Congreso Internacional en Ingeniería Electrónica. Mem. Electro 2015, Vol. 37, pp 19-24, Chihuahua, Chih. México
http://depi.itchihuahua.edu.mx/display/memorias_electro/MemoriaElectro2015
ISSN 1405-2172

Plataforma Plug and Play para Células de Manufactura Basada en el Estándar ISO 9506

Carreón Espinoza Victor David Acosta Cano de los Ríos José Eduardo González Gurrola Luis Carlos

Universidad Autónoma de Chihuahua
Facultad de Ingeniería
Circuito Universitario Campus II
Chihuahua, Chih. México
victorcarreone@gmail.com

RESUMEN

Se presenta el desarrollo de una plataforma débilmente acoplada basada en un arquitectura de referencia con la finalidad de ofrecer una solución a los problemas de integración equipo de manufactura/Software de control. En el desarrollo de la plataforma se aplica el concepto de acoplamiento débil y tecnologías de cero configuración como UPnP (*Universal Plug and Play)*, Servicios WCF (*Windows Communication Foundation),* y elementos del estándar ISO 9506 -MMS-. La plataforma así desarrollada es validada con la implementación de una célula, observando flexibilidad tanto el proceso de fabricación a realizar como en la modificación del conjunto de equipos que forman a la célula.

Palabras Clave: Sistemas flexibles de manufactura, UPnP, WCF, ISO 9506, acoplamiento débil.

ABSTRACT

The development of a loose coupled platform based in a reference architecture is proposed that addresses the integration problem of manufacturing equipment to Control software. The solution is based on the loose coupling concept using zero configuration techniques such as UPnP (*Universal Plug and Play)*, WCF (*Windows Communication Foundation)* Services, and elements from the ISO 9506 standard. The proposed solution is validated through a manufacturing cell development implemented by real manufacturing equipment. The solution support not only process flexibility but also equipment modification flexibility.

Keywords: Flexible manufacturing systems, UPnP, WCF, ISO 9506, Loose Coupling.

1. INTRODUCCIÓN.

La fabricación de la mayoría de los productos requiere aplicaciones de control, programación de dispositivos de piso de producción, y mecanismos para coordinar el orden en que se llevan a cabo las operaciones de fabricación. Las condiciones actuales de un mercado turbulento impone requerimientos de flexibilidad al piso de producción. Entre los tipos de flexibilidad requerida para enfrentar tales condiciones se encuentra la facilidad para el cambio de procesos y en ocasiones de equipo. En esta dirección existen varias propuestas basadas en tecnologías de información tradicionales para su aplicación en el ámbito industrial, siguiendo estándares como IEC 62264 [1], en dichas propuestas se hace énfasis en el uso de una arquitectura orientada a servicios y la implementación de protocolos para descubrir dispositivos conectados en la red y los diversos servicios que ofrecen. En [2] se presenta una plataforma de software basada en tecnologías que son empleadas en desarrollos de software tradicionales, como arquitecturas orientadas a servicios (SOA's), servicios web y comunicaciones Ethernet aplicando protocolos ampliamente difundidos y aceptados tales como TCP/IP.

Una manera de facilitar la configuración y la integración de los dispositivos es mediante una arquitectura que exponga los servicios que ofrece cada dispositivo. En [3] se hace uso de un *"módulo interconector"* que detecta un dispositivo nuevo y pone a disposición del usuario los servicios que la célula de manufactura es capaz de realizar con la configuración actual. Esta detección y descripción de servicios puede ser implementada por alguna de las arquitecturas *conectar y operar* que existen, como las descritas en [4] *Jini, DPWS (Devices Profile for Web Services), UPnP (UPnP)* empleadas para descubrir dispositivos y los servicios que ofrecen en una red.

Algunas de las tecnologías existentes que ofrecen la capacidad de encontrar dispositivos en un sistema y exponer los servicios que pueden brindar, sin la necesidad de realizar una configuración manual, se encuentran en diferentes trabajos reportados en la literatura [5], [6], [7], [8]; donde hacen uso de plataformas middleware y SDK's (*Software Development Kits*) en robots.

En el presente artículos se plantea el desarrollo de una arquitectura que permita encapsular e integrar dispositivos físicos como instrumentos de medición, PLC's (*Programmable logic controller)*, robots, etc., haciendo uso de estándares denominados de cero configuración. Se ofrece una solución basada en conceptos de acoplamiento débil, que permite la incorporación de equipos de fabricación, [9].

Del análisis de las alternativas de arquitecturas de cero configuración disponibles (*JINI, DSSP – Decentralized Software Services Protocol, DPWS – Devices Profile for Web Services , UPnP – Universal Plug and Play*) se tomó la decisión de utilizar *Universal Plug and Play* debido a que cuenta con mayor soporte, es libre de plataforma,

independiente del lenguaje de programación y hace uso de estándares como XML (*extensible markup language)*, HTTP (*hypertext transfer protocol*), HTML (*Hypertext Markup Language)*, TCP/IP que son utilizados para el desarrollo de aplicaciones tradicionales y comunicaciones. Mediante tecnologías como *UPnP* es posible describir los servicios que cada uno de los dispositivos puede realizar, y a través de puntos de control manipular y conocer el estado de los dispositivos que componen el sistema. Para fortalecer el aspecto industrial de la arquitectura planteada se utilizaran algunos elementos del estándar ISO 9506 referente a la especificación de mensajes de manufactura o MMS.

El presente documento se organiza de la siguiente manera: en la sección 2 se encuentra el marco teórico. La sección 3 se enfoca en el desarrollo del proyecto y muestra la aplicación de las diferentes tecnologías mencionadas en el marco teórico para llevar a cabo el proyecto. En la sección 4 se presenta la validación de la arquitectura. En la sección 5 se muestran los resultados, seguidos de la sección 6 donde se encuentran las conclusiones.

2. MARCO TEÓRICO.

2.1. Protocolos de cero configuración.

Los protocolos de descubrimiento de servicios (SDP – *"Service Discovery protocols")* hacen posible que la coordinación y la interacción entre los dispositivos de una red sea llevada a cabo de una manera simple y escalable [10].

La arquitectura que se va a adoptar debido a que cuenta con más soporte y documentación es *Universal Plug and Play* de Microsoft, también se hará uso de las herramientas de Intel para UPnP (*Developer Tools for UPnP Technologies*), las cuales facilitan la creación de descripciones de servicios para los dispositivos necesarios.

Universal Plug and Play.

La arquitectura UPnP (*Universal Plug and Play*) es un desarrollo de Microsoft que permite la conectividad de computadoras y/o dispositivos inalámbricos dentro de una red, UPnP es una arquitectura abierta y se apoya del protocolo TCP/IP (*Transmission Control Protocol / Internet Protocol)* y tecnologías web para su funcionamiento. Debido al uso de estándares es posible implementar la arquitectura en cualquier lenguaje de programación y sistema operativo. Esta arquitectura cuenta con soporte y documentación como son las las herramientas de Intel para UPnP (*Developer Tools for UPnP Technologies*), las cuales facilitan la creación de descripciones de servicios para los dispositivos necesarios.

Dentro de la especificación de UPnP se pueden encontrar dos entidades fundamentales: Dispositivos y Puntos de control.

Dispositivos y Puntos de Control.

Es una entidad de la arquitectura que provee información descriptiva de sí misma, como la información del fabricante, modelo, número de serie, etc. Dentro del dispositivo se cuenta con un conjunto de servicios, los cuales definen la funcionalidad del dispositivo. En la Figura 1 se muestra el diagrama de clases del dispositivo UPnP. Para identificar los diagrama de clases. Para identificar los servicios cada uno de estos cuenta con un URI (*Uniform Resource Identifier*) llamado serviceId, que es el identificador único de cada servicio. Cada servicio también cuenta con tres URLs (*Uniform Resource Locator)* que proporcionan la información necesaria para poder ser controlados por los puntos de control.

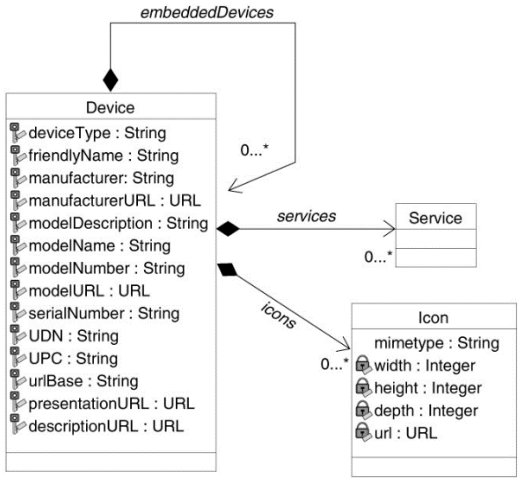

Figura 1 - Diagrama de Clases de Dispositivo UPnP.

Los puntos de control de la arquitectura UPnP son la entidad capaz de consumir los servicios que ofrecen los dispositivos, también es posible subscribirse al servicio para conocer los cambios de estado de los dispositivos mediante las variables de estado.

2.2. ISO 9506 Manufacturing Message Specification.

Es un estándar internacional cuyo objetivo principal es ofrecer un mecanismo de comunicaciones entre equipos o dispositivos de fabricación y aplicaciones. Los servicios que MMS ofrece son los suficientemente genéricos de manera que pueden aplicarse a una gran variedad de dispositivos industriales, [11].

La estructura de MMS está basada en una arquitectura cliente-servidor, donde el servidor o servidores están representados por un objeto llamado *Virtual Manufacturing Device (VMD)*, el cual a su vez es un contenedor de otros objetos como variables, dominios, invocaciones de programa, eventos, semáforos y registros de actividad.

El estándar tiene una estructura orientada a objetos donde el principal objeto definido por el estándar es el dispositivo de manufactura virtual o *VMD (Virtual Manufacturing Device)*. El objeto VMD es un contenedor donde se encuentran contenidos otros objetos de MMS, Figura 2. Por medio de las llamadas a servicios el cliente es capaz de accesar los objetos en el servidor (VMD). Para el desarrollo del trabajo fueron

utilizados dos de los objetos especificados por el estándar MMS, dichos objetos son: Domain y ProgramInvocation.

Tales objetos establecen atributos y acciones para manejar la ejecución de un programa en los dispositivos virtuales de manufactura. El objeto Domain representa un recurso en memoria de un dispositivo físico, mientras que el objeto ProgramInvocation se conforma por diversos objetos domain y establece estados y acciones para la manipulación de la ejecución de programas. En la Figura 3 puede observarse el diagrama de estados definidos para el objeto ProgramInvocation.

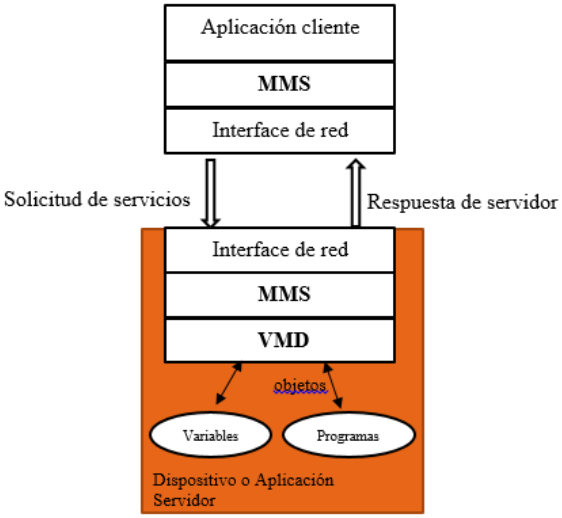

Figura 2 - VMD (Virtual Manufacturing Device)

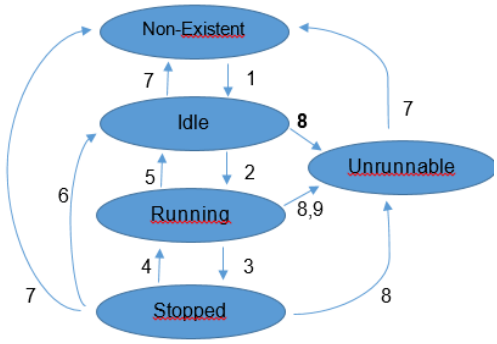

Figura 3 - Estados de objeto ProgramInvocation

3. DESARROLLO DE LA SOLUCION PROPUESTA.

Tomando como referencia la arquitectura planteada en [12], se desarrollaron los diferentes objetos de la arquitectura, Figura 4, utilizando las tecnologías *universal plug and play*, WCF (*Windows communication foundation)*, y el estándar ISO 9506 MMS.

En el nivel inferior del esquema se encuentra el objeto propulsor, este objeto es el encargado de comunicarse directamente con el dispositivo físico, en él se establecen los mecanismos necesarios para realizar una comunicación exitosa, como son métodos para poder enviar y recibir mensajes al dispositivo y el código necesario para poder manejar el medio físico de comunicación con el que cuenta el controlador del dispositivo.

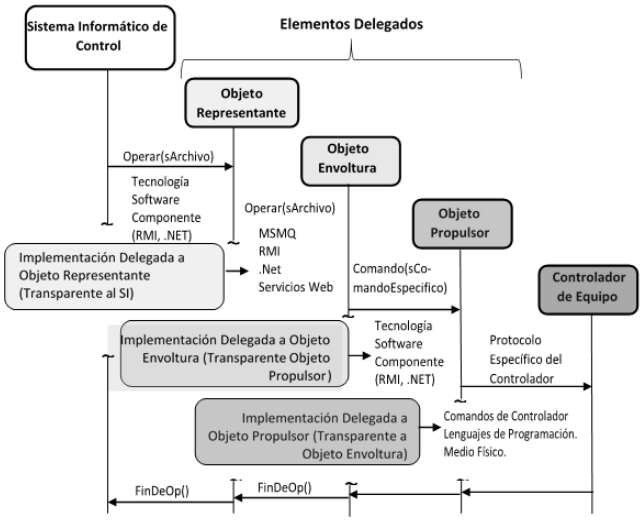

Figura 4 - Arquitectura de referencia [12]

En el nivel inmediato superior se encuentra el objeto envoltura que se encarga de encapsular al objeto propulsor y a su vez es el punto de entrada de comunicación disponible para que el software de control a través del objeto representante sea capaz de enviar los comandos de operación. Estos dos elementos de la arquitectura están basados en la misma plataforma de desarrollo, por lo que es posible el uso de librerías dinámicas para el desarrollo del objeto propulsor, lo cual facilita incorporarlo al objeto envoltura.

En la parte superior se encuentran dos elementos más, el software de control y el objeto representante, se propone que estos dos elementos se implementen en la misma computadora basados en la misma plataforma o framework de desarrollo; con la finalidad de simplificar la comunicación entre ellos. Los comandos de operación son enviados desde el software de control, recibidas por el objeto representante para posteriormente ser enviadas hacia el objeto envoltura. La comunicación entre los objetos representante y envoltura puede darse entre diferentes plataformas, por esto es recomendable que la comunicación entre ellos sea mediante un protocolo independiente de plataforma con la finalidad de brindar una mayor flexibilidad.

3.1. Programación de instrumentos.

Uno de los principales requerimientos consiste en la capacidad de incorporar diferentes tipos de dispositivos al sistema informático de control. Un tipo común de dispositivos son los instrumentos de medición. Los instrumentos de medición presentan diferentes interfaces de comunicación, lo cual es una característica que debe satisfacer la solución aquí propuesta.

La integración de los instrumentos de medición en la arquitectura UPnP requiere establecer una comunicación entre la computadora y el instrumento, dicha comunicación puede realizarse por medio de VISA (*Virtual Instrument Software Architecture*). VISA es una arquitectura que permite la programación y configuración de sistemas de instrumentación, y soporta distintos medios de comunicación como GPIB (*General Purpose Instrumentation Bus)*, Serial, Ethernet o USB.

3.2. Desarrollo del objeto propulsor.

Mediante el uso del IDE Visual Studio 2012, el Framework .Net 4.5 y el lenguaje de programación C#, se desarrolló el objeto propulsor en un proyecto tipo librería de clases, dichas clases contienen los métodos y mecanismos para realizar la comunicación con el instrumento. El objeto envoltura integra al objeto propulsor en tiempo de ejecución por medio de técnicas de reflexión que ofrece el framework .Net (incorporación dinámica de código).

Las instrucciones de operación de los dispositivos son especificadas mediante un archivo de texto, Figura 5, en el cual se indica el dispositivo al que se desea enviar el comando mediante su dirección VISA y las acciones a realizar por medio de comandos SCPI.

```
?ASRL3::INSTR   <-- Dirección VISA.
REMS            <-- Comando SCPI (instrumento en
-L                  modo remoto).
*RST
-L
CONT
-L              <-- Comando para indicar que hay que leer un valor del
<MEAS1?             instrumento.
-L
->0.1
?TCPIP0::10.80.15.5::inst0::INSTR
*RST
FUNC DC
VOLTage:OFFSet 2.6
OUTP ON
?ASRL3::INSTR
REMS
-L
*RST
-L
VDC
-L
<MEAS1?
-L
->4.7
-<5.3
```

Figura 5 – Instrucciones Enviadas al dispositivo

3.3. Desarrollo del objeto envoltura.

La aplicación del objeto envoltura debe de tener la capacidad de alojar un servicio en sí misma. El framework de .Net ofrece las librerías de WCF (*Windows Communications Foundation*) para el desarrollo de aplicaciones orientadas a servicios, dichas librerías fueron implementadas en la aplicación envoltura para poder brindar sus funciones (acciones) mediante un servicio web.

El desarrollo del objeto envoltura consiste en una aplicación tipo Windows Forms con la capacidad de incluir en tiempo de ejecución el objeto driver para controlar el dispositivo físico deseado, y utiliza las librerías de WCF para exponer un servicio y manipular el dispositivo físico desde otra aplicación.
La capacidad de agregar un propulsor en tiempo de ejecución, aporta flexibilidad a la aplicación, debido a que con una sola implementación de objeto envoltura es posible incluir distintos propulsores según el equipo que se necesite utilizar.
Una vez que se obtiene la instancia de la clase propulsor se utiliza dentro de la definición del servicio del por parte del objeto envoltura manipular al dispositivo físico.

3.4. Desarrollo del objeto envoltura como Servicio WCF.

Otra implementación de la clase envoltura fue realizada como un servicio web de WCF (Windows Communications Foundation). Para alojar un servicio dentro de la aplicación es necesario incluir la referencia a System.ServiceModel de .Net la cual va a permitir hacer uso de las clases necesarias para configurar y arrancar el servicio web dentro de la aplicación.
Antes de iniciar el servidor en la aplicación se necesita definir el contrato de servicio el cual sirve para configurar su comportamiento y designar los métodos que puede llevar a cabo.

Una de las características especiales del servicio desarrollado es que el servidor tiene la capacidad de avisar al cliente cuando ha terminado su operación, en la Figura 6 se muestra el funcionamiento de dicho servicio.

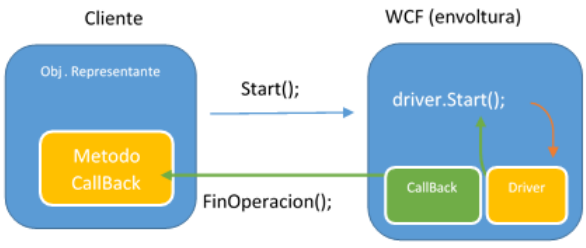

Figura 6 - Servicio WCF dúplex

3.5. Sistema de control – objeto representante.

En la parte superior de la arquitectura de referencia, se encuentra el sistema informático de control, se trata de una aplicación con una interfaz gráfica desarrollada con el

framework de .Net; es la entidad principal del sistema, y se encarga de inicializar las instancias de los objetos representantes necesarios. El sistema de control se encuentra conectado a una base de datos de SQL server de donde obtiene la información necesaria del proceso de fabricación, requerida para la creación de las instancias a utilizar y las operaciones a realizar. Para acceder a la información de la base de datos se utilizó Entity Framework de .Net, esta tecnología de acceso a datos es un ORM (Object Relational Mapping) permite manejar los objetos de la base de datos en el paradigma orientado a objetos.

Al iniciar operación, el sistema de control realiza una consulta a la base de datos para identificar los equipos disponibles y las direcciones de sus servicios, posteriormente crea una instancia del objeto representante de cada uno de ellos.

3.6. Comunicación entre objeto representante y envoltura.

Al iniciar operación el sistema de control crea e inicializa las instancias de objetos representantes necesarios para consumir los servicios de los objetos envoltura, dichos objetos representantes son los clientes que tienen la capacidad de invocar las diversas acciones expuestas por el objeto envoltura correspondiente. Durante la instanciación de los objetos representantes se recibe como parámetro la dirección en la que se encuentra el servicio que va a consumir.

Durante la etapa de desarrollo del objeto representante fue necesario agregar una referencia al servicio web utilizado por el objeto envoltura; para agregar la referencia es necesario que la aplicación del objeto envoltura y su servicio estén en ejecución, dentro del proyecto de Visual Studio de la aplicación representante. Esta referencia permite al objeto representante crear la instancia del cliente requerida para consumir las operaciones del servicio.

3.7. Universal Plug and Play.

En la arquitectura de referencia, Figura 4, se puede apreciar que la comunicación entre el objeto representante y el objeto envoltura puede realizarse utilizando diferentes tecnologías. En la sección anterior se describe la comunicación mediante servicios de WCF, a continuación se presenta otra alternativa que se basa en la arquitectura UPnP (*Universal Plug and Play).* UPnP es una arquitectura cliente-servidor donde el dispositivo UPnP toma el rol de servidor y el punto de control el del cliente. El dispositivo UPnP también incluye un servicio que define el conjunto de acciones que el dispositivo es capaz de realizar. Por lo tanto se propone utilizar el dispositivo UPnP como el objeto envoltura de la arquitectura y al punto de control como el objeto representante.

La implementación se realizó mediante las herramientas que Intel proporciona para el desarrollo de aplicaciones compatibles con UPnP (Intel Developer Tools for UPnP). El primer paso para desarrollar el objeto envoltura de acuerdo a UPnP es definir el servicio con la herramienta Service Author, Al finalizar la definición del servicio se obtiene un archivo xml de acuerdo a la especificación de UPnP. Dicho archivo se utiliza para generar el código del dispositivo UPnP, por medio de la herramienta llamada Device Builder. Una vez que se genera el código es posible ejecutar la aplicación y el dispositivo estará disponible dentro de la red UPnP, [10].

4. VALIDACIÓN.

Como parte de la validación del trabajo realizado, se desarrolló una célula de manufactura con base en la solución aquí propuesta. La función de la célula de manufactura es el ensamblado (simulado) de piezas de una tablilla electrónica y realizar una serie pruebas eléctricas.

La célula de manufactura se formó utilizando diferentes dispositivos en diferentes estaciones. En la figura 7 se muestra un diagrama de los elementos incluidos en el sistema.
El sistema cuenta con 2 estaciones y un elemento transportador, las estaciones son las siguientes:

- Estación de pruebas eléctricas. Formada por un multímetro fluke 45, generador de funciones Agilent, y un osciloscopio tektronix.
- Estación de proceso con base en un PLC Allen Bradley ControlLogix
- El elemento transportador es un robot motoman UP20 controlado por el controlador XRC2001.

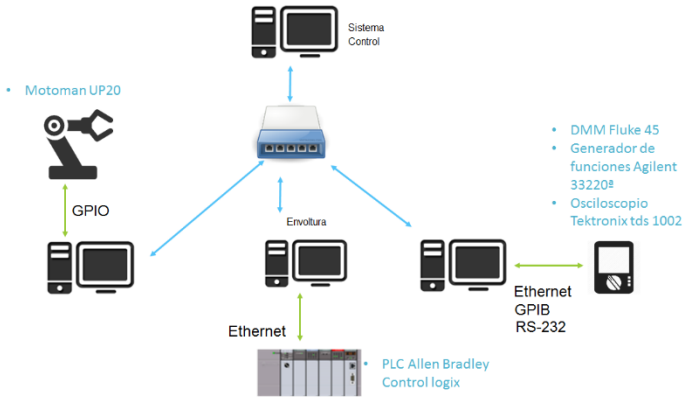

Figura 7 - Elementos del sistema

Los pasos de operación de la célula de manufactura son los siguientes:

- El sistema de control indica el inicio de operación al robot.
- Al iniciar el proceso el robot debe transportar la tablilla a la estación del PLC, en esta estación se simula que el PLC coloca los componentes en la tablilla.

- La estación del PLC anuncia el fin de operación al sistema de control.
- El sistema de control ordena transportar la tablilla a la estación de pruebas.
- La estación de pruebas recibe del sistema de control las pruebas con las que debe de funcionar.
- Al finalizar las pruebas la estación indica al sistema de control el fin de operación y si las pruebas fueron satisfactorias o fallidas.
- El sistema de control indica al transportador si la pieza es válida o fue descartada por errores en las pruebas.

Cada una de las estaciones del sistema y el elemento transportador está compuesto por los equipos y una pc; en la pc se ejecuta la aplicación envoltura que expone sus acciones a las instancias del objeto representante instanciadas por el sistema de control. Así mismo, el sistema de control envía los comandos de operación a cada dispositivo mediante los respectivos objetos representantes. En la Figura 8 se puede observar el diagrama de bloques completo de la solución desarrollada.

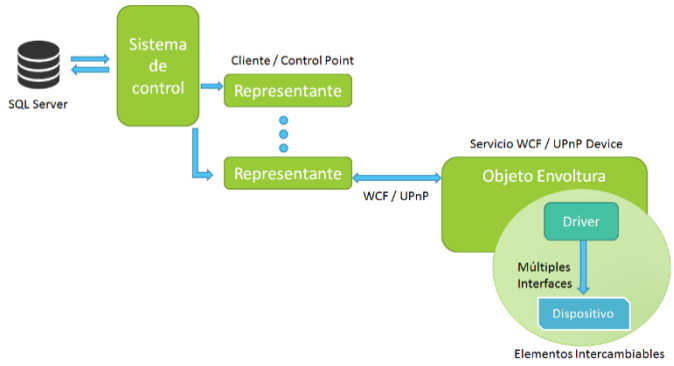

Figura 8 - Diagrama de bloques de la solución desarrollada.

5. RESULTADOS.

El resultado principal del presente trabajo de investigación es una manera de aplicar protocolos de cero configuración y servicios web, en este caso UPnP y WCF respectivamente. La arquitectura de referencia figura 4, brinda una plataforma sobre la cual se puede llevar a cabo la integración de nuevos dispositivos a una célula de manufactura de manera sistemática. Además, no solo se utilizan las tecnologías de cero configuración, sino que se aplican elementos del estándar MMS; los cuales aportan un enfoque de aplicación del estándar MMS planteado para la integración de dispositivos de manufactura.

De esta manera la célula de manufactura presenta flexibilidad para modificar su comportamiento en cuanto a las operaciones a realizar en cada estación, así como para la incorporación de nuevos equipos al sistema sin la necesidad de modificar el código fuente del software de control. Así mismo, .la solución planteada sirve de base para futuros desarrollos de células de manufactura automatizadas.

6. CONCLUSIONES Y RECOMENDACIONES.

La solución planteada soporta el desarrollo de células de manufactura automatizadas, eliminando la necesidad de modificar el código fuente del sistema de control al hacer cambios en el proceso o en la integración de nuevos equipos.

Uno de los puntos a trabajar a futuro es realizar pruebas con dispositivos en plataformas distintas a .Net, si bien se hizo el esfuerzo por utilizar arquitecturas que fueran estándar como UPnP y WCF (Servicios Web) para la comunicación entre los objetos envoltura y representante (proxy) de la arquitectura, aún es necesario desarrollar dispositivos con otras herramientas diferentes a .Net o distinto sistema operativo, para validar que en efecto se puede realizar la comunicación entre el objeto representante y objeto envoltura independientemente de la plataforma en que se encuentren.

REFERENCIAS.

[1] Stamatis Karnouskos, "A SOA-based architecture for empowering future collaborative cloud-based industrial automation," IEEE, 2012.

[2] Tommaso Cucinotta, "A Real-Time Service-Oriented Architecture for Industrial Automation," *IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, VOL. 5, NO. 3.*, 2009.

[3] Martin Naumann, "Control Architecture for Robot Cells to Enable Plug'n'Produce," in *2007 IEEE International Conference on Robotics and Automation*, Roma, Italy., 2007.

[4] A. Valera, "Aplicación de la arquitectura orientada a servicios universal plug and play para facilitar la integración de robots industriales en lineas de producción," *Revista Iberoamericana de Automática e Informática industrial*, 2012.

[5] Stjepan Sucic, "Integrating DPWS and OPC UA device-level SOA features into IEC 61850 applications," *IEEE*, 2012.

[6] Veiga G., "ON THE USE OF SERVICE ORIENTED SOFTWARE PLATFORMS FOR INDUSTRIAL ROBOTIC CELLS," Mechanical Engineering Department, University of Coimbra, Portugal. Department of Computer Science, Lund University, Sweden, 2007.

[7] Chul Ahn, "UPnP SDK for Robot Development," in *SICE-ICASE International Joint Conference* , Bexco, Busan, Korea, 2006.

[8] Sang Chul Ahn, "UPnP Approach for Robot Middleware," in *Robotics and Automation, 2005. ICRA 2005*, 2005.

[9] Orton y Weick, "Loosely Coupled Systems: A Reconceptualization," *The Academy of Management Review*, pp. 203-223, 1990.

[10] S. Helal, "PROTOCOLS FOR SERVICE DISCOVERY IN DYNAMIC AND MOBILE NETWORKS," *International Journal of Computer Research*, pp. 1-12, 2002.

[11] I. Systems Integration Specialists Company, "Overview and Introduction to the Manufacturing Message Specification," 1995.

[12] J. Acosta-Cano y F. Sastrón Báguena "Loose Coupling Based Reference Scheme for Shop Floor-Control System/Production-Equipment Integration," *Journal of Applied Research and Technology*, pp. 447-469, 2013.