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

AMPARO DIAZ MUÑOZ

Monografía de trabajo de grado

Director
Mg. Juan Fernando Flórez Marulanda

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE ELECTRÓNICA, INSTRUMENTACIÓN Y CONTROL
POPAYÁN
2011

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE ILUSTRACIONES

Pág.

LISTA DE ANEXOS

RESUMEN

En esta monografía se presenta el diseño e implementación de una arquitectura OPC basada en *software* libre, concebida para integrar sistemas de control operando en Rtai-Lab, Matlab y adicionalmente con controladores lógicos programables (PLCs), que no está restringida a un único tipo de plataforma de control basada en PC.

Esta arquitectura OPC consta de un servidor y dos clientes: las plataformas de control que actúan como fuente de datos y un sistema de monitoreo y supervisión con fuente de datos OPC, que permite el diseño y monitoreo desde cualquier computador con acceso a Internet.

En el capítulo 1 se presentan las características: de las plataformas de control basadas en Computador Personal (PC), del protocolo de comunicaciones industriales OPC y de los sistemas de monitoreo y supervisión. En el capítulo 2 se realiza un diseño y modelado de la arquitectura de integración OPC. En el capítulo 3 se realiza la implementación del servidor OPC XML y los clientes en las plataformas de control. En el capítulo 4 se presenta la implementación de un sistema de monitoreo y supervisión con soporte OPC. En el capítulo 5 se describe la integración del sistema y las pruebas realizadas en el laboratorio de control de procesos del Programa de Ingeniería en Automática Industrial (PIAI).

Finalmente en el capítulo de conclusiones y recomendaciones se consignan las experiencias del desarrollo del proyecto y las expectativas para trabajos futuros.

Palabras claves: Control basado en PC, OPC, Sistema de monitoreo y supervisión.

INTRODUCCIÓN

Desde sus inicios, los sistemas digitales influenciaron en forma extrema el desarrollo de muchas actividades. En aplicaciones de control a nivel de laboratorio e investigación en la academia, y en algunos casos en la industria, es común el empleo de plataformas de control basados en computadores convencionales (PC), no solo para la realización del control de procesos, sino también para la integración con otros niveles de la pirámide de automatización que se presenta en las empresas de manufactura. Particularmente a nivel de laboratorio e investigación en la academia estas plataformas de control se soportan en herramientas comerciales conocidas como: Matlab (1) y Labview (2) o en *software* libre como Rtai–Lab (3).

Por otra parte la necesidad de interoperabilidad entre los equipos de automatización, ha impulsado una unificación masiva de los sistemas industriales, por parte de los principales desarrolladores de *software* para automatización y control de procesos industriales. La fundación OPC, es una de las principales organizaciones preocupada por reducir los problemas de compatibilidad entre los diferentes protocolos de redes y sistemas industriales existentes (4), esta fundación se encarga de administrar el conjunto de especificaciones OPC (OLE para el control de procesos). OPC inicialmente se soportó en el estándar DCOM de Microsoft, actualmente, ha evolucionado con los estándares OPC XML y OPC UA permitiendo la interoperabilidad de múltiples sistemas operativos, y herramientas avanzadas de monitoreo y control en plantas industriales (5). OPC también ha permitido el monitoreo y supervisión de datos medidos en campo de forma remota con diversas tecnologías, lo cual ha generado grandes resultados en la industria de la automatización, puesto que se tiene un control total sobre los procesos de control (6).

Así, en este proyecto denominado arquitectura para la integración de plataformas de control basadas en PC mediante OPC, se presenta el diseño e implementación de un sistema que permite integrar diversas plataformas de control como son Matlab, Rtai-Lab y adicionalmente PLC. La integración de estas plataformas de control se la realiza mediante dos estándares OPC: DCOM y XML, donde se aporta la implementación de dos clientes OPC en Rtai-Lab (con especificaciones DCOM y XML), un cliente OPC XML en Matlab, un servidor OPC XML multiplataforma que permite una comunicación industrial con todos los clientes OPC XML independientemente de cual sea su fabricante, y adicionalmente un sistema de monitoreo web con soporte OPC.

La arquitectura implementada consta de siete (7) módulos: módulo servidor OPC DCOM, módulo servidor OPC XML, módulo cliente OPC DCOM en Matlab, módulo OPC XML en Matlab, módulo OPC DCOM en Rtai-Lab, módulo XML en Rtai-lab y módulo de monitoreo y supervisión web.

Para el módulo servidor OPC DCOM se realizo un análisis de disponibilidad de servidores OPC DCOM existentes en el mercado, concluyéndose que existen soluciones tanto comerciales como libres que permiten comunicar las plataformas de control, pero se trabaja con el servidor KepseverEX por sus características técnicas, y por el conocimiento que se tiene de este servidor en el laboratorio de control de procesos del PIAI. Para el módulo servidor OPC XML se construye un servidor de acceso a datos, que traduce los datos provenientes de las fuentes datos (Matlab y Rtai-Lab) para que sea compatible con la especificación OPC XML DA. Para el módulo cliente OPC DCOM en Matlab se trabaja con librerías existentes en la biblioteca de Matlab, las cuales permiten la lectura y escritura de objetos ítems en los servidores OPC DCOM. Para el módulo OPC XML en Matlab se construye un programa que permite comunicar a Matlab/Simulink con los servidores OPC XML. Para los módulos cliente OPC en Rtai-lab se implementan cuatro bloques (Dcom_Read, Dcom_Write, Xml_Read, Xml_Write) en Scicos/Scilab que permiten la comunicación de Rtai-Lab con los servidores OPC en las tecnologías OPC DCOM DA y OPC XML DA. El módulo de monitoreo y supervisión soportado en tecnología web, permite el diseño de diagramas representativos del proceso (mímicos) con información real del proceso para el monitoreo del mismo desde un equipo cliente con acceso a internet, este módulo ofrece las siguientes funcionalidades: procesar datos, diseñar pantallas, representación de señales de alarma, visualización gráfica dinámica de las variables de proceso y un módulo gestor de contenido, que posibilita la actualización, mantenimiento y ampliación de la web con la colaboración de múltiples usuarios.

Los módulos diseñados en este proyecto están conformados por elementos de programación, base de datos, y lógicas de control basadas en estándares abiertos reduciendo así sustancialmente los costos de implementación.

Finalmente se logró la integración de Matlab y Rtai-Lab de dos maneras diferentes: mediante el estándar OPC DCOM y mediante el estándar OPC XML, pero en forma adicional se logró integrar al conjunto anterior el PLC micrologix 1500 por medio del estándar DCOM. El sistema se validó usando las plantas de nivel y presión existentes en el laboratorio de control de procesos, y una planta virtual implementada en Matlab.

1. FUNDAMENTOS TEORICOS

1.1. CONTROL BASADO EN PC

El conjunto conformado por una computadora personal con hardware para adquisición/envío de datos (digitales y analógicos) y comunicación con redes industriales, *software* de control para PC y equipos de instrumentación y control automático se denomina: "sistema de control basado en PC" (7), (8). Estos sistemas posibilitan el desarrollo de aplicaciones de control en lenguajes de programación de PC, que se comunican directamente con elementos conectados a un bus industrial como: módulos de entrada/salida, variadores de velocidad, terminales de visualización, etc (9).

Entre las ventajas que presenta el control basado en PC se encuentran: el manejo, visualización y procesamiento de datos para tareas de automatización, flexibilidad para integración de componentes de diferentes fabricantes a nivel *software* y hardware, trabajo en redes industriales por medio de interfaces de datos como OPC, además de la posibilidad de implementación en sistemas de bajo nivel y los bajos costos de desarrollo y aplicación en el área de procesos continuos con respecto a un sistema de control distribuido (DCS) (10), (11).

1.1.1. Partes de un sistema de control basado en PC

Un sistema de control basado en PC está formado por tres partes básicas: la computadora personal con su hardware y *software* de base asociados, el *software* para control basado en PC, y el o los dispositivos de entrada y salida. Una característica importante de un sistema de control basado en PC es que cada una de estas partes es un producto distinto, usualmente diseñado y comercializado por proveedores diferentes (12).

El *software* para control basado en PC, también conocido como: plataforma de control (término usado de ahora en adelante en este documento para reconocerlo), *software* de supervisión, *software* para fuente de datos o *software* para control de procesos, está específicamente diseñado para su uso en computadoras personales estándar comunicadas con multitud de equipos industriales. Se caracteriza por un alto grado de adaptabilidad a los condicionamientos de las demás partes, buscando adaptarse a la mayor cantidad posible de equipos digitales de control industrial.

Ni la computadora o el *software* permiten la conexión de elementos de campo en forma directa. Para ello se utilizan dispositivos de E/S (Entrada/Salida), equipos digitales que toman la señal del instrumento de campo, realizándole procesos de digitalización, multiplexación y transmisión a la PC.

Si bien, en la mayor parte de las aplicaciones de los sistemas de control basado en PC se utiliza computadoras personales, aparecen con alguna frecuencia casos en los que éstas son reemplazadas o complementadas con computadoras de mayor capacidad. Hay que destacar que la definición de sistema de control basado en PC, que se ha dado, no distingue el lugar en el que reside el algoritmo de control, dado que se puede ejecutar en la PC o en un dispositivo de entrada o salida.

1.1.2. Plataformas de control basadas en PC

Una plataforma de control basada en PC se constituye básicamente de cuatro (4) programas y una base de datos dinámica. El primer programa permite que el valor de sus variables cambie instante a instante en forma asociada a las variables del proceso. Otro programa permite crear las pantallas que posibilitan al usuario ver la información de la base de datos, un tercer programa se ocupa de ejecutar la estrategia de control, y un cuarto programa, el de visualización, que toma las pantallas anteriormente creadas y las “conecta” a la base de datos dinámica (12).

Existen plataformas de control basadas en PC, destinadas para laboratorio o entornos industriales. Las primeras van enfocadas para procesos académicos o de investigación y emplean PC de características estándar; mientras que las segundas emplean computadores industriales (IPC), los cuales se encuentran dotados de características especiales como: mayor modularidad con respecto a componentes hardware y capacidad en el manejo de aplicaciones *software* específicas en tiempo real, además de mejores protecciones ante ambientes hostiles con ruido electromagnético y condiciones ambientales extremas, haciéndolo una opción robusta que puede reemplazar o complementar a la solución realizada con un controlador lógico programable (PLC).

Dentro de las diferentes empresas que ofrecen plataformas de control basadas en PC industriales se encuentran:

- ***Siemens:*** ofrece productos como *SIMATIC WinAC RTX* con interfaces hombre máquina, paneles de operación táctiles y de membrana, entre otros. Integración con los estándares OPC XML y DCOM (13).

- ***Beckhoff:*** fue el pionero de los sistemas de control basado en PC, actualmente, implementa la nueva especificación OPC UA en sus controladores basados en PC (14).

1.1.3. Plataformas académicas de control basadas en PC

Debido a los avances tecnológicos a nivel de instrumentación y control que contribuyen constantemente con el mejoramiento a nivel productivo en las

empresas de manufactura, y a la cada vez más frecuente aplicación de estrategias de control complejas y robustas que integran todas las tecnologías mencionadas; es necesario familiarizar a los estudiantes con los ambientes industriales actuales y las estrategias de control que se aplican en ellos, para que de esta manera logren un nivel competitivo de conocimiento y práctica, acorde con las exigencias del mercado laboral. Para este efecto, existen plataformas académicas de control privativas como *Labview,* Matlab y versiones de distribución libre y código abierto como *Rtai-Lab.*

- ***Labview*:** de la empresa *National Instruments,* se utiliza para crear sistemas de control distribuido (DCS) mediante programación gráfica, ofreciendo una integración con dispositivos *hardware* para control de instrumentos de cualquier fabricante, capacidad de interacción con otros componentes *software*, comunicación con interfaces de comunicación, entre las que se encuentra el estándar OPC – DCOM (15).

- ***Matlab*:** de la empresa *MathWorks,* es un completo entorno de programación de lenguaje de alto nivel, que permite desarrollar una amplia gama de aplicaciones, desde procesamiento digital de señales e imágenes hasta diseño de control, el cual, junto con el programa de simulación digital *Simulink*, son ampliamente utilizados en los textos académicos de control. Matlab actualmente cuenta con librerías que permiten la comunicación de con servidores OPC –DCOM (16).

- ***Rtai–Lab*** (17)**:** proporciona una serie de aplicaciones en forma de diagramas de bloques como: interfaz gráfica, procesamiento numérico, *drivers* de comunicación para el desarrollo de sistemas de monitoreo y control; que pueden ser compilados y ejecutados en el sistema operativo de tiempo real Linux/RTAI. Rtai–Lab representa una valiosa opción dentro de las herramientas de fuente abierta para ejecutar, diseñar, sintonizar e implementar sistemas de control en tiempo real dentro de un ambiente Linux para las instituciones educativas de los países tercermundistas como de los países desarrollados. Rtai-Lab está basado en las siguientes herramientas (18):

  - ✓ *Scilab/Scicos*: *Scilab* es un *software* CACSD **(***Computer Aided Control System Design* – Diseño de sistemas de control asistido por computadora) de fuente abierta para computación numérica. *Scilab* incluye *Scicos* un editor de diagrama de bloques que puede ser usado para crear simulaciones y generar automáticamente código compilado implementando tareas en tiempo real. La última versión usada de *Scilab/Scicos* es la 4.1.2.
  - ✓ *Rtai–Lib*: es una paleta de bloques de *Scicos* que permite diseñar diagramas de bloques con sensores y actuadores. Esta provee una interfaz entre RTAI y el hardware de adquisición de señales. Los diagramas de

bloques que usa Rtai–Lab pueden ser compilados dentro del *software Scilab/Scicos*.

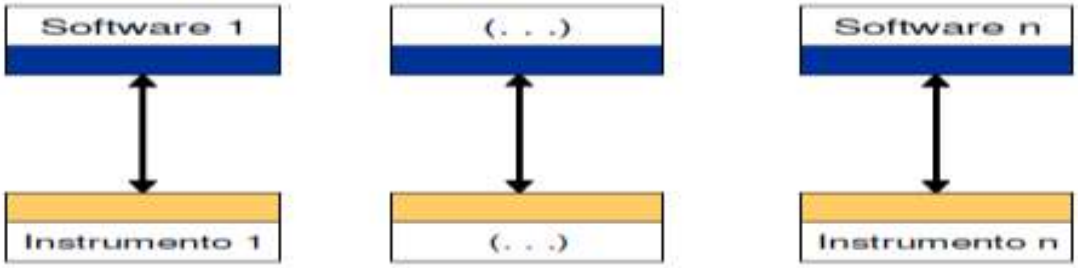
- ✓ *Xrtailab*: es un *software* que emula un osciloscopio que se puede conectar a procesos que se ejecutan en tiempo real. Esto permite visualizar y monitorear señales y eventos de tiempo real usando indicadores, *scopes* y *leds* simulados. *Xrtailab* también permite ajustar parámetros de las aplicaciones de tiempo real mientras se está ejecutando el código de tiempo real. Esta aplicación hace parte de RTAI.
- ✓ *Comedí:* proporciona los *drivers*, librerías de funciones y una API *(Application Program Interface* – Interfaz de programación de aplicaciones*)* para interactuar con cientos de dispositivos hardware de adquisición de señales.

1.2. FUNDAMENTOS DE OPC

OPC es una especificación técnica no propietaria definida por la fundación OPC consiste básicamente en un sistema de interfaces estándar basado inicialmente en OLE/COM de *Microsoft*. Con OPC es posible inter–operar dispositivos industriales con sistemas de información o aplicativos de escritorio (5).

La Figura 1-1 representa la arquitectura simplificada de un sistema de control distribuido tradicional, en este se ve como cada aplicación de usuario debe tener su propio *driver* para comunicarse con los dispositivos físicos. OPC es un estándar de comunicación que garantiza la comunicación entre dispositivos de distintos fabricantes, permitiendo la comunicación entre aplicaciones de control y de supervisión con independencia de la red en la que trabaje.

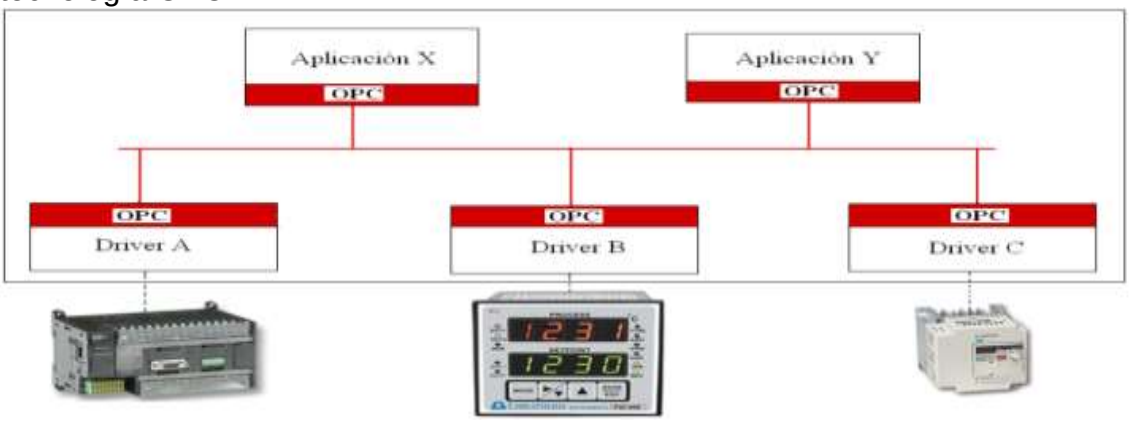
Figura 1-1: Sistema tradicional de control distribuido de procesos



Fuente: (5)

En la Figura 1-2, se muestra la arquitectura simplificada de un sistema de control distribuido que incorpora tecnología OPC. La creación de aplicaciones bajo ésta tendencia se hace mucho más simple, pues el desarrollo de complicados *drivers* de comunicación se ve reemplazada por la simple utilización de llamadas a procedimientos locales o remotos entre la aplicación de usuario y el cliente OPC.

Figura 1-2: Sistema de control distribuido de procesos que incorpora una tecnología OPC



Fuente: (5)

1.2.1. Arquitectura OPC (19)

La figura 1-3 representa la arquitectura OPC, la cual es un modelo cliente – servidor que brinda una plataforma para extraer datos de una fuente y a través de un servidor comunicarlos a cualquier aplicación cliente de manera estándar. Los componentes OPC se pueden clasificar en dos categorías:

- ***cliente OPC:*** un cliente OPC puede comunicarse con cualquier servidor OPC sin importar el tipo de objeto ítem que recoge esos datos.
- ***servidor OPC:*** es una aplicación que permite el acceso a objeto ítems de un sistema automatizado (datos de campo) desde otras aplicaciones (clientes OPC).

Figura 1-3: Esquema de una Arquitectura OPC

Fuente: **(5)**

Cada uno de los elementos del modelo cliente–servidor se presentan a continuación:

1. Comunicaciones OPC cliente - OPC servidor: un servidor OPC puede soportar comunicaciones con múltiples clientes OPC simultáneamente.

2. OPC Server - traducción de datos/mapeo: la función principal de un servidor OPC es la de traducir los datos provenientes de la Fuente de Datos, de un protocolo propietario en el formato OPC, para que sea compatible con una o varias especificaciones OPC.

3. Comunicaciones servidor OPC - fuente de datos: los servidores OPC se comunican con las fuentes de datos como pueden ser: dispositivos, controladores, aplicaciones, etc. La Fundación OPC no específica la forma como un servidor OPC debe comunicarse con una fuente de datos debido principalmente a la enorme variedad. Cada dispositivo, controlador o aplicación diferente utiliza un protocolo distinto que incluso puede comunicar sobre distintos medios físicos (Serie RS485 o RS232, *Ethernet, Wireless*, etc).

1.2.2. Especificaciones OPC

OPC define varias interfaces desarrolladas para un determinado campo de aplicación como: *OPC DA (Data Access), la cual* permite leer, modificar y monitorear variables del proceso; OPC A&E (*Alarm&Event*) que sirve para retransmitir alarmas a cualquier cliente OPC; OPC HDA (*Historical Data Access*) que sirve para acceder a todos los valores del proceso contenidos en la base de datos. En general OPC permite intercambiar datos no críticos en el tiempo, entre sistemas autónomos o dispositivos de distintos fabricantes que utilizan Ethernet como medio de comunicación común.

Dentro de las especificaciones OPC existentes se encuentran:

- ***OPC DCOM-DA*** *(Data Access):* fue la primera especificación creada por la fundación OPC en Agosto de 1996, la cual permite leer, modificar y monitorizar variables del proceso. La desventaja de esta tecnología es que sólo está disponible para computadoras con un sistema operativo de Microsoft (5).

- ***OPC XML-DA*** (XML acceso a datos)*:* estándar XML desarrollado por la Fundación OPC para resolver el problema de interoperabilidad, fue promocionado en Julio de 2003*.* El rendimiento del OPC XML DA es mucho menor que el del OPC DA (5).

- ***OPC-UA*** (Arquitectura Unificada)*:* representa las especificaciones más recientes para control de procesos e inter-conectividad de sistemas automáticos. OPC UA depende de los servicios web para el transporte de datos. Fue publicado en agosto de 2006 (20).

1.2.3. Modelo jerárquico OPC (5)

La figura 1-4 representa el modelo jerárquico de objetos definidos por la fundación OPC. Los objeto ítems definidos para este modelo son:

- ***El objeto servidor:*** contiene información sobre la configuración del servidor OPC y sirve de contenedor para los objetos tipo grupo.
- ***El objeto grupo:*** sirve para organizar los datos que leen y escriben los clientes permitiendo establecer conexiones por excepción entre los clientes y los objetos ítems de un grupo. Un grupo puede ser público, es decir, compartido por varios clientes OPC.
- ***El objeto ítem:*** este objeto representa conexiones a fuentes de datos en el servidor y cuenta con los siguientes atributos: *Value, Quality y Time Stamp*. Los accesos a los objetos ítems OPC se hacen a través de los grupos OPC y los clientes pueden definir el ritmo al cual el servidor les informa sobre cambios en los datos.

Figura 1-4: Modelo de objetos OPC

Fuente: Modificado de **(5)**

1.3. SISTEMAS DE MONITOREO Y SUPERVISIÓN (21), (22), (23)

Una de las partes más importantes de los sistemas de automatización, y que forma parte del último nivel de supervisión, lo constituye el sistema conocido como *Supervisory Control And Data Acquisition* (SCADA).

Los programas que constituyen a un sistema *SCADA*, además de visualizar datos de planta o de campo, poseen varios componentes que le permiten: establecer alarmas, visualizar tendencias de las variables medidas, comunicarse con los dispositivos de campo, generar datos históricos y otras funciones para cumplir con el objetivo establecido anteriormente. La figura 1-5 ilustra algunos de los principales componentes de un *software* SCADA.

Figura 1-5: Componentes de un *software* SCADA

Fuente: (24)

1.3.1. Funciones principales de un sistema de monitoreo y supervisión (24)

Dentro de las principales funciones de un sistema de monitoreo y supervisión se encuentran:

- ***Supervisión remota de instalaciones y equipos****:* brinda la posibilidad de conocer el estado de desempeño de las instalaciones y los equipos alojados en la planta, permitiendo a los operadores dirigir tareas de mantenimiento y manejar estadísticas de fallas.
- ***Control remoto de instalaciones y equipos****:* mediante el sistema, se puede activar o desactivar los equipos de forma automática o manual desde un lugar remoto, por ejemplo: abrir válvulas, activar interruptores, prender motores, etc., desde la sala de control de una planta de manufactura. Además, es posible ajustar parámetros, valores de referencia, algoritmos de control, etc.
- ***Procesamiento de datos:*** en esta función, se procesan y analizan el conjunto de datos adquiridos en planta o en campo que conforman la información que alimenta el sistema. Luego, se compara con datos anteriores y de otros puntos de referencia, obteniendo como resultado una información confiable y veraz.
- ***Visualización gráfica dinámica:*** el sistema es capaz de brindar imágenes en movimiento, que representen el comportamiento del proceso. Esto le da al operador, la impresión de estar presente dentro de una planta real. Los gráficos también pueden corresponder a curvas de las señales analizadas en el tiempo.

- ***Generación de reportes:*** el sistema permite generar informes con datos estadísticos del proceso, en un tiempo determinado por el operador.
- ***Representación de señales de alarma:*** a través de las señales de alarma se logra alertar al operador frente a una falla o la presencia de una condición perjudicial o fuera de lo aceptable, estas señales pueden ser tanto visuales como sonoras.
- ***Almacenamiento de información histórica:*** se cuenta con la opción de almacenar los datos adquiridos, para su posterior procesamiento, análisis y generación de información. El tiempo de almacenamiento dependerá del operador o del autor del programa.
- ***Programación de eventos:*** esta función se refiere, a la posibilidad de programar subprogramas que brinden automáticamente: reportes, estadísticas, gráfica de curvas, activación de tareas automáticas, etc.

1.3.2. OPC en sistemas de monitoreo y supervisión

Las comunicaciones dentro de cualquier sistema industrial, juegan un papel primordial en el momento de su implementación y funcionamiento, principalmente si sus componentes se encuentran dispersos dentro de la planta o geográficamente distantes. Se deben tener en cuenta aspectos de seguridad, distancias entre dispositivos, volumen y tipo de datos intercomunicados, entre otros, para seleccionar el tipo de medio de comunicación (25).

Debido a esto, se han ido desarrollando sistemas de monitoreo y supervisión que cumplen con protocolos y estándares como OPC, brindando niveles de confiabilidad suficientes, para garantizar la salida y la llegada de un mensaje y así considerar la interoperabilidad del equipamiento de diversos fabricantes.

Entre algunos sistemas de monitoreo y supervisión remota con soporte OPC de carácter comercial se encuentran:

- ***OPC Systems.NET:*** incluye una gama de productos.net para SCADA, además, comparte datos con servidores OPC, *SQL Server, Oracle, Access,* y *MySQL* (26).

- ***Sielco Sistemi:*** es una empresa italiana que ofrece soluciones SCADA/HMI, siendo su *software* SCADA de fácil uso, flexible y de bajo costo. Ofrece soluciones SCADA para la industria y la automatización de edificios (27).

- ***ATVISE:*** presenta en el mercado, una solución *software* HMI, totalmente equipada con tecnología web y brinda soporte OPC – UA (28).

- ***RSview Enterprise:*** abarca desde aplicaciones a nivel de máquina autónomas, hasta programas HMI de nivel supervisor distribuidos. Esta solución de visualización proporciona valor desde el diseño inicial y la instalación, hasta las operaciones y el mantenimiento continuo del sistema (29).

Dentro del *software* de código abierto se encuentran algunos sistemas de monitoreo y supervisión con soporte OPC:

- ***Freescada:*** es un sistema SCADA para Windows (2000/XP/Vista) que proporciona a los usuarios finales, herramientas flexibles para la visualización y el control interactivo de cualquier proceso industrial. El sistema utiliza servidores OPC – DCOM para la adquisición de datos (30).

- ***Openscada:*** es un sistema diseñado en java para el control de procesos. El protocolo de comunicación de base es OPC DCOM (31).

- ***Scada System:*** sistema basado en SCILAB que incluye: OPC, *Modbus, Mysql,* interfaz TCP/IP, y la simulación de controladores PID y control *Fuzzy* ó difuso (32).

1.4. RESUMEN

En este capítulo se realiza la fase de documentación y recolección de información básica para el desarrollo de este proyecto. Inicialmente se detalla el control basado en PC el cual posibilita el desarrollo de aplicaciones de control en lenguajes de programación de PC, luego se explica el estándar OPC como sistema de interfaz estándar que permite interoperar dispositivos industriales con sistemas de información, y finalmente se describen los sistemas de monitoreo y supervisión web.

2. DISEÑO Y MODELADO DE LA ARQUITECTURA

En este capítulo, se describen todas las propiedades y características que tendrá la arquitectura para la integración de plataformas de control basadas en PC mediante OPC. Inicialmente se presentan los objetivos, alcances y limitaciones que debe cumplir la arquitectura propuesta, incluyendo una descripción general de los subsistemas que la constituyen. Luego se describe la metodología empleada para obtener los requerimientos, se obtienen los requerimientos básicos y de usuarios, se realiza su análisis para definir los requerimientos funcionales y no funcionales necesarios para la implementación de cada módulo de la arquitectura. Finalmente se realiza un resumen del diseño de la arquitectura propuesta.

2.1. OBJETIVOS

2.1.1. Objetivo general

Diseñar e implementar una arquitectura de integración, monitoreo y supervisión remota para plataformas de control basadas en Matlab y Rtai-Lab, mediante OPC.

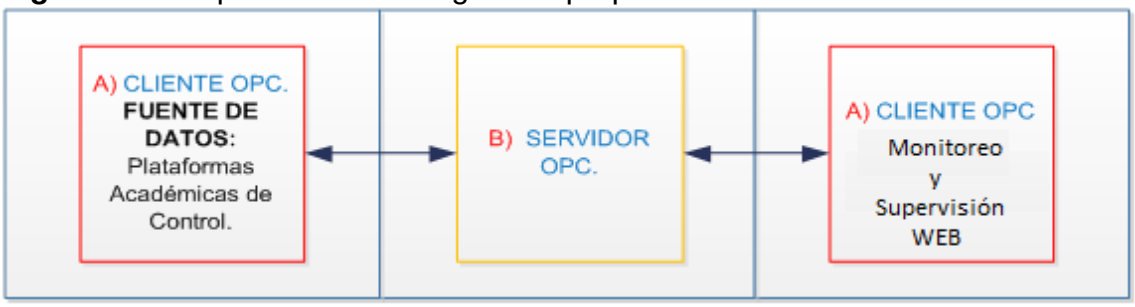
2.1.2. Objetivos específicos

- Diseñar una arquitectura de integración para plataformas de control basadas en Matlab y Rtai-Lab mediante OPC.
- Diseñar e implementar un servidor OPC con soporte de datos desde Matlab y Rtai-Lab.
- Diseñar e implementar un sistema de monitoreo y supervisión remota con soporte OPC.
- Comprobar la comunicación OPC de la arquitectura de integración con una aplicación de control basada en PC en el laboratorio de control de procesos.

2.2. DESCRIPCION GENERAL DE LA ARQUITECTURA DE INTEGRACION PARA LAS PLATAFORMAS DE CONTROL

La figura 2-1 ilustra la arquitectura OPC propuesta, basado en la arquitectura OPC especificada en el numeral 1.2.2 del capítulo 1 de este documento. Este es un modelo cliente – servidor que extrae datos de las plataformas académicas de control y, a través de un servidor OPC, los comunica a un módulo de monitoreo y supervisión web.

Figura 2-1: Arquitectura de integración propuesta



Fuente: propia

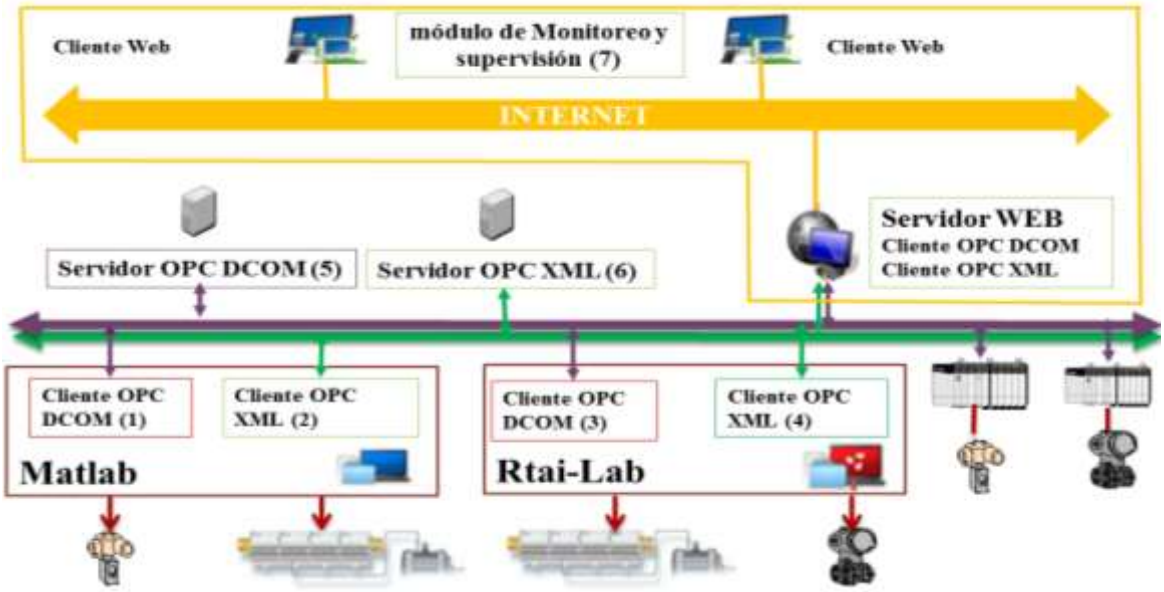
Según la figura 2-1, la arquitectura consta de tres subsistemas básicos:

A. Comunicaciones entre las fuentes de datos y el servidor OPC: las fuentes de datos son las plataformas académicas de control establecidas para este proyecto, quienes envían datos al servidor OPC mediante módulos cliente.
B. Servidor OPC: el servidor OPC traduce los datos provenientes de las plataformas académicas de control en el formato OPC, para que sea compatible con una o varias especificaciones OPC.
C. Comunicaciones entre el servidor OPC y el sistema de monitoreo y supervisión web: el sistema de monitoreo y supervisión toma los datos del servidor mediante un cliente OPC y se encarga de visualizarlos.

La figura 2-2 representa los componentes de la arquitectura para la integración de plataformas de control basadas en PC mediante OPC. En esta arquitectura, se cuenta con las plataformas de control Matlab y Rtai-Lab, que se encargan de realizar la fuente de datos (línea roja), efectuar el control y enviar los datos mediante cuatro módulos clientes: cliente DCOM Matlab (ver recuadro 1 en la figura 2-2), cliente XML Matlab (ver recuadro 2), cliente DCOM Rtai-Lab (ver recuadro 3), cliente XML Rtai-Lab (ver recuadro 4), los módulos cliente OPC DCOM envían los datos (línea café) al servidor OPC DCOM (ver recuadro 6) y los módulos OPC XML, a su vez envían los datos (línea morada) al servidor OPC XML (ver recuadro 5). Cada uno de los servidores se encarga de traducir los datos al formato OPC estándar y enviarlos a los clientes consumidores que en este caso es un módulo de monitoreo y supervisión web (ver recuadro 7) con soporte OPC DCOM y XML. Este sistema de monitoreo y supervisión se compone de un servidor web, el cual obtiene los datos mediante los clientes OPC DCOM y XML implementados en el mismo, y se encarga de distribuir a clientes web (línea naranja), desde donde los operarios pueden realizar diseño de mímicos y monitoreo/supervisión de las plantas de automatización. Así mismo las plataformas académicas de control se pueden comunicar con otros dispositivos de control que soporten el estándar OPC (PLC y otros sistemas de monitoreo y supervisión) mediante los servidores. Es importante resaltar que para integrar Matlab y Rtai-Lab solo se puede hacer usan-

do un único estándar, ya sea el DCOM o el XML. Por ejemplo si el estándar a utilizar es el DCOM la comunicación se realizaría entre los módulos 1, 3 y 6.

Figura 2-2: Componentes de la Arquitectura de integración para Matlab y Rtai-Lab mediante OPC



Fuente: propia

De la figura 2-2 se puede determinar que la arquitectura de integración para las plataformas de control, se compone de siete (7) módulos los cuales se detallan a continuación:

- ***Módulo cliente OPC DCOM en Matlab (MCODM):*** este módulo contará con un proceso de control capaz de actuar como cliente OPC DCOM para el acceso a los datos, que leerá los datos provenientes de Matlab y/o Simulink y los enviará al servidor OPC DCOM.
- ***Módulo cliente OPC XML en Matlab (MCOXM):*** este módulo contará con un proceso de control capaz de actuar como cliente OPC XML para el acceso a los datos, que leerá los datos provenientes de Matlab y/o Simulink y los enviará al servidor OPC XML.
- ***Módulo cliente OPC DCOM en Rtai-Lab (MCODR):*** este módulo contará con un proceso de control capaz de actuar como cliente OPC DCOM para el acceso a los datos, que leerá los datos provenientes de Rtai-Lab específicamente desde Scilab/Scicos y los enviará al servidor OPC DCOM.
- ***Módulo cliente OPC XML en Rtai-Lab (MCOXR):*** este módulo contará con un proceso de control capaz de actuar como cliente OPC XML para el acceso a

los datos, que leerá los datos provenientes de Rtai-Lab específicamente desde Scilab/Scicos y los enviará al servidor OPC XML.

- ***Módulo servidor OPC DCOM (MSOD):*** traduce los datos provenientes de las fuentes de datos (Matlab y Rtai-Lab) para que sea compatible con la especificación OPC DCOM - DA.
- ***Módulo servidor OPC XML (MSOX):*** traduce los datos provenientes de las fuentes de datos (Matlab y Rtai-Lab) para que sea compatible con la especificación OPC XML DA.
- ***Módulo de monitoreo y supervisión web (MM&SW):*** captura las variables de los procesos de Matlab y Rtai-Lab mediante la comunicación con los servidores OPC DCOM/XML y los representa gráficamente.

2.3. ALCANCES Y LIMITACIONES DEL PROYECTO

Considerando los objetivos establecidos en la sección 2.1, para el desarrollo de la arquitectura de integración mediante OPC, se establecen por defecto dos plataformas académicas de control: Matlab y Rtai-Lab. La razón es que son dos importantes herramientas *software* utilizadas en el programa de Ingeniería en Automática Industrial de la universidad del Cauca, para la enseñanza de los sistemas de control a nivel continuo, digital y de procesos.

Para establecer la comunicación entre las plataformas de control y el servidor OPC, se trabajará con OPC DA (*Data Access*) en los estándares OPC DCOM y OPC XML. En cuanto al estándar OPC-UA no se logró realizar un estudio en profundidad de su funcionamiento, ya que los códigos de ejemplo están restringidos por la fundación OPC, por lo que no se consideró su utilización dentro del proyecto.

Igualmente están por fuera del alcance del presente proyecto utilizar las interfaces OPC A&E (*Alarm&Event*), OPC HDA (*Historical Data Access*), ver sección 1.2.3, dado que dentro de los objetivos del trabajo de grado la prioridad es realizar el acceso a datos y no el manejo de alarmas y de históricos.

2.3.1. Alcances para el módulo cliente OPC DCOM en Matlab (MCODM)

Matlab cuenta con tres formas de comunicarse con servidores OPC (33): mediante funciones de una librería OPC, interfaz gráfica GUI y *OPC-Toolbox*, por lo tanto para integrar Matlab y Rtai-Lab mediante el estándar DCOM se puede hacer uso de cualquiera estos elementos de comunicación.

2.3.2. Alcances para el módulo cliente OPC XML en Matlab (MCOXM)

Matlab no cuenta con un módulo cliente OPC XML, por esta razón se implementa un *software* que permita el intercambio de información entre programas diseñados en Matlab/Simulink y el servidor OPC XML.

2.3.3. Alcances para el módulo cliente OPC DCOM en Rtai-Lab (MCODR)

Rtai-Lab no cuenta con un módulo cliente OPC DCOM, por esta razón se implementa un *software* que permita el intercambio de información entre programas diseñados en Scilab/Scicos y el servidor OPC DCOM.

2.3.4. Alcances para el módulo cliente OPC XML en Rtai-Lab (MCOXR)

Rtai-Lab no cuenta con un módulo cliente OPC XML, por esta razón se implementa un *software* que permita el intercambio de información entre programas diseñados en Scilab/Scicos y el servidor OPC XML.

2.3.5. Alcances para el módulo servidor OPC DCOM (MSOD)

Se realizó un análisis de disponibilidad de servidores OPC DCOM existentes en el mercado, concluyéndose que existen soluciones tanto comerciales como libres. Entre las empresas que ofrecen servidores OPC DCOM están: *Kepware Communications* (34), *Advosol* (35), *Beijer Electronics* (36), *Iconics* (37), entre muchos otros. Los servidores comerciales explorados en este proyecto son: *KepServerEx* (34), *MatrikonOPC* (38), y en los servidores OPC de codigo abierto se encuentra *LightOPC* (39). Teniendo en cuenta la anterior información, se concluye que para comunicar las plataformas de control mediante el estandar DCOM se puede hacer uso de cualquiera de estos servidores.

2.3.6. Alcances para el módulo servidor OPC XML (MSOX)

La disponibilidad de los servidores OPC XML es mínima por lo tanto se diseña e implementa un servidor que permita traducir los datos provenientes de las fuentes de datos (Matlab y Rtai-Lab), para que sea compatible con el estándar OPC XML DA. No está dentro del alcance de este módulo realizar la fuente de datos desde otros dispositivos de control, como PLC, por medio de esta especificación.

2.3.7. Alcances para el módulo de monitoreo y supervisión web (MM&SW)

Existen herramientas *software* tanto comerciales como libres que permiten el monitoreo y supervisión de procesos, como se indicó en la sección 1.3.3, que cumplen más o menos con todos las características de los sistemas de monitoreo y supervisión explicadas en 1.3.1. Analizando el nivel de portabilidad y accesibilidad que presentan estas herramientas; se ve que una de sus limitaciones es la necesidad del usuario de instalar la aplicación en cada cliente donde se

requiera utilizar el SCADA, y considerando que no siempre se contará con la herramienta para su instalación, con el sistema operativo que la soporte o con la interfaz desarrollada en aquellas aplicaciones, entre otras limitantes, entonces se plantea en este proyecto implementar un sistema de monitoreo y supervisión, soportado en tecnología web, que permita el diseño de mímicos y monitoreo de procesos desde un equipo cliente con acceso a internet, sin necesidad de instalar *software* alguno adicional, pues este estará instalado en el servidor web.

Con respecto al desarrollo del módulo de monitoreo y supervisión se abarcarán los siguientes funcionalidades (según numeral 1.3.1): procesamiento de datos, diseñador de pantallas HMI, representación de señales de alarma y visualización gráfica dinámica representativa del proceso. Debido a que los sistemas de monitoreo y supervisión están conformados por muchos elementos que para su implementación requieren de un extenso tiempo, en este proyecto está fuera del alcance: manejo de históricos, manejo de estadísticas, comunicación por medio de otros protocolos, *driver* de comunicaciones; aunque adicionalmente se incluye un módulo gestor de contenido, que posibilita la actualización, mantenimiento y ampliación de la web con la colaboración de múltiples usuarios.

Existen otros aspectos que se deben tomar en cuenta dentro de los alcances y que se irán especificando dentro del desarrollo del documento.

2.4. INGENIERIA DE REQUERIMIENTOS (40), (41), (42)

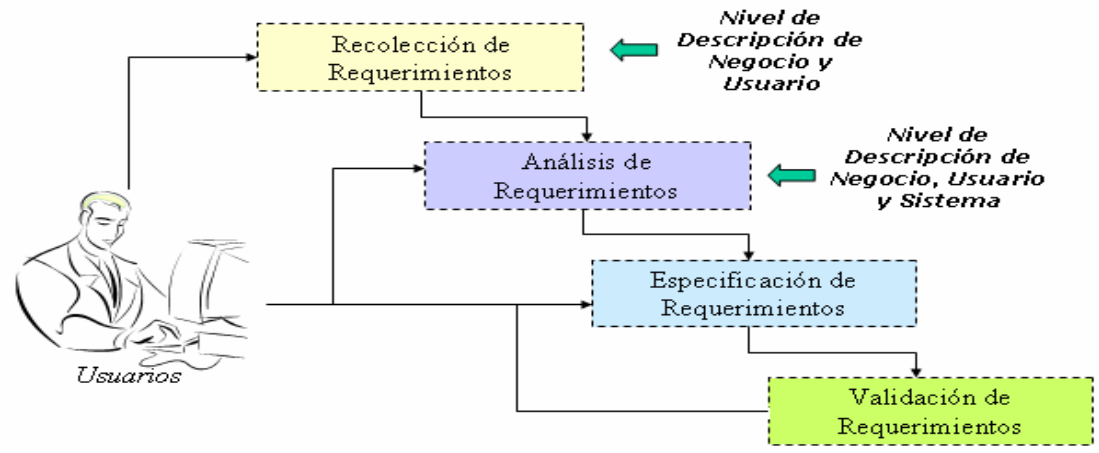
2.4.1. Definición

El marco conceptual sobre el cual se desarrolla este proyecto está constituido principalmente por el proceso de ingeniería de requerimientos. Este se rige bajo un enfoque tradicionalista de la Ingeniería de *Software*, y tiene como objetivo el identificar, analizar, documentar y validar los requerimientos que van a ser desarrollados para un sistema o producto de *Software*. La figura 2-3 presenta las diferentes fases del proceso de ingeniería de requerimientos las cuales se listan a continuación:

- ***La recolección:*** es la fase inicial en la cual se trata de descubrir los requerimientos e identificar los límites del sistema a través de la consulta a los participantes del sistema.
- ***Análisis:*** es el proceso de analizar las necesidades de los clientes y los usuarios para llegar a una definición de los requerimientos de *software*.
- ***Especificación:*** consiste en el desarrollo de un documento que de manera clara y precisa contenga y especifique cada uno de los requerimientos del sistema de *software*.

- ***Verificación:*** es el proceso de asegurar que la especificación de requerimientos de *software* sea acorde con los requerimientos del sistema, conforme a los estándares de documentación de la fase de requerimientos, y que a su vez este documento sea una base sólida para la arquitectura y el diseño.

Figura 2-3: Niveles de descripción de requerimientos utilizados el proceso de ingeniería de requerimientos



Fuente: modificada de (43)

Existen diferentes clasificaciones de los requerimientos, representativas de distintos autores; sin embargo, se hará uso a una de las clasificaciones más aceptadas. Esta clasificación se relaciona directamente con la noción de sistema o solución basada en *software*, por tanto se enfoca a establecer y diferenciar las propiedades de los requerimientos dentro de estos sistemas.

- ***Requerimientos funcionales:*** describen las interacciones entre el sistema y su entorno (usuarios u otros sistemas), sin tener en cuenta cuestiones de implementación. Se estudian y representan en el modelo de casos de uso.
- ***Requerimientos no funcionales:*** describen aspectos del sistema visibles por el usuario que no se relacionan en forma directa con el comportamiento funcional del sistema. Se recogen en los casos de uso con los que están relacionados, o en la especificación complementaria.
- ***Requerimientos de implementación:*** son necesidades del cliente que restringen la implementación (por ejemplo, lenguaje de programación, plataforma hardware, servidor de páginas web).

Para el análisis de requerimientos solo se trabajará con los módulos que demandan el diseño, según los alcances establecidos en el numeral 2.3 estos módulos son: MCOXM, MCODR, MCOXR, MSOX, MM&SW.

2.4.2. Recolección de requerimientos

Esta fase se compone de dos etapas: en la primera etapa se identifican los requerimientos básicos del sistema escritos en un nivel de descripción, y en la segunda etapa se especifican los requerimientos de usuario.

a) Recolección de requerimientos básicos para cada módulo

A continuación se presentan los requerimientos básicos que deberán cumplir cada uno de los módulos que servirán de base para el funcionamiento de la arquitectura para la integración de las plataformas de control basadas en PC.

Módulo cliente OPC XML en Matlab (MCOXM): para establecer los requerimientos de este módulo, se analizaron algunos clientes comerciales OPC existentes en el mercado como *dOPC Explorer* (44), OPC XML *Trial Client* (45), estos requerimientos básicos se listan a continuación:

- 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.

Módulo cliente OPC DCOM en *Rtai-Lab* (MCODR): para establecer los requerimientos básicos de este módulo, se analizó el OPC Toolbox (33) de Simulink/Matlab por sus similitudes que este presenta con Scilab/Scicos, estos requerimientos básicos se listan a continuación:

- 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.
- Presentar tres bloques uno de configuración del servidor, otro de lectura, y otro de escritura de objetos ítems.
- Permitir al usuario configurar los parámetros de cada bloque.

Módulo cliente OPC XML en Rtai-Lab (MCOXR): a continuación se listan los requerimientos básicos de este módulo, los cuales se obtuvieron mediante el análisis del OPC *Toolbox* de Matlab:

- 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 Scicos/Scilab y enviarlos al servidor OPC XML luego de iniciado el cliente.
- Presentar tres bloques uno de configuración del servidor, otro de lectura, y otro de escritura de objetos ítems.
- Permitir al usuario configurar los parámetros de cada bloque.

Módulo servidor OPC XML (MSOX): para definir los requerimientos básicos de este módulo se analizó la documentación ofrecida por la fundación OPC, estos requerimientos se listan a continuación:

- 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.

Módulo de monitoreo y supervisión web (MM&SW): para definir los requerimientos básicos que debe cumplir este módulo, se realizó un análisis de algunos de los sistemas de monitoreo y supervisión descritos en el numeral 1.3.3 de este documento. En este proyecto se implementa un sistema de monitoreo y supervisión soportado en tecnología web con los siguientes requerimientos básicos:

- 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. Estas señales deben ser visuales.
- 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.

b) <u>Descomposición de los objetivos del sistema en requerimientos de usuario</u>

En esta etapa se descomponen los requerimientos básicos previamente establecidos en requerimientos de usuario, se utilizan los caso de uso los cuales son una secuencia de interacciones entre el sistema y uno o más actores, en la que se considera al sistema como una caja negra y en la que los actores obtienen resultados observables (46), (47), (48).

En el caso de estudio, se diseñaron cinco (5) casos de uso básicos para el funcionamiento del sistema:

- ***Caso 1:*** Módulo cliente OPC XML en Matlab (MCOXM)
- ***Caso 2:*** Módulo cliente OPC DCOM en Rtai-Lab (MCODR)
- ***Caso 3*****:** Módulo cliente OPC XML en Rtai-Lab (MCOXR)
- ***Caso 4:*** Módulo servidor OPC XML (MSOX)
- ***Caso 5*****:** Módulo de monitoreo y supervisión web (MM&SW)

Las plantillas y los diagramas de casos de uso se presentan en el anexo A (plantillas y diagramas para casos de uso y requerimientos básicos de la Arquitectura).

2.4.3. Análisis de requerimientos

Con base en los casos de uso se establecieron los requerimientos básicos para cada uno de los módulos, los cuales se especifican en la sección II del anexo A (plantillas para casos de uso y requerimientos básicos).

2.4.4. Especificación de requerimientos finales

En esta etapa se definen y clasifican los requerimientos finales que tendrá el sistema, de acuerdo a los requerimientos básicos registrados en la sección II del anexo A (plantillas para casos de uso y requerimientos básicos)

a) <u>Requerimientos funcionales</u>

Las tablas 2-1, 2-2, 2-3, 2-4 y 2-5, registran una lista de requerimientos funcionales de la arquitectura de integración, los cuales, se obtuvieron de un proceso de filtrado y depuración de los requerimientos básicos expuestos en la sección II del anexo A (Plantillas para casos de uso y requerimientos básicos), y del análisis de información en documentos sobre OPC. En la primera columna se especifica los sub-módulos, en la segunda columna se presenta un identificador de cada requerimiento, y en la tercera columna se realiza la descripción de los requerimientos.

Tabla 2 -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 menús: 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 2 -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 2 -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 2 -4: Requerimientos funcionales para el módulo servidor OPC XML

Sub Módulo	ID	Requerimientos Funcionales
	RFMSOX1	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 2 -5: Requerimientos funcionales para el módulo de monitoreo y supervisión web (MM&SW)

Sub-Mód	ID	Requerimientos Funcionales
	RFMM&SW1	El módulo de monitoreo y supervisión debe contar con tres sub-módulos básicos: gestor de contenidos, diseño de proyectos y 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 HMI.
Diseño de proyectos	**RFMM&SW8**	Crear: se debe permitir crear un nuevo proyecto
	RFMM&SW9	Eliminar: se debe permitir eliminar el proyecto seleccionado.
	RFMM&SW10	Editar: Se debe permitir editar el proyecto seleccionado.
	RFMM&SW11	Visualizar: se debe permitir visualizar el proyecto seleccionado.
	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, ícono 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	**RFMM&SW20**	Ejecutar las acciones de mando pre-programadas a partir de los valores actuales de variables leídas.
	RFMM&SW21	Visualizar, almacenar y controlar alarmas
	RFMM&SW22	Visualizar eventos
	RFMM&SW23	Graficar variables en función del tiempo

b) Requerimientos no funcionales:

En esta sección se especifica los requerimientos no funcionales. Estos describen los atributos que debe tener el *software* una vez construido (portabilidad, eficiencia, confiabilidad, robustez, rendimiento, etc).

En la tabla 2-6 se relaciona la lista de requerimientos no funcionales generales que deben cumplir todos los módulos de la arquitectura de integración. En la primera columna se indica el código asociado a cada requerimiento, en la columna 2 se especifica el tipo de requerimiento y en la columna tercera se describe el requerimiento.

Tabla 2 -6: Requerimientos no funcionales generales

NUMERO	TIPO	REQUERIMIENTO
RNFG1	Tiempo de respuesta.	Cada uno de los módulos deben utilizar herramientas que le permitan un tiempo de respuesta menor a un segundo.
RNFG2	Disponibilidad	Cada uno de los módulos deben estar disponible 100% o muy cercano a esta disponibilidad durante el horario hábil establecido para la comunicación OPC.
RNFG3	Escalabilidad	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.
RNFG4	Facilidad de uso e Ingreso de información	Cada uno de los módulos debe: tener una interfaz atractiva y amigable, ppresentar mensajes de error que permitan al usuario identificar el tipo de error.
RNFG5	Flexibilidad	El sistema debe ser diseñado y construido con los mayores niveles de flexibilidad en cuanto a la parametrización de los tipos de datos, de tal manera que la administración del sistema sea realizada por un administrador funcional del sistema.
RNFG6	Mantenibilidad	Cada uno de los módulos de la arquitectura deberá estar documentado, con los manuales de administración y de usuario.
RNFG7	Operatividad	Los módulos deben ser de fácil operación por el área técnica.
RNFG8	Validación de Información	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

Cada módulo debe cumplir requerimientos propios no funcionales los cuales se especifican en la tabla 2-7, en la primera columna se indica el módulo, la segunda columna representa el código asociado a cada requerimiento, y en la tercera columna se describe cada requerimiento no funcional.

Tabla 2 -7: Requerimientos no funcionales específicos en cada módulo

MÓD	CÓDIGO	REQUERIMIENTO
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.
	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&SW2	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.

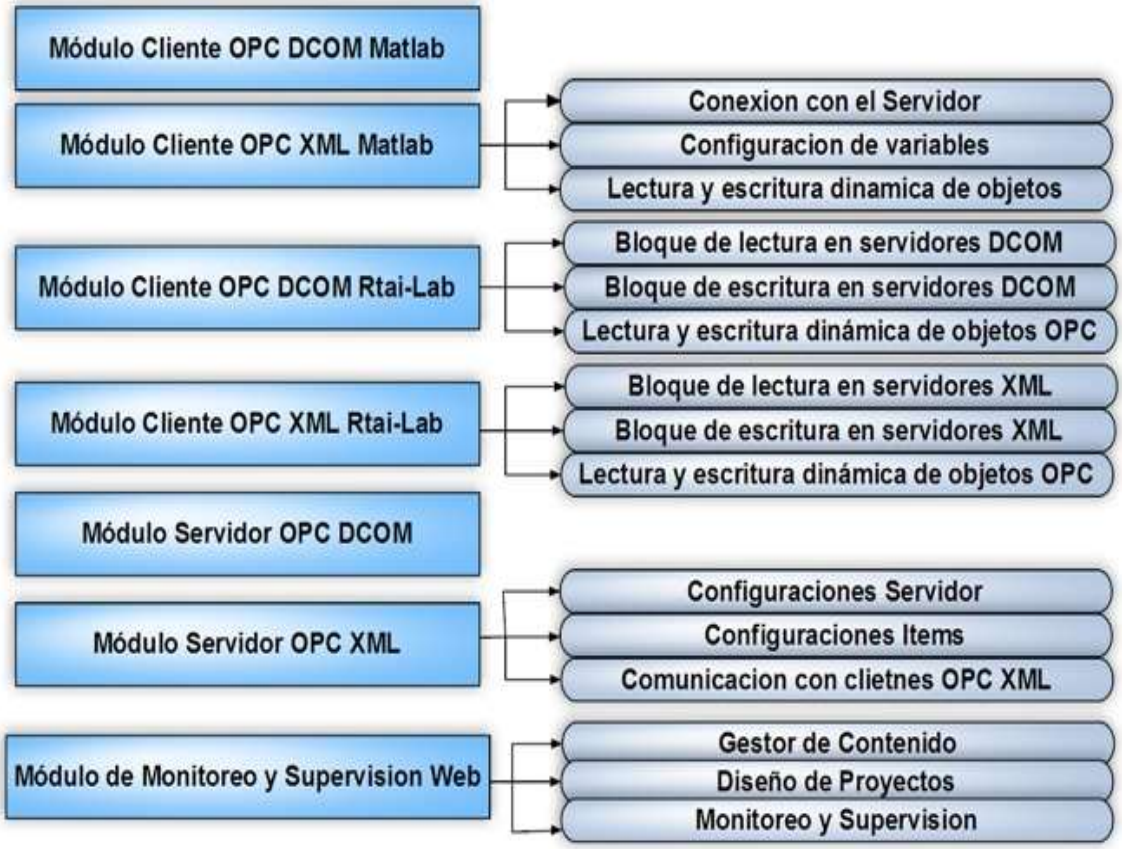
Fuente: propia

2.5. RESUMEN

La arquitectura para la integración de plataformas de control basadas en PC mediante OPC según las especificaciones del numeral 2.3 se compone de los siguientes módulos: cliente OPC DCOM Matlab (MCODM), cliente OPC XML en Matlab (MCOXM), cliente OPC DCOM Rtai-Lab (MCODR), cliente OPC XML Rtai-Lab (MCOXR), servidor OPC DCOM (MSOD), servidor OPC XML (MSOX), módulo

de monitoreo y supervisión web (MM&SW), cada uno de estos módulos se subdivide en sub-módulos, previamente explicados, ver figura 2-4.

Figura 2-4: Especificación de módulos y sub-módulos de la arquitectura propuesta



Fuente: propia

En este capítulo se definen los objetivos y alcances, que tendrá la arquitectura para la integración de plataformas de control basadas en PC, además se especifican los módulos y sub-módulos que hacen parte de esta arquitectura, junto con los requerimientos funcionales y no funcionales como base para la construcción e implementación de dichos módulos, los cuales se exponen en los siguientes capítulos.

3. CONSTRUCCION DE LOS MÓDULOS SERVIDORES Y CLIENTES OPC

Basado en los requerimientos recopilados en el capítulo dos, en esta sección se describe la construcción de los módulos servidores OPC DCOM/XML, y los clientes OPC para Matlab y Rtai-Lab, en cada módulo construido se realiza una descripción general, se detallan las herramientas *software* de programación, las restricciones generales, el diagrama de archivos, el funcionamiento interno y finalmente se describe el uso de cada uno de los módulos.

3.1. MÓDULO SERVIDOR OPC DCOM (MSOD)

En el ítem 2.3.5 se expuso que para el módulo servidor OPC DCOM, se haría uso de cualquiera de los servidores OPC existentes en el mercado, de los cuales algunos se describen a continuación:

- ***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. El servidor *KEPServerEx* está compuesto de dos partes, la primera proporciona toda la conectividad OPC y *Dynamic Data Exchange* (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 (34).

- ***Matrikon OPC*:** para los integradores, desarrolladores y otros usuarios de OPC, este servidor es herramienta gratuita para ayudar a probar y solucionar problemas de aplicaciones OPC (clientes) y las conexiones. El servidor de simulación MatrikonOPC soporta de forma nativa la especificación de seguridad de la fundación OPC. Esto es crucial para asegurar la aplicación de arquitecturas de OPC (38).

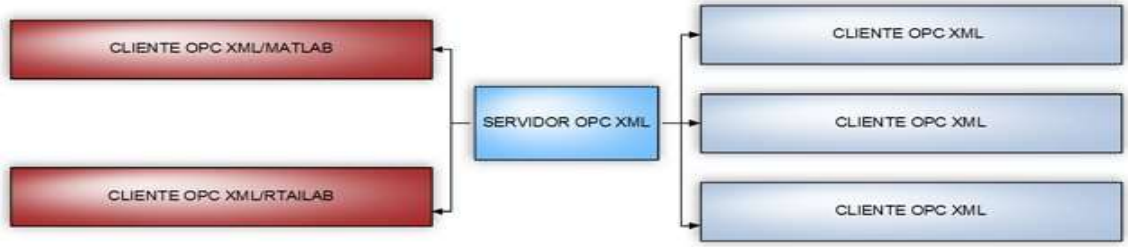
Dentro de los servidores OPC de codigo abierto se encuentra LIGHTOPC, el cual fue desarrollado por el equipo Lab34 a finales del año 2000 como alternativa a los desarrolladores de servidores OPC comerciales, de manera que su acceso fuera libre y gratuito. *LightOPC* permite el desarrollo sobre plataformas Win32 de servidores OPC v.1.0 y v.2.0. La actual versión nos ofrece, entre otras cosas: compatibilidad DCOM total, Soporta desarrollos como aplicaciones dentro del proceso (.dll) y fuera del proceso (.exe),posibilidad de interfaz en C (además de C++), Servicio *logging*, Permite el tipo *array* para los ítems, la arquitectura interna esta optimizada para el uso de la cache de la CPU, *LightOPC* ofrece, a través de las librerías lightOPC.dll y unilog.dll (para el servicio *logging*), una serie funciones que nos hacen transparente el manejo de objetos servidor, grupos, e Ítems,

proporcionando la adición de ítems, lectura y actualización de valores mediante funciones sencillas (39).
Aunque para el módulo MSOD se puede hacer uso de cualquiera de los servidores OPC DCOM descritos anteriormente u otros, para la integracion de Matlab y Rtai-lab se utiliza *KEPServerEx* por dos razones: la primera, por el conocimiento que se tiene de este servidor en el laboratorio de control de procesos y la segunda por las especificaciones tecnicas que brinda este servidor: facilidad de aprendizaje, se puede agregar múltiples controladores (PLC, controladores de base de datos y de aplicaciones específicas) sin tener que preocuparse por el aprendizaje de nuevos protocolos de comunicación.
En el anexo B (manual de usuario del módulo servidor OPC DCOM KEPServerEx) se detalla una guia de instalacion y configuracion del servidor OPC DCOM *KEPServerEx,* para la comunicación con el PLC *Micrologix* 1500 y para las plataformas Matlab/Rtai-Lab.

3.2. MÓDULO SERVIDOR OPC XML (MSOX)

Un servidor OPC permite recuperar datos de dispositivos o redes subyacentes, tales como buses de campo. En tal situación, el servidor OPC XML-DA es similar a un *proxy*, que recupera los datos en un lado de buses de campo o dispositivos y envía los datos a los clientes del otro lado, tal como se muestra en la figura 3-1.

Figura 3-1: Esquema de comunicación del servidor OPC XML



Fuente: Modificado de (49)

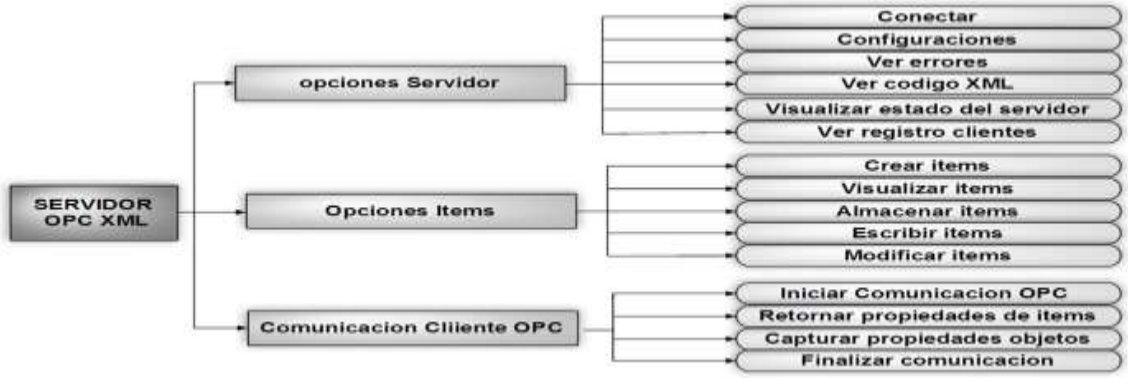
En el caso de estudio, la función principal del módulo MSOX según el numeral 2.3, es el de traducir los datos provenientes de las fuentes datos (Matlab y Rtai-Lab) para que sea compatible con la especificación OPC XML DA.

Los servidores OPC XML no son de fácil disponibilidad, por lo tanto para la recolección de datos desde Matlab y Rtai-Lab mediante el estándar OPC XML, se diseña un servidor que permita capturar dichos datos y convertirlos en el formato estándar OPC XML DA. Inicialmente se realiza una descripción de los componentes del módulo, las restricciones generales, las herramientas usadas para la construcción, un diagrama de archivos, el funcionamiento interno y finalmente se describe el uso del módulo.

3.2.1. Componentes del módulo MSOX

Basado en el requerimiento RFMSOX1 establecido en la tabla 2-4, se detallan los componentes del módulo servidor OPC XML, los cuales se representan en la figura 3-2.

Figura 3-2: Diagrama componentes básicos del servidor OPC XML



Fuente: propia

Los componentes presentados en la figura 3-2 se describen a continuación:

Opciones del servidor: el usuario debe poder configurar todos los parámetros correspondientes al servidor OPC XML: conexión (**RFMSOX2)**, configuraciones **(RFMSOX3)**, visualizar errores (**RFMSOX4)**, ver código XML (**RFMSOX5)**, ver registro de clientes **(RFMSOX6) y** visualizar estado del servidor **(RFMSOX7)**.

Opciones de ítems: el usuario debe poder crear **(RFMSOX8)**, visualizar **(RFMSOX9)** y escribir cada uno de los objetos ítems del servidor **(RFMSOX10)**. Dentro de las características que se pueden establecer en cada objeto ítem se encuentran: el nombre, tipo de datos y el valor. El nombre corresponde a la identificación general de cada ítem, los tipos de datos deben ser consistentes con los proporcionados por OPC Data Access, y el valor lo determina el usuario. Todos los objetos ítems deben poder almacenarse en un archivo para su posterior uso **(RFMSOX11 y RFMSOX12)**.

Comunicaciones con clientes OPC: el servidor debe aceptar solicitud de conexión con todos los clientes OPC XML DA (**RFMSOX13- RFMSOX16**), retornar a los clientes las propiedades de los objetos ítems (**RFMSOX14**), capturar propiedades de objetos ítems **(RFMSOX15)** y finalizar la comunicación con los clientes (**RFMSOX17**).

3.2.2. Restricciones generales

- El computador donde se instala el servidor OPC XML debe estar conectado a una red LAN, o tener acceso a internet.
- En el equipo donde se instale el servidor OPC XML, debe tener activado el puerto por el que se realizará la comunicación. Ver anexo H (guía para activar los puertos y configuración del componente de *Windows* DCOM).

Limitaciones de HW:

- ***Procesador:*** cualquier procesador Intel/AMD x86, o superior.
- ***RAM:*** 1G o superior.
- ***Sistema operativo:*** Windows XP o superior – Linux (cualquier versión).
- ***Tamaño en disco duro:*** 220 Megabytes.

3.2.3. Herramientas utilizadas en el módulo MSOX

Para la construcción del servidor OPC XML se utiliza un paquete de librerías denominadas *PyOPC*, el cual logra satisfacer los requerimientos funcionales expuestos en la tabla 2-4, y los no funcionales: código abierto (**RNFMSOX1)**, multiplataforma (**RNFMSOX3**), facilidad de uso, extensible y reutilizable **(RNFG3)**, expuestos en las tablas 2-6 y 2-7. Cabe aclarar que PYOPC fue la única herramienta que se encontró en internet, que permite el diseño de servidores OPC en el estandar XML, y que satisface los requerimientos establecidos anteriormente. El servidor OPC XML requiere para su implementación, las siguientes herramientas *Software*:

- ***Python:*** lenguaje interpretado, orientado a objetos, que permite mantener interacción con el sistema operativo de forma sencilla. Está disponible en MS–Windows, GNU/Linux y Mac (50).
- ***PyOPC:*** es un conjunto de librerias para el diseño de clientes y servidores OPC diseñadas en Python, con las siguientes caracteristicas: código abierto, multiplataforma, facilidad de uso, extensibles y reutilizables. El Módulo XDAServer.py de este paquete permite el diseño del servidor OPC XML DA (49).

Aunque Python tiene una extensa colección de bibliotecas, no cumple con los requisitos para la construcción de servidores que procesan solicitudes simultáneas y manejan el protocolo SOAP (*Simple Object Access Protocol)* y HTTP (*Hypertext Transfer Protocol*) (51), por lo tanto se incluyen dos nuevas librerías necesarias para que el servidor OPC XML funcione, estas tecnologías básicas se listan a continuación:

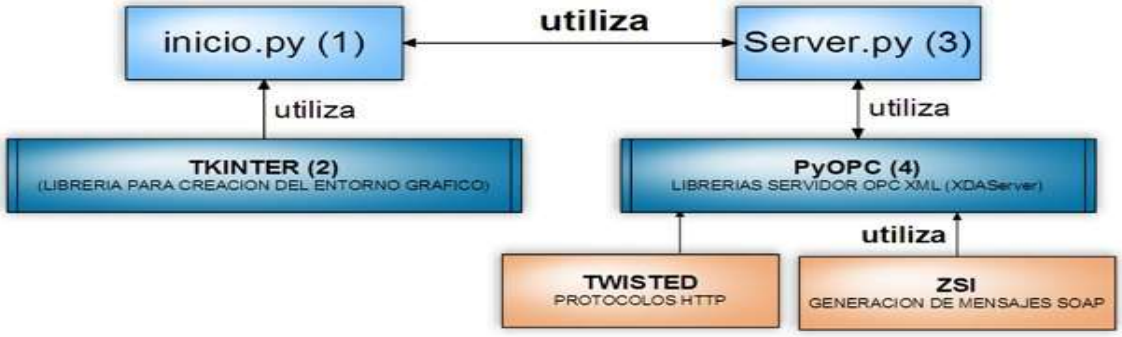
- ***Twisted:*** infraestructura cliente/servidor escrito en lenguaje Python, implementa una variedad de protocolos de Internet, como HTTP y SMTP (*Simple Mail Transfer Protocol*), y utiliza un mecanismo basado en eventos que permiten el desarrollo de clientes y servidores que pueden manejar peticiones concurrentes (52).
- ***Zolera Soap Infrastructure (ZSI)*** (53)***:*** se utiliza para el análisis y la generación de los mensajes SOAP.

Por otra parte, PyOPC simplemente proporciona las librerías necesarias para la comunicación OPC XML, en el caso de estudio, se diseñó una interfaz HMI que permita cumplir con los requerimientos **RNFG4**, **RNFMSOX1**, **RNFMSOX3** de las tablas 2-6 y 2-7. Para la interfaz del servidor OPC XML, se evaluaron y eligieron las herramientas con las que se pueda desarrollar aplicaciones con interfaces gráficas de usuario en Python, dentro de las herramientas disponibles, la decisión se redujo a dos posibilidades Tkinter (54) y wxPython (55). Sobre la facilidad de instalación, en Linux no hay problema con ninguno de los dos; Tkinter viene con la distribución de cPython, para Mac y Windows tampoco dan demasiados problemas (56). En cuanto a facilidad de uso y aprendizaje; Tkinter tiene más cantidad de material disponible, y cuenta con comunidades activas dispuestas a ayudar que wxPython, por lo tanto, se ha utilizado Tkinter ya que adicionalmente su aprendizaje es más fácil.

3.2.4. Diagrama de archivos del servidor OPC XML

Para la implementación del servidor OPC XML se crearon dos archivos: uno denominado “inicio.py”, ver figura 3-3, 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. En el anexo I sección II (esquema de archivos y código fuente de los módulos implementados) se presenta el código fuente asociado al MSOX.

Figura 3-3: Diagrama de archivos del servidor OPC XML

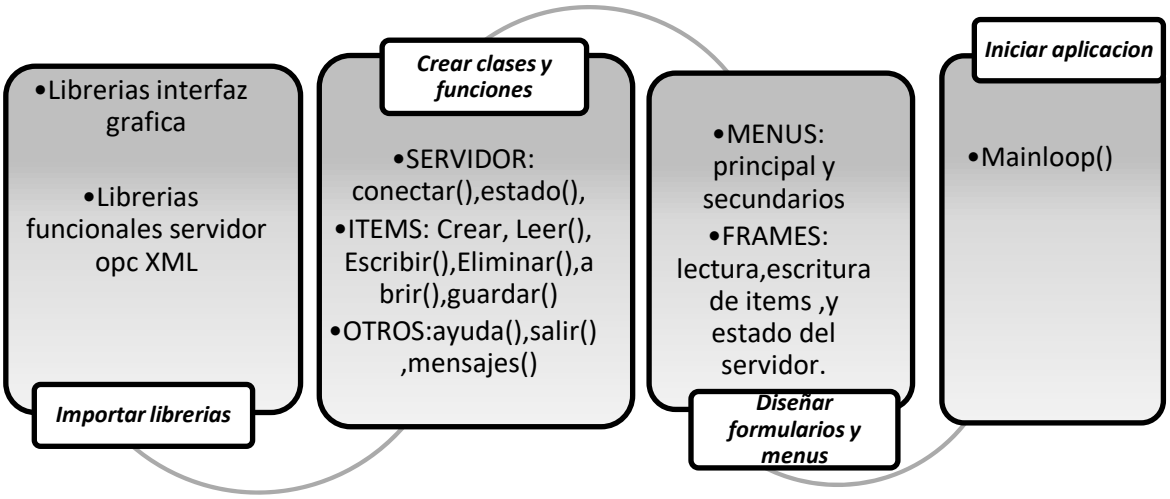


Fuente: propia

3.2.5. Funcionamiento interno del módulo MSOX

La figura 3-4 indica los pasos que se desarrollaron para la construcción del servidor OPC XML, el primer paso consiste en importar las librerías que permiten el diseño gráfico y de comunicación OPC (Tkinter y Server), luego se definen las clases y las funciones que se encargan de tomar/procesar los datos provenientes de los menús/formularios, se diseñan los menús/formularios y se asocian a una función del archivo "Server.py", por último se crea un *main-loop* que es el hilo principal que se encarga de ir llamando a cada componente en el orden adecuado, hasta que el servidor finalice.

Figura 3-4: Pasos para la construcción del servidor OPC XML



Fuente: propia

A continuación se detalla el funcionamiento del archivo Server.py el cual contiene toda la lógica del servidor OPC XML. Este módulo contiene una clase llamada MyXDAServer, la cual hereda funciones de la clase BasicXDAServer que viene incluida en el módulo XDAServer del paquete *software* PyOPC. Para entender el funcionamiento del módulo MSOX se explican los componentes mencionados en el ítem 3.2.1.

Configuraciones del servidor: dentro de las funciones que debe cumplir el servidor OPC XML-DA, es el de permitir que el usuario establezca el puerto de comunicación, visualice los errores que se presentan, entre otras configuraciones las cuales se detallan a continuación:

- ***Conexión:*** **(RFMSOX2)** inicialmente se define la función "conectar", la cual recibe el puerto por donde se va a establecer la comunicación OPC, luego llama a la clase implementada "*MyXDAServer*" que contiene las funciones del

servidor OPC XML, se establece el sitio, el puerto del servidor y finalmente mediante “reactor.run()” se inicia el servidor.

- ***Configuraciones:*** **(RFMSOX3)** a continuación se listan las propiedades de configuración del servidor OPC XML que ofrece la librería BasicXDAServer, de acuerdo al siguiente formato:

  Nombre propiedad (valor predeterminado): descripción

  - ✓ *AutoItemCache (True):* esta opción permite el almacenamiento automático en caché de los ítems OPC.
  - ✓ *WritePurgeCache (True):* indica si el objeto ítem de caché debe eliminarse después de que un objeto ítem es escrito.
  - ✓ *BufferSize (100):* indica el tamaño máximo de objetos ítems en el búfer.
  - ✓ *ThreadedParsing (True):* una de las tareas más intensivas de la CPU es el análisis de los mensajes SOAP. Con el fin de acelerar el funcionamiento del servidor, es posible ejecutar el programa de análisis en un hilo separado, para que el servidor puede ejecutar otras solicitudes en paralelo.
  - ✓ *ThreadPoolSize (5):* la cantidad máxima de subprocesos simultáneos.
  - ✓ *SubscriptionPingRate (10000):* tasa de ping en milisegundos.
  - ✓ *MaxPingRate (86400000 = 1 día):* La tasa de ping máximo que podrá ser indicada por un cliente.
  - ✓ *MaxSamplingRage (100):* la tasa máxima de muestreo en milisegundos que los clientes pueden capturar.
- ***Visualizar errores:*** **(RFMSOX4)**, el código error_log_fn='error.log' permite almacenar en el archivo error todos los errores registrados en el servidor OPC XML.

- ***Ver registro de clientes:*** **(RFMSOX5)**, el código access_log_fn='access.log' permite visualizar en el archivo Access la fecha, la dirección del host, y la acción que ejecuta el cliente OPC XML.

- ***Ver código XML:*** **(RFMSOX6)**, el código http_log_fn = 'http.log' permite almacenar en el archivo http el código XML que se genera en el servidor cuando un cliente solicita la comunicación.

Configuración ítems:

- ***Crear Ítems:*** **(RFMSOX8)**, XDAServer permite implementar un objeto *ItemContainer,* que contiene la lista de objetos ítems. Dicha lista se crea en

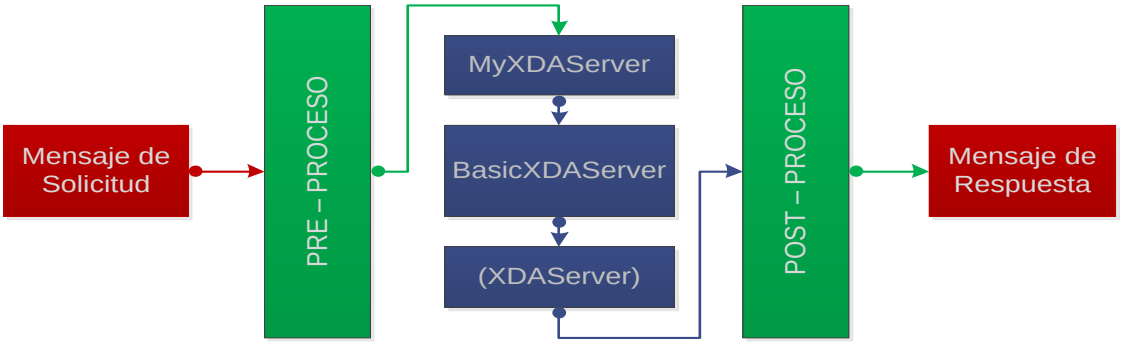
el momento que el usuario mediante un formulario, establece las propiedades de cada uno de los objetos.

- ***Visualizar ítems:*** **(RFMSOX9)**, para que el usuario pueda visualizar los objetos ítems configurados en el servidor, se hace uso de la propiedad "*read*" que ofrece el módulo XDAServer.

- ***Escribir ítems:*** **(RFMSOX10),** para cambiar el valor a cada ítem, se hace uso de la propiedad "*Write*" que ofrece el módulo *XDAServer*.

Comunicación con los clientes OPC: la figura 3-5 presenta un diagrama de comunicación entre un cliente y el servidor OPC XML. XDAServer analiza automáticamente los mensajes entrantes de los clientes a través de la librería SOAP y crea los objetos OPC apropiados que representan el mensaje entrante, después de eso inicia la etapa de pre procesamiento la cual se ocupa de retornar el mensaje de salida.

Figura 3-5: Proceso de Comunicación del MSOX y los clientes OPC XML



Fuente: Modificado de **(49)**

Después que el servidor se ha inicializado, los métodos de funcionamiento del servidor se pueden ejecutar, y los clientes pueden iniciar las siguientes actividades:

Solicitudes de escritura: **(RFMSOX14)**, el cliente puede especificar la opción "*ReturnValuesOnReply*". Si esta opción está establecida en true, el servidor vuelve a leer los valores indicados.

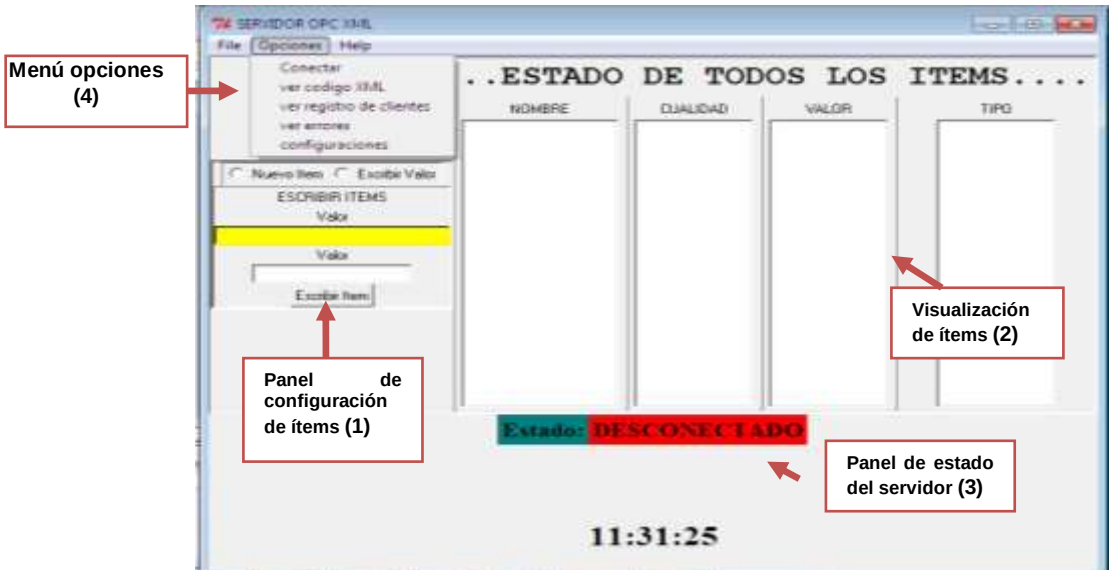
Solicitud de lectura: **(RFMSOX15)**, el cliente puede saber el valor, cualidad, tiempo de respuesta, tipo de datos de cada objeto ítem.

Búsqueda: **(RFMSOX16)**, los clientes pueden buscar los objetos OPC.

3.2.6. Usando el servidor OPC-XML

Cuando el usuario ejecuta el archivo “inicio.py”, aparece la pantalla principal (ver imagen 3-1), donde se presentan tres paneles, el primero es de configuración (recuadro 1), donde el usuario puede crear, o editar las propiedades de cada ítem, el segundo es el de visualización (recuadro 2), donde se presenta cada una de los objetos ítems con sus respectivas propiedades, y el tercero indica el estado del servidor (recuadro 3), además se presenta un menú: *File*, opciones y *Help*, en el menú “File” se puede guardar, abrir los archivos correspondientes a los ítems, en el menú “opciones” se establecen las propiedades del servidor (ver numeral 4), y “help” indica un manual de usuario para el manejo del servidor.

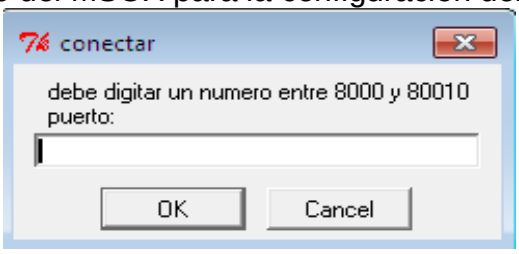
Imagen 3-1: Pantalla principal del servidor OPC XML



***Fuente**: propia*

Para inicializar el servidor OPC XML se selecciona el menú “opciones/conectar” del formulario principal, posteriormente se define el puerto por el cual se establecerá la comunicación OPC, ver Imagen 3-2.

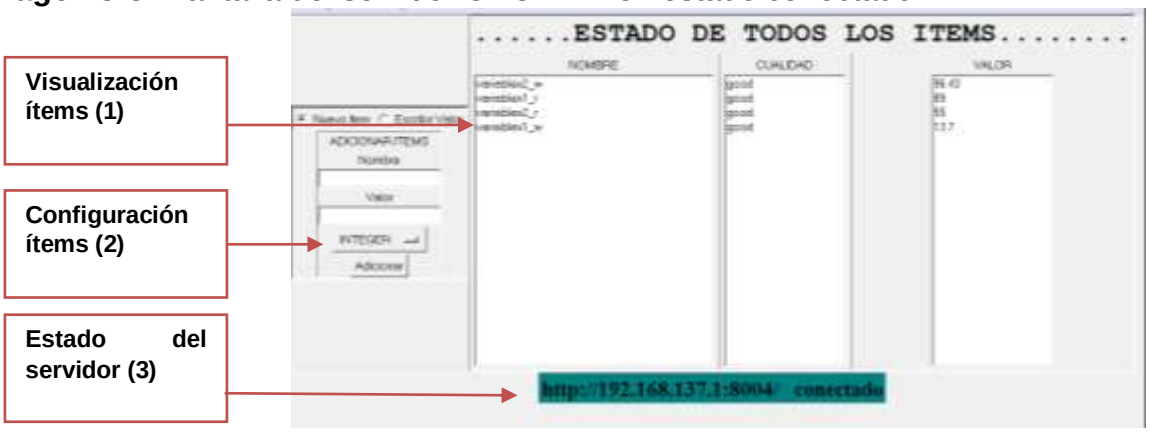
Imagen 3-2: Formulario del MSOX para la configuración del puerto



Fuente: propia

La imagen 3-3 indica la pantalla del servidor OPC XML cuando el servidor se ha inicializado correctamente. En el recuadro tres, se presenta la dirección completa del servidor OPC XML (http://192.168.137.1:8004), mediante la cual los clientes pueden comunicarse, esto indica que el usuario puede empezar a adicionar, modificar (recuadro 2), y visualizar los objetos ítems del servidor OPC (recuadro 1), y los clientes OPC XML pueden iniciar la solicitud de conexión.

Imagen 3-3: Pantalla del servidor OPC XML en estado conectado



Fuente: propia

3.2.7. Instalación y manual de usuario MSOX

En el anexo H (guía para activar los puertos y configuración del componente de Windows DCOM) se presenta una guía de activación de los puertos correspondientes, y en el anexo C (guía de instalación y manual de usuario del módulo servidor OPC XML) se presenta un manual de usuario que permite la instalación y el uso del servidor OPC XML.

3.3. CLIENTE OPC DCOM EN MATLAB (MCODM) (57).

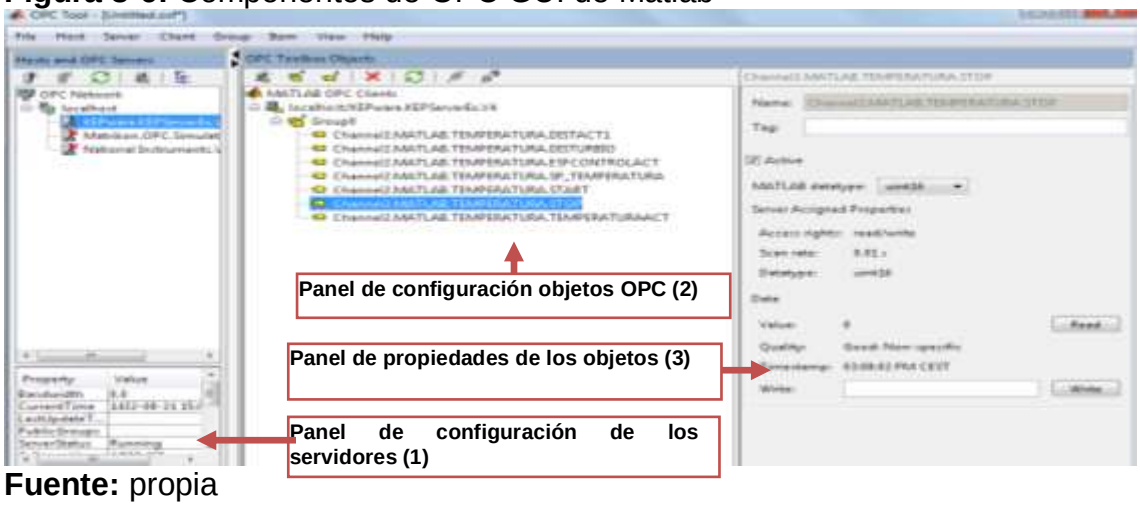
Según lo descrito en el numeral 2.2, este módulo debe permitir el intercambio de información entre programas diseñados en Matlab y un servidor OPC DCOM, pero como se mencionó en el numeral 2.31, Matlab cuenta con tres formas de comunicarse con servidores OPC (33): mediante funciones de una librería OPC, interfaz gráfica GUI, *OPC-Toolbox*, por lo tanto para integrar Matlab y Rtai-Lab mediante el estándar DCOM se puede hacer uso de cualquiera de estos métodos. A continuación se detalla brevemente, cada una de las formas de comunicación de Matlab con los servidores OPC DCOM.

3.3.1. Comunicación OPC DCOM mediante Interfaz gráfica de usuario (GUI)

MATLAB cuenta con una interfaz que ayuda a administrar las conexiones con los servidores OPC y los objetos ítems del servidor. Herramienta permite la lectura y escritura de datos OPC y registra los valores en el *Workspace*.

La figura 3-6 ilustra la pantalla que se presenta al digitar "OPCtool" en la ventana de comandos de Matlab, dicha pantalla está compuesta por tres paneles, el primero permite establecer la comunicación con los host y servidores remotos, el segundo permite importar los objetos de los servidores OPC DCOM y el tercero permite leer y escribir en los objetos ítems del servidor.
Los pasos que se deben seguir para establecer una comunicación con los clientes OPC son los siguientes: adicionar la dirección IP del host y el nombre del servidor, ver gráfica 3-6 recuadro 1, adicionar los objetos del servidor (recuadro 2), escribir y leer en los objetos OPC (recuadro 3).

Figura 3-6: Componentes de OPC GUI de Matlab



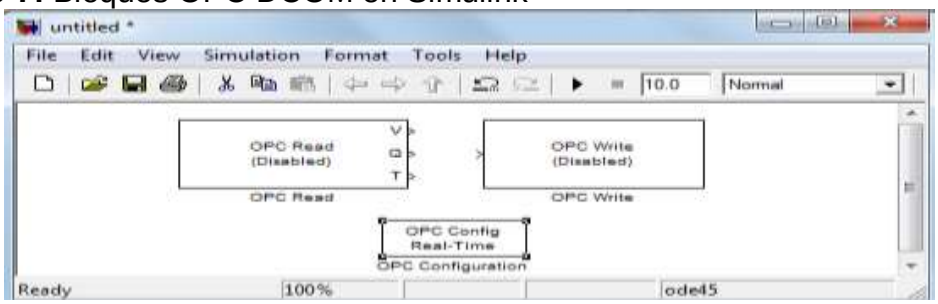
Fuente: propia

3.3.2. Comunicación Matlab/ OPC DCOM mediante *OPC-Toolbox*

Matlab tiene una herramienta denominada *OPC-Toolbox* para adquirir información desde y hacia el servidor OPC DCOM mediante *Simulink*. Esta herramienta consta de tres bloques: uno para la configuración de los clientes OPC en el modelo, en el cual se puede definir el comportamiento de los errores/eventos OPC, y los dos bloques restantes (OPC Write, OPC Read), que permiten crear un grupo de objetos que contienen las variables, y permiten la lectura/escritura de variables en forma individual o grupal.
Para usar los bloques en el modelo de Simulink, se debe seleccionar la librería *OPC Toolbox,* la cual contiene cuatro bloques para la comunicación OPC, *(OPC Configuration,OPC read, OPC write*, OPC Quality parts) dar clic en cada uno de

los bloques de la paleta como lo ilustra la figura 3-7, establecer las propiedades dando clic derecho sobre cada bloque e interconectarlos con otros bloques funcionales de Matlab.

Figura 3-7: Bloques OPC DCOM en Simulink



Fuente: (33)

3.3.3. Comunicación Matlab/OPC DCOM mediante funciones

Los usuarios pueden ejecutar todas las funciones OPC directamente desde la línea de comandos de MATLAB o incorporarlos en sus propias aplicaciones de Matlab. Por ejemplo para la creación de un cliente OPC se utiliza la función “opcda” a la cual se le envían la dirección IP del equipo y el nombre del servidor.

Para más detalles sobre los módulos cliente OPC DCOM dirigirse a (33), donde se presenta una guía completa de los módulos cliente OPC DCOM en Matlab.

3.4. MÓDULO CLIENTE OPC XML EN MATLAB (MCOXM)

En este numeral se describe la construcción del módulo cliente OPC XML en Matlab, el cual, según el numeral 2.3.2, debe permitir el intercambio de información entre programas diseñados en Matlab/Simulink y servidores OPC XML. Para el desarrollo de este módulo, inicialmente se realiza el diagrama general de del MCOXM, se detallan los componentes básicos, se exponen las restricciones generales, se describen las herramientas *software* utilizadas para la construcción, se realiza un diagrama de archivos, se detalla el funcionamiento interno de cada componente, y finalmente se describe el uso de este módulo.

La figura 3-8 presenta el diagrama general del módulo MCOXM, el cual tendrá, en funcionamiento 4 bloques actuando simultáneamente: Matlab/Simulink, el cliente OPC XML, el servidor OPC XML y otros clientes OPC XML. La información proveniente del entorno Matlab/Simulink (ver figura 3-8, recuadro 1) se almacena en el espacio de trabajo de Matlab (Workspace), desde donde el módulo cliente XML (recuadro 2), captura los datos para enviarlos al servidor XML –DA (recuadro 3), a su vez otros clientes (recuadro 4) acceden al servidor y realizan la captura de datos, esto ocurre hasta que se cierre la comunicación con el cliente OPC XML.

Figura 3-8: Diagrama general del módulo cliente OPC XML en Matlab

Fuente: propia

3.4.1. Componentes del módulo MCOXM

Basados en el requerimiento **RFMCOXM1** establecido en la tabla 2-1, se detallan los componentes del módulo cliente OPC XML en Matlab, los cuales se representan en la figura 3-9, y se exponen a continuación:

Conexión con el servidor: permite al usuario ingresar la dirección del servidor OPC XML **(RFMCOXM2)**, y con base a la información recolectada establece la comunicación OPC XML (**RFMCOXM3).**

Configuración de variables: permite al usuario seleccionar los ítems de lectura y escritura con los que se va a trabajar **(RFMCOXM4)**, y los almacena en el Workspace **(RFMCOXM5).**

Lectura y escritura dinámica de objetos: permite consultar periódicamente al servidor OPC XML las propiedades de los objetos ítems **(RFMCOXM6)**, además permite cambiar el valor de los ítems en el servidor OPC XML con parámetros establecidos por el usuario o por el entorno Matlab **(RFMCOXM7),** y finaliza la comunicación con el servidor OPC XML cuando el usuario lo requiera **(RFMCOXM8-RFMCOXM9).**

Figura 3-9: Diagrama componentes básicos del MCOXM

Fuente: propia

3.4.2. Restricciones generales del MCOXM

- El equipo debe tener instalado Matlab (cualquier versión).
- El servidor OPC XML debe estar inicializado.
- El computador donde se instala el módulo cliente OPC XML debe estar conectado a una red LAN, o tener acceso a internet.

Limitaciones de HW

- ***Procesador:*** cualquier procesador Intel/AMD x86, o superior.
- ***RAM:*** 1G o superior
- ***Sistema operativo:*** Windows XP x64 Edition Service Pack 2 o superior

3.4.3. Herramientas software utilizadas en el MCOXM

Inicialmente se intentó implementar el módulo cliente OPC XML en Matlab haciendo uso de algunas librerías XML existentes para esta plataforma, desafortunadamente no se logró la comunicación OPC y por lo tanto se optó por utilizar el módulo XDAClient perteneciente al paquete PyOPC como una librería adicional de Matlab. PyOPC permite cumplir con los requerimientos **RFMCOXM3, RFMCOXM4, RFMCOXM6, RFMCOXM7, RFMCOXM9, RNFG7, RNFMCOXM1, RNFMCOXM2, RNFMCOXM3** establecidos en las tablas 2-1, 2-6 y 2-7.

Para la implementación de este módulo se requiere, además de Matlab, las siguientes herramientas *software*:

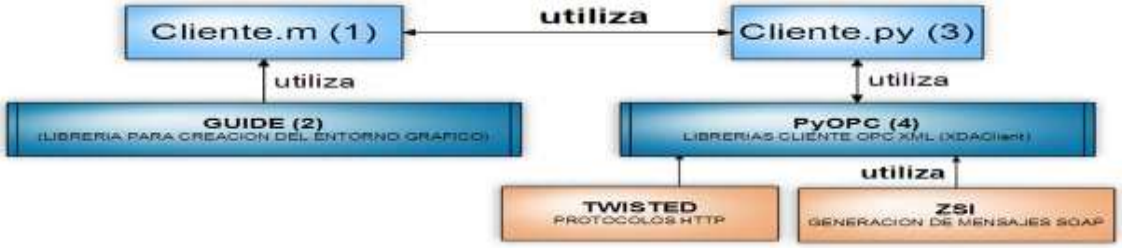
- ***PyOPC*** (49)***:*** ofrece la clase XDAClient apartir de la cual se puede implementar clientes OPC XML DA.
- ***Python*** (50), ***Twisted*** (52)**,** ***Tkinter*** (54)**,** ***Zsi*** (58)**.**

Para la interfaz gráfica de usuario se utilizó el editor *guide* de Matlab, un ambiente de desarrollo que proporciona un conjunto de herramientas, que simplifican el proceso de diseño y construcción de interfaces gráficas de usuario o GUIs. *Guide* tiene las características básicas de todos los programas visuales como Visual Basic y Visual C++ (59).

3.4.4. Diagrama de archivos

La figura 3-10 presenta el diagrama de archivos utilizados en la implementación del cliente OPC XML en Matlab. Se crearon dos archivos: el primero denominado “cliente.m” (ver recuadro1), encargado de la interfaz gráfica, y el segundo cliente.py (ver recuadro 3), que permite la comunicación OPC XML haciendo uso, del módulo XDAClient de PyOPC (ver recuadro 4).

Figura 3-10: Diagrama de archivos del MCOXM

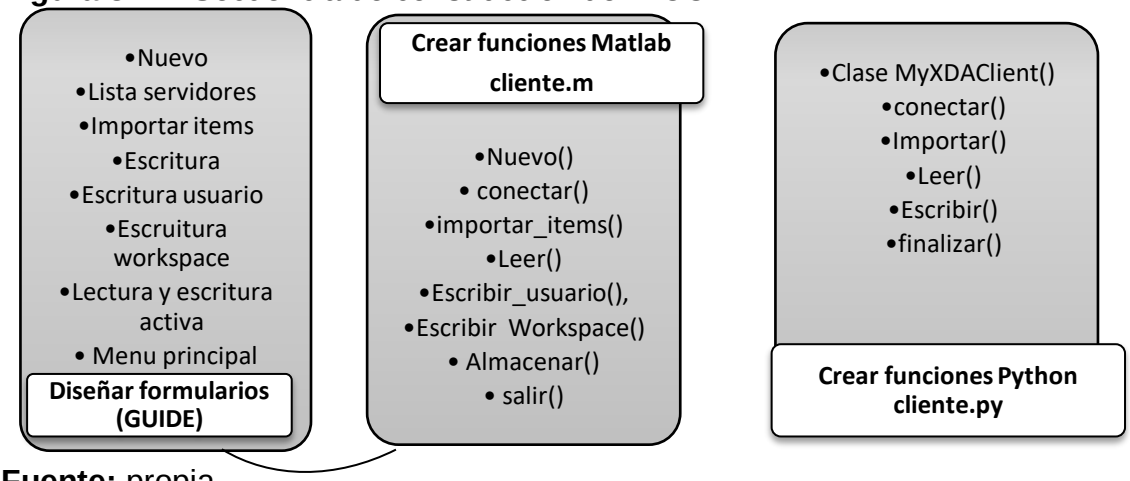


Fuente: propia

3.4.5. Funcionamiento interno

La figura 3-11 presenta la secuencia de construcción del módulo MCOXM; inicialmente se creó la interfaz gráfica, mediante la herramienta Guide que proporciona Matlab, luego en el archivo "cliente.m" se crearon las funciones que se encargan de tomar los datos del formulario, establecer la comunicación OPC mediante la librería cliente.py y retornar los valores correspondientes al usuario o al entorno Matlab.

Figura 3-11: Secuencia de construcción del MCOXM



Fuente: propia

Los formularios y menús creados con la herramienta GUIDE están distribuidos de la siguiente manera:

- ***Formulario nuevo:*** consta de un cuadro de texto, donde se ingresa la dirección del servidor y un botón que permite al usuario establecer la conexión con el servidor.
- ***Formulario lista servidores:*** permite visualizar en una lista todos los servidores OPC XML, con los cuales se ha establecido la comunicación.

- ***Formulario importar ítems:*** presenta una lista de todos los objetos ítems, existentes en el servidor OPC XML, y permite al usuario seleccionar que ítems son los que va a utilizar para la lectura y escritura de datos.
- ***Formulario escritura:*** permite seleccionar entre las dos formas de escritura.
- ***Formulario de escritura de usuario:*** permite al usuario especificar el valor del objeto ítem
- ***Formulario de escritura desde el Workspace:*** Asocia el objeto ítem a una variable almacenada en el Workspace, para que el valor de esta variable se actualice periódicamente en servidor OPC XML.
- ***Formulario de lectura y escritura activa:*** Permite al usuario activar la lectura y escritura dinámica de datos.

En la parte superior del formulario principal se presenta un menú con las opciones: archivo, importar variables y comandos.

- ***Menú archivo:*** presenta dos opciones: Nuevo y salir.
- ***Importar variables:*** visualiza el formulario importar variables.

A continuación se describe cada una de las funciones implementadas en el archivo "cliente.m", y su interacción con la librería cliente.py diseñada en Python. Estas funciones están clasificadas de acuerdo a los sub-módulos expuestos en el numeral 3.4.1.

a) Conexión servidor

Función nuevo: permite visualizar el formulario nuevo **(RFMCOXM2)**.

Función conectar: cuando el usuario hace clic en el botón adicionar del formulario "nuevo", se llama a la función conectar () de la clase MyXDAclient, para establecer la comunicación OPC, si la conexión es exitosa se adiciona este servidor en el *formulario* lista de servidores **(RFMCOXM3)**.

b) **Configuración de ítems**

Función importar ítems: esta función se encarga de presentar el formulario importar ítems, y almacenar en el Workspace los nombres de las los objetos ítems seleccionados para su posterior uso. Hace uso de la función importar de la librería inicio.py **(RFMCOXM4 y RFMCOXM5)**.

c) Lectura y escritura dinámica de objetos

Función escribir-usuario: se encarga de capturar los valores de los objetos ítems escritos por el operario, y ennviarlos al servidor OPC XML, mediante el uso de la función **"**write**"** del módulo cliente.py.

Función escribir-*Workspace*: se encarga de enviar al servidor los datos provenientes de las variables almacenadas en el *Workspace*.

Función leer: se encarga de capturar las propiedades de los objetos ítems del servidor OPC XML y enviarlos al *Workspace*, haciendo uso de la función "read" del módulo client.py.

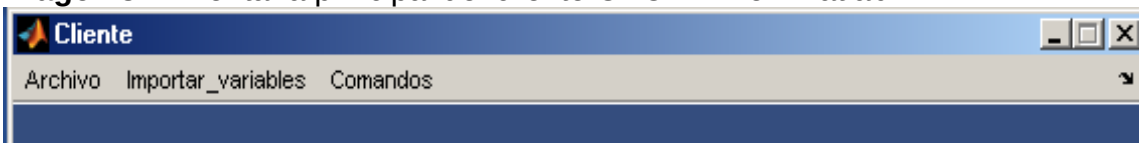
Función de lectura y escritura activa de datos: se encarga de llamar periódicamente a las funciones leer/escribir para que capturen y visualicen el valor de cada objeto ítem.

Función salir: se encarga de presentar el menú de finalización y terminar la comunicación OPC.

3.4.6. Usando el módulo cliente OPC XML en Matlab

Al ejecutar el archivo cliente.m aparece la ventana principal del cliente OPC XML, la cual contiene los menús: "Archivo", "Importar_variables" y "Comandos" , ver imagen 3-4.

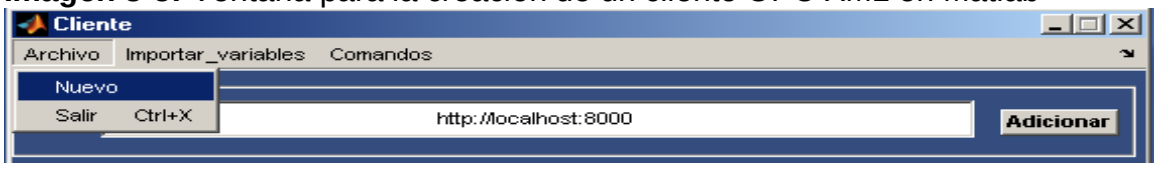
Imagen 3-4: Ventana principal del cliente OPC XML en Matlab



Fuente: propia

Para conectarse con un servidor OPC XML se debe ir a menú "archivo/nuevo" donde aparece un formulario que permite digitar el nombre del servidor, ver imagen 3-5, cuando el usuario presiona el botón "Adicionar", se realiza la petición de conexión al servidor, si la conexión es exitosa el nombre del servidor se adiciona al formulario "lista de servidores".

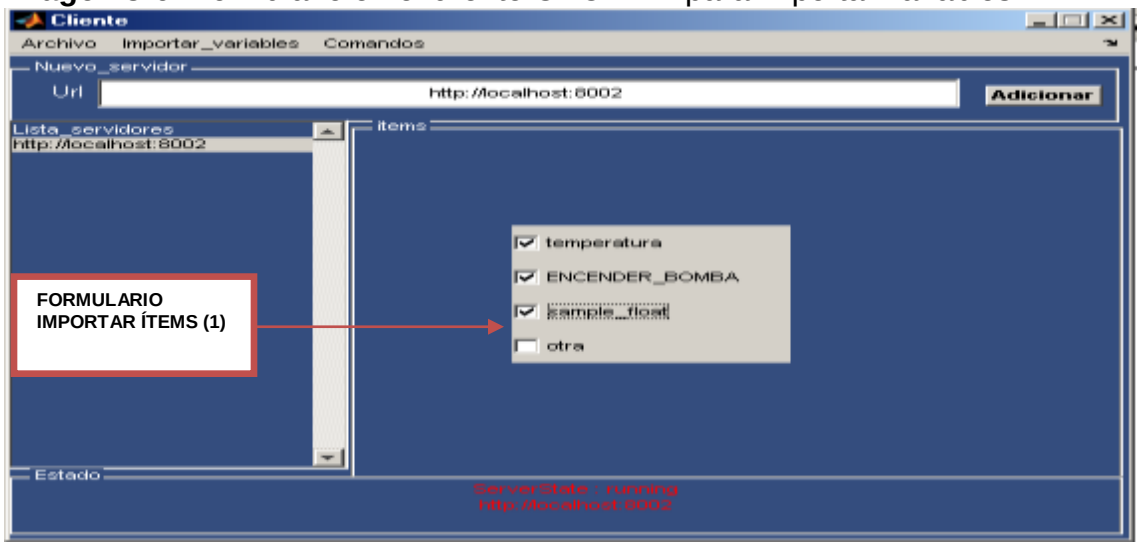
Imagen 3-5: Ventana para la creación de un cliente OPC XML en Matlab



Fuente: propia

Al seleccionar el menú "importar_variables" se presenta un formulario con todos los objetos ítems disponibles en el servidor OPC XML, ver recuadro 1 de imagen 3-6, cuando se ha seleccionado los objetos ítems a utilizar, se almacena la información en el workspace para su posterior uso.

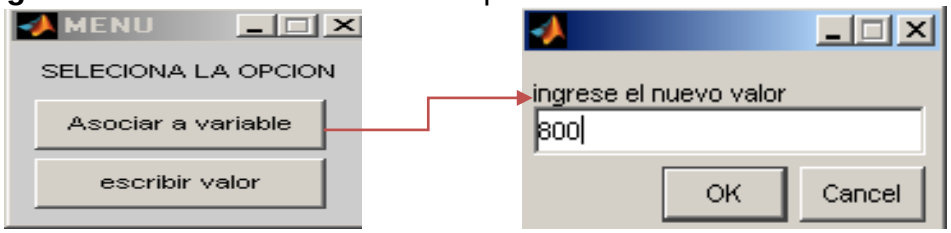
Imagen 3-6: Formulario en el cliente OPC XML para importar variables.



Fuente: propia

Después de que el nombre de las variables se almacena en el *Worspace*, se debe ir al menú" comandos" para configurar las variables de escritura. Para la escritura de datos se presentan dos opciones, una es asociarla a una variable almacenada en el *Workspace*, y la otra es escribir un valor directamente sobre un formulario; si el usuario selecciona la segunda opción "escribir valor" (ver imagen 3-7a, se presenta otro formulario, ver imagen 3-7b, donde puede ingresar el valor del objeto del ítem seleccionado, para luego enviarlo al servidor OPC XML.

Imagen 3-7: Formulario del MCOXM para la escritura en servidores OPC XML



a) Formulario de escritura b) Formulario de escritura de usuario

Fuente: propia

Todas las variables importadas junto con sus valores provenientes del servidor OPC XML, se almacenan en el *Workspace*, para que sus valores puedan ser utilizados por otros programas de Matlab o el entorno *Simulink*.

3.4.7. Instalación y manual de usuario

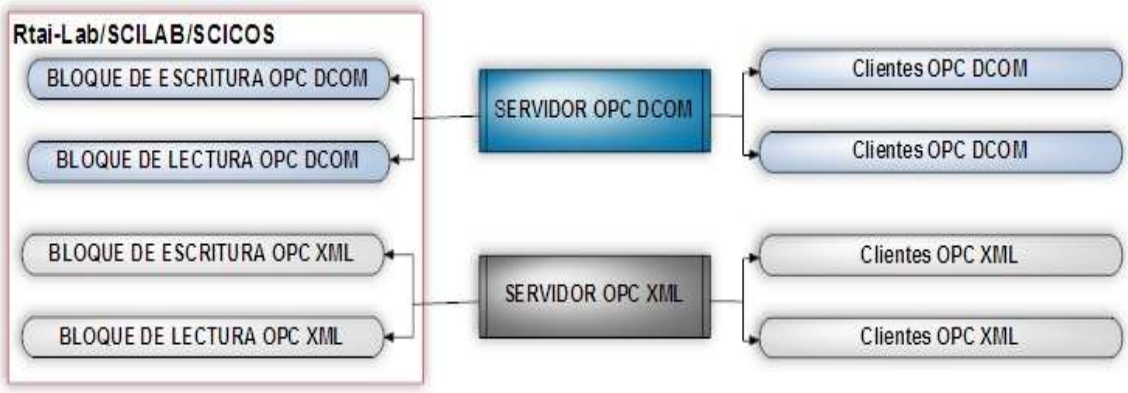
En el anexo D (guía de instalación y manual de usuario del cliente OPC XML en Matlab) se presenta una guía de instalación y manual de usuario del módulo cliente OPC XML en Matlab.

3.5. MÓDULOS CLIENTE OPC DCOM (MCODR) Y XML (MCODR) EN RTAI-LAB

En esta sección se describe la construcción de los módulos cliente OPC DCOM y XML en Rtai-Lab definidos en la sección 2-3, inicialmente se realiza el diagrama general de comunicación entre los clientes OPC en Rtai-Lab y los servidores OPC, luego se detallan los componentes básicos según los requerimientos expuestos en las tablas 2-2 y 2-3, se exponen las restricciones generales, se describen las herramientas utilizadas para su implementación, se realiza un diagrama de archivos, se detalla el funcionamiento interno de cada componente y finalmente se describe el uso de estos módulos.

Para la implementación de los módulos OPC en Rtai-Lab se hace uso de *Scicos,* un editor de diagrama de bloques que puede ser utilizado para crear simulaciones e implementar tareas en tiempo real. Se diseñaron cuatro bloques (lectura y escritura tanto para DCOM como XML) los cuales son capaces de comunicarse con los servidores OPC DCOM y XML, ver figura 3-12.

Figura 3-12: Esquema general de los módulos cliente OPC DCOM/XML en Rtai-Lab



Fuente: propia

3.5.1. Componentes básicos de los módulos MCODR/MCOXR.

Basados en los requerimientos **RFMCODR1 y RFMCOXR1** expuestos en la tabla 2-2y 2-3, se detallan los componentes de los módulos cliente OPC DCOM y XML en Rtai-Lab, los cuales se exponen a continuación:

Componentes del MCODR

Bloque de lectura en servidores OPC DCOM: se debe permitir al usuario configurar el bloque de lectura con los parámetros: IP, host, dirección servidor

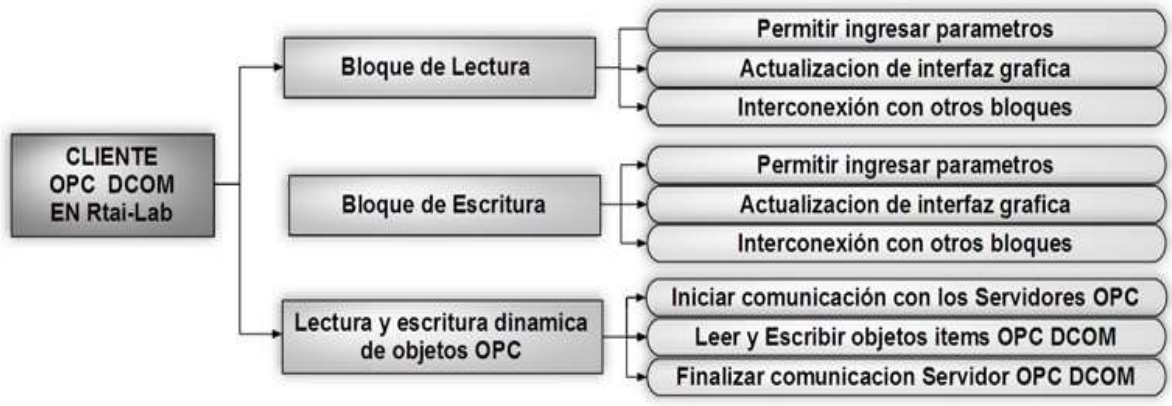
OPC DCOM, número de salidas, dirección de las variables **(RFMCODR2)**; después de configurados estos parámetros se debe actualizar la configuración funcional y gráfica del bloque, con los nuevos parámetros **(RFMCODR3)**, finalmente se debe permitir conectar las salidas del bloque con otros bloques creados en Scicos, para que la información proveniente de este se pueda visualizar **(RFMCODR4)**.

Bloque de escritura en servidores OPC DCOM se debe permitir al usuario configurar el bloque de escritura con los parámetros: IP, host, dirección servidor OPC DCOM, el número de entradas, dirección de las variables **(RFMCODR5)**, después de configurados estos parámetros se debe actualizar la interfaz gráfica del bloque con los nuevos parámetros **(RFMCODR6),** finalmente se debe permitir conectar las entradas del bloque con otros bloques creados en Scicos, para que la información se pueda enviar al servidor OPC DCOM **(RFMCODR7)**.

Lectura y escritura dinámica de objetos OPC: se debe iniciar la comunicación cuando el usuario lo solicite, en esta parte se debe estar leyendo y escribiendo constantemente en los objetos ítems de los servidores OPC **(RFMCODR8-RFMCODR9)**, se debe permitir al usuario finalizar la comunicación con el servidor OPC **(RFMCODR10).**

En la figura 3-13, se presentan los componentes y requerimientos para el módulo cliente OPC DCOM en Rtai-Lab.

Figura 3-13: Componentes del módulo cliente DCOM en Rtai-Lab



Fuente: propia

Componentes del módulo MCOXR

Bloque de lectura en servidores OPC XML: se debe permitir al usuario configurar el bloque de lectura con los siguientes parámetros: dirección servidor OPC XML, el número de salidas, dirección de las variables **(RFMCOXR2)**, se debe actualizar la configuración funcional y gráfica del bloque con los nuevos parámetros

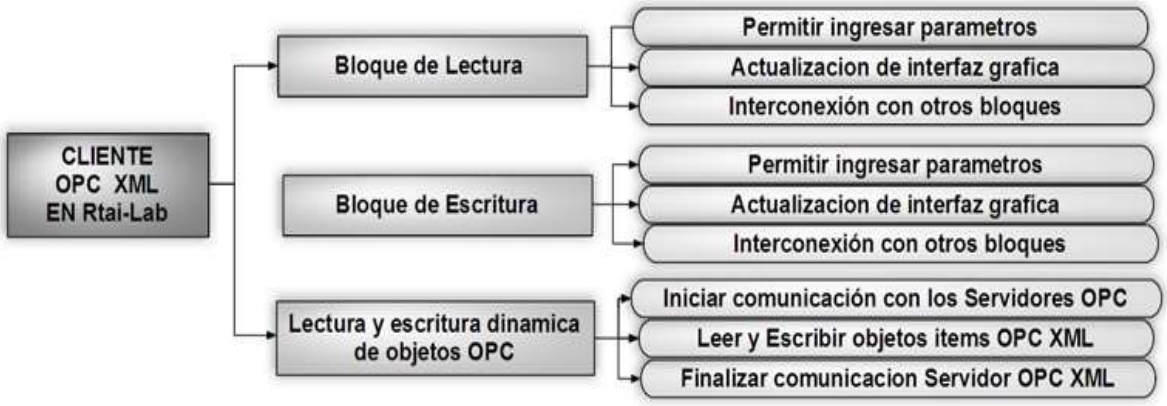
(RFMCOXR3), finalmente se debe permitir conectar las salidas del bloque con otros bloques creados en Scicos, para que la información proveniente de este se pueda visualizar **(RFMCOXR4)**.

Bloque de escritura en servidores OPC XML: se debe permitir al usuario configurar el bloque de escritura con los siguientes parámetros: dirección servidor OPC XML, el número de entradas, dirección de las variables **(RFMCOXR5)**, después de configurados estos parámetros se debe actualizar la configuración funcional y gráfica del bloque, con los nuevos parámetros **(RFMCOXR6),** finalmente se debe permitir conectar las salidas del bloque con otros bloques creados en Scicos, para que la información proveniente de este se pueda visualizar **(RFMCOXR7)**.

Lectura y escritura dinámica de objetos OPC: se debe iniciar la comunicación cuando el usuario lo solicite, en esta parte se debe estar leyendo y escribiendo constantemente en los objeto ítems de los servidores OPC **(RFMCOXR8)**, los bloques de lectura deben capturar los valores provenientes de los servidores OPC DCOM/XML. Cada valor de la salida debe tener asociado un valor proveniente de cada ítem del servidor **(RFMCOXR9),** se debe permitir al usuario finalizar la comunicación con el servidor OPC **(RFMCOXR10).**

En la figura 3-14, se presentan los componentes y requerimientos para el módulo cliente OPC XML en Rtai-Lab.

Figura 3-14: Componentes del módulo cliente OPC XML en Rtai-Lab



Fuente: propia

3.5.2. Restricciones generales de MCODR/ MCOXR

Limitaciones de HW

- ***Procesador:*** cualquier procesador Intel/AMD x86, o superior.
- ***RAM:*** 1G o superior

- ***Sistema operativo:*** cualquier distribución Linux que permita la instalación de Rtai-Lab.
- ***Tamaño en disco duro:*** 220 Megabytes

Otras restricciones

- El computador donde se instale los módulos MCODR y MCOXR debe estar conectado a una red LAN, o tener acceso a internet.
- Rtai-Lab debe estar previamente instalado y configurado.
- Los servidores OPC con los que se establece la comunicación deben estar inicializados.
- Rtai-Lab debe tener configurado Scilab/Scicos (60), en su versión 4.1.2 o superior.

3.5.3. Herramientas software utilizadas en MCODR/ MCOXR

Para cumplir con los requerimientos establecidos en las tablas 2-2, 2-3, 2-6 y 2-7, se realizó un análisis de las herramientas que permiten la implementación de clientes OPC DCOM y XML. Dentro de las librerías que permiten el diseño de clientes en el estándar DCOM se encuentran: JEASYOPC y OpenOPC (61). Realizando el análisis de los requerimientos no funcionales presentados en la tabla 2-.7, se estableció que la única librería que cumple los requerimientos **RNFMCODR2** y **RNFMCODR4** es OpenOPC, ya que JEASYOPC no permite la instalación en Linux, por lo tanto para el módulo MCDR se hace uso de OpenOPC (61), que es un conjunto de herramientas cliente OPC-DCOM-DA de fuente abierta diseñadas en Python.

Para el módulo cliente OPC XML en Rtai-Lab se hizo uso de la herramienta PyOPC, la cual contiene una librería (XDAclient) para la comunicación con servidores OPC en la tecnología XML, además permite cumplir los requerimientos establecidos en las tablas 2-3, 2-6, 2-7.
Tanto OpenOPC como PyOPC requieren para su funcionamiento Python (50), un lenguaje de programación multiplataforma, que viene instalado por defecto en todas las versiones de Linux.

3.5.4. Diagrama de archivos

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.

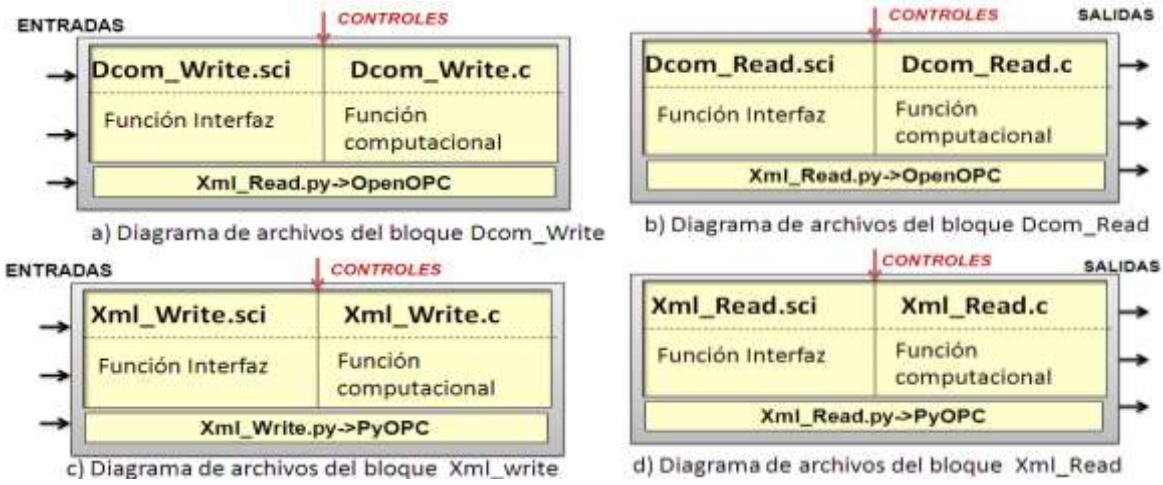
Librerías en *Python*: para los bloques OPC DCOM se hace uso de la librería OpenOPC, y para los bloques OPC XML se utiliza PyOPC.

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 3-15: Diagrama de archivos del MCODR y MCOXR



Fuente: propia

3.5.5. Funcionamiento interno de los módulos clientes OPC en Rtai-Lab

Como se mencionó en la sección 3.5.1 el MCODX está compuesto por tres sub-módulos: el bloque de lectura (Dcom_Read), el bloque de escritura (Dcom_Write) y la inicialización de proceso. A continuación se describen las características, interfaz de usuario y función computacional de los dos sub-módulos Dcom_Read y Dcom_Write.

a) Bloque Dcom_Read

Este bloque se encarga de leer las propiedades de los objetos ítems creados en los servidores OPC DCOM, y tiene asociados los requerimientos funcionales: **RFMCODR2, RFMCODR3, RFMCODR4,** establecidos en la tabla 2-2. Dentro de las características que presenta este bloque se encuentran:

- ***Entradas:*** cero.
- ***Salidas:*** el usuario puede leer máximo cinco variables por bloque.
- ***Parámetros que el usuario puede configurar****:* nombre de host, cantidad de salidas del bloque, nombre del servidor OPC DCOM, dirección de las variables asociadas a cada salida.
- ***Función interfaz:*** Dcom_Read.sci
- ***Función Computacional:*** Dcom_Read.c
- ***Librería OPC:*** Dcom_Read.py

La función Interfaz: en el recuadro rojo de la figura 3-16 se presenta una parte del código de la función Dcom_Read.sci, el cual contiene la inicialización de la estructura de datos, para la generación de la interfaz gráfica del bloque OPC Dcom Read.

Figura 3-16: Código para la construcción de la interfaz del bloque Dcom_Read

```
case 'define' then
        variables1=[];
        varnam='servidora';
         name='MBX1'
        ipaddr='0'
        slope=0;iout=0;
        stt=0;out=1;
        ip='192.168.120.76';
        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(name);
                    length(ipaddr);
                    ascii(name)';
                    ascii(ipaddr)']
        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''
    x=standard_define([3 2],model,exprs,gr_i)
end
endfunction
```

Fuente: propia

Dialogo para adquirir los parámetros del bloque: **(RFMCODR2)**, cuando el operario hace clic derecho sobre el bloque Read_Dcom se presenta un formulario que permite ingresar dos parámetros: el primer parámetro que se requiere es el nombre de host (un nombre descriptivo, o una dirección IP como 192.168.16.32) que identifica a ese equipo en la red, este es utilizado para determinar la disposición servidores OPC en este equipo, el segundo parámetro es el número de salidas que tendrá el bloque, este valor debe ser un número entre 1-5.

El siguiente código permite crear un formulario con los parámetros mencionados anteriormente:

[ok,ip,outport,exprs]=getvalue('Parametros OPC_read_dcom', 'IP'; 'No de salidas'],list('str',1,'vec',-1),exprs).
Después de escrita la IP del host, se presenta un segundo formulario que muestra una lista de todos los servidores disponibles, y permite la selección del servidor OPC DCOM a utilizar, esto se realiza mediante la función fservidorDR, la cual esta implementada en Scilab y utiliza la función OPC.server() de la librería Dcom_read.py.

Luego se presenta un formulario que lista todos objetos ítems disponibles en el servidor, el operario deberá seleccionar in objeto ítem para cada salida. Un cuarto formulario indica los objetos ítems del servidor en cuadros de texto para que el usuario pueda modificarlos. Finalmente se actualizan los parámetros, con los datos ingresados en los formularios y se grafica el bloque con el nuevo número de salidas establecidas **(RFMCODR3)**.

Función Computacional: la función computacional implementada para el bloque Dcom Read se denomina Dcom_Read.c, la cual se encarga de capturar continuamente los datos de los objeto ítems del servidor OPC DCOM mediante la librería Dcom_Read.py, y enviar su valor correspondiente a cada salida del bloque. El simulador Scicos, mediante la función de interfaz Dcom_Read.sci, llama a la función computacional Dcom_Read.c para realizar diferentes tareas (task), utilizando una variable status (flag), y le suministra, como parámetros, la IP del host, el nombre del servidor OPC DCOM y los objetos ítems que el usuario ha seleccionado.

La imagen 3-8 presenta el código de la función Dcom_Read, la cual hace uso de las siguientes funciones:

- ***Actualización de las salidas (inout):*** se encarga de llamar a la función "leer()" del archivo "Dcom_Read.py" para obtener los valores de las salidas, a esta función le envía como parámetros: la IP del host, el nombre del servidor OPC DCOM y la dirección de los objetos ítems. La función "read()" lee las propiedades de cada objeto ítem del servidor OPC y retorna el nombre y el

valor de cada ítem, finalmente a cada salida del bloque se le asigna el valor correspondiente de cada objeto ítem.

- ***Tiempo de salida de eventos (end)****:* esta función se encarga de finalizar la comunicación con el servidor OPC.

- ***Inicialización (init):*** esta función permite establecer valores iníciales de cada una de las salidas y otras variables auxiliares.

Imagen 3-8: Código para la implementación de la función Dcom_Read.c

```
void Dcom_Read(scicos_block *block,int flag)
{
  if (flag==1){              /* set output */
    inout(block);
  }
  else if (flag==5){         /* termination */
    end(block);
  }
  else if (flag ==4){        /* initialisation */
    init(block);
  }
}
```

Fuente: propia

b) <u>Bloque OPC Dcom Write</u>

Este bloque se encarga de la escritura de los objetos ítems en los servidores OPC DCOM, y debe satisfacer los requerimientos funcionales: **RFMCODR5**, **RFMCODR6** y **RFMCODR7** establecidos en la tabla 2-2. Dentro de las características que presenta este bloque se encuentran:

- ***Salidas:*** cero.
- ***Entradas:*** el usuario puede fijar por bloque máximo cinco entradas.
- ***Parámetros que ingresa el usuario:*** El nombre de host, número de entradas, nombre del servidor OPC DCOM y dirección de las variables para cada entrada.
- ***Función interfaz:*** Dcom_Write.sci
- ***Función Computacional:*** Dcom_Write.c
- ***Librería OPC:*** Dcom_Write.py

<u>La función Interfaz:</u> en el recuadro rojo de la figura 3-17, se presenta una parte del código de la función Dcom_Write.sci, el cual contiene la inicialización de la estructura de datos, para la generación del bloque OPC Dcom Write. En la línea 3 se realiza la llamada a la función computacional Dcom_Write.c.

Figura 3-17: Código para la construcción de la interfaz del bloque Dcom_Write

```
case 'define' then
        variables1=[];
        varnam='servidora';
         name='MBX1'
        ipaddr='0'
        slope=0;iout=0;
        stt=0;out=1;
        ip='192.168.120.76';
        rpar=[out];
        model=scicos_model()
        model.sim=list('Dcom_write',4)
        model.in=[1:in]'
         model.evtin=1
        model.out=[]
        model.rpar=rpar
        model.ipar=[length(name);
                length(ipaddr);
                ascii(name)';
                ascii(ipaddr)']
        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'']' ;
    x=standard_define([3 2],model,exprs,gr_i)
end
endfunction
```

Fuente: propia

Dialogo para adquirir los parámetros del bloque: los parámetros que el operario debe configurar en este bloque son similares al bloque Dcom_Read expuesto anteriormente; el primer parámetro de información que se requiere es el nombre de host (un nombre descriptivo, o una dirección IP como 192.168.16.32) que identifica a ese equipo en la red, el número de entradas, el nombre del servidor y el nombre de las variables a las que se le va a enviar el valor, la diferencia consiste en que en este bloque se generan entradas y no salidas. La función interfaz recibe los parámetros, los actualiza y gráfica el bloque con el nuevo número de entradas establecidas.

Función Computacional: la función computacional Dcom_Write.c se encarga de capturar continuamente los datos de cada entrada del bloque y enviarlos a los objetos ítems del servidor OPC DCOM. El simulador Scicos llama a la función computacional Dcom_write.c mediante la función de interfaz Dcom_Write.sci, para realizar diferentes tareas (task) utilizando una variable status (flag). La imagen 3-9 presenta el código de la función Dcom_Write, la cual hace uso de las siguientes funciones:

- ***Inicialización (init):*** permite inicializar las variables auxiliares y el servidor OPC DCOM.

- ***Actualización de entradas y envio de datos al servidor (inout):*** las tareas realizadas en este punto son: leer los valores correspondientes a cada entrada, asignar los datos a cada variable (cada entrada debe tener asociado el nombre de una variable, se crea una estructura para poder enviar esta información al servidor), y escribir la estructura en el servidor OPC DCOM mediante la librería Dcom_Write.py.

- ***Tiempo de salida de eventos (end):*** esta función se encarga de finalizar la comunicación con el servidor OPC DCOM.

Imagen *3-9:* Código para la implementación de la función Dcom_Write.c

```
void Dcom_Write(scicos_block *block,int flag)
{

  if (flag==1){          /* set output */
    inout(block);
  }
  else if (flag==5){     /* termination */
    end(block);
  }
 else if (flag==4){     /* termination */
    init(block);
  }
}
```

Fuente: propia

En sección 3.5.1 se detalló el MCODX, el cual está compuesto por tres sub-módulos: el bloque de lectura (Xml_Read), el bloque de escritura (Xml_Read) y la inicialización de proceso. A continuación se detalla la construcción de los bloques Xml_Read y Xml_Write. Inicialmente se describen las características, luego se detalla la interfaz de usuario y finalmente se expone la función computacional.

c) Bloque OPC Xml Read

Este bloque se encarga de la lectura de objetos ítems desde los servidores OPC XML, y satisface los requerimientos funcionales **RFMCOXR2**, **RFMCOXR3** y **RFMCOXR4** establecidos en la tabla 2-3. Este bloque presenta las siguientes características:

- ***Entradas:*** cero
- ***Salidas:*** se puede leer máximo cinco variables por bloque.
- ***Parámetros que debe ingresar el usuario*****:** nombre del servidor OPC XML, número de salidas, dirección de la variable para cada salida.
- ***Función interfaz:*** Xml_Read.sci
- ***Función Computacional:*** Xml_Read.c
- ***Librería OPC*****:** Xml_Read.py

La función Interfaz: el código de la función interfaz del bloque Xml_Read es similar al bloque Dcom_Read, simplemente se realiza un cambio en la línea 3; se cambia la función computacional Dcom_Read.c por Xml_Read.c, como se indica en el siguiente código: **model.sim=list(Xml_Read',4).** La imagen 3.10 presenta el bloque construido con la función Xml_Read.sci, en este caso, el valor por defecto del número de salidas es uno.

Imagen 3-10: Bloque Xml_Read

Fuente: propia

Dialogo para adquirir los parámetros del bloque: cuando el operario hace clic derecho sobre el bloque Xml_Read, se visualiza un formulario que permite configurar dos parámetros: el primero es el nombre del servidor OPC XML, el segundo parámetro es el número de salidas que tendrá el bloque. Las siguientes líneas de código permiten la visualización de este formulario:

[ok,ip,outport,exprs]=getvalue('Parametros OPC xml Read', 'IP';'No de salidas'],list('str',1,'vec',-1),exprs)

Luego de ingresar el nombre del servidor, y la cantidad de salidas para el bloque, se presenta un segundo formulario, en el cual, se presenta los objetos ítems disponibles en el servidor establecido, esto se realiza mediante la función "fvariablesXR", como lo indica la siguiente línea de código:

[variables3, variables4]=fvariablesXR(servidor_x,ip,outport)

El siguiente paso consiste en actualizar los parámetros creados inicialmente con los datos ingresados en los formularios y graficar el bloque con el nuevo número de salidas establecidas.

Función Computacional: la función computacional implementada para el bloque Xml_Read se encarga de capturar continuamente los datos de los objetos ítems del servidor OPC XML y enviar su valor correspondiente a cada salida del bloque.
El simulador Scicos mediante la función de interfaz Xml_Read.Sci llama a la función computacional Xml_Read.c para realizar diferentes tareas (task). La imagen 3-11 presenta el código de la función Xml_Read, la cual hace uso de las siguientes funciones:

- ***Inicialización (init):*** establecer valores iníciales para la salidas e inicializar la comunicación con los servidores OPC XML.
- ***Actualización de las salidas (inout):*** la función Xml_Read.c le envía el nombre del servidor y de las variables a la función implementada en Python Xml_Read.py, esta función captura las propiedades de cada objeto ítem del servidor OPC y las retorna para que se asigne los datos a cada salida.
- ***Tiempo de Salida de eventos (end):*** esta función se encarga de finalizar la comunicación con el servidor OPC.

Imagen 3-11: Código para la implementación de la función Xml_Read.c

```
void Xml_Read(scicos_block *block,int flag)
{
  if (flag==1){          /* set output */
    inout(block);
  }
  else if (flag==5){     /* termination */
    end(block);
  }
  else if (flag ==4){    /* initialisation */
    init(block);
  }
}
```

Fuente: propia

d) Bloque OPC Xml Write

Este bloque se encarga de la escritura de objeto ítems en los servidores OPC XML y debe satisfacer los requerimientos funcionales **RFMCOXR5**, **RFMCOXR6** y **RFMCOXR7** establecidos en la tabla 2-3. Dentro de las características que presenta este bloque se encuentran:

- ***Salidas:*** Cero
- ***Entradas:*** se puede escribir por bloque máximo cinco variables.
- ***Parámetros que debe establecer el usuario****:* nombre del servidor OPC XML, número de entradas, dirección de la variable para cada entrada.
- ***Función interfaz:*** Xml_Write.sci
- ***Función computacional:*** Xml_Write.c
- ***Librería OPC:*** Xml_Write.py

La función interfaz: el código de la función interfaz del bloque Xml_Write es similar al bloque Dcom_Write, simplemente se realiza un cambio en la línea 5 donde se cambia la función computacional Dcom_Write por Xml_Write, como se indica en el siguiente código: **model.sim=list(Xml_Write',4)**

La imagen 3-12 presenta el bloque construido con la función Xml_Write.sci, en este caso, el valor por defecto del número de entradas es uno.

Imagen 3-12: Bloque Xml_Read

Fuente: propia

Dialogo para adquirir los parámetros del bloque: cuando el operario hace clic derecho sobre el bloque, se visualiza un formulario, que permite configurar dos

parámetros: el primero es el nombre del servidor OPC XML, el segundo parámetro es el número de salidas que tendrá el bloque. Esto se realiza mediante la siguiente línea de código:

[ok,ip,outport,exprs]=getvalue('Parametros OPC xml Write', 'IP';'No de Entradas'],list('str',1,'vec',-1),exprs)

Al ingresar los parámetros en el primer formulario, se presenta un segundo formulario con los objetos ítems disponibles en el servidor seleccionado, esto se realiza mediante la siguiente línea de código:

[variables3, variables4]=fvariablesXW(servidor_x,ip,outport)

El siguiente paso consiste en actualizar los parámetros creados inicialmente con los datos ingresados en los formularios y graficar el bloque con el nuevo número de salidas establecidas. La configuración de estos parámetros es igual que el bloque Xml_Read por lo tanto no se presenta nuevamente en este documento.

Función computacional: se encarga de capturar continuamente los datos de las entradas de los bloques y enviar su valor correspondiente a cada objeto ítem del servidor OPC XML. El simulador Scicos mediante la función de interfaz Xml_Write.sci llama periódicamente a la función computacional Xml_Write.c para realizar diferentes tareas. La imagen 3-13 presenta el código de la función Xml_Write, la cual hace uso de las siguientes funciones:

- ***Inicialización (init):*** permite inicializar las variables auxiliares y el servidor OPC XML.
- ***Actualización de entradas y envío de datos al servidor (inout)****:* las tareas realizadas en este punto son: leer los valores correspondientes a cada entrada, asignar los datos a cada variable, y escribir la estructura en el servidor OPC XML mediante la librería Xml_Write.py.
- ***Tiempo de salida de eventos (end)****:* esta función se encarga de finalizar la comunicación con el servidor OPC XML.

Imagen 3-13: Código para la implementación de la función Xml_Write.c

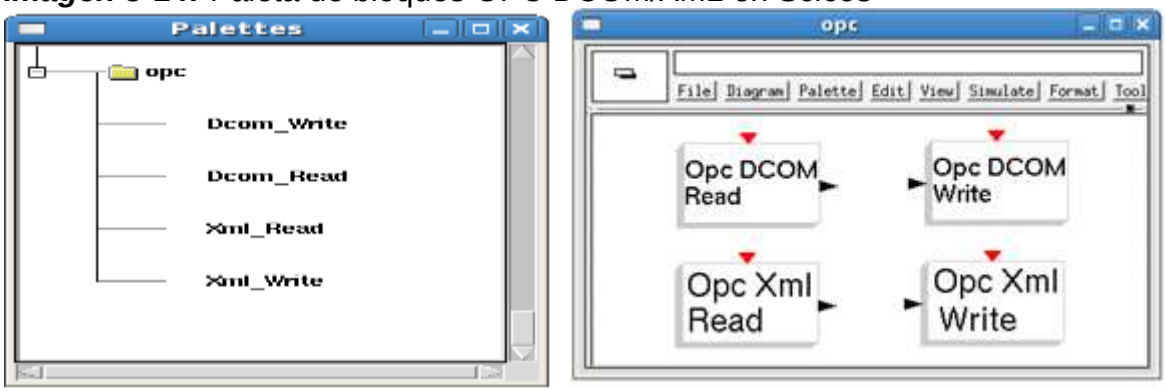
```
void Xml_Write(scicos_block *block,int flag)
{
  if (flag==1){              /* set output */
    inout(block);
  }
  else if (flag==5){         /* termination */
    end(block);
  }
 else if (flag==4){          /* termination */
    init(block);
  }
}
```

Fuente: propia

3.5.6. Usando los módulos OPC en Rtai-Lab

Para implementar los bloques en Scicos se debe guardar las librerías que contienen las funciones de interfaz (Dcom_Read.sci, Xml_Read.sci, Dcom_Write.sci, Xml_Write.sci) en la carpeta /usr/local/Scilab-4.1.2-rtailab/macros, y las computacionales en la carpeta /usr/local/Scilab-4.1.2-rtailab/device, luego se compilan estas librerías, se ingresa a Scilab/Scicos para adicionar cada uno de los bloques a una paleta llamada OPC. Para mayor información ir al anexo E sección II-2 (guía de instalación y configuración de los clientes OPC DCOM/XML en Rtai-Lab). Si todo está configurado correctamente, la paleta se debe visualizar como se indica en la imagen 3-14.

Imagen 3-14: Paleta de bloques OPC DCOM/XML en Scicos



a) Paleta en árbol b) paleta con visualización de bloques

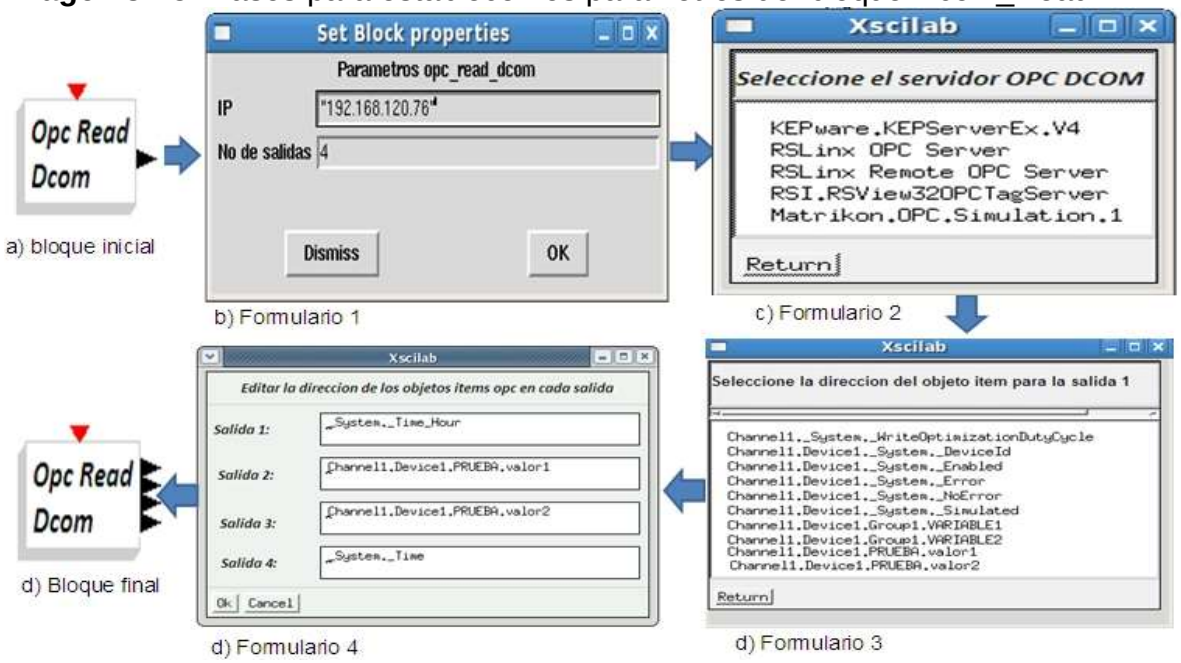
Fuente: propia

El operario debe seleccionar el bloque de su interés, configurar los parámetros correspondientes y realizar las respectivas conexiones. La imagen 3-15 presenta los formularios necesarios para la configuración de los parámetros del bloque Dcom_Read; el primer formulario, ver imagen 3-15b, permite definir la dirección del servidor OPC DCOM y el número de salidas, al presionar ok se presenta el segundo formulario, ver imagen 3-15c, donde indica una lista de servidores disponibles, al seleccionar el servidor con el que se va a trabajar, se presenta un tercer formulario, ver imagen 3-15-d, que lista los objetos ítems pertenecientes a dicho servidor, y finalmente se presenta el cuarto formulario, ver imagen 3-15e, que permite editar los objetos ítems finales para obtener el bloque final, ver imagen 3-15f.

La imagen 3-16 presenta los formularios necesarios para la configuración de los parámetros del bloque Dcom_Write; el primer formulario, ver imagen 3-16b, permite definir la dirección del servidor OPC DCOM y el número de entradas, al presionar ok se presenta el segundo formulario, ver imagen 3-16c, donde indica una lista de servidores disponibles, al seleccionar el servidor con el que se va a trabajar, se presenta un tercer formulario, ver imagen 3-16d, que lista los objetos

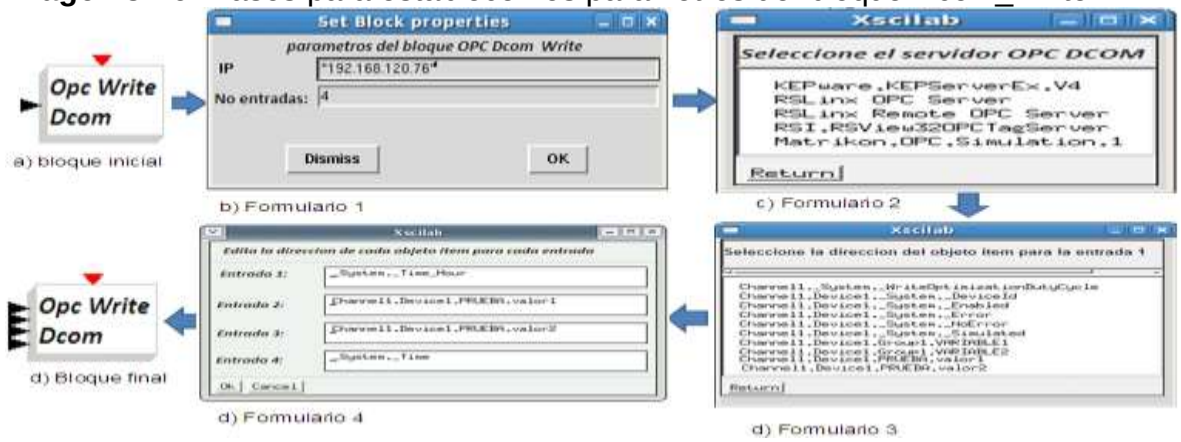
ítems pertenecientes a dicho servidor, y finalmente se presenta el cuarto formulario, ver imagen 3-16e, que permite editar los objetos ítems finales para obtener el bloque final, ver imagen 3-16f.

Imagen 3-15: Pasos para establecer los parámetros del bloque Dcom_Read



Fuente: propia

Imagen 3-16: Pasos para establecer los parámetros del bloque Dcom_Write

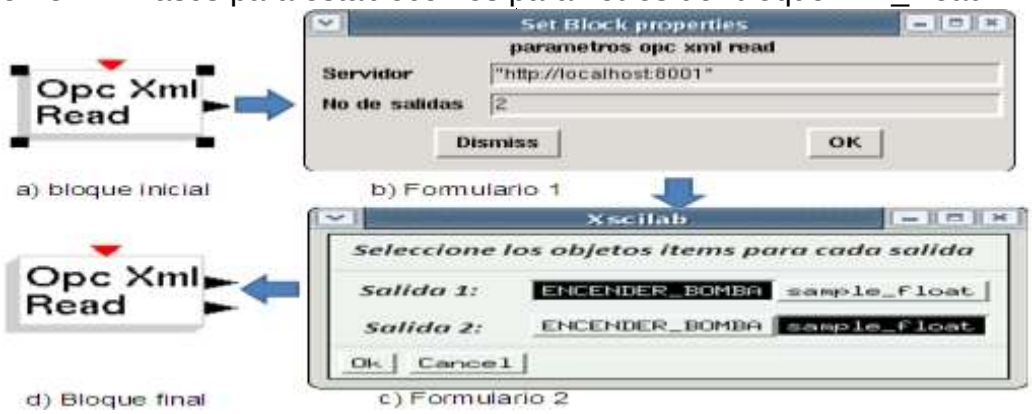


Fuente: propia

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 imagen 3-17a, hacer clic derecho en el bloque para que se presente un segundo formulario, ver imagen 3-17b, que permite ingresar la dirección del servidor OPC XML y el número de salidas, si el usuario presiona ok se presenta un tercer formulario, ver ima-

gen 3-17c, con todos los objetos ítems configurados en el servidor OPC XML, se debe seleccionar el objeto ítem correspondiente a cada salida, finalmente se diseña y configuran los nuevos parámetros, ver recuadro 3-17d.

Imagen 3-17: Pasos 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 imagen 3-18a, hacer clic derecho en el bloque para que se presente un segundo formulario, ver imagen 3-18b, 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 imagen 3-18c, 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 recuadro 3-18d.

Imagen 3-18: Pasos para establecer los parámetros del bloque Xml_Write

Fuente: propia

Después de establecidos los parámetros para cada bloque, se inicia la interconexión con otros bloques de scicos/Rtai-Lab, se realiza la compilación y se genera la tarea de tiempo real asociada al diagrama, estos pasos se indican en la sección III

del anexo E (guía de instalación y configuración de los clientes OPC DCOM/XML en Rtai-Lab).

3.5.7. Instalación y manual de usuario

En el Anexo E (guía de instalación y configuración de los clientes OPC DCOM/XML en Rtai-Lab) se presenta una guía de instalación y manual de usuario de los módulos cliente OPC DCOM/XML para Rtai-Lab.

3.6. RESUMEN

En este capítulo se detalla la selección y construcción de los módulos: MSOD, MSOX, MCODM, MCOXM, MCODR y MCOXR para la arquitectura de integración de plataformas de control basadas en PC mediante OPC. Para el módulo servidor OPC DCOM se opta por el uso del servidor KepserverEX por sus características técnicas, y por el conocimiento que se tiene de este servidor en el laboratorio de control de procesos, para el módulo servidor OPC XML se describe la construcción de un servidor utilizando unas librerías denominadas PyOPC y Tkinter, para el módulo cliente OPC DCOM en Matlab se especifica las herramientas existentes que permiten la comunicación con servidores OPC DCOM, para el Módulo OPC XML en Matlab se detalla la construcción de un programa que permite interconectar a Matlab/Simulink con los servidores OPC XML, y finalmente se describe la implementación de cuatro bloques (Dcom_Read, Dcom_Write, Xml_Read, Xml_Write) en Scicos/Scilab que permiten la comunicación de Rtai-Lab con los servidores OPC en las tecnologías OPC DCOM DA y OPC XML DA.

4. IMPLEMENTACION DEL MÓDULO DE MONITOREO Y SUPERVISION WEB (MM&SW)

Basado en los requerimientos y alcances del capítulo 2 con respecto al MM&SW, en este capítulo se desarrolla el módulo de monitoreo y supervisión web con soporte OPC. Inicialmente se realiza una descripción general del módulo, luego se detallan los sub-módulos según los requerimientos expuestos en la tabla 2-5, las restricciones generales, las herramientas utilizadas, el diagrama de archivos, el funcionamiento interno y finalmente se describe el uso del MM&SW.

4.1. DETALLE DE LOS SUBMÓDULOS

El propósito del módulo de monitoreo y supervisión web es que los usuarios puedan diseñar pantallas, y monitorear los procesos desde cualquier punto con conexión a internet, ver numeral 2.3.7, para tal fin se implementa una aplicación web que obtiene información de los servidores OPC DCOM/XML, y permite la creación y ejecución de pantallas HMI, visualización de alarmas, eventos y tendencias.

Según el requerimiento **RFMM&SW1** de la tabla 2-5, el sistema se compone de tres sub-módulos los cuales se presentan en la figura 4-1, y se exponen a continuación:

4.1.1. Gestor de contenido

Un gestor de contenido permite crear y mantener un sitio web con facilidad, encargándose de los trabajos más tediosos que hasta ahora ocupaban el tiempo de los administradores de la web (62). Según la tabla 2-5, este sub-módulo debe permitir: gestión de usuarios (**RFMM&SW3**), visualizar usuarios conectados **(RFMM&SW4)**, crear y visualizar encuestas (**RFMM&SW5**), edición de contenido (**RFMM&SW6**) y acceder a enlaces de interés (**RFMM&SW7**).

4.1.2. Diseño de proyectos

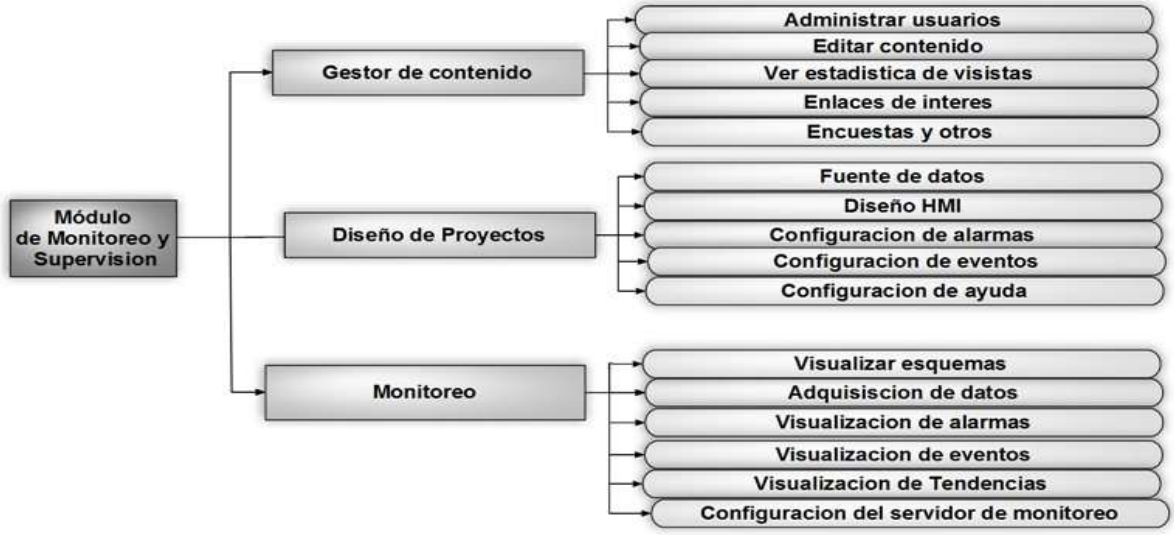
Este sub-módulo debe permitir representar gráficamente los procesos que reflejan el comportamiento dinámico de los sistemas de control de procesos y los datos de campo. Según la tabla 2-5 este sub-módulo debe cumplir los siguientes requerimientos funcionales: crear **(RFMM&SW8)**, editar **(RFMM&SW9)**, eliminar **(RFMM&SW10),** visualizar **(RFMM&SW11)** pantallas HMI**,** importar objetos ítems de los servidores OPC DCOM/XML **(RFMM&SW12),** utilizar objetos de diseño predefinidos **(RFMM&SW13)**, importar imágenes desde otras aplicaciones **(RFMM&SW14),** utilizar objetos de diseño de tipo industrial **(RFMM&SW15)**, animar cada objeto en función de las variables **(RFMM&SW16- RFMM&SW17.)**,

administrar sistemas de alarmas y eventos **(RFMM&SW18)** y creación de un manual de usuario por cada proyecto.

4.1.3. Monitoreo de procesos

Este sub-módulo debe permitir el monitoreo de sistemas de control de procesos. Según la tabla 2-5 este sub-módulo debe cumplir los siguientes requerimientos funcionales: ejecutar las acciones de mando pre-programadas a partir de los valores actuales de variables leídas (**RFMM&SW19),** visualizar, almacenar y controlar alarmas (**RFMM&SW20),** visualizar eventos **(RFMM&SW21),** graficar variables en función del tiempo **(RFMM&SW22),** contar con un manual de ayuda **(RFMM&SW23)** y salir del sistema **(RFMM&SW24).**

Figura 4-1: Diagrama general de los componentes del módulo de monitoreo y supervisión web



Fuente: propia

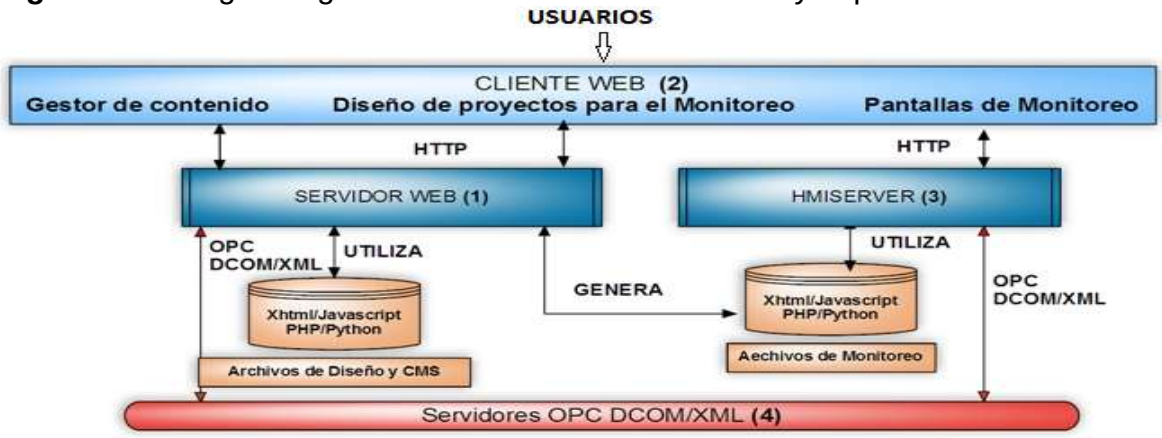
4.2. FUNCIONAMIENTO GENERAL DEL MM&SW

La figura 4-2 representa el diagrama general de comunicaciones del módulo de monitoreo y supervisión web, el cual es un sistema cliente/servidor, que se compone de:

- ***Un servidor web (recuadro 1):*** programa que sirve para atender y responder a las diferentes peticiones de los navegadores, proporcionando los recursos que soliciten y usando el protocolo HTTP, se utiliza el sub-módulo gestor de contenido y diseño de proyectos.

- ***HMIServer (recuadro 3):*** Conjunto programas encargados del intercambio de información entre los servidores OPC y los clientes web, se utiliza para el sub-módulo de monitoreo.
- ***Cliente web (recuadro 2):*** programa mediante el cual el usuario solicita al servidor web el envío de información. Permite la visualización de pantallas al usuario mediante un navegador.
- ***Archivos HTML y XHTML:*** aquí se establece el código fuente de las páginas web (HTML, Java script, PHP).
- ***Servidores OPC DCOM/XML (recuadro 4):*** se encargan de la captura de datos de las plataformas/dispositivos de control.

Figura 4-2: Diagrama general del módulo de monitoreo y supervisión web



Fuente: propia

La descripción de funcionamiento del módulo de monitoreo y supervisión web, se realiza en las siguientes dos etapas:

1. El servidor web (recuadro 1), espera peticiones de los clientes web, recuadro 2, los cuales mediante navegadores permiten al usuario realizar las peticiones, los usuarios realizan las peticiones (ver enlaces, identificarse, registrarse, diseñar pantallas, configurar alarmas etc.), el cliente web envía la petición al servidor web, el servidor web recibe la petición, busca el recurso, y lo retorna utilizando la misma conexión por la que recibió petición.

2. El usuario ingresa a las pantallas de monitoreo por medio de un cliente web, el servidor HMIServer (recuadro 3) inicia el intercambio de datos con los servidores OPC (recuadro 4), y almacena la información en memoria para que el cliente web pueda hacer uso de ella, y la visualice en la pantalla del navegador.

El funcionamiento del módulo de monitoreo y supervisión web consiste en cíclicamente ejecutar las dos anteriores etapas.

4.3. RESTRICCIONES GENERALES DEL MM&SW

4.3.1. Limitaciones de Hardware

- ***Procesador:*** cualquier procesador Intel/AMD x86, o superior.
- ***RAM:*** 1G o superior.
- ***Sistema operativo:*** Windows XP o superior / Linux cualquier versión.
- ***Tamaño en disco duro:*** 1 Gigabyte.

4.3.2. Otras restricciones

- El computador donde se instala el servidor web debe estar conectado a una red LAN, o tener acceso a internet.
- Los clientes web deben acceder a la página mediante un navegador web moderno que permita la reproducción de páginas XHTML.
- Los servidores OPC con los que se establece la comunicación deben estar inicializados y configurados previamente.

4.4. HERRAMIENTAS UTILIZADAS EN EL MM&SW

A continuación se listan las tecnologías utilizadas en la construcción del MM&SW:

- ***HTML (Lenguaje de Marcado de Hipertexto):*** es la base de casi todas las páginas web, y se utiliza como base para las pantallas del MM&SW.
- ***SVG (Scalable Vector Graphics)****:* permite gráficos interactivos de alta calidad que se incrustan directamente en las páginas web, los gráficos SVG son soportados por casi todos los navegadores web modernos (63).
- ***JSON (Javascript Object Notation):*** este protocolo fue desarrollado para proporcionar un medio sencillo y fiable de las comunicaciones, con funcionalidades de alto nivel (64). Es utilizado en el MM&SW para el transporte de datos desde el servidor HMIServer y el cliente web.

Para la selección de las herramientas se realizó un análisis del *software* que permite cumplir los requerimientos funcionales expuestos en la tabla 2-5 y los requerimientos no funcionales expuestos en la tablas 2-6 y 2-7: las herramientas *software* deben ser de fuente libre **(RNFMM&SW5),** deben permitir su instalación en diversos sistemas operativos (**RNFMM&SW4**). A continuación se listan las herramientas *software* utilizadas en el MM&SW:

- ***MYSQL 3.23.X o superior*** (65)***:*** es un sistema de gestión de base de datos relacional, multiusuario. MYSQL es utilizado en este módulo para almacenar información acerca de los usuarios y el sistema de monitoreo y supervisión.

- ***PHP 4.2.X o superior*** (66)***:*** es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas.
- ***Apache 1.13.19 o superior*** (67)***:*** el servidor HTTP Apache es un servidor web HTTP de código abierto para plataformas Unix, Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP y la noción de sitio virtual. Apache es utilizado en este proyecto para los sub-módulos gestor de contenido y diseño de proyectos.
- ***Javascript:*** es el lenguaje común de programación para páginas web, y es utilizado por el MM&SW para controlar la visualización de los datos y el intercambio de datos con el servidor.
- ***Wampserver*** (68)***:*** *software* que permite instalar y configurar fácilmente en el sistema las últimas actualizaciones tanto del servidor web Apache, como del lenguaje de programación PHP y del servidor de base de datos MySQL.
- ***Python 2.4 o superior****:* lenguaje de programación en donde se implementa el servidor HMI y los clientes OPC DCOM (50).
- ***PyOPC:*** librería que permite la implementación del cliente OPC XML (49).
- ***OpenOPC:*** librería que permite la implementación del cliente OPC DCOM (61).
- ***Svg-Edit:*** *es un* editor en línea de imágenes SVG, utilizado en el MM&SW para la construcción de los esquemas HMI (69).
- ***RTE (Cross-Browser Rich Text Editor):*** es un editor de texto diseñado en Javascript (70), utilizado en el sub-módulo de diseño de pantallas para que el usuario pueda editar la ayuda de cada proyecto.
- ***Hmiserver*****:** servidor diseñado en Python (71), utilizado en el MM&SW para la implementación del sub-módulo monitoreo.
- ***Flot:*** es una biblioteca de Javascript, que produce gráficas de conjuntos de datos (72).

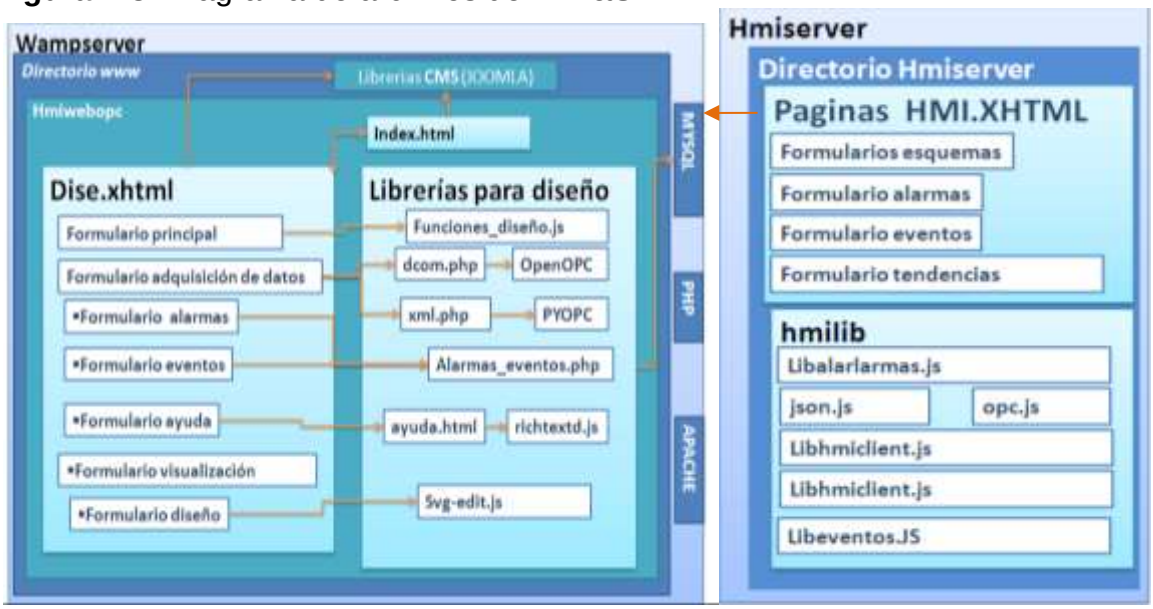
Para la implementación del módulo gestor de contenido se realizaron pruebas, con los siguientes CMS (*Content Management System*) de fuente abierta existentes en Internet: WordPress, Plone, Joomla, Drupal; se validó cada gestor según los requerimientos expuestos en el ítem 2.4.5, y finalmente se seleccionó *Joomla*, porque cumple todos los requerimientos y presenta mayores facilidades (73). Dentro de las características que presenta *Joomla* se encuentran: reducción de los ciclos de desarrollo, disponibilidad de plantillas, edición/publicación de contenidos con un editor similar a Word (**RFMM&SW6)**, visualización previa antes de las publicaciones, publicación automática en fechas predefinidas, ordenar noticias y/o entradas, email, y formato especial para impresión, arquitectura de la información y sistema de navegación (menús), gestor multimedia con generación automática de enlace para publicar ficheros PNGs/ PDFs/ DOCs/ XLSs/ GIFs/ JPEG, envíos remotos de nuevos contenidos para los artículos/noticias/enlaces **(RFMM&SW7)**,

reutilización de los desarrollos de un proyecto a otro, menor dependencia del fabricante de *software* y del integrador, gestor de hilos, capacidad de archivo de contenidos caducados en vez de eliminación directa, papelera, gestor de banners: noticias/votos/encuestas **(RFMM&SW5)**, valoración de contenidos, estadísticas de visitas **(RFMM&SW4)**, total control sobre el aspecto estético del sitio y/o descarga de plantillas instalables en un solo clic, disposición de módulos modificable y administración de usuarios **(RFMM&SW3)**.

4.5. DIAGRAMA DE ARCHIVOS DEL MM&SW

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 de proyectos, 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" contiene el código para 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 de proyectos, esta página se compone de varios formularios, los cuales hacen uso de librerías diseñadas en Javascript, PHP y Python. Cada vez que el usuario modifica un proyecto, esta información se almacena en MYSQL. Cuando el usuario en el formulario de visualización selecciona el proyecto 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 y supervisión, cuando se finaliza el monitoreo se borran todos los archivos asociados a él.

Figura 4-3: Diagrama de archivos del MM&SW



Fuente: propia

4.6. FUNCIONAMIENTO INTERNO DEL MM&SW

4.6.1. Gestor de contenido

Como se mencionó en el numeral 4.2 este sub-módulo debe permitir: gestión de usuarios **(RFMM&SW3)**, visualizar usuarios conectados **(RFMM&SW4)**, crear y visualizar encuestas (**RFMM&SW5**), edición de contenido **(RFMM&SW6)**, acceder a enlaces de interés **(RFMM&SW7)**.

Gestión de usuarios **(RFMM&SW3)**: los usuarios del módulo de monitoreo y supervisión web pueden dividirse en dos categorías principales: invitados y usuarios registrados. Los invitados podrán navegar libremente por todo el contenido, pero tienen restringido el acceso a cierto tipo de contenidos reservados para usuarios registrados, los invitados tampoco podrán monitorear ni diseñar los sistemas de control. Los usuarios registrados pueden acceder al área restringida del sitio, recibiendo privilegios especiales. Los usuarios registrados se dividen en dos grupos: usuarios del sitio (operarios), usuarios del administrador. Los usuarios del sitio disfrutan de ciertos derechos adicionales sobre los visitantes, entre los que se puede incluir la capacidad para crear y publicar contenido en el sitio web. Los usuarios del administrador *(mánager,* administrador y súper-administrador), habitualmente se conocen como administradores del sitio; pueden crear usuarios, modificar menús, contenido, crear proyectos para el monitoreo y supervisión.

Edición de contenidos **(RFMM&SW6):** para la edición de artículos o páginas de contenido, *Joomla* dispone de una barra de herramientas semejante a la de los procesadores de textos con las que se puede dar formato al contenido, añadir imágenes, insertar hipervínculos, etc.

Encuestas **(RFMM&SW5):** el componente '*Poll*' (encuestas), proporciona un método sencillo de crear encuestas breves. Las diferentes encuestas pueden mostrarse en cualquier página o grupo seleccionado. El administrador puede crear y editar las encuestas.

Enlaces **(RFMM&SW7)**: el componente '*Web links*', permite al administrador construir un directorio de enlaces, organizado por categorías.

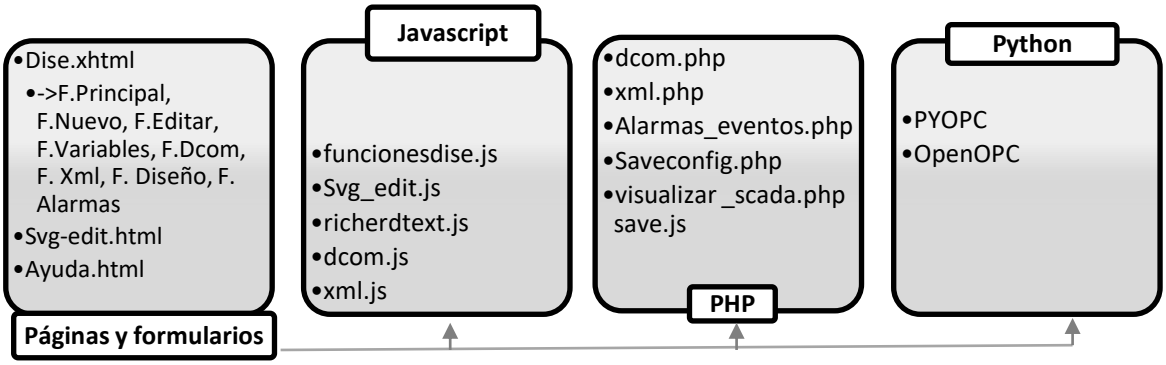
Usuarios conectados. '*Who's Online*' **(RFMM&SW4)**: sistema 'Who's Online', 'Quien esta en línea', presenta una estadística de los usuarios y miembros conectados al sitio web.

Joomla presenta otras utilidades que no están dentro de los requerimientos establecidos y que pueden ser leídas en el manual de usuario de *Joomla* (74).

4.6.2. Diseño de proyectos

La figura 4-4 presenta los componentes del sub-módulo de diseño de proyectos, en el primer recuadro se listan los nombres de cada formulario perteneciente a la página dise.xhtml, en el segundo las funciones diseñadas en javascript, en el tercer recuadro las funciones de PHP, y en el cuarto las librerías en Python.

Figura 4-4: Diagrama de elementos del sub-módulo de diseño de proyectos



Fuente: propia

Para la construcción de este sub-módulo inicialmente, se crea un formulario principal en la página "dise.xhtml", el cual presenta el siguiente menú: nuevo, editar, eliminar, visualizar. El formulario "nuevo", permite ingresar el nombre con el cual se guarda el proyecto, y mediante la función "nuevo ()" de la librería "save.php" se valida y guarda la información en la base de datos **(RFMM&SW8).** El formulario "editar" **(RFMM&SW1)**, presenta un menú con las opciones: fuente de datos, diseño HMI, configuración de alarmas, configuración de eventos, configuración de ayuda, cada uno de los menús tiene asociado los siguientes formularios:

Formulario fuente de datos (**RFMM&SW12)**: este formulario permite crear las variables del módulo de monitoreo y supervisión web, las cuales se las puede importar desde los servidores OPC DCOM, XML, o simplemente el usuario puede establecer los valores. Para la comunicación con los objetos ítems creados en los servidores OPC DCOM se hace uso de la librería "dcom.php", la cual utiliza funciones del módulo OpenOPC. Para capturar las propiedades de objetos ítems de los servidores OPC XML, se hace uso de la de la librería "xml.php", y del módulo XDAClient del paquete PyOPC. Para crear variables auxiliares se hace uso de la función "variables.php". Posteriormente cada variable, con sus propiedades se almacena en la base de datos para que el usuario pueda utilizarlas en el momento de diseñar los esquemas HMI, crear gráficas, y configurar las alarmas/eventos.

Formulario diseño HMI: presenta el siguiente menú:

- ***Nuevo:*** tiene asociado un hipervínculo a la página "svg_edit.html", en la cual se diseñan los esquemas HMI, haciendo uso de las librerías "variables1.php" y "svg_edit.js", la primera librería captura las variables almacenadas en la base de datos, y svg_edit.js valida los datos de los formularios e implementa las paletas de dibujos (**RFMM&SW14)**. Esta página permite al usuario: crear los esquemas de visualización con objetos y formas predefinidos botones, marcos, llaves, tuberías, tanques, y asociar cada objeto ítem incluido en la pantalla en función de alguna variable **(RFMM&SW16)**. Al finalizar se guarda el código en formato SVG y Javascript en la base de datos, para la utilización en el módulo de monitoreo.

- ***Modificar:*** permite modificar cada uno de los esquemas creados, hace uso de las librerías: funciones_dise.js y save.php
- ***Eliminar:*** permite eliminar los esquemas que se han creado, hace uso de la librería funciones_dise.js y save.php
- ***Visualizar:*** presenta una vista previa de cada uno de los esquemas HMI diseñados.

Formulario configuración de alarmas **(RFMM&SW18)**: este formulario permite introducir el nombre la variable, la condición y el mensaje de salida de la alarma, la información introducida en este formulario se almacena en la base de datos para la reutilización en el módulo de monitoreo. Hace uso de la librería alarmas_eventos.php.

Formulario configuración de eventos **(RFMM&SW18)**: este formulario permite introducir el nombre la variable, la condición y el mensaje de salida de cada evento, hace uso del módulo Alarmas_eventos.php.

Formulario configuración de ayuda **(RFMM&SW19)**: es importante que en el momento de monitorear y supervisar un sistema de automatización, el operario tenga la suficiente información disponible acerca del funcionamiento del sistema y del proceso a controlar, para ello se utiliza un editor de texto (*RTE)*, que permite al diseñador crear una guía la cual aparecerá en el menú ayuda de cada proyecto.

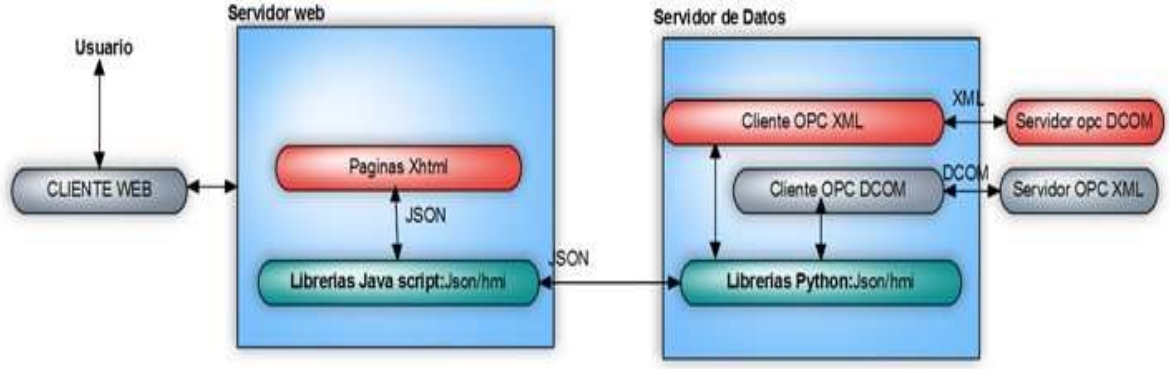
4.6.3. Monitoreo

En el sub-módulo de diseño de proyectos, el operario ya ha creado los esquemas HMI, configurado las alarmas, los eventos y la ayuda; en este sub-módulo se toma el código asociado a cada esquema, se crean los archivos xhtml y las librerías necesarias para el monitoreo de sistemas de control de procesos, se inicia el servidor de datos para cada proyecto, y se presenta una página donde se puede monitorear los sistemas de control de procesos. Este sub-módulo cuenta con el

Hmiserver el cual consta de dos servidores: un servidor web y un servidor de datos para el intercambio de información entre los servidores OPC y el cliente web.

Inicialización de los servidores web y de datos: la figura 4-5 presenta el esquema de comunicación entre los usuarios, el cliente web, el servidor web, el servidor de datos, y los servidores OPC DCOM/XML. Cuando el usuario selecciona el enlace visualizar del menú principal de la página de diseño, se presenta una página con el menú: iniciar servidor, visualizar proyecto y diseño. Al presionar "iniciar servidor" se ejecuta el servidor web y el servidor de datos, internamente se les configura los puertos de comunicación y otros parámetros. Además se crean los archivos necesarios para visualizar los procesos, alarmas, eventos y tendencias, se crea un archivo con el nombre del proyecto y la extensión xhtml; este archivo contiene el código HTML/SVG, también se crean las librerías en Javascript que permiten capturar los valores de las variables OPC, visualizar alarmas, eventos y tendencias en la página, por último se crean las librerías en Python para la comunicación con los servidores OPC DCOM/XML, y un archivo de configuración de las variables.

Figura 4-5: Esquema de comunicación del sub-módulo de monitoreo y supervisión



Fuente: propia

Al presionar clic en el enlace visualizar proyecto, internamente se redirecciona a la URL de la página web donde se puede iniciar el proceso de monitoreo de sistemas de control de procesos.

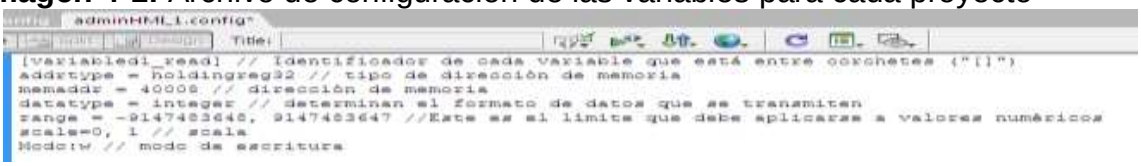
cliente HMI: compuesto por dos partes básicas: la biblioteca de codificación Json, que permite el intercambio de datos entre el servidor y el cliente, y la biblioteca HMI que ofrece las siguientes características: codificación/ decodificación de mensajes, búfer de lectura/escritura, funciones HMI, gestión de alarmas/eventos, actualización automática de la pantalla. A continuación se listan las librerías implementadas en Javascript para el funcionamiento del cliente HMI:

- ***MB_AlarmDisplay:*** contiene los mensajes de los sucesos para cada Alarma.

- ***MB_AlarmHistoryDisplay:*** guarda las alarmas reconocidas por el usuario.
- ***MB_EventDisplay:*** contiene los mensajes de los sucesos para cada evento.
- ***MBT_ReadList:*** almacena el nombre de las variables creadas en el archivo de configuración de direcciones.
- ***MBT_AlarmTex:*** almacena los mensajes que se muestran al operador en el caso de activarse una alarma.
- ***MBT_EventText:*** almacena los mensajes que se presentan al operador en el caso de ocurrir algún evento.
- ***DisplayList:*** es un *array* de objetos que controla los gráficos, texto, tablas, etc.
- ***Libhmiclient:*** captura los datos del servidor de datos, e implementa otras funciones para el monitoreo.
- ***Flot:*** *p*roduce gráficas de conjuntos de datos (72).

Comunicación cliente web/servidor web/servidor de datos/servidor OPC: para que las variables sean compatibles entre el servidor web y el servidor de datos, se requiere de un archivo de configuración con los siguientes parámetros: nombres de los objetos ítems, direcciones, modo de acceso, y otras definiciones establecidas en el sub-módulo de diseño. La imagen 4-1 presenta un ejemplo de configuración para una variable.

Imagen 4-1: Archivo de configuración de las variables para cada proyecto



Fuente: propia

Los tipos de datos disponibles para la configuración de variables son:

- ***Boolean:*** booleanos - (0/1 o *True/False*)
- ***Integer:*** entero - tipos de datos enteros
- ***Float:*** flotante - los tipos de datos de punto flotante
- ***String:*** cadena - los tipos de datos String

El proceso de comunicación entre el cliente web y el servidor de datos es el siguiente: el servidor de datos escribe la información de cada variable en una dirección de memoria establecida en el archivo de configuración, el cliente HMI mediante las librerías Json/Libhmiclient captura estos valores y actualiza la página para que el usuario pueda ver el cambio de datos; así cuando el valor en el servidor OPC cambia, se envía al servidor HMI mediante la función

WriteServerData(func, direc, Qty, [valor2]); **func** representa el formato de los registros, **direc** la dirección en memoria de la variable, **Qty** la cantidad de registros, y **[valor2]** representa un arreglo con los valores tomados del servidor OPC.

Cuando el usuario modifica los parámetros de las variables en la página de monitoreo, la función Libhmiclient toma los valores y los convierte en formato Json, el servidor de datos captura los valores mediante el uso de la función GetServerData(func, adr, qty); donde **func** debe ser un numero entre 1 y 4.(1 lee múltiples datos boléanos, 2 lee múltiples entradas discretas, 3 lee múltiples registros de salida, lee múltiples registros de entrada). **Adr** es la dirección en memoria de cada variable, **Qty:** Representa la cantidad de variables que se van a leer. El valor que se obtiene a partir de esta función es enviado a los servidores OPC, si el servidor OPC es DCOM se utiliza la siguiente función: OPC.write(direccion_varop,valor); donde direccion_varop es la dirección de la variable en el servidor OPC y valor es el dato que se toma de la función Getserverdata. El código para escribir en una variable el servidor OPC es XML es: Xda.write (dirección_variable,valor).

4.7. USO DEL MM&SW

Los usuarios: invitados, registrados y administradores, pueden ingresar mediante cualquier navegador moderno que soporte las páginas XHTML (Mozilla, Opera, Google Croome), y digitar la URL http://ip_servidor/mm&sw/, al ingresar aparece la página principal con diferentes enlaces y un formulario de registro, ver imagen 4-2.

Imagen 4-2: Pantalla principal del módulo de monitoreo y supervisión web

Fuente: propia

Cuando los usuarios se identifican correctamente, se indica una página la cual permite: diseñar encuestas, observar los usuarios en línea, crear anuncios, editar

el contenido y diseñar/monitorear los sistemas de control de procesos, ver imagen 4-3.

Imagen 4-3: Pantalla del módulo de monitoreo y supervisión web para usuarios registrados.

Fuente: propia

Por otra parte el administrador puede ingresar al panel de control, mediante la URL http://ip_servidor/mm&sw/administrator, registrarse y obtener los siguientes privilegios: gestionar usuarios, crear menús, editar contenido, adicionar componentes, configurar parámetros de presentación, entre otros, ver imagen 4-4.

Imagen 4-4: Página que permite administrar el gestor de contenido

Fuente: propia

Diseño de pantallas

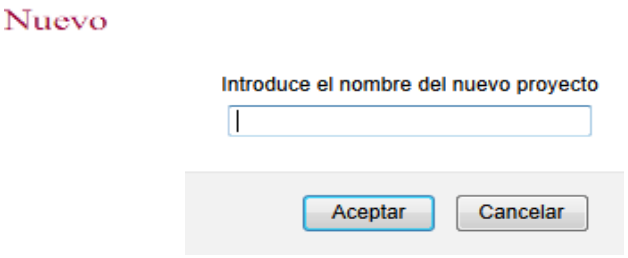
Al dar clic en el enlace “Diseño y Monitoreo” de la página principal, aparece otra página donde se presenta un menú con las opciones: nuevo, eliminar, editar, visualizar, inicio; ver imagen 4-5.

Imagen 4-5: Menú principal de la pantalla de diseño de proyectos

Fuente: propia

Cuando el usuario selecciona la opción “nuevo”, se presenta un formulario que permite ingresar el nombre del proyecto, ver imagen 4-6, esta información se almacena en la base de datos para su posterior uso.

Imagen 4-6. Formulario para crear un nuevo proyecto para el monitoreo y supervisión.



Fuente: propia

Luego de almacenado el nombre del proyecto, se puede iniciar con el diseño de esquemas, para esto el usuario debe seleccionar “editar” del menú principal, para que se indique otro menú con los siguientes ítems: fuente de datos, diseño HMI, configurar alarmas, configurar de eventos, configurar ayuda, ver imagen 4-7.

Imagen 4-7: Menú “Editar” del sub-módulo de diseño de proyectos en el MM&SW

Fuente: propia

A continuación se detalla cada uno de los ítems del menú “editar”.

Fuente de datos: antes de empezar a diseñar los esquemas de los procesos, se debe seleccionar la fuente de los datos e importar los objetos ítems de los servido-

res 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 imagen 4-8.

Imagen 4-8: Menú principal que permite la selección de la fuente de datos

Fuente: propia

Cuando el usuario selecciona como fuente de datos servidor OPC DCOM, aparece un formulario que permite digitar la dirección IP del host donde se encuentra el servidor. Al ingresar la IP se presenta otro formulario que permite seleccionar el nombre servidor y la ruta donde está el grupo de ítems OPC DCOM y un cuarto formulario permite visualizar todos los objetos ítems existentes en el objeto grupo del servidor OPC; se debe elegir los objetos ítems que se van a utilizar para que esta información se almacene en la base de datos, ver imagen 4-9.

Imagen 4-9: Formulario que permite importar ítems desde un servidor OPC DCOM

Fuente: propia

Si el usuario selecciona la opción XML, se presenta un formulario donde se solicita el nombre del servidor OPC XML que al ser registrado se presenta otro formulario, ver imagen 4-10, que permite importar las los objetos ítems creados en dicho servidor para que se almacenen en la base de datos; 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.

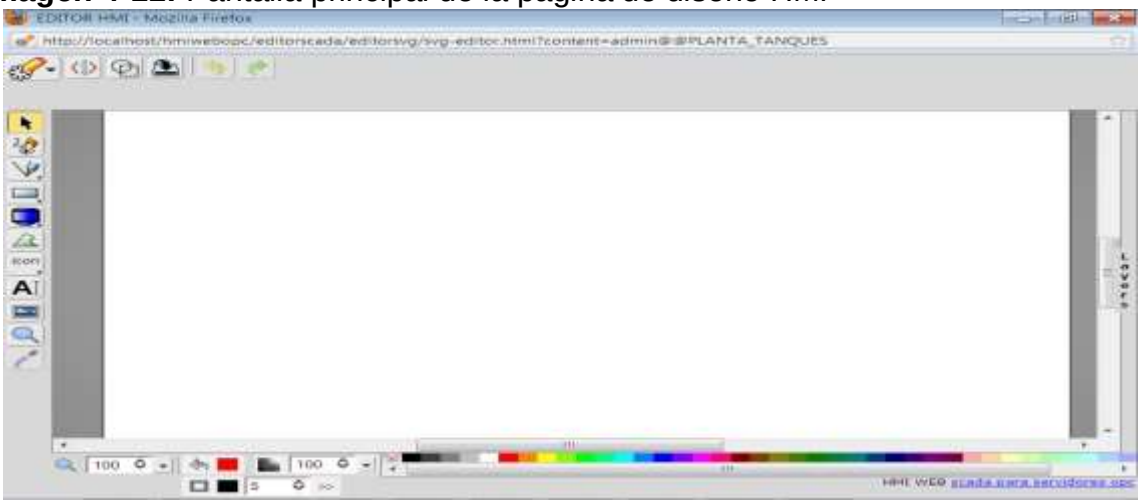
Imagen 4-10: Formulario que permite importar variables desde un servidor OPC XML

Fuente: propia

Diseño HMI: permite representar gráficamente los procesos que reflejan el comportamiento dinámico de los sistemas de automatización y los datos de campo. Al seleccionar "Diseño HMI" aparece un nuevo menú con las opciones: nuevo, modificar, eliminar, visualizar.

El ítem "nuevo" permite ir a una página donde se presenta una pantalla diseño como lo indica la imagen 4-11, en esta página se puede crear un esquema con imágenes en formato SVG y asociarlas a funciones creadas en Javascript para las respectivas animaciones.

Imagen 4-11: Pantalla principal de la página de diseño HMI



Fuente: propia

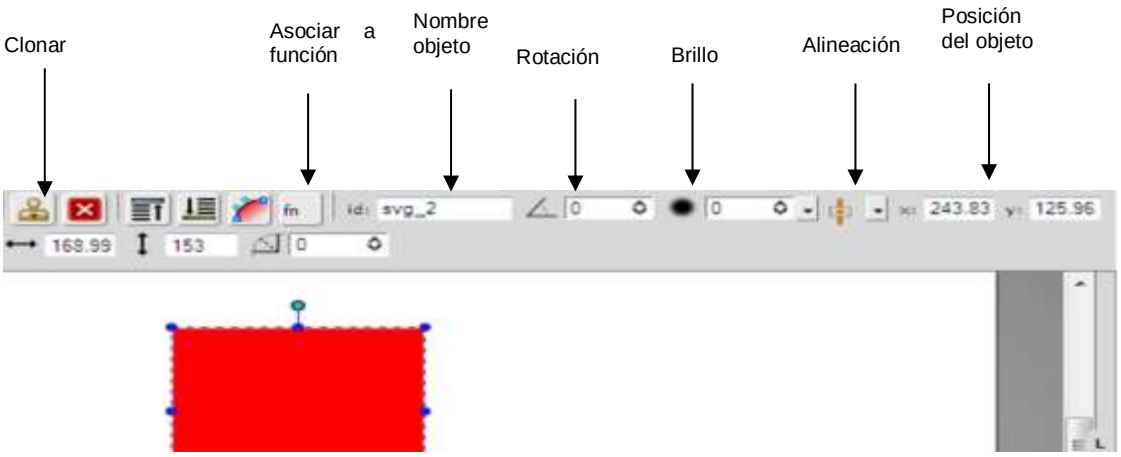
Esta página esta compuesta por:

- ***Menú Esquema*** **:** permite importar y exportar imágenes y la configuración de las propiedades del esquema: nombre, tamaño, lenguaje y color.

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

- ***Paleta de configuracion de objetos:*** La imagen 4-12 presenta la paleta de propiedades de cada objeto seleccionado, dentro de los parametros que se pueden configurar estan: la posicicion, el angulo de rotacion, el color, la intensidad entre otros.

Imagen 4-12: Paleta de propiedades de cada objeto ítem

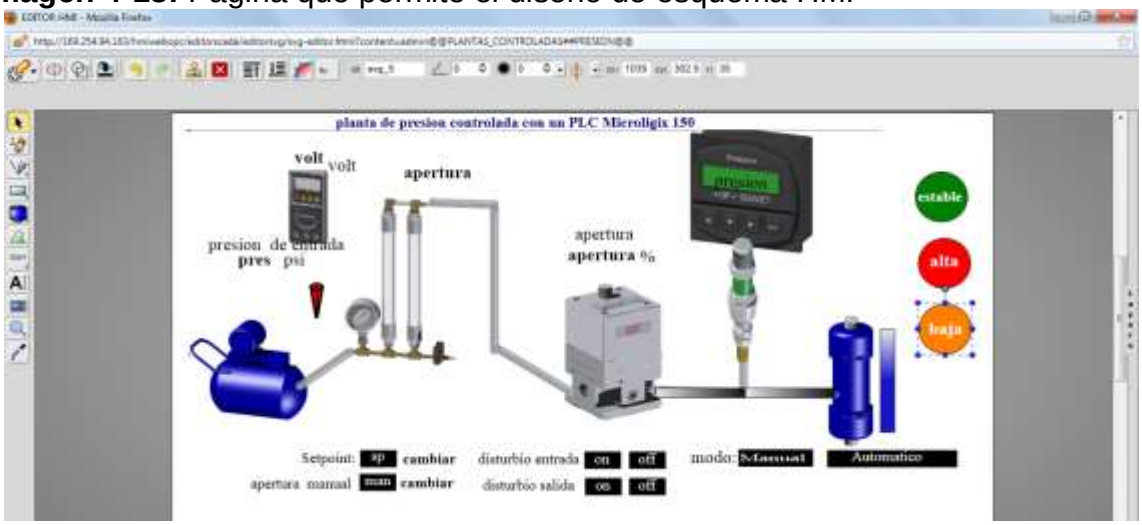


Fuente: propia

- ***Generacion de movimiento y manejo de datos:*** El usuario puede asociar cada una de las variables OPC a cada imagen, ya sea para la visualizacion o para el envio de datos dando clic en el icono fn. Dentro de las funciones implementadas se encuentran: *Writeinmediate:* permite al usuario enviar un valor a un objeto ítem del servidor OPC, *stringDysplay:* toma el valor de una variable y lo visualiza en la pantalla, *Rotate:* rota la imagen según el valor de la variable, *SlideDisplay*: visualiza el cambio de una variable, cambiando de color la imagen, *TankLevel:* visualiza el nivel de un tanque, *PilotLight:* cambia el color de la imagen dependiendo del valor recibido.

La imagen 4-13 presenta una pantalla con un esquema diseñado, haciendo uso de las variables creadas en el menú “fuente de datos” de la pantalla de diseño.

Imagen 4-13: Página que permite el diseño de esquema HMI



Fuente: propia

Finalmente se selecciona el icono  "guardar", el cual se encarga de llamar a las respectivas funciones para que almacenen la información y se presente nuevamente la pantalla de diseño.

Configuración de alarmas: cuando el usuario selecciona la opción "configurar alarmas", se presenta un formulario que permite: asignar un nombre a cada alarma, asociarlo a una variable, establecer una condición, fijar el valor y el mensaje de salida, ver imagen 4-14, esta información se almacena en la base de datos para el uso en el módulo de monitoreo.

Imagen 4-14: Formulario para la creación y configuración de alarmas

Fuente: propia

Configuración de eventos: cuando el usuario selecciona la opción "configurar eventos", se presenta un formulario que permite: asignar un nombre a cada evento, asociarlo a una variable, establecer una condición, fijar el valor y el

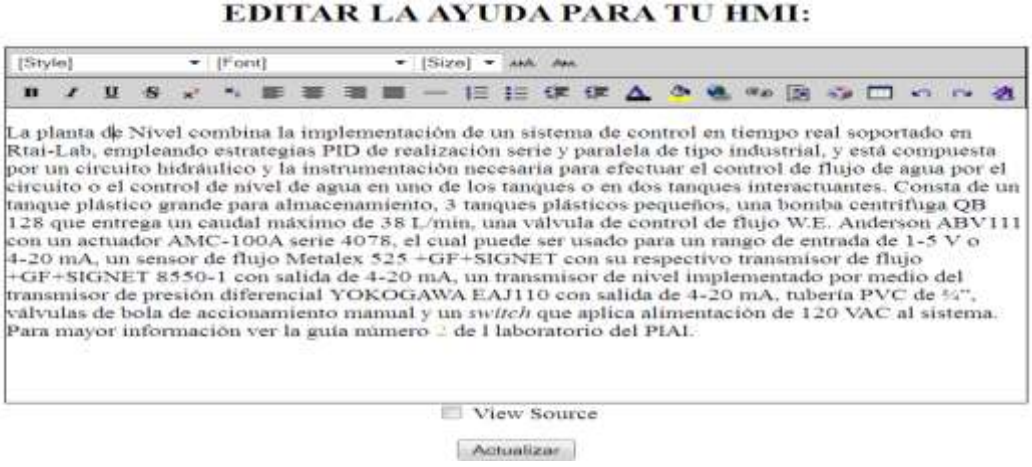
mensaje de salida, ver imagen 4-15, esta información se almacena en la base de datos para el uso en el módulo de monitoreo.

Imagen 4-15: Formulario para la creación y configuración de eventos

Fuente: propia

Configuración de ayuda: al dar clic en "configurar ayuda" del menú principal se presenta un formulario, ver imagen 4-16, donde el usuario registra información referente al uso del proyecto, esta información se almacena en la base de datos para que pueda ser utilizada en el sub-módulo de monitoreo.

Imagen 4-16: Formulario que permite configurar la ayuda para cada proyecto



Fuente: propia

Monitoreo

Cuando el usuario selecciona la opción "visualizar" del menú principal de la página de diseño, aparece un formulario con una lista de proyectos disponibles, si el usuario selecciona el nombre del proyecto, aparece una página con los menús: configuración servidor, visualizar proyecto, diseño, inicio. Al presionar el botón

"iniciar servidor" se ejecuta el servidor HMIServer y se presenta una pantalla que permite el monitoreo y supervisión con las siguientes opciones: esquemas, alarmas, eventos, tendencias, ayuda y salir, ver imagen 4-19.

Imagen 4-17: Pagina que permite el monitoreo

Fuente: propia

La imagen 4-19 presenta una pantalla que permite visualizar las alarmas que se han configurado en el módulo de diseño, 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.

Imagen 4-18: Visualización de alarmas

Alarm:	Estado de Alarma:	Tiempo:	Tiempo OK:	Contador:
La instrumentacion esta apagada	Alarma Activa	Sun Aug 21 2011 02:41:04 GMT-0500	Not OK	1

Fuente: propia

La imagen 4-19 presenta una pantalla que permite visualizar los eventos que se han configurado en el módulo de diseño en ella se ilustra una tabla con la fecha en que ocurrió el evento, el nombre y el estado.

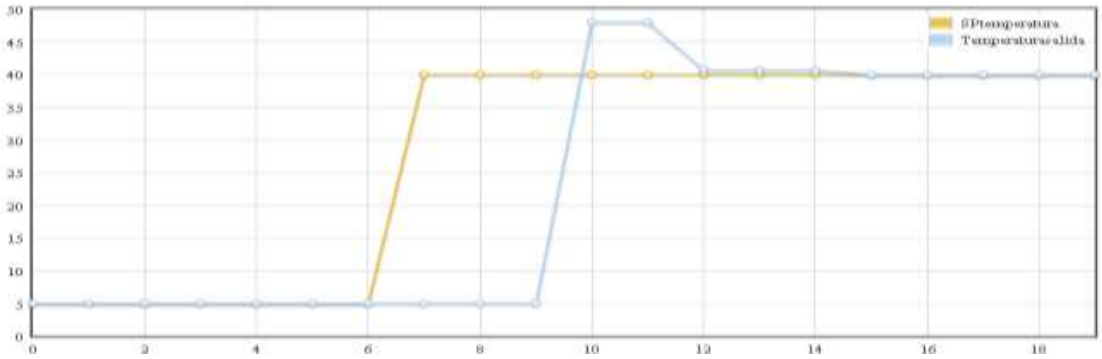
Imagen 4-19: Visualización de eventos

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

Fuente: propia

La imagen 4-20 presenta una pantalla con las tendencias de un control de temperatura en un reactor químico; la línea amarilla representa el *setpoint*, la línea azul la temperatura de salida medida en C°, y el eje de la abscisa el tiempo en segundos.

Imagen 4-20: Tendencias con variación en el tiempo



Fuente: propia

Al presionar clic en el menú “salir” aparece una página de configuración con las opciones: inicio, configuración, control, ayuda, la opción inicio presenta un formulario que permite recargar las variables en el servidor, y finalizar el monitoreo, en configuración se presenta las variables existentes y sus configuraciones, en control se visualiza el estado y las propiedades del servidor HMIServer, y en ayuda se presenta un manual de usuario de esta página. Al finalizar el servidor, se eliminan los archivos creados para el monitoreo y se cierra la conexión establecida. Esta página también contiene un menú con las opciones: Inicio, configuración, control y ayuda.

4.8. INSTALACION Y CONFIGURACION DEL MÓDULO DE MONITOREO Y SUPERVISION WEB

El anexo F (guía de instalación y manual de usuario del módulo de monitoreo y supervisión web) contiene la guía para la instalación y el manual de usuario del MM&SW.

4.9. RESUMEN

En este capítulo se detalla la construcción del módulo de monitoreo y supervisión soportado en tecnología web, que permite el diseño de mímicos y monitoreo de procesos desde un equipo cliente con acceso a internet. Dentro de los componentes del MM&SW se encuentran: procesamiento de datos, diseñador de pantallas, representación de señales de alarma, visualización gráfica dinámica, y

un módulo gestor de contenido, que posibilita la actualización, mantenimiento y ampliación de la web con la colaboración de múltiples usuarios.

5. INTEGRACION Y EVALUACIÓN DE LOS MÓDULOS IMPLEMENTADOS.

En este capítulo se describe la integración y validación de los módulos expuestos en los capítulos 3 y 4 (MSOD, MSOX, MCODM, MCOXM, MCODR, MCOXR, y MM&SW). Se realizaron dos pruebas que permitieron la integración de Matlab y Rtai-Lab de dos maneras diferentes: mediante el estándar OPC DCOM y mediante el estándar OPC XML, y adicionalmente se integró a estas dos plataformas con el PLC Micrologix 1500. Este capítulo inicia con la descripción general de las pruebas realizadas, luego se puntualiza los sistemas controlados, y finalmente se detallan las pruebas realizadas con los resultados obtenidos.

5.1. DESCRIPCIÓN GENERAL DE LAS PRUEBAS REALIZADAS

Para la integración de Matlab, Rtai-Lab y el PLC Micrologix 1500 se utilizaron sistemas de control ya implementados en el laboratorio de control de procesos del PIAI, los cuales fueron modificados para que pudieran realizar la lectura y escritura de objetos ítems en los servidores OPC. Para la prueba de integración mediante el estándar OPC DCOM se hizo uso de las plantas de Nivel y Presión y de una planta simulada en Matlab, y para la integración mediante el estándar OPC XML se hace uso de la planta de nivel y de una planta simulada en Matlab.

La figura 5-1 presenta el esquema general de integración mediante el estándar OPC DCOM, el cual está conformado por las plantas, los esquemas de control, el MCODM, MCODX, el servidor OPC DCOM kepserverEX y el módulo de monitoreo y supervisión web. Las plantas de nivel, temperatura y presión son controladas en Rtai-Lab, Matlab y el PLC Micrologix 1500 respectivamente, los datos obtenidos son enviados al servidor OPC KeepserverEX para que sean capturados por el MM&SW y se puedan representar gráficamente.

Figura 5-1: Esquema general de integración de las plataformas de control mediante OPC DCOM

Fuente: propia

La figura 5-2 presenta el esquema general de integración mediante el estándar OPC XML, el cual está conformado por las plantas, los esquemas de control, el MCOXM, el MCOXR, el MSOX y el MM&SW. Las plantas de nivel/temperatura son controladas en Rtai-Lab/Matlab respectivamente y los datos obtenidos son enviados al servidor OPC XML para que sean capturados por el MM&SW y representados gráficamente.

Figura 5-2: Esquema general de integración de las plataformas de control mediante OPC XML.

Fuente: propia

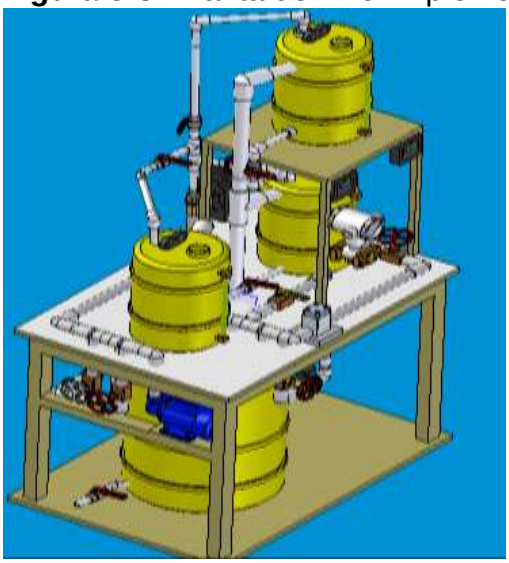
5.2. DESCRIPCIÓN GENERAL DE LOS ESQUEMAS DE CONTROL IMPLEMENTADOS EN LAS PLATAFORMAS Y EN EL PLC

A continuación se describe las plantas y esquemas de control implementados en Matlab, Rtai-Lab y el PLC Micrologix 1500, los cuales se detallan mejor en la sección III del ANEXO H (guía para la integración de Matlab, Rtai-Lab y el PLC micrologix 1500 mediante el estándar OPC DCOM).

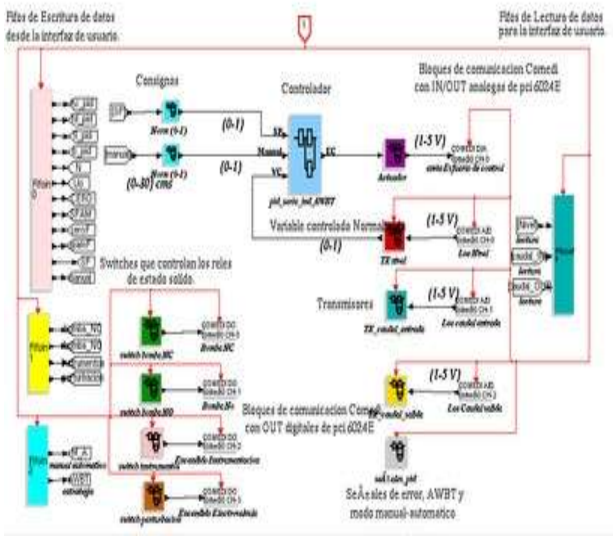
5.2.1. Control PID implementado en Rtai-Lab para la planta de nivel

La figura 5-3a presenta la planta de nivel del laboratorio de control de procesos del PIAI conformada por un circuito hidráulico y la instrumentación necesaria para efectuar el control de nivel de agua en uno de los tanques. La figura 5-3b presenta los diagramas de bloques implementados en Scicos para el control de esta planta.

Figura 5-3: Planta de nivel implementada en el laboratorio del PIAI



a) Planta de Nivel del PIAI



b) diagrama de bloques Scicos

Fuente: propia

5.2.2. Control regulatorio mediante PLC para la planta de presión del Laboratorio de Control de Procesos

La planta de presión existente el laboratorio de control de procesos del PIAI está diseñada para implementar un control regulatorio de presión por medio de un PLC micrologix 1500 de Allen Bradley, ver figura 5-4.

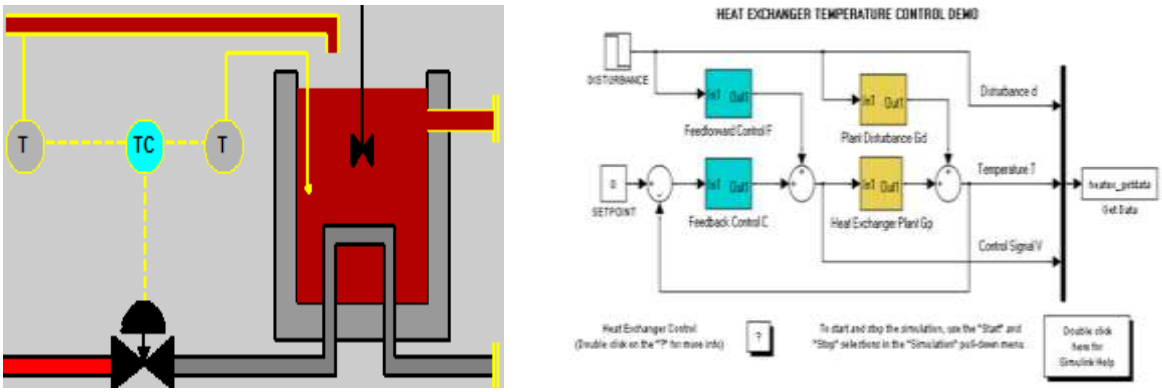
Figura 5-4: Planta de presión implementada en el laboratorio del PIAI

Fuente: propia

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

La figura 5-5a ilustra una representación de un intercambiador de calor, gobernado mediante un controlador PI implementado en Simulink, ver figura 5-5b. La entrada superior proporciona el líquido que se mezcla en el tanque, el cual debe mantenerse 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.

Figura 5-5: Control de temperatura en Matlab para un intercambiador de calor



a) Esquema del intercambiador de calor b) diagrama de bloques en Simulink
Fuente: propia

5.3. PRUEBA 1: INTEGRACIÓN MEDIANTE EL ESTÁNDAR OPC DCOM

Como se describe en el numeral 5.1, la primera prueba consiste en intercambiar información entre Matlab, Rtai-Lab y el PLC micrologix 1500 haciendo uso del servidor OPC DCOM KepserverEX, así mismo esta información es representada y visualizada en el MM&SW. La información detallada de los pasos para realizar esta prueba se organizó en una guía de prácticas denominada: guía para la integración de Matlab, Rtai-Lab y el PLC Micrologix 1500 mediante el estándar OPC DCOM disponible en el anexo H.

5.3.1. Requerimientos hardware y software

La tabla 5-1 presenta los requerimientos hardware y software necesarios para la realización de la práctica de integración de Matlab, Rtai-Lab y el PLC Micrologix 1500 mediante el estándar OPC DCOM.

Tabla 5 -1: Requerimientos para la integración de plataformas académicas mediante el estándar OPC DCOM

			Requerimientos		
Componentes		**Descripción**	**Hardware**	**Software**	
Procesos	**Control 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	Diagrama de bloques para el control de nivel. MCODR
	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 conexión a una red LAN.	Matlab	*MCODM* Diagrama de bloques para el control de temperatura
	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.	
Servidor OPC DCOM		Encargado de integrar la información de las plataformas de control y el PLC.	Un PC con conexión a una red LAN	Servidor KEPServerEX V4.5	
Módulo de monitoreo y supervisión web		Permite 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	

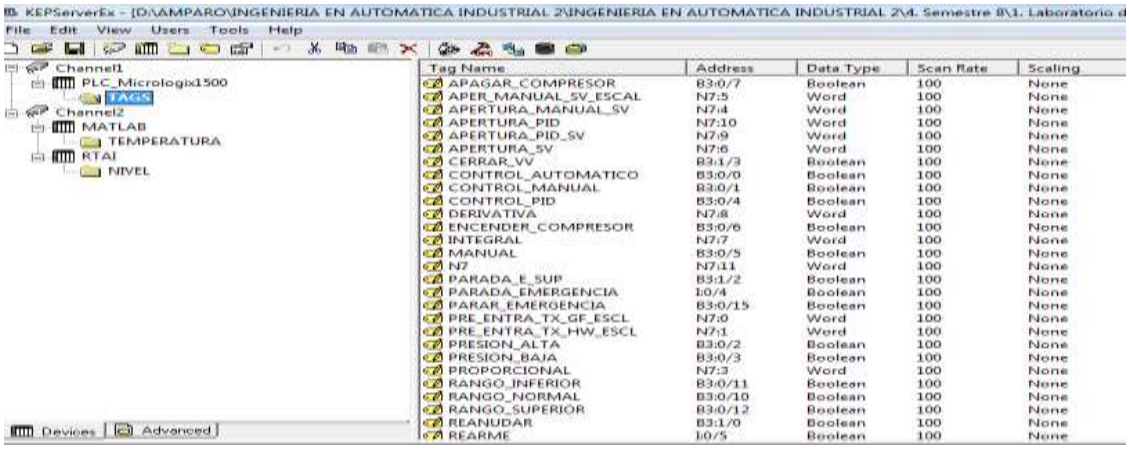
Fuente: propia

5.3.2. Procedimiento para la integración de las plataformas de control

Para lograr la integración entre Matlab, Rtai-lab y el PLC Micrologix 1500, primero se realizó la configuración de los objetos ítems en el servidor OPC DCOM (MSOD), luego se configuraron los módulos cliente OPC DCOM en cada plataforma, y finalmente se realizaron los supervisorios en el MM&SW. Cada uno de estos pasos se describe a continuación:

Configuración del servidor OPC DCOM Keepserver: inicialmente se configuraron los objetos ítems en el servidor OPC DCOM, según el anexo B (manual de usuario del módulo servidor OPC DCOM keepServerEX). La figura 5-6 presenta una pantalla con los objetos ítems configurados en el servidor KepeserverEX para el PLC micrologix 1500, Matlab y Rtai-Lab.

Imagen 5-1: Pantalla que presenta los objetos ítems configurados en el servidor OCP DCOM

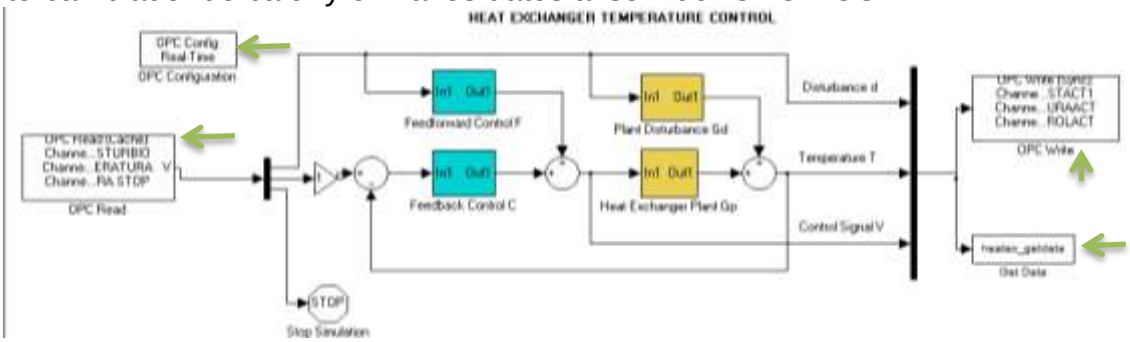


Fuente: propia

Integración de los módulos OPC con los esquemas de control

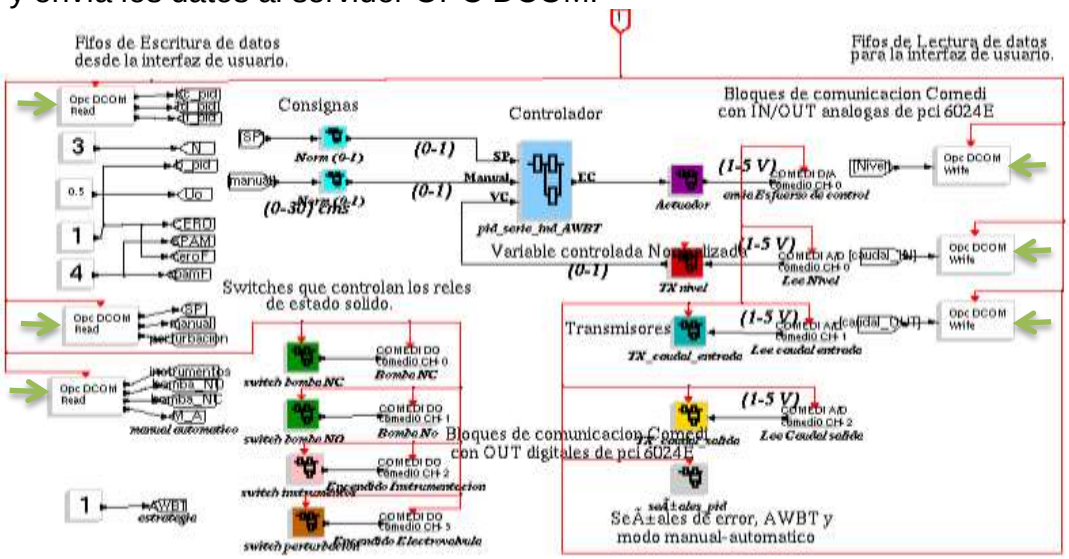
Luego de configurados los objetos ítems en el servidor KepServerEX se modificaron los diagramas de bloques en Scicos/Simulink para establecer la comunicación entre las plataformas y el servidor OPC DCOM. La imagen 5-2 presenta el diagrama de bloques del control de temperatura, y la imagen 5.3 presenta el diagrama que permite el control para la planta de nivel y la comunicación con el servidor KepServerEX (flechas verdes bloques OPC DCOM adicionados).

Imagen 5-2: Diagrama de bloques en Simulink que permite el control del intercambiador de calor y envía los datos al servidor OPC DCOM.



Fuente: propia

Imagen 5-3: Diagrama de bloques en Scicos que permite el control de la planta de nivel y envía los datos al servidor OPC DCOM.

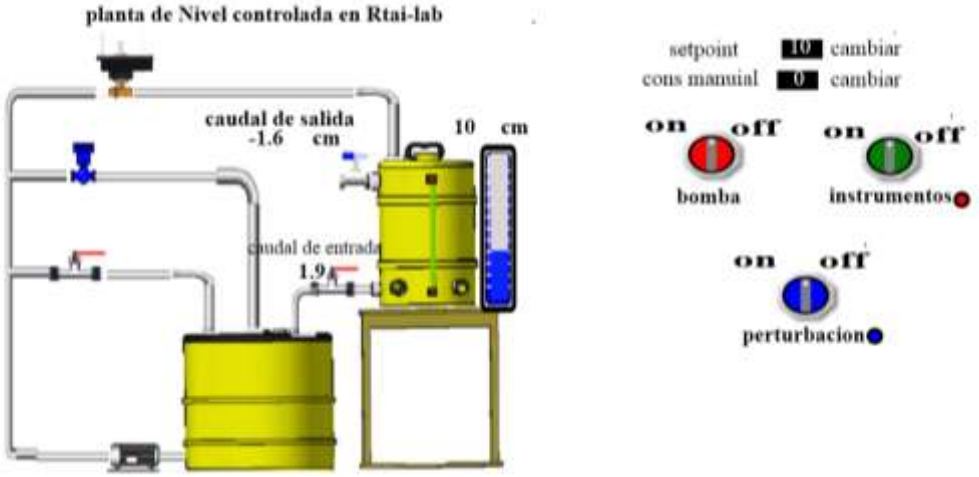


Fuente: propia

Diseño de los supervisorios en el MM&SW

Después de la comunicación entre las plataformas y el servidor KepServerEX, se diseñaron los supervisorios en el MM&SW según el anexo F (guía de instalación y manual de usuario del Módulo de monitoreo y supervisión web). Las imágenes 5-4, 5-5 y 5-6 presentan las pantalla de monitoreo para las plantas de nivel/presión e intercambiador de calor respectivamente.

Imagen 5-4: Supervisorio para la planta de nivel diseñado en el MM&SW



Fuente: propia

Imagen 5-5: Supervisorio para la planta de presión diseñado en el MM&SW

Fuente: propia

Imagen 5-6: Supervisorio para el intercambiador de calor, diseñado en el MM&SW

Fuente: propia

5.3.3. Resultados

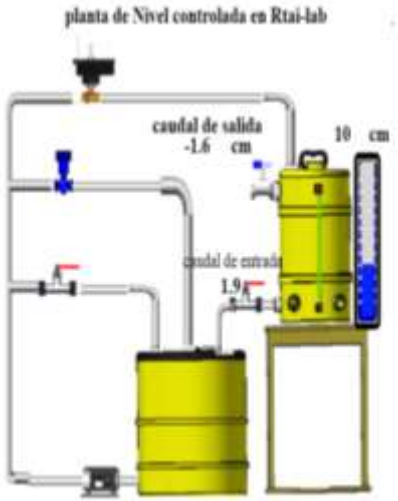
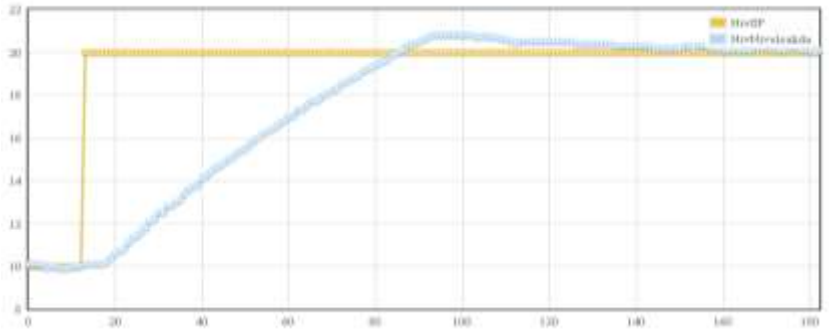
Para verificar la comunicación entre el MM&SW y las plantas de nivel, presión y temperatura se realizaron cambios en los valores de consigna, se aplicaron disturbios y se generaron alarmas. La tabla 5-2 presenta un resumen de las acciones realizadas y los resultados obtenidos los cuales se ven reflejados en las tendencias, alarmas y eventos.

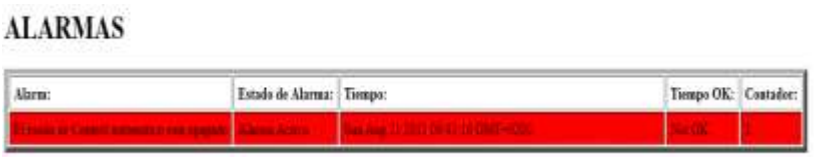
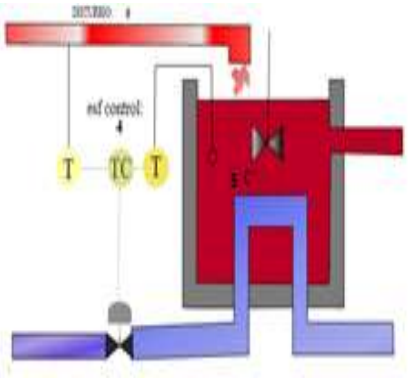
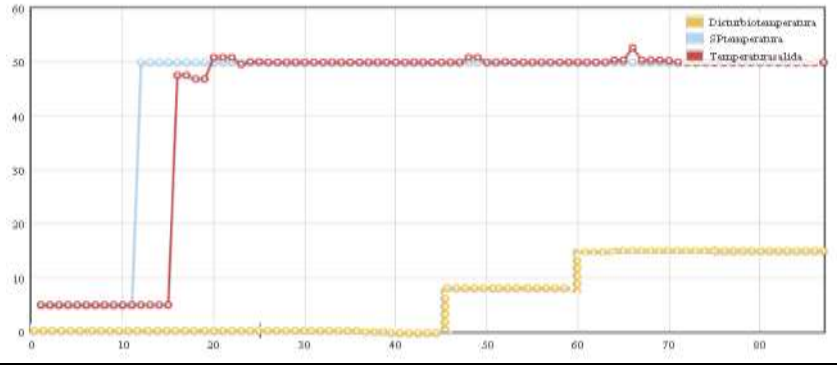
Con la realización de esta prueba se logró integrar a Matlab, Rtai-Lab y el PLC Micrologix 1500 mediante el estándar OPC DCOM. El tiempo de respuesta entre las plantas y el MM&SW fue de aproximadamente un segundo, teniendo como limitante la velocidad de la red.

Tabla 5 -2: Acciones realizadas en el MM&SW y resultados obtenidos en la prueba de integración mediante el estándar OPC DCOM

<table>
<tr><th>PLANTA</th><th>ACCIÓN REALIZADA</th><th>PROCESO</th><th>EVENTOS/ALARMAS/TENDENCIAS</th></tr>
<tr><td>Nivel</td><td>Se estableció inicialmente un valor en el Setpoint de nivel de 10 cm

Se cambió el Setpoint de nivel a 20 cm.

Se apagó la instrumentación de la planta.</td><td> 
</td><td> 
Eventos:
<table>
<tr><th>Evento #:</th><th>Fecha:</th><th>Evento:</th><th>Estado:</th></tr>
<tr><td>1313911977</td><td>Sun Aug 21 2011 09:32:59 GMT+0200</td><td>El nivel del tanque es mayor a 15 cm</td><td>1</td></tr>
</table>
ALARMAS
<table>
<tr><th>Alarm:</th><th>Estado de Alarma:</th><th>Tiempo:</th><th>Tiempo OK:</th><th>Contador:</th></tr>
<tr><td>El nivel del tanque esta apagado</td><td>Alarma Activa</td><td>Sun Aug 21 2011 09:40:04 GMT+0200</td><td>Not OK</td><td>1</td></tr>
</table></td></tr>
<tr><td>Presión</td><td>Se fijó la presión en 16 psi (control automatico).</td><td>  </td><td>Eventos:
<table>
<tr><th>Evento #:</th><th>Fecha:</th><th>Evento:</th><th>Estado:</th></tr>
<tr><td>1313912373</td><td>Sun Aug 21 2011 09:39:38 GMT+0200</td><td>La presion es alta</td><td>1</td></tr>
</table></td></tr>
</table>

	Se estableció la apertura de la electroválvula a 20%.		
Temperatura	Se realizó un cambio en el setpoint de 5 a 50 grados, y se aplicaron diferentes valores de disturbio.		

5.4. PRUEBA 2: INTEGRACIÓN MEDIANTE EL ESTÁNDAR OPC XML

Como se describe en el numeral 5.1, la segunda prueba consiste en intercambiar información entre Matlab, Rtai-Lab y el PLC Micrologix 1500 haciendo uso del MSOX. La información detallada de los pasos para realizar esta prueba se organizó en una guía de prácticas denominada: guía para la integración de Matlab, Rtai-Lab y el PLC *Micrologix* 1500 mediante el estándar OPC XML disponible en el anexo I.

5.4.1. Requerimientos hardware y software

La tabla 5-3 presenta los requerimientos hardware y software necesarios para la realización de la práctica de integración de Matlab, Rtai-Lab y el PLC *Micrologix* 1500 mediante el estándar OPC XML.

Tabla 5 -3: 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**	Controla la planta de nivel y la comunicación remota con el servidor OPC XML.	Elementos de la planta de nivel. Un PC con conexión a una red LAN.	Rtai-Lab	Diagrama de bloques para el control de nivel.
					MCOXR
	Control de la planta de temperatura en Matlab	Controla la planta de temperatura y la comunicación remota con el servidor OPC XML.	Un PC con conexión a una red LAN.	Matlab	*MCOXM*
					Diagrama de bloques para el control de temperatura
Servidor OPC XML		Iintegra la información de las plataformas de control.	Un PC con conexión a una red LAN	MSOX	
Módulo de monitoreo y supervisión web		Permite 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	

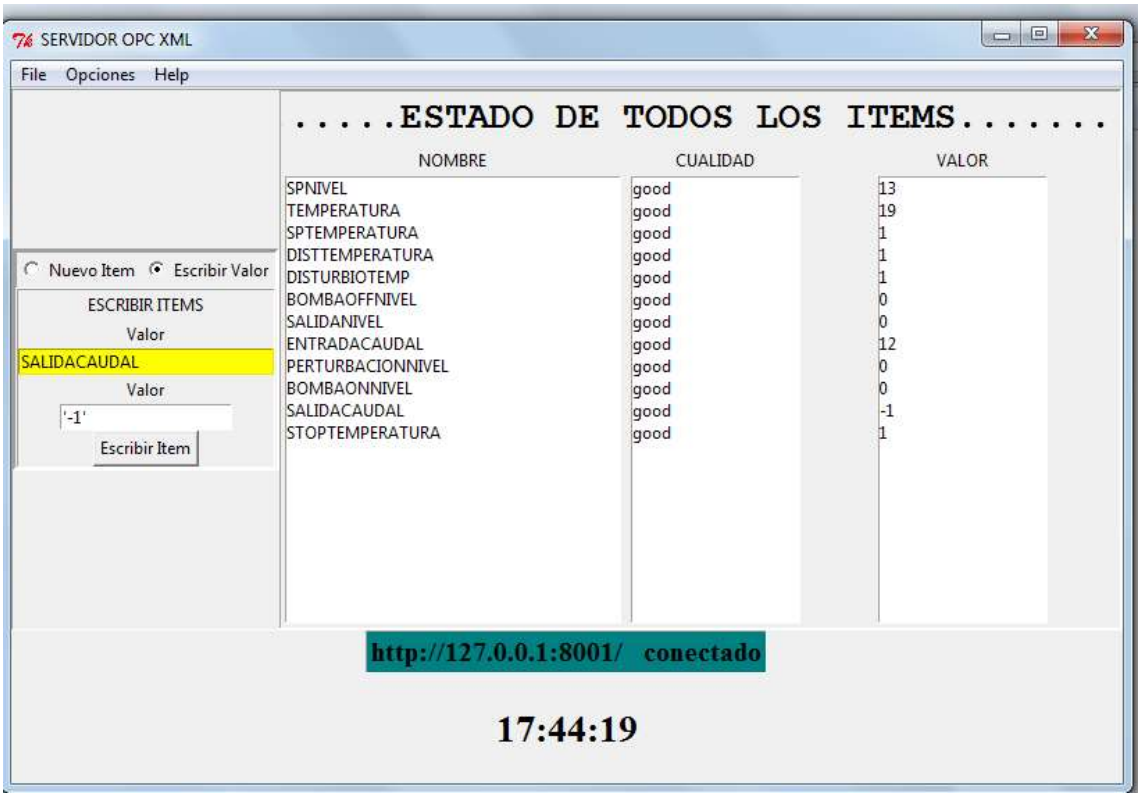
Fuente: propia

5.4.2. Procedimiento para la integración de las plataformas de control

Para lograr la integración primero se realizó la configuración de los objetos ítems en el servidor OPC XML (MSOX), luego se configuraron los módulos cliente OPC XML en cada plataforma, y finalmente se realizaron los supervisorios en el MM&SW. Cada uno de estos pasos se describe a continuación:

Configuración del MSOX: Se configuraron los objetos ítems necesarios para la comunicación entre Matlab y Rtail-Lab en el MSOX, según el anexo D (manual de usuario del MSOX), ver imagen 5-7.

Imagen 5-7: Pantalla que presenta los objetos ítems configurados en el servidor OCP XML.

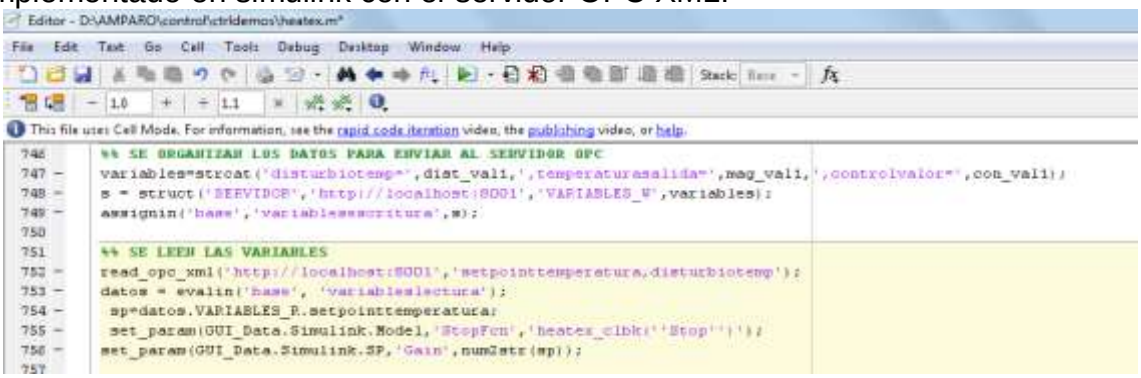


Fuente: propia

Integración de los módulos OPC con los esquemas de control

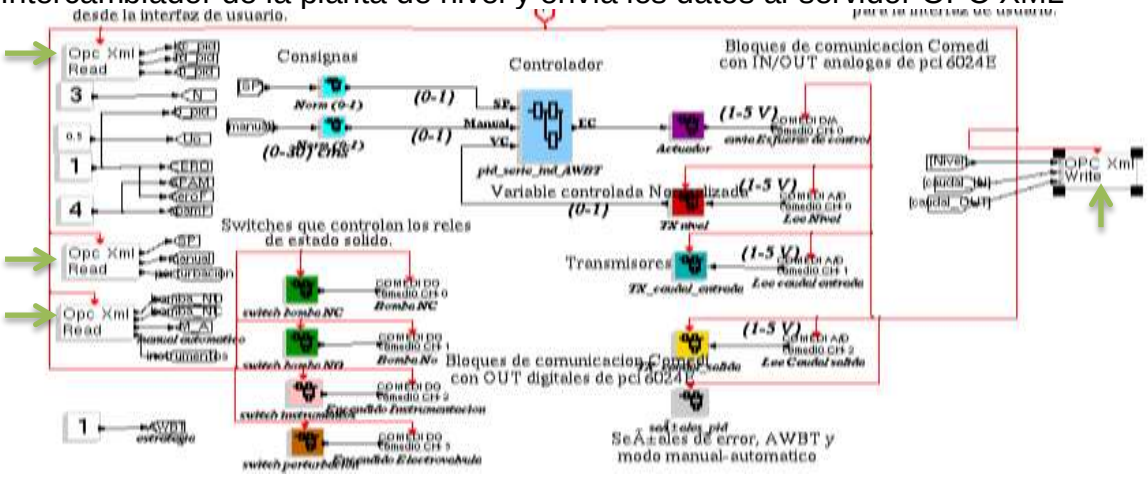
Luego de configurados los objetos ítems en el servidor OPC XML, se modificó el programa en Matlab, ver imagen 5-8 y el diagrama de bloques en Scicos, ver imagen 5-9 para establecer la comunicación entre las plataformas y el servidor OPC XML.

Imagen 5-8: Código en Matlab que comunica al control de temperatura implementado en simulink con el servidor OPC XML.



Fuente: propia

Imagen 5-9: Diagrama de bloques en Scicos que permite el control del intercambiador de la planta de nivel y envía los datos al servidor OPC XML



Fuente: propia

<u>Diseño de los supervisorios en el MM&SW</u>

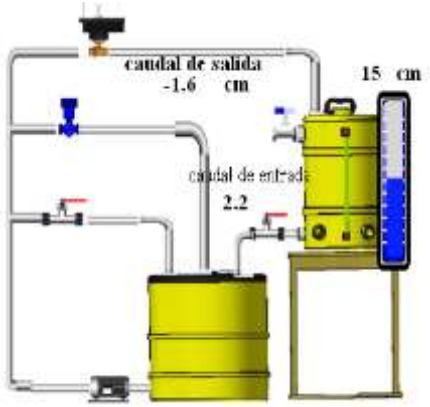
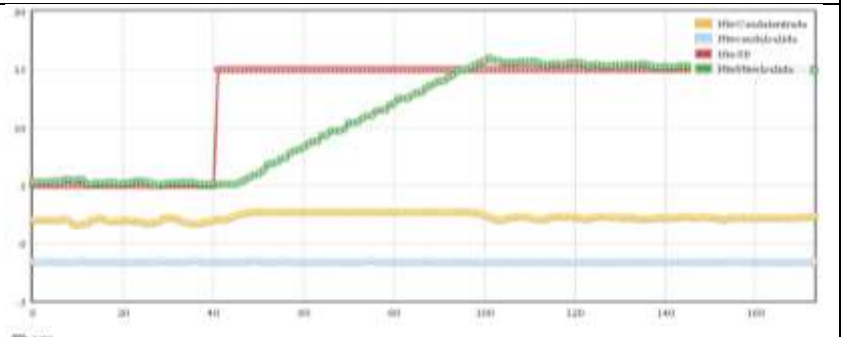
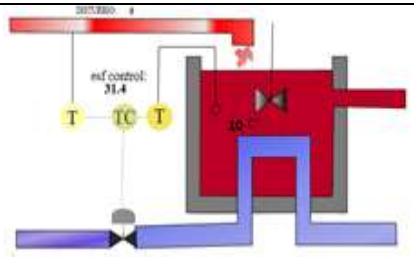
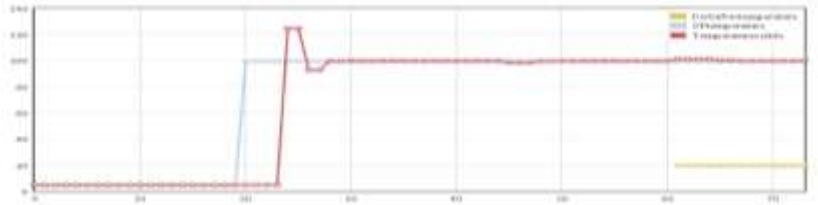
Después de la comunicación entre las plataformas y el MSOX, se diseñaron los supervisorios en el MM&SW, para los cuales se utilizó los mismos esquemas mostrados en las imágenes 5-4 y 5-6.

5.4.3. Resultados

Con la realización de esta prueba se logró integrar a Matlab, Rtai-Lab y el PLC *Micrologix* 1500 mediante el estándar OPC XML. La tabla 5-4 presenta un resumen de las acciones realizadas y los resultados obtenidos los cuales se ven reflejados en las tendencias, alarmas y eventos.

.

Tabla 5 -4: Acciones realizadas en el MM&SW y resultados obtenidos mediante en la prueba de integración mediante el estándar OPC XML

PLANTA	ACCIÓN REALIZADA	PROCESO	EVENTOS /ALARMAS TENDENCIAS
Nivel	Se realizó un cambio en el un valor en el *Setpoint* de nivel de 5 a 15 cm.		 Show: **Eventos:** Evento #: | Fecha: | Evento: | Estado: 1313912207 | Sun Aug 21 2011 02:36:49 GMT-0500 | El tanque esta lleno | 1 **ALARMAS** Alarm: | Estado de Alarma: | Tiempo: | Tiempo OK: | Contador:
	Se apagó la instrumentación de la planta de nivel		
Temperatura	Se realizó un cambio en el setpoint de 5 a 50 grados, y se generó un disturbio de 15 c.		

5.5. VALIDACION DE LOS MODULOS DE LA ARQUITECTURA

A partir de las pruebas descritas en los numerales 5-3 (prueba1: integración mediante el estándar OPC DCOM) y 5-4 (prueba2: integración mediante el estándar OPC XML) se realizó la validación de cada uno de los módulos diseñados en la arquitectura para la integración (MSOX, MCODR, MCOXM, MCOXR, MM&SW).

Para validar los módulos, se retomaron los requerimientos funcionales y no funcionales de cada módulo de la arquitectura de integración descritos en el numeral 2-4-4 del capítulo 2, y en cada práctica se observó el cumplimento de cada uno de los requerimientos. La validación de los requerimientos funcionales se presenta en las tablas G-1, G-2, G-3, G-4 y G-5, del anexo G (validación de los módulos de la arquitectura), en estas tablas se especifica los sub-módulos, el identificador de cada requerimiento, se realiza la descripción de los requerimientos, y se define si se cumple o no el requerimiento.

Las tablas G-6 y G-7 del anexo G (validación de los módulos de la arquitectura) presentan los resultados obtenidos en la validación de los requerimientos no funcionales, en ellas se presenta el identificador y la descripción del requerimiento, y se detalla la validación.

5.6. RESUMEN

Con la realización de estas pruebas se logró la integración de las dos plataformas académicas de control Matlab y Rtai-Lab, y adicionalmente el PLC Micrologix 1500. Se validó la arquitectura propuesta con las plantas de nivel y presión existentes en el laboratorio de control de procesos del PIAI, y gracias a estas pruebas se verifico el funcionamiento y la validación de los módulos MSODX.

6. CONCLUSIONES Y RECOMENDACIONES

El objetivo general de este trabajo es “Diseñar e implementar una arquitectura de integración, monitoreo y supervisión remota para plataformas de control basadas en PC mediante OPC”, durante el desarrollo del mismo se realizó el diseño y la implementación de un sistema OPC basado en *software* libre que en primer lugar no está restringida a un único tipo de plataforma de control basado en PC, está concebida para integrar sistemas de control operando en Rtai-Lab, Matlab y adicionalmente en PLCs.

Durante el desarrollo del trabajo se concentró en el estándar de comunicaciones OPC y se pudo determinar la importancia de este tipo de alternativas de comunicación que representan una buena solución de integración para las llamadas “islas de automatización” que son muy comunes en la industria. En el caso de la aplicación desarrollada para las plantas de control del laboratorio de control de procesos industriales se pudo constatar esto, porque gracias a ello se pudo establecer la comunicación entre Matlab, Rtai-Lab y el PLC.

El módulo de monitoreo y supervisión web implementado, permite el procesamiento de datos, diseño de pantallas HMI, representación de señales de alarma, visualización gráfica dinámica, y un módulo gestor de contenido, que posibilita la actualización, mantenimiento y ampliación de la web con la colaboración de múltiples usuarios, se recomienda adicionarle otros sub-módulos funcionales como: manejo de históricos, manejo de estadísticas, entre otros.

Los estándares OPC XML/DCOM son muy flexibles y permiten el tratamiento de datos simples, cadenas, arreglos, sin embargo OPC UA es un estándar OPC completo; por lo tanto sería interesante plantear como trabajo futuro el desarrollo de un sistema que involucre la tecnología OPC UA.

Para un profesional con el perfil de un ingeniero en Automática Industrial, es importante incluir dentro de su formación académica la información y la práctica en sistemas de tipo *open source* a nivel industrial, ya que constituyen una opción interesante en el desarrollo de las empresas de producción en nuestro entorno económico, por lo tanto se recomienda para tesis futuras la exploración del servidor OPC DCOM *LightOPC*, expuesto en el numeral 3.2.

BIBLIOGRAFÍA

1. **Mathworks.** Matlab. [En línea] [Citado el: 15 de 01 de 2011.] http://www.mathworks.com/help/techdoc/ref/guide.html.

2. **National Instruments.** NI LabVIEW. [En línea] [Citado el: 08 de 12 de 2010.] http://www.ni.com/labview/.

3. **RTAI.** RTAI - the RealTime Application Interface for Linux from DIAPM. [En línea] [Citado el: 08 de 12 de 2010.] https://www.rtai.org/.

4. **OPC Foundation.** What is the OPC Foundation? [En línea] [Citado el: 16 de 02 de 2011.] http://www.opcfoundation.org/Default.aspx/01_about/01_history.asp?MID=AboutOPC.

5. —. What is OPC? [En línea] [Citado el: 16 de 02 de 2011.] http://www.opcfoundation.org/Default.aspx/01_about/01_whatis.asp?MID=AboutOPC.

6. **Echeverri, S.** *Tecnología opc para la adquisición de datos de variables industriales.* s.l. : Technical report,Universidad de San Buenaventura, 2005.

7. **Automata.** Control Basado En PC. [En línea] [Citado el: 12 de 10 de 2010.] http://automata.cps.unizar.es/Historia/Webs/control_basado_en_pc.htm.

8. *Real-Time Linux Target: A MATLAB-based graphical control environment.* **Rockwell, Geoffrey, y J. Bradley.** s.l. : IEEE Conf. on Control Applications, 1998.

9. **Domínguez, T.** *Control automático del proceso productivo.* s.l. : Tecnica Industrial, 2005.

10. **SIEMENS.** Control basado en PC abierto, flexible y fiable. [En línea] [Citado el: 06 de 11 de 2011.] http://www.siemens.com.br/templates/get_download2.aspx?id=1571&type=FILES.

11. **Ogata, K.** *Ingeniería de Control Moderna.* 2007.

12. **Webelectronica.** Sistemas de control basados en PC. [En línea] [Citado el: 12 de 02 de 2011.] http://www.webelectronica.com.ar/news15/nota09.htm.

13. **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.

14. **Beckhoff.** Robust industrial design PCs with highest performance components. [En línea] [Citado el: 15 de 12 de 2010.] http://www.beckhoff.com.

15. *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.

16. *Real-time control using Matlab Simulink.* **F.Teng.** Nashville, Tennessee, USA : In IEEE International Conference on Systems, Man, and Cybernetics, 2000. 42697-2702.

17. **J. F. Florez, E. Diaz, y Y. Cabezas.** Simulacion y control en cascada de una planta pomtm en tiempo real con RTAI-Lab". 2008, págs. 852–859.

18. **Resquín,F.** RTAI – Real Time Application Interface. [En línea] [Citado el: 12 de 11 de 2010.] www.jeuazarru.com/docs/RTAI.pdf.

19. *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, págs. 5–6.

20. **opcfoundation.** OPC Unified Architecture. [En línea] [Citado el: 16 de 02 de 2011.] http://www.opcfoundation.org/Default.aspx/01_about/UA.asp?MID=AboutOPC.

21. **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.

22. *technical information bulletin 04-1 - Supervisory Control and Data Acquisition (SCADA) Systems.* **COMMUNICATION TECHNOLOGIES.** 2004, pág. 4.

23. **STELLER, L, QUESADA, M y 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.

24. **Mendiburo.H.** Sistemas Scada. [En línea] [Citado el: 23 de 02 de 2011.] http://www.galeon.com/hamd/pdf/scada.pdf.

25. **ControlGlobal.com.** Integrators must differentiate between a plant-based OPC mentality and SCADA applications when attempting to apply industrial standards to SCADA-based implementations. [En línea] [Citado el: 17 de 02 de 2011.] http://www.controlglobal.com/articles/2005/441.html.

26. **Systems.NET, OPC.** Open Automation Software. *http://www.opcsystems.com.* [En línea] [Citado el: 15 de 12 de 2010.]

27. SCADA/HMI Products . [En línea] Sielco Sistemi . [Citado el: 12 de 11 de 2010.] http://www.sielcosistemi.com/.

28. The HMI & SCADA Revolution. [En línea] ATVISE. [Citado el: 12 de 11 de 2010.] http://www.atvise.com/.

29. *RSVIEW soluciones de visualización que le ayudan a alcanzar el éxito.* **Automation, Rockwell.** s.l. : Publicación RSVIEW-BR001A-ES-P, 2005.

30. **Openscada.** The OpenSource SCADA System. [En línea] [Citado el: 17 de 11 de 2010.] http://openscada.org.

31. —. The OpenSource SCADA System. [En línea] [Citado el: 12 de 11 de 2010.] http://openscada.org/.

32. SCADA Systems. [En línea] [Citado el: 12 de 11 de 2010.] http://www.scadasystems.net/.

33. **MathWorks.** OPC Toolbox. [En línea] [Citado el: 16 de 02 de 2011.] http://www.mathworks.com/help/toolbox/opc/.

34. **KEPWARE.** KEPServerEX OPC and Communications Server Features . [En línea] [Citado el: 14 de 12 de 2011.] http://www.kepware.com/.

35. **Advosol.** Advosol Advanced OPC Solutions. [En línea] [Citado el: 15 de 03 de 2011.] http://www.advosol.com/.

36. **beijerelectronics.** OPC Server - Gets you connected. [En línea] [Citado el: 15 de 03 de 2001.] http://www.beijerelectronics.com.

37. **ICONICS.** OPC Connectivity. [En línea] [Citado el: 15 de 3 de 2011.] http://www.iconics.com/.

38. **MatrikonOpc.** OpcServers. [En línea] [Citado el: 07 de 01 de 2011.] http://www.matrikonopc.com/.

39. **LightOPC.** The Free OPC Server Toolkit. [En línea] [Citado el: 12 de 11 de 2010.] http://www.ipi.ac.ru/lab43/lOPC-en.html.

40. *Requirements management and requirements engineering: You can't have one without the other.* **Nancy, R.** 5, s.l. : Cutter IT Journal, 2000, Vol. 13.

41. **SAWYER, Peter y Kontoya, Gerald.** Software requirements. *Software Engineering Book Of Knowledge.* [En línea] http://www.swebok.org.

42. **Marcia C. F. Carvalho & Zair Abdelouahab.** Um Método para Elicitação e Modemagem de Requisitos Baseado em Objetivos. [En línea] 08 de 10 de 2010. http://www.inf.puc-rio.br/wer01/Eli-Req-5.pdf.

43. *Requirements management and requirements engineering: You can't have one without the other.* **Nancy, R.** 5, 2000, Vol. 13.

44. **dOPC Explorer.** OPC Software Products. [En línea] [Citado el: 15 de 03 de 2011.] http://www.dopc.kassl.de/.

45. **Opcconnect.** Free Stuff - OPC Clients. [En línea] [Citado el: 11 de 03 de 2011.] http://www.opcconnect.com/freecli.php.

46. *Diagramas de Casos de Uso.* **Cáceres., J.** Universidad de Alcalá. Departamento de Ciencias de la Computación. Alcalá de Henares, España : s.n., 2008.

47. **COCKBURN, Alistair.** Basic Use Case Template. [En línea] http://members.aol.com/acockburn>.

48. **Clickear.com.** Los casos de uso. [En línea] [Citado el: 12 de 11 de 2011.] http://www.clikear.com/manuales/uml/faseplanificacion.aspx.

49. **The Python Community.** PyOPC. [En línea] [Citado el: 12 de 02 de 2011.] http://pyopc.sourceforge.net/.

50. —. Python Programming Language – Official Website. [En línea] [Citado el: 12 de 02 de 2011.] http://www.python.org/ .

51. **W3C.** Simple Object Access Protocol (SOAP). [En línea] [Citado el: 15 de 02 de 2011.] http://www.w3.org/TR/soap/.

52. **Twisted.** Twisted Matriks Lab. [En línea] [Citado el: 18 de 12 de 2010.] http://twistedmatrix.com/trac/.

53. **SOURCEFORGE.NET.** Python Web Services. [En línea] [Citado el: 15 de 3 de 2011.] http://pywebsvcs.sourceforge.net/.

54. **Python.** TkInter. [En línea] [Citado el: 3 de 1 de 2010.] http://wiki.python.org/moin/TkInter.

55. **python.** wxPython. [En línea] [Citado el: 24 de 10 de 2010.] http://www.wxpython.org/.

56. **John W. Shipman.** Tkinter 8.4 reference: a GUI for Python. [En línea] http://infohost.nmt.edu/tcc/help/pubs/tkinter/.

57. OPC Toolbox . [En línea] Matlab. [Citado el: 02 de 01 de 2011.] http://www.mathworks.com/products/opc/.

58. **w3c.** Guía breve de tecnologías XML. [En línea] [Citado el: 10 de 02 de 2011.] http://www.w3c.es/divulgacion/guiasbreves/tecnologiasxml.

59. Users guide. [En línea] Matworks. [Citado el: 11 de 12 de 2010.] http://www.mathworks.com/help/techdoc/ref/guide.html.

60. **Scilab.** Scilab The Free Software for Numerical Computation. [En línea] [Citado el: 16 de 02 de 2011.] http://www.scilab.org/.

61. **SOURCEFORGE.** About OpenOPC. [En línea] [Citado el: 03 de 01 de 2010.] http://openopc.sourceforge.net/.

62. Introducción a los CMS. [En línea] 2007. [Citado el: 11 de 12 de 2010.] http://www.alterpime.net/documentos/Introduccion_a_los_CMS.pdf.

63. **W3C SVG Working Group.** Scalable Vector Graphics (SVG). [En línea] [Citado el: 01 de 10 de 2011.] http://www.w3.org/Graphics/SVG/.

64. **JSON.** Introducing JSON. [En línea] [Citado el: 5 de 1 de 2011.] http://www.json.org.

65. MYSQL.COM. [En línea] [Citado el: 12 de 11 de 2010.] www.mysql.com.

66. PHP. [En línea] [Citado el: 11 de 12 de 2010.] http://www.php.net/.

67. APACHE. [En línea] [Citado el: 11 de 12 de 2010.] www.apache.org/.

68. **DRUPAL DEVELOPPEUR.** WampServer. [En línea] [Citado el: 04 de 10 de 2010.] http://www.wampserver.com/en/.

69. **Google Project Hosting .** svg-edit. [En línea] [Citado el: 5 de 1 de 2010.] http://code.google.com/p/svg-edit/.

70. **Whizzywig.** Web based rich text editor for free. [En línea] http://unverse.net/Whizzywig-web-based-rich-text-editor.

71. **sourceforge.** What's new in HMIServer. [En línea] http://mblogic.sourceforge.net/whatsnew/whatsnewhmiserver.html.

72. **Google Project Hosting.** flot,Attractive Javascript plotting for jQuery. [En línea] [Citado el: 10 de 01 de 2011.] http://code.google.com/p/flot/.

73. *Comparing Open Source CMSes: Joomla, Drupal and Plone.* **Bonfield, B and Quinn,S.** s.l. : Idealware, 2007.

74. **Joomla.** Bienvenido a Joomla. [En línea] [Citado el: 12 de 01 de 2010.] http://ayuda.joomlaspanish.org/ayuda-joomla/.

75. **Deiretsbacher, K.-H. y E.al.** *Using opc via dcom with windows xp service pack 2.* s.l. : Technical report, OPC Fundation , 2005.

76. **H.Himmelbauer.** PyOPC A Python Framework for the OPC XML-DA 1.0 Standard,Klosterneuburg. [En línea] [Citado el: 11 de 12 de 2011.] pyopc.sourceforge.net.