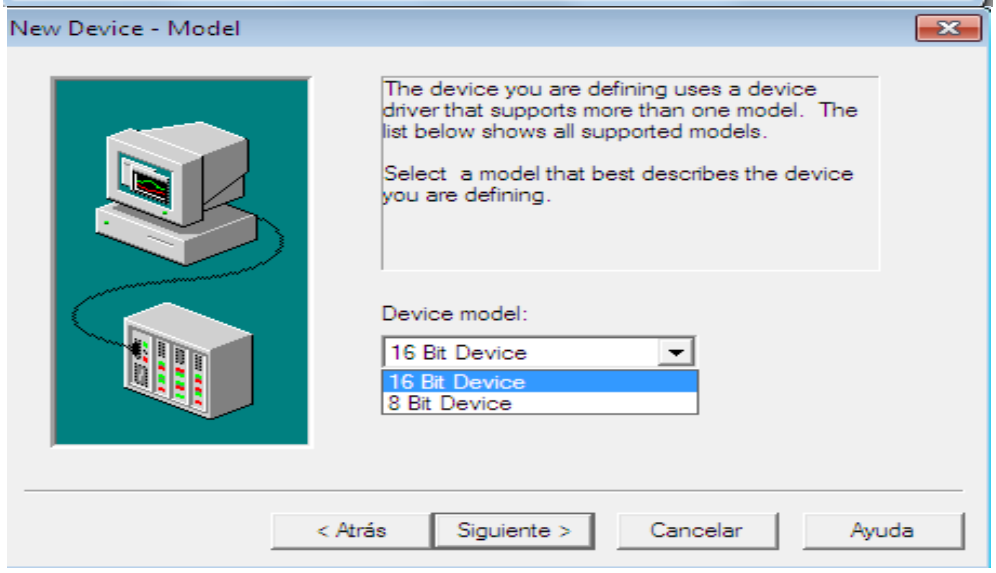
Fuente: propia

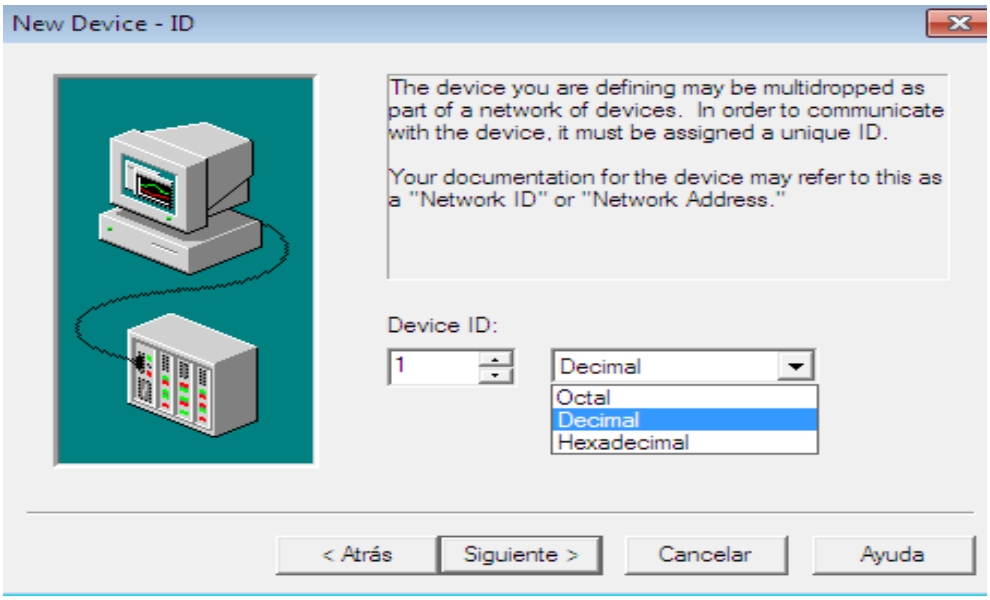
2.2. Crear un dispositivo de comunicación

Una vez se tenga el canal creado, se debe configurar un dispositivo (**device**), como se indica en la ilustración H-20, en la primera pantalla del asistente se asigna un nombre al dispositivo (en este caso MATLAB), al hacer clic en el botón <siguiente>, el asistente presenta los dispositivos existentes para el canal de comunicación configurado; dependiendo del tipo de variables se selecciona el modelo del dispositivo (de 8 o 16 bits), luego se asigna un identificador de red para cada dispositivo (en este caso “Device ID: 1”), y se finaliza el asistente.

Ilustración H-20: pasos para la creación de un dispositivo en KEPserverEX para la planta en Matlab Intercambiador de Calor


New Device - Name
A device name can be from 1 to 256 characters in length.
Names can not contain periods, double quotations or start with an underscore.
Device name:
MATLAB
< Atrás
Siguiente >
Cancelar
Ayuda


New Device - Model
The device you are defining uses a device driver that supports more than one model. The list below shows all supported models.
Select a model that best describes the device you are defining.
Device model:
16 Bit Device
16 Bit Device
8 Bit Device
< Atrás
Siguiente >
Cancelar
Ayuda

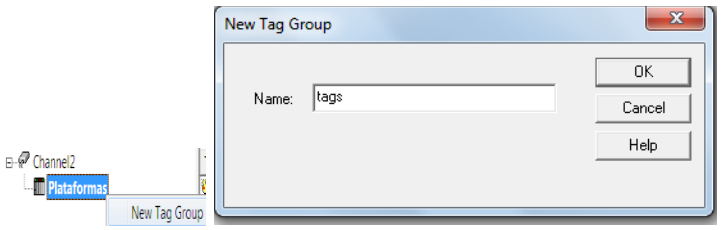


Fuente: propia

2.3. **Crear las tags asociadas al intercambiador de calor en Matlab**

Teniendo el canal y el dispositivo creado es necesario establecer un grupo de objetos ítems para cada plataforma de control, para fijar el nombre del grupo se hace clic derecho en el nombre del dispositivo y se establece el nombre (en este caso **tags**) como se muestra en la ilustración H-21.

Ilustración H-21: Creación de un grupo de objetos ítems para la planta en Matlab Intercambiador de Calor

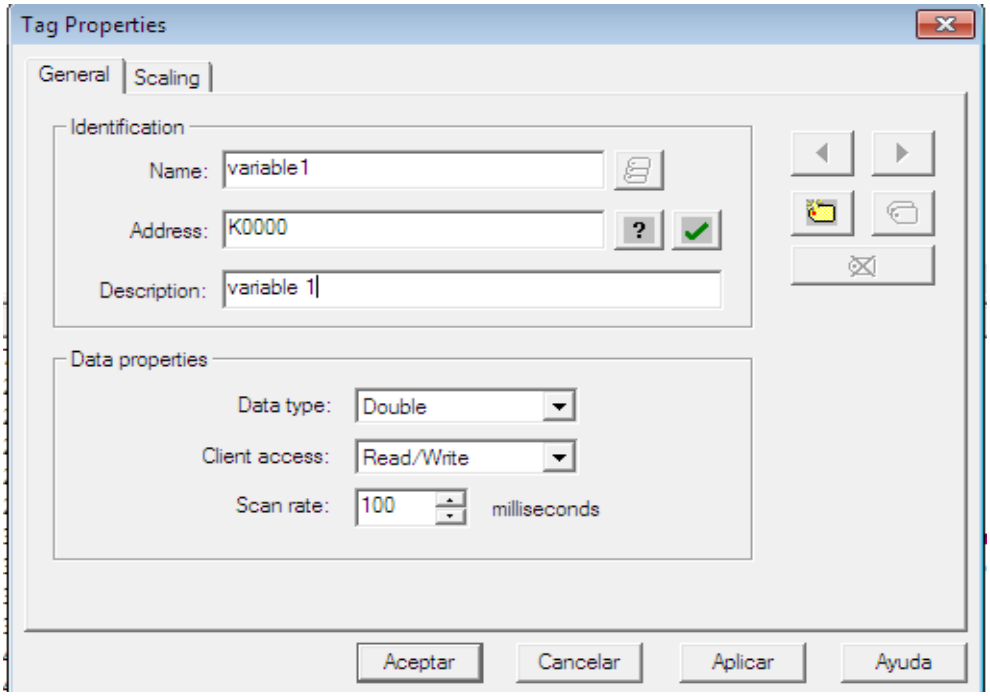


Fuente: propia

Para crear los ítems se hace clic en el grupo **tags**, donde aparece un formulario, ver ilustración H-22, que permite configurar las siguientes propiedades: nombre, dirección (presionar en ? para consultar el valor correspondiente de dirección), tipo de datos. Para finalizar se hace clic en el botón y se verifica si en el panel

"Event Log" (panel inferior) de la interfaz se ha desplegado algún error. La tabla H-3 presenta los nombres de las tags que se debe configurar para la comunicación entre el Servidor OPC y la planta simulada en Matlab.

Ilustración H-22: Formulario de configuración de objetos en KEPserverEX para la planta en Matlab Intercambiador de Calor



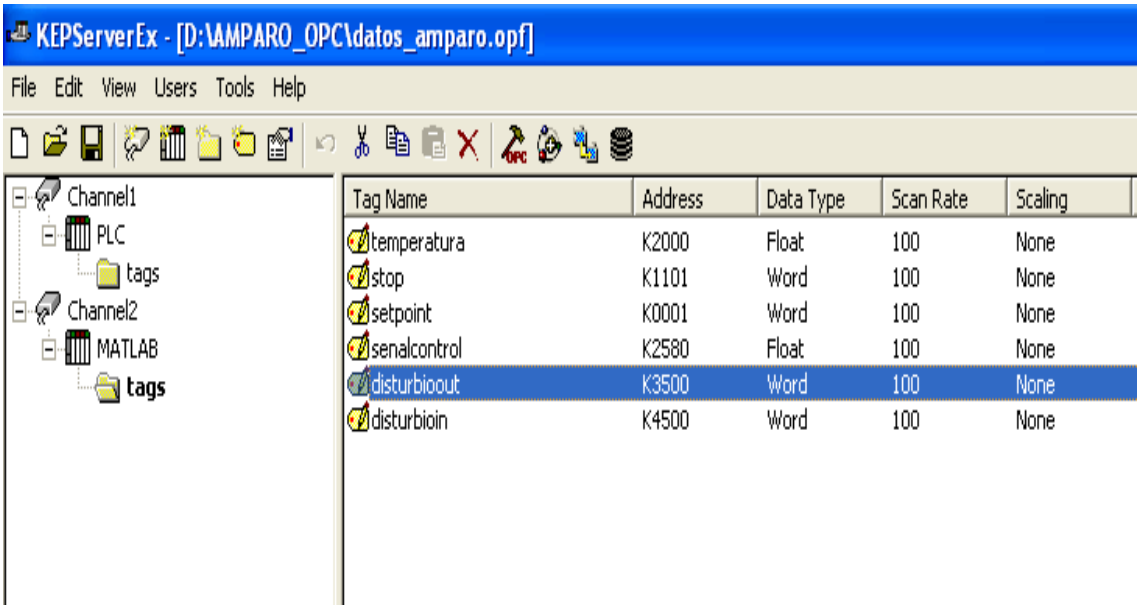
Fuente: propia

Tabla H -3: Clasificación de las variables (tags) de la planta Intercambiador de calor, simulada y controlada en Matlab, con su respectiva dirección de memoria.

SEÑALES	DIRECCION	TIPO DESEÑAL
Setpoint temperatura	K2000	Análoga
Temperatura de salida	K2580	Análoga
Esfuerzo de control	K1283	Análoga
Disturbio	K4024	Análoga
Disturbio salida	K3500	Análoga
Stop	K4512	Discreta

Si las tags estan bien configuradas se debe presentar una pantalla como la que se presenta en la ilustración H-23.

Ilustración H-23: Lista de ítems creados en KEPserverEX del intercambiador de calor simulado y controlado en Matlab.



Fuente: propia

3. Configuración de los objetos ítems en el servidor KepserverEX para el control de nivel de un tanque implementado en Rtai-Lab

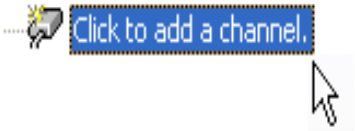
Iniciar la máquina virtual WinOPC en el computador de la planta SEDs, verificar que se encuentre instalado el servidor KepserverEX, en caso de no encontrarse instalado se deben seguir los pasos descritos en el numeral II del anexo B (manual de usuario del módulo servidor OPC DCOM KEPServerEx).

Ejecutar el servidor OPC KepserverEX instalado en la máquina virtual WinOPC y efectuar los siguientes pasos:

3.1. crear un canal de comunicación

Para crear un canal de comunicación se hace clic en el icono "*click to add channel"* en el panel izquierdo, ver ilustración H-24.

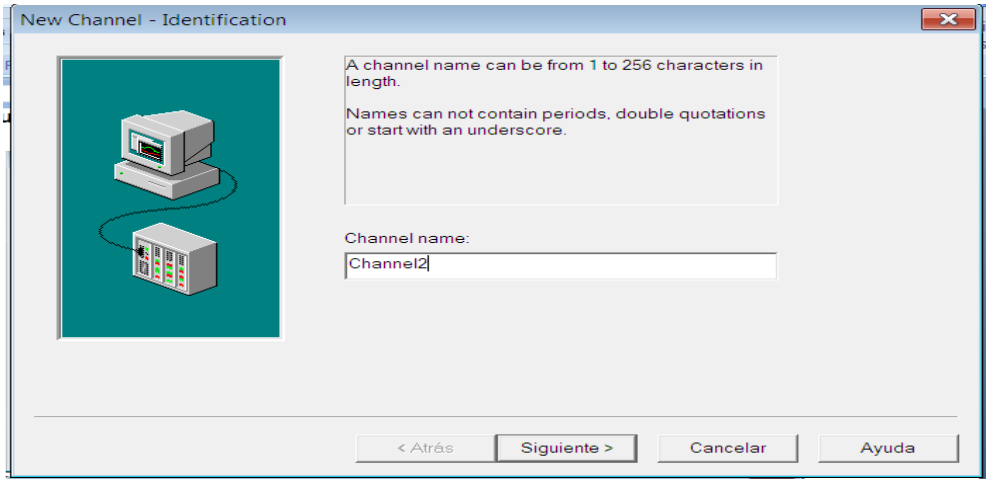
Ilustración H-24: Asistente de configuración de canal en servidor KEPserverEX para la planta de nivel

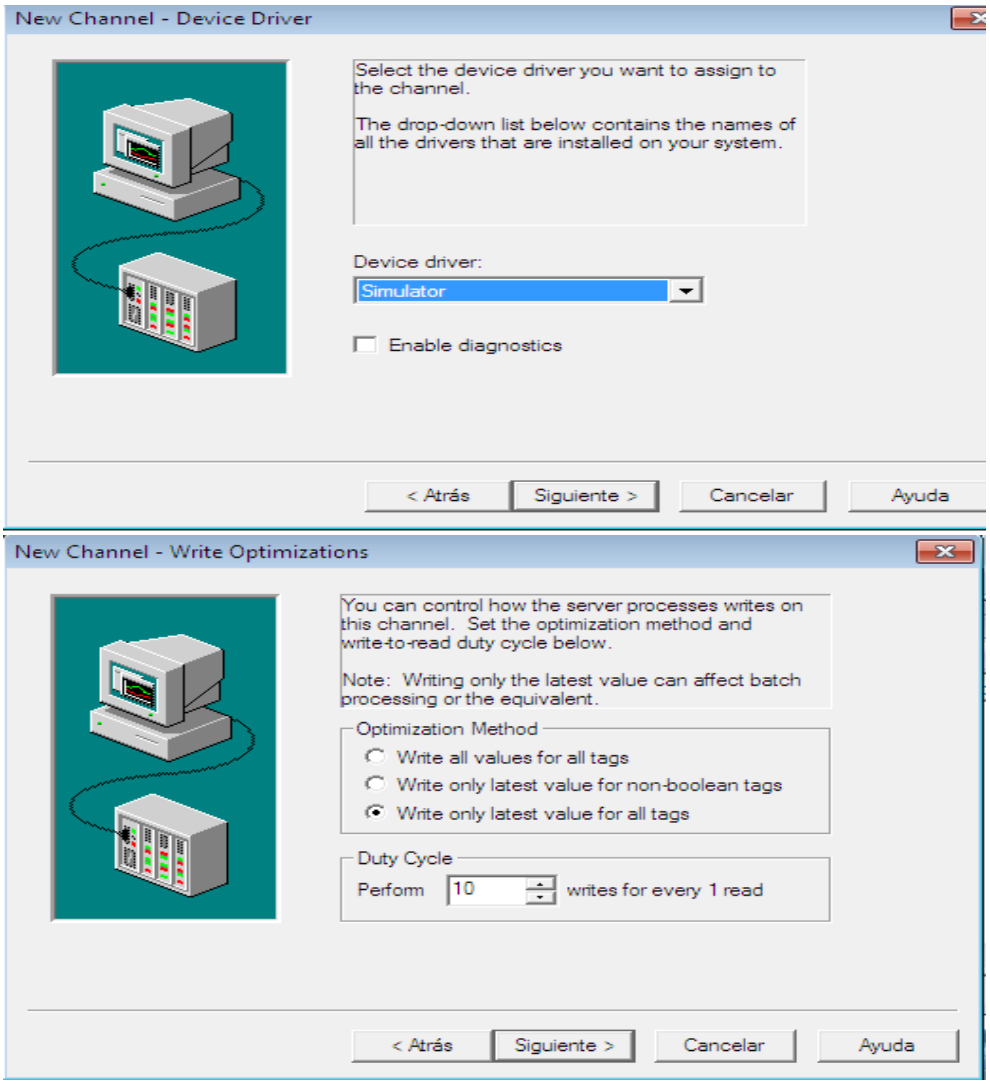


Fuente: propia

En este asistente se configuran todas las características del canal de comunicación, luego de darle un nombre al canal (Channel3) se da clic en el botón <siguiente> y se configura el driver que se va a usar, para este caso no se requiere driver por lo tanto se utiliza la opción "*simulator*", al dar clic en el botón <siguiente> se presenta una pantalla que permite la configuración de algunos parámetros y finalmente se finaliza el asistente, ver ilustración H-25.

Ilustración H-25: Pasos para la configuración de canal en el servidor KEPserverEX para la planta de nivel



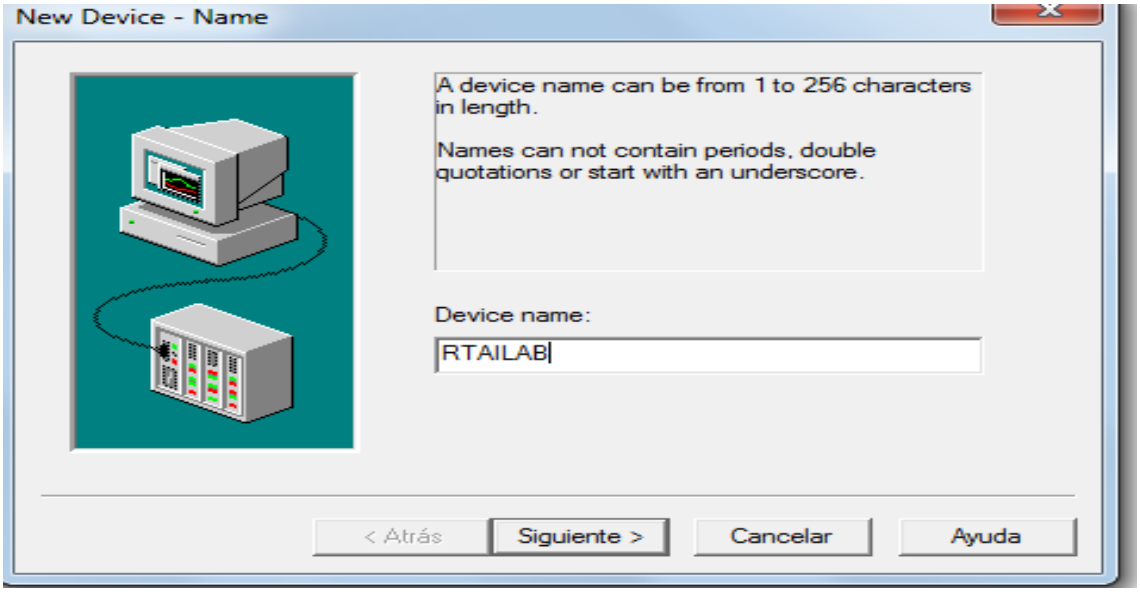


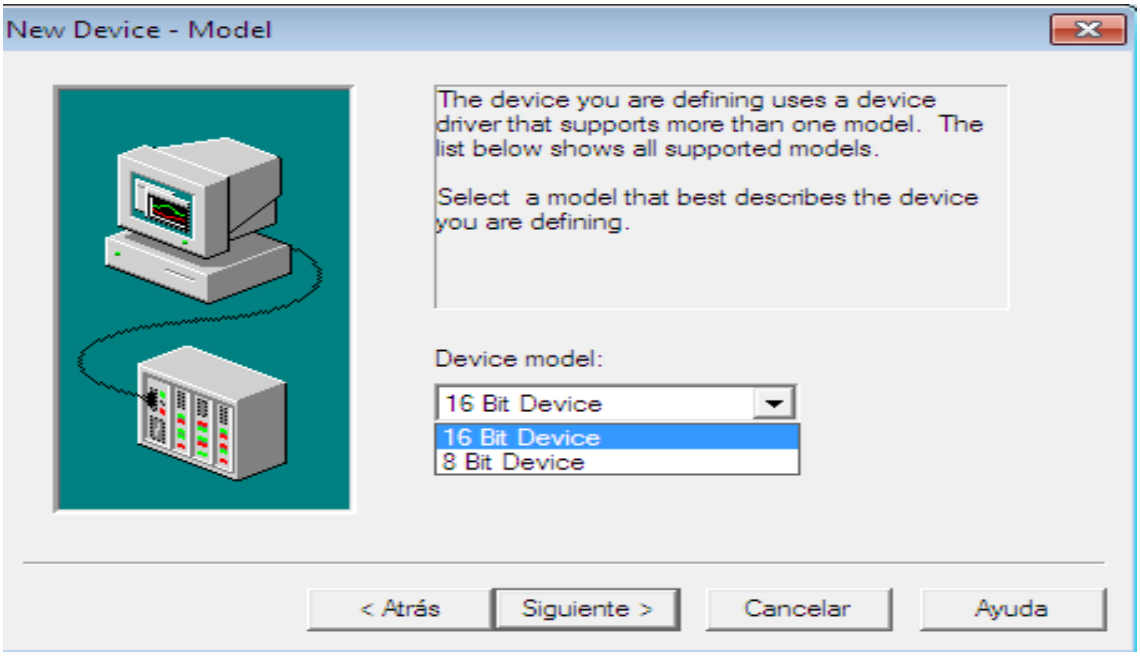
Fuente: propia

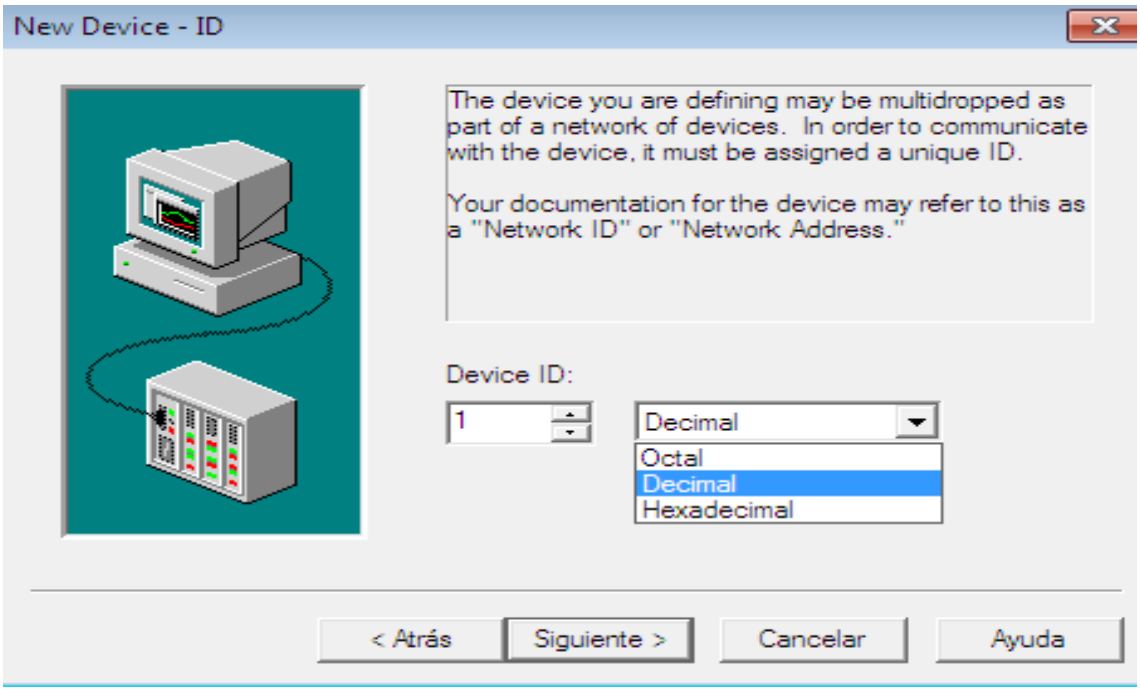
3.2. Crear un dispositivo de comunicación

Una vez se tenga el canal creado, se debe configurar un dispositivo (Rtai-Lab), como se indica en la ilustración H-26, en la primera pantalla del asistente se asigna un nombre al dispositivo (en este caso Rtai-Lab), al hacer clic en el botón <siguiente>, el asistente presenta los dispositivos existentes para el canal de comunicación configurado; se selecciona el modelo del dispositivo (16 bits), luego se asigna un identificador de red para cada dispositivo (en este caso “Device ID: 1”), y se finaliza el asistente.

Ilustración H-26: pasos para la creación de un dispositivo en KEPserverEX para la planta de nivel





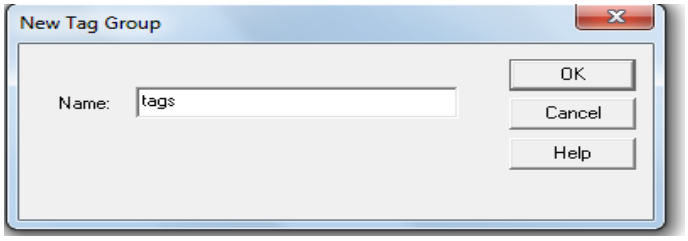


Fuente: propia

3.3. **Crear las tags asociadas a la planta de Nivel**

Teniendo el canal y el dispositivo creado es necesario establecer un grupo de objetos ítems para cada plataforma de control, para fijar el nombre del grupo se hace clic derecho en el nombre del dispositivo y se establece el nombre como se muestra en la ilustración H-27.

Ilustración H-27: Creación de un grupo de objetos ítems en el dispositivo para la planta de nivel

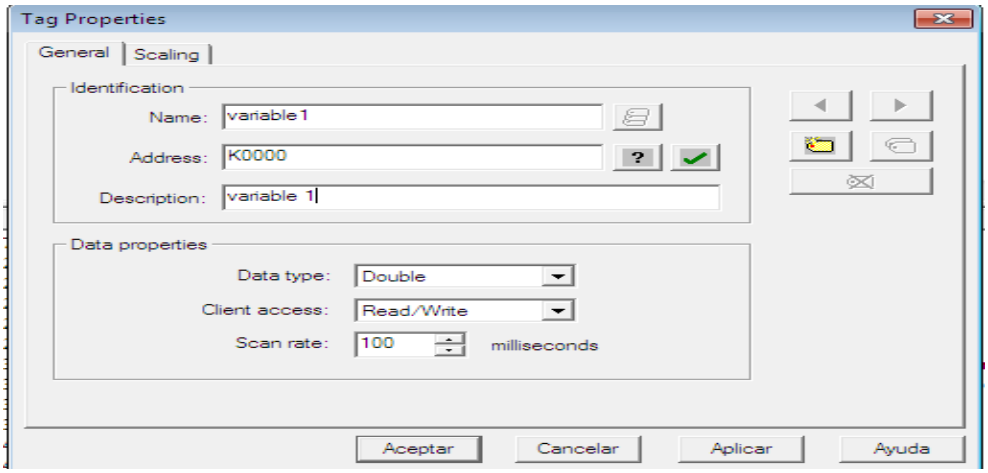


Fuente: propia

Para crear los ítems se hace clic en el grupo tags, donde aparece un formulario, ver ilustración H-28, que permite configurar las siguientes propiedades: nombre,

dirección (presionar en ? para consultar el valor correspondiente de dirección según el tipo de datos), tipo de datos. Para finalizar se hace clic en el botón ✓ y se verifica si en el panel "Event Log" (panel inferior) de la interfaz se ha desplegado algún error. La tabla H-4 presenta los nombres y la dirección de las tags que se debe configurar para la comunicación entre el Servidor OPC y la planta de Nivel controlada en Rtai-Lab. En el caso de que se necesite agregar más variables, se debe hacer clic en el icono ? donde explica como asignar la dirección.

Ilustración H-28: Formulario de configuración de objetos en KEPserverEX para la planta de nivel



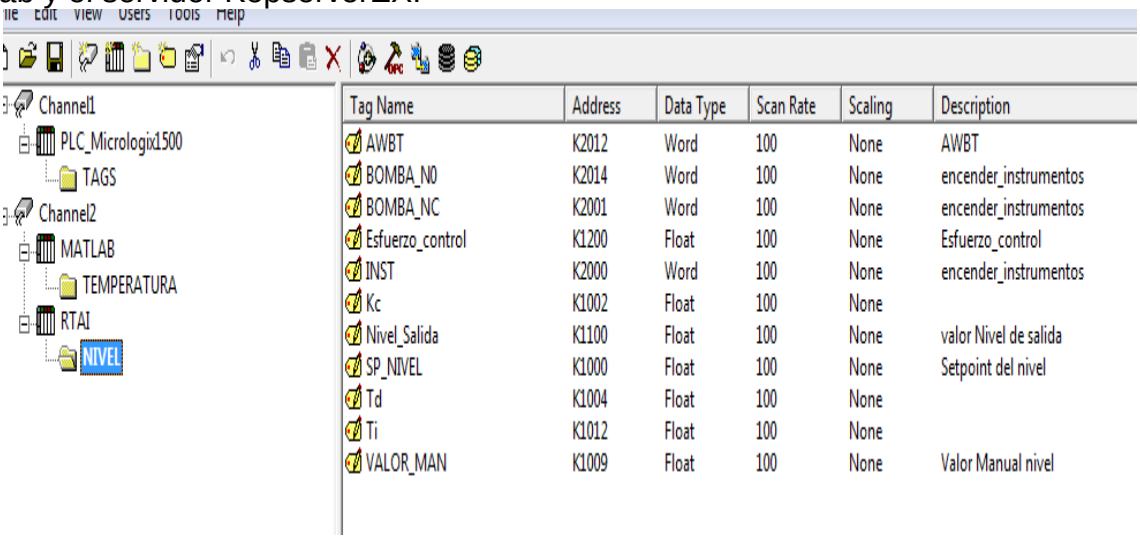
Fuente: propia

Tabla H -4: Clasificación de las variables (tags) para la planta de nivel con su respectiva dirección de memoria.

SEÑALES	DIRECCION	TIPO DESEÑAL
Caudal de Entrada	K1100	Análoga
Caudal de salida	K2580	Análoga
Nivel	K1283	Análoga
Disturbio	K4024	Discreta
Manual/Automático	K4512	Discreta
Setpoint Nivel	K4513	Análoga
Instrumentacion	K4514	Discreta
BombaOn	K4515	Discreta
BombaOff	K4516	Discreta
Kc	K4517	Análoga
Td	K4518	Análoga
Ti	K4519	Análoga
AWBT		

Si las tags estan bien configuradas se debe presentar una pantalla similar la que se presenta en la ilustracion H-29.

Ilustración H-29: Configuración de objetos ítems para la comunicación entre Rtai-Lab y el servidor KepserverEX.



Tag Name	Address	Data Type	Scan Rate	Scaling	Description
AWBT	K2012	Word	100	None	AWBT
BOMBA_N0	K2014	Word	100	None	encender_instrumentos
BOMBA_NC	K2001	Word	100	None	encender_instrumentos
Esfuerzo_control	K1200	Float	100	None	Esfuerzo_control
INST	K2000	Word	100	None	encender_instrumentos
Kc	K1002	Float	100	None	
Nivel_Salida	K1100	Float	100	None	valor Nivel de salida
SP_NIVEL	K1000	Float	100	None	Setpoint del nivel
Td	K1004	Float	100	None	
Ti	K1012	Float	100	None	
VALOR_MAN	K1009	Float	100	None	Valor Manual nivel

Fuente: **Propia**

Nota. Algunas de las variables se pueden fijar por defecto en Scicos, por tal razón no es necesario configurarlas en el servidor OPC.

VI. CONFIGURACIÓN DE LOS CLIENTES OPC DCOM EN LAS PLATAFORMAS DE CONTROL

Para poder configurar los clientes OPC DCOM en Matlab y Rtai-Lab, se debe verificar la dirección **IP** asociada a cada computador donde están instalados los servidores OPC DCOM, en este caso en el computador de la planta de presión y en la máquina virtual **WinOPC** instalada en el computador de la planta de SED. En la tabla H-5 colocar la dirección IP asociada a cada computador.

Tabla H -5: Dirección IP de los equipos donde están instalados los servidores OPC.

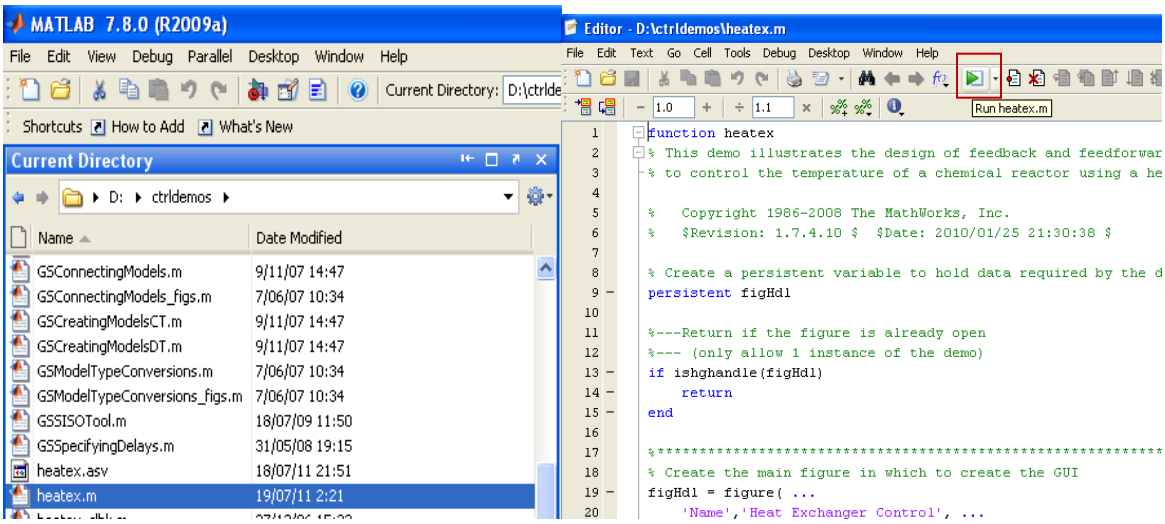
IP PC PLANTA DE PRESION	IP MAQUINA VIRTUAL PC PLANTA SEDS

Fuente: **Propia**

1. Configuración del Módulo Cliente OPC DCOM en Matlab

Ubicarse en el computador de la planta de presión, abrir Matlab, ubicarse en el directorio **D/ctroldemo** y abrir/ejecutar el archivo **heatex.m,** ver ilustración H-30**.**

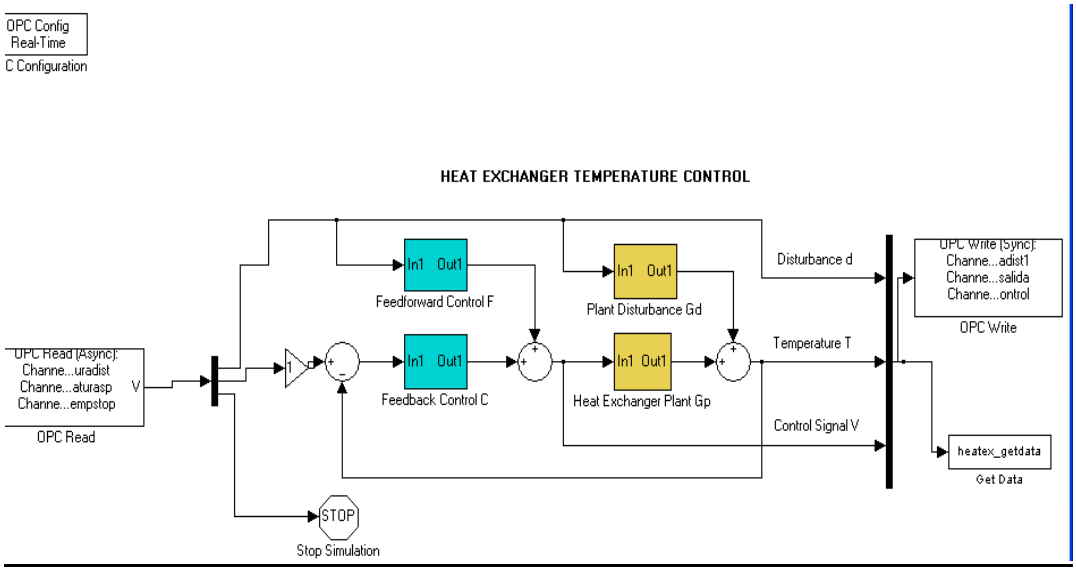
Ilustración H-30: Pasos para ingresar al control del intercambiador de calor en Matlab.



Fuente: **Propia**

Debe aparecer un diagrama en Simulink, heatex_sim.mdl, como el que se presenta en la ilustración H-31.

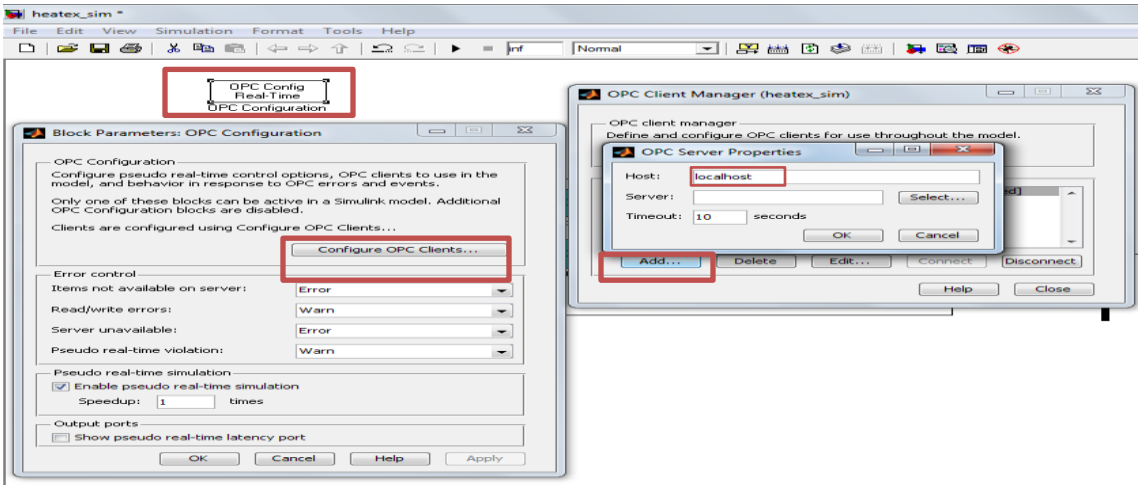
Ilustración H-31: Diagrama de bloques, archivo heatex_sim.mdl, en Simulink que permite el control de temperatura del intercambiador de calor y envía los datos al servidor OPC DCOM.



Fuente: **Propia**

En el diagrama heatex_sim.mdl en Simulink, el primer paso es configurar los parametros del servidor OPC DCOM, para ello se hace clic derecho en el bloque opc *configuration (OPC Conf Real Time),* posteriormente hacer clic en el boton *configure OPC Clients* donde aparece un formulario que permite ingresar la direccion ip, del computador donde se encuentra instalado el servidor con las tags de MATLAB (en este caso la planta de presión) y seleccionar el servidor a utilizar, en este caso el KepServerEX, ver ilustracion H-32.

Ilustración H-32: Pasos para la configuración del cliente OPC en Simulink para el intercambiador de calor.

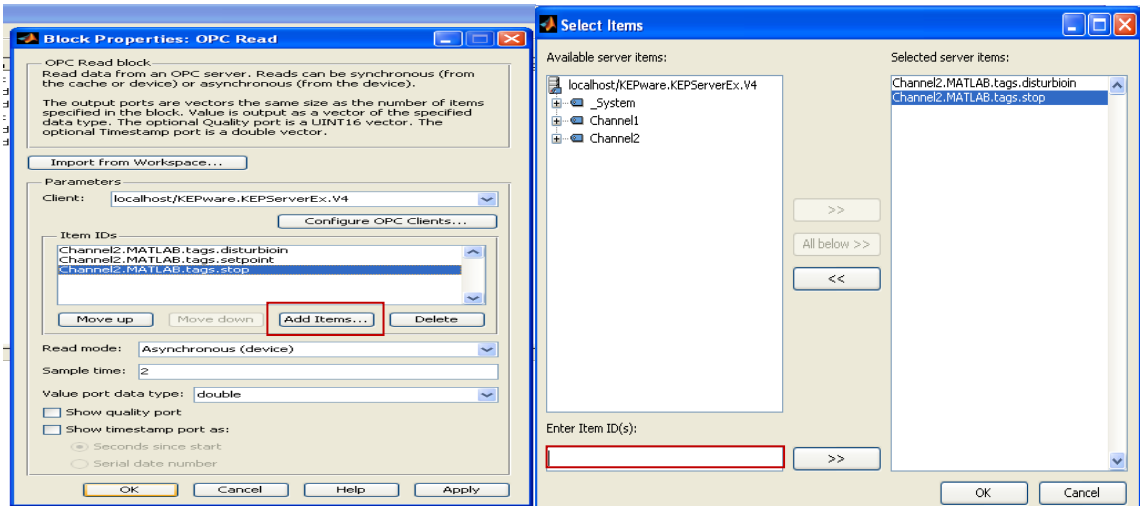


Fuente: **Propia**

Configurar las tags para el bloque de lectura desde el Servidor OPC DCOM

En el diagrama heatex_sim.mdl en Simulink, el segundo paso es hacer clic derecho en el bloque OPC Read, ver ilustración H-31, hacer clic en el botón **add Ítems**, en la ventana desplegada adicionar la dirección completa (ver recuadro rojo) donde esta cada una de las tags (setpoint temperatura, disturbio, stop) creadas en el servidor KepeserverEX, ver ilustración H-33.

Ilustración H-33: Pasos para la configuración del Bloque OPC Read en simulink

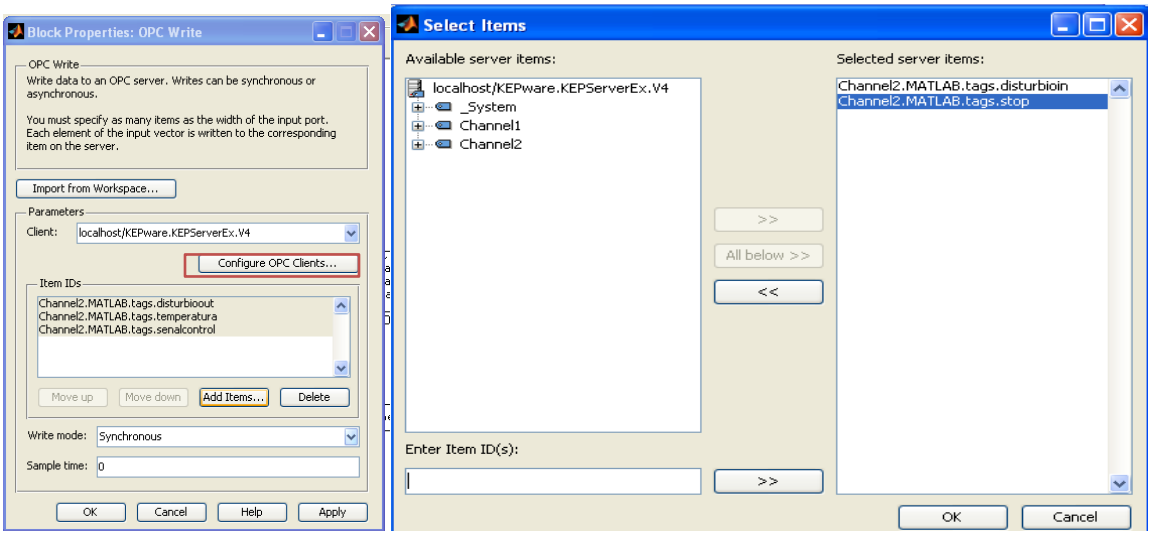


Fuente: **Propia**

Configurar las tags para el bloque de escritura desde el Servidor OPC DCOM

En el diagrama heatex_sim.mdl en Simulink, el tercer paso es hacer clic derecho en el bloque OPC Write, ver ilustración H-31, hacer clic en el botón **add Items** y en la ventana desplegada digitar la dirección completa donde esta cada una de las tags (temperatura, esfuerzo de control, disturbio) creadas en el servidor KepeserverEX, ver ilustración H-34.

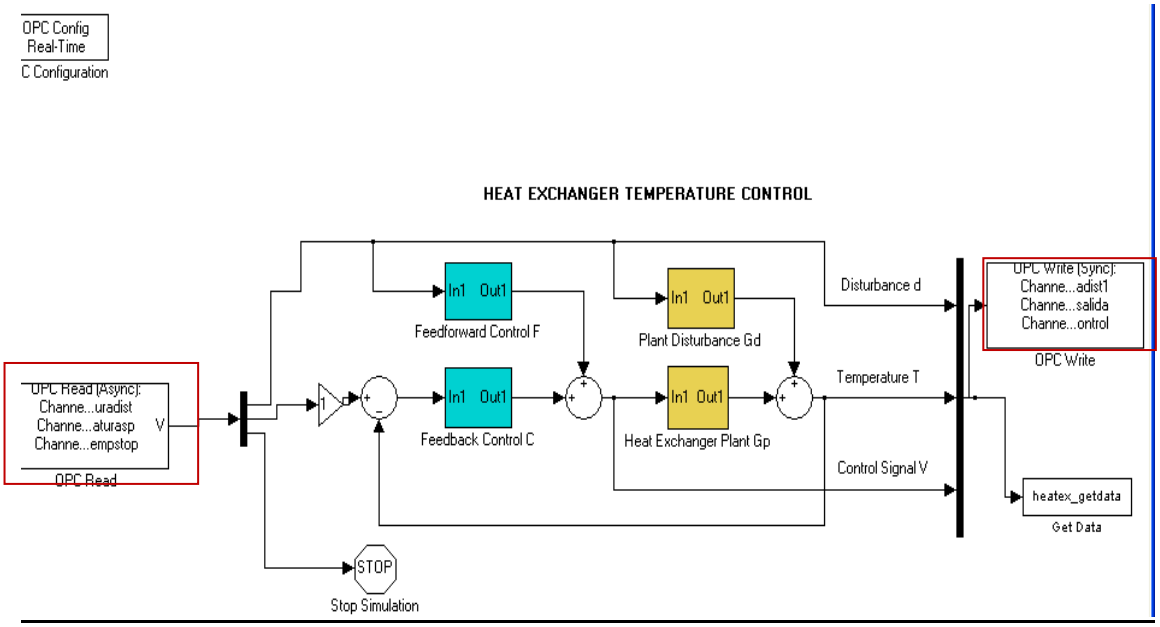
Ilustración H-34: Pasos para la configuración del Bloque OPC Write en Simulink



Fuente: **Propia**

Finalmente se debe verificar el orden de la demultiplexación y multiplexación de las tags en cada bloque, respectivamente, ver ilustración H-35. El orden de configuración para el bloque de lectura es “disturbio, setpoint, stop”; y para el bloque OPC Write es “disturbio salida, temperatura de salida, señal de control”.

Ilustración H-35: Diagrama de control de temperatura incluido el cliente OPC DCOM.

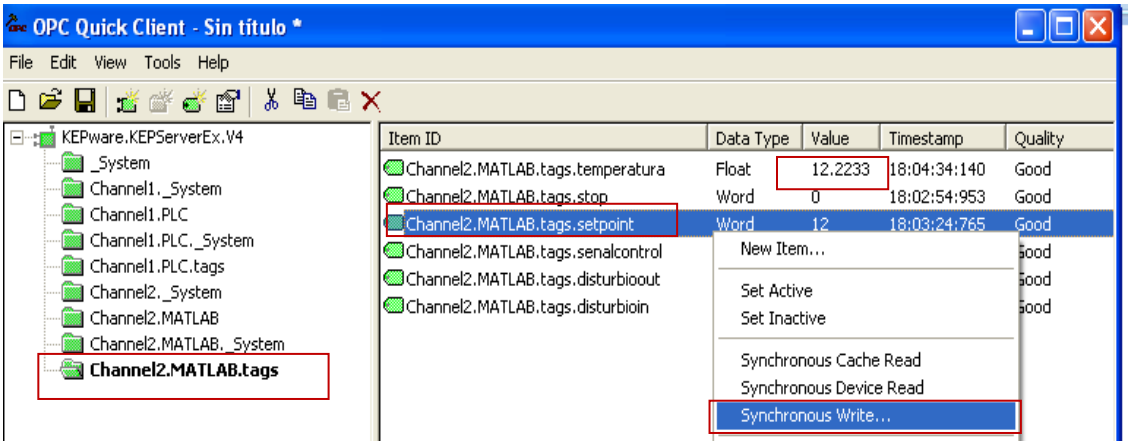


Fuente: **Propia**

Verificación de la comunicación entre Matlab y el servidor KepeserverEX

Finalmente se debe realizar un chequeo para saber si la comunicación entre Matlab y el Servidor OPC DCOM KepserverEX se ha realizado correctamente. Para ello en el computador de la planta de presión, se hace clic en el icono del servidor KepserverEX, seleccionar la **ruta channel2.MATLAB.tags,** hacer clic derecho en el objeto ítem channel2.MATLAB.tags.Setpoint, opción Syncronous Write, y cambiar el valor de la variable **setpoint**, verificar el cambio en la variable temperatura en la misma pantalla, ver ilustración H-36.

Ilustración H-36: Pasos para verificar la comunicación entre el servidor OPC DCOM y el intercambiador de calor en Matlab.



Fuente: **Propia**

2. Configuración del Módulo cliente OPC DCOM en Rtai-Lab

Después de establecida la comunicación entre el servidor OPC DCOM y Matlab para la planta del intercambiador de calor, en esta sección 2 se procede a configurar los bloques OPC en Scicos para la comunicación entre Rtai-lab, para la planta de nivel, y el servidor OPC KepserverEX instalado en la máquina virtual que corre en el computador de SEDS.

Para ello se debe ubicar en el computador de la planta de nivel, encenderlo, ingresar por Fedora Core 2.6.23-rtai, y ejecutar los siguientes pasos: cargar módulos de Rtai, verificar conexión con el servidor, configurar los bloques de lectura/escritura, generar código y ejecutar la tarea; como se describe a continuación.

2.1. Cargar módulos

Para implementar y ejecutar tareas en Rtai-Lab se deben cargar los módulos que contienen las librerías necesarias de Rtai, efectuando los siguientes pasos:

- Abrir la ventana “Terminal” en la barra de herramientas que aparece en la parte inferior del escritorio, ver ilustración H-37, y digitar las dos siguientes instrucciones:

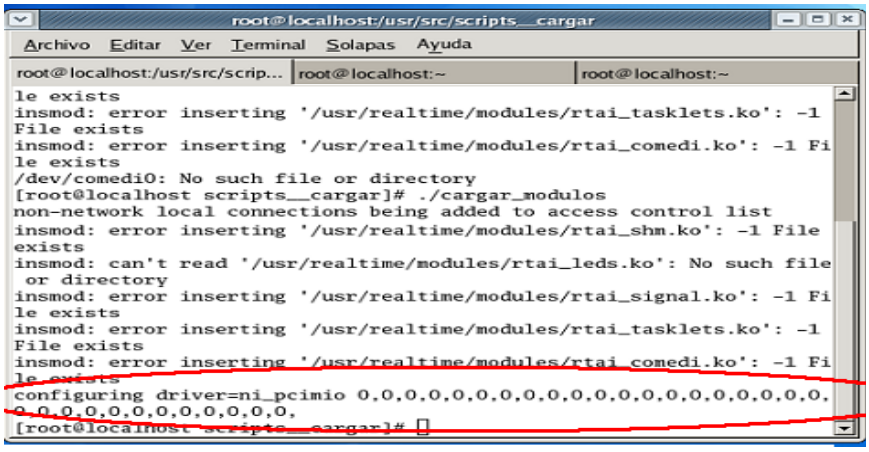
cd /usr/src/scripts__cargar/ (enter)
./cargar_módulos (enter)

Ilustración H-37: Apertura de ventana terminal de comandos.

Fuente: propia

Esta última instrucción se debe repetir hasta que aparezca en pantalla una fila de ceros 0,0,0…..,0,0, que confirma que los módulos se han cargado completamente, ver ilustración H-38.

Ilustración H-38: Ventana que indica que los módulos han sido cargados.



Fuente: propia

2.2. Verificar funcionamiento de red y de servidor OPC

Retomar la IP de la máquina virtual XpOPC instalada en el computador de la planta de SED (ver tabla H-5).

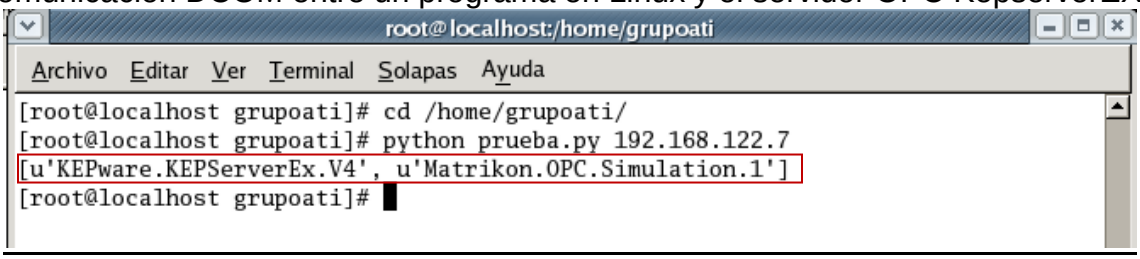
Para verificar la conexión OPC DCOM desde Linux con el servidor OPC DCOM instalado en la máquina virtual del computador SED, se debe de abrir un terminal y digitar las siguientes instrucciones, ver la ilustración H-39:

cd /home/grupoati (enter)

python prueba.py [escribir IP planta SEDS] (enter)

Al presionar enter, debe aparecer una lista con todos los servidores OPC DCOM disponibles, que indica que la comunicación OPC se ha efectuado correctamente.

Ilustración H-39: Terminal con las instrucciones para efectuar la prueba de comunicación DCOM entre un programa en Linux y el servidor OPC KepserverEX.



Fuente: propia

Si la configuración no es exitosa, se debe:

- Verificar que el computador de la planta de nivel esté conectado a la red, o que la red este activa.
- Ejecutar un ping hacia la máquina virtual instalada en el computador de SEDS.
- Verificar que el servidor OPC DCOM KepserverEX este inicializado.

Reconfigurar variables en Scicos

Una vez la prueba de comunicación OPC DCOM en Linux es satisfactoria, se debe iniciar con la configuración de los bloques OPC DCOM en Scicos. Para ello se debe configurar los objetos ítems en Scicos realizando los siguientes pasos:

Abrir un terminal, y digitar las siguientes instrucciones, ver ilustración H-40:

cd home/grupoati (enter)

su grupoati (enter)

scilab (enter)

Debe aparecer una ventana de Scilab, ver ilustración H-41, donde se digita **scicos,** y se debe abrir el archivo **pidserieopc.cos,** ver ilustración H-42.

Ilustración H-40: Pantalla que indica como ingresar a Scilab desde un terminal

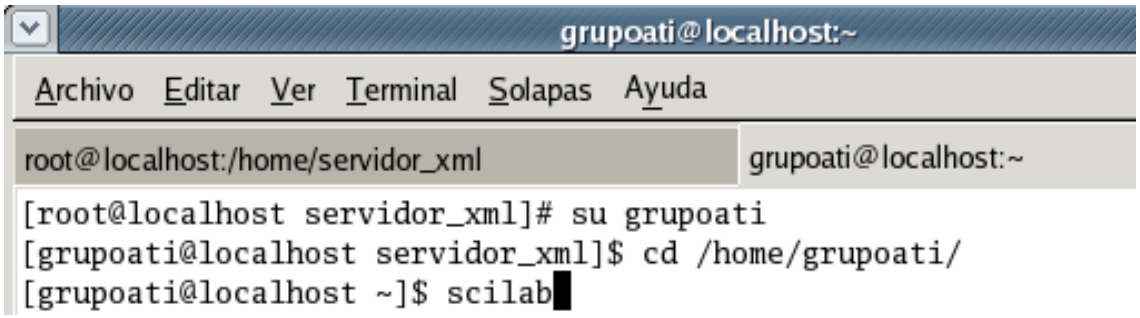
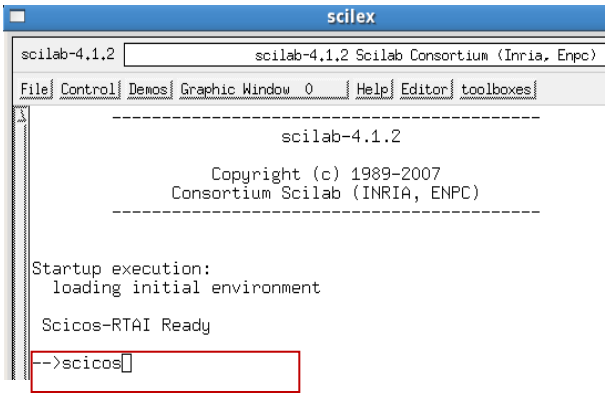
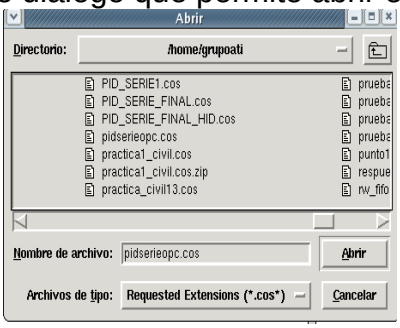


Ilustración H-41: Pantalla que indica como ingresar a Scicos.



Fuente: propia

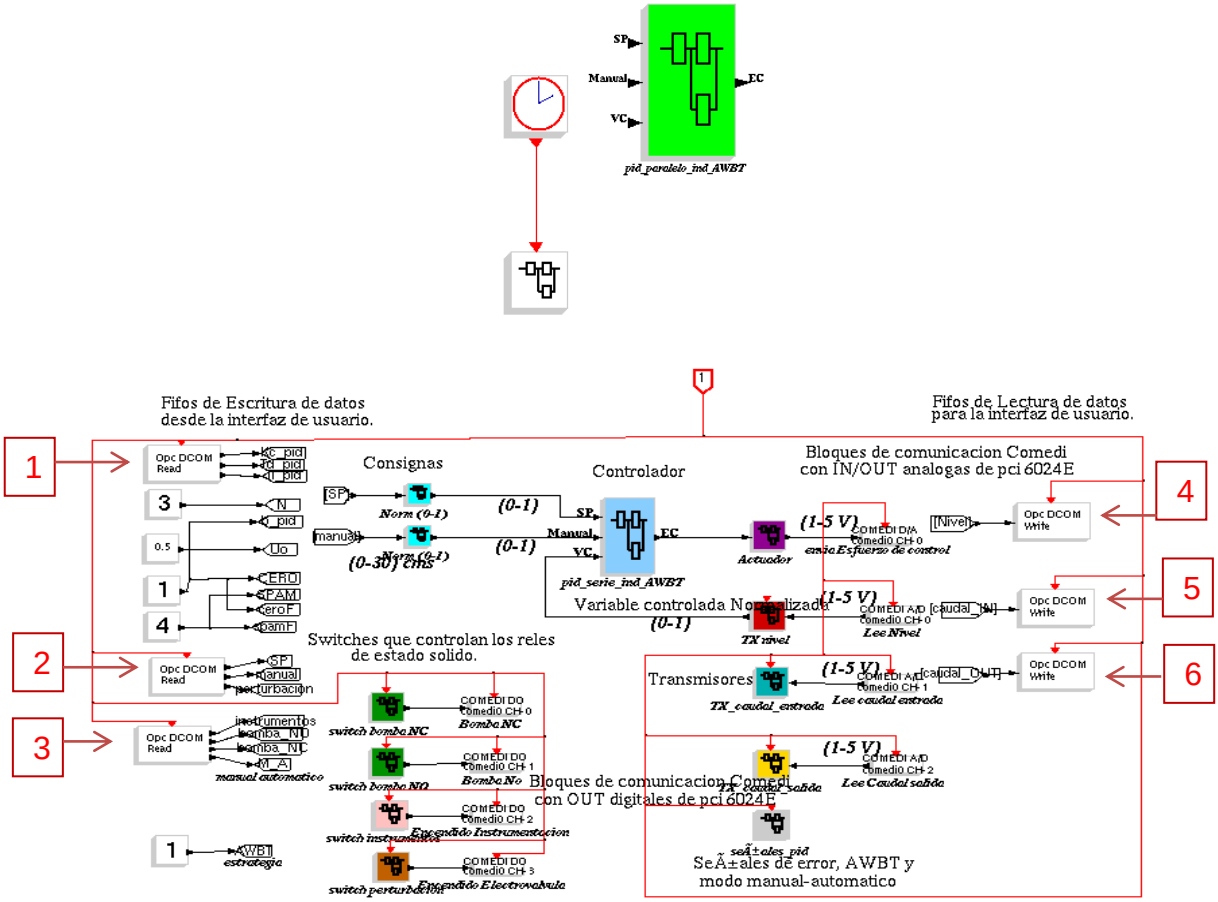
Ilustración H-42: cuadro de dialogo que permite abrir el archivo pidserieopc.cos.



Fuente: propia

Al abrir el archivo se presenta un superbloque, al dar clic en ese superbloque se presenta el diagrama que permite controlar la planta de nivel, y enviar los valores de las variables de interés al servidor OPC, ver ilustración H-43.

Ilustración H-43: cuadro de dialogo que permite abrir el archivo pidserieopc.cos.



Fuente: propia

El diagrama de la ilustración H-43 está conformado por tres bloques de lectura (ver recuadros 1, 2 y 3), que se encargan de capturar la información proveniente del servidor OPC DCOM instalado en la máquina virtual que se ejecuta en el computador de la planta de SEDS, además presenta tres bloques de escritura, (ver recuadros 4, 5 y 6), que se encargan de actualizar en el servidor OPC, los valores de las variables medidas en la planta de nivel (nivel, caudal de entrada y caudal de salida).

Para configurar los parámetros de cada uno de los bloques descritos anteriormente, se debe llenar la tabla H-6, con la dirección IP donde está el servidor, el nombre del servidor (en este caso KepserverEX), y la dirección completa de cada tag, tal como se hizo en el punto 3 de la sección V de este anexo.

Tabla H -6: Tabla para ingresar los parámetros configuración para el cliente OPC DCOM en Rtai_Lab.

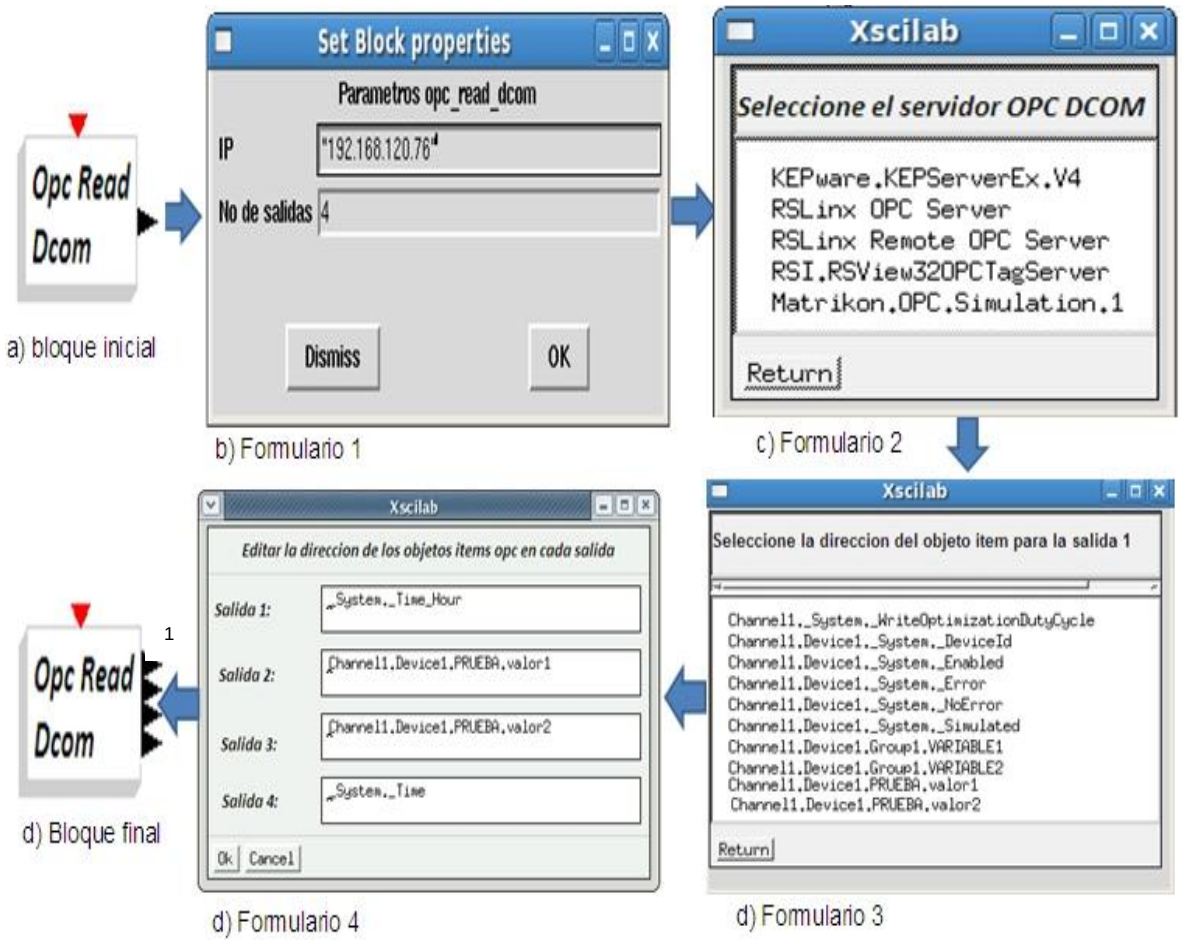
BLOQUE	IP	SERVIDOR OPC DCOM	TAGS ASOCIADAS	
			SEÑAL SCICOS Nombre en el diagrama de bloques en scicos de la variable	**TAGS SERVIDOR** Ruta de cada ítem en el servidor, ejemplo: channel1.rtai.tags.kc
OPC Read recuadro 1			*Kc_pid*	
			Td_pid	
			Ti_pid	
OPC Read recuadro 2			*Sp* (Setpoint nivel)	
			Manual (valor nivel manual)	
			Perturbación	
OPC Read recuadro 3			*Instrumentos*	
			Bomba N_0	
			Bomba N_C	
			M_A (Manual o automático)	
OPC Write recuadro 4			*Nivel*	
OPC Write recuadro 5			*Caudal_in* (Caudal de entrada)	
OPC Write recuadro 6			*Caudal_out* (Caudal de salida)	

Después de registrar y verificar nombres de las tags según la tabla H-6, se debe empezar a configurar los bloques OPC DCOM en Scicos efectuando los siguientes pasos en cada uno de los seis bloques OPC:

- Hacer clic derecho en el bloque OPC DCOM, aparece un formulario, ver ilustración H-44a, que permite definir la IP del equipo donde está instalado el servidor.
- En el formulario colocar la IP según la columna 2 de la tabla H-6 y hacer clic en OK para que se presente otro formulario que permite seleccionar el nombre del servidor, ver ilustración H-44b.
- Seleccionar el nombre del servidor según la columna 3 de la tabla H-6, y hacer clic en OK, para que se presenten los formularios que permiten seleccionar la dirección de cada tag, ver ilustración H-45c.

- Según la columna 5 de la tabla H-6 seleccionar en cada formulario la dirección de cada tag, hacer clic en OK hasta que se presente un último formulario, ver ilustración H-44d, que permite verificar las tags configuradas para este bloque y finalmente presionar Ok.

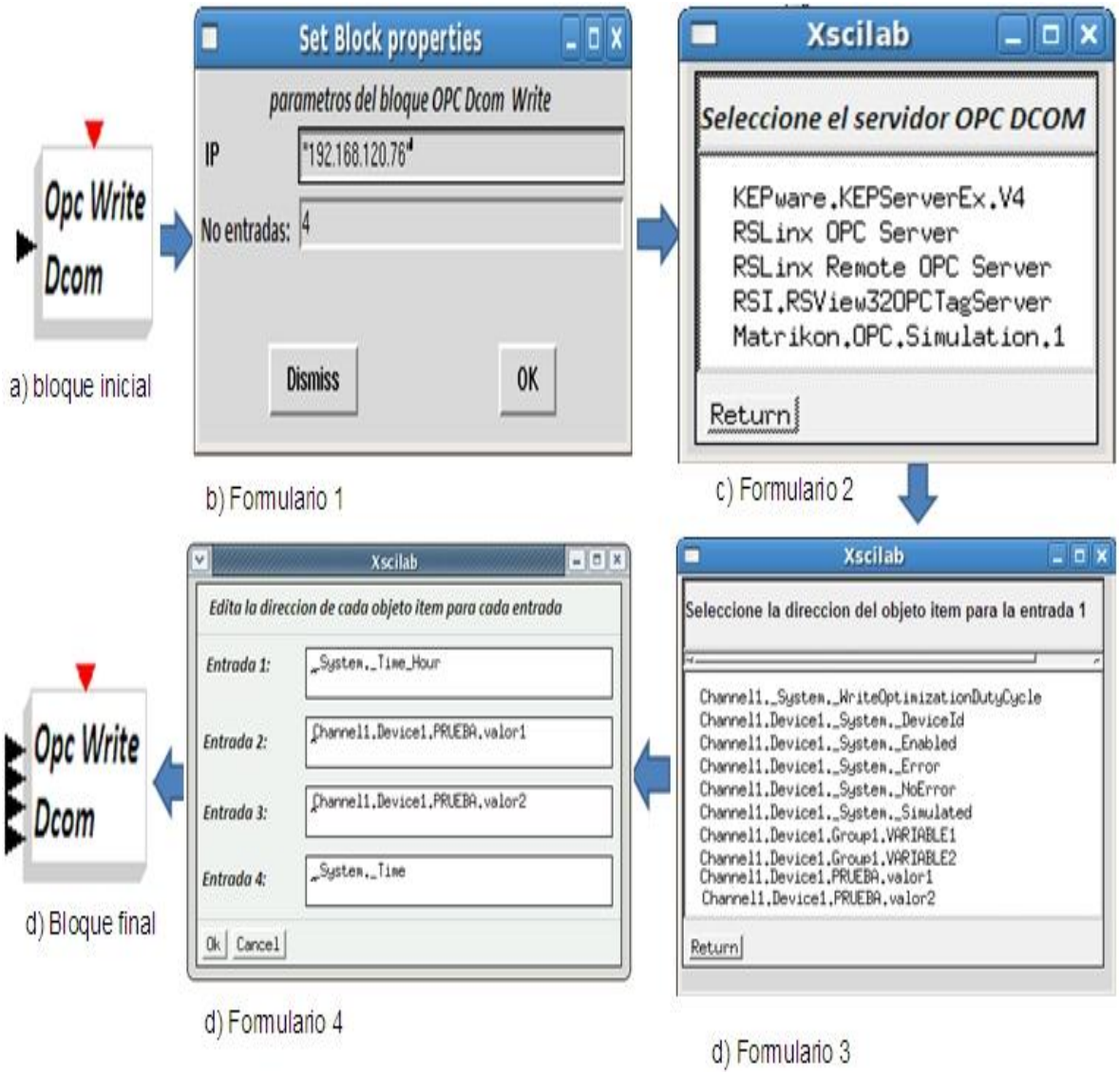
Ilustración H-44: Pasos para configurar los parámetros de un bloque Dcom_Read.



Fuente: Propia.

La ilustración H-45 presenta los pasos necesarios para la configuración de los parámetros del bloque Dcom_Write.

Ilustración H-45: Pasos para configurar los parámetros de un bloque Dcom_Write.



Fuente: Propia.

2.3. Generar el código en Rtai-Lab

Una vez configurados los bloques OPC DCOM del archivo pidserieopc.cos se debe guardar, realizar su compilación y generar la tarea de tiempo real asociada al diagrama.

Se hace *clic* en el menú *RTAI → RTAI Set Target, se debe* seleccionar el súper bloque generado y aparece la ventana mostrada en la ilustración H-46, en donde el usuario puede modificar las propiedades del ejecutable, en este caso se llama **pruebaopc** al ejecutable. Si el código se ha generado correctamente debe aparecer una pantalla como la que se presenta en la ilustración H-47.

Ilustración H-46: opciones para generación de código tarea de tiempo real para el superbloque pidserieopc

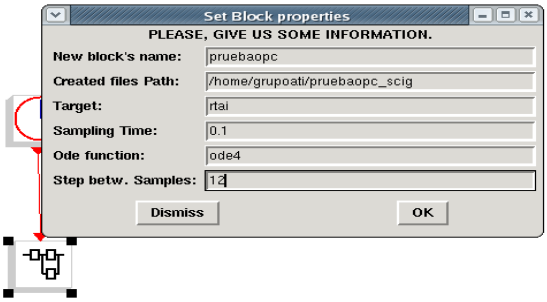
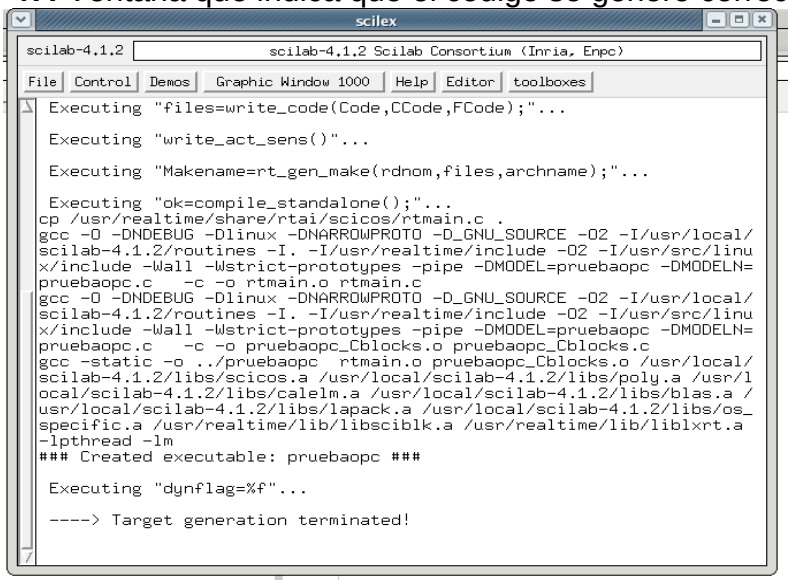


Diagrama para crear la tarea de tiempor real para el PID serie.

Ilustración H-47: Ventana que indica que el código se generó correctamente.



Fuente: Propia.

2.4. Ejecutar la tarea

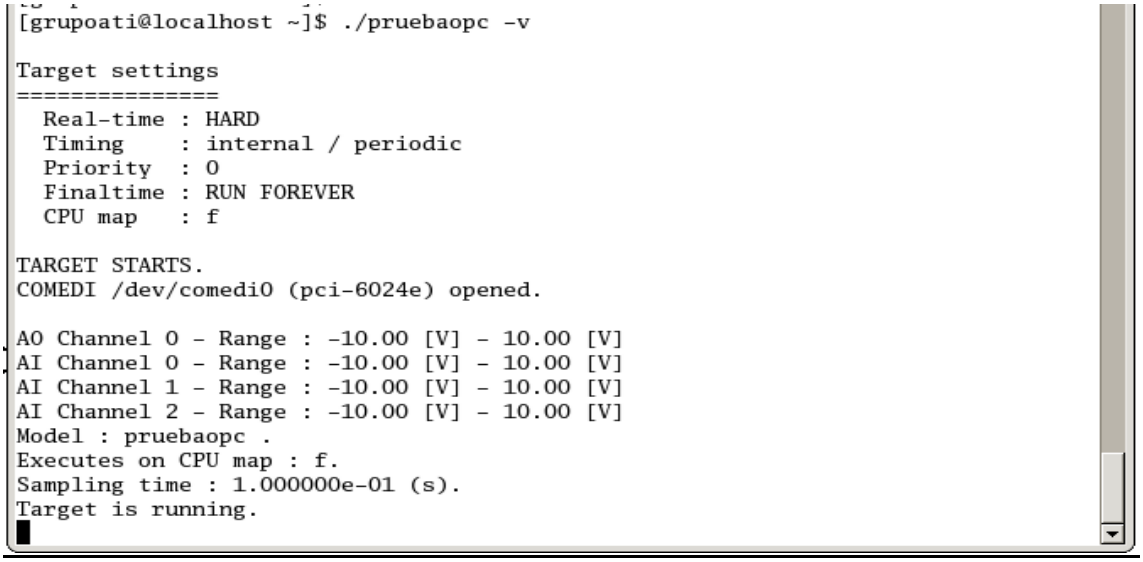
Antes de ejecutar la tarea se debe configurar algunos de los parámetros de las variables en el servidor KepserverEX instalado en la máquina virtual que corre en

el computador de la planta de SEDS, ingresar los parámetros Kp, Ti, Td, y verificar que la instrumentación y la bomba estén apagadas.(valores en cero).

Para iniciar la tarea de Rtai-Lab se debe abrir un nuevo terminal y digitar las siguientes instrucciones (ver ilustración H-48):

cd /home/grupoati
./pruebaopc –v

Ilustración H-48: Tarea del control de Nivel ejecutándose.



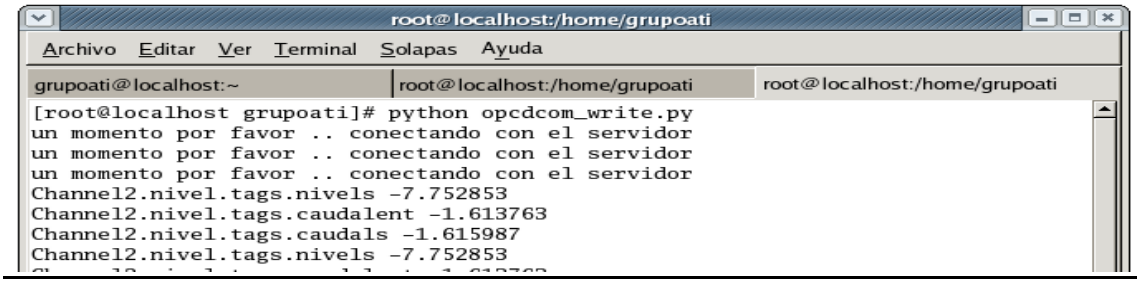
Fuente: propia

Para leer y escribir en el servidor OPC DCOM se debe hacer clic derecho en el terminal, abrir una nueva solapa, repetir este procedimiento dos veces, y en la primera solapa digitar **python opcdcom_read.py,** ver ilustración H-49. En la segunda solapa digitar **python opcdcom_write.py**, ver ilustración H-50.

Ilustración H-49: Instrucciones para iniciar la lectura en el módulo cliente OPC DCOM en Rtai-Lab.

Fuente: propia

Ilustración H-50: Instrucciones para iniciar la Escritura en el módulo cliente OPC DCOM en Rtai-Lab.



Fuente: propia

- En el cliente OPC del servidor KepserverEX, fijar los parámetros del controlador y el setpoint.

- En el cliente OPC del servidor KepserverEX, poner en uno la variable instrumentación (verificar que el transmisor, motobomba y demás instrumentos se encuentren encendidos).

- En el cliente OPC del servidor KepserverEX, poner en uno la variable bombaon y cambiar nuevamente este valor a cero, verificar que la bomba se encienda y que la variable **nivels** siga la variable Setpoint.

VII. DISEÑO DE LOS SISTEMAS DE MONITOREO PARA LAS PLANTAS DE PRESION, NIVEL Y TEMPERATURA

Si la comunicación entre el PLC/Matlab/Rtai-Lab y los servidores OPC DCOM se ha efectuado correctamente, se procede a diseñar los esquemas de monitoreo, configurar las alarmas y los eventos para las plantas de nivel, presión, temperatura. Los pasos que se deben seguir para el diseño y monitoreo mediante en MM&SW son los siguientes:

Iniciar el servidor

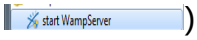
- Iniciar la máquina virtual (**Hmiwebopc**) que se encuentra en el servidor del laboratorio de control de procesos.
- Verificar la IP en la máquina virtua y escribirla en la tabla H-7

Tabla H -7: Tabla para ingresar la dirección IP de la máquina virtual instalada en el computador que hace de servidor en el LCP

IP máquina virtual servidor	

Verificar el cliente

- Ubicarse en otro computador cualquiera con conexión a internet, abrir Mozilla, y en la barra de direcciones colocar la siguiente dirección: http://ip_maquina_virtual/hmiwebopc, por ejemplo: si la direccion IP de la máquina virtual es 192.168.122.200 ingresar: http://192.168.122.200/hmiwebopc

En el caso que aparezca el mensaje la página no está disponible, se debe verificar que en la máquina virtual se esté ejecutando el servidor, para esto ir a inicio e iniciar el servidor wampserver(

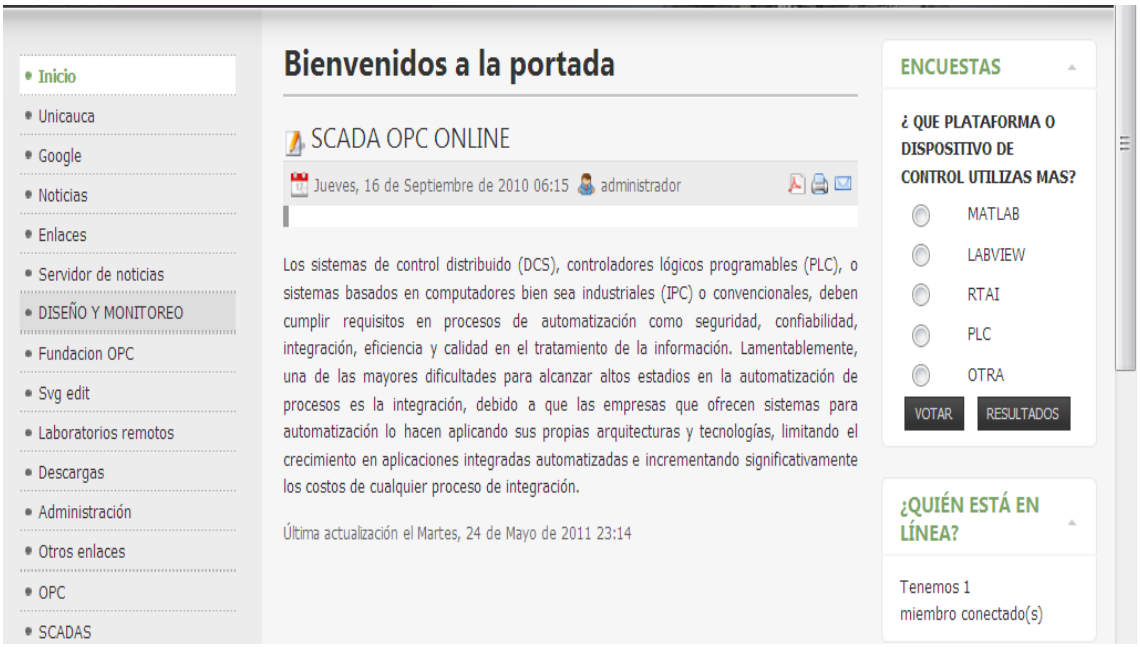
). Verificar que los computadores estén conectados a la red.

- Al ingresar la dirección URL en el navegador, se presenta una página de inicio, ver ilustración H-50, la cual está compuesta por un menú principal, y un formulario para que los usuarios registrados ingresen sus datos y tengan acceso a información, funciones restringidas para los usuarios invitados.

Ilustración H-51. Página principal del sistema HMI

Para que se active en el menú principal los enlaces para diseño de proyectos, encuestas, usuarios en línea entre otros se debe ingresar el nombre de usuario (**admin**) y la contraseña (**scadaopc**), ver ilustración H-52.

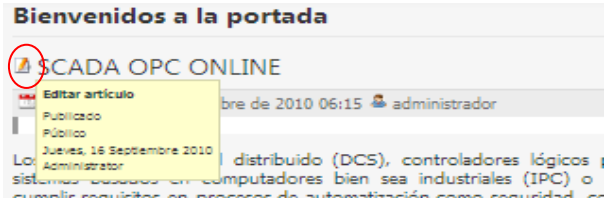
Ilustración H-52. Página del sistema de monitoreo para usuarios registrados



Fuente: Propia.

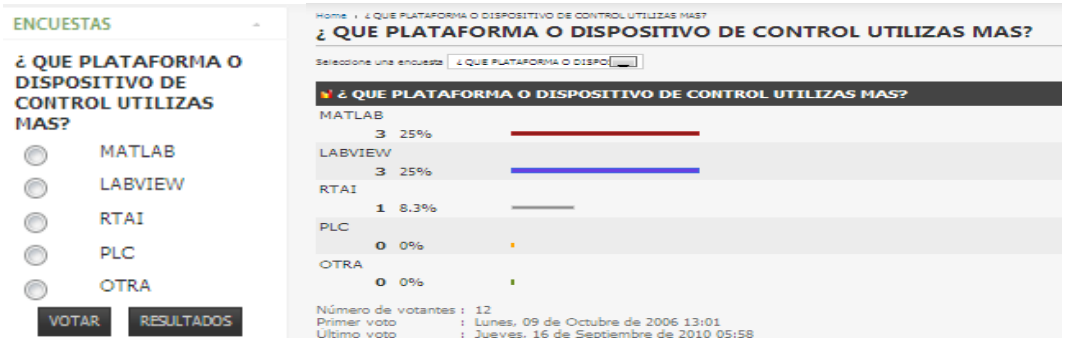
Explorar las herramientas que se presentan para los usuarios registrados, los cuales pueden editar los artículos, ver ilustración H-53, diseñar encuestas ver ilustración H-54, y utilizar todas las herramientas que joomla presenta (actualización de noticias, anuncios, entre otros, ver el manual de usuario de Joomla (3).

Ilustración H-53. Edición de artículos en el sistema de monitoreo



Fuente: Propia.

Ilustración H-54. Selección de encuestas



Fuente: Propia.

Diseño y monitoreo

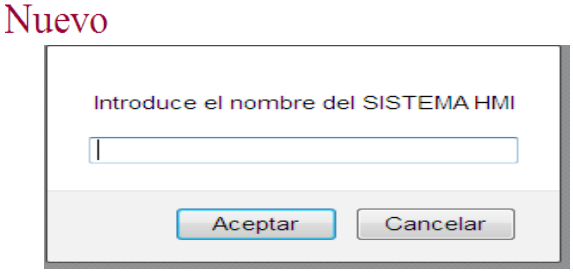
Para ingresar al sub-módulo de diseño y monitoreo, se debe hacer clic en el enlace “diseño y monitoreo” del menú principal; al presionar este enlace aparece una página con el menú: nuevo, eliminar, editar, visualizar, ayuda, inicio, ver ilustración H-55.

Ilustración H-55: Menú principal de la pantalla de diseño.

Fuente: propia

El primer paso es establecer el nombre del proyecto, haciendo clic en la opción “nuevo”, para que se presente un formulario como el que se indica en la ilustración H-55.

Ilustración H-56: Formulario para crear un nuevo proyecto



Fuente: propia

Luego de establecido el nombre del proyecto, se puede iniciar con el diseño; haciendo clic en “editar” del menú principal, para que se presente un formulario que permite seleccionar el proyecto con el que se va a trabajar, ver ilustración H-57.

Ilustración H-57: Menú “Editar” del sub-módulo de diseño en el MM&SW

Fuente: propia

Al seleccionar el nombre del proyecto, se presenta otro menú con las opciones: fuente de datos, diseño gráfico, configurar eventos, configurar alarmas y configurar ayuda, ver ilustración H-58.

Ilustración H-58: Menú “Editar” del sub-módulo de diseño

Fuente: propia

A continuación se detalla cada ítem del menú “editar”.

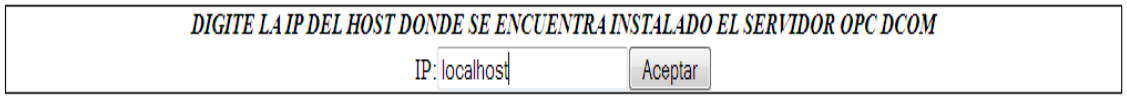
Fuente de datos: antes de empezar a diseñar los esquemas de los procesos, se debe seleccionar la fuente de los datos, importar los objetos ítems de los Servidores OPC y almacenarlos en la base de datos para su posterior uso. Al seleccionar “fuente de datos” se presenta un formulario con las opciones: servidor OPC DCOM, servidor OPC XML u otra, ver ilustración H-59.

Ilustración H-59: Menú principal para la selección de la fuente de datos

Fuente: propia

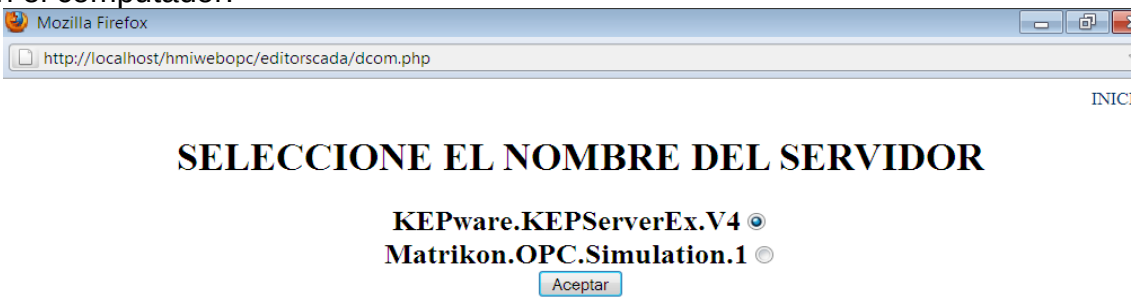
Se debe seleccionar como fuente de datos “servidor OPC DCOM”, para que se presente un formulario que permite digitar la dirección IP del host (equipo planta de presión o máquina virtual SEDS) donde se encuentra el servidor OPC DCOM, ver ilustración H-60. Luego de que se ingresa la IP se presenta un formulario que permite seleccionar el nombre servidor, ver ilustración H-61, al seleccionar el nombre del servidor se presenta un tercer formulario que permite establecer la ruta completa donde está el grupo de ítems OPC DCOM, ver ilustración H-62. Finalmente un cuarto formulario permite visualizar todos los objetos ítems existentes en el objeto grupo del servidor OPC, ver ilustración H-63, se debe elegir los objetos ítems que se van a utilizar y hacer clic en guardar.

Ilustración H-60: Formulario para ingresar la dirección IP del host donde está el servidor OPC DCOM.



Fuente: propia

Ilustración H-61: Formulario que permite seleccionar los servidores disponibles en el computador.

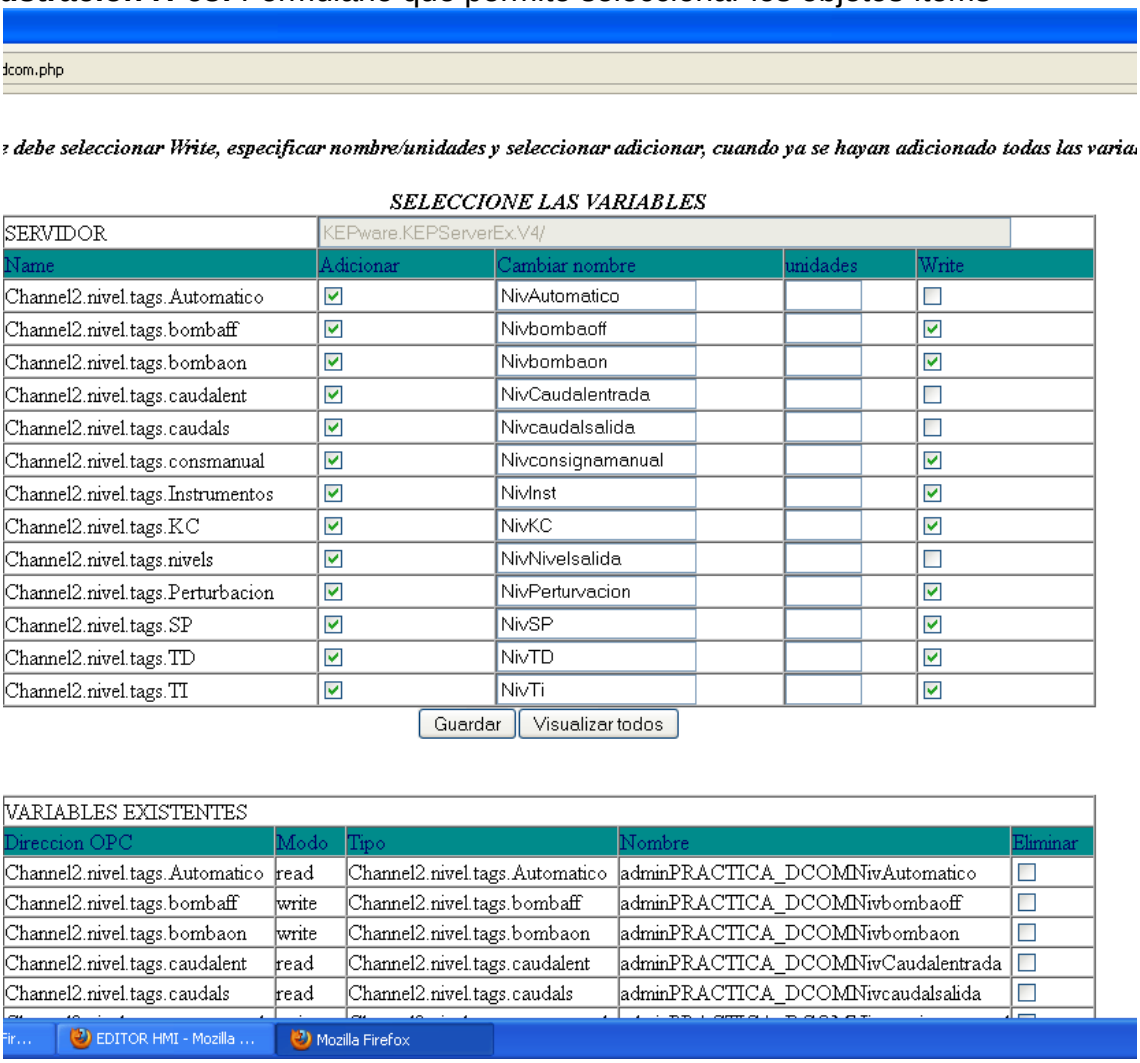


Fuente: propia

Ilustración H-62: Formulario para ingresar la ruta donde está el grupo de objetos ítems

Fuente: propia

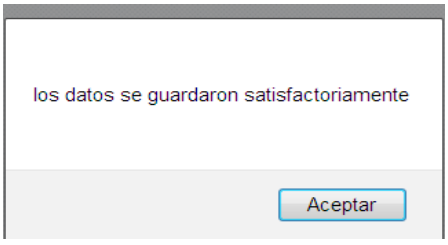
Ilustración H-63: Formulario que permite seleccionar los objetos ítems



Fuente: propia

Al hacer clic en guardar, debe aparecer un formulario como el que se indica en la ilustración H-64.

Ilustración H-64: Formulario que afirma que los datos se guardaron correctamente.

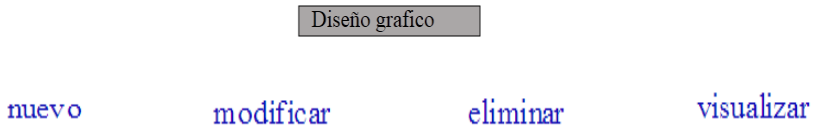


Fuente: propia

Se debe importar todas las variables de los servidores OPC DCOM instalados y configurados en la planta de presión y en la máquina virtual de SED.

Diseño de esquemas de los procesos: Al seleccionar Diseño gráfico, aparece un nuevo menú con las opciones: nuevo, modificar, eliminar, visualizar, ver ilustración H-65.

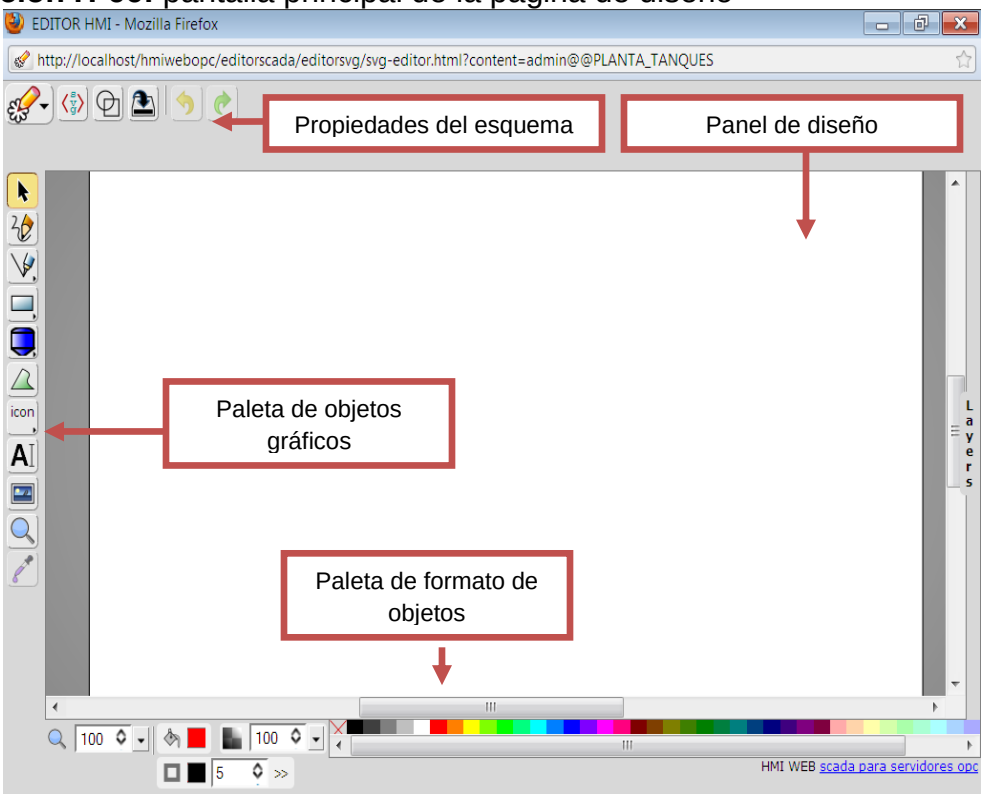
Ilustración H-65: Menú diseño grafico



Fuente: propia

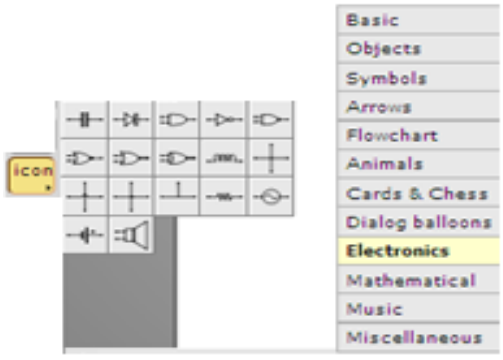
El item "nuevo" permite ir a una página donde se presenta una pantalla que permite diseñar HMI, ver la ilustracion H-66, en esta página se puede crear un esquema con imágenes en formato SVG y asociarlas a funciones creadas en JavaScript para las respectivas animaciones.

Ilustración H-66: pantalla principal de la página de diseño

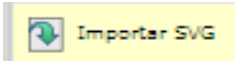


Esta página esta compuesta por:

- ***Propiedades del esquema:***
  - ✓  *P*ermite importar, exportar imágenes, configurar las propiedades del esquema: nombre, tamaño, lenguaje y color.
  - ✓  Permite visualizar código fuente y modificarlo
  - ✓  Muestra solo bordes de la figura
  - ✓  Clona un objeto
  - ✓  Agrupar varios elementos en uno solo elemento.
- ***Paleta de objetos gráficos:*** permite obtener objetos de diseño predeterminados y de instrumentación (círculos, cuadrados, texto, líneas, tanques, etc.).
  - ✓  permite seleccionar un objeto
  - ✓  Lápiz
  - ✓  Dibuja líneas
  - ✓ 
  - ✓  Tanques, columnas, pilotos
  - ✓  Permite crear un dibujo.

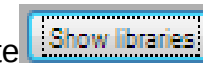


  - ✓  Permite importar imágenes jpg, gif, png etc.

Además de la paleta de objetos existentes, también se puede importar gráficos con formato SVG, haciendo clic en el icono  y luego 

.

También existen dos librerías, que permiten importar gráficos SVG, se debe ir a  luego 

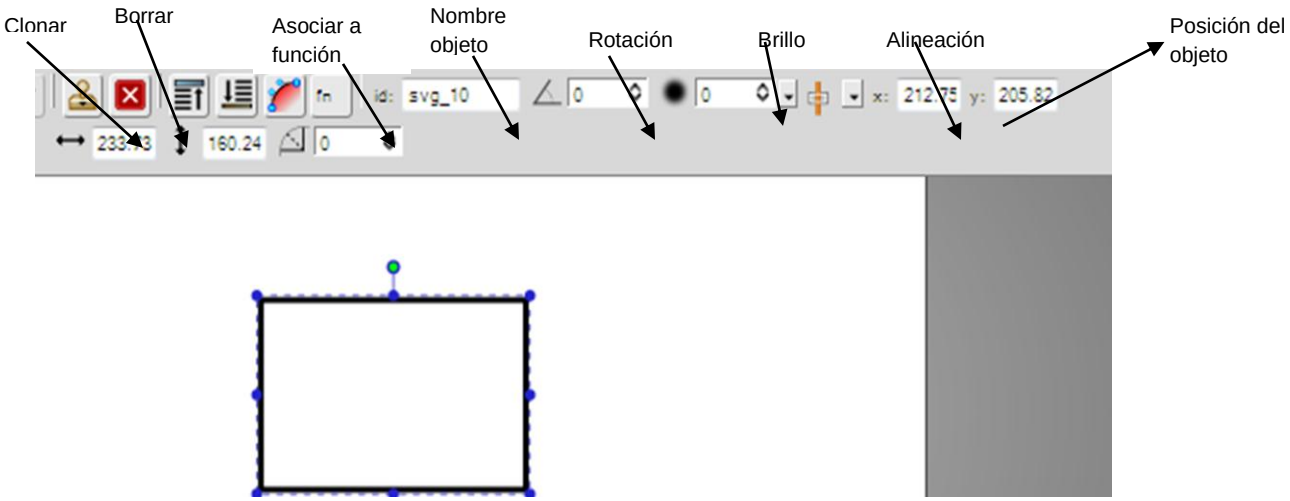
y finalmente 

.

- ***Panel de diseño:*** en este panel se puede crear los esquemas de los procesos.

Paleta de configuracion de objetos SVG: A cada uno de los objetos se les puede configurar las propiedades como lo indica la ilustración H-67. Al dar clic derecho en cada objeto se pueden configurar los siguientes parametros: la posicicion, el angulo de rotacion, el color, la intensidad entre otros.

Ilustración H-67. Formulario de configuración de objetos



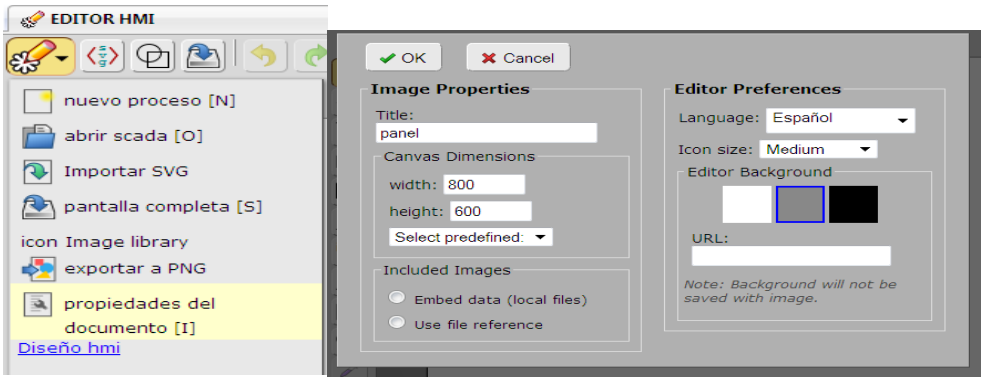
Fuente: Propia.

Para crear un esquema interactivo, que permita el monitoreo de procesos, se deben seguir los pasos que se listan a continuación:

5. Establecer el nombre del esquema
6. Diseñar el esquema
7. Asociar cada elemento del esquema a una función de Javascript
8. Guardar

Para colocarle el nombre al esquema se presiona el icono, se hace clic en **propiedades**, colocar un título y establecer las propiedades de la ventana del editor, ver ilustración H-68.

Ilustración H-68. Formulario que permite configurar las propiedades del esquema.



Fuente: propia

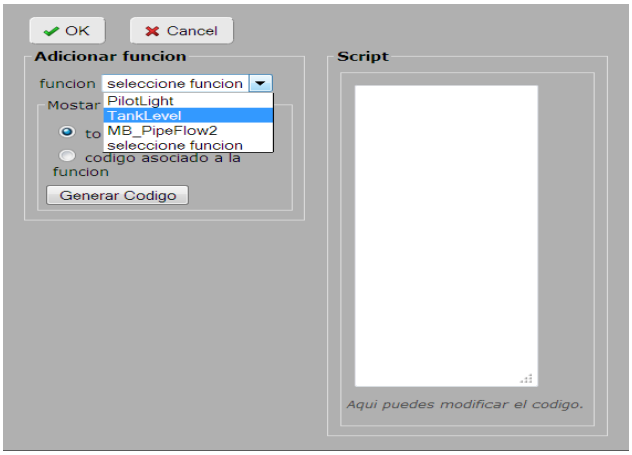
El segundo paso es diseñar el esquema, el cual hace uso de la paleta de objetos gráficos. La ilustración H-69 presenta un ejemplo de un esquema diseñado mediante gráficos SVG.

Ilustración H-69. Ejemplo de un esquema diseñado en el MM&SW

Las imágenes en formato SVG encadenadas con el lenguaje de programación JavaScript permiten crear animaciones. Este sistema permite asociar cada grafico SVG a una funcion en javaScript y hacer uso de las variables OPC, ya sea para la visualizacion o para el envio de datos.

Para asociar cada objeto grafico a una funcion, se debe hacer clic en el icono fn, donde dependiendo del tipo de objeto se presenta un formulario con diferentes funciones, ver ilustracion H-70.

Ilustración H-70. Formulario para asociar un gráfico a una función.



Fuente: Propia.

Dentro de las funciones asociadas a cada imagen tenemos:

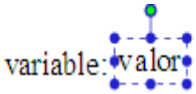
- ***Writeinmediate():*** permite al usuario enviar un valor a un elemento del servidor OPC, haciendo clic en el grafico. Esta funcion esta dispoinble para todos los objetos graficos.
- ***stringDysplay():*** Toma el valor de una variable y lo visualiza en la pantalla. . Esta funcion esta disponible unicamente para el objeto texto.
- ***Rotate():*** rota la imagen según el valor de una variable. Esta funcion esta disponible para todos los objetos graficos.
- ***Translate():*** mueve la imagen según el valor de una variable. Esta funcion esta disponible para todos los objetos graficos.
- ***SlideDisplay():*** visualiza el cambio de una variable, cambiando de color la imagen. Se utiliza para simular que hay flujo en una tuberia, o que esta circulando una corriente. Esta funcion esta disponible para los rectangulos y cuadrados.

- ***TankLevel():*** visualiza el nivel de un tanque. Esta funcion esta disponible para los rectangulos y cuadrados.
- ***PilotLight():*** cambia el color de la imagen dependiendo del valor recibido, si es uno en rojo, si es cero en verde.

A continuación se detalla el uso de estas funciones:

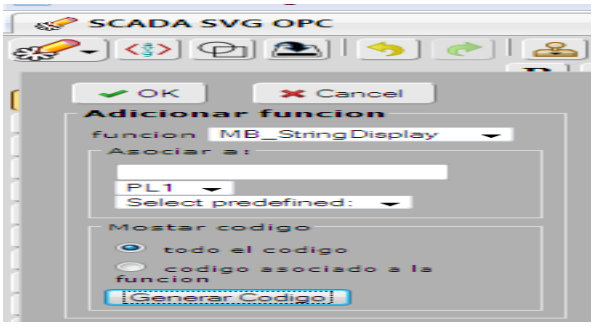
stringDisplay(): si se quiere visualizar datos de una variable se debe dar clic en el icono  ubicado en la paleta de objetos graficos, colocarlo en el panel de diseño, ver ilustración H-71, digitar un texto, en este caso "**valor**", seleccionar este objeto y presionar el botón  para que se presente un formulario como el de la ilustración H-72; seleccionar la función MB_StringDisplay, y escoger la variable OPC que se quiere visualizar, y finalmente presionar el botón "generar código" y luego OK.

Ilustración H-71. Formulario para visualizar el valor de una variable.



Fuente: Propia.

Ilustración H-72. Formulario para configurar las opciones de visualización de valores de objetos ítems.

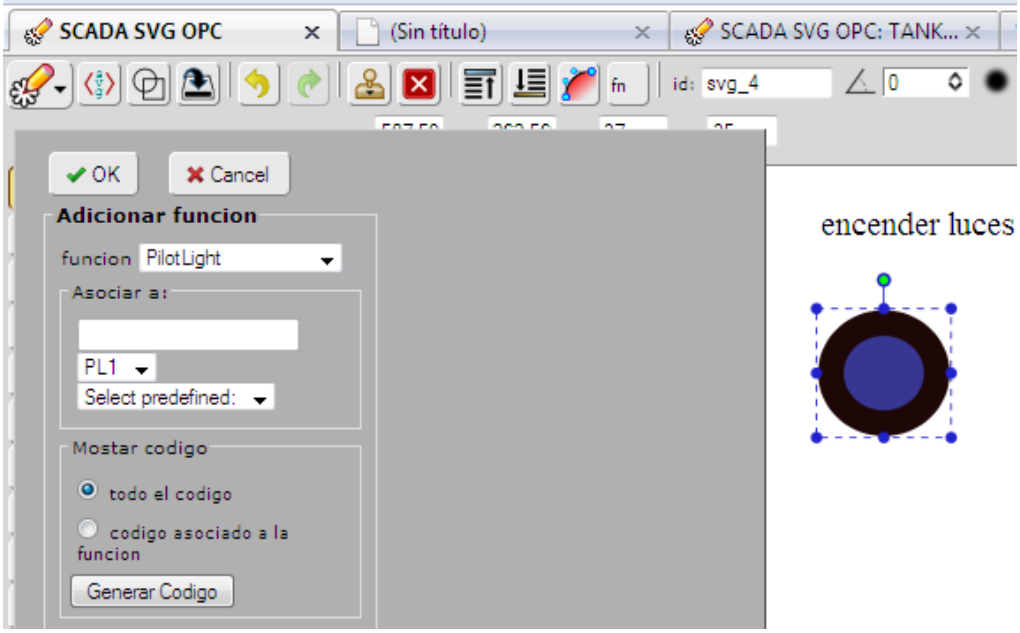


Writeinmediate: Para cambiar el valor de una variable, se debe seleccionar un objeto, hacer clic en el icono 

, seleccionar la función Writeinmediate, asociar a la variable OPC a la que se le cambiara el valor y hacer clic en generar código.

PilotLight: si se quiere colocar simular un piloto, se selecciona un circulo se configura el tamaño, se selecciona la función PilotLight, se asocia a una variable y se presiona generar código, ver ilustración H-73.

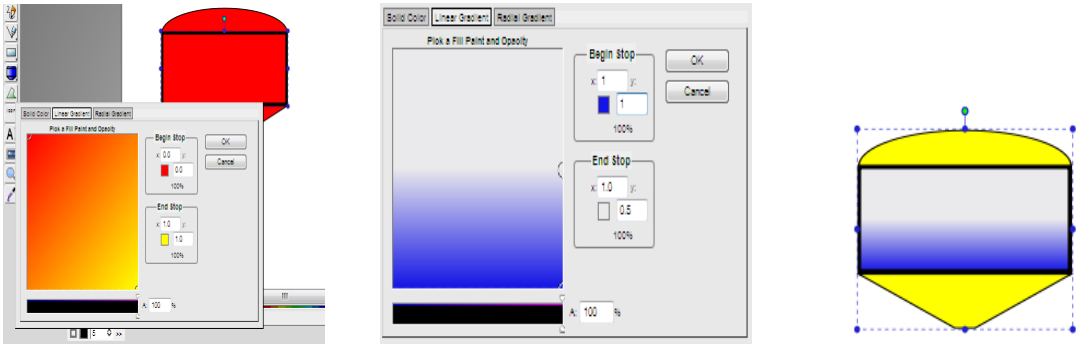
Ilustración H-73. Configuración de un círculo que actuara como piloto.



Fuente: Propia.

Si se quiere visualizar el nivel del tanque se coloca el tanque con el icono  , se sobrepone un cuadrado sobre el tanque, se selecciona este cuadrado, se presiona sobre el icono  que se encuentra en la parte inferior, para que aparezca un formulario de colores, del cual se debe seleccionar **linear gradient,** establecer los colores y configurar los parámetros **Begin stop:** X=1, Y=1 **end stop:** x=1 Y=0.5 y OK. Ver ilustración F-29.

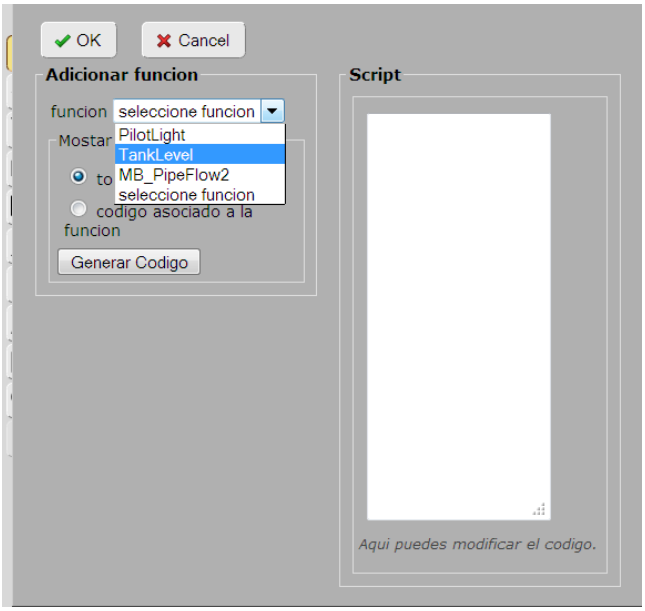
Ilustración H-74. De un rectángulo para que permita la visualización de nivel de un tanque.



Fuente: Propia.

Seccionar el cuadrado y dar clic en el icono **asociar función**, selecciona la función **tank_level**, el rango máximo con que se llena el tanque, elegir la variable OPC y presionar generar código, ver Ilustración H-75.

Ilustración H-75. Formulario de configuración del nivel de un tanque.



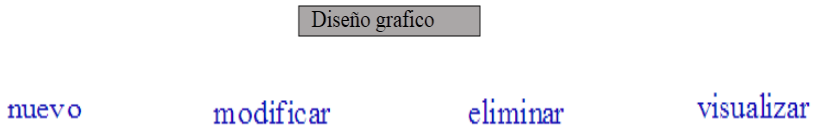
Fuente: Propia.

Cuando el diseño esté listo se debe presionar el icono  (guardar), y luego  para que se almacene la información y permita el retorno a la página principal.

Se debe realizar el diseño de los sistemas de monitoreo para las plantas de presión, nivel y temperatura, utilizando los pasos mencionados anteriormente.

Dentro de diseño gráfico también se puede modificar, visualizar, eliminar cada uno de los esquemas diseñados, para cualquiera de estas opciones se debe hacer clic en cada menú y seleccionar el esquema al que se va a modificar, eliminar o visualizar, ver ilustración H-76.

Ilustración H-76. Formulario de diseño gráfico en el MM&SW

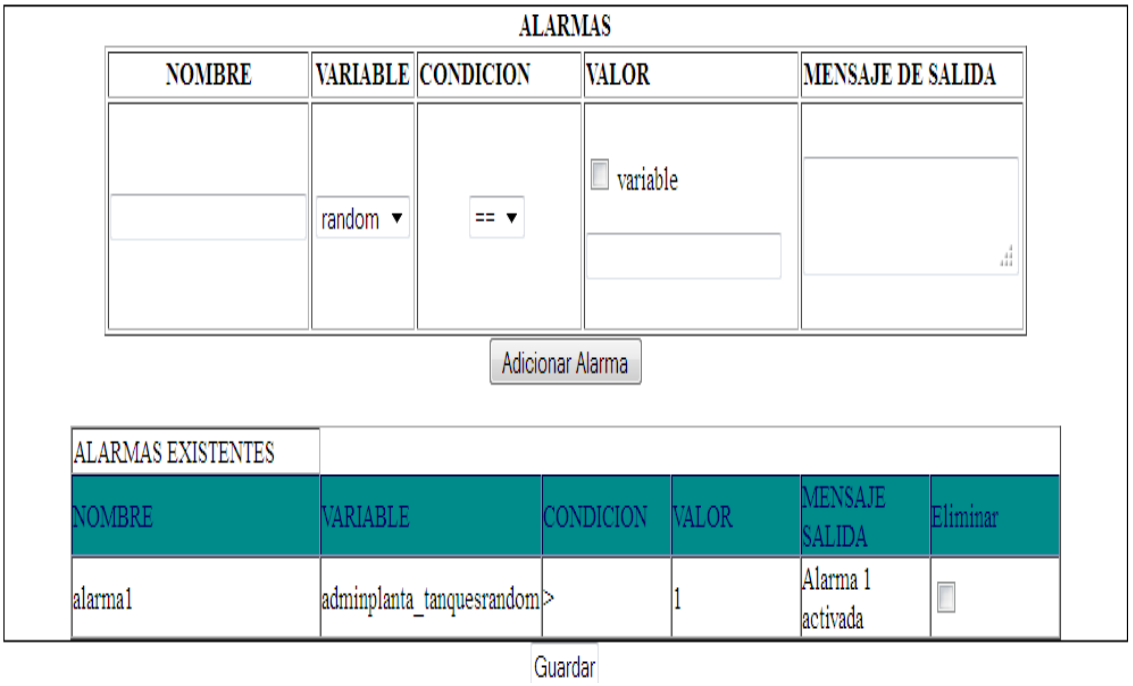


Fuente: Propia.

Configuración de alarmas: Se debe configurar las alarmas, seleccionando la opción “editar/Configurar Alarmas”, para que se presente un formulario, ver ilustración H-77, de configuración. Los parámetros que se deben establecer para adicionar las alarmas son: nombre de alarma, variable, condición, y mensaje de salida. En cuadro de texto de nombre, se debe colocar la identificación de la alarma que se va a adicionar, en la celda “variable” se debe seleccionar una de las variables existentes que generara la alarma, en la opción “condición” se debe seleccionar entre los signos existentes en la lista (menor, mayor, igual, menor o igual, mayor o igual); la variable se la puede comparar con un valor por el usuario, o compararla con otra variable, y finalmente se debe configurar el mensaje que describirá el motivo de la alarma.

En la imagen H-77, también se puede visualizar una alarma ya configurada, en este caso el nombre es **alarma1,** esta alarma presenta el mensaje de salida Alarma1 activada cuando la variable random sea mayor a uno.

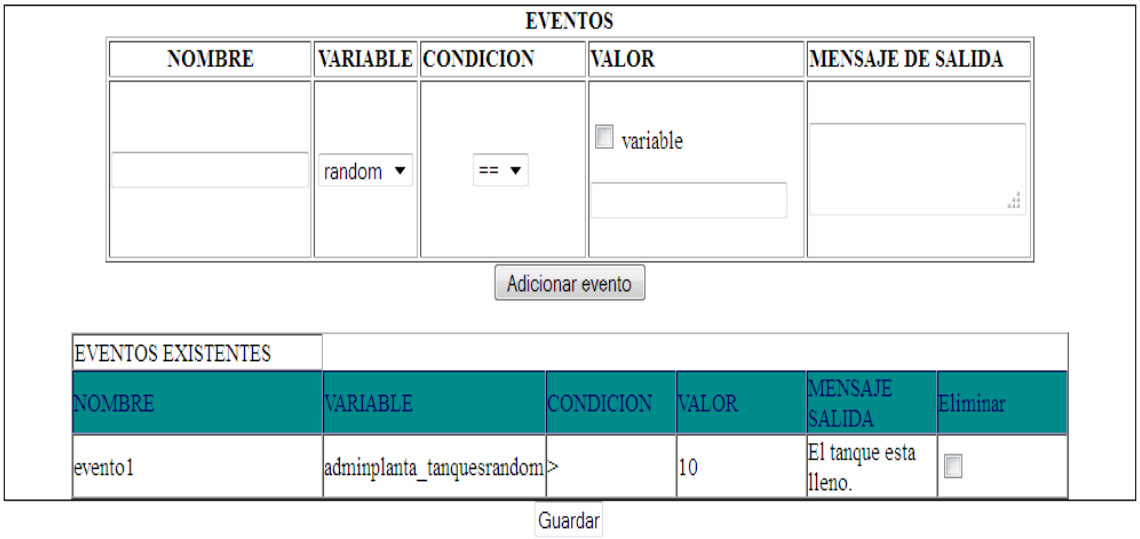
Ilustración H-77. Configuración y visualización de Alarmas



Fuente: propia

Configuración de eventos: cuando el usuario selecciona la opción "Configurar eventos", se presenta un formulario, ver ilustración H-78, con las mismas opciones de configuración de las alarmas, se debe configurar los parámetros de cada evento, hacer clic en adicionar evento, y finalmente en guardar.

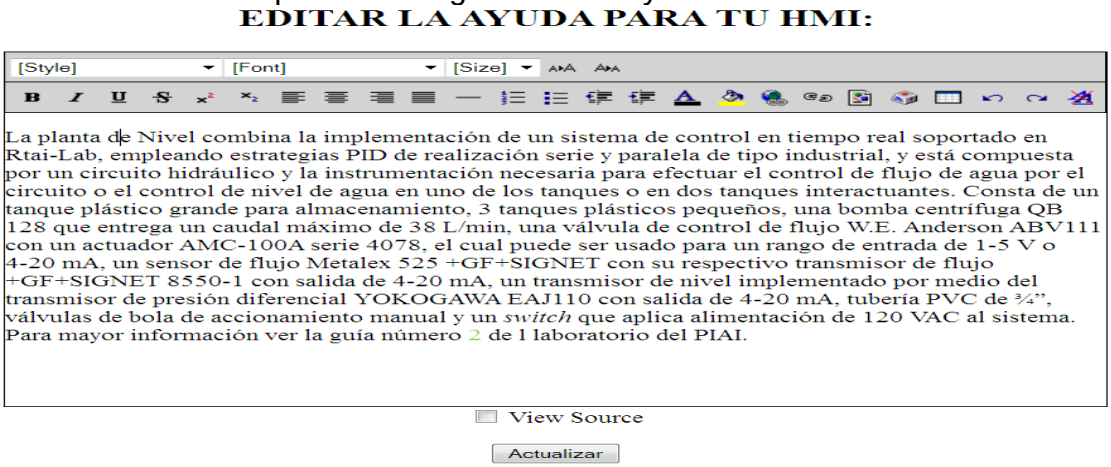
Ilustración H-78. Creación y visualización de eventos



Configurar ayuda: al hacer clic en "editar"-"ayuda" se presenta un editor que permite establecer información referente al uso del HMI, ver ilustración H-79, este editor presenta funciones similares a *Word*, permite adicionar tablas, imágenes, entre otros. Al ingresar información internamente se genera el código HTML el cual puede ser modificado presionando la opción *view source.*

Se debe ingresar las instrucciones para el manejo de cada HMI, hacer clic en actualizar, y finalmente en guardar, (siempre se debe actualizar antes de guardar).

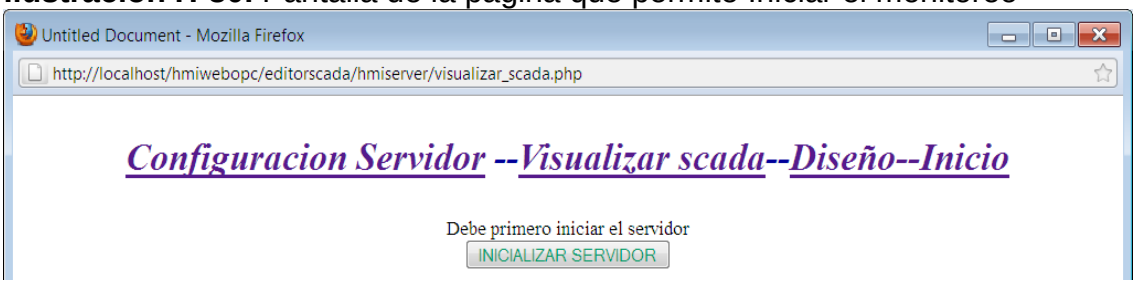
Ilustración H-79. Editor para la configuración de ayuda del HMI



Para monitorear las plantas se debe ingresar a "**visualiazar**", seleccionar el nombre del proyecto, debe aparecer una página con los menús: configuración servidor, iniciar monitoreo, visualizar scada, diseño, Inicio e inicializar servidor, ver ilustración H-80. El menú inicio lo enlaza a la página principal, el menú diseño lo retorna a la página anterior, y el botón "iniciar servidor" permite establecer la

conexión con el servidor para el monitoreo y supervisión. Antes de hacer clic en Iniciar Monitoreo, se debe presionar primero el botón “iniciar servidor” y esperar cinco segundos.

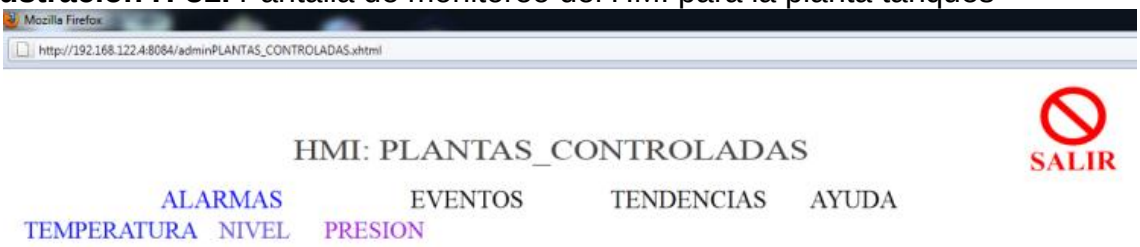
Ilustración H-80. Pantalla de la página que permite Iniciar el monitoreo



Fuente: propia

Al hacer clic en **iniciar monitoreo** se presenta una nueva página que permite la visualización de los procesos diseñados, las alarmas, tendencias y eventos., ver ilustración H-81.

Ilustración H-81. Pantalla de monitoreo del HMI para la planta tanques



Fuente: propia

Las alarmas, los eventos y tendencias presentan el mismo formato para todos los proyectos.

Menú Alarmas: en el MM&SW las alarmas son mensajes de texto que indican que una condición de falla está presente.

La ilustración H-82 presenta una pantalla que permite visualizar las alarmas que se han configurado en el módulo de diseño. Al hacer clic en el menú **alarmas** se presenta una tabla con el nombre, el estado, el tiempo en que fue activada, y el número de veces que se repite esta acción.

Ilustración H-82. Pantalla de la página que permite Iniciar el monitoreo

Alarm:	Estado de Alarma:	Tiempo:	Tiempo OK:	Contador:
Alarma 1 activada	Alarma Activa	Thu Jun 23 2011 14:20:24 GMT-0500	Not OK	1

Fuente: propia

El mensaje de alarma permanecerá visible hasta que el mensaje ha sido reconocido por el operador y la entrada en representación de la culpa se ha restablecido.

Las alarmas pueden estar en los siguientes estados:

- Inactivo: no hay ningún mensaje de alarma asociado a la espera de ser reconocido.
- Activo: una condición de falla está presente y el mensaje de alarma no ha sido reconocido por el operador.
- Reconocido: a condición de falla está presente, y el operador ya la ha visualizado el mensaje.

Las alarmas sólo en los estados: "activo", "reconocido" se presentan en pantalla, no habrá mensajes visibles en la tabla de alarma para el estado "inactivo".

Las alarmas son detectadas por el servidor HMI mediante el control de un valor booleano que se define en la configuración. Cuando ese valor booleano es *True*, se activa la alarma, y si el valor booleano es *False* se desactiva.

Cada mensaje de alarma contendrá la siguiente información:

- El texto de alarma: Esta es una descripción de la alarma.
- El estado de alarma: describe el estado actual de la alarma (por ejemplo, "inactivo", "activo", "reconocido", etc.).
- El tiempo: Presenta el tiempo en que el servidor ha detectado la alarma.
- Aceptar el tiempo: este es el momento en que el usuario acepto la falla.
- Contador: número de veces que la alarma se ha presentado (el valor medido ha cambiado de false a true). Una vez que la alarma ha desaparecido de la pantalla, este número se restablece.

- "Reconociendo" la alarma significa que el operador ha tomado una acción para indicar que ha visto los mensajes de alarma.

Historial de alarma: cuando un mensaje de alarma ha cambiado desde el activo a estado inactivo, una copia del mensaje se agrega a la historial de alarma. El historial de alarmas es un registro secuencial de alarmas.

Menú Eventos:

Los eventos son mensajes que representan a la ocurrencia de incidentes sobre los cuales debe ser el operador notificado, pero que el operador normalmente no necesita tomar ninguna acción. A diferencia de las alarmas, el operador no reconoce los eventos.

Cada vez que se genera un evento, se presenta un mensaje en la pantalla de eventos. Esto significa que el mismo evento puede aparecer varias veces en la tabla de visualización de eventos, ver la ilustración H-83.

Ilustración H-83. Formulario de visualización de eventos

Evento #:	Fecha:	Evento:	Estado:
1308856822	Thu Jun 23 2011 14:20:24 GMT-0500	El tanque esta lleno.	1

Fuente: propia

La tabla de eventos incluye la siguiente información:

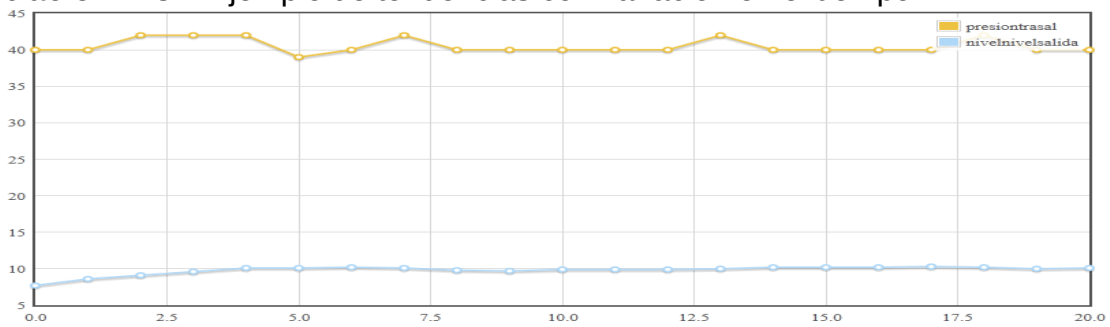
- El número de evento: este es un número secuencial generado por la pista del servidor los mensajes de eventos.
- Tiempo del evento: este es el momento en que se generó el evento.
- Evento. descripción del evento.
- Estado: un valor que indica el estado del evento (0 para apagado, 1 para el).

Menú Tendencias:

En cada proyecto se presenta una gráfica que permite representar todas las variables creadas en el módulo de diseño, ver ilustración H-83, aquí se debe seleccionar las variables que se quieren representar.

La ilustración H-84 presenta una pantalla con una gráfica de cuatro variables creadas en el módulo de diseño, el eje de la abscisa representa el tiempo en minutos, y el eje Y representa el valor que tiene cada objeto ítem creado en el servidor OPC.

Ilustración H-84. Ejemplo de tendencias con variación en el tiempo



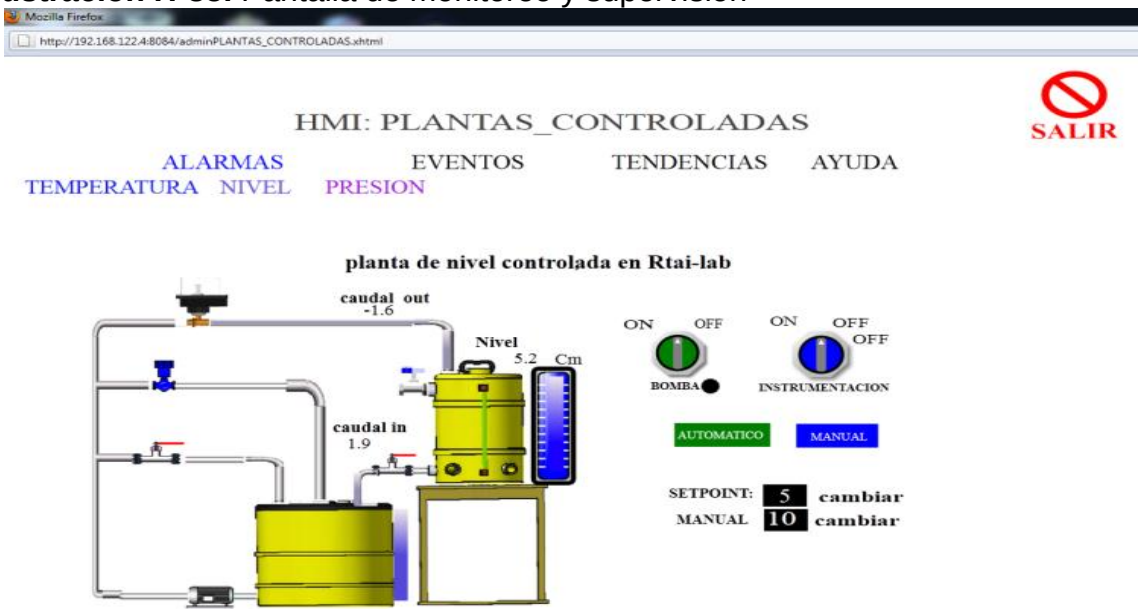
Fuente: Propia

Menú Ayuda:

Al hacer clic en ayuda, se presenta una página con un manual de usuario para cada proyecto.

Esquemas para el monitoreo: hacer clic en el menú correspondiente a cada esquema, para que se presente un formulario con los objetos gráficos creados en el módulo de diseño. La ilustración H-85 presenta un ejemplo de un ejemplo de una página del MM&SW que permite el monitoreo de la planta de nivel.

Ilustración H-85. Pantalla de monitoreo y supervisión



Fuente: Propia

III. PRUEBAS y RESULTADOS OBTENIDOS

En los supervisorios se pueden visualizar los mismos valores establecidos en los objetos ítems de cada servidor, se puede iniciar a realizar pruebas, en caso contrario se debe verificar que pasó a paso la comunicación entre cada módulo, algunos de los pasos que se deben seguir en caso de que presenten falla se listan a continuación:

- Verificar que todos los computadores asociados estén conectados a internet.
- Realizar un ping hacia el computador que se crea está fallando
- Verificar que el puerto DCOM este activado donde están instalados los servidores OPC

1. Cambio de consignas

Realizar cambio en las consignas y escribir el valor de salida obtenido en la siguiente tabla:

Planta	Consigna	Valor de salida visualidos
Nivel		
Presión		
temperatura		

Alarmas

En la siguiente tabla colocar la condición por la que se debe generar la alarma y la imagen de salida de cada una de las alarmas generadas.

Planta	Condición	Pantalla donde se genera la alarma

Eventos

En la siguiente tabla colocar la condición por la que se debe generar la alarma y la imagen de salida de cada uno de los eventos generados.

Planta	Condición	Pantalla que permite visualizar el evento

Tendencias

En la siguiente tabla describir y colocar imágenes de las tendencias de cada una de las plantas.

Planta	descripción	tendencias
Nivel		

Presión		
Temperatura		

Tiempo de respuesta de cambio de variables:_____

Aplique disturbios en cada una de las plantas, observe y describa el efecto que producen._____
_____

Inconvenientes al realizar la práctica: _____

Finalización

Para finalizar el monitoreo **siempre** se debe hacer clic en "salir" parte superior izquierda, para que aparezca la página de configuración del servidor HMIServer, ver ilustración H-86, se presenta un formulario que permite recargar las variables en el servidor, y finalizar el monitoreo.

Ilustración H-86. Página que permite finalizar el monitoreo

Para finalizar el cliente en Rtai-Lab se debe ubicar el el computador de la planta de Nivel y en el terminal donde se ejecuto la tarea presionar las teclas **control +C**

Para finalizar el cliente en Matlab se debe ubicar el el computador de la planta de Presion y en la pantalla de supervision del intercambiador de calor presionar STOP.

Cerrar los servidores OPC DCOM ubicados en el computador de la planta de presion y en la maquina virtual del computador de la planta de SEDS.

ANEXO I. PRUEBA DE INTEGRACION DE MATLAB Y RTAI-LAB MEDIANTE OPC XML

VIII. INTRODUCCION

En este anexo se presenta una guía para la integración de Matlab y Rtai-Lab mediante el estándar OPC XML. El objetivo de esta práctica es que los estudiantes conozcan las ventajas que brinda OPC XML, y afiancen los conocimientos adquiridos en el transcurso de la carrera.

Para la integración de Matlab y Rtai-Lab, se hace uso de la planta de Nivel del laboratorio de control de procesos del PIAI, de una planta una planta virtual implementada en Matlab y de los módulos: MSOX, MCOXM, MCOXR, y el MM&SW. La ilustración I-1 presenta el esquema general de integración mediante el estándar OPC XML. Las plantas de nivel y temperatura son controladas en Rtai-Lab y Matlab respectivamente, y los datos obtenidos son enviados al MSOX para que sean capturados por el MM&SW y se puedan representar gráficamente.

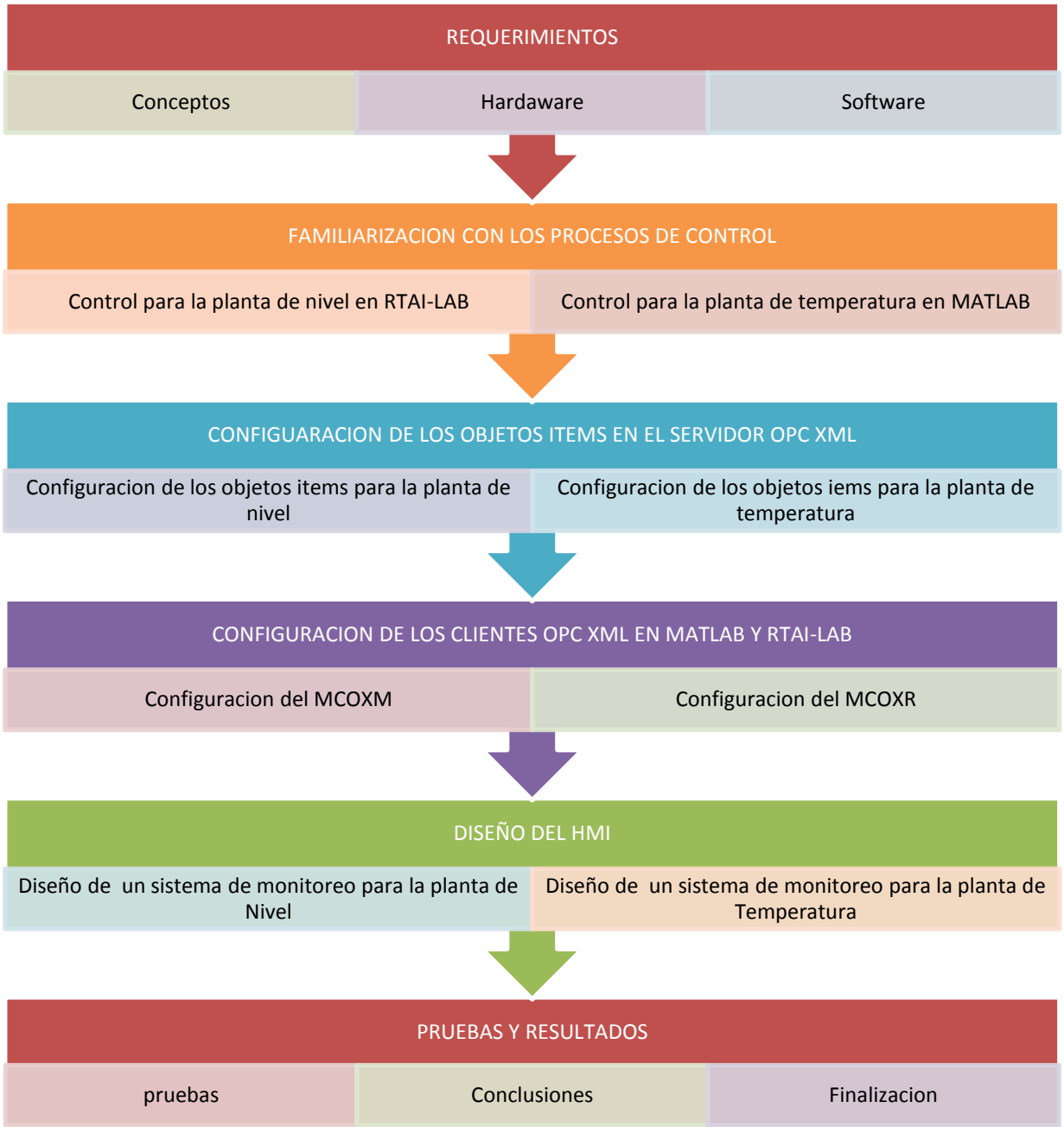
Ilustración I-1: esquema general de la integración de los elementos de control mediante el estándar OPC XML

Fuente: Propia

IX. PROCEDIMIENTO

Los pasos necesarios para la integración de Matlab, Rtai-Lab se representan en la ilustración I-2 y se explican en los numerales IV, V, VI de este anexo.

Ilustración I-2: Pasos para la integración de Matlab, Labview mediante el estándar OPC XML



X. REQUERIMIENTOS

Para la realización de esta práctica se deben manejar los siguientes conceptos, los cuales se explican en el capítulo 1 de la monografía:

- Control basado en pc.
- Plataformas de control basado en PC: Matlab y Rtai-Lab.
- OPC estándar XML
- Sistema HMI

La tabla I-1 presenta los requerimientos hardware y software necesarios en la integración de Matlab, Rtai-Lab.

Tabla I -1: Requerimientos para la integración de plataformas académicas mediante el estándar OPC XML

			Requerimientos		
Componentes		**Descripción**	**Hardware**	**Software**	
Procesos	**Control de en Rtai-lab para la planta de Nivel**	Se encarga de controlar la planta de nivel y de la comunicación remota con el servidor OPC XML.	Elementos de la planta de nivel. Un PC con conexión a una red LAN.	Rtai-Lab: Se encuentra instalado en el computador de la planta de nivel del LCP. Aquí se efectúa el control de la planta de nivel.	Diagrama de bloques de un control PID serie para la planta de nivel. El archivo **Pidserieopc** se encuentra en la carpeta **home/grupoati** del computador de la planta de Nivel.
					MCOXR: Paleta de bloques OPC XML, ver numeral III del anexo E (GUÍA DE INSTALACION Y CONFIGURACION DE LOS CLIENTES OPC DCOM/XML EN RTAI-LAB).
	Control de la planta de temperatura en Matlab	Se encarga de controlar la planta de temperatura y de la comunicación remota con el servidor OPC XML.	Un PC con Matlab y conexión a una red LAN.	Matlab: Se encuentra instalado en el computador de la planta de Presión del LCP	MCOXM:
					Diagrama de bloques para el control de temperatura: Ubicarse en el disco D del computador de la planta de presión, ir a **contrl1**, y ejecutar el archivo: heatex.m
MSOX		Encargado de integrar la información de las plataformas de control, mediante el estándar OPC XML.	Un PC con sistema operativo Windows o Linux y conexión a una red LAN para la adquisición de datos desde Matlab y Rtai-Lab.	MSOX Instalado en la máquina virtual XpOPC del computador de la planta de SEDS	➢ Iniciar la máquina virtual XpOPC ubicada en la planta de SED y verificar que en el escritorio se encuentre una carpeta con el nombre Servidor OPC.(en este servidor se configuraran las tags de Rtai-Lab y Matlab)

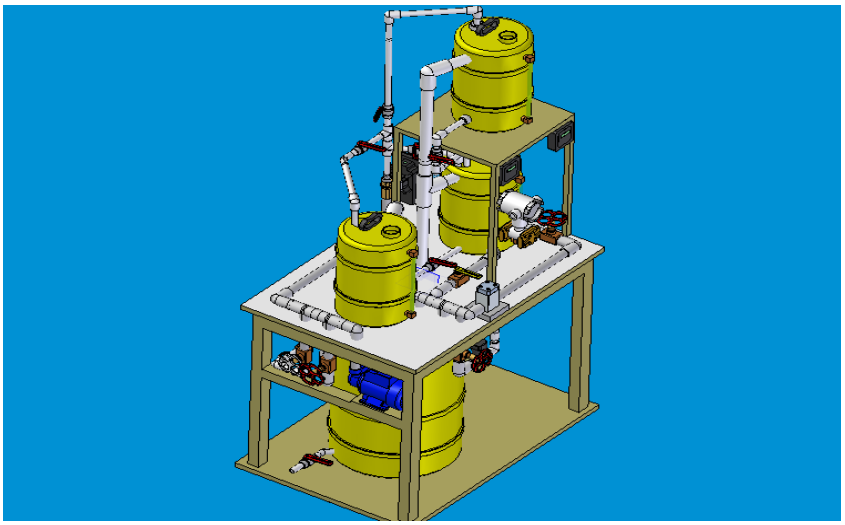
Sistema HMI	Encargado de permitir el diseño de mímicos y el monitoreo de los procesos mediante la web.	Un PC que cumple la función de servidor WEB del MM&SW ubicado en el computador que hace de servidor en el LCP. Un PC que cumple la función de cliente WEB del MM&SW. Cualquier computador con conexión a internet.	Módulo de Monitoreo y supervision Web	Se debe verificar que en el disco D del equipo servidor del laboratorio de control de procesos se encuentre la máquina virtual **hmiweopc**

XI. FAMILIARIZACION CON LOS PROCESOS DE CONTROL

1. Control PID implementado en Rtai-Lab para la planta de Nivel

El Laboratorio de Control de Procesos de la Universidad del Cauca para el programa de Ingeniería en Automática Industrial cuenta con una Planta de Nivel (PN) que combina la implementación de un sistema de control en tiempo real soportado en Rtai-Lab, empleando estrategias PID de realización serie y paralela de tipo industrial. Está compuesta por un circuito hidráulico y la instrumentación necesaria para efectuar el control de flujo de agua por el circuito o el control de nivel de agua en uno de los tanques. Consta de un tanque plástico grande para almacenamiento, 3 tanques plásticos pequeños, una bomba centrífuga QB 128 que entrega un caudal máximo de 38 L/min, una válvula de control de flujo W.E. Anderson ABV111 con un actuador AMC-100A serie 4078, el cual puede ser usado para un rango de entrada de 1-5 V o 4-20 mA, un sensor de flujo Metalex 525 +GF+SIGNET con su respectivo transmisor de flujo +GF+SIGNET 8550-1 con salida de 4-20 mA, un transmisor de nivel implementado por medio del transmisor de presión diferencial YOKOGAWA EAJ110 con salida de 4-20 mA, tubería PVC de ¾", válvulas de bola de accionamiento manual y un *switch* que aplica alimentación de 120 VAC al sistema.

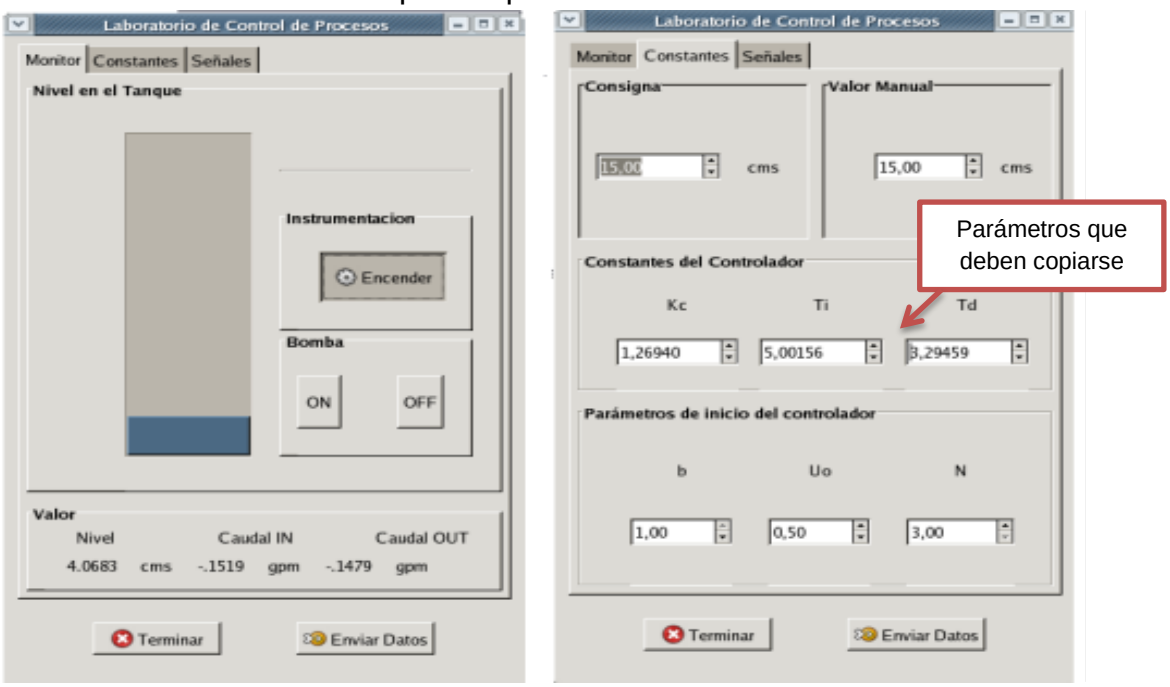
Ilustración I-3: Planta de Nivel controlada en Rtai-Lab



Fuente: **Propia**

Para poder comunicar la planta de nivel con el servidor OPC XML, se debe realizar la guía “CONTROL REGULATORIO PID SERIE”, existente en el laboratorio de control de procesos del PIAI. Cuando se haya realizado esta práctica se deben copiar los parámetros Kc, Ti, y Td los cuáles serán utilizados más adelante en la configuración de valores del servidor Kepserver, ver ilustración I-4.

Ilustración I-4: Parámetros para la planta de nivel.

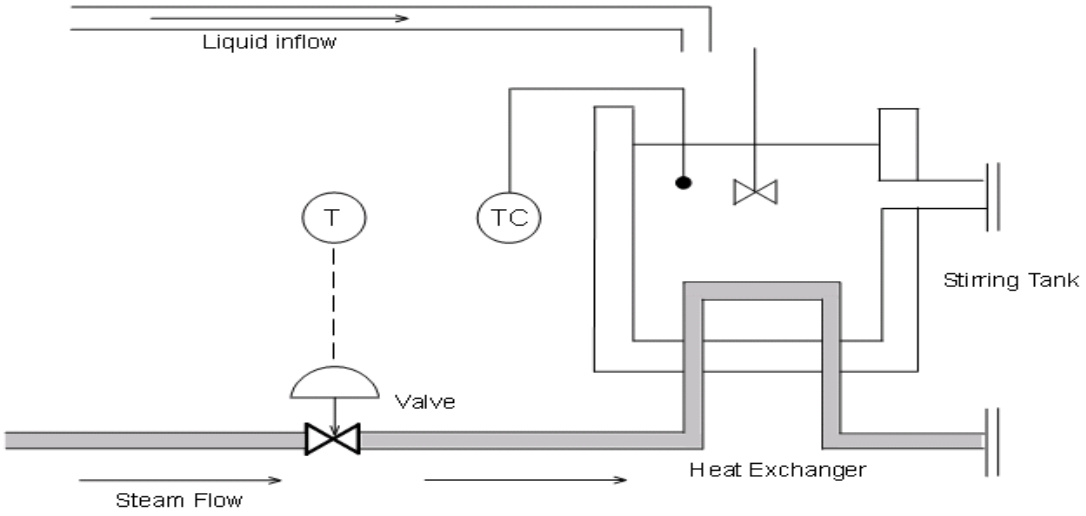


Fuente: **Propia**

2. Control PI implementado en Matlab para regular la temperatura de un Reactor Quimico

Matlab presenta una demostración de un reactor químico "tanque agitado" el cual se controla mediante un controlador PI. La entrada superior proporciona el líquido que se mezcla en el tanque, el cual debe ser mantenido a una temperatura constante mediante la variación de la cantidad de vapor suministrado al intercambiador de calor (tubo inferior) a través de su válvula de control. Las variaciones en la temperatura del flujo de entrada son la principal fuente de las perturbaciones en este proceso, ver ilustración I-5.

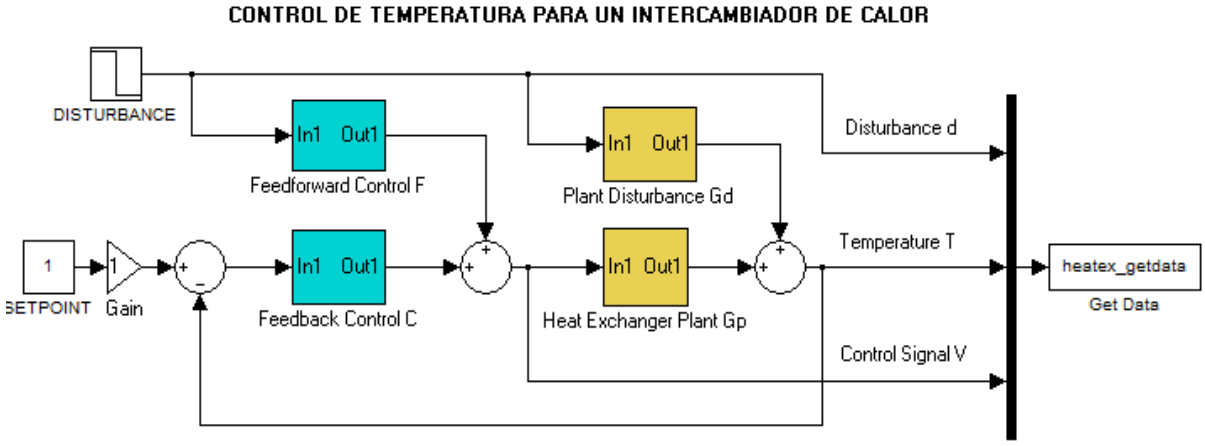
Ilustración I-5: reactores de agitación con intercambiador de calor.



Fuente: Matlab

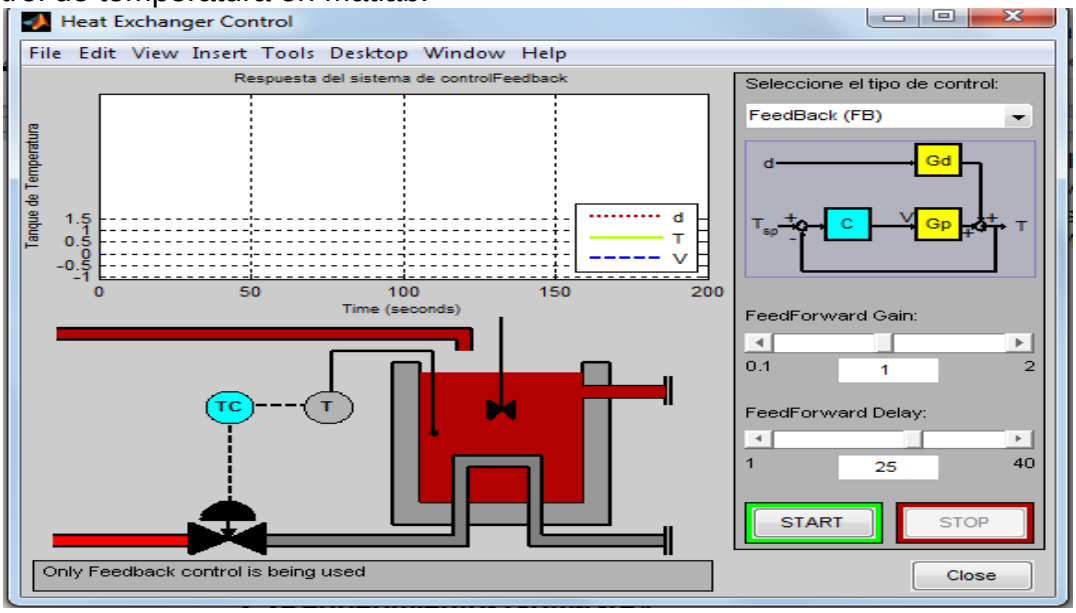
La Ilustración I-6 presenta el diagrama de bloques en simulink para el control de este proceso y la ilustración I-6 presenta la pantalla que permite establecer y visualizar los parámetros del control de temperatura implementado en simulink.

Ilustración I-6: Diagrama de bloques en simulink



Fuente: Matlab

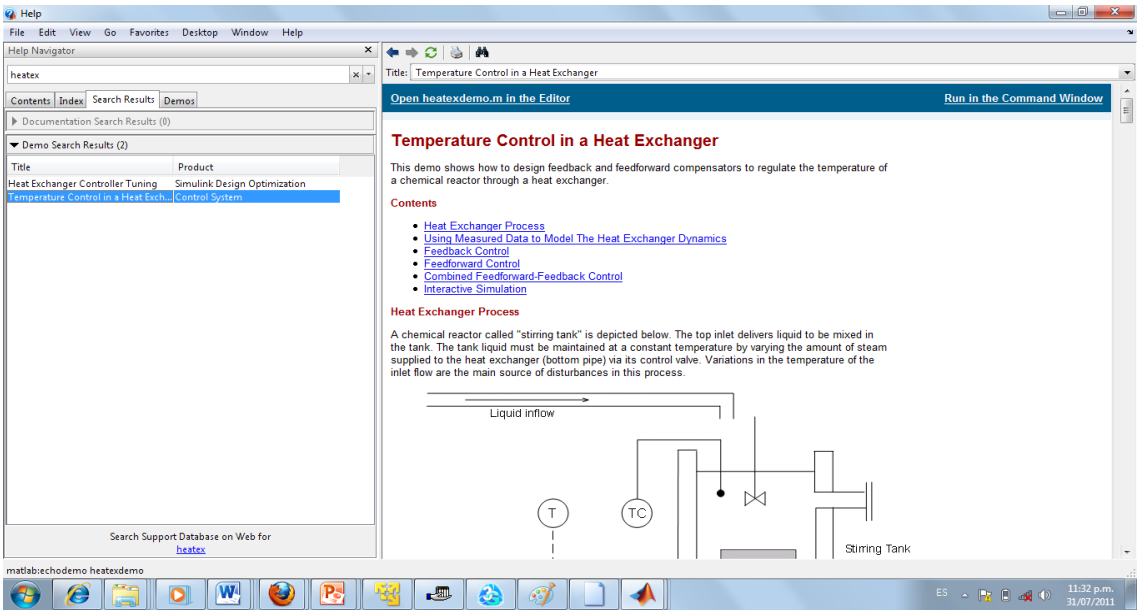
Ilustración I-7: Pantalla que permite establecer y visualizar parámetros para el control de temperatura en Matlab.



Fuente: Matlab

Para familiarizarse con esta planta se debe seguir la guía presente en la ayuda de Matlab. En el buscador de la ayuda, se debe digitar **heatex**, ejecutar el programa como lo indica la ilustración I-8.

Ilustración I-8: ayuda de Matlab donde presenta el funcionamiento del control de temperatura para un intercambiador de calor.



Fuente: Matlab

XII. CONFIGURACION DEL LOS OBJETOS ÍTEMS EN EL SERVIDOR OPC XML

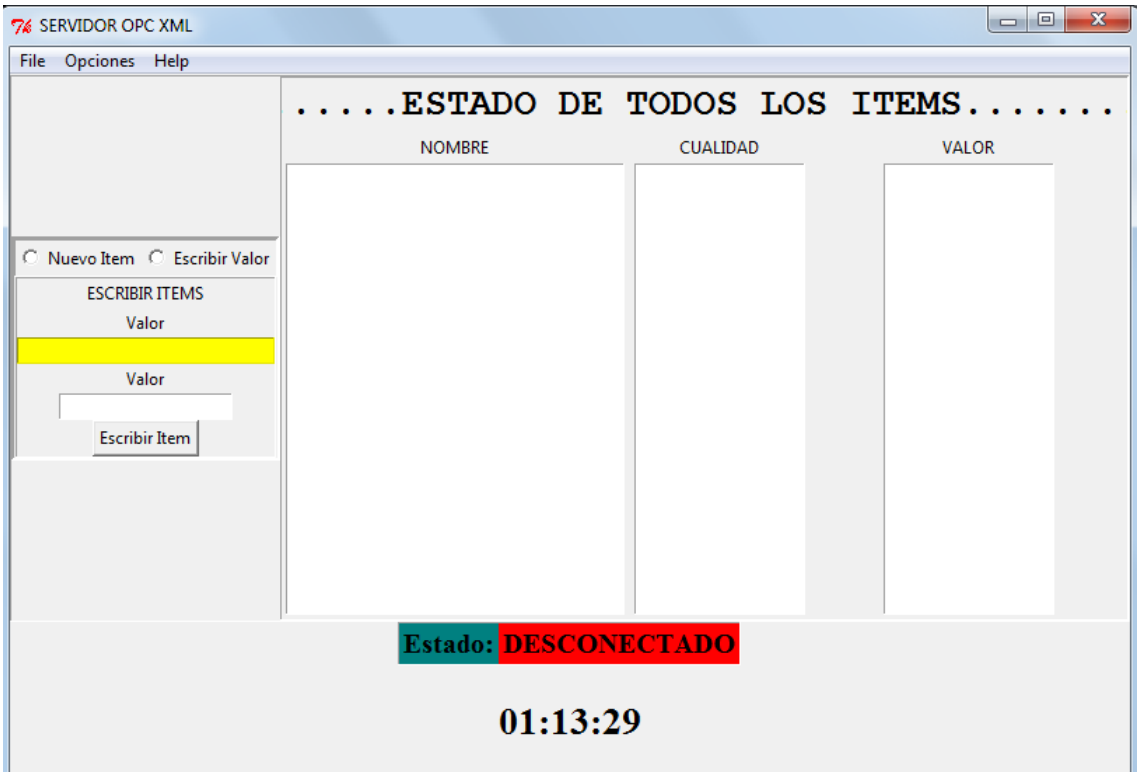
Después de la familiarización con las plantas de Nivel y temperatura, se inicia el proceso de interconexión de las plantas, mediante el estándar OPC XML.

Se trabaja con un MSOX instalado en la máquina virtual **XpOPC** de la planta de SED, en este servidor se configuran los objetos ítems tanto para Matlab como para Rtai-Lab.

1. Ejecutar el MSOX

Para ejecutar el MSOX, se debe ubicar en el computador de SEDS, ejecutar la máquina virtual **Winopc,** abrir la carpeta “SERVIDOR_XML” y ejecutar el archivo inicio.py. Debe presentarse una pantalla como la que se indica en la ilustración I-7.

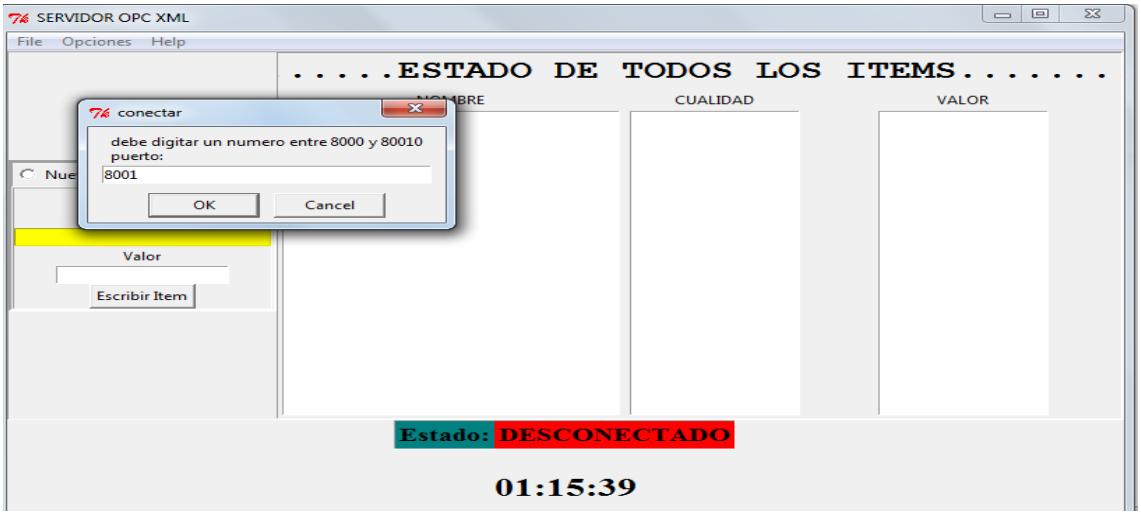
Ilustración I-9: Pantalla de inicio del servidor OPC XML.



Fuente: **Propia**

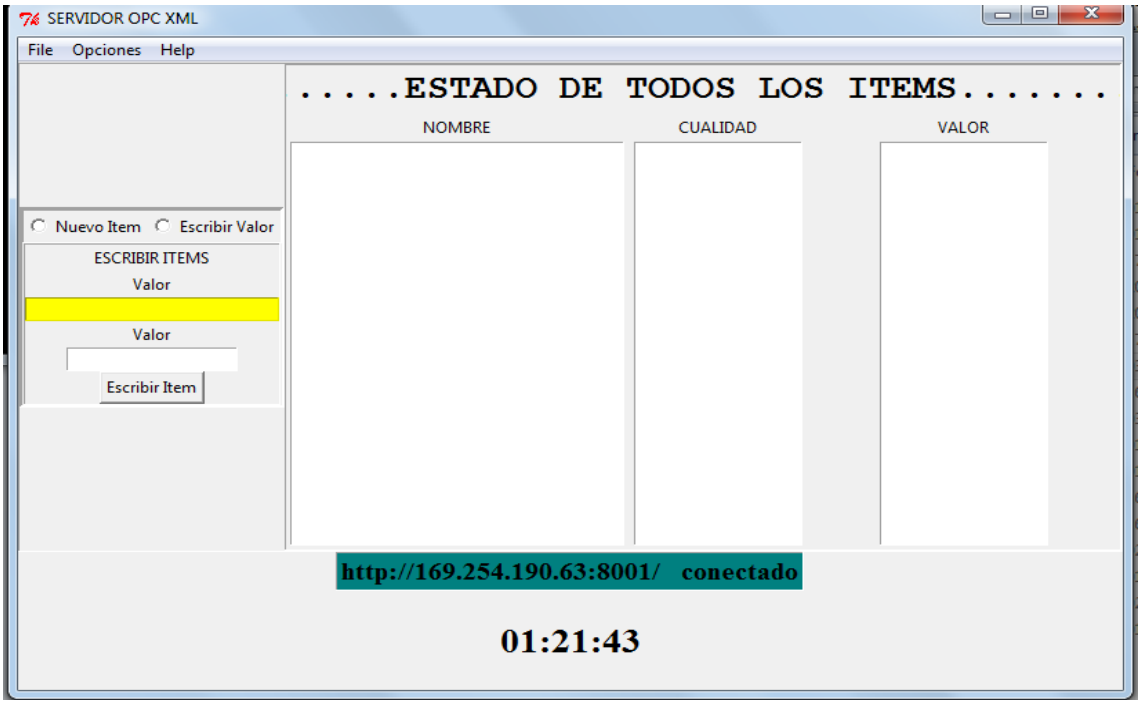
El paso siguiente es conectar el servidor, para ello se debe ir a **opciones/conectar,** y especificar el puerto 8001, ver ilustración I-10, I-11.

Ilustración I-10: Formulario de conexión del servidor OPC XML.



Fuente: Propia

Ilustración I-11: Servidor OPC XML conectado.

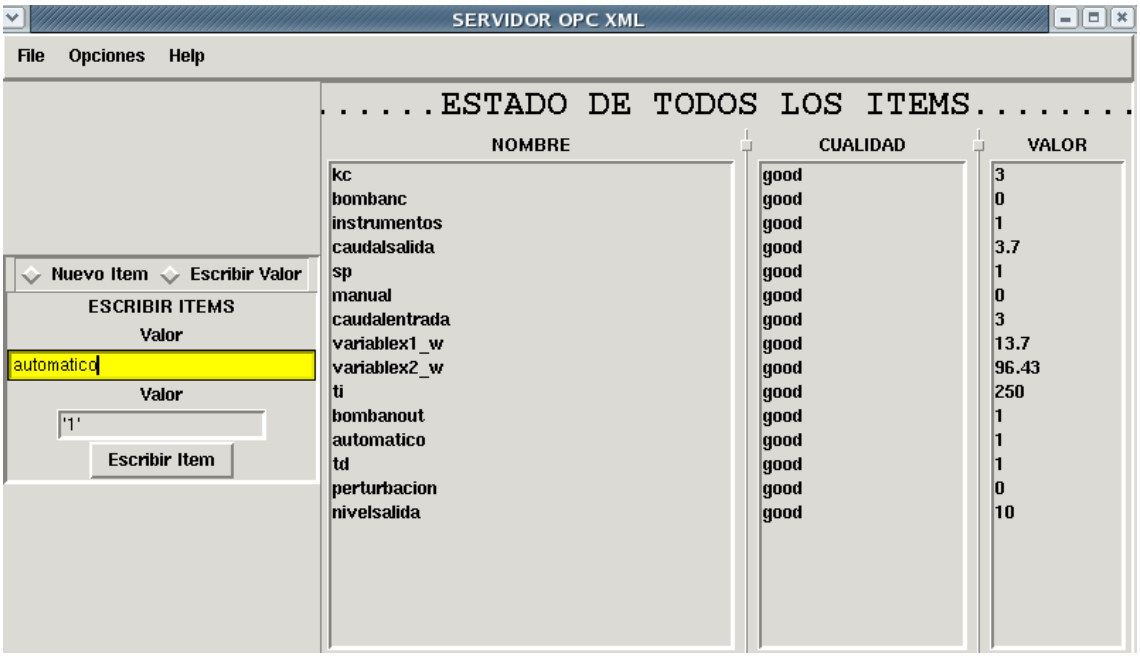


Fuente: Propia

2. **Configurar tags**

Para verificar las tags conectadas ir a **file/open** y abrir el archivo **tags** existente en la carpeta Servidor_XML, debe aparecer una pantalla con todas las variables configuradas para Matlab y Rtai-Lab, ver ilustración I-12. En caso de que no se encuentre el archivo se deben crear todas las variables.

Ilustración I-12: Configuración de objetos ítems en servidor OPC XML.



Fuente: Propia

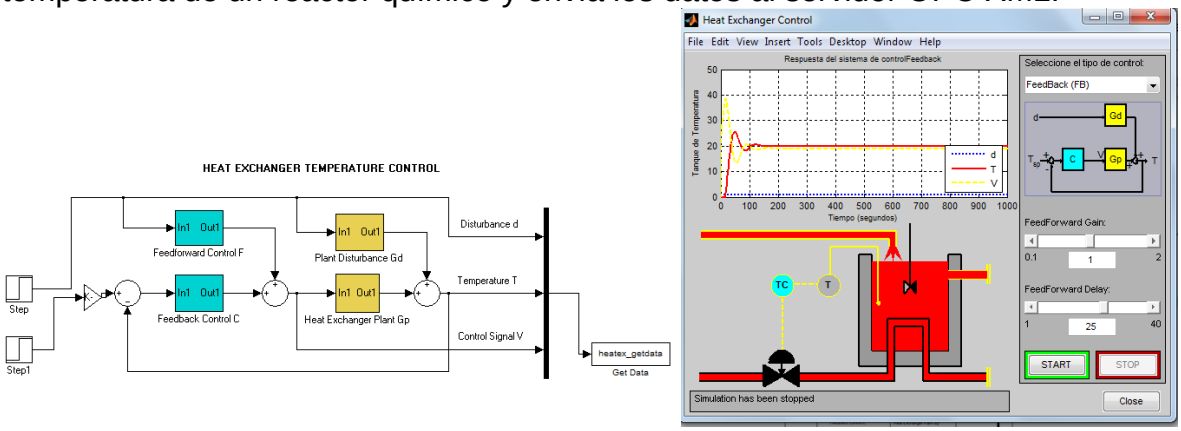
XIII. CONFIGURACIÓN DE LOS CLIENTES OPC XML EN LAS PLATAFORMAS DE CONTROL

1. Configuración del Módulo Cliente OPC XML en Matlab

1.1. Cargar archivos en Matlab

En la carpeta D/ctroldemo1 del computador de la planta de presión ejecutar el archivo **heatex.m.** Debe aparecer un diagrama en simulink como el que se presenta en la ilustración I-13.

Ilustración I-13: Diagrama de bloques en simulink que permite el control de temperatura de un reactor químico y envía los datos al servidor OPC XML.

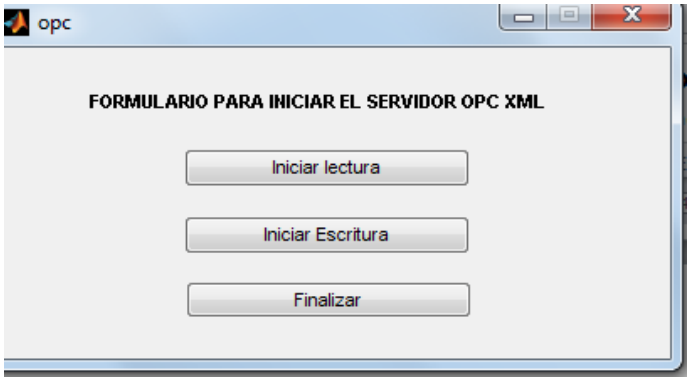


Fuente: Propia

1.2. Iniciar la lectura/escritura OPC

Abrir el archivo **opc.m,** ejecutarlo para que se presente un formulario que permite el inicio de la lectura/escritura.

Ilustración I-14: Formulario que permite el inicio de la lectura/escritura de datos desde Matlab/servidor OPC XML.

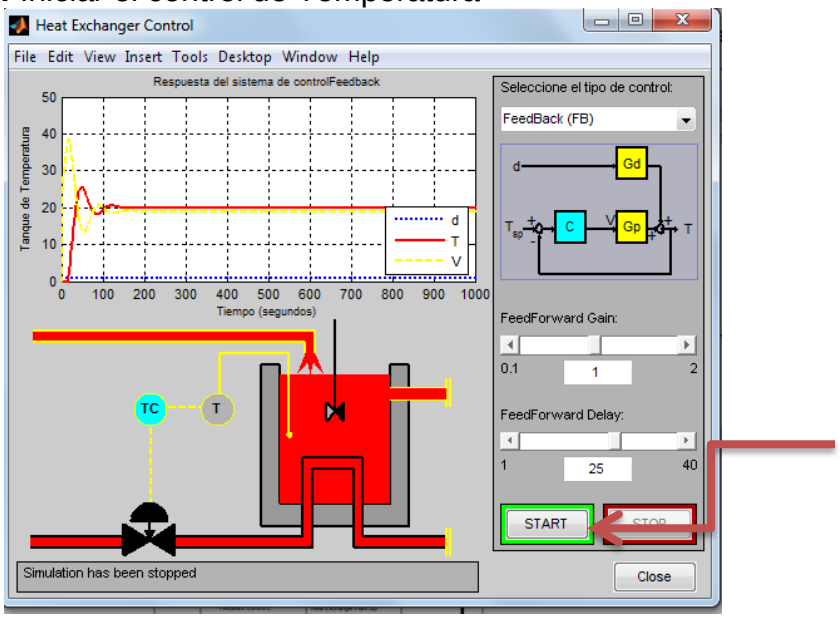


Fuente: Propia

1.3. Iniciar el control de temperatura.

El siguiente paso es presionar el botón Start del panel de control en Matlab, ver ilustración I-15.

Ilustración I-15: Iniciar el control de Temperatura

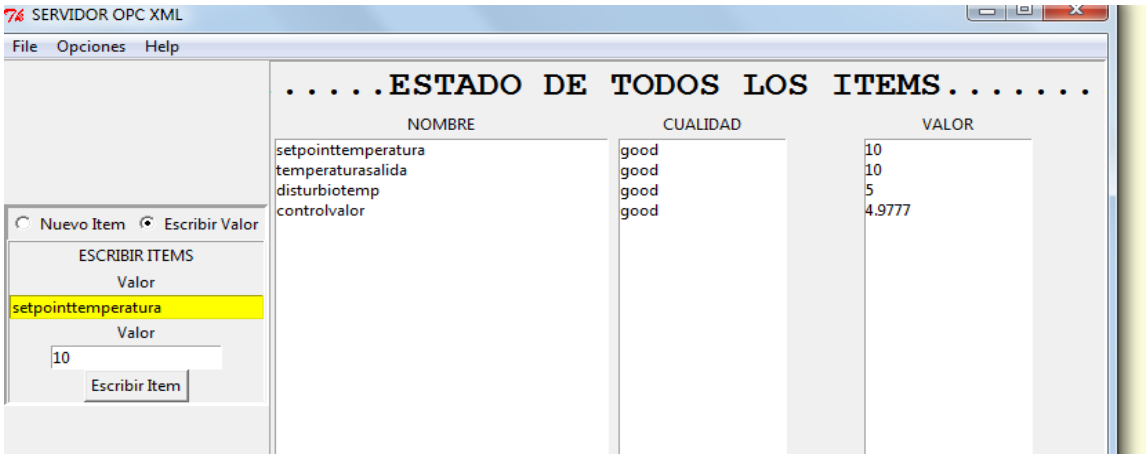


Fuente: Propia

1.4. Verificar conexión

Se debe verificar en el servidor OPC XML que se esté efectuando la lectura y escritura, ver ilustración I-16 (ubicarse en el servidor y cambiar el valor de la variable **setpointtemperatura,** ver el cambio en **temperatura salida**).

Ilustración I-16: pantalla del servidor OPC XML donde se verifica la lectura y escritura de datos.

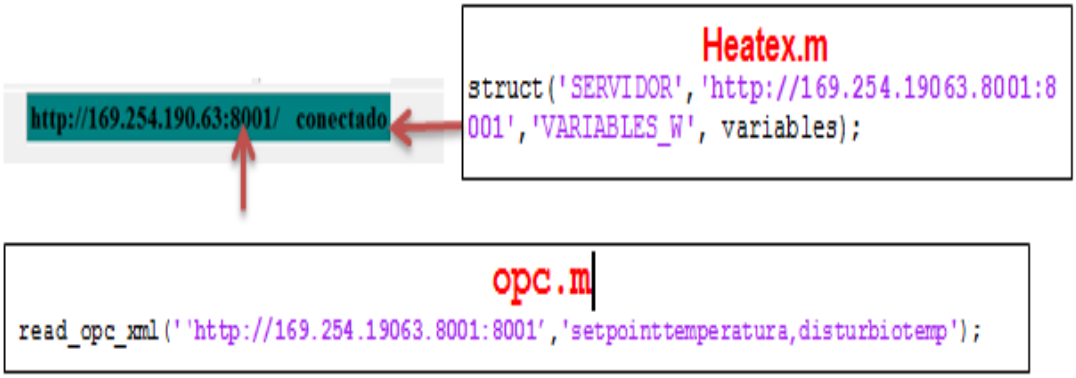


Fuente: Propia

1.5. Posibles fallas

Si la comunicación no es satisfactoria, se debe verificar que la IP donde está instalado el MSOX, sea la misma que la que tiene el cliente en Matlab. La ilustración presenta un ejemplo de configuración de IP, en este caso la ip en el servidor es **169.254.190.8001,** esta dirección debe coincidir con el especificado en la línea **748** del archivo heatex.m y con la línea **90** del archivo OPC, ver ilustración I-17.

Ilustración I-17: Configuración de la IP en el MCOXM



Fuente: Propia

Además se debe verificar que el nombre de las variables escritas en la línea **90** del archivo **opc.m** coincida con las creadas en el MSOX.

2. Configuración del Módulo cliente OPC XML en Rtai-Lab

A continuación se presentan los pasos necesarios para la configuración del cliente OPC XML en Rtai-Lab:

2.1. Encender el computador

- Ubicarse el computador de la planta de nivel, encenderlo, ingresar por Rtai-lab.

2.2. Cargar módulos:

Para implementar y ejecutar tareas en Rtai-Lab se deben cargar los módulos que contienen las librerías necesarias de Rtai, efectuando los siguientes pasos:

- Abrir la ventana “Terminal” en la barra de herramientas que aparece en la parte

inferior del escritorio, ver ilustración I-18, y digitar la siguiente instrucción:

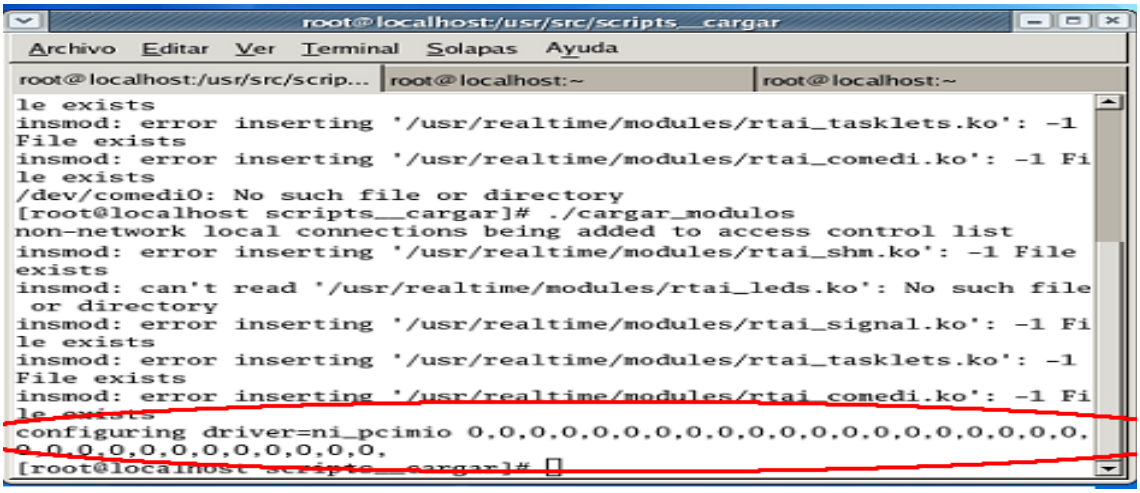
cd /usr/src/scripts__cargar/ (enter)
./cargar_módulos

Ilustración I-18: Apertura de ventana Terminal de Comandos.

Fuente: propia

Esta última instrucción se debe repetir hasta que aparezca en pantalla una fila de ceros 0,0,0…..,0,0 para que carguen los módulos completamente, ver ilustración I-19.

Ilustración I-19: Ventana que indica que los módulos han sido cargados.



Fuente: propia

2.3. Verificar funcionamiento de red y de servidor

Antes de ejecutar la tarea se debe verificar que el servidor instalado en la máquina virtual XpOPC ubicado en la planta de SED, tenga la misma dirección asignada inicialmente (**192.168.122.7**). Si no la tiene, en lo posible asignarle esta IP de lo

contrario se debe modificar el diagrama de bloques en Scicos y esto lleva más trabajo.

Para verificar la conexión OPC XML desde Linux con el servidor instalado en la máquina virtual de SED se debe de abrir un terminal y digitar las siguientes instrucciones:

cd /home/grupoati

python pruebaopcxml.py IP

Al presionar *enter*, debe aparecer el estado de este servidor, si el estado es conectado esto indica que se puede pasar automáticamente al paso 2.5 (ejecutar tarea).

2.4. Configurar los bloques OPC

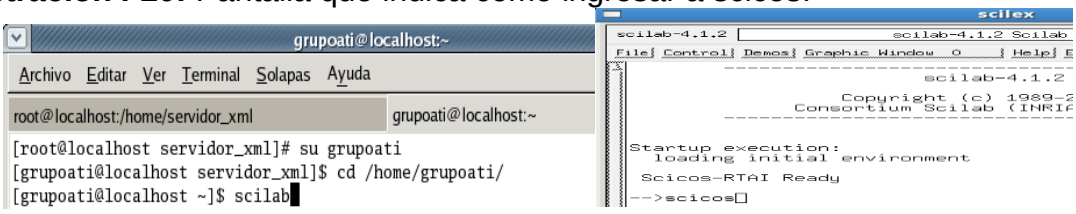
Si la configuración no es exitosa, se debe:

- Verificar que el computador de la planta de nivel esté conectado a la red y que la red esta activada
- Verificar que el servidor OPC XML instalado en el equipo de SEDS esté conectado.

Reconfigurar variables en Scicos

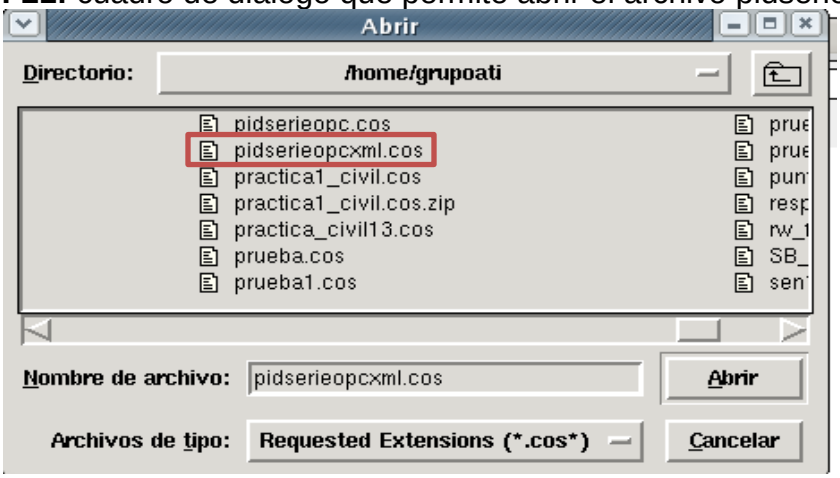
En el caso de que la IP del servidor sea diferente a **192.168.122.7** se debe ingresar a Scilab/scicos, ver ilustración I-20, abrir el archivo **pidserieopcxml.cos,** ver ilustración I-21 y configurar los parámetros según la sección III del anexo E (Guía de instalación y configuración de los clientes OPC DCOM/XML en Rtai-Lab).

Ilustración I-20: Pantalla que indica como ingresar a scicos.



Fuente: propia

Ilustración I-21: cuadro de dialogo que permite abrir el archivo pidserieopcxml



Fuente: propia

Hacer doble clic en el súper bloque, ver ilustración I-22.

Ilustración I-22: Súper bloque en Scicos

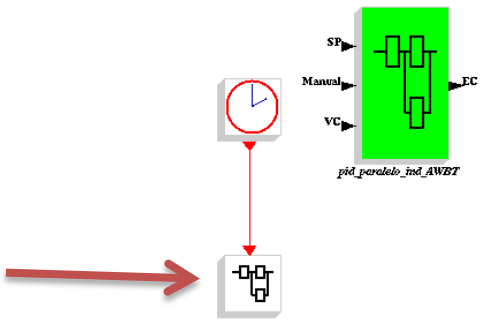
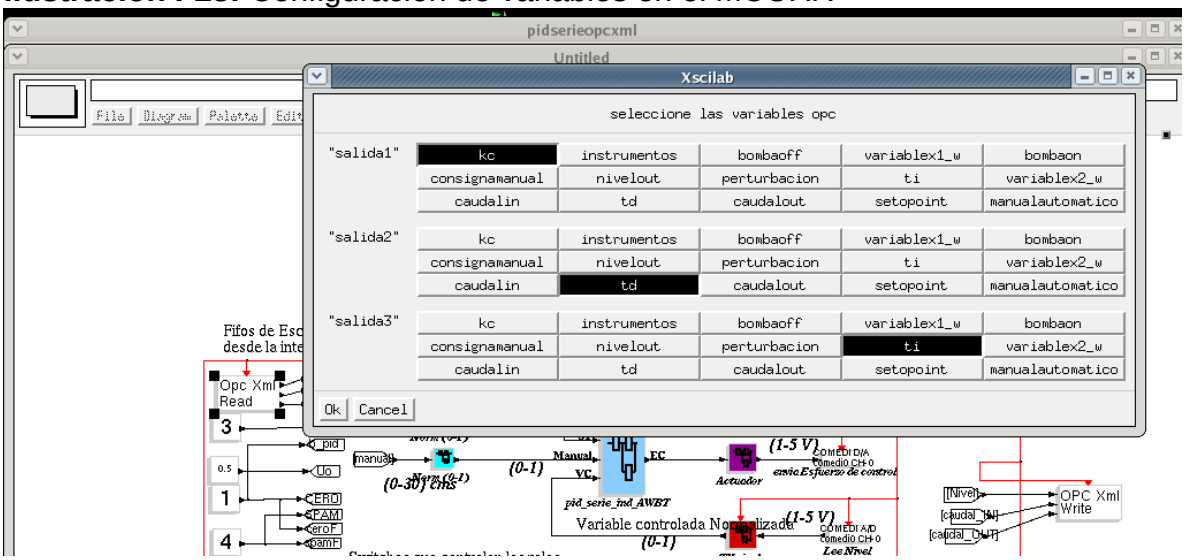


Diagrama para crear la tarea de tiempor real para el PID serie y enviar datos al servidor opc xml.

Fuente: propia

Hacer clic en cada bloque y configurar las variables, ver ilustración I-23.

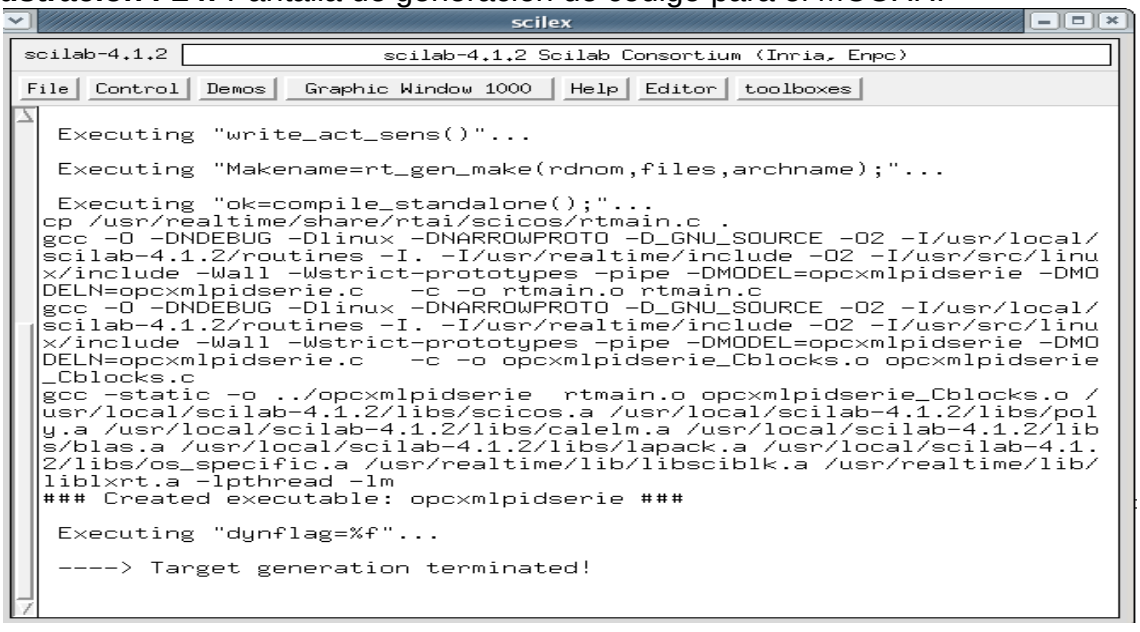
Ilustración I-23: Configuración de variables en el MCOXR



Fuente: propia

Ejecutar la tarea: ir a Rtai/codegen, asignar un nombre **opcxmlpidserie,** y ok, debe aparecer una pantalla como la que se indica en la ilustración I-24.

Ilustración I-24: Pantalla de generación de código para el MCOXR.



2.5. Ejecutar la tarea

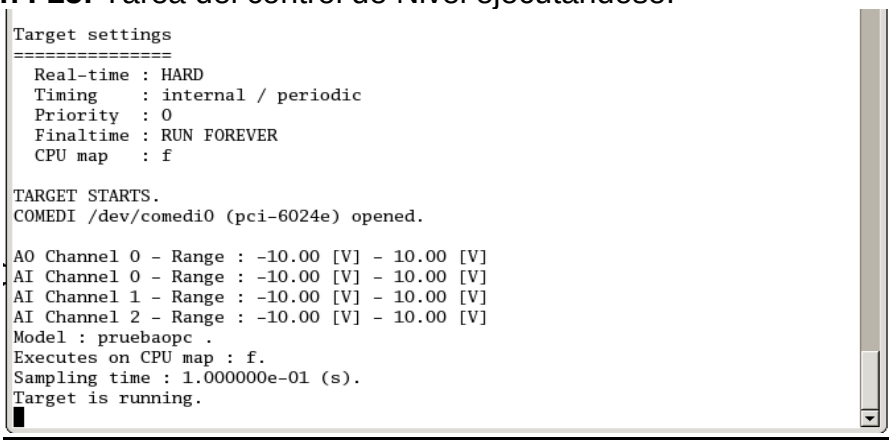
Antes de ejecutar la tarea se debe configurar los parámetros de las variables en el servidor KepserverEX, se deben ingresar los parámetros Kp, Ti, Td, y verificar que la instrumentación como la bomba estén apagados.

Si la comunicación OPC DCOM entre Fedora y el servidor OPC KepsereverEX se ha realizado correctamente, se puede iniciar la tarea de ejecución, se debe abrir un nuevo terminal y digitar las siguientes instrucciones (ver ilustración I-25):

cd /home/grupoati

./opcxmlpidserie –v

Ilustración I-25: Tarea del control de Nivel ejecutándose.



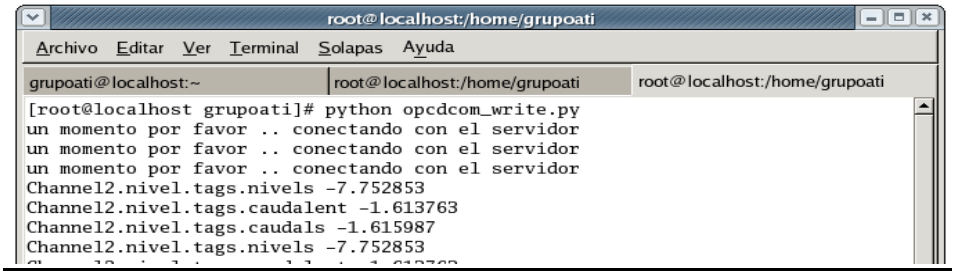
Fuente: propia

2.6. Inicio del cliente OPC en Rtai-lab

Abrir una nueva solapa, repetir este procedimiento dos veces, y en la primera solapa digitar **python opcxml_read.py,** ver ilustración I-26. En la segunda solapa digitar **python opcxml_write.py**, ver ilustración I-27.

Ilustración I-26: Código en el terminal para iniciar la lectura en scicos desde el MSOX

Ilustración I-27: Código en el terminal para iniciar la lectura en scicos desde el MSOX



Fuente: propia

- En el MSOX fijar los parámetros del controlador, fijar el setpoint.
- En el MSOX poner en uno la variable instrumentación, verificar que el transmisor, motobomba y demás instrumentos se encuentren encendidos.
- En el MSOX, poner en uno la variable bombaon y cambiar nuevamente este valor a cero, verificar que la bomba se encienda y que la variable **nivels** siga la variable Setpoint.

XIV. HMI PARA LAS PLANTAS DE PRESION, NIVEL Y TEMPERATURA

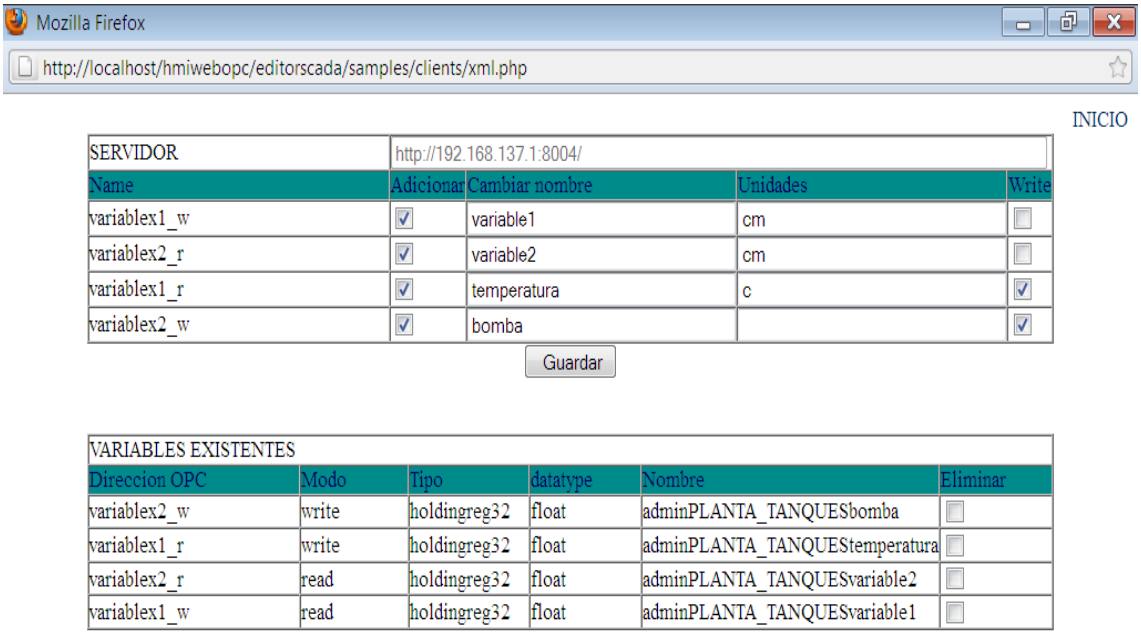
Luego de que se haya establecido la comunicación entre Matlab/Rtai-Lab y los servidores OPC XML, se procede a diseñar y monitorear las plantas.

Al hacer clic “editar/fuente de datos/XML”, se presenta un formulario donde se solicita el nombre del servidor OPC XML, cuando el usuario digita el nombre del servidor, ver ilustración I-28, se presenta otro formulario que permite importar las los objetos ítems creados en dicho servidor para que se almacenen en la base de datos, , ver ilustración I-29, a cada variable se le debe establecer un nuevo nombre, las unidades y el acceso (Lectura o escritura), cuando las variables estén listan se puede guardar para su posterior uso.

Ilustración I-28: Formulario permite ingresar la dirección del servidor OPC XML.

Fuente: propia

Ilustración I-29: Formulario que permite seleccionar los objetos ítems presentes en el servidor OPC XML.



Fuente: propia

El diseño y monitoreo de las plantas se lo efectúa igual que el expuesto en el anexo H numeral VII (Guía de integración de Matlab y Rtai-Lab mediante el estándar OPC DCOM).

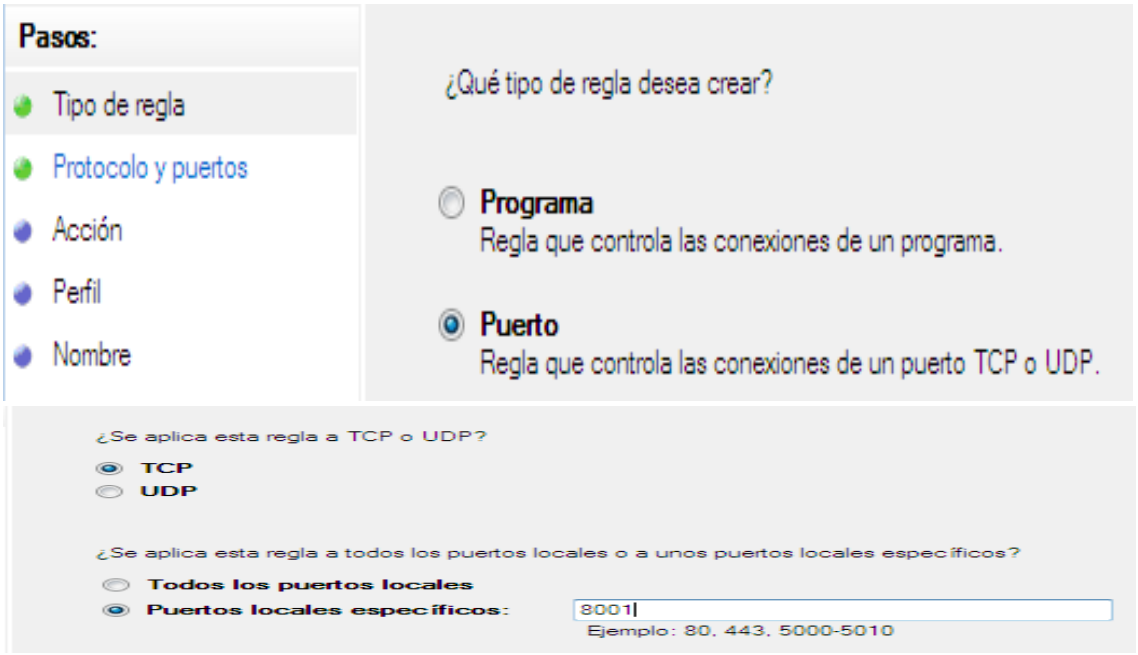
ANEXO J. GUÍA PARA ACTIVAR LOS PUERTOS Y CONFIGURACIÓN DEL COMPONENTE DE WINDOWS DCOM

I. PASOS PARA ACTIVAR PUERTOS TCP

1. Activar los puertos en Windows 7

- Abrir Firewall de Windows desde: Panel de control\Sistema y seguridad
- En el panel izquierdo, haga clic en Configuración avanzada. Si se le solicita una contraseña de administrador o una confirmación, escriba la contraseña o proporcione la confirmación.
- En el cuadro de diálogo Firewall de Windows con seguridad avanzada, en el panel de la izquierda, haga clic en Reglas de entrada y, en el panel de la derecha, haga clic en Nueva regla.
- Siga las instrucciones del Asistente para nueva regla de entrada, ver ilustración J-1.

Ilustración J-1. Pasos para configurar los puertos en Windows 7

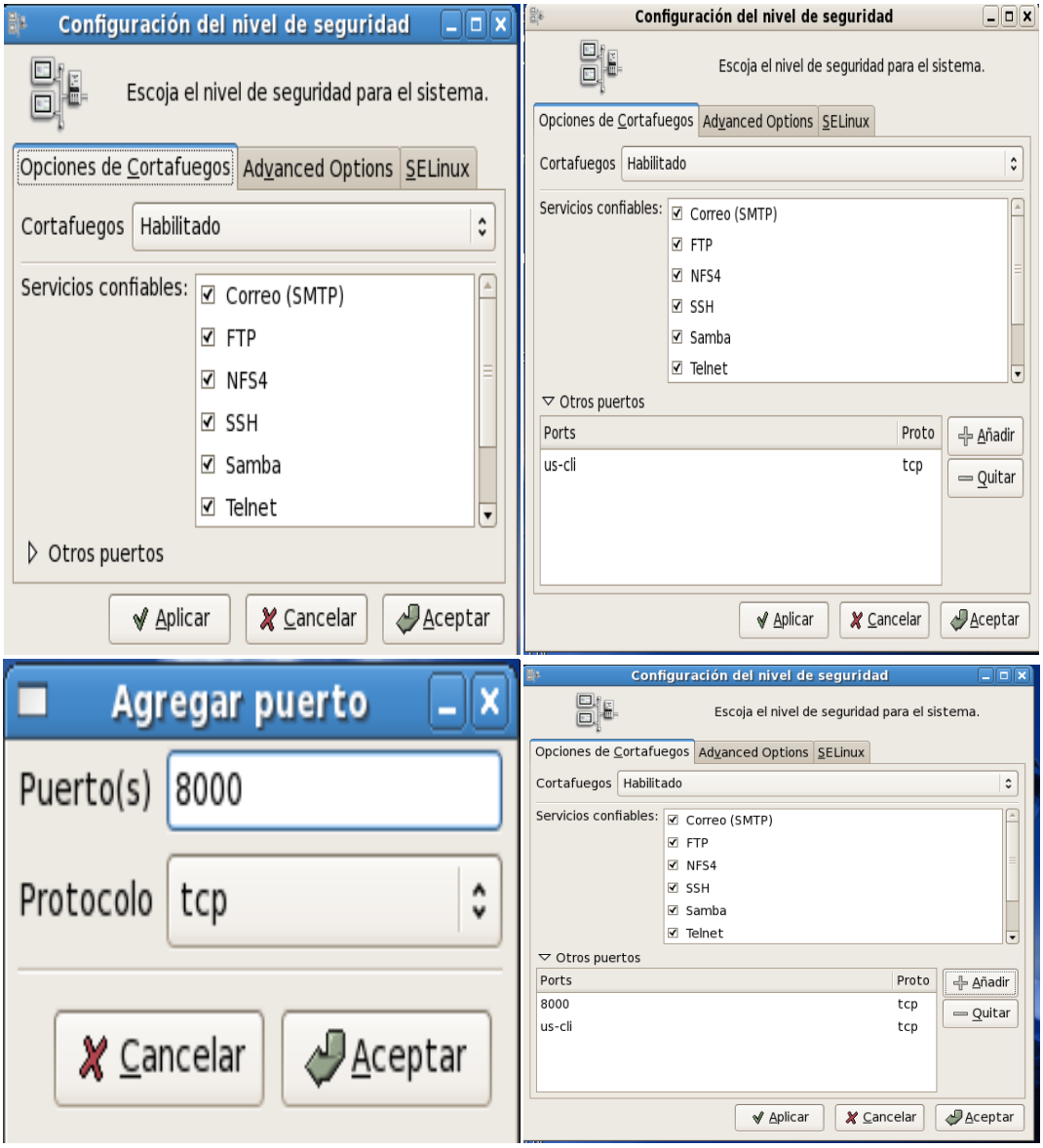


Fuente: propia

2. Activar puertos en Linux

Para activar los puertos en Linux (Fedora) se deben seguir los siguientes pasos: ir al menú del escritorio "sistema"-"configuracion"-"Opciones de cortafuegos y SELinux", seleccionar otros puertos en el formulario que se presenta, ver ilustración J-2, hacer clic en "añadir", escribir el puerto que se desea activar.

Ilustración J-2. Pasos para configurar los puertos en Fedora



Fuente: propia

II. CONFIGURACIÓN DEL COMPONENTE DE WINDOWS DCOM

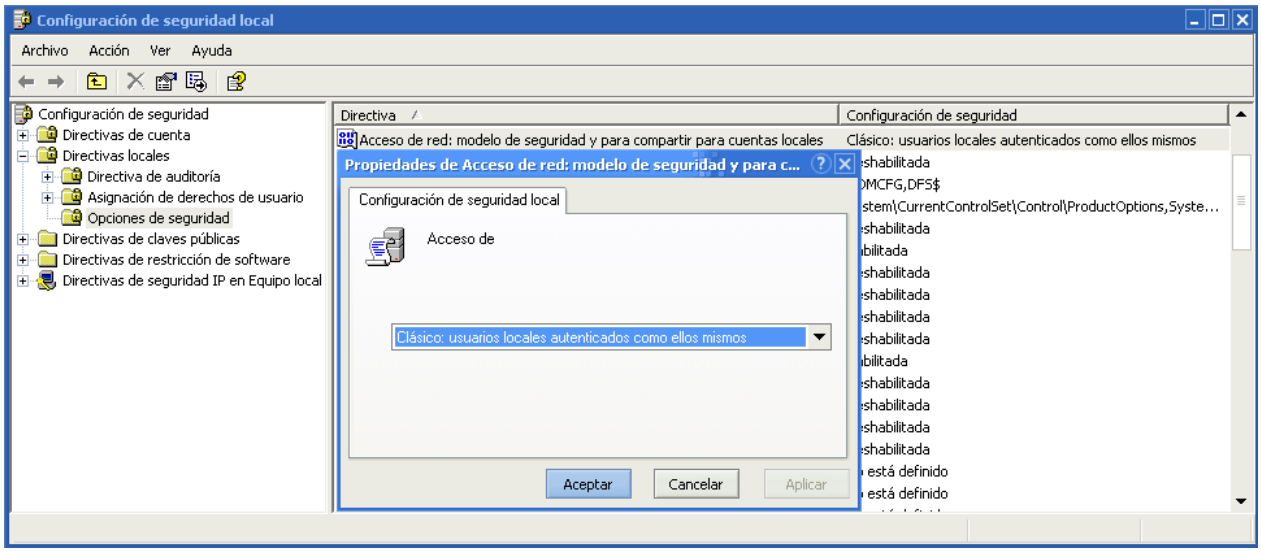
Debido a que la tecnología OPC está basada en COM y DCOM de Microsoft® para el intercambio de datos, es necesario configurar el componente distribuido de Windows® DCOM, cuando se requiere una conexión remota entre un cliente y un servidor OPC. Para esto se siguen los siguientes pasos:

Desactivar la seguridad de Windows: El primer paso es desactivar el firewall de Windows® y cualquier otro software que restrinja el acceso al computador.

Deshabilitar "Uso compartido simple de archivos" en Windows: Se debe configurar las políticas de seguridad locales del sistema operativo, para esto se utilizan las herramientas administrativas en el Panel de Control o ejecutando el comando "secpol.msc" en el menú inicio de Windows.

En el árbol de consola, localizar la carpeta opciones de seguridad en la configuración de seguridad, políticas locales, para encontrar "Acceso de red: modelo de seguridad y para compartir para cuentas locales" en donde se selecciona la configuración "Clásico: usuarios locales autenticados como ellos mismos", ver ilustración J-3.

Ilustración J-3. Configuración de seguridad local



Fuente: Propia.

Configurar las opciones del sistema DCOM: Para lograr esto es necesario: Ejecutar el comando “DCOMCNFG” en el menú de inicio de Windows® para tener acceso a la ventana “Servicios de Componentes”, en donde se localiza la carpeta de computadores dentro de la carpeta de servicios de componentes, haciendo clic derecho sobre “Mi PC” se puede ver la opción propiedades, ver ilustración J-4.

Ilustración J-4. Ventana de servicios de componentes

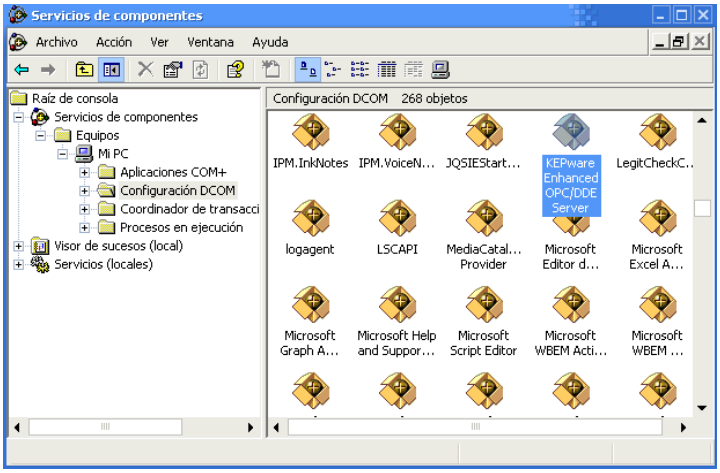
Fuente: Propia.

En la pestaña “Propiedades predeterminadas” es necesarios realizar los siguientes cambios:

- Habilitar la opción “Habilitar COM distribuido en este equipo”
- En el nivel de autenticación predeterminado, seleccionar “conectar”
- En el nivel de suplantación predeterminado, seleccionar “Identificar”
- En la pestaña “Protocolos predeterminados”, seleccionar “TCP/IP orientado a la conexión” y eliminar el resto de los protocolos DCOM, que existan si no se necesitan para otras aplicaciones.
- En la pestaña “Seguridad COM” se adicionan los permisos para habilitar el acceso que permite iniciar una aplicación. Para esto, en la sección “permisos de acceso”, en el botón “Editar valores predeterminados” se agrega a la lista de “Nombres de grupos o usuarios” el usuario “Todos” con “Acceso local” y “Acceso remoto” y en el botón “Editar límites” se agregan a la lista de “Nombres de grupos o usuarios” los usuarios “Todos” y “ANONYMOS LOGON”, también con “Acceso local” y “Acceso remoto”. Lo mismo se hace en la sección “Permisos de inicio y activación”, luego se hace clic en Aceptar.

Configurar el servidor específico DCOM: En la ventana "Servicios de componentes" se navega en el árbol de consola hasta el icono "Mi PC" y en la carpeta "Configuración DCOM" (ilustración J-5) se busca el objeto que representa el servidor OPC para seleccionar en el menú propiedades, en la pestaña "Identidad" la opción "El usuario interactivo".

Ilustración J-5. Carpeta de configuración DCOM



Fuente: Propia.

Restablecer la seguridad de Windows: Cuando se establece la comunicación Cliente/Servidor OPC, es importante restablecer la seguridad de Windows®, así:

- Activar el Firewall de Windows® y agregar excepciones en dos niveles distintos:
- Nivel de aplicación: especificar las aplicaciones que necesitan la comunicación OPC
- Nivel de puerto y protocolo: habilitar el puerto específico para el tráfico TCP o UDP.
- Modificar las listas de control de acceso para permitir o negar las cuentas de usuario que requieren la comunicación OPC, sin quitar el acceso a la cuenta "ANONYMOUS LOGON" ya que la necesita OPCEnum.

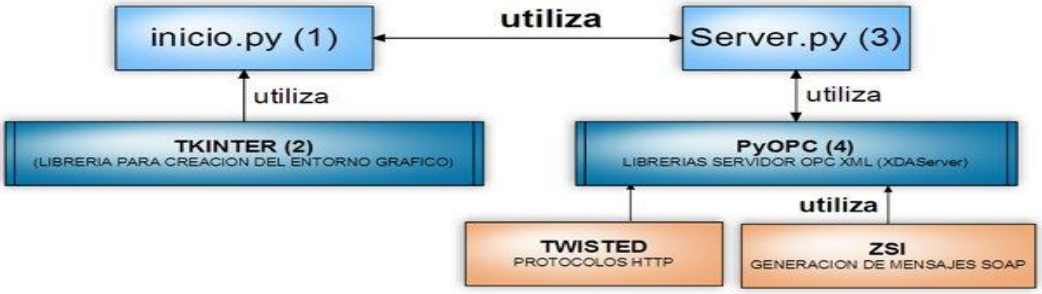
Con esto queda configurado el componente DCOM Windows®, permitiendo la comunicación remota por medio de OPC entre servidores y clientes a través de Ethernet.

ANEXO K. ESQUEMA DE ARCHIVOS Y CODIGO FUENTE DE LOS MODULOS IMPLEMENTADOS

I. MODULO SERVIDOR OPC XML

Para la implementación del Servidor OPC XML se crearon dos archivos: uno denominado “inicio.py”, ver figura K-1, recuadro 1, donde se encuentran las funciones de interfaz gráfica, el cual hace uso de la librería TKINKER (recuadro 2), y el otro denominado “Server.py” (recuadro 3) en el cual se implementan las clases para el funcionamiento interno del Servidor OPC XML, haciendo uso de las librerías PYOPC (recuadro 4), TWISTED Y ZSI.

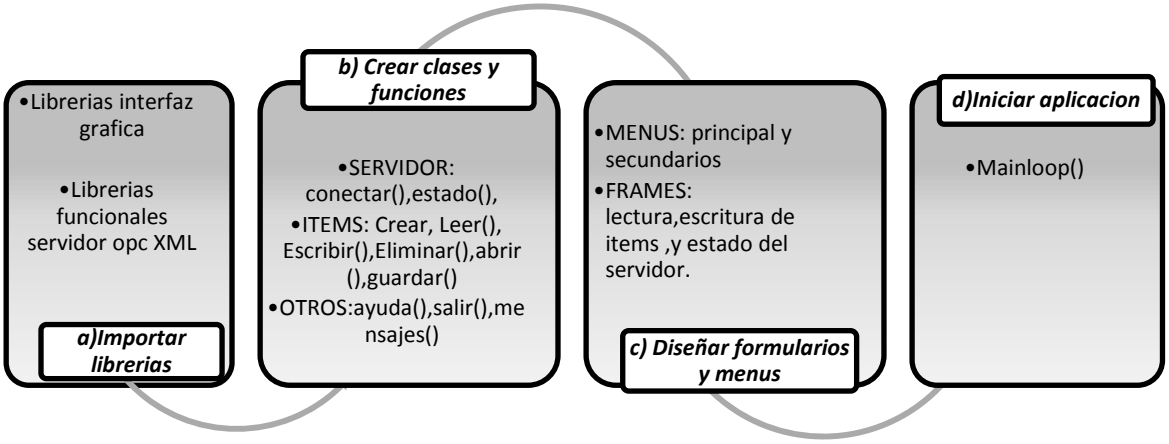
Figura K-1: Diagrama de archivos del Servidor OPC XML.



Fuente: propia.

1. Archivo inicio.py

Figura K-2: Pasos para construir el archivo inicio.py



Fuente: propia.

A continuación se presenta el código para el archivo inicio.py

a) Importar las librerías

inicio.py - E:\Junio\servidor_xml\inicio.py

File Edit Format Run Options Windows Help

```
# -*- coding: cp1252 -*-
#LIBRERIAS NECESARIAS PARA LA INTERFAZ GRAFICA
from Tkinter import *
from tkSimpleDialog import  askstring
import Tkinter as tk
from tkFileDialog   import askopenfilename,asksaveasfile
from tkFont import Font

# LIBRERIAS NECESARIAS PARA SERVIDOR OPC XML
from twisted.internet import reactor,defer,tksupport
from PyOPC.servers.basic import BasicXDAServer
import socket
import server
from PyOPC.OPCContainers import *
from PyOPC.TWXDAClient import TWXDAClient
OPERATIONS = 3
#SE CAPTURA LA IP DEL EQUIPO PARA PRESENTARLA EN PANTALLA
ip=socket.gethostbyname(socket.gethostname())
SERVIDOR_IM=server.SERVIDOR()

#FORMULARIO GENERAL
root = tk.Tk()
root.geometry("700x500")
root.title("SERVIDOR OPC XML")
tksupport.install(root)
```

b) Crear clases y funciones

inicio.py - E:\Junio\servidor_xml\inicio.py

File Edit Format Run Options Windows Help

```
# CLASE SERVIDOR .... IMPLEMENTA LAS FUNCIONES QUE PERMITEN VISUALIZAR TODO LO CORRESPONDIENTE A SERVI
class SERVIDOR:
    ####INICIA EL SERVIDOR--- INICIA PIDIENDO EL PUERTO
    def conf_pto(self):
        global puerto,estado,puerto1
        try:
            if(puerto==''):
                puerto= askstring("conectar", " debe digitar un numero entre 8000 y 80010 \n puerto:")
                f = Font(font=("Courier",15, "bold"))
        except:
            puerto= askstring("conectar", " debe digitar un numero entre 8000 y 80010 \n puerto:")
            f = Font(font=("Courier",15, "bold"))
        try:
            print puerto
            a=SERVIDOR_IM.conectar(puerto)
            print a
            if(a=='2'):
                OTROS().alert('el puerto esta en uso o el servidor esta ocupado')
                puerto=''
        except:
            print 'servidor s'
    def configuraciones(self):
        conf = tk.Tk()
        conf.geometry("100x100")
        conf.title("SERVIDOR OPC XML")
        VendorInfo="SERVIDOR OPC XML PARA LAS PLATAFORMAS DE CONTROL"
        ProductVersion="1.0"
        ###ActivarO Desactivar almacenamiento en cachE automAtico DE LOS ITEMS
    def verregistro(self):
        conf = tk.Tk()
        conf.geometry("600x400")
        conf.title("LISTADO DE ACCESOS AL SERVIDOR OPC XML")
        f1 = open("access1.log", 'r')
        scrollbar = Scrollbar(conf, orient=VERTICAL)
        valorreg= Listbox(conf,height=100,width=100)
        scrollbar.config(command=valorreg.yview)
        valorreg.config(yscrollcommand=scrollbar.set)
        scrollbar.pack(side=RIGHT, fill=Y)
        valorreg.pack(side=LEFT, expand=YES, fill=BOTH)
        try:
            cont=0
            for line in f1:
                cont =cont +1
                valores=line.split(' ')
                if(valores[3]!=ip):
                    valorreg.insert(cont,line)
        finally:
            f1.close()
    def vererrores(self):
        import os
        os.startfile("error.log")
    def vercodigoxml(self):
        import os
        os.startfile("http.log")
```

Clase ítem

inicio.py - E:\Junio\servidor_xml\inicio.py

File Edit Format Run Options Windows Help

```
#### TODO LO REFERENTE A LOS ITEMS
class ITEMS:
    # funcion para abrir un archivo con items
    def Open(self):
        try:

            # Get the name of the input file
            file_input_name = askopenfilename(parent=root, title='Open file', filetypes=[('py files', '*.py')]
            f = open(file_input_name, 'r')
            valor = f.read()

            address='http://' + str(ip) + ':' + str(puerto) + '/'
            xda = TWXDAClient(OPCServerAddress=address,  ReturnErrorText=True)
            d = xda.twWrite(eval(valor))
            d.addCallback(print_options)
            d.addErrback(handleError)
        except:
            OTROS().alert('debe primero iniciar el servidor')
    def Save(self):
        address='http://' + str(ip) + ':' + str(puerto) + '/'
        xda = TWXDAClient(OPCServerAddress=address,  ReturnErrorText=True)
        d = xda.twBrowse( ReturnAllProperties=False)
        d.addCallback(print_options1)
        d.addErrback(handleError)

    def Add_item(self):
        global tipo,nombre
        Label(otro,text='Nombre').pack(side=LEFT)
        nombre = Entry(otro)
        nombre.pack(side=LEFT)
        Label(otro,text='tipo').pack(side=LEFT)
        tipo = Entry(otro)
        tipo.pack(side=LEFT)
        remove_button = Button(otro, text="Adicionar", command = ITEMS().Add_itemf)
        remove_button.pack()
    def nuevo_it(self):
        frames.nuevo_item.pack()
        frames.otro2.pack_forget()
    def write_it(self):
        frames.nuevo_item.pack_forget()
        frames.otro2.pack()
```

```
def Add_itemf(self):
    add=''
    try:

        nombre1=frames.nombre.get()
        print nombre1
        tipo1=frames.variable.get()
        valor1=frames.valor.get()


        if(nombre1=='' or tipo1==''):
            OTROS().alert('los datos no estan completos')
        else:
            if(puerto==None):
                otros().alert('Debe inicializar el servidor')
            else:
                address='http://' + str(ip) + ':' + str(puerto) + '/'
                xda = TWXDAClient(OPCServerAddress=address,  ReturnErrorText=True)
                add='[ItemContainer(ItemName="' + nombre1 + '", Value='+ valor1 + ')]'

                d = xda.twWrite(eval(add))
                d.addCallback(print_options)
                d.addErrback(handleError)
                OTROS().alert('Item Agregada')
                nombre1=''
                tipo1=''
                d=''

    except:
        OTROS().alert('Debe inicializar el servidor-')
```

Clase otros

```
class OTROS:

    def alert(self,mensaje):
        root1=Tk()
        f = Font(font=("Courier", 20, "bold"))
        w = Label(root1, text=mensaje, font=f)
        w.pack()
        w = Button(root1, text="Quit!", command=root1.destroy)
        w.pack()

def print_options((ilist,options)):
    a='['
    #print ilist;
    for i in ilist:

        valor=i.ItemName
        a=a + 'ItemContainer(ItemName="' + str(valor) + '"),'
    a=a+ ']'
    address='http://' + str(ip) + ':' + str(puerto) + '/'
    xda = TWXDAClient(OPCServerAddress=address,  ReturnErrorText=True)
    d = xda.twRead(eval(a))
    d.addCallback(print_options3)
    d.addErrback(handleError)

def print_options1((ilist,options)):
    a='['
    #print ilist;
    for i in ilist:
        valor=i.ItemName
        a=a + 'ItemContainer(ItemName="' + str(valor) + '"),'
    a=a+ ']'
    address='http://' + str(ip) + ':' + str(puerto) + '/'
    xda = TWXDAClient(OPCServerAddress=address,  ReturnErrorText=True)
    d = xda.twRead(eval(a))
    d.addCallback(print_options2)
    d.addErrback(handleError)
    global OPERATIONS
    OPERATIONS -= 1
    if OPERATIONS == 0:
        reactor.stop()
```

```
def handleError(failure):
    print "An Error occured"
    print failure.getTraceback()
    reactor.stop()
def handleResult((ic,op)):
    print op
    print ic
    reactor.stop()

def handleError(failure):
    print "An Error occured"
    print failure.getTraceback()
    reactor.stop()

def items_read(puerto):
    print puerto


def iniciar_v():
    a=True
    while a:
        try:
            address='http://' + str(ip) + ':' + str(puerto) + '/'
            xda = TWXDAClient(OPCServerAddress=address,  ReturnErrorText=True)
            d = xda.twBrowse( ReturnAllProperties=False)
            d.addCallback(print_options1)
            d.addErrback(handleError)
        except:
            print 'no'
```

Código para crear los menús

```
########SE CREAN LOS MENUS
class menu():
    menubar = Menu(root)
    items1=ITEMS()
    # create a pulldown menu, and add it to the menu bar
    filemenu = Menu(menubar, tearoff=0)
    filemenu.add_command(label="Abrir", command=items1.Open)
    filemenu.add_command(label="Guardar", command=ITEMS().Save)
    filemenu.add_separator()
    filemenu.add_command(label="Exit", command=root.destroy)
    menubar.add_cascade(label="File", menu=filemenu)

    # create more pulldown menus
    editmenu = Menu(menubar, tearoff=0)
    editmenu.add_command(label="Conectar", command=SERVIDOR().conf_pto)
    editmenu.add_command(label="ver codigo XML", command=SERVIDOR().vercodigoxml)
    editmenu.add_command(label="ver registro de clientes", command=SERVIDOR().verregistro)
    editmenu.add_command(label="ver errores", command=SERVIDOR().vererrores)
    editmenu.add_command(label="configuraciones", command=SERVIDOR().configuraciones)
    menubar.add_cascade(label="Opciones", menu=editmenu)

    helpmenu = Menu(menubar, tearoff=0)
    helpmenu.add_command(label="About", command=hello)
    menubar.add_cascade(label="Help", menu=helpmenu)
    root.config(menu=menubar)
```

Código para crear los formularios

inicio.py - E:\Junio\servidor_xml\inicio.py

File Edit Format Run Options Windows Help

```
### SE CREAN LOS FRAMES

class frames():
    separator = Frame(width=800, height=800, bd=1, relief=SUNKEN)
    otro4=Frame(separator,width=800, height=500, bd=1, relief=SUNKEN)
    otro=Frame(otro4,width=800, height=500, bd=1, relief=SUNKEN)
    items_frame = Frame(separator,width=800, height=800, bd=1, relief=SUNKEN)
    otro1=Frame(root,width=200, height=100, bd=1, relief=SUNKEN)
    otro2=Frame(otro,width=800, height=900, bd=1, relief=SUNKEN)
    separator.pack()
    otro4.pack(side=LEFT)
    otro.pack(side=LEFT)
    items_frame.pack(side=LEFT)
    f = Font(font=("Courier", 20, "bold"))
    left = Label(items_frame, font =f, text="......ESTADO DE TODOS LOS ITEMS.....")
    left.pack()
    ##### PANELES MOVIBLES
    m1 = PanedWindow(items_frame)
    m1.pack(fill=BOTH, expand=1)

    #####NOMBRE
    m2 = PanedWindow(m1, orient=VERTICAL)
    m1.add(m2)
    left = Label(m2, text="NOMBRE")
    scrollbar = Scrollbar(m2, orient=VERTICAL)
    nombre_item= Listbox(m2, yscrollcommand=scrollbar.set,height=20,width=20)
    nombre_item.bind('<ButtonRelease-1>', hello)
    left.pack()
    nombre_item.pack()

    #####CUALIDAD
    m3 = PanedWindow(m1, orient=VERTICAL)
    m1.add(m3)
    left = Label(m3, text="CUALIDAD")
    scrollbar = Scrollbar(m3, orient=VERTICAL)
    cualidad = Listbox(m3, yscrollcommand=scrollbar.set,height=20,width=15)
    left.pack()
    cualidad.pack()
```

```
#####VALOR
m4 = PanedWindow(m1, orient=VERTICAL)
m1.add(m4)
left = Label(m4, text="VALOR")
scrollbar = Scrollbar(m4, orient=VERTICAL)
valor_item= Listbox(m4, yscrollcommand=scrollbar.set,height=20,width=15)
left.pack()
valor_item.pack()

#####TIPO
m5 = PanedWindow(m1, orient=VERTICAL)
m1.add(m5)
left = Label(m5, text="TIPO")
scrollbar = Scrollbar(m4, orient=VERTICAL)
tipo_item = Listbox(m5, yscrollcommand=scrollbar.set,height=20,width=15)
left.pack()
tipo_item.pack()
v = IntVar()

selecionar=Frame(otro,width=800, height=900, bd=1, relief=SUNKEN)
selecionar.pack()
Radiobutton(selecionar, text="Nuevo Item", variable=v, value=1, command=ITEMS().nuevo_it
).pack(side=LEFT)
Radiobutton(selecionar, text="Escribir Valor", variable=v, value=2,command=ITEMS().write_it).pack(side=LEFT)

nuevo_item=Frame(otro,width=800, height=900, bd=1, relief=SUNKEN)
Label(nuevo_item,text='ADICIONAR ITEMS').pack()
Label(nuevo_item,text='Nombre').pack()
nombre = Entry(nuevo_item)
nombre.pack()
valor1=Label(nuevo_item,text='Valor')
valor1.pack(fill=BOTH, expand=1)
valor = Entry(nuevo_item)
valor.pack()
variable = StringVar(nuevo_item)
variable.set("INTEGER") # default value
tipo = OptionMenu(nuevo_item, variable, "FLOAT", "STRING", "INTEGER")
tipo.pack()
remove_button = Button(nuevo_item, text="Adicionar", command = ITEMS().Add_itemf)
remove_button.pack()
```

```
    ####escribir items
    otro2.pack()
    Label(otro2,text='ESCRIBIR ITEMS').pack()
    item_w=Label(otro2,text='Valor')
    item_w.pack(fill=BOTH, expand=1)

    enter1 = tk.Entry(otro2, width=30, bg='yellow')
    enter1.pack()

    valor1_w=Label(otro2,text='Valor')
    valor1_w.pack(fill=BOTH, expand=1)
    valor_w = Entry(otro2)
    valor_w.pack()
    write_button = Button(otro2, text="Escribir Item", command = ITEMS().write_itemf)
    write_button.pack()
    #### estado del servidor
    variable33 = StringVar()
    variable33.set('Estado:')
    conectado1=Label(otro1,textvariable=variable33,font=('times', 15, 'bold'), bg='#008080')
    conectado1.pack(side=LEFT)
    variable3 = StringVar(otro1)
    variable3.set('DESCONECTADO')
    desconectado=Label(otro1,textvariable=variable3,font=('times', 15, 'bold'), bg='red')
    desconectado.pack(side=LEFT)
    otro1.pack()


####################################33
```

Código que presenta el tiempo en pantalla y el inicia la aplicación

```
# PRESENTA EL TIEMPO
def tick():
 global time1
 # get the current local time from the PC
 time2 = time.strftime('%H:%M:%S')
 # if time string has changed, update it
 if time2 != time1:
     time1 = time2
     clock.config(text=time2)
 # calls itself every 200 milliseconds
 # to update the time display as needed
 # could use >200 ms, but display gets jerky
 try:
     address='http://' + str(ip) + ':' + str(puerto) + '/' xda =
     TWXDAClient(OPCServerAddress=address,  ReturnErrorText=True) d =
     xda.twBrowse( ReturnAllProperties=False)
     d.addCallback(print_options) d.addErrback(handleError)

 except:
     ss=1



 clock.after(1000, tick)

tick()
menu()
frames()
clock.pack(fill=BOTH, expand=1)
# SE INICIA EL SERVIDOR
mainloop()
```

2. Librería server.py

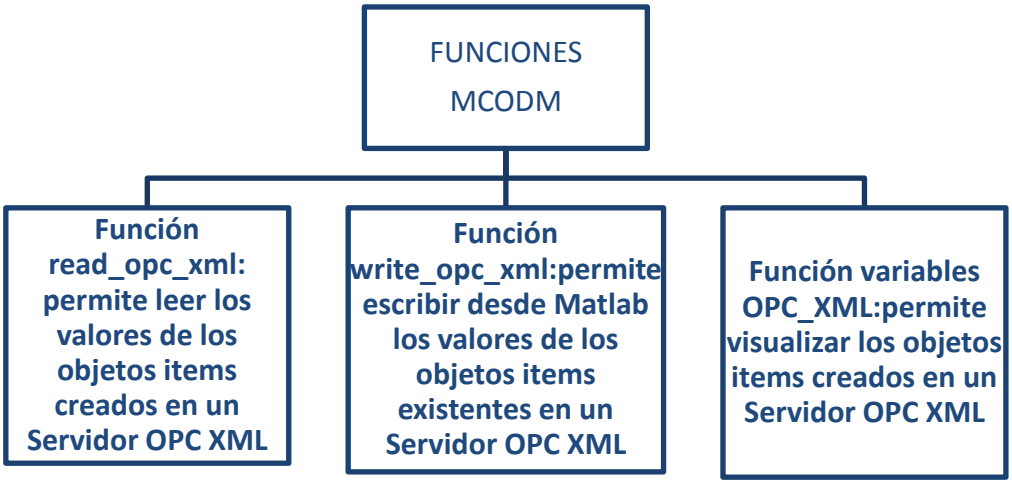
server.py - E:\Junio\servidor_xml\server.py

File Edit Format Run Options Windows Help

```
class SimpleXDAServer(BasicXDAServer):
    OPCItems = sample_items.TestOPCItems
    Ignore_ReturnItemPath=False
    Ignore_ReturnItemName=False
    VendorInfo = 'SERVIDOR OPC XML datos'

# CONECTAR EL SERVIDOR
class SERVIDOR:
    def conectar(self,puerto):
        try:

            from twisted.web import resource, server
            xdasrv = SimpleXDAServer(http_log_fn = 'http.log',access_log_fn='access1.log')
            root1 = resource.Resource()
            root1.putChild('',xdasrv)
            site = server.Site(root1)
            reactor.listenTCP(int(puerto), site)
            reactor.run()
        except:
            return '2'
# ESCRIBE LOS ITEMS
class ITEMS:

    def openit(self,ip,puerto,valor):
        address='http://' + str(ip) + ':' + str(puerto) + '/'
        xda = TWXDAClient(OPCServerAddress=address,  ReturnErrorText=True)
        print xda
        d = xda.twWrite(eval(valor))
        return d

#LEE ITEMS
class ITEMSR:

    def openit(self,ip,puerto,valor):
        address='http://' + str(ip) + ':' + str(puerto) + '/'
        xda = TWXDAClient(OPCServerAddress=address,  ReturnErrorText=True)
        print xda
        d = xda.twRead(eval(valor))
        return d
```

II. Modulo cliente OPC XML en Matlab

Se crearon tres funciones En Matlab, para la comunicación con los Servidores OPC XML, las cuales se presentan en la ilustración ..



1. Función read_opc_xml

```
%funcion que lee los valores de los objeto items
%esta funcion devueve el estado del servidor y las variables con
sus valores
%las variables deben ir separadas por una coma
%ejemplo:
read_opc_xml('http://localhost:8000','temperatura,otravariable')

function [estado,variables_f]=read_opc_xml(servidor,variablesr)
b= 'leer_variables.py  ';
g=[servidor,' '];
    g=[g,variablesr];
    c=[b,g];
    [dato,variables]=system(c);

    variables=strread(variables,'%s','delimiter',';');
    estado=variables(1);
    [t1,t2]=size(variables);
    for k=2:t1
    eval(char(variables(k)))
    end
```

```
20.      variables_f=VARIABLES
21.      s =
   struct('VARIABLES',variables_f,'SERVIDOR',servidor,'VARIABLES_W',VA
   RIABLES);
22.      assignin('base','imfile',s);
23.        datos = evalin('base', 'imfile');
24.
```

Función write_opc_xml

```
1. %Funcion que escribe las variables en el servidor OPC XML
2. %el nombre del servidor debe ir entre comillas, y las variables con
   sus
3. %valores tambien, ejemplo:
   write_opc_xml('http://localhost:8000','variable1=12,variable2=12')
4. function []=write_opc_xml(servidor,variablesw)
5. % FUNCION EN PYTHON
6. b= 'escribir_variables.py  ';
7. g=[servidor,' '];
8. g=[g,variablesw];
9. c=[b,g];
10.      system(c);
11.
12.
```

Función variables OPC_XML

```
1. %%presenta las variables que hay en el servidor OPC
2. %el servidor debe ir en comillas
3.
4. function [estado,variables_f]=variables_opc_xml(servidor)
5. b= 'importar_variables.py  ';
6. g=[servidor,9];
7. c=[b,g];
8. [dato,variables]=system(c);
9. variables=strread(variables,'%s','delimiter',';');
10.      % otro1=otro1(2);
11.      estado=variables(1);
12.      [t1,t2]=size(variables);
13.      variables_f=variables(2:t1);
14.      variables_f(1);
15.      s =
   struct('VARIABLES',variables_f,'SERVIDOR',servidor,'VARIABLES_W',1)
   ;
```

```
16.     assignin('base','imfile',s);
17.       datos = evalin('base', 'imfile');
```

Función cliente.m

```
function varargout = Cliente(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Cliente_OpeningFcn, ...
                   'gui_OutputFcn',  @Cliente_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Cliente is made visible.
function Cliente_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Cliente (see VARARGIN)

% Choose default command line output for Cliente
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Cliente wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = Cliente_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
clc
clear all
```

```
% --------------------------------------------------------------------
function Archivo_Callback(hObject, eventdata, handles)
% hObject    handle to Archivo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
function Nuevo_Callback(hObject, eventdata, handles)
% hObject    handle to Nuevo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

h2=handles.Nuevo_servidor;
set(h2,'visible','on');

function Servidor_Callback(hObject, eventdata, handles)
% hObject    handle to Servidor (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Servidor as text
%        str2double(get(hObject,'String')) returns contents of Servidor
as a double


% --- Executes during object creation, after setting all properties.
function Servidor_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Servidor (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in Adicionar.
function Adicionar_Callback(hObject, eventdata, handles)
% hObject    handle to Adicionar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
Servidor_n=get(handles.Servidor, 'String')
oldstring=get(handles.Lista_servidores, 'String');
[t1,t2]=size(oldstring);
cont=0;
```

```
h2=handles.Lista_servidores;
set(h2,'visible','on');
for k=1:t1,s= oldstring(k);
   comp1=strcmp(s,Servidor_n);
    if(comp1==1)
       cont=cont+1;
    end
end
   if (cont==0)
       servidorn(hObject, eventdata, handles)

   end

function servidorn(hObject, eventdata, handles)
a=get(handles.Servidor, 'String');
oldstring=get(handles.Lista_servidores, 'String');

       if isempty(oldstring)
            newstring = a;

       elseif ~iscell(oldstring)
            newstring = {oldstring a};

       else
            newstring = {oldstring{:} a};

       end

   set(handles.Lista_servidores, 'String',newstring);
% --- Executes on selection change in Lista_servidores.
function Lista_servidores_Callback(hObject, eventdata, handles)
% hObject    handle to Lista_servidores (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns Lista_servidores
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
Lista_servidores


% --- Executes during object creation, after setting all properties.
function Lista_servidores_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Lista_servidores (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --------------------------------------------------------------------
function Importar_variables_Callback(hObject, eventdata, handles)
try

set(handles.Var_escritura,'visible','off');
h2=handles.leer12;
set(h2,'visible','off');

seleccionado_ser=get(handles.Lista_servidores,'String');
lista=get(handles.Lista_servidores,'Value');
var1 = seleccionado_ser{lista(1)};
b= 'simple.py  ';
g=[var1,9];
c=[b,g];
% Define callbacks for context menu selec_items
% while (2) > 1
[dato,otro]=system(c)
otro1=strread(otro,'%s','delimiter',';');
% otro1=otro1(2);
estado1=strcmp(otro1(1),'ServerState : running');
estado_servidor=[otro1(1) var1];
h=handles.selec_items;
set(h,'visible','on');
if(estado1==1)
    set(handles.Estado1, 'String',estado_servidor);
 [t1,t2]=size(otro1);
s = struct('VARIABLES',[],'SERVIDOR',var1,'VARIABLES_W',1);
assignin('base','imfile',s);
  datos = evalin('base', 'imfile');

for t=2:t1
    otro1(t)
    % Create the button group.


% Create three radio buttons in the button group.

uno= char(otro1(t));

u(t) = uicontrol('Style','checkbox','String',otro1(t),...
    'pos',[80 300-30*t 140
30],'Tag',uno,'parent',h,'HandleVisibility','off');

d=get(u(t));

% Initialize some button group properties.
```

```
 set(u(t),'Callback',@selcbk1);
% set(h,'SelectedObject',[]);  % No selection

% set(h,'SelectionChangeFcn',@selcbk);
%     set(handles.variables_todas, 'String',otro1(selec_items));


end
else
    set(handles.Estado1, 'String','desconectado');
end

catch
    msgbox('debe seleccionar primero un servidor','Error ICon','error')
end


function selcbk1(source,eventdata)
global seleccionado
valor=get(source,'Value');
if(valor==0)
    v = evalin('base', 'imfile');
    datos1=get(source,'Tag');
[a,b]=size(v.VARIABLES);
    cont=0;
for k=1:b,s= v.VARIABLES(k);
   comp1=strcmp(s,datos1);
        if(comp1==1)
            v.VARIABLES{k}='';
         end

end

 v.VARIABLES(strcmp('',v.VARIABLES)) = [];



assignin('base','imfile',v);

end

if(valor==1)
    v = evalin('base', 'imfile');
    datos1=get(source,'Tag');

%
% %       datos2= [v.VARIABLES;datos1]
%    v.VARIABLES={v.VARIABLES;datos1}
```

```
% %    seleccionado = cell([datos1, v]);
% datos2={v.VARIABLES;datos1}
%  s = struct('VARIABLES',datos1)
[a,b]=size(v.VARIABLES);

cont=0;
for k=1:b,s= v.VARIABLES(k);
   comp1=strcmp(s,datos1);
        if(comp1==1)
            cont=cont+1;
         end

end
 if (cont==0)

        v.VARIABLES{b+1}=datos1;

 end




% cont=0
%     for l=0:b
%         v.VARIABLES
%         if(l==datos1)
%             cont=1
%         end
%     end
%
%   if (cont==0)
%       v.VARIABLES{b+1}=datos1;
%   end



    assignin('base','imfile',v);

end

% --------------------------------------------------------------------
function Salir_Callback(hObject, eventdata, handles)
% hObject    handle to Salir (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
try
h2=handles.leer12;
set(h2,'visible','off');
Leer_Callback(hObject, eventdata, handles)
close()
```

```
catch exception
    close()
end



% --------------------------------------------------------------------
function Comandos_Callback(hObject, eventdata, handles)
% hObject    handle to Comandos (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
function Leer_Callback(hObject, eventdata, handles)
% hObject    handle to leer12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)}
try
v = evalin('base', 'imfile');
if(length(v.VARIABLES)>0)
set(handles.Var_escritura,'visible','off');
set(handles.Var_escritura, 'String',v.VARIABLES);
global gg
try
stop(gg)
catch exception
end
set(handles.Var_escritura,'visible','off');
ha=handles.selec_items;
h2=handles.leer12;
set(h2,'visible','on');
set(ha,'visible','off');

u(1) = uicontrol('Style','checkbox','String','iniciar_lectura y
escritura',...
    'pos',[80 30 140
30],'Tag','iniciar_lect','parent',h2,'HandleVisibility','off');
% Initialize some button group properties.
  set(u(1),'Callback',@selcbk);
% set(h2,'SelectedObject',[]);  % No selection
% set(h2,'Visible','on');
else
     h = msgbox('debe primero importar las variables','Error
ICon','error')
end
catch exception
    h = msgbox('debe primero importar las variables','Error
ICon','error')
end
```

```
function selcbk(source,eventdata)
   global gg
    selecc=get(source,'Value');
%    pause(1)
%    while(selecc==0)
%         selecc
%         disp('.')
%         delay(1)
%
%         end
  if(selecc==1)
gg = timer('ExecutionMode', 'fixedSpacing','StartDelay', 1, 'Period', 1);
gg.StartFcn = {@my_callback_fcn, 'Servidor inicializado'};
gg.TimerFcn = { @my_callback_fcn3, 'leer'};
gg.StopFcn = { @my_callback_fcn1, 'lectura Finalizado'};

start(gg)
end
 if(selecc==0)
    stop(gg)
 end


function my_callback_fcn3(obj, event, string_arg)
try
v = evalin('base', 'imfile');

var2=v.SERVIDOR;
b= 'simpleread.py  ';
g=[var2,9];
c=[b,g];
[dato,otro]=system(c);
otro1=strread(otro,'%s','delimiter',';');
[j,p]=size(otro1);
for h=1:j
    variablesa=otro1(h);
    ddd=strvcat(variablesa);
   otro2=strread(ddd,'%s','delimiter',',');
     nombre=otro2(1);
     valor=otro2(2)
   valor= eval(valor{1,1})
     nombre1=strvcat(nombre);
     nombre2=compactar(nombre1);

     otro4=strcmp(v.VARIABLES, nombre2);
     [z,l]=size(otro4);
     otro=0;
     for dd=1:l
         if(otro4(dd)==1)
               assignin('base',nombre2,valor);
               a= evalin('base', nombre2);
```

```
        end
    end

%         if (otro==1)
%          assignin('base',nombre2,str2double(valor));
%           a= evalin('base', nombre2);
%         end









end

catch exception

a=2
end
datos()

function x=compactar(f)

n=length(f);

x='';

for i=1:n

if f(i) ~= ' '

x = strcat(x, f(i));

end

end


%
% event_type = event.Type;
% event_time = datestr(event.Data.time);
%
% msg = [event_type txt1 event_time];
% disp(msg)
```

```
function my_callback_fcn(obj, event, string_arg)

txt1 = ' event occurred at ';
txt2 = 'string_arg';
%
% event_type = event.Type;
% event_time = datestr(event.Data.time);
%
% msg = [event_type txt1 event_time];
% disp(msg)



function my_callback_fcn1(obj, event, string_arg)

txt1 = ' event occurred at ';
txt2 = string_arg;

event_type = event.Type;
event_time = datestr(event.Data.time);

msg = [event_type txt1 event_time];
disp(msg)
disp(txt2)

 % --------------------------------------------------------------------
function Escribir_Callback(hObject, eventdata, handles)
try
    set(handles.leer12,'visible','off');
    set(handles.selec_items,'visible','off');

v = evalin('base', 'imfile')
if(length(v.VARIABLES)>0)
    v.VARIABLES
set(handles.Var_escritura,'visible','on');
set(handles.Var_escritura, 'String',v.VARIABLES);

else
     h = msgbox('debe primero importar las variables1','Error
ICon','error')
end
catch exception
    h = msgbox('debe primero importar las variables','Error
ICon','error')
end


% --- Executes on selection change in Var_escritura.
```

```
function Var_escritura_Callback(hObject, eventdata, handles)
% hObject    handle to Var_escritura (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
K = menu('SELECIONA LA OPCION DE ESCRITURA','Asociar item a una variable
existente en el Workspace','escribir valor directamente')
contents = get(hObject,'String')
%returns Var_escritura contents as cell array
selec=  contents{get(hObject,'Value')}
if(K==2)
valor = inputdlg('ingrese el nuevo valor')
v = evalin('base', 'imfile');
var2=v.SERVIDOR;
b= 'write.py  ';
g=[var2,9];
a = strcat(selec, ',', valor);
a=char(a)
c=[b,g,a];
[dato,otro1111]=system(c);
end
if(K==1)
v = evalin('base', 'imfile');
workspace = inputdlg('ingrese el nombre de la variable')
c=char(workspace)
valor2=strcat('v.VARIABLES_W.',selec,'=','c')
eval(valor2)
try
    v.VARIABLES_W1{length(v.VARIABLES_W1)+1}=selec

catch
   v.VARIABLES_W1={selec}
end
assignin('base','imfile',v);

end

function datos()

try

v = evalin('base', 'imfile');
b=length(v.VARIABLES_W1);
for k=1:b,s= v.VARIABLES_W1(k);
    try
    valor3=strcat('v.VARIABLES_W.',s)
    valor4=eval(char(valor3))
    valor1 = evalin('base',valor4);
var2=v.SERVIDOR;
b= 'write.py  ';
g=[var2,9];
a = strcat(s, ',',char(num2str(valor1)))
```

```
a=char(a)
c=[b,g,a];
[dato,otro1111]=system(c);
  catch exception
        ms='no se puede escr'
    end


end
catch exception
        ms='no se puede escr'
    end


%returns selected item from Var_escritura
% --- Executes during object creation, after setting all properties.
function Var_escritura_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Var_escritura (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in listbox3.
function listbox3_Callback(hObject, eventdata, handles)
% hObject    handle to listbox3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns listbox3 contents as
cell array
%        contents{get(hObject,'Value')} returns selected item from
listbox3
% --- Executes during object creation, after setting all properties.
function listbox3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end


function Asociar_a_variables_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listbox3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --------------------------------------------------------------------
```

III. CLIENTES DCOM/XML EN RTAI-LAB

Para implementar un bloque en scicos se deben crear dos archivos: uno en Scilab con extensión *.sci que contiene el código de la apariencia y las propiedades del bloque, y un script en código en C para la ejecución del programa en tiempo real. Adicional a estos dos archivos, se debe incluir las librerías en Python encargadas de la comunicación OPC.

La figura 3-15 presenta un diagrama archivos de los cuatro bloques OPC implementados (Dcom_Read, Dcom_Write, Xml_Read, Xml_Write); cada bloque está compuesto por:

La función Interfaz *(Interfacing function):* determina la geometría, color, número de puertos, tamaño, icono, etc. Esta función también proporciona el dialogo del usuario con el bloque, se escribe en código Scilab.

La función Computacional *(Computacional function):* especifica el comportamiento dinámico de cada bloque y es programada en C.

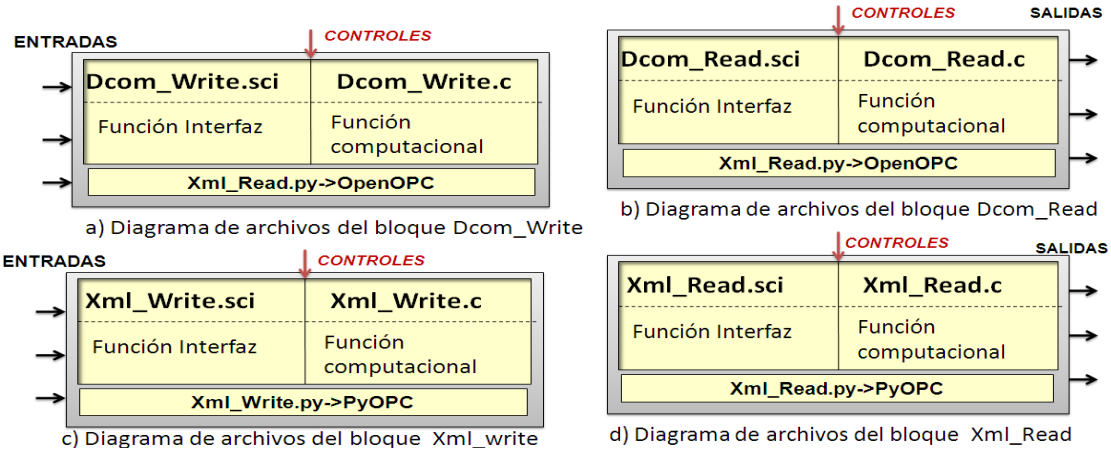
Librerías en Python: para los bloques OPC DCOM se hace uso de la librería OpenOPC, y para los bloques OPC XML se utiliza PyOPC.

Controles: los cuatros bloques requieren de una señal de reloj que permite realizar la actualización de parámetros en forma periódica.

Salidas: corresponden a los valores provenientes de los objetos ítems de los servidores

Entradas: corresponden a los valores provenientes de otros bloques los cuales, se envían a los objetos ítems de los servidores OPC.

Figura K-3: Diagrama de archivos del MCODR y MCOXR



a) Diagrama de archivos del bloque Dcom_Write

b) Diagrama de archivos del bloque Dcom_Read

c) Diagrama de archivos del bloque Xml_write

d) Diagrama de archivos del bloque Xml_Read

A continuación se presenta el código de las cuatro funciones de Interfaz y las cuatro funciones Computacionales.

Funcion Dcom_Read.sci

```
function [servidor_x]=fservidorDR(ip)
try
cd /usr/local/scilab-4.1.2/routines/scicos
   direc1='python opcdcomsci.py 1 ' + ip;
   servidor_x= unix_g(direc1);
   servidor_x=eval(servidor_x)';
   n=x_choose(servidor_x,['SELECCIONE EL SERVIDOR'],'Return');
   servidor_x=servidor_x(n);
catch
   x_message('no se puede establecer informacion con el equipo remoto');
   servidor_x=1;
end

endfunction

function [variables3,variables4]=fvariablesDR(servidor_x,ip,outport)
try
   direc1='python opcdcomsci.py 2 '+ ip + ' ' + servidor_x;
   variables= unix_g(direc1);
   variables=eval(variables)';
   variables3=[];
   labels=[];
   salida_x=[];
   for j = 1:outport
      salida=string(cellstr(['SELECCIONE LA VARIABLE PARA LA
SALIDA:',string(j)]));
      salida_x(j)=string(cellstr(['salida:',string(j)]));
      n=x_choose(variables,[salida],'Return')
      variables3(j)=string(variables(n));
   end
```

```
    variables4=''
    sig=x_mdialog('edite las salidas',salida_x,variables3)
    variables5=[]
    for j = 1:outport
       variables3(j)=string(sig(j));
       variables4=variables4 + string(variables3(j)) + ','

   end
a=length(variables4)-1
variables4=part(variables4,1:a)
catch
variables3=["a"]
variables4=''
end
endfunction
// FUNCION PRINCIPAL
function [x,y,typ]=Dcom_Read(job,arg1,arg2)

x=[];y=[];typ=[];
select job
// SE GRAFICA EL BLOQUE
case 'plot' then
  standard_draw(arg1)
  // SE DEFINEN ENTRADAS
case 'getinputs' then
  [x,y,typ]=standard_inputs(arg1)
    // SE DEFINEN SALIDAS
case 'getoutputs' then
  [x,y,typ]=standard_outputs(arg1)
case 'getorigin' then
  [x,y]=standard_origin(arg1)

    // PERMITE CONFIGURAR PARAMETROS
case 'set' then
  x=arg1;
  graphics=arg1.graphics;exprs=graphics.exprs
  model=arg1.model;
  while %t do
    exprs1=exprs;
    // PRESENTA FORMULARIO DE PARAMETROS
    [ok,ip,outport,exprs]=getvalue('Parametros opc_read_dcom',..
    ['IP';'No de salidas'],list('str',1,'vec',-1),exprs)
    if ~ok then break,end
      in=[];
    if ok then
    // PRESENTA FORMULARIO DE SELECCION DEL SERVIDOR
    [servidor_x]=fservidorDR(ip);
    if(servidor_x=='1') then
    x_message('No se puede establecer comunicacion:'+ip);
    else
    // PRESENTA FORMULARIO DE SELECCION DE VARIABLES
        x_message('esto puede tardar un poco..');
```

```
          [variables3,variables4]=fvariablesDR(servidor_x,ip,outport)
          if exists('outport') then out=ones(outport,1), else out=1, end
       [model,graphics,ok]=check_io(model,graphics,in,out,1,[])

    exprs=[ip,sci2exp(outport)]
      model.rpar=[outport];

    varnam=servidor_x +'@@'+ eval(ip) + '@@' +  variables4;
    model.ipar=[length(varnam);str2code(varnam)];
      graphics.exprs=exprs
      x.graphics=graphics;x.model=model
    end
      break

   end
   end
   // VALORES INICIALES
case 'define' then
    variables1=[];
    varnam='servidor@@ip@@a';
    slope=0;iout=0;
    stt=0;out=1;
    ip='169.254.228.145';
    rpar=[out];
    model=scicos_model()
    model.sim=list('dcom_read',4)
    model.in=[]
     model.evtin=1
    model.out=-[1:out]'
    model.rpar=rpar
    model.ipar=[length(varnam);str2code(varnam)];
    model.blocktype='c'
    model.nmode=1
    model.nzcross=1
    model.dep_ut=[%f %t]
    exprs=[sci2exp(ip),sci2exp(out)]
  gr_i=['xstringb(orig(1),orig(2),[''Opc DCOM''
;''Read''],sz(1),sz(2),''fill'');']
    x=standard_define([3 2],model,exprs,gr_i)
end
endfunction
```

Función Dcom_Write.sci

```
function [servidor_x]=fservidorDW(ip)
try
cd /usr/local/scilab-4.1.2/routines/scicos;
   direc1='python opcdcomsci.py 1 ' + ip;
   servidor_x= unix_g(direc1);
```

```
   servidor_x=eval(servidor_x)';
   n=x_choose(servidor_x,['SELECCIONE EL SERVIDOR'],'Return');
   servidor_x=servidor_x(n);
catch
   x_message('no se puede establecer informacion con el equipo remoto');
   servidor_x=1;
end

endfunction

function [variables3,variables4]=fvariablesDW(servidor_x,ip,inport)
try
   direc1='python opcdcomsci.py 2 '+ ip + ' ' + servidor_x;
   variables= unix_g(direc1);
   variables=eval(variables)';
   variables3=[];
   labels=[];
   salida_x=[];
   for j = 1:inport
      salida=string(cellstr(['SELECCIONE LA VARIABLE PARA LA
ENTRADA:',string(j)]));
      salida_x(j)=string(cellstr(['entrada:',string(j)]));
      n=x_choose(variables,[salida],'Return')
      variables3(j)=string(variables(n));
   end
   variables4=''
   sig=x_mdialog('aqui puede editar los nombres de las
variables',salida_x,variables3)
   variables5=[]
   for j = 1:inport
      variables3(j)=string(sig(j));
      variables4=variables4 + string(variables3(j)) + ','
   end
  a=length(variables4)-1
variables4=part(variables4,1:a)


catch
variables3=[]
variables4=''
end
endfunction
// FUNCION PRINCIPAL -AQUI INICIA
function [x,y,typ]=Dcom_Write(job,arg1,arg2)
// Copyright INRIA
x=[];y=[];typ=[];
select job
case 'plot' then
  standard_draw(arg1)
case 'getinputs' then
  [x,y,typ]=standard_inputs(arg1)
case 'getoutputs' then
```

```
  [x,y,typ]=standard_outputs(arg1)
case 'getorigin' then
  [x,y]=standard_origin(arg1)
  // PERMITE LA CONFIGURACION DE PARAMETROS
case 'set' then
  x=arg1;
  graphics=arg1.graphics;exprs=graphics.exprs
  model=arg1.model;
  while %t do
    exprs1=exprs;
    [ok,ip,inport,exprs]=getvalue('Parametros opc_read_dcom',..
    ['IP';'No entradas'],list('str',1,'vec',-1),exprs)
    if ~ok then break,end
   if ok then
      out=[],
    if exists('inport') then in=ones(inport,1), else in=1, end
      [model,graphics,ok]=check_io(model,graphics,in,out,1,[])

    [servidor_x]=fservidorDW(ip);
    if(servidor_x=='1') then
    x_message('No se puede establecer comunicacion:'+ip);
    else
         x_message('Un momento...');
         [variables3,variables4]=fvariablesDW(servidor_x,ip,inport)

    exprs=[ip,sci2exp(inport)]

      model.rpar=[inport];

    varnam=servidor_x + '@@'  +eval(ip) + '@@' +  variables4 + '@@';

    model.ipar=[length(varnam);str2code(varnam)];
      graphics.exprs=exprs
      x.graphics=graphics;x.model=model
    end
      break

   end
   end
   // DEFINE VALORES INICIALES
case 'define' then
    varnam='SERV@@IP@@VAR@@';
   in=1;
   servidor='matrikon'
    ip='169.254.228.145';
    rpar=[in];
    model=scicos_model()
    model.sim=list('Dcom_Write',4)
    model.in=-[1:in]'
     model.evtin=1
    model.out=[]
```

```
    model.rpar=rpar
    model.ipar=[length(varnam);str2code(varnam)];
    model.blocktype='c'
    model.nmode=1
    model.nzcross=1
    model.dep_ut=[%f %t]
    exprs=[sci2exp(ip),sci2exp(in)]
  gr_i=['xstringb(orig(1),orig(2),[''Opc DCOM''
;''write''],sz(1),sz(2),''fill'');']
    x=standard_define([3 2],model,exprs,gr_i)
end
endfunction
```

Function Xml_Read.sci

```
function [variables_x]=fservidorXR(serv)
cd /usr/local/scilab-4.1.2/routines/scicos;
try
direc1='python xmlopc.py '+ serv;
variables_x=unix_g(direc1);
catch
variables_x='';
end
endfunction

function [variables3,variables4]=fvariablesXR(variables_x,variables_v)

    variables1= eval(variables_v);
    variables=strcat(['[',variables_x,']']);
    variables=eval(variables);
    l=list();
    for j = 1:outport
        i=j+1;
        try
            dato1=variables1(j);
        catch
            dato1='j';
        end
        valor=find(variables==dato1);
        if(valor>=-1) then
            salida=cellstr(['salida',string(j)]);
            dato=list(string(salida),valor,variables);
            l(j)=dato;
        else
            salida=cellstr(['salida',string(j)]);
            dato=list(string(salida),1,variables);
            l(j)=dato;
        end
```

```
    end

    rep = x_choices('seleccione las variables opc',1);
    cont=0;
    variables4=''
    for l=rep;
        cont=cont+1;
        variables4=variables4 + variables(l) + ','
        variables3(cont)=variables(l),end

    a=length(variables4)-1;
    variables4=part(variables4,1:a);


endfunction


function [x,y,typ]=Xml_Read(job,arg1,arg2)
// Copyright INRIA
x=[];y=[];typ=[];
select job
case 'plot' then
  standard_draw(arg1)
case 'getinputs' then
  [x,y,typ]=standard_inputs(arg1)
case 'getoutputs' then
  [x,y,typ]=standard_outputs(arg1)
case 'getorigin' then
  [x,y]=standard_origin(arg1)
case 'set' then
  x=arg1;
  graphics=arg1.graphics;exprs=graphics.exprs
  model=arg1.model;
  while %t do
    exprs1=exprs;
    [ok,serv,outport,exprs]=getvalue('parametros opc xml read',..
    ['Servidor';'salidas'],list('str',1,'vec',-1),exprs)
    if ~ok then break,end
in=[],
    if exists('outport') then out=ones(outport,1), else out=1, end
      [model,graphics,ok]=check_io(model,graphics,in,out,1,[])
      if ok then
    [variables_x]=fservidorXR(serv);
    if(variables_x=='1') then
    variables3=['b','b','b'];
        x_message('No se puede establecer comunicacion:'+serv);
    else
     [variables3,variables4]=fvariablesXR(variables_x,exprs1(3))

    exprs=[serv,sci2exp(outport),sci2exp(variables3)]
      model.rpar=[outport];
```

```
    varnam='python opcxml.py ' + eval(serv) + ' 0 ' + variables4;
    model.ipar=[length(varnam);str2code(varnam)];
      graphics.exprs=exprs
      x.graphics=graphics;x.model=model
    end
      break

   end
   end
case 'define' then
    variables1=['a','b','c'];
    varnam='python opcxml.py http://localhost:8000 0 a';
    slope=0;iout=0;
    stt=0;out=1;
    serv='http://localhost:8001';
    rpar=[out];
    model=scicos_model()
    model.sim=list('Xml_Read',4)
    model.in=[]
     model.evtin=1
    model.out=-[1:out]'
    model.rpar=rpar
    model.ipar=[length(varnam);str2code(varnam)];
    model.blocktype='c'
    model.nmode=1
    model.nzcross=1
    model.dep_ut=[%f %t]
    exprs=[sci2exp(serv),sci2exp(out),sci2exp(variables1)]


  gr_i=['xstringb(orig(1),orig(2),[''Opc Xml''
;''Read''],sz(1),sz(2),''fill'');']
    x=standard_define([3 2],model,exprs,gr_i)


end
endfunction
```

Xml_write.sci

```
function [variables_x]=fservidorXW(serv)
cd /usr/local/scilab-4.1.2/routines/scicos
try
direc1='python xmlopc.py '+ serv;
variables_x=unix_g(direc1);
catch
variables_x='';
end
endfunction
```

```
function [variables3,variables4]=fvariablesXW(variables_x,variables_v)

    variables1= eval(variables_v);
    variables=strcat(['[',variables_x,']']);
    variables=eval(variables);
    l=list();
    for j = 1:inport
        i=j+1;
        try
            dato1=variables1(j);
        catch
            dato1='j';
        end
        valor=find(variables==dato1);
        if(valor>=-1) then
            salida=cellstr(['Entrada',string(j)]);
            dato=list(string(salida),valor,variables);
            l(j)=dato;
        else
            salida=cellstr(['salida',string(j)]);
            dato=list(string(salida),1,variables);
            l(j)=dato;
        end

    end

    rep = x_choices('Seleccione las variables OPC ',l);
    cont=0;
    variables4=''
    for l=rep;
        cont=cont+1;
        variables4=variables4 + variables(l) + ','
        variables3(cont)=variables(l),end
        a=length(variables4)-1
    variables4=part(variables4,1:a)


endfunction


function [x,y,typ]=Xml_write(job,arg1,arg2)
// Copyright INRIA
x=[];y=[];typ=[];
select job
case 'plot' then
  standard_draw(arg1)
case 'getinputs' then
  [x,y,typ]=standard_inputs(arg1)
case 'getoutputs' then
  [x,y,typ]=standard_outputs(arg1)
```

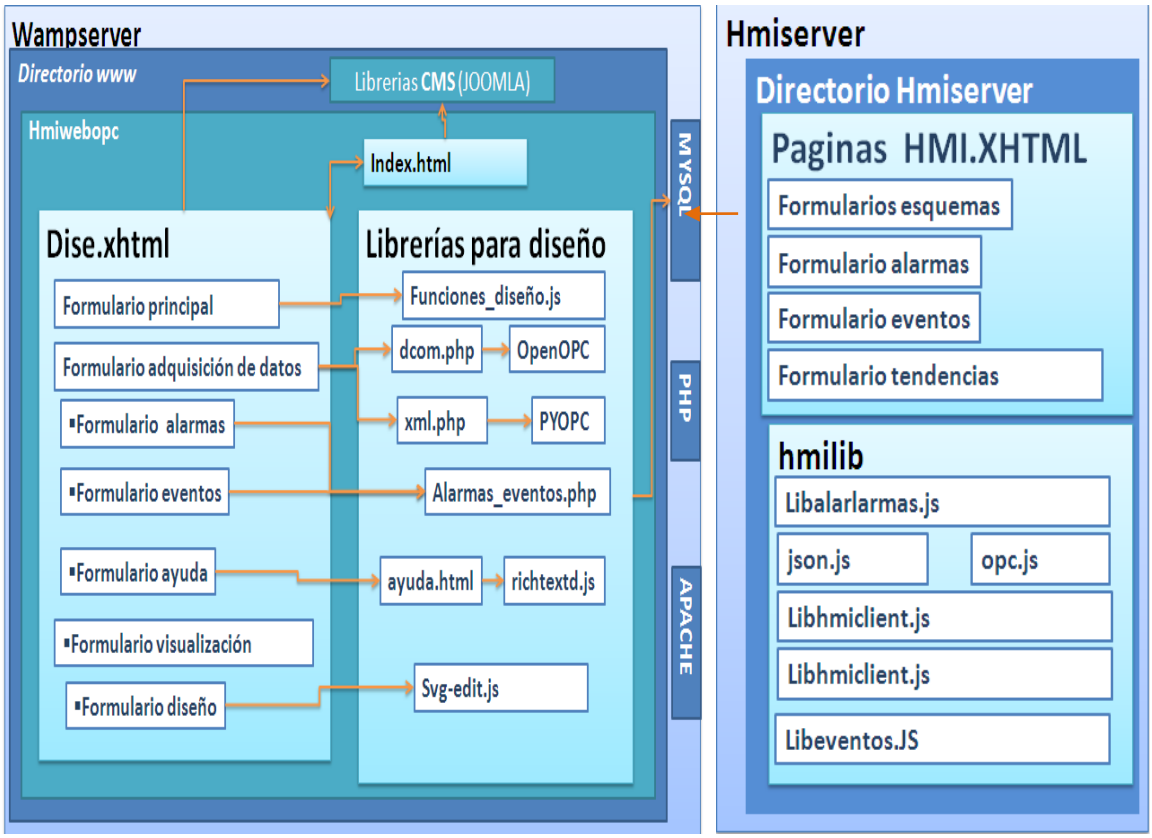
```
case 'getorigin' then
  [x,y]=standard_origin(arg1)
case 'set' then
  x=arg1;
  graphics=arg1.graphics;exprs=graphics.exprs
  model=arg1.model;
  while %t do
    exprs1=exprs;
    [ok,serv,inport,exprs]=getvalue('Parametros Opc XML Write',..
    ['Servidor';'No de entradas'],list('str',1,'vec',-1),exprs)
    if ~ok then break,end
out=[],
    if exists('inport') then in=ones(inport,1), else in=1, end
      [model,graphics,ok]=check_io(model,graphics,in,out,1,[])
      if ok then
    [variables_x]=fservidorXW(serv);
    if(variables_x=='1') then
    variables3=['b','b','b'];
        x_message('No se puede establecer comunicacion:'+serv);
    else
     [variables3,variables4]=fvariablesXW(variables_x,exprs1(3))

    exprs=[serv,sci2exp(inport),sci2exp(variables3)]

      model.rpar=[inport];
      varnam='python opcxml_w.py ' + eval(serv) + ' 0 ' + variables4;

      model.ipar=[length(varnam);str2code(varnam)];
      graphics.exprs=exprs
      x.graphics=graphics;x.model=model
    end
      break

   end
   end
case 'define' then
    variables1=['a','b','c'];
    variables2='a'+ ',b', + 'c';
    varnam='python opcxml_w.py http://localhost:8000 0 ' + variables2;
    slope=0;iout=0;
    stt=0;in=2;
    serv='http://localhost:8001';
    rpar=[in];
    model=scicos_model()
    model.sim=list('Xml_Write',4)
    model.in=-[1:in]'
    model.out=[]'
    model.evtin=1
    model.rpar=rpar
    model.ipar=[length(varnam);str2code(varnam)];
    model.blocktype='c'
    model.nmode=1
```

```
    model.nzcross=1
    model.dep_ut=[%f %t]
    exprs=[sci2exp(serv),sci2exp(in),sci2exp(variables1)]


  gr_i=['xstringb(orig(1),orig(2),[''Opc Xml''
;''Write''],sz(1),sz(2),''fill'');']
    x=standard_define([3 2],model,exprs,gr_i)


end
endfunction
```

IV. MODULO HMI WEB

Los archivos de las páginas web y librerías se almacenan en el directorio de cada Servidor WEB, para los sub-módulos gestor de contenido y diseño se utiliza el directorio “www” del servidor Wampserver, dentro de dicho directorio se encuentran dos carpetas, una donde están todas las librerías del gestor de contenido (Joomla), y la otra “hmiwebopc” donde están las páginas asociadas al sub-módulo de diseño. El archivo “Index.html” es la página inicial del gestor de contenido, ver imagen 5-1, y el archivo “dise.xhtml” contiene el código de la página principal del sub-módulo de diseño, esta página se compone de varios formularios, los cuales hacen uso de librerías diseñadas en Javascript, PHP y Python. Cada vez que el usuario diseña las pantallas HMI, esta información se almacena en MYSQL, cuando el usuario en el formulario de visualización, selecciona el sistema que va a monitorear, se realiza una consulta a la base de datos y se generan los archivos en el directorio del servidor Hmiserver necesarios para el monitoreo, cuando el usuario finaliza el monitoreo se borran todos los archivos asociados a él.

Figura K-4: Diagrama de archivos del MM&SW



Fuente: propia

Index.html

index.html index.php

```php
<?php
/**
* @version      $Id: index.php 14401 2010-01-26 14:10:00Z louis $
* @package      Joomla
* @copyright    Copyright (C) 2005 - 2010 Open Source Matters. All rights reserved.
* @license      GNU/GPL, see LICENSE.php
* Joomla! is free software. This version may have been modified pursuant
* to the GNU General Public License, and as distributed it includes or
* is derivative of works licensed under the GNU General Public License or
* other free or open source software licenses.
* See COPYRIGHT.php for copyright notices and details.
*/

// Set flag that this is a parent file
define( '_JEXEC', 1 );

define('JPATH_BASE', dirname(__FILE__) );

define( 'DS', DIRECTORY_SEPARATOR );

require_once ( JPATH_BASE .DS.'includes'.DS.'defines.php' );
require_once ( JPATH_BASE .DS.'includes'.DS.'framework.php' );

JDEBUG ? $_PROFILER->mark( 'afterLoad' ) : null;

/**
 * INITIALISE THE APPLICATION
 *
 * NOTE :
 */
// set the language
$mainframe->initialise();

JPluginHelper::importPlugin('system');

// trigger the onAfterInitialise events
JDEBUG ? $_PROFILER->mark('afterInitialise') : null;
$mainframe->triggerEvent('onAfterInitialise');

/**
 * ROUTE THE APPLICATION
 *
 * NOTE :
 */
$mainframe->route();

// authorization
$Itemid = JRequest::getInt( 'Itemid');
$mainframe->authorize($Itemid);

// trigger the onAfterRoute events
JDEBUG ? $_PROFILER->mark('afterRoute') : null;
$mainframe->triggerEvent('onAfterRoute');

/**
 * DISPATCH THE APPLICATION
 *
 * NOTE :
 */
$option = JRequest::getCmd('option');
$mainframe->dispatch($option);

// trigger the onAfterDispatch events
JDEBUG ? $_PROFILER->mark('afterDispatch') : null;
$mainframe->triggerEvent('onAfterDispatch');

/**
 * RENDER  THE APPLICATION
 *
 * NOTE :
 */
$mainframe->render();

// trigger the onAfterRender events
JDEBUG ? $_PROFILER->mark('afterRender') : null;
$mainframe->triggerEvent('onAfterRender');

/**
 * RETURN THE RESPONSE
 */
echo JResponse::toString($mainframe->getCfg('gzip'));

```

Dise-xhtml

Importar librerías

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Expires" content="0"></meta>
<meta http-equiv="Last-Modified" content="0"></meta>
<meta http-equiv="Cache-Control" content="no-cache, mustrevalidate"></meta>
<meta http-equiv="Pragma" content="no-cache"></meta>
<script type="text/javascript" src="hmiserver/hmipages/scadastodos.js"></script>
<script type="text/javascript" src="hmiserver/hmipages/usuarios.js"></script>
<script type="text/javascript" src="hmiserver/hmipages/usact.js"></script>
<script type="text/javascript" src="demosupport/demosim/datos_scada.py"></script>
<script type="text/javascript" src="json.js"></script>
<script type="text/javascript" src="funcionesdise.js"></script>
<script type="text/javascript" src="generar_scada.js"></script>
<script type="text/javascript" src="lib/variables.js"></script>
<script type="text/javascript" src="variables1.js"></script>
<script type="text/javascript" src="full.js"></script>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Untitled Document</title>
```

Inicio del cuerpo de la página

```
</head>
<body onload="init();" oncontextmenu="return false"  >
```

Formulario principal

```
</head>
<body onload="init();" oncontextmenu="return false"  >


<svg width="950" height="90" id= "svgprincipal" xmlns="http://www.w3.org/2000/svg">
 <!-- Created with SVG-edit - http://svg-edit.googlecode.com/ -->
 <defs>
  <radialGradient id="svg_8" cx="0.5" cy="0.5" r="0.5">

  </radialGradient>
 </defs>
 <g>
  <title>Layer 1</title>
  <rect fill="url(#svg_8)" stroke="#000000" stroke-width="1" x="7" y="0.79994" width="890" height=
"86.00001" id="svg_7"/>
  <text fill="#0f1f70" stroke="#000000" stroke-width="0" x="438.1" y="31.8" id="svg_6" font-size="32"
 font-family="serif" text-anchor="middle" xml:space="preserve" font-weight="bold" font-style="italic"
>HMI WEB CON SOPORTE OPC</text>
  <text fill="#162a99" stroke="#000000" stroke-width="0" x="50" y="71.8" id="nuevom" font-size="32"
font-family="serif" text-anchor="middle" xml:space="preserve"  onclick="nuevo()">Nuevo</text>
  <text fill="#162a99" stroke="#000000" stroke-width="0" x="200" y="71.8" font-size="32" font-family=
  <text fill="#162a99" stroke="#000000" stroke-width="0" x="50" y="71.8" id="nuevom" font-size="32"
font-family="serif" text-anchor="middle" xml:space="preserve"  onclick="nuevo()">Nuevo</text>
  <text fill="#162a99" stroke="#000000" stroke-width="0" x="200" y="71.8" font-size="32" font-family=
"serif" text-anchor="middle" xml:space="preserve" id="eliminarm1" onclick="eliminar()">Eliminar</text
>
  <text fill="#162a99" stroke="#000000" stroke-width="0" x="400" y="71.8" font-size="32" font-family=
"serif" text-anchor="middle" xml:space="preserve" id="editarm" onclick="editar()" >Editar</text>
   <text fill="#162a99" stroke="#000000" stroke-width="0" x="600" y="71.8" font-size="32" font-family
="serif" text-anchor="middle" xml:space="preserve" id="generar_scadam" onclick="generar_scada1()">
Visualizar</text>
    <text fill="#162a99" stroke="#000000" stroke-width="0" x="800" y="71.8" font-size="32"
font-family="serif" text-anchor="middle" xml:space="preserve" id="svg_5" onclick="home()">Inicio</
text>
 </g>
</svg>
```

Formulario editar

```
<div id="menu_scada" style="display:none">
<svg width="855" height="50" xmlns="http://www.w3.org/2000/svg" >
<text x="15" y="90" id="evtText2">Typing Inactive</text>
 <!-- Created with SVG-edit - http://svg-edit.googlecode.com/ -->
 <g>
  <title>Layer 1</title>
  <g id="svg_12">
   <rect id="svg_4" height="30" width="140" y="11.000671" x="20" stroke-width="5" stroke="#2a0ec9"
fill="#0083ff"  onclick="importar_variables()"/>
   <text transform="matrix(1, 0, 0, 0.666667, 0, 12.6107)" xml:space="preserve" text-anchor="middle"
font-family="serif" font-size="21" id="fuentededatosm" y="25" x="87" stroke-width="0" stroke=
"#2a0ec9" fill="#000000" onclick="importar_variables()">Fuente de datos</text>
  </g>
  <g id="svg_13">
   <rect id="svg_6" height="30" width="140" y="11.000733" x="195" stroke-width="5" stroke="#2a0ec9"
fill="#0083ff" onclick="menu_proceso()"/>
   <text transform="matrix(1.00787, 0, 0, 0.727273, -1.89764, 10.3178)" id="disem" xml:space=
"preserve" text-anchor="middle" font-family="serif" font-size="20" y="24.9393" x="270" stroke-width=
"0" stroke="#2a0ec9" fill="#000000" onclick="menu_proceso()">Dise&ntilde;o grafico </text>
  </g>
  <g id="svg_14">
   <rect id="svg_8" height="30" width="140" y="11.000733" x="358" stroke-width="5" stroke="#2a0ec9"
fill="#0083ff" onclick="ver_eventos()" />
   <text transform="matrix(1.01575, 0, 0, 0.727273, -6.12598, 10.3178)" id="eventosm" xml:space=
"preserve" text-anchor="middle" font-family="serif" font-size="20" y="24.9393" x="420" stroke-width=
"0" stroke="#2a0ec9" fill="#000000"  onclick="ver_eventos()"> Config. Eventos</text>
  </g>
  <g id="svg_15">
```

```
   <rect id="svg_10" height="30" width="140" y="11.000763" x="530" stroke-width="5" stroke="#2a0ec9"
onclick="ver_alarmas()" fill="#0083ff"/>
   <text transform="matrix(1, 0, 0, 0.757576, 0, 8.92898)" id="ver_alarmasm" xml:space="preserve"
text-anchor="middle" font-family="serif" font-size="20" y="26.735" x="600" stroke-width="0" stroke=
"#2a0ec9" fill="#000000" onclick="ver_alarmas()"> Config. Alarmas</text>
  </g>
  <g id="svg_25">
   <rect id="svg_23" height="30" width="140" y="11.000763" x="700" stroke-width="5" stroke="#2a0ec9"
onclick="ayuda()" fill="#0083ff"/>
   <text id="ayudam" transform="matrix(1, 0, 0, 0.757576, 0, 8.92898)" xml:space="preserve"
text-anchor="middle" font-family="serif" font-size="20" y="26.735" x="780" stroke-width="0" stroke=
"#2a0ec9" fill="#000000" onclick="ayuda()">Ayuda</text>
  </g>
 </g>
</svg>
```

Menú diseño

```
<div id="menu_proceso" style="display:none">
<svg width="940" height="50" xmlns="http://www.w3.org/2000/svg">
 <!-- Created with SVG-edit - http://svg-edit.googlecode.com/ -->
 <g>
  <text fill="#1313b5" stroke="#2a0ec9" stroke-width="0" x="758.833" y="48.1683" font-size="25"
font-family="serif" text-anchor="middle" xml:space="preserve" transform="matrix(1, 0, 0, 0.674895, 0,
 1.02018)" id="visualizar_p"  onclick="visualizar_cargar_proceso()">visualizar</text>
  <title>Layer 1</title>
  <text fill="#1313b5" stroke="#2a0ec9" stroke-width="0" x="561.833" y="52.0335" font-size="25"
font-family="serif" text-anchor="middle" xml:space="preserve" id="eliminar_p" transform="matrix(1, 0,
 0, 0.674895, 0, 1.02018)" onclick="eliminar_cargar_proceso()">eliminar</text>
 0, 0.674895, 0, 1.02018)" onclick="eliminar_cargar_proceso()">eliminar</text>
  <text fill="#1313b5" stroke="#2a0ec9" stroke-width="0" x="358.833" y="45.0375" font-size="25"
font-family="serif" text-anchor="middle" xml:space="preserve" id="modificar_p" transform="matrix(1,
0, 0, 0.640022, 0, 7.15225)" onclick="modificar_cargar_proceso()">modificar</text>
  <text fill="#1717b5" stroke="#2a0ec9" stroke-width="0" x="168.833" y="44.7656" id="svg_27"
font-size="25" font-family="serif" text-anchor="middle" xml:space="preserve" transform="matrix(1, 0,
0, 0.538292, 0, 9.3592)" onclick="nuevo_proceso()">nuevo</text>
 </g>
</svg>
</div>
```

Formulario para selección de HMI

```
<div id="seleccion_scada" style="display:none" align="center">
seleccione el nombre del sistema HMI
<form id="scadaver" align="center">
<select id="eliminar_scada1"   onchange="el_scada();" style="display:none" ></select>
<select id="editar_scada1"  onchange="ed_scada()" style="display:none"></select>
<select id="visualizar_scada1"  onchange="visua_scada()" style="display:none">  </select>
<select id="visualizar_scada2" name="visualizar_scada2" onchange="visua_scada2()"  ></select>
</form>
```

Formulario para la visualización del Scada.

```
<form name="myform3" style="display:none" id="myform3" action="hmiserver/visualizar_scada.php" method
="POST">
<input name="variablesvalor" type="text" id="variablesvalor" style="display:none"  />
<input name="variablesnombre" type="text" id="variablesnombre" style="display:none"  />
<input name="alarmas1" type="text" id="alarmas1" style="display:none"  />
<input name="eventos1" type="text" id="eventos1" style="display:none"  />
<input name="usuario1v" type="text" id="usuario1v" style="display:none"  />
<input name="dcomt" type="text" id="dcomt" style="display:none"  />
<input name="xmlt" type="text" id="xmlt"  style="display:none"  />
<input name="scada1v" type="text" id="scada1" style="display:none"  />
<input name="opcionv" type="text" id="opcionv" value="1" style="display:none"  />
<input name="aceptarscada" type="submit" value="Aceptar" /></form>

</div>
<div align="center">
  <select name="select" id="editar_proceso11" onchange="tipo_proceso()" style="display:none">
  </select>
  <div align="center"></div>

 <form action="hmiserver/hmipages/save5.php" id="f2" name="f2" method="post"  style="display:none"
align="center"><textarea id="content" cols="30" rows="30" name="content" style="display:none">
scadastodosvar ={}</textarea>
 <textarea id="usuario" cols="30" rows="30" name="usuario" style="display:none">gg</textarea>
 <textarea id="tipo1" cols="30" rows="30" name="tipo1" style="display:none">gg</textarea>
 <textarea id="scadastodosvar" cols="30" rows="30" name="scadastodosvar" style="display:none">gg</
textarea>
   <input name="submit" type="submit" style="background-color: #999999" value=" aqui para Guardar"
align="center"/>
</form> </div>
```

Formulario que se encarga de almacenar los datos del HMI para guardarlos en la base de datos

```
 <form action="hmiserver/hmipages/save5.php" id="f2" name="f2" method="post"  style="display:none"
align="center"><textarea id="content" cols="30" rows="30" name="content" style="display:none">
scadastodosvar ={}</textarea>
 <textarea id="usuario" cols="30" rows="30" name="usuario" style="display:none">gg</textarea>
 <textarea id="tipo1" cols="30" rows="30" name="tipo1" style="display:none">gg</textarea>
 <textarea id="scadastodosvar" cols="30" rows="30" name="scadastodosvar" style="display:none">gg</
textarea>
   <input name="submit" type="submit" style="background-color: #999999" value=" aqui para Guardar"
align="center"/>
</form> </div>
```

Aquí se coloca el código de las alarmas y eventos, esto se hace mediante una función en Javascript.

```
 <div id="divXCambiar" align="center" style="display:none; width: 100%;border:1px solid #000000;"></
div>
```

Finaliza el cuerpo de la página y el código HTML

```
</body>
</html>
```

ANEXO L. ARTICULO CIENTIFICO

Bibliografía

1. **W3C.** Simple Object Access Protocol (SOAP). [Online] [Cited: 02 15, 2011.] http://www.w3.org/TR/soap/.
2. **w3c.** Guía Breve de Tecnologías XML. [Online] [Cited: 02 10, 2011.] http://www.w3c.es/divulgacion/guiasbreves/tecnologiasxml.
3. **Joomla.** Bienvenido a Joomla. [Online] [Cited: 01 12, 2010.] http://ayuda.joomlaspanish.org/ayuda-joomla/.
4. **The Python Community.** Python Programming Language – Official Website. [Online] [Cited: 02 12, 2011.] http://www.python.org/ .
5. —. PyOPC. [Online] [Cited: 02 12, 2011.] http://pyopc.sourceforge.net/.
6. **Twisted.** Twisted Matriks Lab. [Online] [Cited: 12 18, 2010.] http://twistedmatrix.com/trac/.
7. **Python.** TkInter. [Online] [Cited: 1 3, 2010.] http://wiki.python.org/moin/TkInter.
8. **J. F. Florez, E. Diaz, y Y. Cabezas.** Simulacion y control en cascada de una planta pomtm en tiempo real con RTAI-Lab". 2008, pp. 852–859.
9. **Scilab.** Scilab The Free Software for Numerical Computation. [Online] [Cited: 02 16, 2011.] http://www.scilab.org/.
10. **P, Schmidt.** *Tutorial Creating a C Function Block in Scicos.* San Francisco, California, : s.n., 2009.
11. PHP. [Online] [Cited: 12 11, 2010.] http://www.php.net/.
12. MYSQL.COM. [Online] [Cited: 11 12, 2010.] www.mysql.com.
13. APACHE. [Online] [Cited: 12 11, 2010.] www.apache.org/.
14. **H.Himmelbauer.** PyOPC A Python Framework for the OPC XML-DA 1.0 Standard,Klosterneuburg. [Online] [Cited: 12 11, 2011.] pyopc.sourceforge.net.
15. **Google Project Hosting.** flot,Attractive Javascript plotting for jQuery. [Online] [Cited: 01 10, 2011.] http://code.google.com/p/flot/.
16. **Google Project Hosting .** svg-edit. [Online] [Cited: 1 5, 2010.] http://code.google.com/p/svg-edit/.
17. **81SAWYER, Peter y KONTOYA, Gerald.** Software requirements. *Software Engineering Book Of Knowledge.* [Online] http://www.swebok.org.
18. *Real-Time Linux Target: A MATLAB-based graphical control environment.* **Rockwell, Geoffrey, y J. Bradley.** s.l. : IEEE Conf. on Control Applications, 1998.
19. **Automata.** Control Basado En PC. [Online] [Cited: 10 12, 2010.] http://automata.cps.unizar.es/Historia/Webs/control_basado_en_pc.htm.
20. **Siemens, R.G.** "A New Computer-Assisted Literary Criticism?", An introduction to A New Computer-Assisted Literary Criticism? s.l. : [A special issue of] Computers and the Humanities 36.3: 259-267, 2002.
21. **Domínguez, T.** *Control automático del proceso productivo.* s.l. : Tecnica Industrial, 2005.
22. **Beckhoff.** Robust industrial design PCs with highest performance components. [Online] [Cited: 12 15, 2010.] http://www.beckhoff.com.
23. *Using labVIEW to prototype an industrial-quality real-time solution for the titan outdoor 4WD mobile robot controller.* **D.Ratner, P.M'Kerrow.** Takamatsu, Japan :

In Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2000. 21428-1433.
24. *Real-time control using Matlab Simulink.* **F.Teng.** Nashville, Tennessee, USA : In IEEE International Conference on Systems, Man, and Cybernetics, 2000. 42697-2702.
25. *OPC Como Alternativa a las Tecnologías Propietarias de Comunicación Industrial.* **J.Lemos, D.Miranda, y A.Arias.** 1, Medellin : ISSN 1657–7663 , 2006, Vol. 3, pp. 5–6.
26. **Systems.NET, OPC.** Open Automation Software. *http://www.opcsystems.com.* [Online] [Cited: 12 15, 2010.]
27. **Openscada.** The OpenSource SCADA System. [Online] [Cited: 11 17, 2010.] http://openscada.org.
28. *RSVIEW soluciones de visualización que le ayudan a alcanzar el éxito.* **Automation, Rockwell.** s.l. : Publicación RSVIEW-BR001A-ES-P, 2005.
29. **Openscada.** The OpenSource SCADA System. [Online] [Cited: 11 12, 2010.] http://openscada.org/.
30. OPC Toolbox . [Online] Matlab. [Cited: 01 02, 2011.] http://www.mathworks.com/products/opc/.
31. *Estrategias docentes colaborativas basadas en la utilización de laboratorios remotos vía Internet.* **Domínguez, M., Fuertes, J.J., Reguera, P., Diez, A.B., Robles, A., Sirgo, J.A.,.** 2006, 14 Congreso Universitario de Innovación Educativa en las Enseñanzas Técnicas EUITI de Gijón.
32. *Virtual and Remote Robotic Laboratory: Comparative Experimental Evaluation.* **Tzafestas, C.S., Palaiologou, N., Alifragis, M.** 3, 2006, IEEE Transactions on education, Vol. 49, pp. 360 - 369.
33. *Estrategias didácticas en el uso de las tecnologías de la información y la comunicación.* **Fandos, M., Jiménez, J.M., Gonzáles, A.P.** 1, 2002, Acción pedagógica, Vol. 11, pp. 28 - 39.
34. Users guide. [Online] Matworks. [Cited: 12 11, 2010.] http://www.mathworks.com/help/techdoc/ref/guide.html.
35. **Ogata, K.** *Ingeniería de Control Moderna.* 2007.
36. **Echeverri, S.** *Tecnología opc para la adquisición de datos de variables industriales.* s.l. : Technical report,Universidad de San Buenaventura, 2005.
37. **SOURCEFORGE.** About OpenOPC. [Online] [Cited: 01 03, 2010.] http://openopc.sourceforge.net/.
38. **Deiretsbacher, K.-H. y E.al.** *Using opc via dcom with windows xp service pack 2.* s.l. : Technical report, OPC Fundation , 2005.
39. *technical information bulletin 04-1 - Supervisory Control and Data Acquisition (SCADA) Systems.* **COMMUNICATION TECHNOLOGIES.** 2004, p. 4.
40. **STELLER, L, QUESADA, M and RETANA, J.** Monografía “Sistemas de control supervisor y de adquisición de datos (SCADA). s.l. : Universidad de Costa Rica, Facultad de ingeniería, Escuela de ingeniería eléctrica, Departamento de Automática, 2001.

41. **SEGURA, JM.** monografia:CRITERIOS DE EVALUACION PARA LA SELECCIÓN DE HERRAMIENTAS SOFTWARE DE CONTROL Y/O SUPERVISION DE PROCESOS INDUSTRIALES - SCADA. s.l. : Universidad del Cauca,Ingenieria Automatica Industrial, 2010.
42. SCADA/HMI Products . [Online] Sielco Sistemi . [Cited: 11 12, 2010.] http://www.sielcosistemi.com/.
43. The HMI & SCADA Revolution. [Online] ATVISE. [Cited: 11 12, 2010.] http://www.atvise.com/.
44. SCADA Systems. [Online] [Cited: 11 12, 2010.] http://www.scadasystems.net/.
45. **Resquín,F.** RTAI – Real Time Application Interface. [Online] [Cited: 11 12, 2010.] www.jeuazarru.com/docs/RTAI.pdf.
46. Introducción a los CMS. [Online] 2007. [Cited: 12 11, 2010.] http://www.alterpime.net/documentos/Introduccion_a_los_CMS.pdf.
47. *Comparing Open Source CMSes: Joomla, Drupal and Plone.* **Bonfield, B and Quinn,S.** s.l. : Idealware, 2007.
48. **Saita, Dr. Fernando A.** CCT CONICET Santa Fe. *CUANDO LA FLUIDODINÁMICA ES EL TEMA.* [Online] Area de Comunicación Social del Ceride, 2003. [Cited: 11 15, 2009.] http://www.santafe-conicet.gov.ar/servicios/comunica/saita.htm.
49. **JSON.** Introducing JSON. [Online] [Cited: 1 5, 2011.] http://www.json.org.
50. **LightOPC.** The Free OPC Server Toolkit. [Online] [Cited: 11 12, 2010.] http://www.ipi.ac.ru/lab43/lOPC-en.html.
51. **MatrikonOpc.** OpcServers. [Online] [Cited: 01 07, 2011.] http://www.matrikonopc.com/.
52. **KEPWARE.** KEPServerEX OPC and Communications Server Features . [Online] [Cited: 12 14, 2011.] http://www.kepware.com/.
53. **John W. Shipman.** Tkinter 8.4 reference: a GUI for Python. [Online] http://infohost.nmt.edu/tcc/help/pubs/tkinter/.
54. **python.** wxPython. [Online] [Cited: 10 24, 2010.] http://www.wxpython.org/.
55. OPC-Fundation. [Online] [Cited: 11 10, 2010.] http://www.opcfoundation.org/.
56. **Mathworks.** Matlab. [Online] [Cited: 01 15, 2011.] http://www.mathworks.com/help/techdoc/ref/guide.html.
57. **National Instruments.** NI LabVIEW. [Online] [Cited: 12 08, 2010.] http://www.ni.com/labview/.
58. **RTAI.** RTAI - the RealTime Application Interface for Linux from DIAPM. [Online] [Cited: 12 08, 2010.] https://www.rtai.org/.
59. **OPC Foundation.** What is the OPC Foundation? [Online] [Cited: 02 16, 2011.] http://www.opcfoundation.org/Default.aspx/01_about/01_history.asp?MID=AboutOPC.
60. —. What is OPC? [Online] [Cited: 02 16, 2011.] http://www.opcfoundation.org/Default.aspx/01_about/01_whatis.asp?MID=AboutOPC.

61. **MathWorks.** OPC Toolbox. [Online] [Cited: 02 16, 2011.] http://www.mathworks.com/help/toolbox/opc/.
62. **ControlGlobal.com.** Integrators must differentiate between a plant-based OPC mentality and SCADA applications when attempting to apply industrial standards to SCADA-based implementations. [Online] [Cited: 02 17, 2011.] http://www.controlglobal.com/articles/2005/441.html.
63. **opcfoundation.** OPC Unified Architecture. [Online] [Cited: 02 16, 2011.] http://www.opcfoundation.org/Default.aspx/01_about/UA.asp?MID=AboutOPC.
64. **Mendiburo.H.** Sistemas Scada. [Online] [Cited: 02 23, 2011.] http://www.galeon.com/hamd/pdf/scada.pdf.
65. **Clickear.com.** Los casos de uso. [Online] [Cited: 11 12, 2011.] http://www.clikear.com/manuales/uml/faseplanificacion.aspx.
66. Clickear.com. [Online] [Cited: 11 12, 2011.] http://www.clikear.com/manuales/uml/faseplanificacion.aspx.
67. **Pyro.** Python Remote Objects (3.x). [Online] [Cited: 02 12, 2011.] http://www.xs4all.nl/~irmen/pyro3/.
68. *Diagramas de Casos de Uso.* **Cáceres., J.** Universidad de Alcalá. Departamento de Ciencias de la Computación. Alcalá de Henares, España : s.n., 2008.
69. **Webelectronica.** SISTEMAS DE CONTROL BASADOS EN PC. [Online] [Cited: 02 12, 2011.] http://www.webelectronica.com.ar/news15/nota09.htm.
70. **SIEMENS.** Control basado en PC Abierto, flexible y fiable. [Online] [Cited: 11 06, 2011.] http://www.siemens.com.br/templates/get_download2.aspx?id=1571&type=FILES.
71. *Requirements management and requirements engineering: You can't have one without the other.* **Nancy, R.** 5, s.l. : Cutter IT Journal, 2000, Vol. 13.
72. —. **Nancy, R.** 5, 2000, Vol. 13.
73. **COCKBURN, Alistair.** Basic Use Case Template. [Online] http://members.aol.com/acockburn>.
74. **Marcia C. F. Carvalho & Zair Abdelouahab.** Um Método para Elicitação e Modemagem de Requisitos Baseado em Objetivos. [Online] 10 08, 2010. http://www.inf.puc-rio.br/wer01/Eli-Req-5.pdf.
75. **Advosol.** Advosol Advanced OPC Solutions. [Online] [Cited: 03 15, 2011.] http://www.advosol.com/.
76. **beijerelectronics.** OPC Server - Gets you connected. [Online] [Cited: 03 15, 2001.] http://www.beijerelectronics.com.
77. **ICONICS.** OPC Connectivity. [Online] [Cited: 3 15, 2011.] http://www.iconics.com/.
78. **dOPC Explorer.** OPC Software Products. [Online] [Cited: 03 15, 2011.] http://www.dopc.kassl.de/.
79. **Opcconnect.** Free Stuff - OPC Clients. [Online] [Cited: 03 11, 2011.] http://www.opcconnect.com/freecli.php.

80. **SOURCEFORGE.NET.** Python Web Services. [Online] [Cited: 3 15, 2011.] http://pywebsvcs.sourceforge.net/.
81. **Whizzywig.** Web based rich text editor for free. [Online] http://unverse.net/Whizzywig-web-based-rich-text-editor.
82. **DRUPAL DEVELOPPEUR.** WampServer. [Online] [Cited: 10 04, 2010.] http://www.wampserver.com/en/.
83. **W3C SVG Working Group.** Scalable Vector Graphics (SVG). [Online] [Cited: 10 01, 2011.] http://www.w3.org/Graphics/SVG/.