Universidad de las Ciencias Informáticas
Facultad 5

Título: Framework para el desarrollo de manejadores de dispositivos, para el proyecto SCADA Guardián del ALBA.

Memoria de Proyecto para optar por el título de

Ingeniero en Ciencias Informáticas

Autor(es): Luis Enrique García Hernández

Tutor(es): Dr. Rafael Arturo Trujillo Codorniu

Co-tutor: Ing. Amado Espinosa Hidalgo

Consultante: Lic. Juan Antonio Fung Goizueta

Ciudad de la Habana, abril de 2009

DECLARACIÓN DE AUTORÍA

Declaro ser autor del presente informe y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los 29 días del mes de abril del año 2009.

_____

Firma del Autor
(Luis Enrique García Hernández)

_____

Firma del Tutor
(Dr. Rafael Arturo Trujillo Codorniu)

DATOS DE CONTACTO

Dr. Rafael Arturo Trujillo Codorniu
Máster en Ciencias Físico-matemáticas en 1979, Universidad de Odessa, URSS. Doctor en Ciencias Físico-matemáticas en 1986, Universidad de Rostov del Don, URSS. Ostenta la categoría docente de Profesor Titular.

Email: rafaeltc@uci.cu

Ing. Amado Espinosa Hidalgo
Ingeniero Informático y profesor asistente del Departamento de Ingeniería y Gestión de Software de la Facultad 5. Posee 6 años de experiencia en la actividad docente y productiva.

Email: aespinosa@uci.cu

Lic. Juan Antonio Fung Goizueta
Licenciado en Física en 1986, Profesor Asistente, Especialista en Microelectrónica de la Universidad de Sofía. Experiencia Docente de 19 años. Experiencia en la coordinación y dirección de proyectos informáticos de 5 años.

Email: fung@uci.cu

AGRADECIMIENTOS

Le agradezco todo lo que soy a mis padres, todo su amor, sus consejos, su apoyo constante y por su ejemplo imperecedero que guía mi vida.

A mi hermana, por su cariño, por su mirada periodística al revisar mi tesis.

A mi abuela Mima por sus atenciones y su amor a lo largo de toda mi vida.

A mi abuela Lidia y mi abuelo José por su cariño y por estar siempre pendientes de mí, a mis primos, mi tío, a Beby, a toda mi familia.

Agradezco al proyecto Emedia la vinculación de la docencia con la producción; la oportunidad de contar con el ejemplo, las enseñanzas y la amistad de Fung y René.

Le agradezco a todos mis compañeros del proyecto productivo SCADA, en especial a mi tutor, Dr. Rafael Trujillo, mi tutor, por sus consejos, sus enseñanzas, por contribuir mucho en mi formación profesional y por su amistad. Agradezco también al Co-tutor, Amado Espinosa por su apoyo para la realización de mi tesis; así como a Moisés Herrera por la revisión de mi tesis y sus consejos técnicos que me fueron de mucha ayuda.

Le agradezco a todas mis amigas, a las niñas de Emedia, Irina, Iliana y Ana Silvia, a Dina. A Ara por su amor.

Le agradezco a todos mis amigos y en especial a Ariel Yasel, Curiel, Raúl René, Tony Cedeño, Ariel Chávez, Manu y Pepe Tony, que son también mis hermanos y que me apoyaron en todo.

DEDICATORIA

A mis padres, mi familia y mis amigos.

SINTESIS

En los procesos de automatización es imprescindible obtener las magnitudes de la planta, de esta forma, es posible conocer el estado y evolución del proceso que estamos controlando. Para el logro de este objetivo emplean dispositivos como pueden ser: autómatas programables, PLC, sensores, etc. Estos equipos poseen múltiples protocolos de comunicación a través de los cuales los sistemas SCADAs pueden obtener datos recopilados del campo.

El presente trabajo explica la realización de un *Framework* orientado a objetos, para el desarrollo de manejadores de dispositivos, del sistema SCADA: Guardián del ALBA. Este *Framework* proporciona una interfaz estándar de acceso a estos instrumentos y oculta al SCADA las diferencias entre los protocolos, así como las características de Hardware de los dispositivos enlazados con él. La generalidad de la arquitectura no impacta negativamente en el rendimiento de los manejadores, garantizando la eficiencia en la adquisición de datos que es el paso inicial para el tratamiento de la información en un sistema SCADA.

Tomando como base la realización de estas piezas de software, el autor expone las ideas que enriquecieron la solución y sus aportes individuales a la misma. Se brinda una panorámica al lector de los contenidos abordados y los principales resultados obtenidos. Se analizan las características fundamentales de las aplicaciones para la supervisión y control de los procesos industriales, y de los manejadores de dispositivos como componentes de gran valor para dichas plataformas de software.

PALABRAS CLAVES

Framework, dispositivo, protocolos industriales, SCADA, manejador.

TABLA DE CONTENIDOS

TABLA DE ILUSTRACIONES

INTRODUCCIÓN

Entre los mecanismos de control de procesos industriales utilizados por el hombre en la actualidad están los SCADAs, acrónimo de *Supervisory Control and Data Acquisition* (en español, Control Supervisor y Adquisición de Datos). Estos sistemas juegan un papel fundamental en la industria, pues comprenden las soluciones de aplicación que necesitan de la captura de información de un proceso o planta industrial, la cual es empleada para realizar análisis con los que se pueden obtener importantes indicadores que permiten una realimentación del proceso. De forma general se consideran como funciones básicas de un sistema SCADA las enunciadas a continuación (Hernández Lugones, 1998):

1. Supervisión Remota de Instalaciones
2. Control Remoto de Instalaciones
3. Procesamiento de Información
4. Presentación de Gráficos Dinámicos
5. Generación de Reportes
6. Presentación de Alarmas
7. Almacenamiento de Información Histórica
8. Presentación de Gráficos de Tendencias
9. Programación de Eventos

En el presente trabajo se exponen características de diseño e implementación del *Framework* denominado *DriversCore*; que es la base para el desarrollo de manejadores de dispositivos del sistema SCADA "Guardián del ALBA". Este sistema de supervisión y control fue desarrollado gracias a la cooperación entre la República Bolivariana de Venezuela y la República de Cuba. En su construcción participaron varias empresas venezolanas y cubanas: DST-AIT PDVSA, ULA, DBAccess, IntelCom, Ingesi, Isca, la Universidad de las Ciencias Informáticas, la Universidad Central de las Villas, el CEDAI, y el Instituto Superior Metalúrgico de Moa. Además permite la supervisión y control automático de los procesos industriales en el sector petrolero, aumentando la eficiencia y seguridad en esta industria. Esta aplicación desarrollada por especialistas cubanos y venezolanos para PDVSA, utiliza software libre, con lo cual contribuye no sólo a las disminución de costos sino también a elevar la soberanía tecnológica de esta rama de la economía.

El SCADA "Guardián del ALBA" es configurable y fácilmente extensible para su operación con nuevos dispositivos, cuenta con una interfaz amigable que apoya a la toma de decisiones eficaces y el

seguimiento efectivo de los procesos en la industria. Es muy flexible y permite la integración con otros sistemas, su adaptación y extensión. El sistema se encuentra en funcionamiento en algunas instalaciones de PDVSA en Venezuela con alentadores resultados.

EL proceso de intercambio de información entre el nivel de supervisión y el nivel de campo se establece comúnmente a través de los protocolos de comunicación, los cuales establecen el "idioma", y las reglas para entablar una comunicación entre los dispositivos de campo, sensores, actuadores y el nivel de supervisión.

Estos protocolos son especificados en capas, dentro de las cuales la más baja se denomina capa física en la mayoría de los estándares, dentro de los que se destaca el modelo OSI. La capa física describe el medio sobre el cual se transmite la información, dentro de estos medios se pueden mencionar, par trenzado y fibra óptica. Sobre estos medio se establecen estándares de comunicación con normas de hardware que permiten una integración en redes de dispositivos que cumplan con dichas normas, dentro de los cuales podemos destacar RS-232, RS-485, Ethernet, etc. Los dispositivos de campo se conectan a estas redes y establecen la comunicación con sus homólogos a través de los protocolos de comunicación. De esta manera los sistemas supervisores que deseen intercambiar información con dichos dispositivos requerirán (Herrera Vázquez, 2008): 1) Acople al medio físico de los dispositivos, 2) Implementación del protocolo de comunicación para establecer una conversación y realizar las peticiones de escritura y lectura de la información de proceso o de estado de los dispositivos.

El DriversCore, ha permitido facilitar el proceso de elaboración de manejadores de dispositivos, basándose siempre en el estricto cumplimiento de los requisitos especificados en el documento: "Interfaz Genérica de los manejadores" (Trujillo Codorniu, y otros, 2008). A partir de esta interfaz se han desarrollado los manejadores de los protocolos Modbus TCP, Modbus RTU, Modbus ASCII, OPC, EtherNet/Ip, ABInterchange(los dos últimos pertenecen a la compañía Allen Bradley) y BSAP de Bristol. Varios de estos manejadores se encuentran en fase piloto en la instalación del patio de Tanques de San Silvestre en Barinas, Venezuela y en otras instalaciones de la industria venezolana.

ANÁLISIS CIENTÍFICO METODOLÓGICO

Al iniciar la fase de elaboración de los manejadores del sistema SCADA, en noviembre de 2006, identificamos la ausencia de un conjunto de clases que sirviera de base para la creación de manejadores, permitiera además traducir las llamadas del módulo de recolección a los manejadores a través de la

interfaz genérica, que propiciara la configuración en tiempo de ejecución de los manejadores y brindara reportes con información suficiente para diagnosticar las posibles causas de las fallas de comunicación. Para darle solución a este problema, el trabajo realizado se concentró en el *proceso desarrollo de manejadores de dispositivos,* para accionar concretamente *en* el *módulo de adquisición de un sistema SCADA.*

El objetivo principal del trabajo fue:

Desarrollar un *Framework*, orientado a objetos para el desarrollo de manejadores de Dispositivos que permitiera: traducir los llamados del SCADA a los manejadores a través de la interfaz genérica, específicamente se estableció:

- Proporcionar una implementación de la interfaz estándar de acceso para los manejadores de dispositivos.
- Ocultar al SCADA las diferencias entre los protocolos y características de Hardware de los dispositivos que con él se enlazan.
- Brindar soporte a múltiples modelos de cooperación (modelo Cliente / Servidor y el modelo Productor / Consumidor).
- Garantizar el mecanismo de análisis de las solicitudes reducibles que le permita al SCADA un mayor control sobre la planificación de las tareas y una menor latencia en el despacho de los comandos que representan solicitudes de alta prioridad.
- Permitir el establecimiento de parámetros de los manejadores.
- Posibilitar el acceso a información de diagnóstico.
- Desarrollar el sistema usando un patrón de diseño arquitectónico multicapas.
- Servir de base para el desarrollo de manejadores, de forma tal, que todas las funcionalidades comunes sean implementadas como parte del DriversCore.

Para poder dar cumplimiento a los objetivos propuestos nos formulamos las siguientes interrogantes:

¿Se lograría implementar un *Framework* que proporcionara una interfaz estándar de acceso a los manejadores, ocultando al SCADA las diferencias entre los protocolos y características de Hardware de los dispositivos que con él se enlazan?

¿Se lograría desarrollar un Framework que contenga todas las funcionalidades comunes a los manejadores, orquestando las funcionalidades incorporadas a través del uso de las relaciones de herencia?

Las tareas de investigación que fueron llevadas a cabo se pueden resumir en las siguientes:

1. Estudio del problema y estado del arte.
2. Análisis y diseño de las clases que conformaran el *Framework* (*DriversCore*), basado en la Interfaz Genérica de los Manejadores (Trujillo Codorniu, 2007a).
3. Implementación, pruebas de unidad y documentación del Framework (DriversCore).
4. Utilización de estructuras compatibles con la mayoría de los compiladores de C++ y con los sistemas operativos GNU/Linux y Windows.
5. Implementación de un manejador (DriverDemo) que sirve de ejemplo para el desarrollo de manejadores de dispositivos.

IMPACTO DE LA PROPUESTA

La bibliografía consultada está disponible en Internet, en la biblioteca de nuestra universidad y en la base bibliográfica del proyecto; estos materiales poseen alto nivel científico. Todo el desarrollo se realizó utilizando software libre. El sistema operativo utilizado fue Debian GNU/Linux.

La implementación del núcleo de los manejadores ha proporcionado una interfaz estándar de acceso a los manejadores, que oculta al SCADA las diferencias entre los protocolos y características de Hardware de los dispositivos que con él se enlazan.

Se ha garantizado un alto rendimiento de los manejadores, gracias al uso de un diseño complejo pero eficiente, para la comunicación entre el SCADA y los manejadores de dispositivos. Dicho mecanismo facilita que las tareas de recolección queden bajo el mando del planificador del SCADA y se minimicen los contextos que tengan que ser creados internamente en los manejadores. De esta manera, en lo posible, los manejadores serían independientes al SCADA.

El *Framework* es multiplataforma al ser compatible con los sistemas GNU/Linux, Windows y Mac OS/X.

Ha demostrado ser lo suficientemente flexible como para asimilar, con todas sus funcionalidades, protocolos de diferente naturaleza, tales como Modbus, *OPC* y EtherNet/Ip.

El mecanismo de análisis de las solicitudes reducibles le permite al SCADA un mayor control sobre la planificación de las tareas y una menor latencia en el despacho de los comandos que representan solicitudes de alta prioridad.

Posibilita que el recolector sea altamente escalable, pues no necesita de la creación de contextos de ejecución en cada driver.

PUBLICACIONES Y TRABAJOS RELACIONADOS CON LA INVESTIGACIÓN

El primer trabajo realizado sobre este tema en la UCI fue una ponencia presentada en la Jornada Científica Estudiantil (JCE) en el mes de mayo del año 2007, “Framework para el desarrollo de manejadores”, obtuvo la categoría de relevante a nivel UCI; dos meses más tarde fue presentado en el XVII Concurso Nacional de Computación, donde obtuvo el premio Relevante.

En julio del año 2008, durante la realización del evento internacional FIE'08 en la ciudad de Santiago de Cuba se presenta el trabajo: “Interfaz Genérica para los manejadores de dispositivos en el proyecto SCADA PDVSA”, tema que fue seleccionado para la presentación de una conferencia magistral.

Durante la ejecución del proyecto SCADA “Guardián del ALBA” el autor del presente trabajo ha participado en el desarrollo de los productos pertenecientes a los entregables de la línea Manejadores de Dispositivos. Hasta la actualidad, por conceptos de ingresos directos, dicha línea de desarrollo ha aportado cerca de 1 millón de dólares al país, con un reporte de utilidades netas del 90%. Estos datos están registrados en los anexos firmados entre PDVSA y la empresa comercializadora ALBET.

DESARROLLO

1. Fundamentación Teórica

SCADA proviene de las siglas de *Supervisory Control And Data Acquisition* (Adquisición de datos y supervisión de control). Los sistemas SCADA permiten realizar supervisión y control de procesos industriales. La mayoría de los SCADAs existentes en el mercado proporcionan información del proceso a diversos usuarios: operadores, supervisores de control de calidad, mantenimiento, etc. Dichos sistemas permiten además, realizar un esquema del proceso con pantallas animadas; registrar variables en función del tiempo y almacenarlas en bases de datos, manipular alarmas de un proceso (registrar alarmas, reconocer alarmas, etc.). La adquisición y escritura de información de y hacia el campo, son funcionalidades básicas de cualquier sistema de control y supervisión. En resumen estos sistemas deben cumplir varios objetivos dentro de los que se destacan:

- Adquisición de datos para recoger, procesar y almacenar, en tiempo real, información relevante sobre la evolución del proceso productivo.
- Supervisión, para observar desde el monitor la evolución del proceso.
- Control, que permita modificar la evolución del proceso (consignas, alarmas, recetas, etc.) actuando por lo general a través de los equipos de control, aunque en ocasiones es posible la actuación directa sobre el proceso mediante las salidas conectadas directamente a la PC que ejecuta el SCADA.

La gran diversidad de firmas desarrolladoras de hardware y software para el control de procesos, unida a la existencia de gran cantidad de estándares de comunicación, obliga al diseño lo más flexible y abierto posible que permita la integración, sin mayores modificaciones, de estándares de comunicación.

Los SCADAs deben poseer un conjunto de manejadores (drivers) de comunicación que son módulos de software que sirven para "hablar" con esos dispositivos de campo; de la misma manera que un sistema operativo tiene *drivers* para interactuar con los dispositivos de entrada/salida como el mouse, teclado, impresora, etc.

Los *drivers* de un SCADA ejecutan dos tareas fundamentales: leer y escribir información de los dispositivos de campo. La comunicación con los dispositivos de campo, los cuales a su vez adquieren datos del proceso y realizan control del mismo, es uno de los componentes fundamentales del software de supervisión, ya que es a través de este componente que se nutren de información en tiempo real.

En estas comunicaciones son fundamentales dos aspectos, el medio físico y el lenguaje o protocolo a utilizar, dado que los tipos de Dispositivos de Adquisición de Datos o DAD (RTU, PLC, Controlador, etc.) son muy variados y los protocolos que ellos pueden manejar son muchos.

1.1 Modelos de comunicación con los dispositivos de campo

El concepto de manejador, es el de un intérprete y por lo tanto se requiere de uno por cada tipo diferente de protocolo o lenguaje, en un inicio existían tantos protocolos como dispositivos de adquisición de datos, lo cual complicaba el desarrollo del software SCADA ya que debían desarrollar un manejador por cada dispositivo que se creaba.

La dificultad de poder mantener esto en el tiempo llevó tanto a los fabricantes de equipos como a los de software SCADA a utilizar estándares, donde sobresale el MODBUS por su sencillez, su amplia difusión y por tratarse de un protocolo abierto disponible a cualquier usuario. La desventaja de esto es que la potencialidad de comunicación del DAD se ve limitada por las características propias de este protocolo.

En general en la mayoría de los manejadores se utiliza el método de direccionamiento a través de identificación de dispositivo a través de un N° de dispositivo y de una identificación del dato a través de un registro.

Los manejadores además de generar la conversión de la información que se envía y se recibe de los dispositivos de campo, informan sobre el estado de estas comunicaciones y en algunos casos sobre estadísticas de los mismos.

Existen drivers que funcionan solo bajo el modelo de Punto – Multipunto con el sistema de encuesta basado en tiempo (por Ejemplo MODBUS, DF1) y otros que permiten utilizar tanto en modo por Excepción, es decir los dispositivos esclavos pueden enviar mensajes en caso de alarmas o eventos importantes (por ejemplo ROC, MODSCAD, BISAP, DNP3). Por otro lado también hay drivers que utilizan modelo Cliente – Servidor (DATA HIGHWAY+).

A continuación se presentan algunas características del estándar para el intercambio de información orientado al control de procesos: OPC, proviene de las siglas de *OLE for Proccess Control* (OPC Foundation, 2009). OPC establece que los fabricantes de cada dispositivo desarrollan un Servidor OPC, el cual por un lado se comunica con su dispositivo y por el otro es capaz de responder o recibir información en un formato estándar OPC. La ventaja de este método es que permite con un solo interpretador de OPC leer cualquier dispositivo que tenga asociado un OPC Server, con lo cual toda actualización o incorporación de nuevos dispositivos será fácil y se ampliará el ciclo de vida de los paquetes.

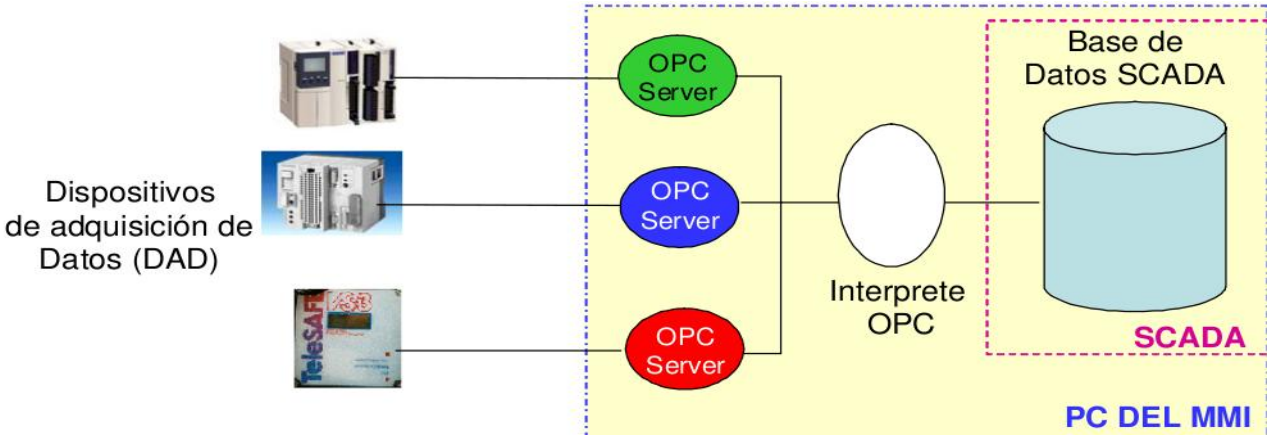

Ilustración 1. Servidores OPC.

En realidad OPC es un conjunto de protocolos entre los que podemos destacar los siguientes (OPC Foundation, 2009):

- OPC-DA (Data Access)- El original, sirve para el intercambio de datos a tiempo real entre servidores y clientes.
- OPC-AE (Alarms & Events)- Proporciona alarmas y notificaciones de eventos.
- OPC B (Batch)- Útil en procesos discontinuos.
- OPC DX (Data eXchange)- Proporciona interoperabilidad entre varios servidores.
- OPC HDA (Historical Data Accsess)- Acceso histórico a datos OPC.
- OPC S (Security)- Especifica cómo controlar el acceso de los clientes a los servidores.
- OPC XML-DA (XML Data Access)- Sirve para el intercambio de datos entre servidores y clientes como OPC-DA, pero en vez de utilizar la tecnología COM/DCOM, utiliza mensajes SOAP (sobre HTTP) con documentos en XML.
- OPC CD (Complex Data)- Permite a los servidores exponer y describir tipos de datos más complicados en forma de estructuras binarias y documentos XML.

Debido a la existencia de diversos modelos de comunicación con los dispositivos de campo, el *framework* para el desarrollo de manejadores fue diseñado e implementado de forma tal que soporta la comunicación entre el SCADA y cualquiera de los modelos de comunicación explicados.

1.2 Arquitectura de los manejadores del SCADA

La función principal de los *drivers* del sistema SCADA Guardián del ALBA es enlazar el SCADA con los diferentes dispositivos del campo. La interacción entre los manejadores y los dispositivos se efectúa mediante el envío y recepción de mensajes a través de un medio físico determinado. Los manejadores se encargan de la adquisición de los datos traduciendo los protocolos de campo a un protocolo genérico

(Interfaz Genérica). Se permite la programación de estos en módulos independientes al sistema para evitar la recompilación del sistema ante incorporaciones de nuevos protocolos.

Los protocolos de comunicación con los dispositivos definen la semántica de los mensajes, y su estructura dejando la trasmisión de los mismos a capas inferiores de transporte (por ejemplo Ethernet para TCP/IP o RS-232 para comunicación serie). El conjunto de los manejadores del sistema SCADA Guardián del ALBA soporta una interfaz general que posibilita la conexión de los diferentes manejadores con independencia de su naturaleza. La eficiencia de dicha interfaz permite un aprovechamiento máximo de las posibilidades de cada protocolo y contribuye a que el diseño del manejador sea simple. Nuestros *drivers* quedan conformados por un sistema multicapa, donde sólo sería necesario desarrollar las capas más cercanas al dispositivo que dichos *drivers* controlarían.

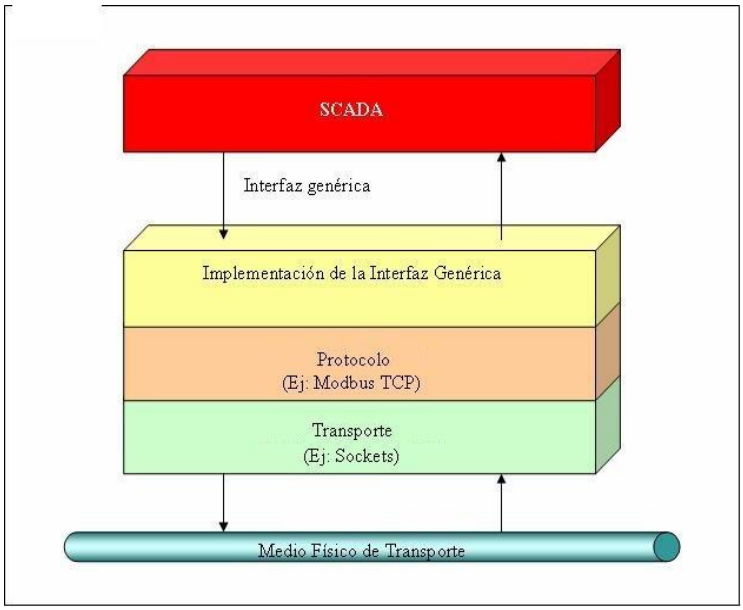

Ilustración 2. Distribución en capas de los manejadores.

La capa de transporte es la encargada de manejar la conexión y la trasmisión de un flujo de datos a través del medio físico. La capa de protocolo proporciona una serie de funciones que encapsulan la construcción e interpretación de los mensajes necesarios para la comunicación. Dicha biblioteca debe utilizar la capa de transporte sin entrar en las especificidades de cómo se envían o reciben los mensajes. Por último la implementación de la interfaz genérica (DriversCore) se traduce en términos de llamadas a la biblioteca del protocolo.

Esta arquitectura presenta las siguientes ventajas (Trujillo Codorniu, y otros, 2008):

- Las dos primeras capas de los manejadores son reutilizables y tienen valor por sí mismas. En particular disponer de la capa de protocolo permite modificar de manera más simple la interfaz genérica en caso necesario.
- Cada capa tiene una funcionalidad lógica propia. La capa de transporte envía y recibe mensajes, la capa del protocolo construye e interpreta los mensajes y la capa de implementación traduce la interfaz genérica en términos del protocolo. A partir de esta separación lógica es posible arrancar en paralelo los trabajos en las tres capas, siempre y cuando estén claras las interfaces correspondientes. La puesta a punto de las dos primeras capas no requiere de la funcionalidad del resto de los módulos del SCADA.

2. Aporte a la solución.

El trabajo realizado es la base para el desarrollo de manejadores del sistema SCADA “Guardián del ALBA”. Los manejadores creados a partir del DriversCore permiten acceder y controlar dispositivos de automatización industrial. Recordemos que el proceso de intercambio de información entre el nivel de supervisión y el nivel de campo se establece comúnmente a través de los protocolos de comunicación, los cuales establecen el “idioma” y las reglas para entablar una comunicación entre los dispositivos de campo, sensores, actuadores y el nivel de supervisión.

Para solventar los requerimientos necesarios para la conexión con los dispositivos de campo el sistema SCADA Guardián del ALBA estructura su adquisición con un módulo de recolección que integra la lógica de escritura y lectura en “tiempo real”, de la información de campo a través de los manejadores de protocolos.

Es importante señalar que el trabajo realizado por el autor se ajusta a la arquitectura definida para el proyecto SCADA “Guardián del ALBA”:

1. Todos los componentes de Software fueron desarrollados en Software Libre y bajo estándares abiertos, acotados dentro de los contemplados en la Arquitectura de AIT y la UCI. Se usó como sistema operativo para el desarrollo a Debian GNU/Linux.
2. Se aplicó el Rational Unified Process (RUP) como metodología de desarrollo de Aplicaciones y UML como notación.

3. Durante la implementación de los componentes se siguió estrictamente el estilo de código definido por el proyecto y las normas para la documentación del código fuente utilizando el doxyen.

2.1 Subsistemas del DriversCore

A continuación se exponen los subsistemas en que se organizó el conjunto de clases (DriversCore) que encapsulan diseños abstractos de soluciones a los problemas existentes en relación al desarrollo de manejadores. Los objetivos principales que persigue este framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones de diseño:

2.1.1 Capa de Transporte: Implementación del componente de software que permite comunicar a los manejadores con los dispositivos utilizando diferentes vías como por ejemplo: la comunicación serial (RS-232, RS-485, etc.), la comunicación vía TCP/IP y una variante denominada serie enviada por TCP/IP (*terminal server o conversores de Ethernet a Serial)*. Para la implementación de este componente se utilizaron bibliotecas de software libre, por ejemplo: para la comunicación TCP/IP se utilizó la biblioteca de transporte Asio C++ (Asio es una librería multiplataforma para la implementación de aplicaciones de red, que brinda un desarrollo constituido por un mecanismo asíncrono de flujos de entrada y salida, usando un moderno enfoque de C++. Entre la amplia gama de funcionalidades sobre la red que brinda podemos destacar las siguientes: soporte para la resolución de direcciones, aceptación de nuevas conexiones, socket orientados a datagrama y funcionalidades de temporizador. (Kohlhoff, Christopher M, 2008)); la biblioteca QExtSerialPort para la comunicación serial. La capa de transporte incluye las siguientes funcionalidades:

- Permite la configuración de los parámetros de comunicación. En el caso de las comunicaciones seriales permite configurar: Nombre del Puerto (ejemplo: /dev/ttyS0 en Linux o COM1 en Windows), paridad, bits de parada, tamaño del dato, control de flujo y velocidad (BaudRate). En el caso de las comunicaciones a través de TCP/IP y las que son hacia terminal server permite configurar: la dirección IP, el puerto. Además en todas las variantes es posible configurar el tiempo de espera máximo para conexión y el tiempo de espera máximo para la lectura.
- Permite realizar de forma eficiente la conexión al medio físico, lectura y escritura de datos.

2.1.2 Capa de Utilitarios: Definición de tipos básicos y constantes usadas en todos los manejadores e incluidas en el DriversCore por ejemplo:

- Las banderas que identifican las capacidades del manejador.

- La palabra de calidad a usarse para las posibles fallas de comunicación asociada a cada dispositivo de control y subcanal de comunicación, para ser usada en las instancias superiores para tipificar los tipos de alarmas asociadas a estas entidades.
- Los posibles eventos de comunicación que puede provocar la llamada a la función de activación del envió de tramas hacia el Sniffer (El Analizador de Tramas para Sistemas SCADA, más conocido como Sniffer, es una herramienta para realizar un monitoreo constante del tráfico a través de la red y la recolección al igual que el reporte estadístico de datos importantes para conocer el estado de las comunicaciones de un sistema SCADA con los dispositivos del campo).
- Los posibles tipos de datos en el dispositivo que debe tener una variable del SCADA.
- Los atributos necesarios para leer (o escribir) valores desde (o hacía) una variable en el dispositivo.

2.1.3 Capa IGC2Cpp: Define la interfaz de los drivers en lenguaje C. Se implementa además el patrón de diseño adaptador (*adapter*), para realizar la traducción de las llamadas que hace el Recolector del SCADA a dicha interfaz; en pedidos a los métodos de la jerarquía de clase subyacente que fue implementada usando el lenguaje C++.

2.1.4 Capa Común: Implementación de funciones útiles a varias capas del núcleo del DriversCore, por ejemplo:

- La función que proporciona cantidad de milisegundos transcurridos desde el comienzo de la época Unix (01/01/1970) hasta el instante de tiempo (hora Universal) en que se invoca al método.
- La función que obtiene el valor de una variable a partir de los valores obtenidos desde el dispositivo de cada una de las direcciones simples que integra la dirección.
- La implementación de IAddress que encapsula el concepto de dirección de una variable.

2.1.5 Capas igCpp y igCppImpl: Incluyen varias áreas funcionales del núcleo de los manejadores entre las que podemos destacar:

- Introspección de manejadores y dispositivos. Esta función se refiere a la capacidad de los objetos de mostrar las propiedades que expone y la capacidad de obtener el valor de esas propiedades a partir del nombre de la misma y modificar ese valor igualmente a partir del nombre. Añadiremos a esa definición la capacidad de mostrar los parámetros de diagnostico y obtener los valores de esos parámetros a partir del nombre del mismo.
- Manejo de las direcciones y clasificación de las variables en bloques de lectura o escritura. Los bloques son un conjunto de variables que tienen un mismo periodo de encuesta y que pueden

recuperarse o escribirse al PLC en una operación atómica del Protocolo. Es requerimiento del SCADA que los drivers realicen las operaciones de lectura de forma atómica. Se asumen en el núcleo de los manejadores los siguientes presupuestos: 1) Las direcciones de protocolo pueden ser ordenadas de manera estricta (d1 ≤ d2 ≤ d3 ≤ … ≤ dn), 2) Dicho ordenamiento es compatible con los bloques en el siguiente sentido: si "di ≤ dj" y las direcciones "di" y "dj" pueden ser recuperadas en una operación atómica entonces si existe "d" tal que "di ≤ d ≤ dj" la dirección "d" puede ser recuperada en la misma operación atómica que "di" y "dj" (los bloques son conexos).

2.2 Carga y manejo de la biblioteca dinámica

Las bibliotecas dinámicas (en el sistemas operativo Windows se les asigna la extensión "dll" de la sigla de Dynamic Linking Library (Bibliotecas de Enlace Dinámico) y en el sistema operativo Linux se les asigna la extensión "so" de la sigla de *shared object*) que contienen manejadores de dispositivo deberán exportar una función con el nombre "InitLibrary". La función devuelve un puntero a la estructura InitLibraryRec donde se almacenan los puntos de entrada a cada una de las funciones de la biblioteca. Es responsabilidad de la aplicación que usa la biblioteca cargar la misma, localizar la dirección del símbolo exportado "InitLibrary" e invocar a la función exportada para obtener los puntos de entrada de todas las funciones. Para ello puede usar el tipo InitLibraryFunc. Un ejemplo hipotético podría ser el siguiente (Trujillo Codorniu, 2007a):

Ejemplo para resolver el símbolo InitLibrary

```
void* lib_handle;        /* handle a la librería
InitLibraryFunc initlibrary
lib_handle = dlopen("/scada/drivers/modbus", RTLD_LAZY);
if (!lib_handle) {
    fprintf(stderr, "Error during dlopen(): %s\n", dlerror());
    exit(1);
}
initlibrary = dlsym(lib_handle, "InitLibrary");
```

Ilustración 3. Carga y manejo de bibliotecas dinámicas.

Para ilustrar mejor el proceso que debe realizar el recolector del SCADA para cargar dinámicamente las bibliotecas que contienen a los manejadores mostraremos el siguiente diagrama de secuencia:

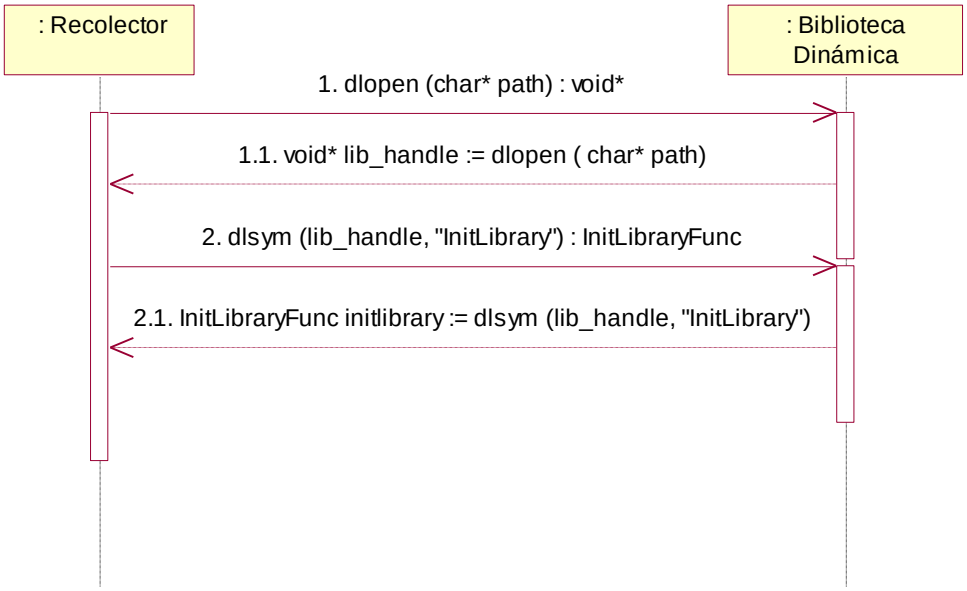

Ilustración 4. Diagrama de secuencia del proceso para cargar bibliotecas dinámicas.

2.3 Manejador Demo

Una vez concluida la implementación del DriversCore, fue necesario desarrollar el manejador “DEMO”. El propósito fue proporcionar a los desarrolladores del software de recolección del proyecto SCADA un prototipo que permita poner a prueba el enlace con la interfaz genérica de los manejadores. La jerarquía de clases sirve como guía para la creación de nuevos manejadores. La creación de nuevos manejadores puede realizarse simplemente implementando algunas funciones virtuales en la mencionada jerarquía de clases. El manejador demo (*DriverDemo*) ha servido como guía para el desarrollo de manejadores de dispositivos y ha resultado una excelente herramienta para las pruebas de arquitectura del SCADA “Guardián del ALBA”.

El diseño del dispositivo virtual “Demo” (Trujillo Codorniu, 2007d) se ha hecho de manera que simule el comportamiento real de algunos protocolos, de manera que puedan probarse las características principales de la interfaz genérica. Contiene registros que pertenecen a 4 categorías diferentes. Registros discretos de entrada, registros discretos de entrada / salida, registros analógicos de entrada y registros analógicos de entrada/salida. Los registros discretos ocupan un byte de memoria y los registros

analógicos ocupan una palabra de 16 bits. Los registros de entrada, tanto discretos como analógicos, cambian su valor en el tiempo a través de una simulación. Los registros de entrada/salida mantienen de forma persistente la última información depositada en ellos, mientras el dispositivo esté en memoria.

2.4 Lugares donde está implantada la solución:

Como parte de la instalación del sistema SCADA “Guardián del ALBA” en el proyecto de la medición fiscal de la producción de PDVSA, se están utilizando los manejadores basados en el DriversCore. Estos *drivers* están recolectando datos de *PLCs* vinculados directamente a la producción en las siguientes empresas mixtas que radican en la República Bolivariana de Venezuela:

- Petroquiriquire (Punceres, Monagas)
- Petromonagas (Monagas)
- Petrocedeño (Anzoátegui).
- Petrocabrutica
- Puerto La Cruz-Anzoátegui).
- Orocual (Monagas).
- José (Puerto La Cruz-Anzoátegui).

CONCLUSIONES

El módulo de clases obtenido cumple con los requisitos planteados de intercambio de información entre el SCADA y los dispositivos de manera eficiente.

El framework DriversCore permite acelerar el proceso de desarrollo de manejadores para el SCADA, posibilita reutilizar código ya existente y promueve buenas prácticas de desarrollo como el uso de patrones de diseño.

Las bibliotecas de software libre utilizadas, permitieron la asimilación de un amplio conjunto de tecnologías que se encuentran al nivel del estado del arte, por ejemplo: la biblioteca de transporte Asio C++ permitió acceder a datos dispuestos en dispositivos a través del protocolo TCP/IP de forma fácil y eficiente; la biblioteca QExtSerialPort para la comunicación serial y las bibliotecas de Qt para obtener y establecer información de tipo en tiempo de ejecución a los parámetros de configuración.

La estructura del núcleo permite, a los desarrolladores de futuros manejadores, reutilizar de manera parcial o total los componentes obtenidos; para ello pueden hacer uso de las siguientes tres opciones:

1. Implementar la interfaz genérica directamente en Lenguaje C.
2. Programar usando el lenguaje C++.
3. Implementarla usando C++ en unión a las bibliotecas QT.

RECOMENDACIONES

Aspectos fundamentales que se recomiendan para versiones futuras del DriversCore:

- Perfeccionar los mecanismos relacionados con el tratamiento de manejadores asíncronos.
- Implementar un modelo que reduzca el acoplamiento entre la capa de protocolo y la capa de transporte.
- Desarrollar mecanismos que faciliten la redundancia en las redes de campo o en los recolectores.
- Efectuar seguimiento de las actualizaciones de la biblioteca de transporte Asio C++ cuyas versiones futuras pudieran valorarse nuevamente para su utilización.

BIBLIOGRAFÍA

Hernández Lugones, Luis Alberto. 1998. Sistema Automatizado de Diagnóstico, Supervisión y/o Control del Consumo de Energía Eléctrica en Instalaciones Industriales y de Servicio. [Word Document] 1998.

Herrera Vázquez, Moisés. 2008. Curso del Sistema Supervisor Guardián del ALBA. [Word Document] 2008.

Kohlhoff, Christopher M. 2008. Rationale. Asio C++ Library. [En línea]. [Citado el: 19 de abril de 2009.] http://think-async.com/Asio/asio-1.4.1/doc/asio/overview/rationale.html

OPC Foundation. 2009. OPC, OLE for Process Control Overview. [En línea] OPC Foundation, 2009. [Citado el: 18 de Abril de 2009.] http://www.opcfoundation.org/DownloadFile.aspx?CM=3&RI=1&CN=KEY&CI=282&CU=20

Trujillo Codorniu, Rafael. 2007. Especificación de la interfaz genérica con el SCADA. [Word Document] Anexo 13 del convenio Marco PDVSA ALBET, 2007a.

—. Especificación de Requerimientos de Software e interfaces públicas del Sniffer. [Word Document] Anexo 31 del convenio Marco PDVSA ALBET, 2007b.

—. Especificación de Requerimientos de Software del Terminal Server. [Word Document] Anexo 31 del convenio Marco PDVSA ALBET, 2007c.

—. Guía de Implementación del Manejador Demo. [Word Document] Anexo 13 del convenio Marco PDVSA ALBET, 2007d.

Trujillo Codorniu, Rafael; García Hernández, Luis y Cedeño Pozo, Antonio. 2008. Interfaz genérica para los manejadores de dispositivos en el proyecto SCADA PDVSA. [PDF] 2008.

GLOSARIO

ASCII: (acrónimo inglés de American Standard Code for Information Interchange). Es un código estándar para el Intercambio de Información. Utiliza 7 bits para representar los caracteres, aunque inicialmente empleaba un bit adicional (bit de paridad) que se usaba para detectar errores en la transmisión.

Bus de campo: Es un solo enlace de comunicaciones, al cual se conectan directamente todos los dispositivos. Existen dos formas de comunicación en esta topología, colisión y maestro/esclavo.

Controlador Lógico Programable (PLC): dispositivos electrónicos usados en automatización industrial para realizar estrategias de control básicas. Por su robustez y características sencillas de control, están cercanas al proceso, permitiendo ejecutar las tareas básicas del control, aun cuando no tenga conexión a las capas superiores del control.

Framework: En los sistemas orientados a objeto un framework es un conjunto de clases que encapsulan diseños abstractos de soluciones a un determinado números de problemas en relación. Los objetivos principales que persigue un framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones.

Protocolo Ethernet-IP: (Ethernet/Industrial Protocol) es un sistema de comunicación diseñado para entornos industriales. EtherNet-IP permite que los dispositivos de control intercambien información crítica con requerimientos estrictos de tiempo.

Protocolos de comunicación: son como reglas de comunicación que permiten el flujo de información entre computadoras o dispositivos diferentes que manejan lenguajes distintos.

Sistemas de Adquisición de Datos y Control Supervisorio (SCADA): Aplicación de software especialmente diseñada para funcionar sobre computadores en el control de producción, proporcionando comunicación con los dispositivos de campo y controlando el proceso de forma automática desde la pantalla del operador, proveyendo toda la información asociada al proceso.

Software libre: Las cuatro reglas esenciales que deben cumplir las aplicaciones para ser consideradas como software libre son:

- Libertad 0: Libertad de ejecutar el programa como quieras.
- Libertad 1: Libertad de estudiar el código fuente y cambiarlo para realizar lo que desees.
- Libertad 2: Libertad de realizar copias y distribuirlas cuando quieras.
- Libertad 3: Libertad de distribuir o publicar versiones modificadas cuando desees.

TCP/IP: Familia de protocolos sobre los cuales funciona Internet, que se ha convertido en el estándar actual de comunicación entre computadoras. Conocida por estas siglas debido a que los dos protocolos más importantes son: el protocolo IP, que se ocupa de transferir los paquetes de datos hasta su destino adecuado y el protocolo TCP, que se ocupa de garantizar que la transferencia se lleve a cabo de forma correcta y confiable.

Trama: Es una unidad de envío de datos. Viene a ser sinónimo de paquete de datos.

Dispositivo: Se denomina dispositivo a un elemento de hardware que alberga información de proceso o estado de sí mismo, que forma parte de un sistema automatizado. Comúnmente los dispositivos son Controladores Lógicos Programables, Computadoras Industriales, Sistemas de Control Distribuidos, sensores o actuadores inteligentes con capacidad de comunicación.