SISTEMA DE GESTIÓN DE PLCs MULTIPLATAFORMA

Fernando Alejandro Campos Peña
German Mauricio Coral Huertas

Director:
Ing. Oscar Amaury Rojas Alvarado

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICAIONES
DEPARTAMENTO DE ELECTRÓNICA INSTRUMENTAL Y CONTROL
Popayán
2005

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

1. INTRODUCCIÓN

Algunos años atrás, la utilización de plataformas diferentes a Windows no era tenida en cuenta debido al gran soporte en cuanto a servicio y calidad de las herramientas desarrolladas sobre esta plataforma. Pero en la actualidad, otros sistemas diferentes a Windows han llegado a adquirir importancia en entornos como los sistemas telemáticos, y recientemente se ha extendido su influencia al sector industrial.

Después de llevar a cabo ciertos estudios de mercadeo, las empresas desarrolladoras de herramientas software para sistemas SCADA se dieron cuenta que sistemas operativos como UNIX y aún Linux habían tenido un gran auge en los últimos años y que su desarrollo y penetración continuaban en aumento. Teniendo en cuenta estos resultados ha surgido la preocupación de brindar a todos los usuarios sin importar el sistema operativo en el que se soporten, la posibilidad de compartir la información y monitorear los procesos sin ninguna desventaja o disminución en la calidad del servicio comparado con el prestado por plataformas Windows.

La Universidad del Cauca y el Grupo de Investigación en Automática Industrial, no siendo ajena a la evolución y los cambios mundiales en el sector industrial, ha querido desarrollar una herramienta que permita validar la portabilidad de un sistema de control en diferentes plataformas. Esta es la verdadera razón del proyecto "Sistema de Gestión de PLCs multiplataforma" que se presenta a continuación.

En este documento final el lector podrá encontrar información relevante sobre Redes Industriales y su evolución en los últimos años, también las posibles herramientas para desarrollar un sistema con las características explicadas anteriormente. El lector, podrá darse cuenta como se desarrolló la idea y ver una descripción detallada de su análisis, diseño e implementación.

En el capitulo "*La Industria y OPC*", se hace un análisis de la situación actual de los sistemas basados en PLCs y las empresas, tanto a nivel nacional, como internacional, además expone los precedentes que conllevaron a pensar en el desarrollo del proyecto. En este capitulo, básicamente se manejan dos arquitecturas, las cuales son el núcleo del trabajo realizado en este proyecto:

- ***Arquitectura 1:*** Cliente XML-DA desarrollado en Java conectado a un servidor OPC-DA por medio de un puente DA/XML-DA:

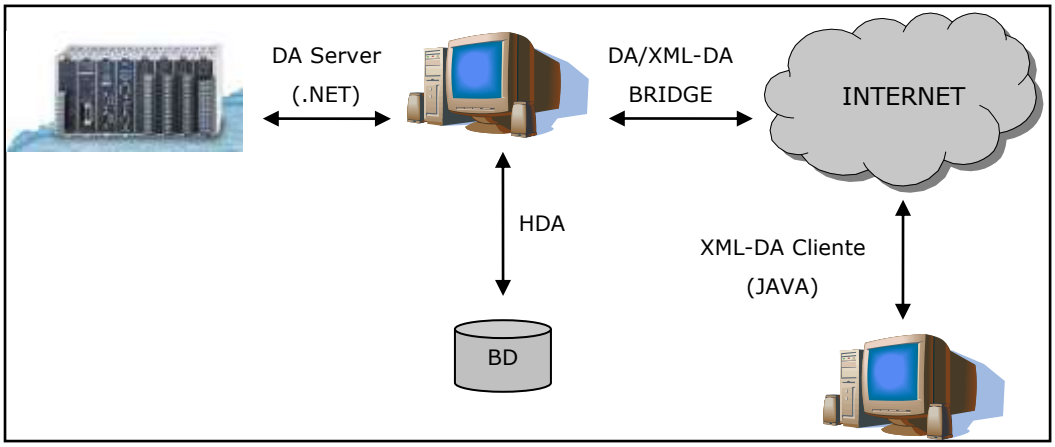

Figura 1.1. Arquitectura 1

En esta arquitectura se establece una conexión entre el PC servidor y el PLC a través de un servidor de Acceso a Datos (DA), adquirido de cualquier empresa desarrolladora de este tipo de servidores y cuya distribución en algunos casos es gratuita, estos servidores tienen como característica principal el hecho de estar desarrollados generalmente bajo el concepto de .NET.

El almacenamiento de los datos se realizará mediante la especificación de Acceso a Datos Históricos (HDA), por medio de la cual se establece la conexión y la comunicación entre el PC Servidor y la Base de Datos.

Para poder realizar una conexión de manera remota a través de Internet se cuenta con un puente o traductor entre las aplicaciones DA y XML-DA, el cual puede ser desarrollado en Java y puede publicar los datos en Internet para brindar acceso remoto

desde cualquier lugar por medio de un cliente basado en la especificación XML-DA y realizado también bajo la plataforma de desarrollo Java.

- ***Arquitectura 2:*** Utilizando un servidor XML-DA desarrollado en un lenguaje diferente a Java:

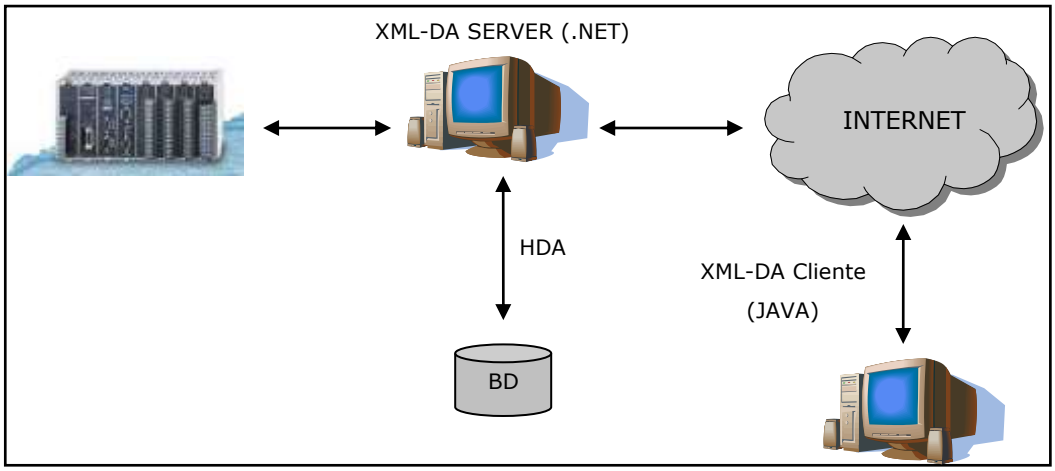

Figura 1.2. Arquitectura 2

En la arquitectura de la figura anterior, se establece una conexión entre el PC servidor y el PLC a través de un servidor de Acceso a Datos basado en XML (XML-DA), adquirido de cualquier empresa desarrolladora de este tipo de servidores y cuya distribución en algunos casos es gratuita, estos servidores tienen como característica principal el hecho de estar desarrollados generalmente bajo el concepto de .NET.

Una vez se obtienen los datos del PLC en el servidor, este se encarga de publicar los mismos en Internet para acceso remoto desde cualquier parte; haciendo uso de un cliente basado en la especificación XML-DA realizado en la plataforma de desarrollo JAVA.

El almacenamiento de los datos se realiza mediante la especificación de Acceso a Datos Históricos (HDA), por medio de la cual se establece la conexión y la comunicación entre el PC Servidor y la Base de Datos.

El capitulo "*OPC Multiplataforma*" describe de forma teórica los principales aspectos tenidos en cuenta en el desarrollo del proyecto, las ventajas y desventajas a la hora de utilizarlos.

Además, describe la forma en que se adaptaron las herramientas software para poder cumplir con las especificaciones del estándar OPC, su arquitectura y forma de actuar en el sistema, además, las ventajas y desventajas de haber implementado dicha arquitectura.

Este capitulo analiza también de que forma se pueden aprovechar los beneficios de los sistemas operativos bajo licencias de carácter libre y su interacción con el proyecto; relacionando la arquitectura manejada, las comunicaciones, los beneficios y las desventajas que consigo trae.

El capitulo "*Diseño Detallado del Sistema*", hace una especificación más precisa de los diferentes subsistemas, a través de los diagramas de casos de uso, diagramas de secuencia, modelos computacionales, modelos de información, diagramas de clases y las tecnologías utilizadas para su desarrollo.

En el capitulo "*Validación*", se describen los diferentes medios utilizados para verificar el buen funcionamiento del sistema y la respuesta obtenida de acuerdo al plan de pruebas establecido para el sistema.

Al final del trabajo se muestran las conclusiones obtenidas y se hace una serie de sugerencias para continuar el trabajo bajo una arquitectura similar y mejorar el sistema en muchos aspectos, especialmente en lo relacionado con su desempeño en redes de gran tamaño.

2. LA INDUSTRIA Y LOS PLCs

2.1 ESTRUCTURA DE LAS REDES INDUSTRIALES

En la empresa coexisten una serie de equipos y dispositivos dedicados al control de una máquina o una parte cerrada de un proceso. Entre estos dispositivos están los controladores lógicos programables (PLCs), computadores de diseño y gestión, sensores, actuadores, etc.

El desarrollo de las redes industriales ha establecido una forma de unir todos estos dispositivos, aumentando el rendimiento y proporcionando nuevas posibilidades.

Las ventajas que se pueden lograr con una red industrial son, entre otras, las siguientes:

- Visualización y supervisión de todo el proceso productivo.
- Obtención de datos del proceso de una manera más rápida o casi instantánea.
- Mejora del rendimiento general de todo el proceso.
- Posibilidad de intercambio de datos entre sectores del proceso y entre departamentos.
- Programación a distancia, suprimiendo la necesidad de permanecer en las instalaciones de la fábrica, para poder hacerlo.

Las ventajas son evidentes, pero esto involucra un determinado costo, el cual debe ser evaluado y puesto a consideración para determinar si la inversión es rentable o innecesaria.

2.2 NIVELES EN UNA RED INDUSTRIAL

En una red industrial coexisten equipos y dispositivos de todo tipo, los cuales generalmente se agrupan jerárquicamente para establecer conexiones de una manera adecuada entre cada área. De esta forma se definen cuatro niveles dentro de una red industrial:

- ***Nivel de Gestión:*** Es el nivel más elevado y se encarga de integrar los niveles siguientes en una estructura de fábrica, e incluso de múltiples factorías. Las máquinas aquí conectadas suelen ser estaciones de trabajo que hacen de puente entre el proceso productivo y el área de gestión, en el cual se supervisan las ventas, reservas, etc. Se emplea una red de tipo LAN (Local Area Network) o WAN (Wide Area Network).

- ***Nivel de Control:*** Se encarga de enlazar y dirigir las distintas zonas de trabajo. A este nivel se sitúan los autómatas de gama alta y los ordenadores dedicados a diseño, control de calidad, programación, etc. Se suele emplear una red de tipo LAN.

- ***Nivel de Campo y Proceso:*** Se encarga de la integración de pequeños dispositivos de campo (PLC compactos, multiplexores de E/S, controladores PID, etc.) dentro de sub-redes o "islas". En el nivel más alto de estas redes se pueden encontrar uno o varios PLC modulares, actuando como maestros de la red o maestros flotantes. En este nivel se emplean redes de bus de campo (field bus).

- ***Nivel de E/S:*** Es el nivel más próximo al proceso. Aquí están los sensores y actuadores, encargados de manejar el proceso productivo y tomar las medidas necesarias para la correcta automatización y supervisión.

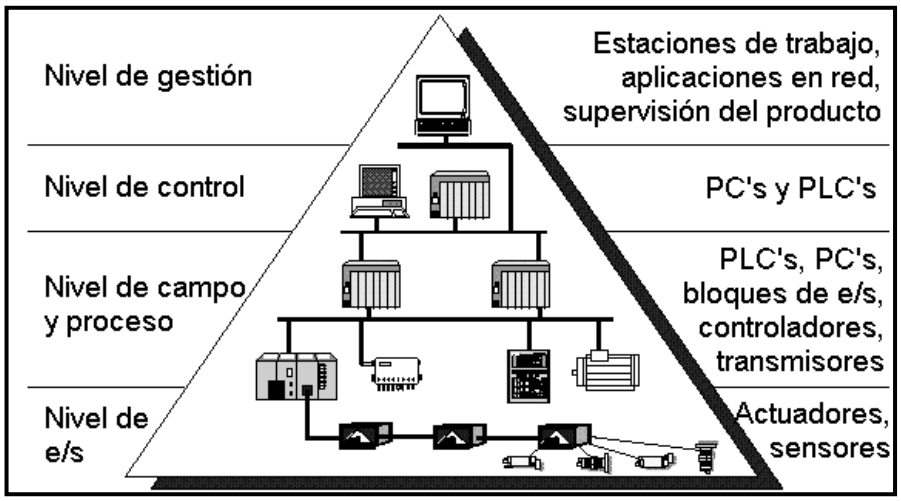

Figura 2.1. Niveles en una Red Industrial

2.3 TIPOS DE REDES INDUSTRIALES Y APLICACIONES

Las comunicaciones en las redes industriales deben poseer unas características particulares para responder a las necesidades de intercomunicación en tiempo real que se deben producir y ser capaces de resistir un ambiente hostil donde existe gran cantidad de ruido electromagnético y condiciones ambientales extremas. En el uso de comunicaciones industriales se pueden identificar dos áreas principales, una comunicación a nivel de campo, y una comunicación hacia el SCADA. En ambos casos la transmisión de datos se realiza en tiempo real, o por lo menos con una demora que no es significativa respecto de los tiempos del proceso, pudiendo ser crítico para el nivel de campo.

Según el entorno donde van a ser instaladas, en un ámbito industrial existen varios tipos de redes:

- [1]***Red de Fábrica (Factoría):*** Para redes de oficina, contabilidad y administración, ventas, gestión de pedidos, almacén, etc. El volumen de información intercambiada

[1] Sistemas Industriales Distribuidos. Universidad de Valencia.

es muy alto, y los tiempos de respuesta no son críticos (sistemas de tiempo real blandos).

- ***Red de Planta:*** Para interconectar módulos y células de fabricación entre sí y con departamentos como diseño o planificación. Suele emplearse para el enlace entre las funciones de ingeniería y planificación con las de control de producción en planta y secuenciamiento de operaciones. Estas redes deben manejar mensajes de cualquier tamaño, gestionar eficazmente errores de transmisión (detectar y corregir), cubrir áreas extensas (puede llegar a varios kilómetros), gestionar mensajes con prioridades (gestión de emergencias frente a transferencia de ficheros CAD/CAM), y disponer de amplio ancho de banda para admitir datos de otras subredes como voz, vídeo, etc.

- ***Red de Célula:*** Para interconectar dispositivos de fabricación que operan en modo secuencial como Robots, Máquinas de control numérico (CNC), Autómatas programables (PLC), Vehículos de guiado automático (AGV). Las características deseables en estas redes son: Gestionar mensajes cortos eficientemente, capacidad de manejar tráfico de eventos discretos, mecanismos de control de error (detectar y corregir), posibilidad de transmitir mensajes prioritarios, bajo coste de instalación y de conexión por nodo, recuperación rápida ante eventos anormales en la red y alta fiabilidad. En este nivel y el nivel de planta podemos ubicar las redes MAP (Manufacturing Automation Protocol) como ejemplo representativo.

- ***Bus de Campo (Field Bus):*** Para sustituir cableado entre sensores-actuadores y los correspondientes elementos de control. Este tipo de buses debe ser de bajo coste, tiempo real, permitir la transmisión serie sobre un bus digital de datos con capacidad de interconectar controladores con todo tipo de dispositivos de entrada-salida sencillos, y permitir controladores esclavos inteligentes. Además, deben gestionar mensajes cortos eficientemente, tener capacidad de manejar tráfico de eventos discretos, poseer mecanismos de control de error (detección y corrección), transmitir mensajes prioritarios, tener un bajo coste de instalación y de conexión por nodo, poder recuperarse rápidamente de eventos anormales en la red y responder rápidamente a los mensajes recibidos. Por regla general, tienen un tamaño pequeño (5 a 50 nodos), utilizan tráfico de mensajes cortos para control y

sincronización entre los dispositivos, y la transferencia de ficheros es ocasional o inexistente. Según la cantidad de datos a transmitir, se dividen en buses de alto nivel, buses de dispositivos (unos pocos bytes a transmitir) y buses actuador/sensor (se transmiten datos a nivel de bit), pero en ningún caso llegan a transmitir grandes bloques de información.

En la actualidad, las empresas utilizan un mismo tipo de software para la gestión de todo el sistema de comunicaciones, pero en el mercado existen diferentes tipos de herramientas para gestionar cada una de las capas y niveles en las redes industriales. Estos sistemas y paquetes software se denominan SCADA (Supervisory Control And Data Acquisition) y comprende diversas funciones como:

- Manejo del soporte o canal de comunicación.
- Manejo de uno o varios protocolos de comunicación.
- Manejo y actualización de una Base de Datos.
- Administración de alarmas (Eventos)
- Generación de archivos históricos.
- Interfaces con el operador (MMI - Man Machine Inteface)
- Capacidad de programación (Visual Basic, C)
- Transferencia dinámica de datos (DDE)
- Conexión a redes

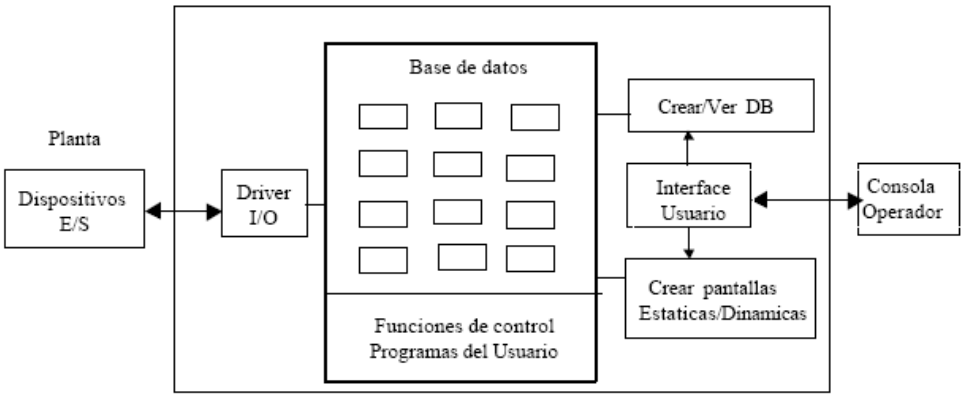

Figura 2.2. Arquitectura básica de un Sistema SCADA

Hay varios paquetes de calidad que se dedican a brindar estos servicios a la industria como: I-FIX, INTOUCH, RFACTORY, TAURUS, REALFLEX, GENESIS, LABVIEW por nombrar proveedores independientes, y fabricantes de equipos de medición y control como Rockwell software RSView32, Siemens WinCC, etc.

Estas herramientas son muy utilizadas pues todo proceso productivo con cierto grado de automatización debe disponer de un sistema de supervisión y control que proporcione la información imprescindible para la toma de decisiones basadas en la propia información del proceso y otras informaciones del resto de la organización.

El software SCADA se ajusta a estas premisas. Tienen 4 niveles principales:

- Gestión Intercambio de información para la toma de decisión estratégica.
- Operación Supervisión, mando y adquisición de datos del proceso.
- Control Dispositivos de control distribuido.
- Sensores y Actuadores Dispositivos de campo e instrumentación.

Los sistemas SCADA deben tener capacidad para comunicarse con múltiples redes de instrumentos, aun siendo de distinta procedencia y fabricantes. Para realizar dicha transacción se utilizan dos tipos de métodos. El primero permite la comunicación con otros paquetes de software por medio de DDE (Dynamic Data Exchange) – DLL (Dynamic Link Libraries) como canal de comunicación, implementados por el sistema operativo, que permite que diversos paquetes de software envíen y reciban datos comunes (Casi exclusivo de Microsoft), pero ahora se pretende dar una mayor flexibilidad entre sistemas operativos, lenguajes de programación y además comunicarlos con otras redes de tipo mundial (Internet) creando lo que se llama ahora arquitectura unificada (Unified Architecture OPC foundation), que integra servicios y redes industriales para cualquier tipo de sistema y brindando todos los beneficios al mismo tiempo. Esta idea surgió de la dificultad presentada al momento de realizar operaciones de control industrial y tratar de compartir información entre dispositivos inteligentes de campo, así como con el resto de la empresa. El problema hasta ahora se había resuelto escribiendo un sin número de protocolos, que definen de que manera se estructuran los datos que transmite cada dispositivo. Esta diversificación obligaba a

los desarrolladores de software SCADA a incorporar centenares de drivers para cada fabricante.

Por esta razón se esta desarrollando una norma de intercambio de datos para el nivel de planta basada en la tecnología OLE (Object Linking and Embedding) denominada OPC (OLE for Process Control), que permite un método para el flujo transparente de datos entre aplicaciones corriendo bajo sistemas operativos basados en Microsoft Windows y se pretende ampliar a cualquier tipo de sistema o plataforma. Se dispone de una versión inicial de la norma desde mayo de 1996.

OPC es un primer paso concreto que permite una red para compartir los datos de los dispositivos a nivel de proceso. No obstante la fundación OPC esta tratando de integrar todos los servicios y hacerlos portables para los diferentes tipos de plataformas existentes en el mercado y en Julio de 2003 lanzó la especificación XML-DA que busca este tipo de integración.

2.4 ARQUITECTURAS DE REDES INDUSTRIALES Y CAPAS OSI

La arquitectura de los dispositivos de campo en general trata de presentar las mismas características, y por tanto son muchas veces bastante parecidas, sin embargo han aparecido numerosos estándares para su implantación industrial y a pesar de tratarse de estándares abiertos, cada protocolo suele estar impulsado por un fabricante diferente, por lo que existe una pequeña batalla enmascarada por el control del mercado a través de la filosofía de sistemas abiertos. Entre los diferentes protocolos existen ciertas diferencias, pero generalmente es posible realizar el mismo tipo de aplicaciones sobre cualquiera de ellos.

En general los buses de campo, las redes de célula, y las redes de planta emplean las capas o niveles de la torre OSI 1 y 2, además de la interfase de usuario. Los niveles del 3 al 7, no están definidos. La optimización de esta arquitectura asegura una transmisión de datos rápida y eficiente. Por ejemplo en el bus de campo PROFIBUS, existe un Direct Data Link Mapper (DDLM) que permite a la interfase de usuario un

acceso sencillo al nivel 2. En Profibus - FMS están definidos los niveles 1, 2 y 7. El nivel de aplicación se compone de FMS (Fieldbus Message Specification) y LLI (Lower Layer Interface). FMS contiene el protocolo de aplicación y otorga al usuario una amplia selección de potentes servicios de comunicación. LLI implementa las distintas relaciones de comunicación y proporciona a FMS, con independencia del dispositivo, un acceso al nivel 2. El nivel 2 de Profibus, denominado FDL (Fieldbus Data Link) implementa el control de acceso al bus y la seguridad en los datos.

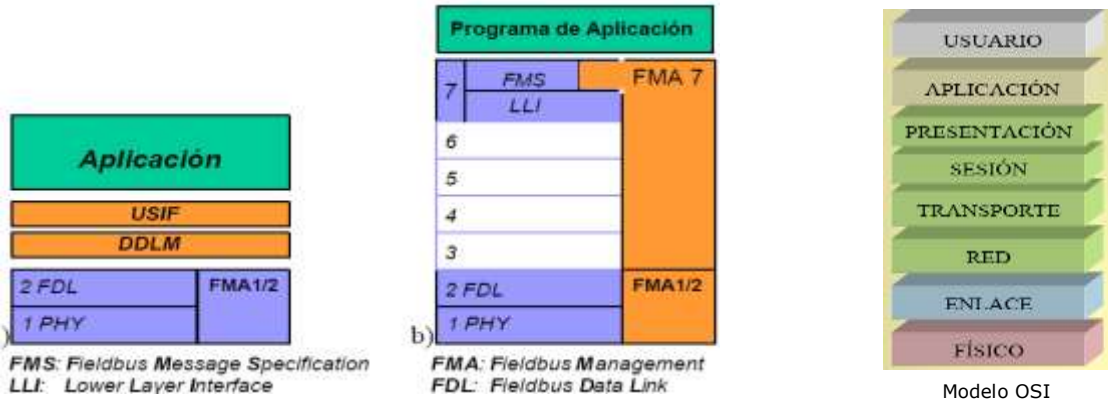

Figura 2.3. Arquitectura de Redes Industriales y Capas del Modelo OSI

La capa 2 es la que ofrece el enlace de datos con las aplicaciones y/o las funciones de mensajes (en caso de FMS). Esta capa ofrece cuatro servicios de comunicación que son solicitados por los niveles superiores a través de los puntos de acceso al servicio (SAPs) del nivel 2. Estos servicios son:

- SDN (Send Data with No Acknowledgement), envío de datos a una estación o a todas (broadcast) sin acuse de recibo.

- SRD (Send and Request Data with Reply), envío de datos a una estación y, al mismo tiempo, petición de datos a la misma, esperando una respuesta con datos inmediata.

- SDA (Send Data with Acknowledgement), envío de datos a una estación con acuse inmediato (sólo disponible en FMS).

- CSRD (Cyclic Send and Request Data with Reply), donde la estación activa posee una lista de sondeo (polling) que ejecuta para consultar a varios nodos de manera cíclica mientras posee el testigo (sólo disponible en FMS).

Actualmente se implementan pasarelas y puentes de estas redes con el protocolo TCP/IP que es el utilizado para transporte en las redes de Internet, lo cual permite aprovechar toda la infraestructura que se tiene, así existen MODBUS y PROFIBUS TCP/IP, los cuales permiten el trasporte de información de dispositivos de campo sobre una red industrial.

Aun con estos "ajustes" que se le realizaron a los protocolos de bus de campo, era muy difícil interactuar entre redes de diferentes compañías, pero con la aparición de la especificación OPC DX (de OPC Foundation), se dio fin a esta batalla entre los productores de protocolos de redes industriales. En general, OPC trabaja desde la parte superior de la torre OSI, pero puede interactuar con cualquier otra parte de las redes y acceder a ellas, y aunque existe dependencia de tener un servidor en cada dispositivo y por cada tipo de servicio que se necesita (DX acceso a redes, DA acceso a datos, Alarmas, etc.) con el lanzamiento de la especificación XML-DA y con los esfuerzos de la OPC Fundation para tener una arquitectura unificada, se está tratando de superar esto.

2.5 PLATAFORMAS Y HERRAMIENTAS DE DESARROLLO EN LA INDUSTRIA

En general, todos los desarrollos, herramientas, controladores para dispositivos, y aplicaciones que se ejecutan en las redes industriales lo hacen bajo la plataforma Windows y con protocolos muy relacionados con Microsoft, lo cual hace las licencias y los paquetes muy costosos y dependiendo del tipo de servicio que preste, de igual manera aumenta el precio.

De esta misma forma todos los lenguajes de programación y los entornos de desarrollo para este tipo de aplicaciones están directamente relacionados con Microsoft y solo

soportan sistemas distribuidos como COM/DCOM, mas no sistemas como RMI o CORBA que en sistemas de comunicación son muy importantes y dan mas flexibilidad y portabilidad al producto. Esto ha hecho que la "pequeña" pelea que existía entre los diferentes productores de protocolos de redes de campo por compatibilidad se haya convertido en un monopolio casi completo para Microsoft, pues todas las herramientas de desarrollo y aquellas herramientas de administración brindan soporte o fueron desarrolladas en algún producto de esta gran industria, pero todo esto no es coincidencia, se debe a el gran soporte y la calidad de los productos ofrecidos por Microsoft, ya que casi ninguna industria en el mundo, se arriesga a utilizar una plataforma como Unix, Apple, o Linux, y mucho menos administrar sus procesos con algún paquete de poca credibilidad y trayectoria.

A pesar de que no existen muchas herramientas de programación, las empresas a nivel internacional se están preocupando por esto y están realizando desarrollos para disminuir esta gran brecha y brindar todos los beneficios en un solo paquete. Empresas que ofrecen integración entre plataformas y bajos precios de licencias.

Estas empresas son Technosoftware AG (Suiza), Softing AG (Alemania), Provea, MERZ, spol. s r.o**.** (Republica checa), ETM Aktiengesellschaft (Austria). El precio que cobran por sus herramientas oscila entre $3.000US y $7.000US, que comparada con las herramientas tradicionales, que pueden oscilar entre los $15.000US y los $30.000US, son significativamente económicas, además de conseguir un ahorro adicional si se escoge como plataforma de trabajo una de carácter libre.

En la actualidad, a nivel nacional ninguna empresa desarrolla estas herramientas, pero algunas instituciones están invirtiendo en I+D para poder dar soporte a este tipo de plataformas y logar una competitividad a nivel internacional y de esta manera, llegar a globalizar el producto.

3. OPC MULTIPLATAFORMA

3.1 ¿QUÉ ES OPC?

OPC es conectividad abierta en automatización industrial y los sistemas empresariales que funcionan en la industria. Se asegura la interoperabilidad a través de la creación y el mantenimiento de especificaciones estándares abiertos. Actualmente hay 7 especificaciones estándares completas o en desarrollo.

Basada en estándares fundamentales y tecnología general del mercado de la computación, la Fundación OPC adapta y crea especificaciones que satisfacen las necesidades especificas de la industria. De esta manera, OPC crea nuevos estándares a las necesidades que surgen y adapta los estándares existentes para utilizar nuevas tecnologías.

OPC es una serie de especificaciones estándares. El primer estándar (originalmente llamado simplemente especificación OPC y ahora llamado la Especificación de Acceso a Datos) resultado de la colaboración de un grupo de proveedores líderes de automatización a nivel mundial en cooperación con Microsoft. Originalmente se basó en las tecnologías OLE, COM (Modelo de Objetos Componentes) y DCOM (COM Distribuido), definiendo un conjunto estándar de objetos, interfaces y métodos para usar en el control de procesos y aplicaciones de automatización y fabricación, facilitando la interoperabilidad. Las tecnologías COM/DCOM proporcionan la estructura para el desarrollo de los productos software. En la actualidad hay cientos de servidores y clientes OPC de Acceso a Datos (DA).

Las especificaciones OPC actuales incluyen:

- ***OPC Acceso a Datos***

  Usada para mover datos en tiempo real desde PLCs, DCSs, u otros dispositivos de control a HMIs (Interfaz Humano-Máquina) o clientes con interfaz. La especificación actual corresponde a la versión 3.0. Esta versión tiene características de versiones anteriores sumado a esto el mejoramiento de las capacidades de búsqueda e incorporación del esquema XML-DA.

- ***OPC Alarmas y Eventos***

  Proporciona notificaciones de alarmas y eventos por demanda (en contraste con el continuo flujo de datos del Acceso a Datos). Esto incluye alarmas de procesos, acciones de operador, mensajes de información, y mensajes de audio.

- ***OPC Batch***

  Esta especificación lleva la filosofía OPC a las necesidades especializadas de procesos de batch. Proporciona interfaces para el intercambio de capacidades de equipo y condiciones actuales de operación.

- ***OPC Intercambio de Datos***

  Esta especificación nos lleva desde cliente/servidor hasta servidor/servidor con comunicación a través de redes Ethernet de bus de campo. Esto proporciona interoperabilidad multi-vendedor. Y agrega configuración remota, y servicios de gestión, monitoreo y diagnóstico.

- ***OPC Acceso a Datos Históricos***

  Donde OPC Acceso a Datos proporcione acceso en tiempo real, continuamente van a cambiar los datos, OPC Acceso a Datos Históricos proporciona acceso a los datos realmente almacenados. Desde un simple sistema de datos seriales hasta un

complejo sistema SCADA, los archivos históricos pueden ser recuperados de manera uniforme.

- ***OPC Seguridad***

  Todos los servidores proporcionan información que es valiosa para la empresa y si es impropiamente actualizada, podría ocasionar consecuencias significativas para los procesos de planta. OPC seguridad especifica como controlar el acceso de los clientes a los servidores con el fin de proteger esta información importante y resguardarla contra modificación no autorizada de los parámetros de los procesos.

- ***OPC Acceso a Datos XML (XML-DA)***

  Proporciona reglas consistentes y flexibles y formatos para publicar los datos de la planta del proceso usando XML, bajo la influencia del trabajo hecho por Microsoft y otros sobre SOAP y servicios Web.

- ***OPC Datos Complejos***

  Es una especificación que acompaña a la de Acceso a Datos y XML-DA para asi permitir a los servidores publicar y describir tipos de datos más complicados, como estructuras binarias y documentos XML.

- ***OPC Commands***

  Se ha formado un grupo de trabajo para desarrollar un nuevo conjunto de interfaces que permitan a los clientes y servidores OPC identificar, enviar y monitorear órdenes de control, las cuales se van a ejecutar en un dispositivo.

3.2 ARQUITECTURA UNIFICADA

En la actualidad la mayoría de sistemas telemáticos tratan de integrarse no solo en la parte de servicios ofrecidos, sino también en cuanto a plataformas y sistemas operativos se refiere. Esto se hace desde su creación para permitir flexibilidad y compatibilidad entre sistemas. Sin embargo, contrario a estos sistemas, el consorcio OPC ha realizado todos sus desarrollos bajo plataformas Windows y su sistema COM/DCOM para la comunicación, generando especificaciones e interfaces muy útiles para todos los fanáticos y desarrolladores de estos entornos. Veinte años después, debido a las exigencias de los usuarios y a que sistemas operativos diferentes a Windows se hacen más prominentes al nivel empresarial e industrial, este concepto tuvo que ser modificado y migrar a una arquitectura unificada (AU).

La AU consiste exactamente en unificar los aspectos OPC construyendo un conjunto de servicios comunes para mover datos e información desde la parte baja de la fábrica hasta los pisos superiores de la empresa. Esta arquitectura se basa en el modelo de servicios dando funcionalidades a todas las especificaciones OPC que se unirán en una sola bajo esta misma arquitectura. Esta arquitectura debe brindar Servicios Web y XML abarcando también la arquitectura Microsoft .NET.

Como se muestra en la figura 3.1, con esta AU se pretende unir a través de XML y .NET todas las partes de la empresa, desde el manejo financiero y planeación de recursos (ERP), puntos de acceso a los servicios, pasando por la parte de manufactura (MES), producción, supervisión (SCADA) y mantenimiento, con todas sus interfaces hombre máquina (HMI), hasta llegar a la parte mas baja de la empresa: la planta, con todos sus dispositivos y sistemas de comunicación.

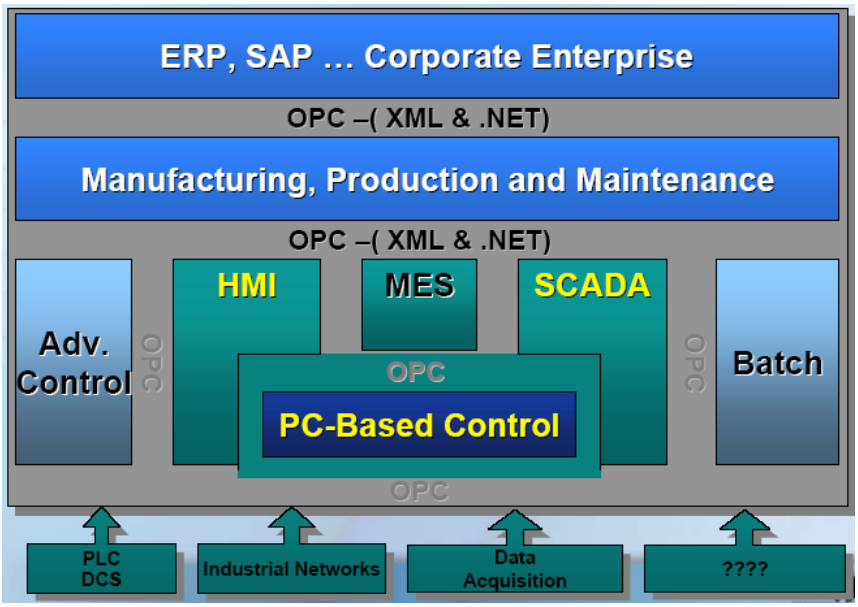

Figura 3.1. Mejoramiento de la Interoperabilidad y Conectividad

Con la arquitectura unificada se incrementa la integridad, redundancia en la información, así como se asegura el fácil desarrollo e integración.

Los productos OPC existentes basados en componentes serán capaces de encajar en los servicios Web de la AU, permitiendo que los productos ya existentes transformen sus datos en información por fuera de las fronteras convencionales de intranet, así como ser interoperables sobre diferentes tipos de plataformas y además solucionando el problema crítico de la escalabilidad.

3.3 OPCIONES DE CONEXIÓN MULTIPLATAFORMA

La necesidad de interoperabilidad entre los productos no solo en redes industriales, sino también en redes telemáticas ha obligado a las diferentes casas desarrolladoras a crear nuevos productos que permitan la conexión entre diferentes sistemas que fueron diseñados para ser totalmente independientes cuando fueron concebidos. Existen diferentes formas de realizar esta tarea, dependiendo del tipo de sistema, red,

protocolo, o dispositivos que quiera comunicar. Por ejemplo en la industria siempre fue muy agresiva la batalla entre protocolos de redes de bus de campo; sin embargo años mas tarde se dejó en el mercado la especificación OPC DX que acabo prácticamente con el problema; sin embargo, el único problema no ha sido la comunicación entre diversos protocolos, sino también el esfuerzo de muchas empresas de monopolizar el mercado con su gran infraestructura y soporte, aislándose de otros sistemas muy buenos pero con menos difusión como es el caso del gigante Microsoft y de otros vendedores no tan grandes como Macintosh, UNIX, Solaris, y Linux. Ante estas diferencias se presenta la necesidad de utilizar un lenguaje de programación que sea ampliamente compatible con los demás sistemas, además de que permita la implementación de protocolos y especificaciones industriales. El problema es que no existe una herramienta tan poderosa que brinde todas las características juntas, de esta forma toca elegir una que brinde algunas ventajas, algunas desventajas, pero que ante todo permita la interoperabilidad.

En el mercado actual los lenguajes de programación como Visual Basic .NET, Delphi, proveen un soporte excelente para el manejo de redes de campo y en general para redes industriales, y cuentan con algo muy importante y es que la mayoría de dispositivos traen sus drivers o manejadores para entornos Windows y comunicaciones COM/DCOM, lo que facilitan los desarrollos por compatibilidad pues todos los protocolos son de la misma familia. Sin embargo, cuando se quiere utilizar una plataforma diferente a Windows, los mejores lenguajes de desarrollo son C++ y JAVA, el primero teniendo una complejidad un poco mayor a la hora de manejarlo, y el segundo basado en el primero, con gran portabilidad e interoperabilidad multiplataforma, gran cantidad de herramientas para comunicaciones, pero poco utilizado en la industria por sus tiempos de respuesta y por no poder acceder a memoria directamente, que es donde los dispositivos de campo guardan la información.

Existen dos formas para acceder a las redes industriales y los dispositivos de campo desde estos dos lenguajes, estas dos formas son:

- Puentes COM/DCOM y OPC-DA
- OPC y XML

3.3.1. Puentes COM/DCOM y OPC-DA

A diferencia de JAVA, C++ no tiene ningún problema de acceso a COM/DCOM por lo tanto a OPC y a los dispositivos de campo, sin embargo, no ofrece tantas herramientas para las comunicaciones telemáticas ni la portabilidad de un paquete desarrollado en JAVA.

Por otro lado JAVA es casi incompatible con el modelo de objetos de Windows (COM/DCOM) y para poder acceder y modificar los datos que se trasmiten por este modelo de comunicación se debe utilizar sus interfaces nativas (Native Interfaces) las cuales recurren a código C++ que es el lenguaje en el cual se creó JAVA.

En el mercado existe gran diversidad de herramientas, unas más completas que otras, pero básicamente las arquitecturas son las mismas. Entre ellas se encuentran:

- **Utilización de "Stubs" para cada objeto COM/DCOM:**

En muchas arquitecturas de objetos remotos como Java y CORBA, un *Stub* es un objeto *proxy* que reside del lado de los clientes y se encarga de delegar todas las invocaciones a métodos remotos a su correspondiente *Skeleton* que reside del lado del servidor. El *Stub* es el objeto con el cual interactúan los clientes. Un *Skeleton* es un objeto Proxy que reside en el servidor y se encarga de delegar los requerimientos que llegan a través de la red al objeto remoto requerido. Esos pedidos a través de la red han sido enviados por el Stub que reside del lado del cliente.

Así, para realizar la comunicación entre un objeto Java y uno COM/DCOM se pueden utilizar "Stubs" especiales, o parecidos que sigan la misma metodología. Algunas herramientas crean "Stubs" compartidos, y "Stubs" genéricos para poder facilitar la comunicación. Los Stubs compartidos son utilizados para convertir métodos COM y Win32 que tienen declaraciones comunes. Semánticamente, los tipos de los argumentos y los valores de retorno de estos métodos son todos muy diferentes. Sin

embargo, todos estos tipos se convierten a los mismos valores. Así, es posible implementar todos estos métodos con un solo punto de entrada JNI.

Por supuesto, no todas las funciones se pueden implementar con un número finito de "Stubs" compartidos. Cuando las declaraciones son diferentes al caso anterior se utilizan los "stubs" genéricos. El "Stub" genérico es un "marshaller" o transformador verdadero, similar a RMI (Remote Method Invocation) o DCOM (Distributed Component Object Model). El "Stub" genérico ve una llamada a una función como una secuencia de eventos. Las funciones intrínsecas serializan una función dentro de un mensaje de requerimiento, y el "Stub" genérico mueve este mensaje dentro del espacio nativo. Allí, otro conjunto de funciones intrínsecas convierten el requerimiento serializado a una pila de llamadas e invocan la función. La secuencia completa hace en proceso inverso para serializar un mensaje de respuesta con valores o excepciones de retorno y lo envía de regreso al llamante.

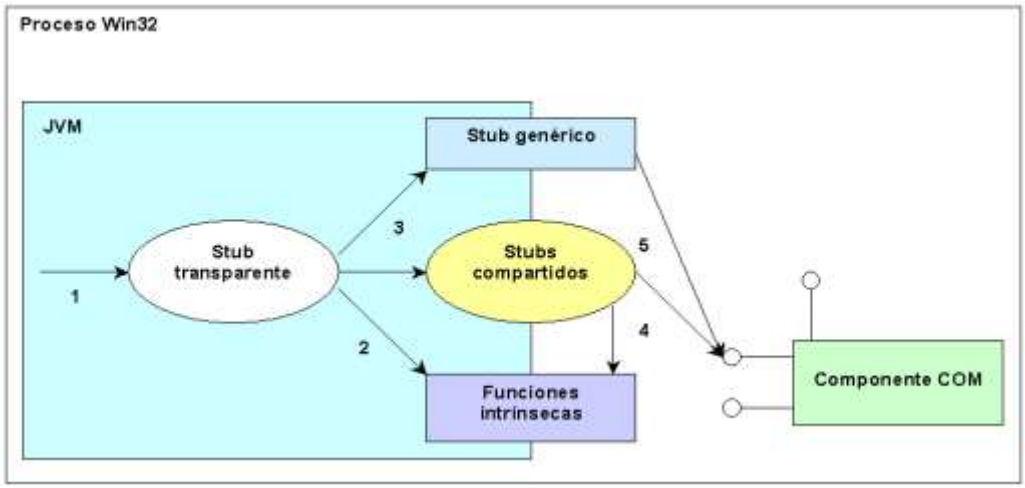

Figura 3.2. Arquitectura con "stubs[2]"

De acuerdo a la convención de nombramiento RMI y CORBA, tal componente se llamaría skeleton; pero a diferencia de estos, los "stubs" genéricos llevan información de tipo.

[2] Natalia Maya, Eva Maya. Anexo A: Puentes Java-COM.

De esta forma se genera un "Stub" transparente a partir de la información de tipo, y es diferente para cada objeto COM o DLL. Los "Stubs" compartidos manejan declaraciones de métodos comunes llamando métodos nativos con declaraciones JNI correlacionadas. El "Stub" genérico maneja los métodos que tienen declaraciones menos comunes y por lo tanto no son "Stubs" compartidos. Tanto el "Stub" genérico como los "Stubs" compartidos utilizan funciones de ayuda (llamadas funciones intrínsecas) para convertir tipos de datos particulares.

- **Utilización de clases Proxy**

Genera las clases "Proxy" C++ o también llamados Beans, para una clase Java simple. Las clases "Proxy" generadas permiten a los desarrolladores instanciar y manipular objetos Java en tiempo de ejecución, como si fueran clases C++ nativas. Las clases "peer" generadas proporcionan un método fácil para que los desarrolladores implementen métodos nativos declarados en sus clases Java, es decir, implementen cualquier función nativa Java en C++. También puede realizar gestión de hilos, gestión de excepciones y conversión automática de tipos, entre otras.

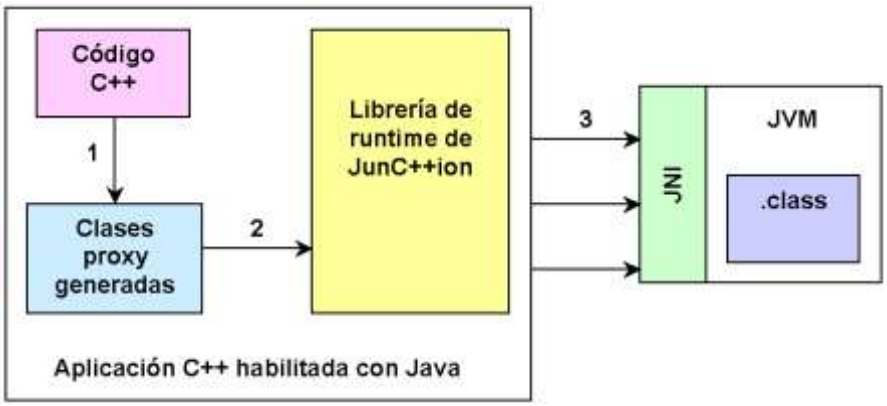

Figura 3.3. Utilización de clases Proxy

Este proceso utiliza clases "skeleton" Java generadas para todas las interfaces COM expuestas. Una clase "skeleton" Java, que convierte llamadas hechas a través de la interfaz COM en llamadas a métodos Java, corre en el servidor y tiene tres importantes responsabilidades. Primero, crea una tabla de funciones virtuales COM basada en la

definición de la interfaz. Segundo, invoca el método apropiado en la implementación de la interfaz. Finalmente, "marshal"/"unmarshal" parámetros de métodos y maneja excepciones Java.

También utiliza una librería de tipo asociada con el servidor de automatización y genera una clase "wrapper" Java para cada interfaz Dispatch expuesta por el servidor de automatización COM (algo como un Stub).

Como se puede observar, las dos arquitecturas son casi iguales, la diferencia es el lenguaje en las clases de frontera, es decir en los Skeletons y Stubs. Entre los diferentes puentes y herramientas que se encuentran en el mercado se encuentran: JNI++, JINTEGRA, Jawin, Jace, R-JAX, JunctionC++, xFunction, JACOB, JCOM, JacoZoon, Bridge2Java entre otras.

El manejo de estas herramientas no es tan fácil al tener que manejar lenguaje C++ y Java a la vez, y pocas son de carácter gratuito o sólo presentan servicios de comunicación local (COM), pero no distribuido (DCOM). Otra desventaja es a la hora de utilizar una invocación de objetos remotos, ya que adhiere retardos no solo por la ejecución de los métodos, sino también por los retardos inherentes al proceso de adquisición de datos directamente de memoria.

- **JCA (Java Connecting Architecture):**

J2EE Connector Architecture (JCA) es un estándar desarrollado por el Java Community Process tratando de integrar servidores de aplicaciones y EIS (Enterprise Information Servers) existentes. Esta herramienta es desarrollada por Sun Microsystem y la comunidad Java, y va mas que todo orientada al manejo e interoperabilidad con EJBs (Enterprise Java Beans).

La J2EE Connector Architecture ofrece una solución interesante al problema de interoperabilidad existentes entre EJB de la plataforma J2EE y componentes COM de la plataforma Microsoft.

La especificación de JCA incluye tres tipos de *Contracts* que homegeinizan la interfase hacia los sistemas externos. Estos *Contracts* son: *Connection Management contract*, que provee los mecanismos de pool de conexiones para soportar la escalabilidad en aplicaciones que acceden al EIS; *Transaction Management contract*, que soporta acceso transaccional al EIS, y *Security contract*, que extiende los contracts de manejo de transacciones para realizar un acceso seguro al EIS.

La conexión entre el Servidor de Aplicaciones y el EIS se realiza a través de los llamados *Resource Adapters*, que implementan los *contracts* anteriores en el EIS, sirviendo de intermediarios entre el Servidor de Aplicaciones y el EIS.

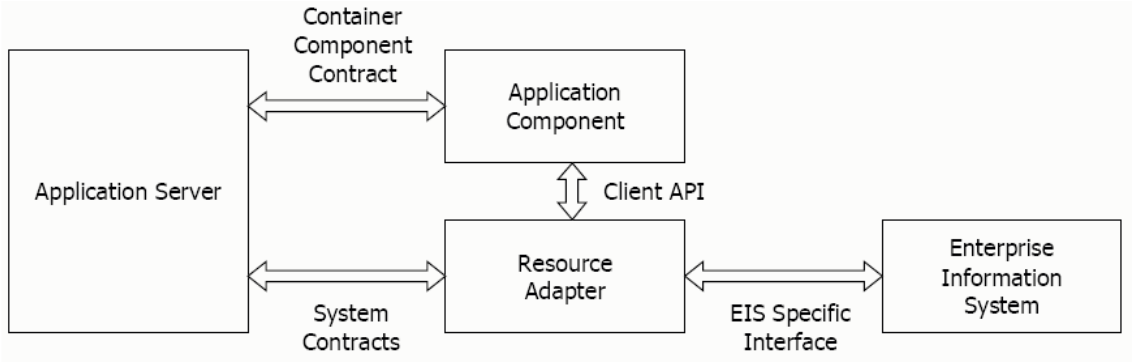

Figura 3.4. Arquitectura de JCA

JCA fue diseñado especialmente para resolver problemas de interoperabilidad, por lo cual al utilizar un JCA sería posible implementar un resource adapter específico para COM que junto a un servidor J2EE con soporte para JCA permita realizar transacciones distribuidas combinando componentes EJB y COM.

Actualmente JCA ha tenido una amplia aceptación en el mercado y existen varios servidores de aplicaciones J2EE con soporte para JCA. También existen gran cantidad de empresas dedicadas al desarrollo de JCA Connectors o resource adapters. Algunas empresas como INTRINSYC (JINTEGRA) y Javector Software han desarrollado resource adapters específicos para COM, permitiendo el manejo de transacciones, seguridad y pool de conexiones para COM. Desde cualquier plataforma, y en cualquier servidor de aplicaciones que soporte la especificación JCA.

Sin embargo, la versión actualmente implementada de JCA (1.0) presenta limitaciones, entre las cuales se destacan el no contar con mecanismos nativos para trabajar con meta-datos, lo cual obliga al desarrollo de estos módulos, comprometiendo el valor de la JCA para el acceso a sistemas externos e información heterogénea. Además, no cuenta con soporte nativo para XML, lo cual limita en forma importante la interfase con sistemas que generan o leen datos en este formato. Solo soporta acceso sincrónico y unidireccional.

3.3.2. OPC y XML

En los últimos años han cobrado importancia los sistemas cooperativos e integradores, que apuntan a brindar servicios con valor agregado reutilizando sistemas existentes. Aún más, en muchos casos se propone especialmente que esta interacción se realice a través del Web.

Estos sistemas integradores deben ser capaces de interactuar con otros sistemas en plataformas diferentes. Por lo tanto, para desarrollar estos sistemas resulta imprescindible resolver la interoperabilidad entre sistemas basados en plataformas heterogéneas.

Esta es la razón principal para que la Fundación OPC se diera a la tarea de crear un estándar XML para aumentar las capacidades de interoperabilidad de los sistemas SCADA que se ejecutaban sobre redes y servicios COM/DCOM. PLCs, DCSs, HMIs y varios programas usan los estándares OPC-COM para intercambiar datos en tiempo real entre dispositivos de campo, sistemas de control y otras aplicaciones de una manera estándar, proporcionando compatibilidad multivendedor e interoperabilidad.

Esta tecnología debe por tanto brindar fácil integración con aplicaciones existentes a través de Internet, ya que las aplicaciones existentes OPC-COM trabajan muy bien en el típico entorno LAN. Sin embargo, cuando se usa DCOM sobre Internet, se tienen muchos inconvenientes y problemas relacionados con la seguridad y el uso de puertos TCP/IP dinámicamente asignados (típicamente no permitido a través de firewalls corporativos).

Otra gran ventaja es que XML es una especificación del W3C diseñada para facilitar su transporte con protocolos de Internet. Este es frecuentemente transportado vía HTTP simplemente como páginas Web HTML ordinarias, pero también se transporta fácilmente por medio de otros protocolos, como FTP y SMTP.

El objetivo principal de una especificación en la cual los Servicios Web y el XML son el núcleo de la arquitectura, es el de brindar flexibilidad y cumplir con los principales objetivos propuestos por OPC para lograr una arquitectura unificada.

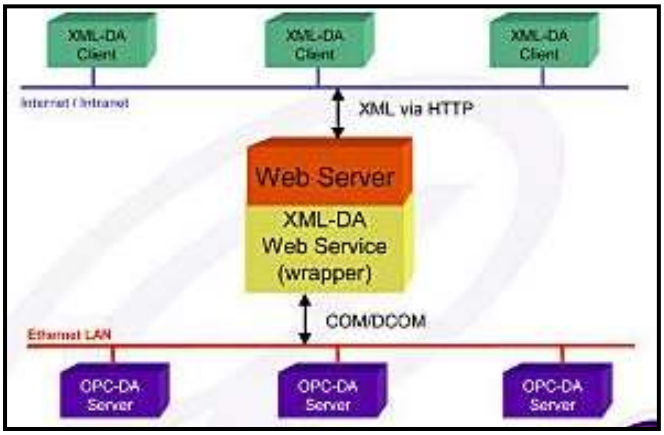

Figura 3.5. Arquitectura básica OPC-XML-DA

3.4 SELECCIÓN Y DESCRIPCIÓN DETALLADA DE ARQUITECTURA

Existen varias alternativas para solucionar el problema de interoperabilidad entre sistemas operativos y plataformas de desarrollo, y cada una de ellas tiene sus puntos fuertes y débiles. Ninguna de ellas muestra ser la solución por excelencia, por lo cual está fuertemente ligada a los requerimientos.

Para elegir la tecnología también deben ser tenidos en cuenta factores como portabilidad, confiabilidad y robustez, costo y soporte nativo. En el caso particular de las redes industriales, su principal servidor es .NET, y se quiere permitir la operación en sistemas operativos como LINUX, lo cual hace a JAVA y J2EE como una fuerte opción de interoperabilidad.

Las dos opciones más estandarizadas son los Servicios Web y los J2EE Connector Architecture (JCA), descritos en la sección anterior. Otras opciones son los adaptadores propietarios que utilizan Proxies o Beans, por ejemplo JINTEGRA, JACOB, ActiveX Bridge y Ja.NET.

En el contexto planteado de conexión J2EE-.NET, los Servicios Web presentan las siguientes ventajas: Puede utilizarse en conexiones sobre HTTP, lo cual lo hace utilizable para conexiones a través de Internet; permite interacción tanto sincrónica como asíncrona; son multilenguaje; tienen un nivel de desarrollo y compromiso empresarial muy importante en múltiples plataformas. Esto se traduce, por ejemplo, en la disponibilidad de herramientas de desarrollo. Como desventaja principal de los Servicios Web, se destaca que no incluye, al momento, control de transacciones y de seguridad.

Por su parte los JCA, presentan como ventaja principal que incluyen mecanismos de control de transacciones y seguridad, compatibles con los de J2EE. Como desventajas de JCA se señalan: desarrollo prácticamente nulo en plataforma Microsoft y especialmente .NET. Mecanismo fuertemente basado en Java.

A continuación se muestran unas tablas comparativas de interoperabilidad en las cuales el autor[3] menciona la posibilidad de utilizar la plataforma CORBA, y se resalta con rojo la combinación buscada en este sistema.

Cliente → Servidor ↓	Microsoft .NET	CORBA	J2EE
MS .NET	Nativa	No analizado	Web Services JCA. Adaptadores propietarios
CORBA	No analizado	Nativa	RMI sobre IIOP.
J2EE	Web Services Adaptarores propietarios	RMI IIOP	Sobre Nativa

Tabla 1. Opciones de interoperabilidad Sincrónica.

[3] **Interoperabilidad entre Servidores de Aplicaciones Heterogéneos.** R. Ruggia, J. Besil, C. Pais, D. Sande.

Cliente → Servidor ↓	Microsoft .NET	CORBA	J2EE
MS .NET	Nativa	No analizado	***Web Services + AXIS***
CORBA	No analizado	Nativa	Gateways
J2EE	Web Services .	Gateways .	Sobre Nativa

Tabla 2. Opciones de interoperabilidad Asíncrona.

En resumen, en la disyuntiva de elegir entre Servicios Web y JCA, se deben tener en consideración los siguientes factores:

- *Uso exclusivo en red local vs. Sistemas abiertos a Internet*. Mientras que en el primer caso los JCA serían una opción aplicable, el segundo lleva a usar HTTP, y por lo tanto Servicios Web.

- *Sistemas centrados en Java y/o J2EE vs. multiplataforma/lenguaje*. El primer caso sería abordable con JCA, mientras que el segundo lleva a usar Servicios Web como una buena forma de hacer portable y transparentes los mecanismos de interoperabilidad. Los otros tipos de conectores presentan como ventajas el hecho de estar desarrollados específicamente para resolver la interacción J2EE con Microsoft, y como desventaja principal la falta de estandarización.

En el caso del sistema que se quiere implementar, los Servicios Web son la tecnología que más se aproxima a lo requerido: Interoperabilidad multiplataforma, para poder ejecutarlo en cualquier sistema operativo, y además se ajustan perfectamente a la especificación XML-DA que esta dirigida a los procesos y a integrar muchas mas funcionalidades de sistemas SCADA y redes industriales.

3.5 APROXIMACIÓN A LA ESPECIFICACIÓN OPC XML-DA

XML-DA es la ultima especificación que la fundación OPC dejo al servicio de la industria y la cual se basa en algunas especificaciones anteriores como OPC-DA, OPC-HDA, OPC para alarmas y alarmas y OPC-Batch.

La razón es permitir operabilidad entre lenguajes y sistemas operativos al utilizar XML, un lenguaje que permite el intercambio de información estructurada entre aplicaciones.

El propósito de la versión 1.0 es el de brindar interoperabilidad entre aplicaciones basándose en compartir, simplificar e intercambiar datos OPC entre diversos niveles de la jerarquía en una red industrial; desde los dispositivos de campo hasta los sistemas de la empresa y una amplia variedad de plataformas.

Los principales objetivos de OPC XML-DA son:

- Soportar el acceso a OPC DA versión 2.0x y 3.0
- Soportar HTTP, y SOAP.
- Dar soporte para Servicios Web asíncronos basados en suscripción.

3.5.1. Conceptos Fundamentales

Servicios Web: Los Servicios Web son una tecnología emergente para permitir exponer servicios a través de la Web. Permite que aplicaciones Web interactúen dinámicamente con otras aplicaciones Web, utilizando para ello estándares abiertos como XML (Extensible Markup Language), UDDI (Universal Description, Discovery, and Integration) y SOAP (Simple Object Access Protocol). Las funciones que pueden ser realizadas por los Servicios Web pueden ir desde simples intercambios de información hasta complicados procesos de negocios. Las empresas vendedoras de servicios pueden encapsular su lógica y procesos de negocio mediante Servicios Web y exponerlas para que sus clientes las consuman a través de la Web.

Aparte de un avance tecnológico los plantean un cambio en los modelos de negocios de las empresas vendedoras de servicios, poniendo en jaque a los mecanismos actuales de venta de software y servicios computacionales. Exponiendo los servicios en la Web se tiene acceso a un espectro mucho mayor de clientes potenciales y facilita enormemente el acceso a dichos servicios.

Las ideas detrás de los Servicios Web no son nuevas, ya estaban presentes en tecnologías como RPC (Remote Procedure Call). Los Servicios Web permiten realizar invocaciones a procedimientos remotos, tanto en redes de porte pequeño como puede ser una Intranet empresarial como en redes de gran porte como Internet. Esto es posible porque están basados en un protocolo simple y liviano para realizar las invocaciones. El protocolo que permite realizar las invocaciones es SOAP. SOAP es un protocolo estándar creado por la W3C, basado en XML. El eXtensible Markup Language es el lenguaje en el que se estructura, describe e intercambia la información llegando a ser el formato preferido para codificación y movimiento de datos en una forma abierta e independiente del sistema. XML (formato de texto altamente estructurado) es comprensible tanto por la máquina como por las personas.

Actualmente las plataformas J2EE y .NET son las principales que incorporan Servicios Web en forma prácticamente nativa.

Los Servicios Web se caracterizan por permitir conexiones tanto sincrónicas request/reply como asíncronas, conectando aplicaciones no necesariamente J2EE y usar HTTP como protocolo, lo cual lo hace muy apropiado para conexiones en Internet. Por el momento no incluye mecanismos nativos de seguridad y control de transacciones, este último en desarrollo.

La conectividad es la clave de la mayoría de esos sistemas empresariales, y muchas de esas aplicaciones ahora abarcan XML como el método preferido del intercambio de datos. Con OPC también abarcando XML, los datos en tiempo real sobre Internet estarán ahora al alcance de las aplicaciones industriales.

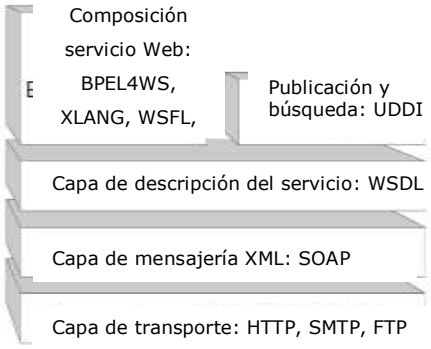

Figura 3.6. Arquitectura de los servicios Web

SOAP (Simple Object Acces Protocol): Está compuesto por cuatro componentes fundamentales, un envoltorio que define un framework para describir los mensajes y cómo estos deben ser procesados, un conjunto de reglas para codificar (o serializar) instancias de tipos de datos definidos por las aplicaciones, una convención de cómo representar invocaciones remotas y sus respuestas, y una convención para vincular el intercambio de mensajes con un protocolo de transporte. El protocolo de transporte que normalmente se utiliza con SOAP y el utilizado por los Servicios Web y XML-DA es HTTP. Esencialmente lo que realiza SOAP es utilizar mensajes para invocar métodos o funcionalidades en un servidor remoto (RPC), de una forma muy similar a RMI o COM, pero sin la necesidad de tener algún objeto publicado y bajo protocolos estándar en Internet.

HTTP (HyperText Transfer Protocol): Utilizar HTTP como protocolo para las invocaciones facilita el uso de la infraestructura Web ya existente en la actualidad en prácticamente toda empresa, para el intercambio de información o publicación de servicios de la empresa. Asimismo, este intercambio es posible realizarlo sin tener que agregar más protocolos a los ya existentes en los Firewalls corporativos para permitir estos flujos de información, lo cual sería necesario si se utilizara algún otro protocolo.

WSDL (Web Service Description Language): Es un lenguaje también basado en XML que permite describir los contratos de cada servicio. Este lenguaje fue desarrollado conjuntamente por Microsoft e IBM y sometido para su aprobación como un estándar. Esta especificación permite conocer por parte de los consumidores los métodos exportados, así como los parámetros y datos retornados.

UDDI (Universal Description, Discovery and Integration): Es una interfase para encontrar servicios Web como si estuvieran localizadas en un directorio de páginas amarillas en Internet. XML-DA no incluye esta interfaz, y aunque es muy fácil de implementar, no es muy funcional en sistemas Industriales cuando se trata de publicación y acceso a dispositivos o datos propios del proceso, pero sí lo podría ser a la hora de comercializar el producto, u ofrecer algún tipo de servicio.

Detección de servidores XML-DA: En la especificación, la fundación OPC no especifica un mecanismo que detecte automáticamente los servidores OPC-XML-DA en un nodo especifico de la red industrial. A pesar de esto, propone utilizar UDDI como protocolo estándar por su amplia aplicación en los servicios Web y la fácil consecución de herramientas desarrolladoras para gestionar los mismos.

3.5.2. Arquitectura Asíncrona de Suscripción

El diseño de suscripción en OPC-XML-DA emplea un estilo "registrar-sacar" (polled-pull). Así el cliente entra en un contrato no muy estricto con el servidor. La aplicación del cliente inicia la suscripción y hace un acuerdo en el periodo de petición-actualización. Con el fin de simular de una mejor forma las capacidades de respuesta (callback) del sistema OPC-DA/COM original se diseñó un mecanismo de "Control de Respuesta". Este mecanismo puede ser utilizado para reducir el tiempo de latencia de un valor reportado de cambio a un cliente y minimizar el numero de viajes de ida y vuelta (Round trips) entre el cliente y servidor. XML-DA soporta los siguientes servicios (transacciones) de suscripción básica:

- Subscribe
- SubscriptionPolledRefresh
- SubscripcionCancel

Subscribe es utilizado para iniciar un contrato de suscripción con el servidor. Se llama a *SubscripcionPolledRefresh* periódicamente para adquirir los últimos cambios de valor en el ítem. *SubscripctionCancel* es usada para terminar el contrato de suscripción con el servidor.

- **Descripción de la Suscripción "Registrar-Sacar" (Subscription Polled Refresh)**

El cliente inicia la suscripción y el servidor devuelve un "handle" o código de manejo como respuesta a la petición. El servidor también retorna cualquier valor inicial (valor, calidad (quality), y fecha (timestamp)) que están de hecho listos y disponibles si la opción *ReturnValuesOnReply* está en *verdadero*. Entonces el cliente entra en un ciclo de *registro* y continúa *Escribiendo* periódicamente por medio de la publicación de una petición de actualización pasando cada vez el código de manejo de suscripción. El servidor responde inmediatamente devolviendo todos los valores y/o los cambios de calidad desde el registro anterior. Este proceso continúa hasta que el cliente no desea mantener más la suscripción, punto en el cual publica una petición cancelación de suscripción al servidor. El servidor limpia los recursos y lleva acabo cualquier otro tipo de acciones necesarias para terminar el contrato de suscripción que llevaba con el cliente.

La interacción de suscripción de registro básica entre el cliente y el servidor se puede ver en la siguiente figura:

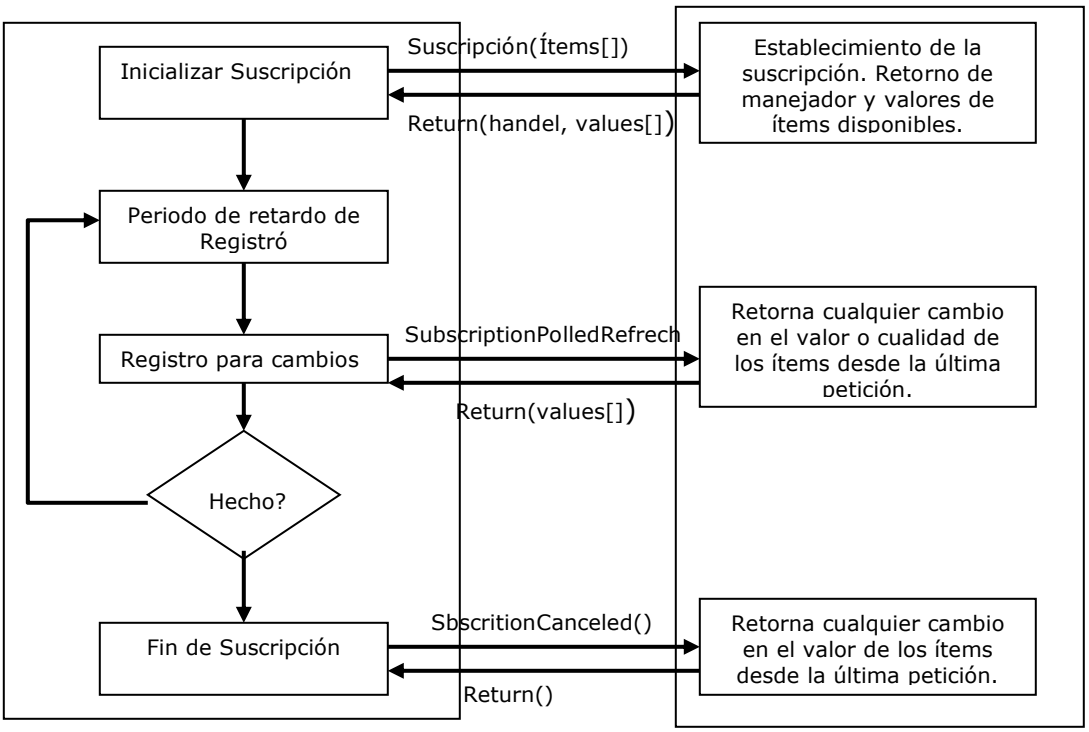

Figura 3.7.Interacción de Suscripción

La aplicación del cliente debe estar lista para manejar errores de condición de los servidores. Dependiendo del tipo de error retornado se realizan las acciones pertinentes. Se espera que un cliente bien diseñado interprete los códigos de error apropiadamente, incluyendo el manejo de tiempos límites de respuesta tanto en la suscripción inicial como en la actualización de registro de suscripción.

Cuando existen errores fatales de manejo de tiempos límites, el cliente debe limpiar la suscripción existente y crear una nueva y ahí si cerrar la anterior. Esto se puede apreciar de una mejor forma en la siguiente figura:

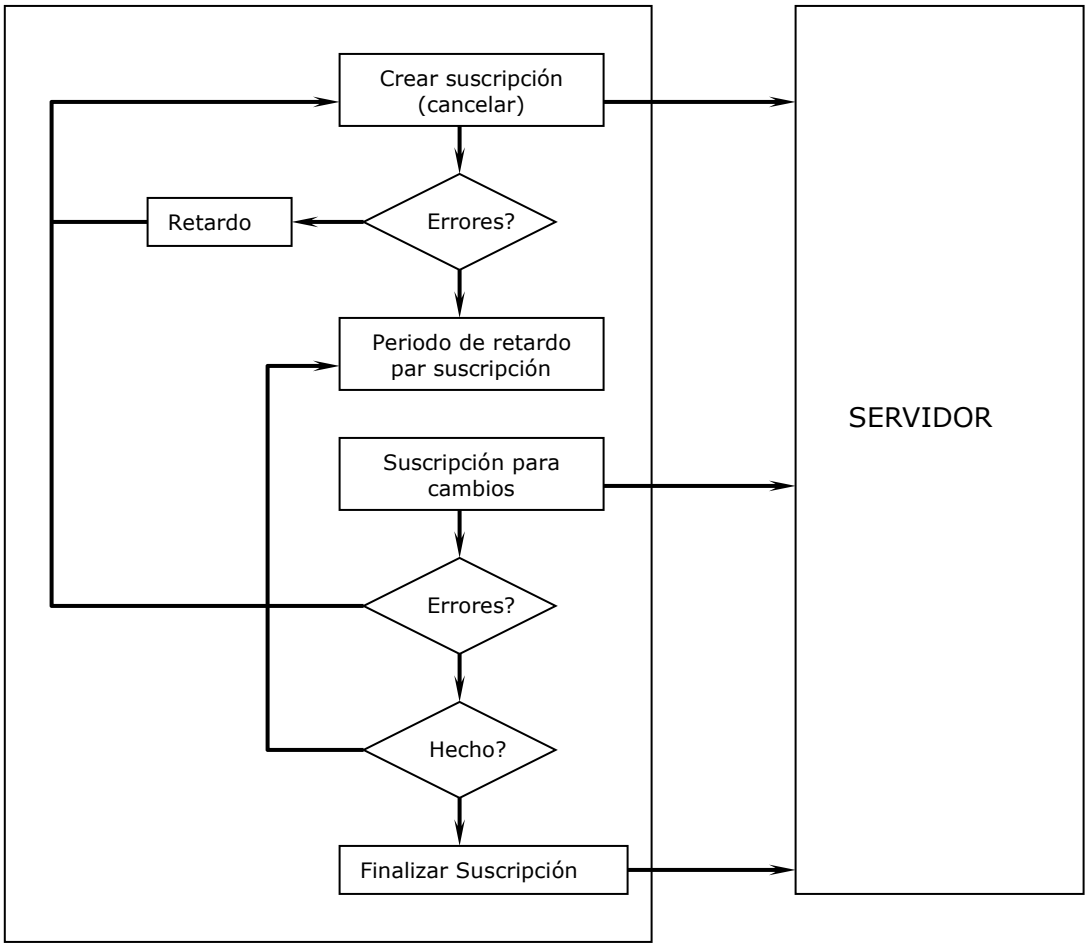

Figura 3.8. Manejo de errores en las Suscripciones

- **Suscripción avanzada "Registrar-Sacar" (Polled-Refresh)**

Para optimizar las respuestas de los servidores y hacerlas mas parecidas al modelo OPC-COM, un usuario avanzado puede utilizar este tipo de aproximación. Esta suscripción utiliza dos tipos de parámetros para actualizar la Suscripción.

  - ***HoldTime:*** Indica al servidor esperar cuando se retorna de una llamada SubscritionPolledRefresh hasta que el tiempo especificado es alcanzado.

- ***Waittime:*** Indica al servidor que espere un tiempo determinado (en milisegundos) después de utilizar que se halla cumplido el *HoldTime,* antes de devolver cualquier reporte de cambio de valor. Este tiempo no es necesario cumplirlo completamente si el servidor entrega los valores antes de completarlo.

Con estos dos métodos aumentan los tiempos de retardo pero se disminuyen los errores y las fallas producidas por tiempos de respuesta demasiado largos.

- **Métodos de XML-DA**

Esta especificación cuenta con unos métodos muy ajustados a las intenciones de la AU propuesta por la OPC: *Browse, GetProperties, GetStatus, Read, Subscribe, SubscriptionCancel, SubscriptionPolledRefresh y Write.*

- ***Browse:*** Busca jerárquicamente los nombres de las variables (TAGs) que se encuentran en el servidor.

- ***GetProperties:*** Retorna información asociada con una o mas TAGs.

- ***GetStatus:*** Retorna información acerca del servidor, tal como: versión, modo actual, estado general, etc.

- ***Read:*** Devuelve el valor, calidad y tiempo de consulta de algunas de las TAGs seleccionadas con el método Browse o dando un nombre específico.

- ***Subscribe:*** Indica una lista de ítems a los cuales el cliente estará conectado por un tiempo determinado.

- ***SubscriptionCancel:*** Remueve la lista de ítems indicada en la suscripción anterior.

- ***SubscriptionPolledRefresh:*** Retorna el valor de todos los ítems indicados en la suscripción anterior que hallan cambiado desde ese momento.

- ***Write:*** Escribe en el servidor uno o mas valores de las TAGs.

Estos métodos y su interacción esta totalmente descrita en la especificación OPC XML-DA 1.0 y 1.01. En este capitulo solo se muestran los conceptos básicos debido a que en el próximo capitulo se describirán detalladamente mediante diferentes tipos de diagramas.

3.6 HERRAMIENTAS DE DESARROLLO

Para lograr los objetivos de este proyecto se realiza una integración de la arquitectura propuesta por la Fundación OPC en su Arquitectura unificada y las herramientas brindadas por Java Sun Microsystems y sus diferentes arquitecturas para manejar Servicios Web (J2EE), eventos y procesos normales en equipos comunes (J2SE), y Aplicaciones móviles (J2ME).

Debido a que la especificación viene orientada hacia herramientas de desarrollo de Microsoft y especialmente a herramientas .NET, no se toman las especificaciones y recomendaciones estrictamente, pero si se lleva a cabo la mejor aproximación posible.

A continuación se presenta un esquema de la configuración general del servicio a implementar:

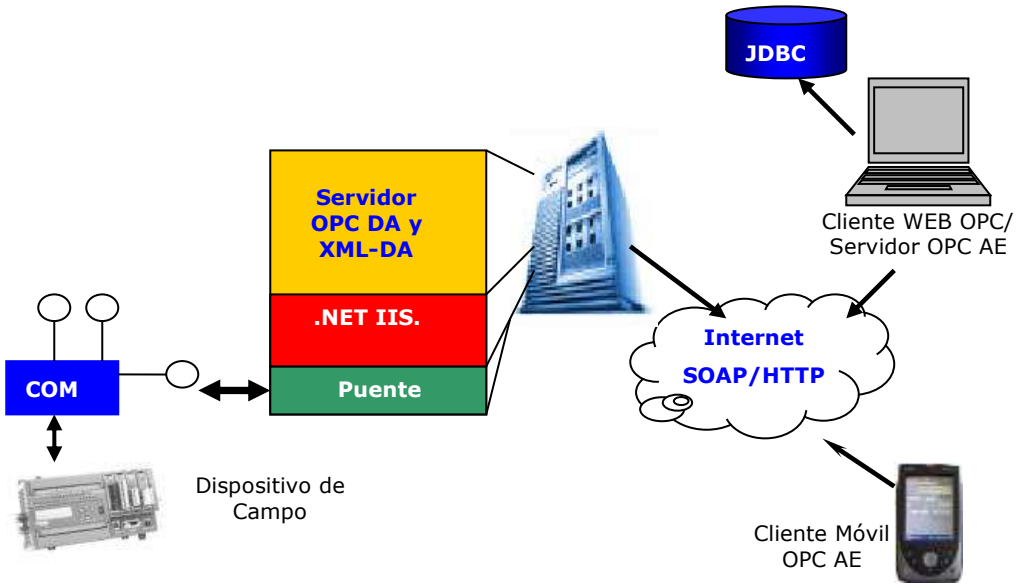

Figura 3.9. Esquema de la configuración general del servicio

En la parte de la izquierda de la figura anterior se puede observar un servidor de datos OPC XML-DA el cual esta montado en una plataforma Windows y bajo el servidor de información en Internet de Windows (IIS). Este se comunica con las aplicaciones y objetos COM por medio de un puente o funcionalidad, pero ninguna de estas dos partes fue desarrollada en el proyecto y son brindadas por herramientas que usualmente se venden junto con los dispositivos de campo que tienen interoperabilidad OPC.

Al lado derecho de la figura se puede observar un cliente Web XML-DA que a su vez es servidor de Alarmas y Eventos OPC, el cual está conectado a una base de datos por medio de JDBC. Este realiza todas sus transacciones utilizando el protocolo SOAP para Servicios Web y HTTP como transporte sobre Internet. Un cliente móvil a su vez está en constante comunicación con el servidor de Alarmas y Eventos a través de servicios J2ME como mensajería corta o servicios WAP. Esta parte será totalmente desarrollada en este proyecto y las herramientas utilizadas para llevar a cabo dicha tarea se presentan a continuación.

3.6.1. Implementación de OPC AE

Para llevar a cabo la especificación de AE propuesta por la Fundación OPC se utiliza la versión estándar de Java J2SE y se emplea especialmente para el manejo de eventos la Clase *Observable* y la interfase *Observer,* para dar acceso a los clientes se utiliza Servicios Web con J2EE.

Como se ha comentado antes, todas las especificaciones de la Fundación OPC van dirigidas a desarrollos en entornos Windows y comunicaciones COM, por esta razón para desarrollar la funcionalidad de un servidor de alarmas y eventos bajo un paradigma multiplataforma solo puede ser desarrollado tomando los conceptos básicos y mas relevantes de la especificación y unirlos a la Arquitectura unificada de la OPC.

Los conceptos más relevantes de la especificación de Alarmas y eventos que se tienen en cuenta en el servidor de alarmas y eventos son:

El servidor de alarmas y eventos debe tener la capacidad y los mecanismos para notificar a los clientes OPC cuando ocurra determinado evento o condición de alarma. También debe proporcionar a los clientes los mecanismos necesarios para determinar los eventos y condiciones soportadas en el mismo servidor y obtener su estado.

Una *alarma* es una *condición* anormal y por consiguiente es un caso especial de una *condición.* Esta condición está directamente relacionada con algún objeto que proporcione información a un cliente OPC, por ejemplo una TAG.
También a una condición se le puede definir (opcionalmente) una sub-condición asociada. Por ejemplo esta sub-condición puede estar asociada con el nivel de la alarma: Alto, medio, bajo, etc.

- ***Servidor de eventos OPC:*** Especifica el tipo de eventos permitido, la suscripción con el cliente OPC, y el mensaje de retorno cuando el servidor se encuentra apagado.

- ***Áreas:*** Es una forma de agrupar las alarmas por conjuntos según el lugar de la planta donde se encuentren.

- ***Condiciones:*** Como se ha explicado antes, una condición es un objeto que puede generar eventos y un estado en particular de la condición es la alarma. Una condición (OPCCondition) tiene siempre asociada una fuente de información (OPCSource) y puede tener asociada una subcondición (OPCSubcondition) tal como se muestra en el siguiente gráfico:

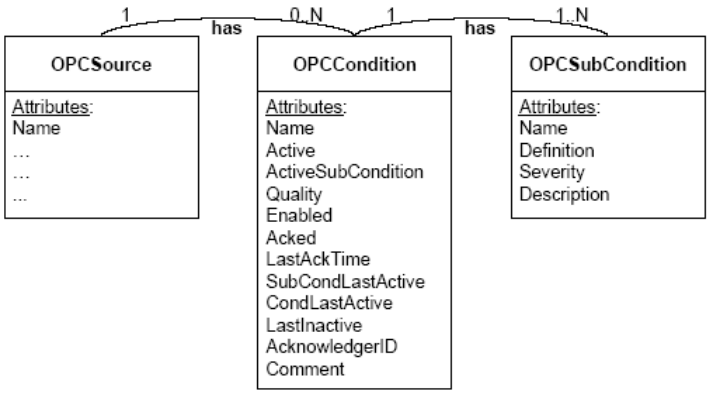

Figura 3.10. Relación entre clases de condición

La condición en si puede tener tres estados: Habilitada (Enabled), Confirmada (Acked), y activa. Para poder tener en cuenta los eventos de esta condición, la condición debe encontrarse habilitada primero que todo. La condición de confirmada o no, se utiliza para tener en cuenta que ya ha pasado por un estado o no, y de esta forma el servidor no mande el mismo evento por estar en el mismo estado sin haber salido de el. El estado activo, es simplemente cuando la condición se encuentra en estado de alarma.

- ***Notificaciones y Eventos:*** Un evento es una ocurrencia detectable que tiene gran importancia para el servidor de alarmas. Una notificación es una respuesta mandada a todos aquellos que están monitoreando dicha variable para que puedan llevar a cabo una respuesta determinada. Esta notificación debe brindar gran parte de la información del estado y de la condición en si, tal como nombre, hora de activación, fuente relacionada, etc.

- ***Suscripción a Eventos y Notificaciones:*** Todo cliente del servidor de alarmas y eventos debe inscribirse al servicio y a las condiciones que quiere monitorear. Debe

ser posible también tener un filtro para que el cliente escoja que tipo de condición desea utilizar con su alarma, tal como tipo de evento, inclusión de rango de prioridad, delimitación de áreas del proceso, etc.

- ***Manejo de errores:*** Errores de parte del servidor o acceso a la fuente de datos, o errores por parte del cliente.

Teniendo en cuenta estos conceptos fundamentales en la especificación, se utilizan dos herramientas fundamentales para la implementación del servicio: Supervisión y notificación de eventos de tipo "Observador y objeto Observable" con la interfase *Observer* y con la Clase *Observable* las cuales permiten generar un evento cada que se genera un cambio en la clase *observable* y como respuesta a esto generar una notificación a cada uno de los objetos que se hallan suscrito para supervisar dicha condición y los cuales son de tipo *Observer.* Y el segundo concepto es el de Servicio Web para el acceso al servicio desde cualquier Terminal, incluyendo dispositivos móviles. Para este último se debe utilizar la plataforma J2EE, XML y preferiblemente SOAP para facilitar la interacción y acople con el paradigma de Arquitectura Unificada propuesto por OPC.

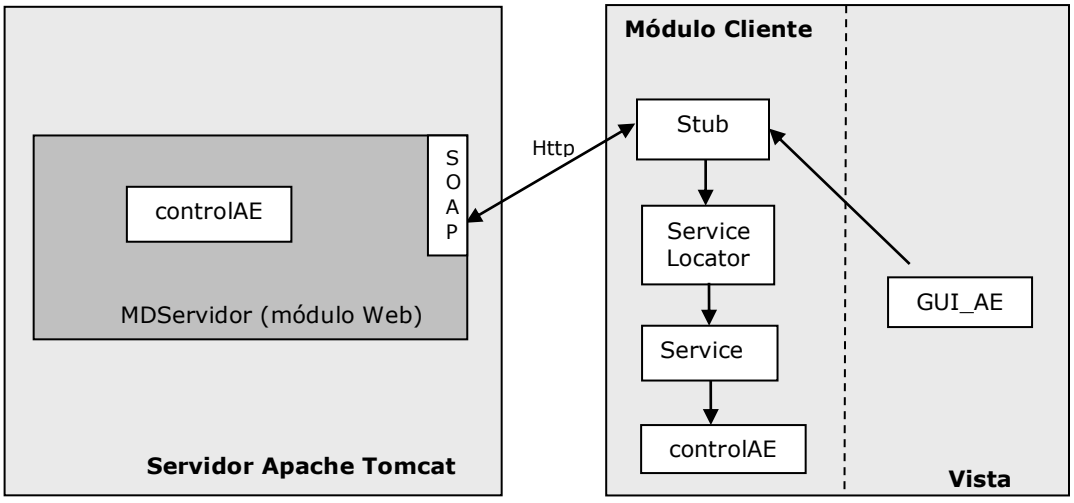

Figura 3.11. Arquitectura de Gestión de Alarmas y eventos en Java.

Como se puede apreciar en la figura básicamente se manejan dos herramientas por parte del servidor de alarmas: Servidor Apache Tomcat y Axis, los cuales son el núcleo de las aplicaciones Web que serán desarrolladas.

- **AXIS**

El kit de herramientas de Axis de Apache es una implementación de código abierto de SOAP, la siguiente generación de Apache SOAP 2.0. Axis es una nueva versión de Apache SOAP 2.0 que utiliza SAX en vez de DOM (APIs de JAVA J2EE para manejo de XML). Se trata de una implementación de SOAP más modular, flexible y de mayor rendimiento que Apache SOAP 2.0. El kit de herramientas de Axis de Apache es compatible con JAX-RPC (API Java para llamada a procedimiento remoto basada en XML) y admite WSDL 1.1.

JAX-RPC define bibliotecas API de Java que los desarrolladores pueden utilizar en sus aplicaciones para desarrollar y consumir servicios web. Mediante JAX-RPC, un cliente Java puede consumir un servicio web que reside en un servidor remoto a través de Internet, incluso aunque el servicio esté escrito en otro idioma y se ejecute en una plataforma diferente. Los clientes que no sean de tipo Java también pueden consumir servicios JAX-RPC.

JAX-RPC utiliza un protocolo de mensajería XML, como SOAP, para transmitir una llamada a procedimiento remoto a través de una red. Por ejemplo, un servicio web de XML-DA que devuelve el valor de determinada TAG recibiría una solicitud HTTP SOAP con una llamada a un método realizada desde el cliente. Mediante JAX-RPC, el servicio extrae del mensaje SOAP la llamada al método, la traduce convenientemente y realiza esa llamada. A continuación, el servicio utiliza JAX-RPC para convertir la respuesta al método otra vez en un mensaje SOAP y enviar el mensaje de vuelta al cliente. El cliente recibe el mensaje SOAP y utiliza JAX-RPC para traducirlo en una respuesta.

El entorno de ejecución JAX-RPC genera stubs y ties, que son clases que permiten la comunicación entre el cliente y el servicio. Un stub, que reside en el cliente, es un objeto local que representa un servicio remoto y actúa como proxy para el servicio. Un objeto tie, que reside en el servidor, actúa como un proxy en el servidor.

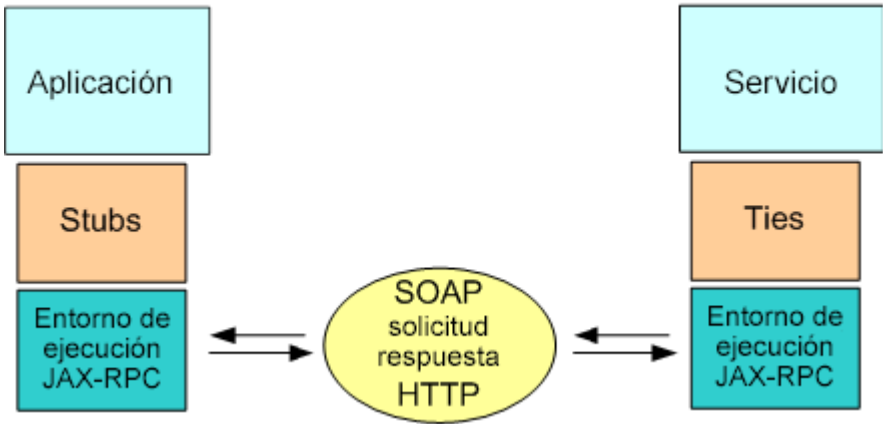

Figura 3.12. Manejo de stubs y Ties con JAX-RPC[4]

- **Apache Tomcat:**

Apache Tomcat es un servidor Web que en su proyecto Jakarta Tomcat incluye una implementación de código abierto de las especificaciones Java Servlet y JavaServer de J2EE.

Los Servlets son piezas de código escritas en Java que se ejecutan en un servidor Web y añaden funcionalidades al mismo, pero sin interfaz gráfica. Estos están diseñados para soportar un modelo pregunta/Respuesta más usado en el mundo Web (Petición/Respuesta)[5]. Estos proporcionan una forma de construir código HTML dinámico, permitiendo a múltiples usuarios recibir múltiples respuestas y a un servidor recibir peticiones simultáneas.

[4] JAX-RPC, consulte http://java.sun.com/xml/jaxrpc/index.html.
Tutorial de JAX-RPC disponible en http://java.sun.com/webservices/docs/1.0/tutorial/doc/JAXRPC.html.

[5] O. Caicedo, D. López. Electiva Aplicaciones Web, Universidad del Cauca.

3.6.2. Cliente Móvil de Alarmas y Eventos

Dado a que la misma idea de Arquitectura Unificada surgió después de pensar en integrar la especificación OPC AE con una aplicación móvil y a la funcionalidad de los dispositivos móviles para gestionar el mantenimiento del proceso, se incluye en el proyecto un modulo de gestión para usuarios móviles, con el cual se ratificará la capacidad multiplataforma propuesta para el mismo.

Para poder realizar esta gestión bajo la plataforma Java, se deben tener algunos conceptos como los son el de Servicios Web (explicado con anterioridad) y algunos otros de J2ME como lo son *MIDLET, WAP, SMS* Y *KSOAP*.

J2ME es una versión para pequeños dispositivos móviles de consumo como PDAs, celulares e incluso electrodomésticos inteligentes. Sus servicios están basados en programas locales (Midlets), que el usuario descarga dinámicamente en el dispositivo y los cuales se ejecutan en el dispositivo sosteniendo una comunicación cliente servidor.

En realidad J2ME es una versión mucho más reducida que J2SE y con algunas modificaciones que permiten adaptar los dispositivos a dichas funcionalidades.

Característica	J2SE	J2ME
Inicio de aplicación	Main	startApp
Recolección de basura	garbageCollector	No
Tipos de datos Float y Doble	Si	No
GUI	Swing y AWT	LCDUI
Preverificación	On - Line	Off – Line
Descriptor	No	Si
Manifiesto	No	Si

Tabla 3. J2ME vs. J2SE[6]

En la tabla anterior se pueden ver algunas de las variaciones de una aplicación J2SE a una aplicación J2ME. Entre las funcionalidades que no tiene J2SE esta el *Manifiesto*, el

[6] O. Caicedo, Desarrollo de aplicaciones móviles, Universidad del Cauca.

cual se utiliza para describir el contenido de los archivos *.jar* y de los *Midlets* de la aplicación, y el *Descriptor*, cuya funcionalidad es tener los requisitos y especificaciones de los dispositivos que podrán utilizar el *Midlet.* Además de estas diferencias, también sobresalen las diferencias entre arquitecturas de máquinas virtuales de J2SE a J2ME, como se puede observar en la siguiente gráfica.

Figura 3.13. Diferencias entre JVM de J2ME, J2SE y J2EE

Mientras que una aplicación J2EE o J2SE tiene la tradicional JVM, una aplicación Web puede tener una CVM (C Virtual Machine) o una KVM (K Virtual Machine). La primera es una implementación de referencia escrita en lenguaje C orientada a dispositivos electrónicos, la segunda es una implementación de JVM reducida especialmente orientada a dispositivos con bajas capacidades.

- **Midlet**

Las aplicaciones que se desarrollan en J2ME y que implementan la especificación MIDP para dispositivos móviles, se denominan *Midlets.* Estos *Midlets* se agrupan en ficheros de tipo *.jar* (formato de contenedores Java) para permitir portabilidad de código, es decir, para poderse ejecutar en cualquier otro dispositivo con las mismas características. Otra gran ventaja es poderse descargar desde cualquier parte de la red móvil, lo cual trae muchas ventajas para sus usuarios.

- **KSOAP (Protocolo para acceso a objetos simples en KVM):**

KSOAP, es la API que da soporte al protocolo SOAP para la plataforma J2ME, basado en KXML, tanto KXML como KSOAP están diseñados para permitir que aplicaciones SOAP y XML se ejecuten en la KVM. KSOAP contiene una clase genérica llamada SoapObject que permite a las aplicaciones inalámbricas construir peticiones SOAP. Además, proporciona la clase HttpTransport, que funciona como bypass de los mensajes SOAP dentro de una aplicación Java, permitiendo a través del método HttpTransport.call() el transporte real de los datos del cliente al servidor y recibir la respuesta. Con KSOAP se habilita a un cliente MIDP acceder a un Servicio Web.

- **WAP:**

WAP, es un estándar de facto para la presentación, envío de información y utilización de servicios adicionales de telefonía sobre dispositivos móviles y otros terminales inalámbricos. WAP nace como producto de un proceso de convergencia entre Internet y los sistemas inalámbricos que comenzó a gestionarse en 1997 cuando Nokia, Motorota, Ericcson y Unwired Planet ahora Phone.com crearon el WAPForum, una organización que tiene como objetivo habilitar sofisticados servicios de información y telefonía a dispositivos móviles. A diferencia de las tecnologías de Internet para PCs, WAP está pensado para dispositivos que tienen algunas limitaciones técnicas inherentes a la tecnología actual tales como:

- Menor potencia de procesamiento
- Menor capacidad de memoria
- Restricciones de suministro de potencia
- Despliegues pequeños
- Mecanismos de entrada diferentes

Los componentes involucrados en las aplicaciones WAP son muy similares a los del World Wide Web (WWW), excepto por la incorporación de una pasarela utilizada para servir de intermediario entre el mundo inalámbrico e Internet.

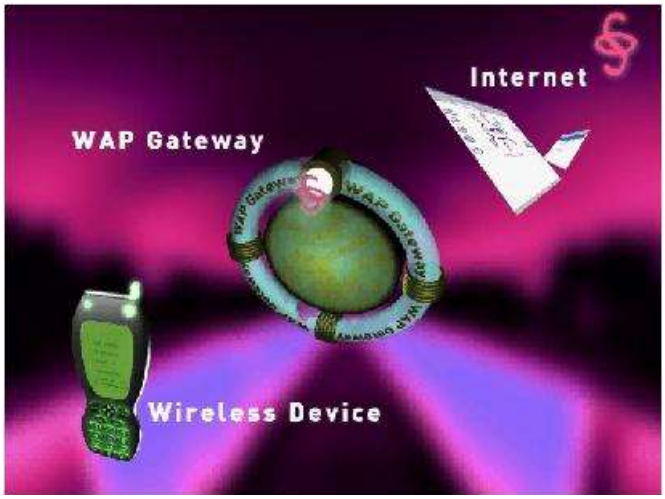

Figura 3.14. Modelo de Implementación WAP

Inicialmente el lenguaje empleado para crear interfaces de usuario en WAP fue WML (Wireless Markup Language) el cual constituía una especialización de XML para dispositivos móviles. Actualmente, la especificación WAP 2.0 establece como lenguaje a XHTML (eXtensible HTML), un lenguaje que extiende la funcionalidad de HTML.

- **SMS (Servicio de mensajería corta)**

El servicio de mensajería corta es un servicio inalámbrico aceptado globalmente que habilita la transmisión de mensajes alfanuméricos entre suscriptores móviles y sistemas externos como e-mail, mensajes escritos y sistemas de correo de voz.

El SMS punto a punto proporciona un mecanismo para transmitir mensajes cortos desde y hacia dispositivos inalámbricos haciendo uso de un centro de mensajería corta (SMSC), el cual actúa como un sistema de almacenamiento y reenvía para dichos mensajes. La red celular proporciona el transporte para los mensajes cortos entre los SMSCs y los dispositivos de mano. Una característica importante de este servicio es que un móvil está habilitado para recibir o enviar mensajes cortos en cualquier momento, independientemente de si se está haciendo o no una llamada de voz o datos. Debido a que SMS también garantiza la entrega del mensaje corto mediante la red, se identifican las fallas temporales y el mensaje será almacenado en el sistema hasta que el destino se encuentre disponible.

3.6.3. Implementación de Cliente OPC XML-DA

Al ser XML-DA una especificación orientada hacia los Servicios Web, básicamente se utilizan las mismas herramientas descritas del lado del Cliente en el servidor de alarmas, solo se debe tener en cuenta que .NET no cumple con la especificación de WSDL para servicios Web, lo cual puede generar algunos errores en la generación de los métodos en el cliente, para lo cual se recomienda corregir el código generado manualmente.

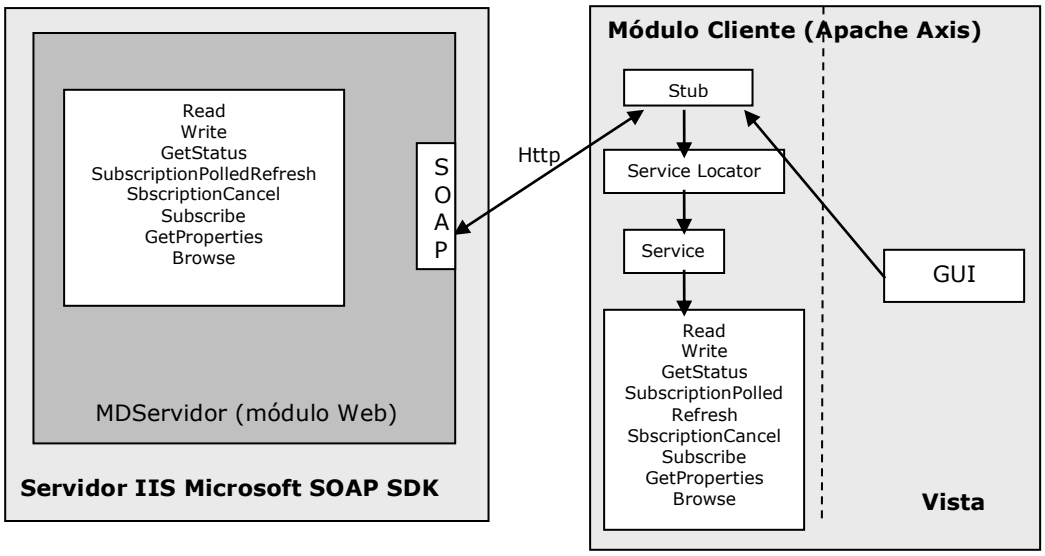

Figura 3.15. Cliente XML-DA en Java.

Para desarrollar el cliente se utiliza la WSDL dada por la Fundación OPC en su especificación y se utiliza Apache Axis para generar las clases y stubs pertinentes.

Al utilizar esta herramienta se generan tres clases: Un *stub* que es con la cual el cliente serializa en SOAP los métodos y las peticiones al servidor, un *Service Locator* que es la implementación abstracta del servicio y un *Service* que es una interfaz abstracta que define una clase de fábrica para obtener una instancia del stub. Estos métodos son explicados con más detalle en el siguiente capitulo.

3.6.4. Manejo de Bases de datos con JDBC (Java DataBase Conectivity)

Indudablemente, las Bases de Datos, en la actualidad son imprescindibles para cualquier sistema que requiera desde una gestión de usuarios, hasta el almacenamiento redundante de información relevante de la empresa. La importancia e impacto de las bases de datos es incuestionable a medida que organizaciones gubernamentales, instituciones académicas, y entidades comerciales crean y mantienen grandes cantidades de información, desde documentos de texto en lenguaje natural, tablas estadísticas, datos financieros y objetos multimedia, hasta datos de naturaleza técnica y científica, los cuales necesitan ser almacenados y administrados adecuadamente.

Los procesos y las redes industriales, como gran negocio que son, manejan gran cantidad de datos almacenados, y por lo tanto requieren de una gestión eficiente de los mismos.

Como ya se ha repetido muchas veces en este documento, a pesar de que la Fundación OPC a intentado crear una arquitectura abierta, ha basado todos sus desarrollos en tecnologías Windows, y por lo tanto al tratar de migrar a otros entornos como UNIX con Java, se deben hacer algunos cambios. En este caso, es imprescindible el manejo de bases de datos con conectores de tipos diferentes a los manejados por OPC-HDA, los cuales se basan en OLE DB, un manejador orientado a objetos tipo COM/DCOM. Normalmente cuando se quiere hacer una conexión a un motor de Bases de Datos se utiliza el API JDBC.

JDBC es un estándar para acceder a cualquier tipo de datos disponible en el mercado, ya sea de carácter libre o no. Esta API está formada por un conjunto de clases e interfases desarrolladas en Java, para ejecutar sentencias SQL, permitiendo obtener los datos de una forma fácil y segura en ambientes Cliente/Servidor a través de Internet o Intranet.

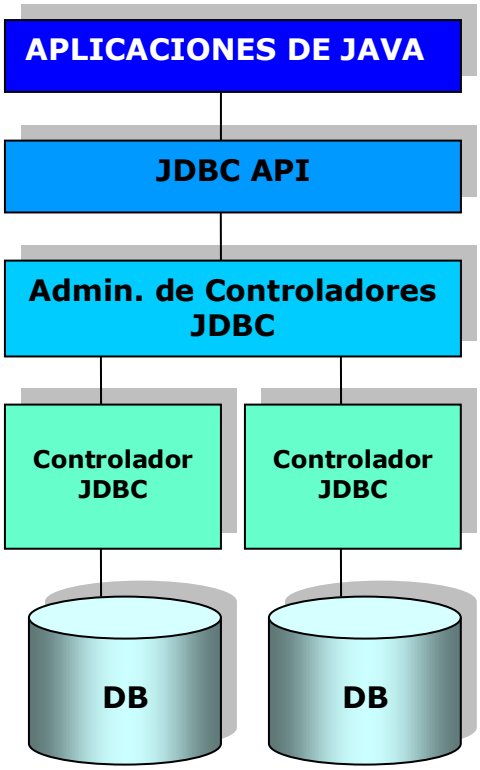

Figura 3.16. Arquitectura JDBC

Básicamente en su arquitectura JDBC tiene dos capas: la primera capa usualmente es un conector o controlador a una Base de Datos determinada que permite dar a el motor de la base de datos las sentencias SQL; la segunda es una API desarrolladora con métodos y atributos que facilitan la extracción y manejo de datos desde una base de datos a una aplicación cualquiera. Usualmente se anexan otras capas a la arquitectura, como lo son el administrador de controladores cuando se tienen varias Bases de Datos y de tipos diferentes, o un puente que permite realizar una mejor conexión entre el motor de la base de datos, y el controlador de Java de la misma.

Un JDBC establece una conexión a Base de datos, después envía las sentencias SQL al motor de la misma, y después procesa la respuesta recibida.

La aplicación a ser utilizada se denomina cliente/servidor, en la cual la misma "habla" directamente con la base de datos, después los controlador JDBC se comunica con el

sistema específico que maneja la base de datos pudiendo estar en otra máquina, con lo que el cliente se comunica por red.

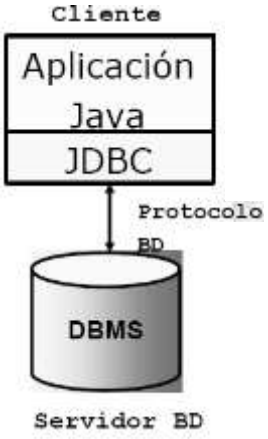

Figura 3.17. Modelo Cliente/Servidor con JDBC

JDBC permite que se escriba cualquier tipo de sentencia SQL. Aunque ésta fuera dependiente de la base de datos sólo se correría el riesgo de incompatibilidad al cambiar de base de datos. También soporta llamadas a procedimientos almacenados y formatos de fecha.

Una vez se obtienen los resultados, la forma de obtener los mismos en la aplicación es muy simple y utiliza en sus métodos sintaxis muy obvia que facilitan el proceso al programador.

3.7 FUTURAS ESPECIFICACIONES XML-DA

La especificación de XML-DA es la primera de una larga línea de especificaciones de Servicios Web que serán realizadas por OPC. Las otras áreas funcionales dirigidas por las otras especificaciones OPC-COM (Alarmas y Eventos, HDA, Batch, etc.) necesitarán ser llevadas al mundo de los Servicios Web. Al igual que con XML-DA, esas nuevas especificaciones permitirán wrappers de manera que la base existente de los servidores COM puedan ser actualizados instantáneamente. Actualmente se esta utilizando la especificación 1.0 publicada en Julio 12 de 2003, y el 18 de Diciembre de 2004 fue publicada la versión 1.01, la cual trae como principales cambios:

- Se describe de una forma más detallada la conversión de datos.
- Se agregó un nuevo tipo de error: E_BADTYPE.
- Se prohíben las conversiones de tipo de datos por parte del servidor cuando se está realizando una transacción de escritura.
- Se clasifica el dato "item value" como de solo escritura.

3.8 DESEMPEÑO DE REDES INDUSTRIALES EN ENTORNOS NO COMUNES

Para la industria, el vincular los procesos a los Servicios Web, y en general exponerlos a la Internet, es un gran riesgo que no hace mucho se ha venido implementando. El uso de estos servicios y de la Red de Redes trae muchas ventajas para quienes la utilizan, pero trae inherentemente muchos problemas especialmente de seguridad. El hecho de exponer los datos de un proceso y el poder leerlos y sobrescribirlos por medio de una red que esta abierta a "todo el mundo" es una de las decisiones que puede tomar una empresa para tomar la delantera en el mercado. Otro gran problema para el sector industrial y de producción son los tiempos de retardo y en general la validez de los valores en un contexto determinado, o lo que se ha llamado en telecomunicaciones y electrónica respuestas en tiempo real debido a que en La Internet es imposible mantener respuestas de tiempo real de carácter duro o estricto (hard real time requests) pues redes de este tipo dependen de muchos factores como el tráfico de la red, configuración de las redes, y medios de transmisión.

Sin embargo al hacer la relación costo beneficio, estas nuevas tecnologías pueden llegar a sorprender. Además de esto, este tipo de arquitecturas de red pueden ser optimizadas a tal punto que la probabilidad de falla llega a valores mínimos. Por ejemplo la parte de seguridad ya esta muy desarrollada y Java especialmente trae muchas herramientas para proteger la información independientemente de un Firewall. Existen arquitecturas y modelos de protección de la información para bases de datos y trasporte en protocolos como HTTP que garantizan a los usuarios en casi un cien por ciento la fiabilidad de la conexión.

Por otro lado, para manejar los retardos y los tiempos de retorno (Round Treaps) generados por la arquitectura y la red, se hace imprescindible una estandarización en el tema. El problema radica en la relación tamaño del paquete de datos solicitado y el tiempo dado para su transporte. Cuando se hace la petición de un dato simple y pequeño, por ejemplo el valor simple de una TAG, se consume mucho tiempo y se obtiene muy poca información, además por la arquitectura de la red, se da menos prioridad a los paquetes que contienen menor información.

Para solucionar estos problemas, se han planteado independientemente algunas soluciones, por ejemplo algunos han planteado la posibilidad de clasificar la información en dos tipos: uno en donde los pedidos sean de gran tamaño y en donde un tiempo de respuesta sea aceptable (la petición de los valores de una lista de Alarmas en un área determinada en su conjunto) y otro para cuando los pedidos son pequeños y es necesario una rápida respuesta (la petición de estado de servidor). Para el primer caso, es viable una solución basada en Servicios Web, mientras que en el otro es muy difícil que el desempeño sea aceptable, y sería mejor tratar de encontrar una solución diferente dependiendo de la prioridad de la transacción.

4. DISEÑO DETALLADO DELSISTEMA

Para un adecuado desarrollo del proyecto y considerando que éste se basa fundamentalmente en el desarrollo de soluciones software se utiliza una metodología que favorece la construcción del sistema con características óptimas de calidad.

Dentro de las diferentes opciones, consideramos que el Modelo de Construcción de Soluciones (MCS), desarrollado dentro del grupo de ambientes de Desarrollo de la Facultad de Ingeniería Electrónica y Telecomunicaciones (FIET) de la Universidad del Cauca, siendo una buena referencia para tomar como guía en el desarrollo del proyecto, aunque no se desarrollarán todas las fases que en él se plantean, si no las que se consideren adecuadas para el desarrollo del sistema. Además, hay que tener en cuenta que debido a las características del proyecto, el MCS se puede complementar con especificaciones que describan de manera más clara el sistema que se pretenda implementar.

En este capitulo se desarrollan especialmente las fases de *Análisis y Diseño,* la parte de implementación se puede obtener en el CD anexo, con el código fuente de la aplicación. La parte de pruebas se presenta en el siguiente capitulo como validación del sistema.

4.1 CARACTERÍSTICAS DEL SERVICIO

Para desarrollar el servicio se tienen en cuenta dos tipos de clientes: el primero fue un usuario frecuente del proyecto *ARIADNA*[7] quien maneja de manera constante un sistema SCADA RSView de Rockwell Automation, y el segundo fue el Ingeniero Oscar A. Rojas, quien actualmente es el encargado de enseñar el manejo de sistemas SCADA

[7] Adquisición Remota de Información Ambiental para Diagnóstico y gestión de recursos Naturales. Códigos VRI 139 (2000) y 524 (2001).

y redes Industriales en la Universidad del Cauca para los programas de Ingeniería Electrónica y Telecomunicaciones e Ingeniería Automática Industrial. Después de entrevistar a los clientes en repetidas ocasiones, los requerimientos obtenidos son:

Con Relación al cliente y Acceso Directo (DA):

- Agregar y eliminar Datos (Tags del proceso).
- Obtener Estado y datos del servidor Automáticamente, es decir, poder desplegar en forma de árbol los servidores que estén disponibles en ese momento en la red industrial.
- Desplegar los datos (Tags) en forma de árbol (Si el servidor lo permite).
- Modificar y leer Datos (Cuando el servidor lo permita).
- Obtener propiedades de los datos tales como calidad, escritura, velocidad, etc.
- Desplegar en una misma pantalla la información de varios dispositivos en la misma Red.
- Pasar datos de un dispositivo a otro.

Con Relación al manejo de datos Almacenados (HDA)

- Acceder al historial de datos dando unos parámetros determinados, como pueden ser fecha inicial, fecha final, y atributo a buscar.
- Validar los datos que se pueden modificar.
- Filtrar el tipo de datos que se pueden incluir en la base de datos. Es decir, solo podrán ser incluidos en la BD, los datos del tipo que se determinen. Por ejemplo: contadores, valores finales, pero nunca valores de temporizadores.
- Crear y eliminar Bases de datos según sus permisos generados en la validación.

Con Relación al cliente Móvil y el manejo de alarmas:

- El sistema debe estar en la capacidad de utilizar un evento generado por la alarma para avisar mediante una aplicación móvil y por medio de mensajería a cualquier persona o personas a cargo de su supervisión. Es decir que varios monitores de alarma podrán recibir estos mensajes si están registrados a supervisar la misma alarma.

- Después que el usuario reciba el mensaje podrá utilizar un servicio WAP con interfaz amigable para obtener toda la información proporcionada por el servidor acerca de las alarmas.

En general debe haber un sistema de validación jerárquico, y la arquitectura debe ser orientada a un modelo Productor/Consumidor; es decir, cuando haya un cambio en los datos, se debe generar un evento que comunique a los usuarios disponibles para que aquellos que necesiten la nueva información la tomen e inicien las acciones necesarias según su función.

4.2 MODELO INICIAL DEL NEGOCIO

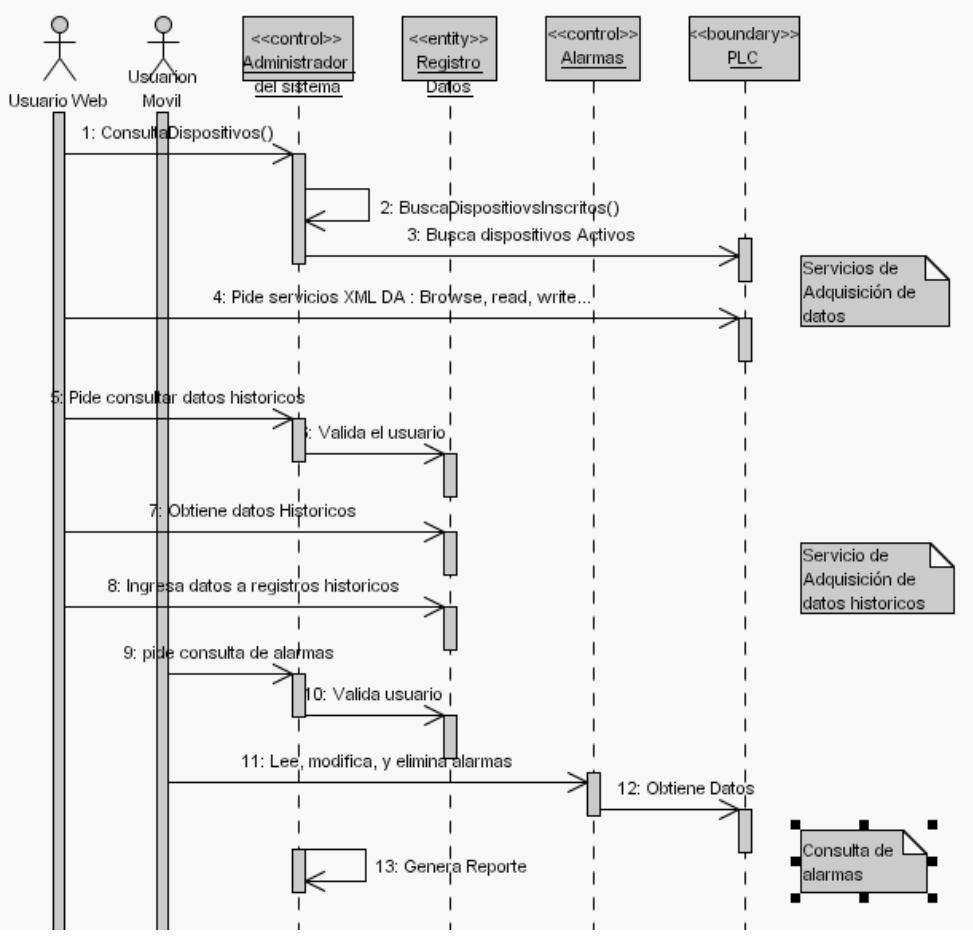

Figura 4.1. Modelo del negocio

4.3 MODELO INICIAL DE CASOS DE USO DEL SISTEMA

4.3.1. Diagrama de Casos de Uso

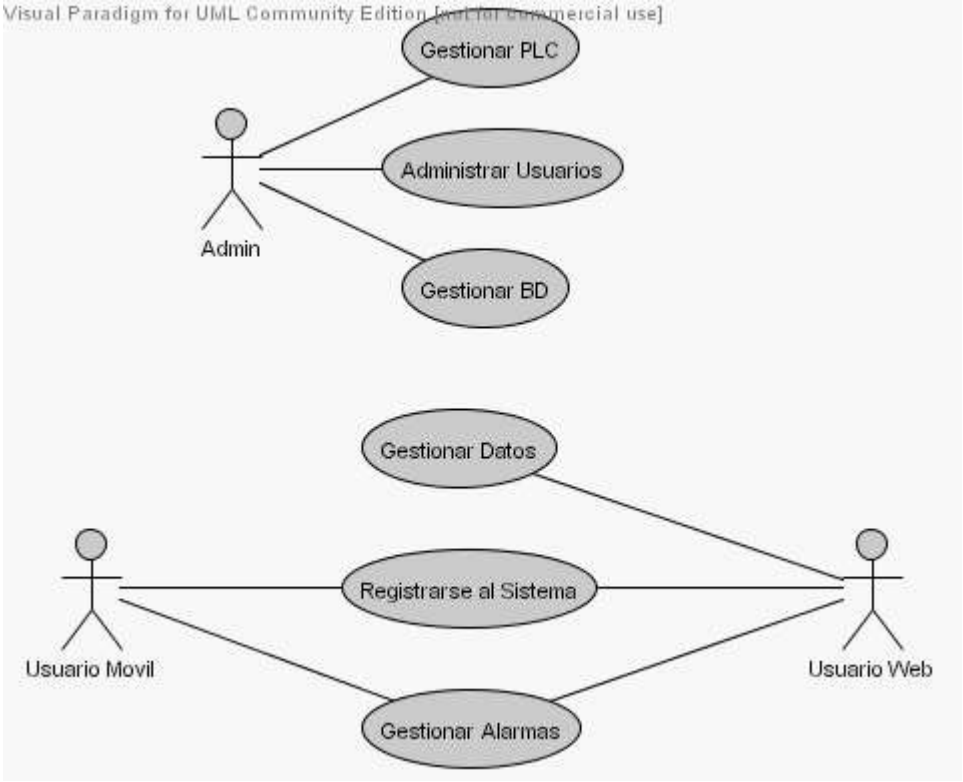

Figura 4.2. Diagrama de Casos de uso Inicial

4.3.2. Descripción Inicial de los escenarios de los Casos de Uso Esenciales

Caso de Uso: Gestionar PLC

Iniciador: Administrador

Propósito: *Manipular de forma local las variables del proceso y el estado de este.

Resumen: El Administrador es el encargado de configurar la red de dispositivos y PLCs, es decir, debe poder de una forma manual y/o automática registrar los dispositivos que están inscritos en la red, publicarlos en algún sitio para que el Usuario Web o Móvil puedan descubrirlos. Además debe gestionar algún tipo de permisos o seguridad para los mismos.

Caso de Uso: Administrar Usuarios

Iniciador: Administrador

Propósito: Gestionar el acceso al sistema

Resumen: El administrador se encarga de permitir o negar el acceso a los usuarios remotos tanto por vía Web, como por vía móvil, además de agregar o eliminar un usuario en cualquier momento.

Caso de Uso: Gestionar Base de Datos

Iniciador: Administrador

Propósito: Permitir acceso total a la BD

Resumen: El administrador está en la capacidad de hacer modificaciones directamente a la Base de Datos, por cuestiones de mantenimiento, o para cuestiones de adición o eliminación de campos o datos en la BD.

Caso de Uso: Gestionar Alarmas

Iniciador: Usuario Móvil, Usuario Web

Propósito: Realizar gestión sobre las Alarmas del proceso.

Resumen: El usuario remoto (móvil o Web) puede gestionar los estados, condiciones y prioridades de las Alarmas de manera remota a través de su dispositivo móvil o a través de Internet.

Caso de Uso: Registrarse al Sistema

Iniciador: Usuario Móvil, Usuario Web

Propósito: Identificarse ante el sistema

Resumen: El usuario en cualquiera de sus dos condiciones (Web, Móvil) deberá registrarse al sistema primero para poder acceder a este.

Caso de Uso: Gestionar Datos

Iniciador: Usuario Web

Propósito: Realizar gestión sobre los Datos Relevantes del proceso

Resumen: El usuario Web puede modificar datos relevantes del proceso, como son Rangos de las variables monitoreadas, entre otros, así como ver el estado actual de estos.

4.4 ARQUITECTURA INICIAL DEL SISTEMA

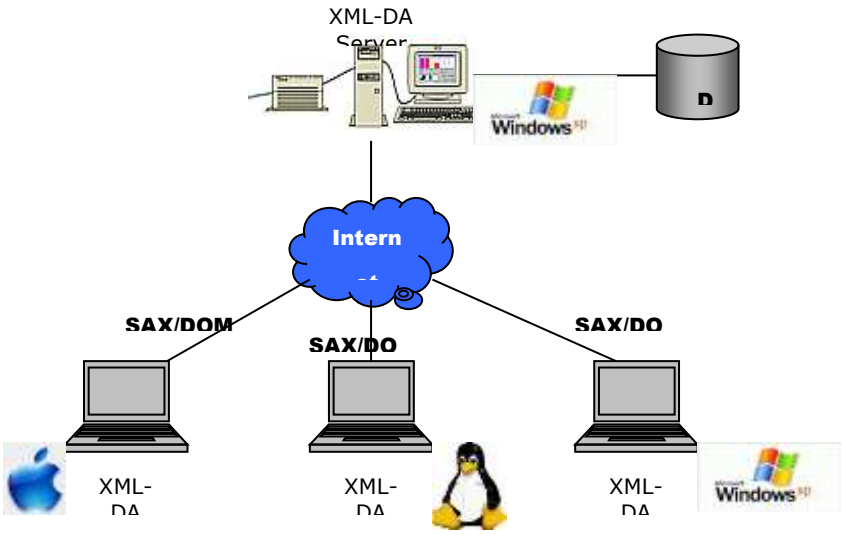

Figura 4.3. Arquitectura inicial del sistema

En la figura anterior se pueden apreciar las características básicas del sistema, que son interoperabilidad entre diferentes plataformas basadas en herramientas Java como lo son SAX y DOM, accediendo a Internet mediante el protocolo SOAP en clientes OPC XML-DA y con un servidor que accede a los datos del PLC o dispositivo de campo en una plataforma estándar Windows.

4.5 VIABILIDAD DE LA ARQUITECTURA INICIAL

4.5.1. Lista de Riesgos

Entorno de ejecución:
RE1. Modelo de comunicación de objetos
RE2. Lenguaje de programación.
RE3. Servidores DA asequibles en las marcas de PLC disponibles

Riesgos tecnológicos:
RT1. Velocidad de respuesta

RT2. Obtención de equipos de trabajo a tiempo.

RT3. Dispositivos de diferentes marcas para validar el proyecto.

Riesgos de desarrollo:

RD1. Compatibilidad de dispositivo y sistema

RD2. Manejo de Arquitectura OPC estándar DA, HDA, EA, XML-DA

RD3. Manejo de programación para Dispositivos móviles.

4.5.2. Registro de gestión de riesgos

ID	DESCRIPCIÓN	IMPACTO	ALTERNATIVAS
RE1	No se han manejado puentes para DCOM / Java, o DCOM / Linux y algunos no son de carácter libre	Crítico: Fracaso en los objetivos principales de hacer el sistema multiplataforma, o por lo menos aumento en los costos.	Hacer un estudio de herramientas existentes de carácter libre, y comercial. Además validar la aplicación con un prototipo inicial.
RE2	La especificación OPC viene con soporte para Visual Basic y C++, pero no para otros lenguajes manejados por los desarrolladores del grupo.	Retardo en las implementaciones de los prototipos	Realizar prototipos de aprendizaje
RE3	La mayoría de los sistemas industriales dan soporte solo para sistemas Windows, por lo tanto no desarrollan clientes ni servidores OPC para entornos diferentes, y los recursos físicos son limitados, así que se reducen las posibilidades de encontrarlos	Critico: Fracaso en desarrollo de los principales objetivos	Realizar prototipos de aprendizaje
RT1	El sistema es un sistema de control, por lo tanto debe tener la mejor velocidad tanto conectado directamente al dispositivo, como	Ineficiencia del sistema.	Análisis de todas las posibilidades de arquitectura, escogencia de la misma teniendo en cuenta este parámetro.

	en la red		Optimización del sistema
RT2	Al utilizar recursos universitarios, estos están superpuestos a retrasos y demoras o una gran cantidad de trámites.	Retrasos en las entregas.	Utilización de equipos propios, negociación con el cliente.
RT3	En el departamento al cual estamos asociados, no se encuentran dispositivos de diferentes marcas para hacer pruebas de compatibilidad.	Inconsistencia y fallas en el sistema.	Buscar empresas o proyectos a nivel local con necesidades parecidas; Utilizar emuladores de sistemas no compatibles.
RD1	Las herramientas que se están desarrollando son de última generación, y el S.W. supervisorio que se tiene en Universidad esta un poco desactualizado, no se sabe si será compatible con algunas herramientas que se basan en especificaciones muy nuevas.	Retrasos en la entrega	Buscar herramientas disponibles para los dispositivos; en caso de no encontrarlas, desarrollar una.
RD2	OPC es una especificación y nunca se ha trabajado por parte de este grupo	Retrasos en la entrega	Desarrollo de tareas de aprendizaje
RD3	No se tienen conocimientos especializados en el área de dispositivos móviles	Retrasos en la entrega	Desarrollo de una aplicación por parte de un grupo aparte; puesto que el manejo de estas herramientas solo se utilizan para validación y no como parte fundamental del proyecto

4.6 LISTA DE PRIORIZACIÓN DE CASOS DE USO DEL SERVICIO

Para realizar la priorización de casos de uso se utilizan unos indicadores con los cuales los evalúan de cero a cinco de acuerdo al criterio del analista a cargo. Los criterios tenidos en cuenta son los riesgos anteriormente mencionados, que necesitan ser disminuidos o anulados según su importancia.

Algunos otros factores que se tuvieron en cuenta para la priorización de los casos de uso fueron:

- El caso de uso es la entrada de otro caso de uso
- El caso de uso que involucra la interacción de varios actores
- El caso de uso interactúa con sistemas ajenos a la aplicación que se vaya a construir

Caso de Uso	**Riesgo de entorno de ejecución**	**Riesgos Tecnológicos**	**Riesgos de desarrollo**	**Otros factores tenidos en cuenta**	**Total**
Gestión De PLC	3	4	2	4	**13**
Gestión de BD	1	0	1	0	2
Administrar Usuarios	1	0	0	0	1
Gestionar Datos	5	4	4	4	17
Gestionar Alarmas	3	2	2	2	**9**
Registrarse al sistema	1	0	0	0	1

Tabla 4. Priorización de Casos de Uso

Con lo anterior se puede concluir que los casos de uso prioritarios para el proyecto son:

- Gestionar Datos
- Gestión de PLC
- Gestionar Alarmas.

4.7 EXTENSIÓN DEL MODELO INICIAL DE CASOS DE USO DEL SISTEMA

4.7.1. Diagrama de Casos de Uso Extendido

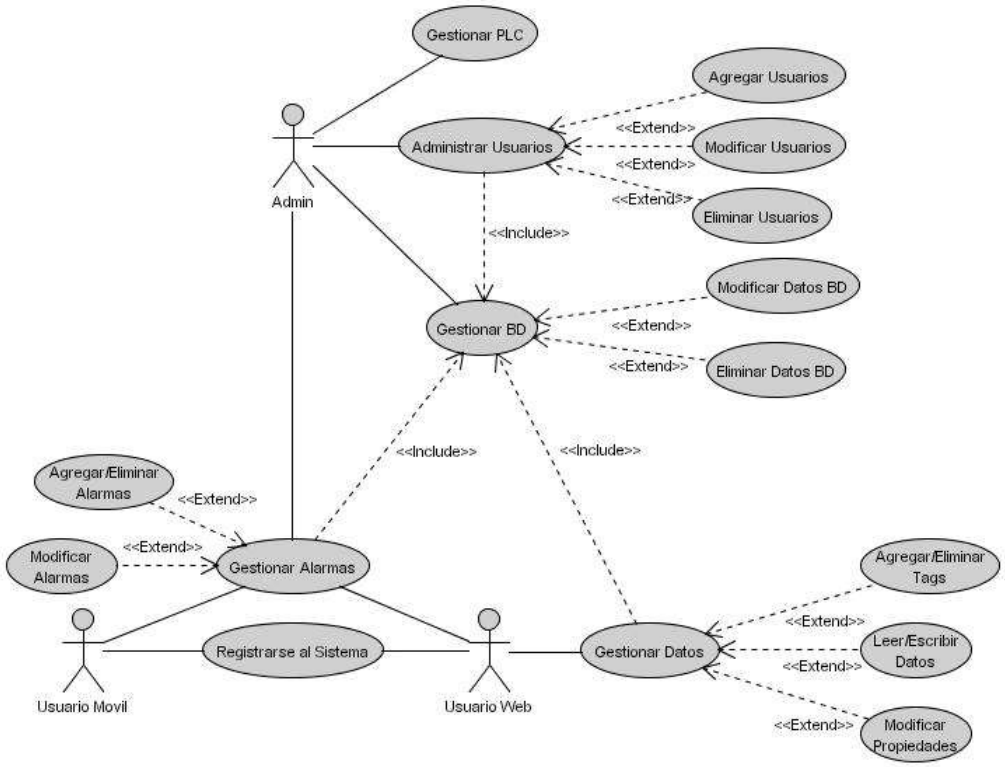

Figura 4.4. Diagrama de casos de uso extendido

4.7.2. Descripción de los escenarios de los Casos de Uso Esenciales

Caso de Uso No. 1: Gestionar PLC

Iniciador: Administrador

Precondición:

- Aplicación iniciada correctamente

- Validación correcta del Administrador

Flujo de Eventos

Flujo Principal:

- Se despliega la interfaz correspondiente a la Gestión de PLCs.
- El administrador busca, registra o elimina equipos en la aplicación.

Postcondiciones:

- Dispositivos registrados o eliminados de la aplicación.

Flujos Alternativos:

- Ninguno.

Excepciones:

- Posibles fallos de acceso o conexión a la red.
- Fallo en la conexión o en el estado de los dispositivos conectados a la red.

Recursos Especiales Utilizados:

- Los puertos de comunicaciones utilizados para la conexión del Equipo Servidor a los dispositivos.
- Recursos e infraestructura de red.

GUIs Relacionadas:

- IU_Gestion_PLC

Requerimientos No Funcionales:

- El administrador debe tener la posibilidad de agregar dispositivos manualmente o por búsqueda en la red.
- La interfaz deberá desplegar la información de varios dispositivos al tiempo.

Caso de Uso No. 2: Administrar Usuarios

Iniciador: Administrador

Precondición:

- Aplicación iniciada correctamente
- Validación correcta del Administrador

Flujo de Eventos

Flujo Principal:

- Se despliega la interfaz correspondiente a la Administración de los Usuarios.

- El administrador busca, agrega, elimina o modifica a los usuarios o sus características para que puedan acceder de manera remota por medio de Internet o a través de su teléfono móvil.

Postcondiciones:

- Usuarios creados, eliminados o modificados en el sistema.

Flujos Alternativos:

- Si el administrador decide agregar un nuevo usuario, deberá ingresar todos los parámetros de configuración correspondiente a éste.

- Si el administrador decide modificar un usuario existente, puede modificar uno o varios parámetros que desee de éste.

- Si el administrador decide eliminar un nuevo usuario, deberá ingresar nuevamente una contraseña que lo identifique como administrador del sistema.

Excepciones:

- Al momento de agregar o modificar un usuario, el administrador puede agregar datos no válidos o dejar campos vacíos.
- Problemas con la conexión a la base de datos.

- El usuario que el administrador desee modificar o eliminar, no exista.

Recursos Especiales Utilizados:

- Ninguno.

GUIs Relacionadas:

- IU_Admin_Usuarios
- IU_Modificar_Usuarios
- IU_Agregar_Usuarios

Requerimientos No Funcionales:

- Ninguno.

Caso de Uso No. 3: Gestionar Base de Datos

Iniciador: Administrador

Precondición:

- Aplicación iniciada correctamente
- Validación correcta del Administrador
- Conexión con la base de datos establecida

Flujo de Eventos

Flujo Principal:

- Se despliega la interfaz correspondiente a la Gestión de la Base de Datos.

- El administrador puede modificar o eliminar los datos almacenados de manera directa en la Base de Datos, en uno o varios campos de las tablas.

Postcondiciones:

- Datos modificados o eliminados de la base de datos.

Flujos Alternativos:

- Si el administrador decide modificar la base de datos, puede modificar uno o varios datos almacenados en esta.

- Si el administrador decide eliminar uno o varios datos de la BD, deberá ingresar nuevamente una contraseña que lo identifique como administrador del sistema.

Excepciones:

- Al momento de modificar un usuario, el administrador puede introducir datos no válidos o dejar campos vacíos.
- Problemas con la conexión a la base de datos.

Recursos Especiales Utilizados:

- Ninguno.

GUIs Relacionadas:

- IU_Gestion_BD
- IU_Modificar_BD
- IU_Eliminar_BD

Requerimientos No Funcionales:

- Filtrar el tipo de datos que se pueden incluir en la base de datos. Es decir, solo podrán ser incluidos en la BD, los datos del tipo que se determinen.

Caso de Uso No. 4: Gestionar Datos

Iniciador: Usuario Web

Precondición:

- Registro exitoso con el sistema

Flujo de Eventos

Flujo Principal:

- Se despliega la interfaz correspondiente a la Gestión Remota de Datos.

- El usuario puede modificar las propiedades de los datos, leer/escribir datos ó agregar/eliminar TAGS.

Postcondiciones:

- Datos modificados en el sistema.

Flujos Alternativos:

- Si el usuario decide modificar las propiedades de los datos, estará en la capacidad de cambiar por ejemplo la velocidad de lectura de los datos, entre otros.

- Si el usuario escoge la opción de leer/escribir datos, podrá hacerlo sin ningún inconveniente simplemente con escoger uno o varios datos del dispositivo que elija y ejecutar operaciones de lectura o escritura con estos.

- Si el usuario escoge la opción agregar/eliminar TAGS, deberá escoger uno o varios dispositivos disponibles en el sistema y realizar las operaciones de adición o eliminación de las TAGS, cumpliendo previamente con todos los requisitos pertinentes.

Excepciones:

- Problemas con la conexión a la aplicación Web o WAP, o con la validación.

- Al momento de modificar realizar cualquiera de las operaciones permitidas el usuario puede introducir datos no válidos o dejar campos vacíos.
- Problemas con la conexión a la base de datos.
- Problemas con la conexión a los dispositivos.

Recursos Especiales Utilizados:

- Recursos e infraestructura de redes Internet ó inalámbrica.
- Los puertos de comunicaciones utilizados para la conexión del Equipo Servidor a los dispositivos.

GUIs Relacionadas:

- IU_Modificar_Propiedades
- IU_RW_Datos
- IU_AD_Tags

Requerimientos No Funcionales:

- Se debe garantizar un fácil acceso a la aplicación, y una vez garantizado este, se debe ofrecer unas interfaces sencillas y amigables.

Caso de Uso No. 5: Registrarse al Sistema

Iniciador: Usuario Web, Usuario Móvil

Precondición:

- Aplicación en el servidor iniciada correctamente
- Conexión con el sistema establecida

Flujo de Eventos

Flujo Principal:

- Se despliega la interfaz correspondiente al acceso remoto vía Web o WAP.

- El usuario inicia su registro mediante la aplicación, para poder tener acceso a la gestión remota de la red industrial.

Postcondiciones:

- Usuario registrado al sistema y listo para realizar la gestión remota.

Flujos Alternativos:

- Si el usuario se registra a través de una aplicación Web, y el registro es exitoso obtendrá acceso a un menú de operaciones de gestión de Datos y Gestión de Alarmas.

- Si el usuario se registra a través de una aplicación Móvil, y el registro es exitoso obtendrá acceso a un menú de operaciones de gestión de Alarmas.

Excepciones:

- Problemas con la conexión a la aplicación Web o WAP, o con la validación.

Recursos Especiales Utilizados:

- Recursos e infraestructura de redes Internet ó inalámbrica.

GUIs Relacionadas:

- IU_Registro_Web
- IU_Registro_WAP

Requerimientos No Funcionales:

- Se debe garantizar un fácil acceso tanto a las aplicaciones Web como WAP, y una vez garantizado este, se debe ofrecer unas interfaces sencillas y amigables.

Caso de Uso No. 6: Gestionar Alarmas

Iniciador: Usuario Web, Usuario Móvil

Precondición:

- Registro exitoso con el sistema.

Flujo de Eventos

Flujo Principal:

- Se despliega la interfaz correspondiente a la Gestión Remota de Alarmas.
- El usuario remoto, en cualquiera de sus dos posibles condiciones (Web ó móvil) gestiona los estados, condiciones, prioridades de las Alarmas.

Postcondiciones:

- Gestión de alarmas realizada.

Flujos Alternativos:

- Ninguno.

Excepciones:

- Problemas con la conexión a la aplicación Web o WAP, o con la validación.
- Fallo en la conexión o en el estado de los dispositivos conectados a la red.

Recursos Especiales Utilizados:

- Recursos e infraestructura de redes Internet ó inalámbrica.

GUIs Relacionadas:

- IU_Gestion_Alarmas

Requerimientos No Funcionales:

- Se debe garantizar un fácil acceso tanto a las aplicaciones Web como WAP, y una vez garantizado este, se debe ofrecer unas interfaces sencillas y amigables.

4.8 ARQUITECTURA DE REFERENCIA PARA EL SERVICIO

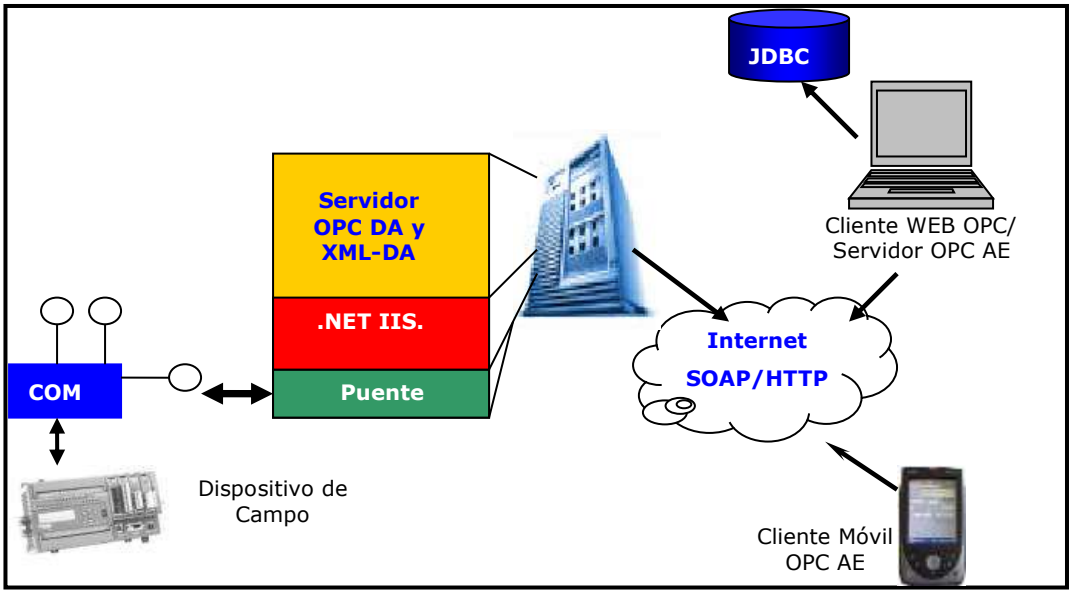

Figura 4.5. Arquitectura de referencia para el servicio

Esta arquitectura esta más especificada que la arquitectura inicial del sistema. En la figura 4.5 se puede apreciar que la parte del servidor de Datos XML-DA tiene un servidor Windows Internet Information Server (IIS) basada en .NET el cual utiliza un puente para comunicarse con objetos COM/DCOM y así poder obtener y modificar los datos que se encuentran en el dispositivo de campo.

Del lado del Cliente se encuentran dos tipos de usuarios: Un dispositivo Móvil que puede acceder a un servidor de Alarmas y Eventos OPC a través de Internet, SOAP (KSOAP) y HTTP; y un cliente OPC DA-XML que accede a los datos a través de Internet, SOAP y HTTP, pero a su vez es un servidor de Alarmas y Eventos OPC, con acceso a datos históricos mediante JDBC.

4.9 REALIZACIÓN DE CASOS DE USO

4.9.1. Gestión PLC

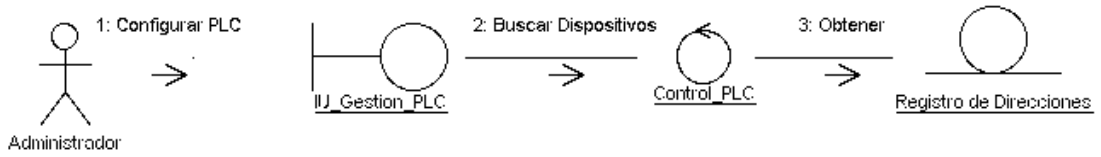

Figura 4.6. Diagrama de colaboración Gestión de PLC

4.9.2. Gestión de Usuarios

- **Agregar Usuarios**

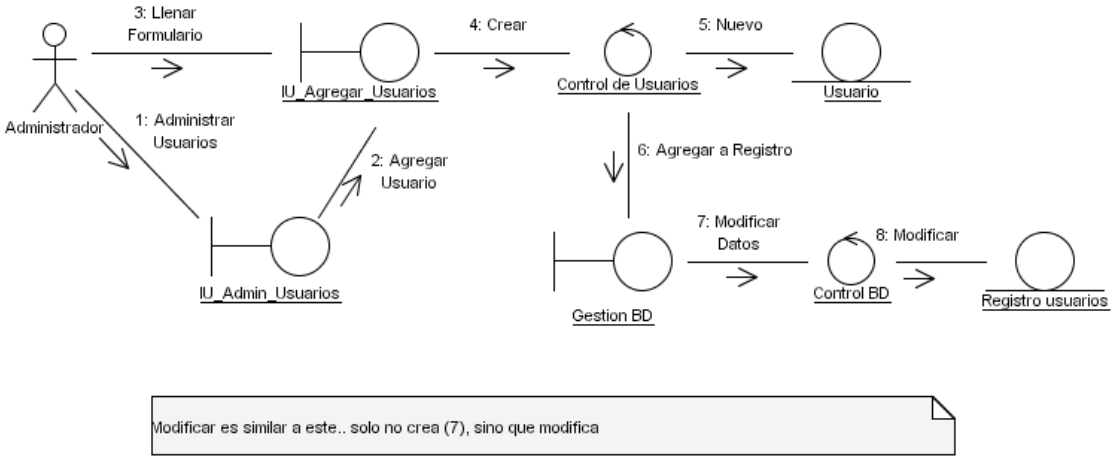

Figura 4.7. Diagrama de colaboración Agregar Usuarios

- **Eliminar Usuario**

Figura 4.8. Diagrama de colaboración Eliminar Usuario

4.9.3. Gestión de BD: Modificar

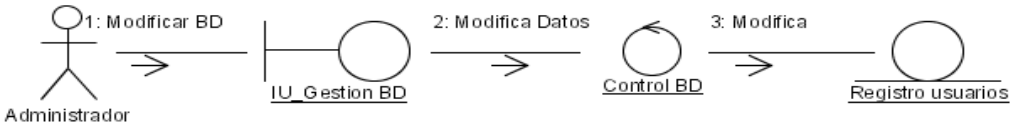

Figura 4.9. Diagrama de colaboración Modificar Usuario

4.9.4. Gestión de alarmas: Agregar/Eliminar

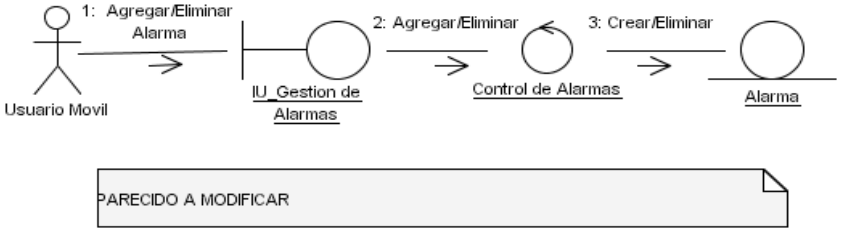

Figura 4.10. Diagrama de colaboración Agregar/Eliminar Alarmas

4.9.5. Registrarse al Sistema

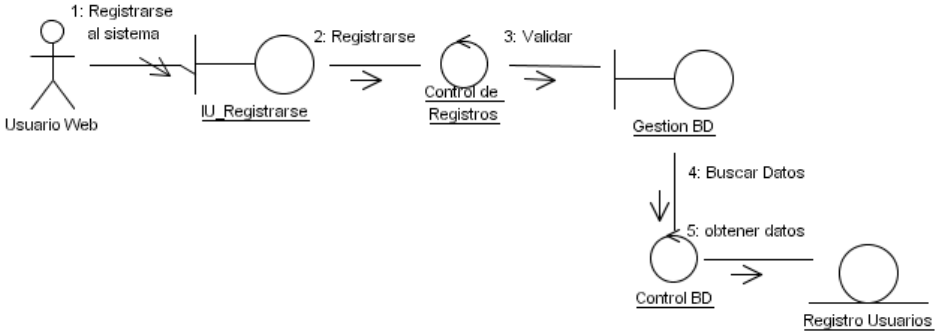

Figura 4.11. Diagrama de colaboración Registrarse al sistema

4.9.6. Gestión de Datos: Modificar

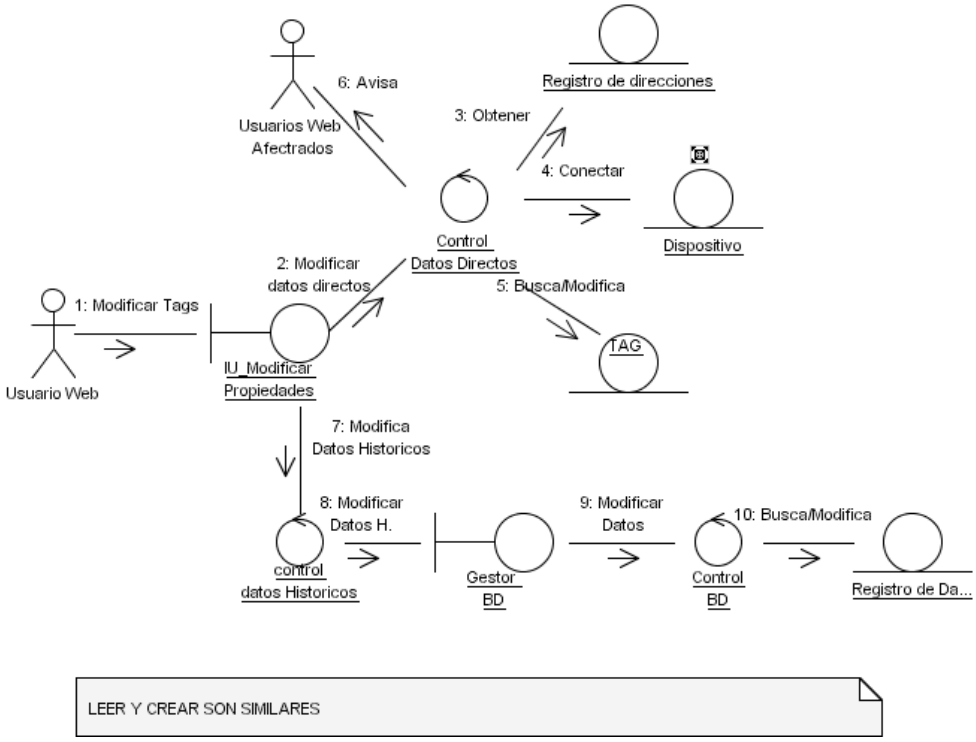

Figura 4.12. Diagrama de colaboración Gestión de datos: Modificar

4.10 PAQUETES Y CLASES DE ANÁLISIS

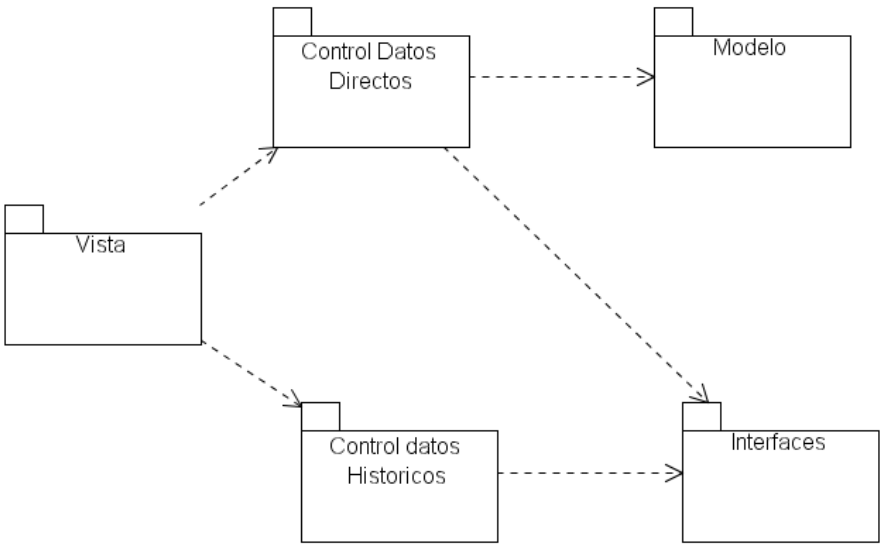

Figura 4.13. Arquitectura inicial del Sistema.

4.10.1. Clases del Paquete Vista

I1. IU_Admin_Usuarios: Se presenta cuando el administrador desea activar cualquier transacción con los usuarios que existen en la base de datos, permite eliminar un usuario y presenta como opciones IU_Modificar_Usuarios, IU_Agregar_Usuarios.

I2. IU_Modificar_Usuarios: Pide algunos datos del usuario que se quiere modificar, y los usuarios que se van a modificar.

I3. IU_Agregar_Usuarios: Pide los datos básicos del usuario que se quiere agregar.

I4. IU_Gestion_BD: Despliega una ventana con todos los servicios que tiene un administrador de base de datos para gestionar los mismos

I6. IU_Modificar_BD: Es una de las opciones de gestionar datos y le pide al administrador unos datos de entrada para encontrar los datos que desea modificar y además los datos nuevos que va a insertar o modificar.

I7. IU_Eliminar_BD: Es una de las opciones de gestionar datos y le pide al administrador unos datos de entrada para encontrar los datos a eliminar.

I8. IU_Modificar_Propiedades: Esta interfaz permite al usuario modificar el valor de las tags y da la opcion de escoger el dispositivo y las TAGs de tipo histórico o de tipo directo.

I9. IU_Leer_Datos: Esta interfaz permite seleccionar de cual dispositivo, grupo, y cuales TAGs quiere mostrar ya sea de datos directos, o dando unos parámetros extras para datos históricos

I10. IU_AD_Tags: Permite escoger un dispositivo para agregar una TAG que no exista y asignarle una dirección del dispositivo, o crear una nueva en el registro de datos históricos.

I11. IU_Registro_Web: Pide al usuario Web ingresar un nombre y una contraseña.
I12. IU_Registro_WAP: Pide al usuario WAP ingresar un nombre y una contraseña.

I13. IU_Gestion_Alarmas: Despliega al usuario opciones para ver estado de alarmas establecidas por el servidor de alarmas, opciones para crear nuevas o modificar las ya existentes.

4.10.2. Clases del Paquete Control Datos Directos

C1. Control de PLC: Por medio de algún proceso manual o automático permite configurar y encontrar los dispositivos que se encuentren activos en la red, y guarda en un registro estas direcciones IP.

C2. Control de Alarmas: Con esta clase se pueden ver, crear, modificar o eliminar características las alarmas del sistema.

C3. Control de Datos Directos: Busca los dispositivos que el administrador le permita ver, y permite al usuario escoger las TAGs para modificarlas, eliminar, o añadir una nueva. También se encarga de realizar algún tipo de acción para que los demás usuarios sepan los cambios que están sucediendo.

4.10.3. Clases del Paquete Control Datos Históricos:

C4. Control de Usuarios: Permite al administrador manipular los usuarios del sistema, sus permisos, claves, etc. Accediendo al registro en la base de datos.

C5. Control BD: Es el encargado de buscar, modificar, insertar, eliminar, etc. Los datos de cualquier tipo en la base de datos del sistema.

C6. Control de Datos Históricos: Busca con ayuda del control de BD los dispositivos que el administrador le permita ver, y permite al usuario escoger las TAGs para modificarlas, eliminar, o añadir una nueva.

4.10.4. Clases del Paquete Modelo

M1. Usuario: Se utiliza para distinguir quien y con que permisos puede acceder al sistema y a la red industrial. Tiene características como nombre y contraseña.

M2. Alarma: Es en realidad una TAG que se revisa en el tiempo. Esta alarma tiene características como prioridad, severidad, rango de alarma, nivel, etc.

M3. Dispositivo: Cualquier aparato que este conectado a un servidor OPC y que se pueda detectar desde la red. Debe tener un nombre y dirección asociado. Dentro de el se pueden asociar las TAGs en grupos y subgrupos. Un dispositivo por ejemplo puede ser un PLC o tarjeta de adquisición.

M4. Registro de direcciones: Es un conjunto de direcciones IP relacionadas con los dispositivos activos que el administrador brinda a los usuarios para poder ubicar los diferentes puntos en la red y sus dispositivos asociados.

M5. TAG: Es el nombre dado a las variables en un dispositivo industrial. Estas pueden cambiar su valor y están asociadas a un espacio en memoria del dispositivo al que pertenecen (PLC, o tarjeta).

4.10.5. Entidades (BD)

E1. Registro Usuarios: Contiene la lista de usuarios con algunos otros campos pertinentes para reconocer a los usuarios del sistema.

E2. Registro Datos: Son los datos relacionados con los dispositivos de campo tales como PLC, tarjetas, etc.

4.11 CREACIÓN DEL SERVICIO

La descripción completa de los servicios ser realizo basándose en la descripción del análisis realizada anteriormente y realizando actualizaciones paulatinas, las cuales son descritas en el Anexo A.

4.11.1. Diseño Inicial de interfaces

Teniendo en cuenta los requerimientos del cliente y las especificaciones del servicio se diseñaron las siguientes interfaces:

- **IU_Registrarse**

  Para registrarse al sistema se utiliza una interfaz sencilla con la cual se pretende obtener información relevante con los usuarios para validar su pertenencia y permisos al sistema. Entre los principales datos que se piden son: Dirección donde se encuentra la Base de datos y contraseña.

  El usuario solo tiene tres oportunidades de errar antes de que el sistema se cierre automáticamente.

Figura 4.14. Interfaz de conexión al servidor

Figura 4.15. Interfaz de validación de usuario.

Una vez el usuario ingrese al sistema debe poder saber cuales servidores están activos y desplegarle la información al usuario de una forma amigable. Teniendo las opciones de agregar un servidor, o eliminar otro.

Figura 4.16. Interfaz de usuario de Servidores XML-DA

- **IU_Leer_Datos:**

Una vez se conecte a un servidor, el usuario debe tener la opción de escoger cualquier servicio del sistema: Leer/Escribir, Ver propiedades, Gestión de alarmas, o almacenar los datos en el registro histórico.

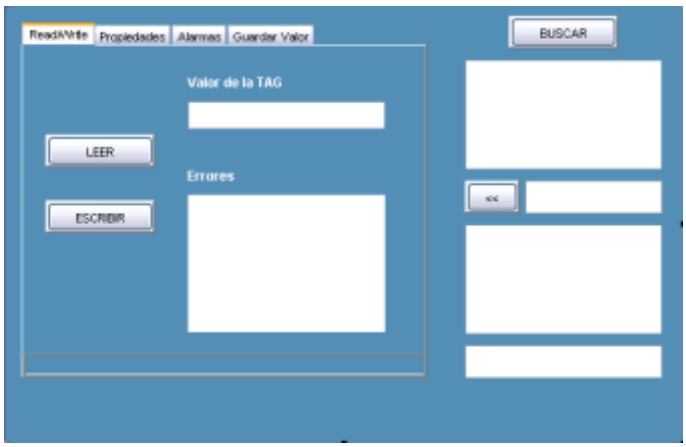

Figura 4.17. Interfaz de usuario para Leer/Escribir Datos desde el servidor.

Todos los servicios deben tener la opción de Browse o navegar por todos los datos disponibles en el dispositivo al que se conectó.

Figura 4.18. Interfaz de usuario de propiedades de TAGs.

En la figura anterior se muestra como se despliega la información de las propiedades de los datos a ser analizados.

Cuando se utilice la opción "Guardar Valor" se debe poder escoger la Tag a almacenar e ingresar las fechas de inicio y de finalización.

Figura 4.19. Interfaz de usuario para Almacenar Datos en registros Históricos

- **IU_Gestion_Alarmas:**

Cuando el usuario pide la opción de desplegar las alarmas podrá ver la siguiente información: Nombre de la alarma, Estado y la hora de activación. También podrá indicar cual fue la última en activarse o en cambiar de estado.

Figura 4.20. Interfaz de usuario para desplegar estado de alarmas.

Cuando el usuario quiera ver mas detalles de una alarma, debe seleccionarla utilizar la opción de ver detalles, en la cual debe desplegarse la siguiente información:

Figura 4.21. Interfaz de usuario para modificar y ver detalles de Alarmas.

Al ver los detalles, el usuario podrá además de obtener más información acerca de la variable, cambiar algunos otros como el tipo de comparación, los valores de comparación, su severidad, su actividad y disponibilidad, pero nunca la Tag asociada y el nombre.

Si desea agregar una alarma nueva, se utilizará la misma interfase, pero para poder agregarla debe haber llenado los datos mínimos requeridos, los cuales se distinguen por tener un asterisco (*).

La opción de "Recomendaciones" es un texto configurable donde se indican algunas sugerencias cuando se active la alarma.

La opción "Observadores" tiene una lista con todos los usuarios que tienen acceso y están recibiendo actualizaciones de los eventos asociados a esa alarma.

5. VALIDACION DEL SISTEMA

La validación del sistema se realizará mediante la planeación y seguimiento de pruebas a los diferentes casos de usos expuestos y desarrollados en este documento, utilizando diferentes herramientas para comprobar que el sistema sea lo más robusto posible y haciéndolo interactuar con herramientas utilizadas para el desarrollo y monitoreo de sistemas de control reales, además el sistema será validado bajo las plataformas Linux y Windows para comprobar su capacidad de ser multiplataforma.

5.1 VALIDACIÓN DE "Gestión de PLC"

5.1.1 Objetivo

Conectar y desconectar la aplicación a diferentes servidores XML disponibles en una LAN y en publicados en Internet.

5.1.2. Descripción

Se tienen tres diferentes servidores de prueba:

- Servidor OPC-DA, utilizando un puente Advosol[8] XDAWGSS en la dirección http://172.16.140.41/xdagwss/OpcDaGateway.asmx conectado a un servidor DA de RSLinx.
- Servidor de prueba de Lixmo[9] http://opcxml.dnsalias.org:8080/XmlDaSampleServer/Service.asmx
- Servidor XML-DA de prueba de Advosol en: http://172.16.140.41/PLC1/OpcXmlDaServer.asmx

[8] Advosol es una compañía Estadounidense desarrolladora de diferentes herramientas para el control y supervisión industrial.
[9] Lixmo. Herramienta OPC DA sobre Linux desarrollada por la Empresa Technosoftware AG.

Dos de ellos van a estar configurados por defecto y van a ser enviados desde la base de datos: El de Lixmo y el XML-DA Advosol de prueba. El de Lixmo debe estar activo pero el Advosol no, de tal forma que uno aparezca activo, pero el otro no.

Después de esto se activa el servidor y se actualiza el sistema, con lo cual se comprueba de forma gráfica la activación de éste. Por último, el usuario puede agregar manualmente el servidor OPC-DA con puente y comprobar que en la siguiente sección ya aparece incluido en la base de datos.

5.1.2 Resultados

Figura 5.1. Prueba 1 de Gestión de PLC.

Como se puede observar en la Figura 5.1, efectivamente se obtuvo la respuesta esperada, únicamente el Servidor 1 se encuentra activo, el cual corresponde al Servidor de Lixmo y siempre permanece activo. Por el contrario, el Servidor 2 (Advosol), se despliega en blanco y negro, con lo cual muestra que se encuentra inactivo.

Figura 5.2. Prueba 2 y 3 de Gestión de PLC

En la Figura 5.2, se puede observar que efectivamente se pudo agregar un Servidor más a la lista de servidores, y que el Servidor 2 ahora se encuentra activo. Ésta imagen se obtiene igualmente al iniciar de nuevo el sistema, teniendo ya el servidor 6 por defecto en la base de datos.

5.2 VALIDACIÓN DE "Gestión de Datos"

5.2.1 Objetivo

Comprobar y utilizar las funcionalidades de *Lectura, Escritura* y *Búsqueda* de datos, en sistemas de control y supervisión.

5.2.2 Descripción

Se utilizan los servidores anteriormente validados y activados, realizando las transacciones básicas planteadas en los objetivos. Para comprobar el correcto funcionamiento se utiliza RSLogix en el caso del servidor OPC DA con puente XML DA, y un cliente de prueba XML-DA desarrollado por Advosol para probar sus servidores.

5.2.3 Resultados

Utilizando el servidor de prueba de Technosoftware con la herramienta Lixmo se obtuvieron los siguientes resultados:

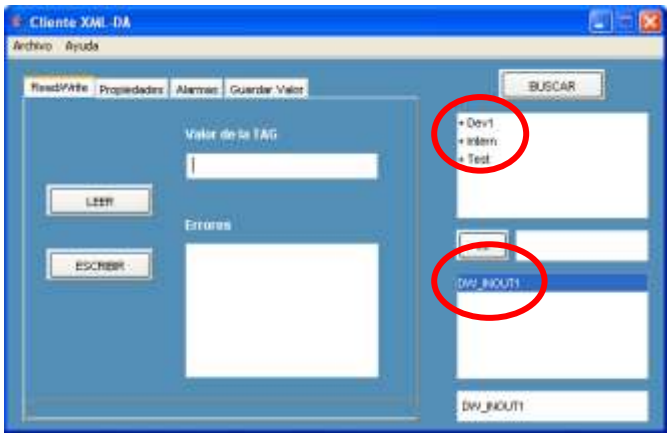

Figura 5.3. Búsqueda de datos en servidor Technosoftware publicado en Internet.

Se puede observar en la Figura 5.3, que en la búsqueda de elementos o TAGs, efectivamente la aplicación SGPM muestra los grupos y variables disponibles en el dispositivo al cual nos conectamos.

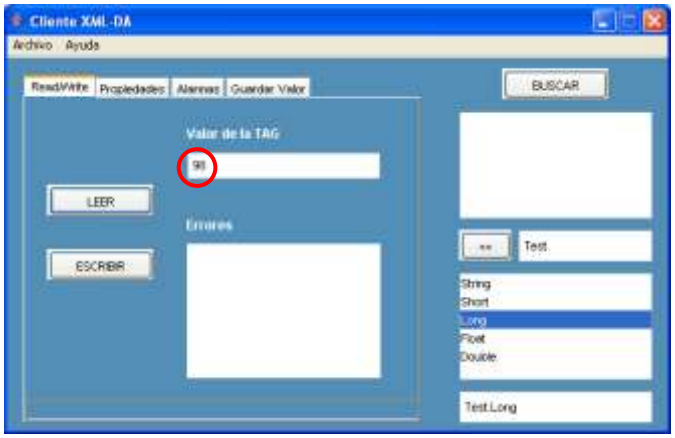

Figura 5.4. Valor leído en la Aplicación SGPM.

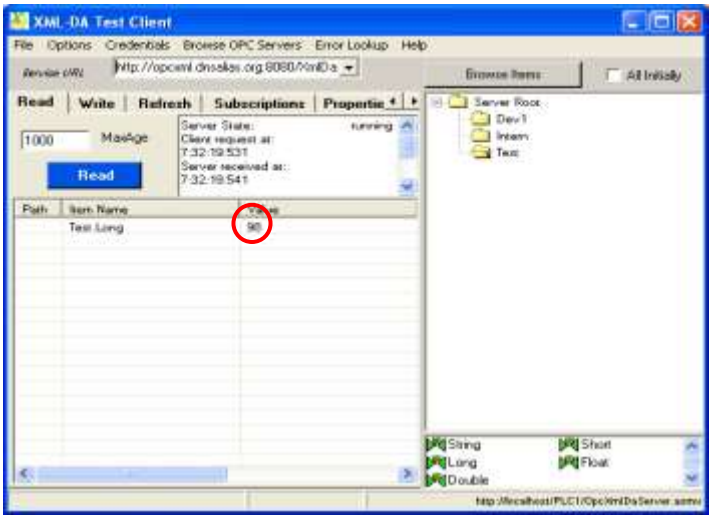

Figura 5.5. Valor leído en cliente XML-DA de prueba.

Utilizando el cliente de prueba se puede observar que efectivamente se lee el mismo valor en ambas aplicaciones.

Para validar la transacción *Escribir* escribimos un valor diferente al leído y después comprobamos el valor con el cliente de prueba:

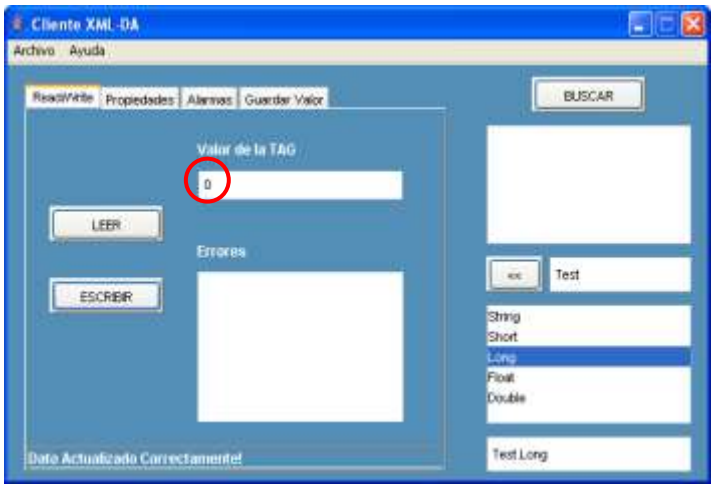

Figura 5.6. Transacción Escribir en cliente SGPM.

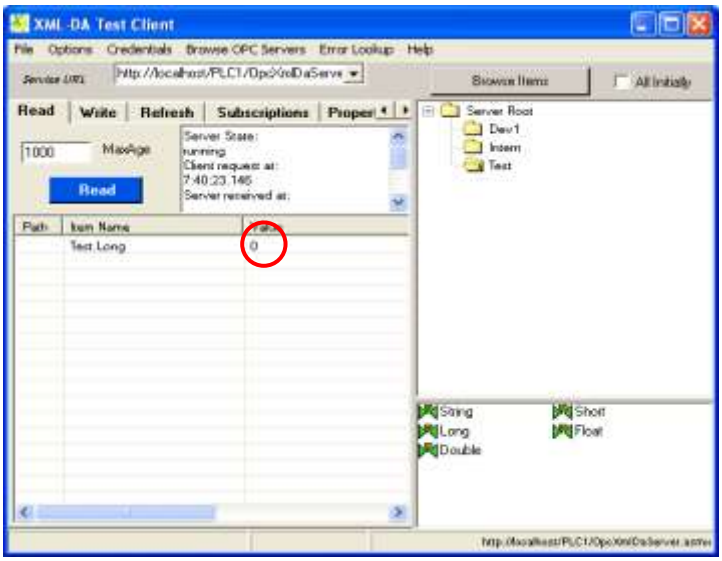

Figura 5.7. Transacción Escribir en cliente XML-DA de prueba.

Realizando las mismas pruebas con el puente XML conectado con el servidor OPC DA de RSLinx se obtuvieron los siguientes resultados:

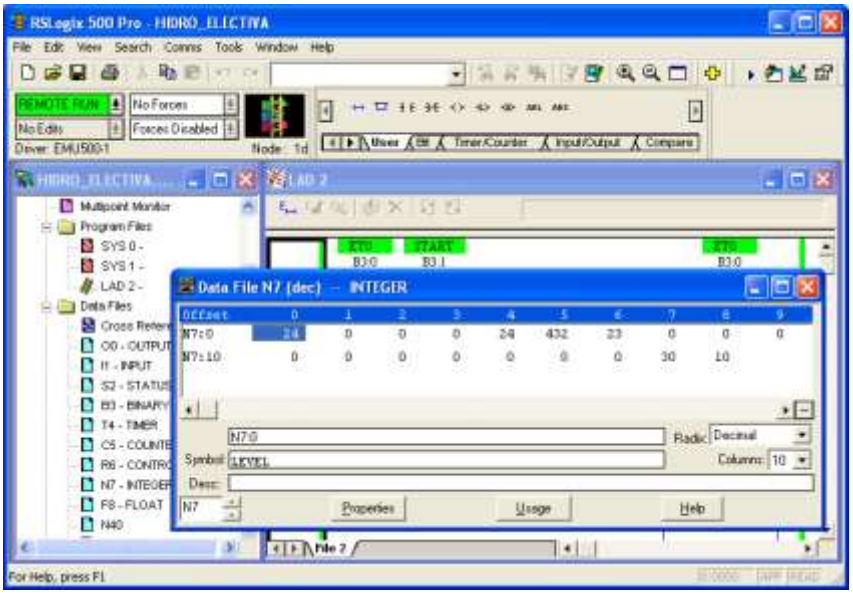

Figura 5.8. Valores de Variables de proceso desde RSLogix.

En la Figura 5.8, se puede apreciar las diferentes variables y registros que se presentan en el proceso, y en el cuadro frontal se ve el registro de variables enteras N7.

Inicialmente al utilizar la transacción *Buscar* de la herramienta SGPM funcionó de manera adecuada, desplegando los grupos de igual forma que el servidor OPC DA de RSLinx organiza las TAGs.

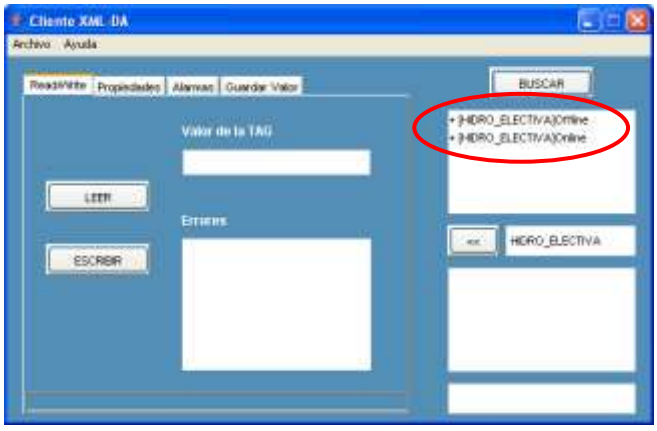

Figura 5.9. Transacción Buscar desde herramienta SGPM.

Pero cuando se quiso entrar a un valor en particular de cada grupo, surgió un error de lectura, en el cual expresaba que no se encontraban elementos dentro del grupo de variables.

Figura 5.10. Error de sistema al buscar Tags dentro del servidor OPC DA.

Cuando se buscó la razón del problema, se encontró que el servidor OPC DA de RSLinx implementa de una forma particular y no estándar la forma de nombrar los grupos y TAGs del sistema, creando una incompatibilidad con las demás herramientas que manejan el estándar XML-DA. Entonces, por ejemplo si se quiere llegar al grupo de TAGs del registro de enteros, usualmente la notación para el nombre de la TAG (*ItemName*) seria: "NombreDeProyecto.Online.N7.N7:6", pero RSLinx lo nombra de la siguiente manera: "[NombreDeProyecto]N7:0", lo cual genera el error anteriormente expuesto.

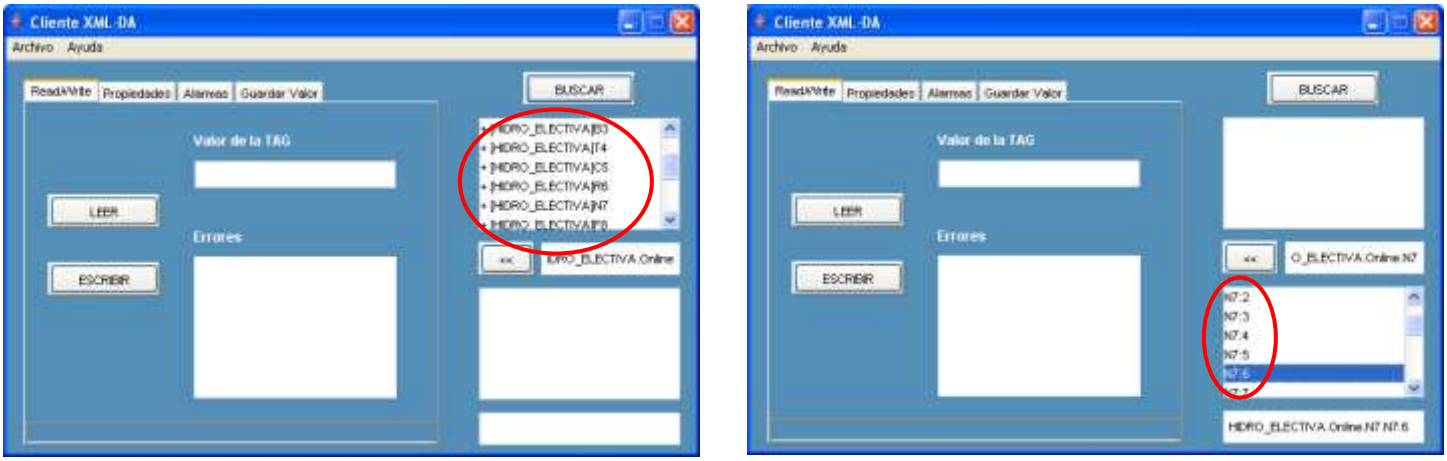

Figura 5.11. Transacción Buscar dando paso a los nombres de Grupos de TAGs.

En la figura anterior se puede ver que si se pueden ver de cualquier forma los nombres de los Grupos y TAGs en el servidor DA, pero se debe hacer paso a paso y utilizando una notación diferente. Ésta falla puede ser corregida tanto en el cliente como en el servidor XML DA, pero debe hacerse cierta configuración inicial.

De igual forma para utilizar la transacción *Leer*, se debe utilizar la notación del RSLinx mencionada anteriormente, e introducirlo manualmente.

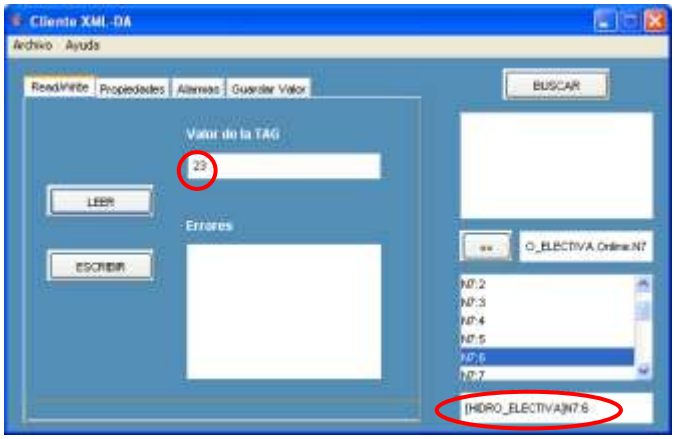

Figura 5.12. Transacción leer desde SGPM.

En el cuadro de texto inferior derecho se puede observar que se introdujo manualmente un nombre de TAG N7:6 con notación diferente, y el valor leído es 23, si se observa en la Figura 5.8, el valor de N7:6 es efectivamente 23.

Cuando se intentó realizar la transacción *Escribir* o la de *Obtener Propiedades*, ocurre un error relacionado con el tipo de datos que maneja el servidor OPC DA y el manejado por XML DA; por lo tanto ninguna de estas dos transacciones se puede realizar con este servidor OPC DA de RSLinx de manera exitosa.

5.3 VALIDACIÓN DE "Administrar Base de Datos"

5.3.1 Objetivo

Gestionar los datos de un servidor OPC DA en un proceso industrial almacenándolos en una base de datos.

5.3.2 Descripción

Se escogerá una variable del proceso y se guardarán sus datos periódicamente a gusto del usuario utilizando el servidor OPC DA de RSLinx. Después, el sistema SGPM debe mostrar una tabla con los valores almacenados, de una forma fácil para el usuario.

5.3.3 Resultados

Como se puede ver en la Figura 5.13, la tabla muestra detalladamente el valor de la TAG y la hora a la cual se guardó su valor.

Figura 5.13. Tabla de valores almacenados en la base de datos.

5.4 VALIDACIÓN DE "Gestión de Alarmas"

5.4.1 Objetivo

Mostrar tanto en un usuario Web como en un usuario móvil la activación de una alarma del proceso.

5.4.2 Descripción

Se tomarán tres variables de proceso y se creará una condición de alarma para cada una de ellas, de tal forma que cuando éstas se activen, generen la alarma y despliegue la información a los usuarios Web y Móvil.

5.4.3 Resultados

Figura 5.14. Cliente Web con una de sus tres alarmas activas.

Como se observa en la imagen anterior, el cliente Web presenta en color rojo y mostrando su estado activo la alarma que se activó por última vez y la fecha de su activación.

En la siguiente figura se puede observar cuando el cliente móvil recibe la misma alarma (Static.Analog.Types.int)

Figura 5.14. Cliente Móvil recibiendo mensaje de alarma activa.

6. CONCLUSIONES Y TRABAJOS FUTUROS

Este trabajo ha tratado diversos temas relacionados con las redes telemáticas e industriales, utilizando medios y herramientas que nunca habían sido utilizadas en el grupo de investigación de Automática industrial, y algunas otras que son nuevas y emergentes a nivel mundial. Por esta razón se pueden sacar diversas conclusiones. Estas conclusiones se pueden clasificar en diversos tipos: Tendencias y evolución de las redes industriales, implementación y desarrollo de redes industriales en entornos y plataformas diferentes, y por ultimo desempeño y respuesta del Sistema de Gestión de PLC multiplataforma en una red industrial real.

Según los estudios realizados a las diferentes arquitecturas y desarrollos en redes industriales, especialmente en la parte de supervisión y monitoreo podemos concluir:

- Desde la parte mas baja de la planta que comunica y transporta los datos crudos, hasta la capa superior donde la empresa hace sus negocios y se da cuenta realmente de sus ganancias, ha existido una gran batalla en cuanto a protocolos propietarios por parte de las diferentes empresas que desarrollan los PLCs y las mismas redes industriales. Pero en los últimos años, el sector de desarrollo se ha agremiado para poder acabar con este problema.

- Las redes industriales tienden a migrar a protocolos mas usados como TCP/IP para la comunicación de su información, aunque con modificaciones para no sacrificar la calidad de la misma. Su principal objetivo es el de permitir una mayor estandarización de su producto y de poder utilizar infraestructuras ya establecidas invirtiendo menos recursos en otras redes diferentes y especializadas.

- Existen arquitecturas y redes desarrolladas sobre plataformas Windows que permiten intercomunicación entre los diferentes protocolos de redes y el fácil

manejo de la misma información por parte de las aplicaciones en la parte más alta de la jerarquía de redes industriales, tales como OPC DX.

- La integración de las redes industriales quiere cubrir absolutamente todos los niveles de su pirámide jerárquica, y lo esta haciendo bajo el paradigma de Servicios Web y portabilidad multiplataforma, tanto a nivel de intranet como Internet por la misma naturaleza del negocio.

En cuanto al desarrollo de sistemas industriales en diferentes plataformas podemos concluir lo siguiente:

- El problema para hacer de los sistemas de supervisión y control unos sistemas portables y estándar en diferentes tipos de plataformas, radica en la arquitectura en la cual se han desarrollado (COM/DCOM), la cual es muy cerrada y de carácter propietario, inhabilitando la comunicación con otros tipos de sistemas de objetos como RMI, CORBA o JMX.

- Las principales herramientas utilizadas para acoplar sistemas de control y monitoreo desarrollados para plataformas Windows a otros sistemas operativos son lenguaje Java y C.

- Básicamente existen dos formas de acoplar un sistema de control basado en comunicaciones de objetos COM/DCOM a otro sistema o plataforma diferente: con puentes y proxies que hagan el acople mediante manejo de librerías y estructuras nativas (caso de Java), o mediante el uso de XML y los Servicios Web.

- Existen en el mercado algunas herramientas (propietarias) que permiten la supervisión de sistemas de control en plataformas Linux y UNIX utilizando puentes para poder pasar los datos de sistemas COM/DCOM a cualquier otro; pero también existe una especificación de carácter libre y con el fin de estandarizar, que permite el acceso a los datos en una red industrial para cualquier plataforma: OPC XML-DA.

Finalmente, las conclusiones que podemos sacar de la herramienta desarrollada, con respecto a los resultados obtenidos y su desempeño son:

- Se pueden generar todas las aplicaciones actualmente requeridas por un sistema de control y supervisión utilizando una plataforma Java y un servidor XML-DA desarrollado en cualquier lenguaje.

- Desarrollando un Cliente Web de la especificación OPC XML-DA en J2EE, se puede tener acceso ilimitado a los datos de un dispositivo de campo y ejecutar la aplicación en cualquier plataforma, ya sea Linux, UNIX, Windows, o un dispositivo móvil.

- Utilizando la especificación OPC XML-DA se pueden integrar todos los servicios necesarios para el control y supervisión de dispositivos de campo en un proceso industrial, y utilizar los mismos desde cualquier sitio remoto con acceso a Internet, o dentro de una LAN.

- Para un sistema de control y supervisión en un proceso industrial, los tiempos de respuesta de la herramienta desarrollada en Java son muy buenos y no generan ningún retardo en el desempeño del mismo. Se puede decir que cumplen con la definición de tiempo real bajo condiciones de red específicas y limitadas.

- La adquisición de los datos por medio de la aplicación XML-DA puede generar errores y terminaciones de la conexión dependiendo del tráfico en la red.

- Algunos procesos, especialmente los relacionados con servicios que manejan poca información como el de ver el estado de un servidor, utilizan muchos recursos para una transacción simple y no son óptimos para un sistema de este tipo, pues pude generar errores y aumentar el trafico en la red.

- Los costos de desarrollo de una herramienta de control y supervisión utilizando sistemas operativos de carácter libre y software de desarrollo de libre distribución, pueden reducir ostensiblemente los costos del mismo, sin reducir su calidad.

- Para poder desarrollar un servidor OPC XML-DA sobre lenguaje Java se necesita más que la simple especificación. Debe desarrollarse algún medio o utilizar uno ya desarrollado que permita obtener los datos directamente desde la memoria o que haga una comunicación con los objetos COM.

Este proyecto tan solo es el inicio de muchas otras posibles aplicaciones siguiendo la arquitectura JFPC, y la aplicación desarrollada tiene solo las funcionalidades básicas de un sistema de control y supervisión, pero la potencialidad de Java en sistemas multi-plataforma utilizando la Arquitectura Unificada propuesta por OPC puede explotarse muchísimo más. Algunas de las sugerencias de cómo explotarse son:

- Desarrollo de un servidor Java OPC XML-DA, utilizando cualquiera de los servidores gratuitos, como lo es Apache Tomcat, y haciendo el paso de datos de memoria a la aplicación Java utilizando librerías Nativas. De esta forma, se podría tener el sistema total independiente de cualquier plataforma Windows.

- Debe desarrollarse completamente un modulo de seguridad, ya sea utilizando la especificación dada por la OPC o utilizando cualquier arquitectura de encriptación o protección existente. Es imprescindible el desarrollo de un modulo con estas características para una aplicación industrial real pues de lo contrario el proceso industrial se vería afectado por cualquier ataque desde Internet.

- Implementación de la especificación OPC-DX para permitir total administración y comunicación de los recursos a nivel de planta, como sensores, buses de campo, etc. y realizar una supervisión completa de todo el proceso.

- Optimización de los servicios Web XML-DA más sencillos y que consumen más recursos con relación a su capacidad de transporte de información con otras herramientas diferentes a Servicios Web, o implementando un servicio donde se cambie el manejo de la información utilizando otras herramientas como UDDI (para el caso del estado de servidores) o envió de la información por ráfagas.

- Mejoramiento del manejo de las interfaces gráficas del sistema, haciéndolo más didáctico y con entornos hoy utilizados para desplegar la información al usuario de una manera más comprensible y simple.
- Optimización del manejo de TAGs del proceso y datos almacenados, para que en la red industrial se puedan hacer las transacciones sin afectar los procesos de supervisión llevados a cabo por otros usuarios, o por lo menos para informar al resto de participantes de las actualizaciones llevadas a cabo en el proceso.

- Implementación de una arquitectura donde el Servidor pueda ser cliente y el cliente pueda ser también servidor, y de esta manera monitorear los valores de las TAGs de una forma continua, sin tener que realizar manualmente la petición del servicio de un cliente a un servidor.