risti
Revista Ibérica de Sistemas e Tecnologias de Informação
Revista Ibérica de Sistemas y Tecnologías de Información

Recebido/Submission: 16/11/2021
Aceitação/Acceptance: 27/01/2021

Control PID didáctico con Servidor OPC UA

Fernando Rivera-Velazquez[1], Eduardo Salazar-Valle[1], Gloria M. Martínez-Aguilar[1]

FernandoRiveraUTT@outlook.com; esalazar@utt.edu.mx; gmartinez@utt.edu.mx

[1] Universidad Tecnológica de Torreón, Departamento Mecatrónica, Carretera Torreón - Matamoros Km 10, S/N, Ejido el Águila, 27400 Torreón, Coahuila México.

DOI: 10.17013/risti.45.77–99

Resumen: El controlador Proporcional Integral Derivativo (PID) ha sido por décadas el controlador más usado en la industria debido a sus diversas ventajas, siendo una de las más importantes la relación costo/beneficio que puede proveer, además de que es una metodología que se ha ido adaptando a las nuevas tecnologías a través de los años. Actualmente con el auge de la industria 4.0 es imperativo proporcionar herramientas educacionales asequibles que permitan adentrar a los educandos en estas tecnologías. La implementación de un controlador PID en un servidor OPC UA permite eliminar las barreras de propietario y eficientizar los procesos productivos a controlar y monitorear. En este trabajo se presenta la metodología de implementación de un control PID didáctico con un servidor OPC UA montado en una Raspberry Pi, que da la oportunidad de introducir a los jóvenes universitarios a un entorno de industria 4.0 para que en su futuro profesional logren una rápida adaptación al sector productivo.

Palabras-clave: PID, OPC UA, Raspberry Pi, Arduino, UA Expert, Visual Studio.

Didactic PID control with OPC UA Server

Abstract: The Proportional Integral Derivative (PID) controller has been the most widely used controller in the industry for decades due to its various advantages, one of the most important being the cost/benefit ratio it can provide, in addition to the fact that it is a methodology that has been adapted to new technologies over the years. Currently, with the rise of industry 4.0, it is imperative to provide affordable educational tools that allow students to delve into these technologies. Implementation of a PID control in an OPC UA server allows to eliminate proprietary barriers and streamline monitoring and controlling in production processes usisng it. This paper presents the methodology for implementing a didactic PID control with an OPC UA server mounted on a Raspberry Pi, which gives the opportunity to introduce young university students to an industry 4.0 environment so that in their professional future they can achieve rapid adaptation to the productive sector.

Keywords: PID, OPC UA, Raspberry Pi, Arduino, UA Expert, Visual Studio.

1. Introducción

El controlador Proporcional Integral Derivativo (PID) emplea tres algoritmos con un propósito muy específico cada uno para llevar a cabo el control. La parte proporcional incorpora cambios proporcionales apropiados para el error (la diferencia entre el punto de referencia y la variable de proceso) a la salida de control. El término integral examina la variable del proceso a lo largo del tiempo y corrige la salida al reducir la compensación de la variable del proceso. El modo de control derivado monitorea la tasa de cambio de la variable del proceso y, por lo tanto, cambia la salida cuando hay variaciones inusuales (Borase, 2021).

Por su estructura simple, fácil implementación y mantenimiento, los controladores PID son los controladores más utilizados en control de movimiento, control de procesos, electrónica de potencia, hidráulica, neumática, e industrias manufactureras, entre otros (Åström, 2001). Además, ofrece un buen rendimiento con una relación costo/beneficio, esto ha permitido que prevalezca y vaya de la mano con tecnologías y aplicaciones modernas. En la mayoría de las aplicaciones de sistemas de control, el 90-95% de los lazos de control son PID (Díaz-Rodríguez, 2019).

En la última década el mundo industrial empezó a converger en la cuarta revolución industrial y los ejes tecnológicos que la componen (Y. Lu, 2017). Dentro de los arquetipos implementados en la industria 4.0 el Internet industrial de las cosas (IIoT) es uno de los más importantes y se compone por una red de redes que conecta equipos industriales, controladores, sensores y actuadores, es decir las "Cosas", para proporcionar alternativas inteligentes de servicios en los sistemas de fabricación, con el objetivo de mejorar la calidad, productividad, eficiencia, confiabilidad, seguridad y protección (E. Sisinni, 2018), (S. Vitturi, 2019).

Tradicionalmente, la automatización de procesos industriales se ha basado en la creación de redes de sensores y actuadores que monitorean y controlan estos mediante leyes de control que se implementan de acuerdo a la naturaleza propia del sistema. Este tipo de redes aprovechan el concepto de sistemas de medición distribuida (DMS). Con la creciente presencia del paradigma IIoT, los DMS llegan más enfocados, ya que los componentes de un sistema IIoT necesitan un nivel aún mayor de interacción para la integración de datos, sensores, comunicación y procesamiento.

Este nuevo escenario DMS habilitado para IIoT se basa en el soporte de sistemas de comunicación eficientes y confiables, que tienen que garantizar una amplia disponibilidad de los datos recopilados de posiblemente instrumentos de medición y/o sensores heterogéneos (D. Bruckner, 2019).

Sin embargo, dentro de estos sistemas es común que los componentes provengan de diferentes fabricantes lo que conlleva que en su mayoría utilicen diferentes formatos para representar la medición datos e incluso que operen intrínsecamente sobre redes heterogéneas. Esto representa una problemática para la modularidad, interoperabilidad y por ende control de los procesos. Una solución clave para este dilema es la Arquitectura Unificada (UA) de vinculación e incrustación de objetos (OLE) para Control de Procesos (OPC) (Mathias, 2020).

OPC UA es un protocolo definido según la norma internacional IEC 62541 que fue concebida para implementar la comunicación de máquina a máquina (M2M) a través de posiblemente diferentes medios físicos, al mismo tiempo que garantiza un alto nivel ciberseguridad. Por lo que es una atractiva y ventajosa oportunidad para el paradigma de medición de IIoT emergente. En particular, su estructura orientada a objetos (Baño, 2019) permite una contextualización completa de la información. Por ejemplo, un objeto OPC UA se podría usar para almacenar el valor de una medición, las características del instrumento/sensor, las unidades de medida, posibles umbrales y así sucesivamente. Características tan importantes que permiten la medición datos múltiples y heterogéneos sin importar su procedencia o manufactura (A. Morato, 2020). En el ámbito industrial un OPC UA se implementa sobre interfaces maquina humano (HMI) o computadores las cuales usualmente tienen un alto costo.

Por otro lado, siguiendo la vertiente del uso de placas de desarrollo como el Raspberry Pi y Arduino, las cuales son sumamente adecuadas y de gran utilidad al momento de querer implementar las arquitecturas OPC UA y DMS, dado que satisfacen los campos de calidad, productividad, eficiencia, aunado a su asequibilidad. Se ha demostrado que los proyectos de esta índole tienen costos rentables en su desarrollo, además de brindar una solución precisa al caso de estudio para el que fueron creados (Charania, 2021), (Vimos, 2018), (Park, 2019), (Mizuya, 2017), (Peña, 2019), (Endeley, 2019), (Korodi, 2017). Y otro punto a favor y de gran importancia es el que estas plataformas cuentan con la posibilidad de escalabilidad para diversas tecnologías 4.0.

Sin embargo, ninguno de estos desarrollos busca un enfoque didáctico lo que hace imperativo la implementación de herramientas pedagógicas que vayan de la mano con el avance de las tecnologías en la industria 4.0, que permitan al educando las nociones necesarias sobre desarrollos de vanguardia en automatización de procesos y todos los enseres que estos conllevan, con el objetivo principal de que con estos conocimientos tengan una inserción en el mundo laboral más adecuado a lo existente en la industria en la actualidad.

El sistema didáctico propuesto implementa un control PID mediante un DMS que utiliza un servidor OPC UA. El entorno desarrollado cumple con la arquitectura de este tipo de sistemas y pretende servir de guía para que los alumnos adquieran nociones claras de la implementación y bondades de los OPC y los controles PID en el área industrial.

2. Diseño del sistema

En la Figura 1 se puede observar de forma gráfica el sistema didáctico que se propone.

El sistema controla y monitorea las variables pertenecientes a los sensores y actuadores conectados a un Arduino Uno en tiempo real, a través de una comunicación cliente-servidor implementada en un OPC UA y donde de acuerdo a las necesidades se aplican y modifican las leyes del control PID.

El servidor OPC montado sobre la Raspberry Pi cuenta con los componentes necesarios (Wi-Fi y Ethernet) para establecer una comunicación a Internet y enlazarse de manera

remota con un cliente alojado en cualquier tipo de arquitectura es decir una HMI o una computadora con sistema operativo (OS) Windows, Linux o MAC. La librería que se utilizó para correr el servidor y cliente es gratuita y ofrece una experiencia suficiente para los fines buscados. En el sistema desarrollado se emplea una tarjeta Arduino como adquisitora de datos y se usaron componentes de bajo costo y de uso común tanto en la vida estudiantil como la industria. A continuación, se da una breve referencia de las tecnologías utilizadas en el sistema propuesto.

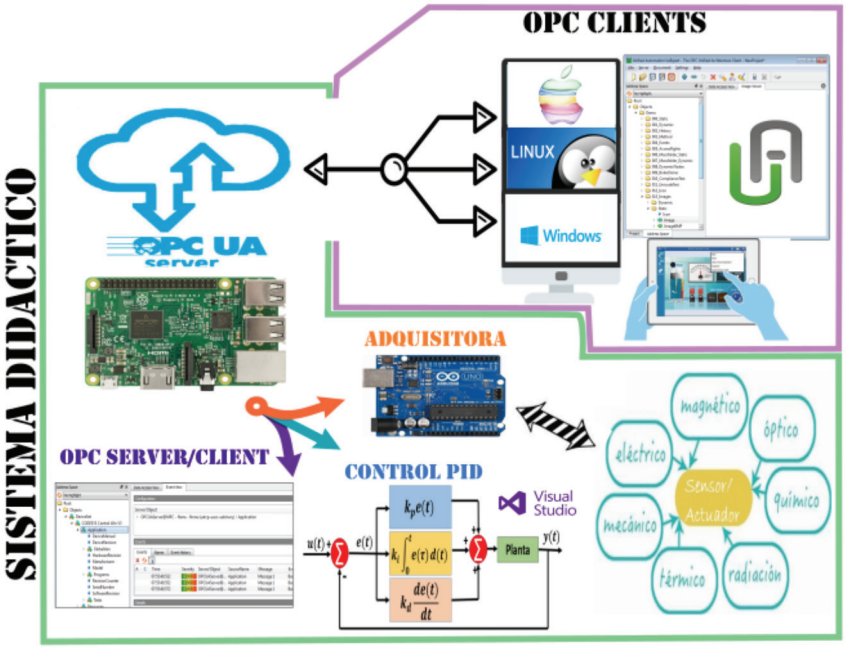

Figura 1 – Esquema de Sistema didáctico propuesto.

2.1. Raspberry Pi

Raspberry Pi es un multiprocesador, cuenta con una tarjeta gráfica, una memoria volátil, RAM, interfaces de dispositivos alámbricos e inalámbricos externos. Una de sus muchas ventajas es que consume poca energía, pero sigue siendo barata y potente. Usualmente la Raspberry Pi usa una tarjeta SD como disco duro, en el cual se le carga su OS basado en Linux llamado Raspbian el cual viene cargado con más de 35,000 paquetes y software precompilado incluido en un formato agradable para una fácil instalación en la placa. La mayoría de sus versiones cuenta con conectividad LAN por Ethernet y/o Wifi de manera nativa, pero es capaz de comunicarse con otros dispositivos externos usando tecnologías de comunicación inalámbrica, redes celulares, NFC, ZigBee, Bluetooth, etc. Python es el lenguaje de programación más utilizado para implementaciones en esta tarjeta, que pueden se diferentes indoles domésticas, comerciales e industriales (Raspberry Pi Foundation, 2021), (Naik, 2019).

2.2. Arduino

Arduino es una plataforma electrónica de código abierto basada en hardware y software amigables, asequibles y fáciles de usar. Utiliza el lenguaje de programación Arduino (basado en Wiring), y el Software Arduino (IDE), basado en Processing. Arduino nació en el Instituto de Diseño de Interacción Ivrea como una herramienta fácil para la creación rápida de prototipos, dirigida a estudiantes sin experiencia en electrónica y programación.

Arduino simplifica el proceso de trabajar con microcontroladores, pero ofrece ventajas adicionales como lo es su precio en el mercado, además es multiplataforma ya que su software se ejecuta en los sistemas operativos Windows, Macintosh OSX y Linux. El entorno de programación de Arduino es fácil de usar para principiantes, pero lo suficientemente flexible para que los usuarios avanzados también lo aprovechen. Además, es de código abierto y extensible lo que hace posible que desarrolladores puedan realizar mejoras a las placas y así aumentar su eficiencia (What is Arduino?, 2018).

2.3. Visual Studio

Microsoft Visual Studio es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador la creación de software, permitiéndonos desarrollar aplicaciones, sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET. Fue creado por la compañía Microsoft y está disponible para sistemas operativos Windows, Linux y macOS, y la vez es compatible con múltiples lenguajes de programación, tales como C++, C#, Visual Basic.NET, F#, Java, Python, Ruby y PHP, al igual que entornos de desarrollo web, como ASP.NET, fue lanzado en 1997, cuenta con versiones gratis y de venta.

Está basado en BASIC (Beginner's All-purpose Symbolic Instruction Code, por sus siglas en ingles), un lenguaje de programación de alto nivel, que puede ser tanto interpretado como compilado, no estructurado y de fácil aprendizaje. Desde Visual Studio 2015 su nuevo IDE, viene preparado para desarrollar aplicaciones para Windows, pero también para Android, iOS y Windows Phone. Esto permite desarrollar en C# nativo sin necesidad de usar Java (Visual Studio, 2019).

2.4. Matlab

MATLAB (abreviatura de MATrix LABoratory, «laboratorio de matrices») es un software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M). Está disponible para las plataformas Unix, Windows, macOS y GNU/Linux. Dispone de dos herramientas adicionales que expanden sus prestaciones: Simulink (plataforma de simulación multidominio) y GUIDE (editor de interfaces de usuario - GUI). Además, se pueden ampliar las capacidades de MATLAB con las cajas de herramientas (toolboxes); y las de Simulink con los paquetes de bloques (blocksets). Es un software muy usado en universidades y centros de investigación y desarrollo (Wang, 2020).

2.5. OPC y su introducción a la Arquitectura Unificada

OPC es un estándar de interoperabilidad para el intercambio seguro y confiable de datos en el espacio de automatización industrial. Es independiente de la plataforma y garantiza el flujo continuo de información entre dispositivos de múltiples proveedores. Es una serie de especificaciones desarrolladas por proveedores de la industria, usuarios finales y desarrolladores de software. Estas especificaciones definen la interfaz entre Clientes y Servidores, así como también Servidores y Servidores, incluido el acceso a datos en tiempo real, monitoreo de alarmas y eventos, acceso a datos históricos y otras aplicaciones. La Fundación OPC es la responsable del desarrollo y mantenimiento de este estándar (What is OPC?, 2017).

Inicialmente, el estándar OPC estaba restringido al OS Windows. Estas especificaciones ahora se conocen como OPC Classic y han disfrutado de una adopción generalizada en múltiples industrias, incluida la manufactura, la automatización de edificios, el petróleo y el gas, energías renovables y los servicios públicos, entre otros. Con la introducción de arquitecturas orientadas a servicios en los sistemas de manufactura, surgieron nuevos desafíos en seguridad y modelado de datos. La Fundación OPC desarrolló las especificaciones de Arquitectura Unificada OPC (OPC UA) para abordar estas necesidades y proporcionar una arquitectura de plataforma abierta con tecnología amplia en funciones, escalable y extensible.

OPC UA se basa en una relación cliente-servidor, en la que la información se estructura siguiendo un modelo de orientación de objetos, donde los objetos se denominan formalmente como “nodos”, que son la entidad fundamental de OPC UA y representan un objeto básico que tiene los atributos necesarios para definir cualquier tipo de elemento de información, por ejemplo, presión, temperatura, etc. Para acceder a la información almacenada dentro de los nodos, OPC UA proporciona una serie de servicios (Montalvo, 2020).

En el contexto de las aplicaciones de medición basadas en IIoT, el protocolo OPC UA se puede explotar de forma rentable para permitir la comunicación segura entre las fuentes heterogéneas de medición de datos. De hecho, las mediciones se pueden almacenar por nodos que pertenecen a uno o más servidores, para que clientes distribuidos puedan acceder sin problemas (S. Schriegel, 2018). Con OPC UA es posible hacer uso de aplicaciones basadas en Windows, Linux y Mac OS.

2.6. FreeOPCUa

FreeOPCUa es una librería de licencia abierta capaz de ejecutarse en distintos lenguajes de programación como Python o C++, también ofrece la disponibilidad de operación en OS como Windows, Linux y MAC. Cabe mencionar que su descarga es totalmente gratuita, y la misma se puede realizar desde diversos repositorios aportados por comunidades como stack overflow y GitHub. Con la instalación de esta librería se puede ejecutar el servidor y cliente para la implementación de un OPC UA.

2.7. Unified Automation y UaExpert

Unified Automation es una empresa que ofrece productos y servicios en el campo de la comunicación estandarizada en la industria de la automatización entre otros. Con

base en la tecnología OPC UA, Unified Automation ofrece un marco de desarrollo de software multiplataforma mediante la conexión con kits de desarrollo de software (SDK) para permitir la integración de información vertical para proveedores de aplicaciones, desde fabricantes de dispositivos integrados hasta desarrolladores de aplicaciones empresariales.

Los productos de Unified Automation permiten a las empresas construir, implementar e integrar fácilmente tecnología de comunicación estándar. Sus productos están diseñados para mejorar la calidad de las aplicaciones y dispositivos del cliente. La automatización unificada permite a los clientes acelerar la innovación, acortar el tiempo de comercialización y aumentar los ingresos. En este ámbito UaExpert es un cliente OPC UA con todas las funciones que demuestra las capacidades de un SDK de cliente OPC UA de C++. UaExpert está diseñado como un cliente de prueba de uso general que admite funciones de OPC UA, es multiplataforma y está programado en C++ (Unified Automation, 2021).

2.8. VNC

VNC es un programa de software libre basado en una estructura cliente-servidor que permite observar las acciones de un dispositivo (servidor) remotamente a través de otro dispositivo (cliente). VNC no impone restricciones, es posible compartir la pantalla de una máquina con cualquier OS que admita VNC conectándose desde otro ordenador o dispositivo que disponga de un cliente VNC. El software de VNC tiene una gran comunidad en todo el mundo, especialmente dentro de los sectores de la industria, el gobierno y la educación.

Desde el año 2016 RealVNC, el desarrollador y proveedor original del software, anuncio la asociación con la Raspberry Pi Foundation para incluir su software VNC en las placas de desarrollo con el Raspberry Pi OS. Con esto, el software RealVNC se convirtió en el estándar oficial para la conectividad de acceso remoto en todas las Raspberry lo cual hace que VNC Viewer se proporcione de forma gratuita para uso educativo o no comercial (VNC Connect, 2021).

2.9. Control PID

Un controlador PID es del tipo continuo y se encarga de modular la señal de control de un sistema en función del error que existe entre el valor medido y el valor deseado. Este tipo de control es ampliamente utilizado en los procesos industriales, incluso cuando se presenten interferencias externas.

Asimismo, resulta fácil de implementar y puede ser utilizado en todo tipo de hardware de manera eficiente debido a que no utiliza muchos recursos. Por otro lado, lo crítico radica en la sintonización, la cual debe de realizarse por un experto en proceso y durante alguna prueba en campo. Una vez sintonizado el controlador, no requiere de ajustes adicionales. La ecuación 1 representa un control PID.

$$c(t) = Kp * e(t) + Ki * \int e(t) + Kd * \frac{\partial e(t)}{\partial t} \quad \text{(Ec.1)}$$

Donde:

- c(t) = señal de control
- e(t) = señal de error
- Kp, Ki, Kd = parámetros del controlador PID

Cabe mencionar que los parámetros del control PID se sintonizan de acuerdo al sistema a controlar (Wang, 1999) y que existen métodos ya establecidos y bien documentados por décadas (Verma, 2020), (Nisi, 2019), (Memon, 2020), (Daful, 2018).

3. Implementación del sistema

El primer paso en el desarrollo del sistema es la configuración del entorno en Raspberry Pi la cual es muy específica, ya que para un buen funcionamiento de la arquitectura OPC UA se deben hacer las configuraciones e instalaciones de paquetes de forma correcta.

Usar la Raspberry Pi con el servidor OPC UA permite tener diferentes configuraciones de clientes. Existen dos opciones bastante practicas dentro de la amplia gama de clientes expertos que hay a disposición:

1. La primera es descargando e instalando un cliente local en la Raspberry Pi y mediante terminal usarlo. Sin embargo, con esta configuración si se desean ver los datos de forma remota es necesario utilizar un programa de escritorio remoto como por ejemplo un VNC Viewer.
2. La segunda opción es con el uso de un programa cliente como lo es el UA Expert que puede ser instalado en una computadora con OS Windows, Linux, o MAC y tener el acceso remoto mediante este sin necesidad de un escritorio remoto.

Ambas opciones de configuración son igual de eficientes ya que las dos cumplen con el objetivo principal de permitir la comunicación entre elementos de distinto fabricante. La implementación de una u otra configuración será determinada por las necesidades del operador o por los recursos con los que cuenten. A continuación, se detalla paso a paso la configuración del sistema propuesto.

3.1. Configuraciones iniciales del entorno

Para el buen funcionamiento del sistema OPC UA, se comienza con la instalación del entorno donde se desarrollan las funciones del servidor, al hacer uso de una tarjeta Raspberry Pi como servidor es necesario la instalación del OS que se aloja en una tarjeta MicroSD. El sitio oficial de Raspberry para la descarga del OS es https://www.raspberrypi.org/software/, en este sistema se utilizó una Raspberry Pi 3B y el OS Raspbian.

3.2. Servidor OPC UA en Raspberry Pi

Con el OS cargado se prosigue a configurar el servidor OPC UA, para ello se utilizó la librería FreeOPCUa, dado que es ideal para ser empleado en Raspberry Pi debido a que no tiene costo alguno, permite una amplia visualización de valores y sus características y que la cantidad de memoria para su almacenamiento en la placa es pequeña. La

instalación de la librería del servidor se inicia con las siguientes instrucciones en la terminal:

```
Pi: - $ sudo apt-get update
Pi: - $ sudo apt install libxml2-dev libxmlsec1-dev libffi-dev
Pi: - $ sudo pip3 install cryptography
Pi: - $ sudo pip3 install freeopcua
```

Para comprobar que el servidor fue creado con éxito, se escribe la instrucción:

```
Pi: - $ python 3
```

Para abrir el entorno de Python y dentro de terminal el IDLE se ingresa:

```
Pi: - $ from opcua import Client
```

Se da Enter y sino no aparece ningún error, se concluye que el servidor fue instalado correctamente. Con esto se monta el servidor en la placa Raspberry Pi y se procede a realizar tanto la configuración para la adquisición de datos como la configuración para la escritura y control de valores.

3.3. Código en Arduino para lectura de datos

Previo al uso del control PID se realizaron pruebas de conexión, para esto se empezó con la adquisición de datos. Los materiales que se utilizan para llevar a cabo el circuito de adquisición son de electrónica básica: un Arduino Uno, un potenciómetro, un Diodo Emisor de Luz (LED), una tablilla de conexión (Protoboard) y cables para la conexión de los componentes.

Para este punto el código que se realizó, regula la intensidad de un LED con un potenciómetro. Las variaciones en la intensidad del LED van de 0-255 y son controladas por el valor del potenciómetro que es recibido por un pin analógico, este valor es transformado a uno digital que cambia la intensidad del LED. El circuito usado se puede apreciar en la figura 2.

El valor digital tiene que ser direccionado a través de una comunicación serial para ser leído y procesado por la placa Raspberry Pi y su código Python de lectura. Una vez realizada la parte de adquisición de datos se procede a realizar la parte de control/escritura.

3.4. Código y esquema en Arduino para escritura de datos

Los elementos usados para la escritura son similares los de lectura: un Arduino Uno, un LED, una Protoboard y cables para la conexión de los componentes. El código en Arduino que se realizó busca encender y apagar un LED estableciendo parámetros de comparación para su ejecución. En este caso se compara el valor leído en la comunicación

serial entre Python y Arduino. Si Arduino lee un “0” apaga el LED, si lee “1” prende el LED. En caso de que se mande un valor diferente a 0 y 1 el programa no realiza ninguna. El circuito utilizado se puede apreciar en la Figura 3.

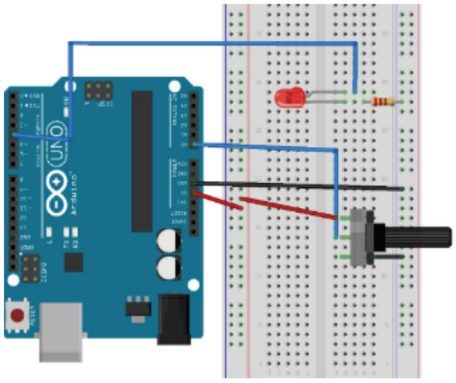

Figura 2 – Diagrama de conexión para envió de datos desde Arduino.

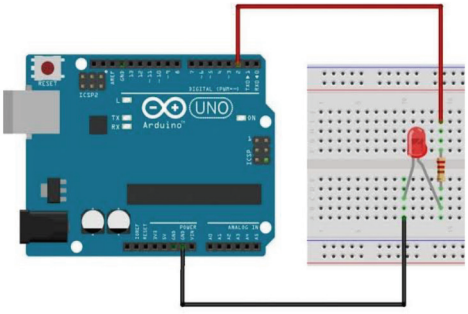

Figura 3 – Diagrama de conexión para recepción de datos en Arduino.

Hasta aquí la parte adquisitora del sistema está listo para enviar y recibir datos por lo que se prosigue con la configuración del servidor para la conexión con esta.

3.5. Código en Python para lectura

Las variables que se crearon para el código de lectura son introducidas en un nodo dedicado a los parámetros, ya que este contendrá toda la información sensada y es el objeto que será leído por los clientes.

El código de Python se encarga de arrancar el servidor y del direccionamiento de datos. Empieza importando el **servidor** de la librería **opcua** y el **serial** de la librería **PySerial**. Esta última es necesaria dado que Arduino envía los datos mediante un puerto USB el cual se toma como un puerto serial a la Raspberry. Con la instrucción **deArduino = serial.Serial(‘PUERTO’,BAUDIOS)** se indica que los datos de la comunicación

serial entre Arduino y Python se guardan en la variable “deArduino”. Con los comandos **server = Server(), url = “opc.tcp://IP:4840”** y **server.set_endpoint(url)** se indica en qué dirección de Internet Protocol (IP) se creara el servidor.

Es imperativo que en el código se cree el nodo que leerán los clientes OPC UA. El mandar la información en forma de nodos permite a los clientes procesar la información de las variables de forma eficiente ya que estos se encuentran compactados en una sola estructura. Finalmente, los datos leídos en la comunicación serial son mandados a una variable que se encarga de su impresión en la consola de Python, esto como un método de verificación de que los datos son recibidos.

3.6. Código en Python para escritura

Para realizar la escritura es necesario la realización de dos códigos uno que correrá en el servidor y otro en el cliente. Para el código que estará en el servidor, de manera similar al código de lectura las variables creadas son introducidas en un nodo dedicado el cual guarda la información del estado del LED y es el objeto que será modificado por el cliente. Para visualizar el estado de operación del LED se imprime la variable en la consola de Python con el objetivo de verificar si el LED se encuentra encendido o apagado.

En el caso del código del cliente, este tiene la finalidad de escribir valores para la manipulación del actuador mandando valores al servidor de “1” ó “0” para posteriormente controlar el encendido y apagado del LED, básicamente establece que el espacio de direcciones se guarde en una variable para poder manipular los valores. Es imprescindible establecer y definir la conexión del cliente con el servidor, además de destacar que la IP corresponde al lugar de trabajo del operador, por lo tanto, para que funcione desde cualquier lugar bastará con introducir la nueva IP del lugar donde se esté operando.

Con esto se tiene todo lo necesario para la lectura y escritura de datos en el servidor y su comunicación vía serial con la tarjeta Arduino.

3.7. Código y esquema del Control PID

Con el sistema configurado adecuadamente para escritura y lectura se procede a la configuración del control PID. Para esto se conecta a Arduino a una interfaz en Visual Studio que funciona como una HMI que permite enviar tanto los Set Points y las ganancias del control PID. En Arduino se procesa la información y se mandan las leyes de control propias del sistema a controlar.

A modo de verificar el correcto funcionamiento del control PID se utilizó un motor de corriente continua (DC) como planta a controlar. Los materiales que se utilizaron son un Arduino Uno, un puente H integrado (L298N), un motor DC con Encoder de 12V, una fuente de energía para alimentar el motor, y cables para la conexión de los componentes.

El código en Arduino que se realizó ejecuta un Control PID que regula las RPM (Revoluciones Por Minuto) a las que gira el motor. En este, primero se declaran todas las variables utilizadas en el control (errores y ganancias), los pines que se utilizan para el manejo del puente H y las fases A y B del Encoder. Después se reciben los datos emitidos por la interfaz, las cuales son: los estados de los botones de Start y Stop, las RPM

deseadas y las ganancias establecidas para la operación del control PID. Posteriormente esas ganancias son guardadas en las variables declaradas como KP, KI y KD. Con los datos guardados se implementan las leyes del control PID, se calcula el error de acuerdo al set point (referencia) establecido y lo sensado en la retroalimentación (mediante un encoder incremental), posteriormente se mandan las RPM al motor de manera iterativa hasta que el error sea mínimo o nulo. El esquema que se utilizó para verificar el PID se puede apreciar en la figura 4.

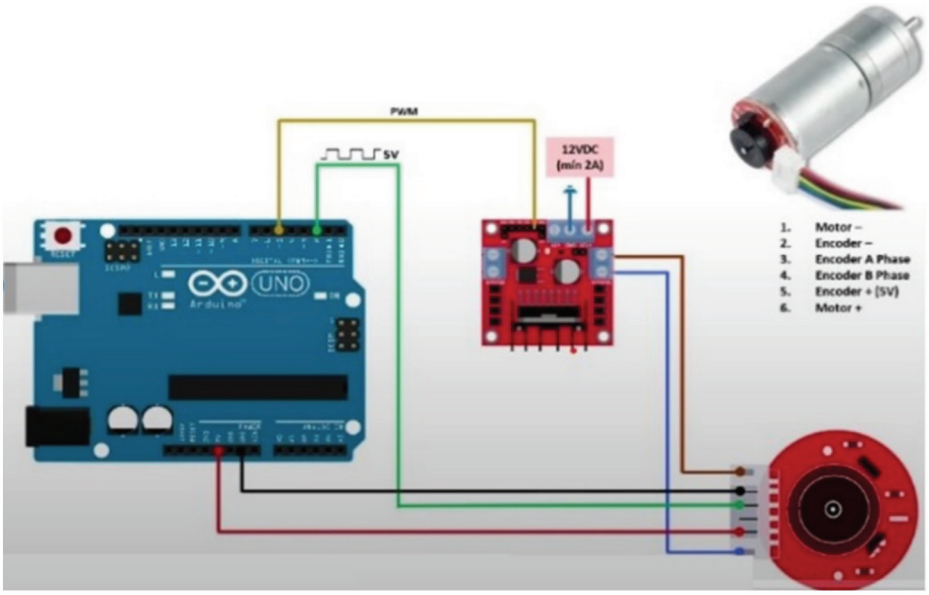

Figura 4 – Esquema de conexión del Control PID.

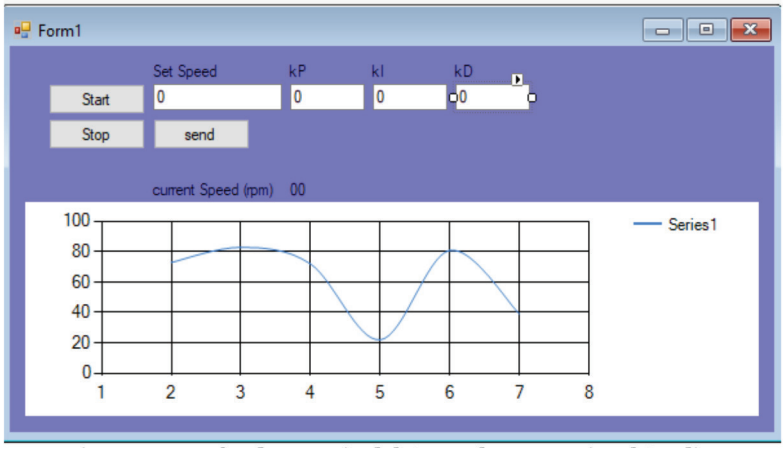

Figura 5 – Interfaz de usuario del Control PID en Visual Studio.

La interfaz de usuario realizada (Figura 5) modifica y actualiza los valores antes mencionados. Lo que se busca con la interfaz es brindarle al usuario una herramienta practica para manipular el control PID a sus necesidades.

Hasta este punto se tiene lo necesario para que la tarjeta Raspberry Pi pueda controlar y monitorear los datos de los sensores y actuadores conectados a la Arduino, pero es necesario que estos datos sean cargados al servidor, por lo que el siguiente paso es crear los códigos en Python que permitan el direccionamiento de datos al servidor FreeOPCUa.

Dentro del código para la lectura de los valores del controlador PID propuesto para el monitoreo de un motor DC, las variables son introducidas en un nodo dedicado a los parámetros, ya que esté contendrá toda la información sensada y es el objeto que será leído por los clientes. El código es idéntico en estructura al código de lectura para saber el rango de operación del LED. Estos códigos permiten el direccionamiento de los valores al cliente OPC UA que deseemos. Es importante indicar que la dirección IP que pongamos en el código tiene que ser la IP del sitio en el que se esté trabajando ya que si se envía a otra dirección será imposible la visualización de los valores.

Con el sistema conectado en su totalidad e implementado el control PID se realizó una sintonización para asegurar su correcto funcionamiento.

3.8.Sintonización del control PID para un motor DC con MATLAB

Para la sincronización del control PID es necesario obtener el modelo matemático del motor utilizado que en este caso en particular es un motor DC de 130 RPM. El motor tiene un eje rígido, por medio de sus ecuaciones eléctricas y mecánicas, al relacionarlas, se obtiene el modelo matemático del motor en el cual la entrada es el voltaje aplicado y la salida es la velocidad rotacional del eje, para esto es necesario conocer los diferentes parámetros de los que se encuentra compuesto:

- Momento de inercia del rotor J = 5.240247 e^{-03}
- Torque del motor (T) = 0.12663
- Coeficiente de amortiguamiento del sistema mecánico (B) = 9304188 e^{-03}
- Constante de fuerza electromotriz K=Ke=Kt. = 0.8442
- Resistencia eléctrica (R) = 3.4 Ω
- Inductancia eléctrica (L) = 1.5 e^{-03} H
- Voltaje de armadura (V_a) = 12 V
- Corriente de armadura (i_a) = 0.15 A
- Velocidad rotacional del eje (ω) = 13.6135 $\frac{rad}{S}$

Con los parámetros se hace uso de la fórmula de la función de transferencia de la velocidad del rotor respecto al voltaje aplicado (ecuación 2):

$$\frac{W}{V} = \frac{K}{LJS^2 + (RJ + LB)S + RB + K^2} \qquad \text{(Ec.2)}$$

Al trasladar los parámetros del motor DC de 130 RPM a la fórmula de función de transferencia se obtiene (ecuación 3):

$$\frac{W}{V} = \frac{0.8442}{7.8603705\ e^{-06} + 17.830795\ e^{-03}\ S\ +\ 744.307879\ e^{-03}} \qquad \text{(Ec.3)}$$

Con la función de transferencia del motor se procede a la obtención de las ganancias del control PID del sistema mediante el software de MATLAB. Se creo un proyecto en Simulink en donde se agregaron los bloques de escalón unitario, el del control PID, el de la función de transferencia y el scope para la visualización de los valores. Como se puede apreciar en la siguiente Figura 6 se tienen ya puestos los valores obtenidos con los parámetros mecánicos y eléctricos del motor DC.

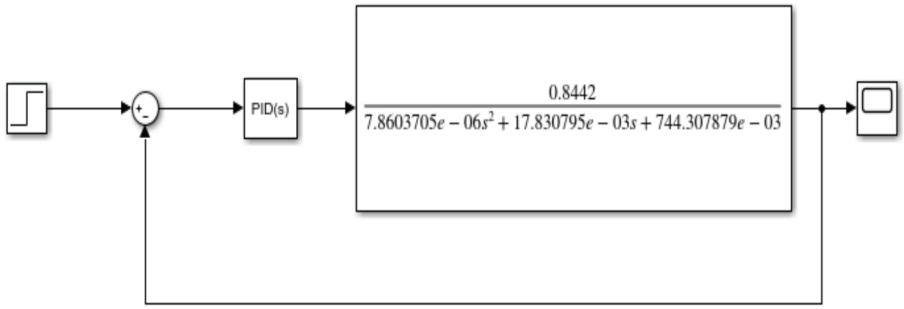

Figura 6 – Diagrama matemático en Simulink.

De acuerdo a lo obtenido en la sintonización del sistema PID los valores indicados para cada una de las variables del control son:

- P = 1.2243
- I = 100.6409
- D = -0.0005251

Estos valores son introducidos en el programa de Visual Studio para la puesta en marcha del motor. En este punto se tiene todo lo necesario para la lectura y escritura de datos en el servidor y su comunicación vía serial con la tarjeta Arduino.

3.9. Cliente OPC UA en Raspberry Pi

La primera opción de cliente se realiza con un cliente OPC UA local en la Raspberry y la configuración para un acceso remoto mediante el uso del programa VNC Viewer. Primero se abre la interfaz de FreeOPCUa, después para permitir el acceso remoto se activa el servicio de conexión VNC en la Raspberry Pi accediendo a las preferencias de la Raspberry Pi, después en la pestaña interfaces se activa la casilla de VNC. Con esto el VNC Server se activa y está disponible siempre que se enciende la placa. Y para determinar la IP a la cual se conectará de manera remota, se accede al icono de VNC que se encentra en la esquina superior derecha en la Raspberry lo que abre una caja de dialogo que muestra la IP de conectividad la cual se debe de introducir en el VNC Viewer del cliente para poder establecer una conexión correcta a un acceso remoto.

3.10. UA Expert como cliente OPC UA.

La segunda opción de clientes es contar con un programa que funja como cliente dedicado a la lectura y representación de los valores enviados por el servidor OPC UA. Dentro de la gran gama de programas dedicados a esto una de las opciones más practicas es el software UA Expert, dado que este se puede instalar en diferentes dispositivos sin importar su OS, además de que es gratuito. Para instalarlo se accede al sitio oficial de Unified Automation, al sitio web: https://www.unified-automation.com/. La ventaja de utilizar este cliente es que se puede acceder a él desde cualquier parte con tan solo tener disponible una conexión a internet. UA Expert es un cliente practico que presenta una interfaz para el usuario amigable donde se puede visualizar claramente los valores que se estén monitoreando.

Una vez terminadas todas las configuraciones necesarias tanto para el servidor, la tarjeta adquisidora, como para ambas opciones de clientes se procede al análisis de estas.

4. Análisis y resultados

Primero se evaluó la lectura de datos mediante las dos opciones de clientes mencionadas, posteriormente la escritura y por último el funcionamiento del controlador PID.

4.1. Conexión a FreeOPCUa Client y visualización de valores

En la primera opción de servidor-cliente se accedió al cliente local, después en VNC viewer se introdujo la dirección IP del VNC server previamente obtenida. Una vez establecida la conexión entre servidor y cliente, se visualizó el nodo de los “Parámetros” que contine la variable “Potenciómetro”. Después se movió la perilla del potenciómetro a un punto al azar y se verifico el valor leído por el FreeOPCUa Client (Figura 7). Con esto se comprueba el correcto funcionamiento de esta opción.

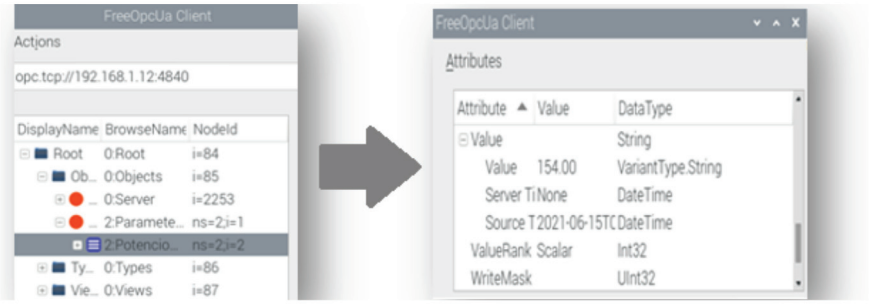

Figura 7 – Lectura de los valores del Nodo (154, que es una intensidad intermedia de LED).

4.2. Conexión a UA Expert y visualizacion de valores

Para esta opción se debe tener que tener corriendo el programa Server.py en el servidor para poder ingresar desde el cliente de manera remota con la plataforma de UA Expert.

Desde el servidor se puede verificar en consola que se envían los datos. Una vez que está operando el servidor y se mandan los valores de la intensidad del LED entre el rango definido de 0 a 255, se procede a correr el cliente UA Expert que nos permite ver esos valores en tiempo real remotamente. Al momento de entrar a la pantalla principal del UA Expert se debe de asociar al cliente con el servidor mediante una dirección IP. Cuando se detecta el servidor, se da click derecho y se escoge la opción de conectar. Si se conecta de forma correcta al servidor se puede visualizar en el árbol de Servers (Figura 8).

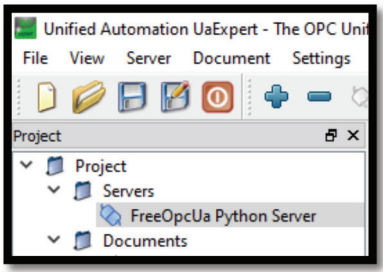

Figura 8 – Visualización de conexión correcta con el Servidor.

En la Figura 9 se puede apreciar que el nodo “Parámetros” contiene los valores de intensidad del LED y permite la visualización de forma continua hasta que se le indique lo contrario con una desconexión. Esto determina que toda la configuración hecha con anterioridad funciona según lo previsto.

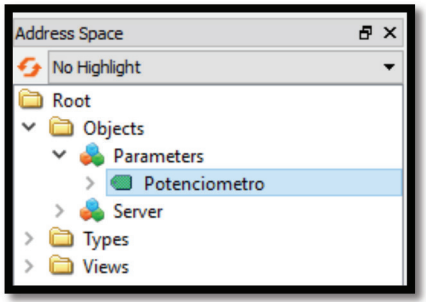

Figura 9 – Visualización de Parámetros en el cliente UA Expert.

Para probar la funcionalidad del servidor OPC UA en la Raspberry Pi y del cliente en UA Expert se realizaron pruebas moviendo el potenciómetro del circuito para lectura de datos. Donde se movió la perilla del potenciómetro y el valor mostrado vario (Figura 10).

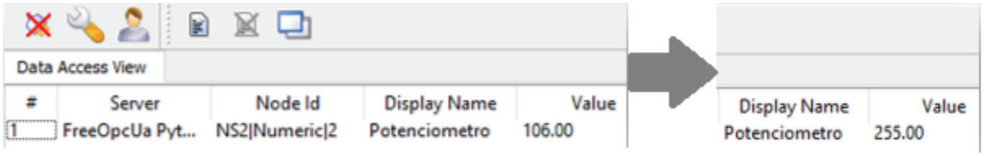

Figura 10 – Visualizacion de cambio de valor en UA Expert.

La visualización de los valores enviados de Arduino al servidor en Raspberry y accedidos por el cliente UA Expert se aprecian de forma clara y práctica, ya que permiten al operador del cliente OPC UA ver todas las características de la variable que se lee. Es decir, se observa la dirección del nodo en la que esta empaquetado el valor, el nombre de la variable en medición, su valor en tiempo real, entre otras tantas opciones más.

Después de comprobar ambas opciones de clientes con la correcta visualización de dato, se procede a la verificación de escritura para esto se tomó en cuenta la opción de cliente local.

4.3. Conexión Cliente-Servidor FreeOPCUa para escritura y visualizacion de valores

Para verificar el funcionamiento en la escritura de valores para el encendido y apagado del LED se mandaron dos valores. El primer valor es “0”, lo cual apaga el LED. Despues el valor “1” que encience el LED, estos valores se tambien se pueden monitorear en la consola del servidor (Figura 11).

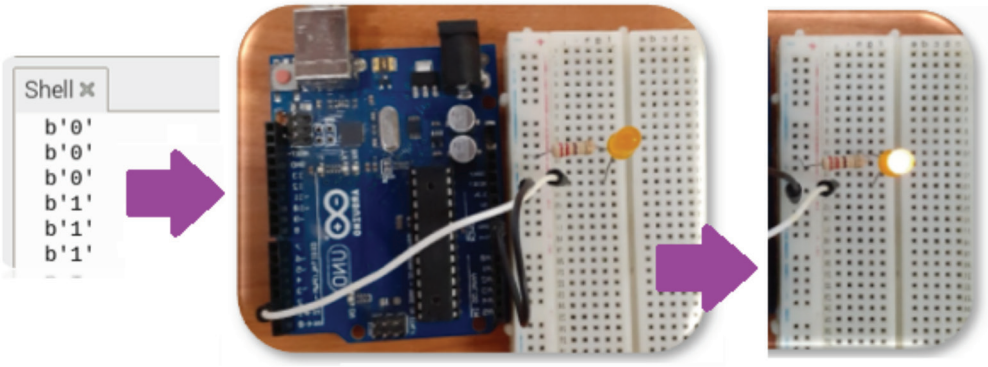

Figura 11 – Valores recibidos en el servidor y leídos en Arduino.

Con esto se termina las pruebas de verificacion del sistema OPC UA en sus dos modalidades (lectura/escritura), comprobando asi su facil implementacion y correcto funcionamiento y solo falta determinar el comportamiento del conrolador PID.

4.4.Visualización de valores del sistema PID

Para esta opción previamente se tiene que tener corriendo el programa en Python encargado de la transferencia de los valores al servidor para poder ingresar desde el cliente de manera remota con la plataforma de UA Expert. En la pantalla principal del UA Expert se debe de asociar al cliente con el servidor del PID mediante su dirección IP.

Para comprobar el correcto funcionamiento del servidor OPC UA y del cliente en UA Expert se realizaron pruebas modificando las RPM del motor. Primero se estableció un SetPoint de 50 en Visual Studio, enviando ese valor a Arduino y este lo trabaja junto a los valores de las ganancias PID previamente establecidas y finalmente mediante una

comunicación serial entre Arduino y Raspberry Pi se mandan los valores al cliente UA Expert como se puede observar en la Figura 12.

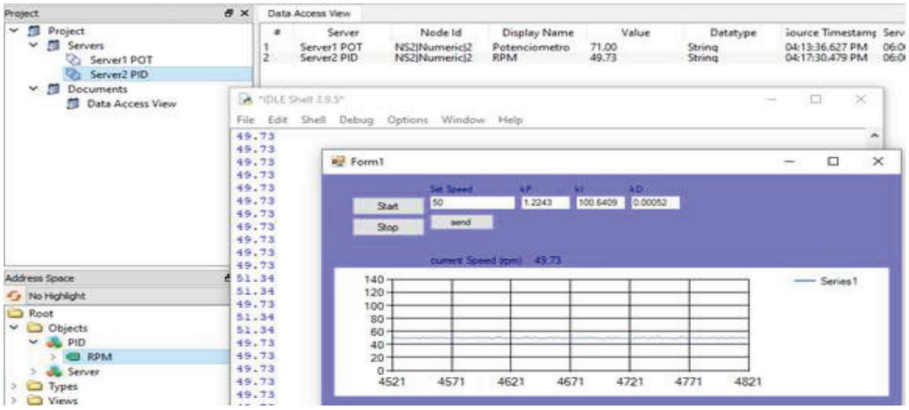

Figura 12 – Visualización del sistema PID en UA Expert, Python y Visual Studio.

La visualización de los valores enviados de Arduino al servidor en Python y accedidos por el cliente UA Expert se aprecian de forma clara y práctica, ya que permiten al operador del cliente OPC UA ver todas las características de la variable que se lee. Con esto se termina las pruebas de verificación del sistema didáctico propuesto.

5. Conclusiones

A lo largo de la historia de la automatización los avances usualmente son extensiones de las tecnologías existentes y esto se puede ver en todos los niveles de la industria. En la actualidad con el surgimiento de la 4ta revolución industrial la tendencia de los ejes que la componen va de la mano de esta idea. Y es de esperarse por los muchos beneficios que ofrece la industria 4.0: ahorro de costes y de tiempo, mejora de la eficiencia y del consumo energético, menos incidencias y mayor capacidad para concentrar los recursos en nuevas actividades que aporten valor añadido, entre otros. Esto hace de vital importancia contar con herramientas didácticas que les permitan a los jóvenes tener una noción pertinente de las aplicaciones y las tecnologías que encontraran en su inserción en el mundo laboral.

En primera instancia al implementar la comunicación OPC UA representa que los alumnos serán capaces de comunicar diferentes ecosistemas de trabajo, dando la oportunidad a que no sea estrictamente necesario el uso de software y equipo de un proveedor en específico, dando pie a una arquitectura abierta, punto clave en la industria 4.0.

Por otro lado, la implementación de un PID remarcando que este tipo de control es ampliamente utilizado en los procesos industriales, incluso cuando se presenten

interferencias externas, además de que es fácil de implementar y puede ser utilizado en todo tipo de hardware de manera eficiente debido a que no utiliza muchos recursos, ofrece a los estudiantes un panorama muy amplio en cuestiones de automatización.

Es importante destacar que el conocimiento que los alumnos adquieren es valioso, ya que comienzan a relacionarse con las distintas placas de desarrollo que existen y por ende con sus entornos de desarrollo integrados (IDE) (Quiles, 2019), además adquieren conocimientos básicos en software para la implementación de servidores, controladores PID y su sintonización. El uso de sistemas OPC UA a diferencia de otros protocolos de envió, procesamiento y recepción de datos es que OPC UA está diseñada para entornos industriales, ya que busca unificar la forma de operar con distintos fabricantes de dispositivos industriales siendo una solución completa que abarca desde la comunicación de los dispositivos hasta su interoperabilidad.

De esta manera el sistema propuesto tiene fines pertinentes para lo que fue creado, además de ser asequible y replicable. El código QR de la Figura 13 otorga el acceso para descargar los códigos de Arduino y Python, así como diagramas de conexión del sistema propuesto, con la finalidad de que cualquier persona pueda implementarlo y familiarizarse con esta tecnología.

Figura 13 – Código QR para descargar los códigos.

Referencias

Daful, A. G. (2018). Comparative study of PID tuning methods for processes with large & small delay times. 2018 Advances in Science and Engineering Technology International Conferences (ASET), pp. 1-7, https://doi.org/10.1109/ICASET.2018.8376915.

Morato, A., Vitturi, S., Tramarin, F. & Cenedese, A., "Assessment of Different OPC UA Industrial IoT solutions for Distributed Measurement Applications," 2020 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), 2020, pp. 1-6

Åström, K. J., & Hägglund, T. (2001). The future of PID control. Control Engineering Practice, 9(11), 1163-1175. https://doi.org/10.1016/S0967-0661(01)00062-4

Baño, H. F. (2019). Sistema empotrado de bajo costo para adquisición de datos basado en el estándar de comunicación industrial OPC-UA (Bachelor's thesis, Universidad Técnica de Ambato. Facultad de Ingeniería en Sistemas, Electrónica e Industrial. Carrera Ingeniería Industrial en Procesos de Automatización).

Borase, R.P., Maghade, D.K., Sondkar, S.Y. (2021). A review of PID control, tuning methods and applications. Int. J. Dynam. Control 9, 818–827. https://doi.org/10.1007/s40435-020-00665-4

Verm, B. & Padhy, P. K. (2020). Robust Fine Tuning of Optimal PID Controller With Guaranteed Robustness. In IEEE Transactions on Industrial Electronics, 67(6), 4911-4920. https://doi.org/10.1109/TIE.2019.2924603.

Díaz-Rodríguez, I. D., Sangjin, H., Bhattacharyya Shankar, P .(2019) Analytical design of PID controllers. Springer.

Bruckner, D., Stanica, M.-P., Blair, R., Schriegel, S., Kehrer, S., Seewald, M., Sauter, T. (2019). An Introduction to OPC UA TSN for Industrial Communication Systems Proc. IEEE, 107(6), 1121–1131.

Endeley, R., Fleming, T., Jin, N., Fehringer, G., & Cammish, S. (2019). A Smart Gateway Enabling OPC UA and DDS Interoperability. 2019 IEEE SmartWorld, Ubiquitous Intelligence & Computin g, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation

Sisinni, E., Saifullah, A., Han, S., Jennehag, U., Gidlund, M. (2018). Industrial internet of things: Challenges, opportunities, and directions. IEEE Trans. Industr. Inform., 14(11), 4724–4734.

Korodi, A., & Silea, I. (2017). Achieving interoperability using low-cost middleware OPC UA wrapping structure. Case study in the water industry. 2017 IEEE 15th International Conference on Industrial Informatics (INDIN).

Mathias, S. G., Schmied, S., Grossmann, D. (2020). An investigation on database connections in OPC UA applications. Procedia Computer Science, 170, 602-609.

Mizuya, T., Okuda, M., & Nagao, T. (2017). A case study of data acquisition from field devices using OPC UA and MQTT. 2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE).

Memon, F., Shao, C. (2020). An Optimal Approach to Online Tuning Method for PID Type Iterative Learning Control. Int. J. Control Autom. Syst. 18, 1926–1935. https://doi.org/10.1007/s12555-018-0840-0

Montalvo, W. (2020). Implementación de OPC UA en una Plataforma Web para la integración de comunicación en el área de producción. RISTI - Revista Ibérica de Sistemas e Tecnologias de Informação, (E26), 667-680.

Naik, S., & Sudarshan, E. (2019). Smart healthcare monitoring system using raspberry Pi on IoT platform. ARPN Journal of Engineering and Applied Sciences, 14(4), 872-876.

Nisi, K., Nagaraj, B. & Agalya, A. (2019). Tuning of a PID controller using evolutionary multi objective optimization methodologies and application to the pulp and paper industry. Int. J. Mach. Learn. & Cyber. 10, 2015–2025. https://doi.org/10.1007/s13042-018-0831-8

Park, H. M., & Wook, J. (2019). OPC UA based Universal Edge Gateway for Legacy Equipment. 2019 IEEE 17th International Conference on Industrial Informatics (INDIN).

Peña, R. R., Fernandez, R. D., Lorenc, M., & Cadiboni, A. (2019). Gateway OPC UA/Modbus applied to an energy recovery system identification. 2019 XVIII Workshop on Information Processing and Control (RPIC).

Qing-Guo, W., Tong-Heng, L., Ho-Wang, F., Qiang, B., Yu, Z. (1999). PID tuning for improved performance. IEEE Transactions on Control Systems Technology, 7(4), 457-465. https://doi.org/10.1109/87.772161.

Quiles Alemañ, J. (2019). Diseño e implementación de un entorno de desarrollo integrado para programación visual de aplicaciones en la generación del Internet

Raspberry Pi Foundation (2021). About Us. Raspberry Pi. https://www.raspberrypi.org/about/

Schriegel, S., Kobzan, T., Jasperneite, J. (2018). Investigation on a distributed sdn control plane architecture for heterogeneous time sensitive networks. 14th IEEE International Workshop on Factory Communication Systems (WFCS), pp. 1–10. IEEE,.

Vitturi, S., Zunino, C., Sauter, T. (2019). Industrial Communication Systems and Their Future Challenges: Next-Generation Ethernet, IIoT, and 5G. Proc. IEEE, 107(6), 944–961.

Unified Automation. (2021). https://www.unified-automation.com/index.html4

Visual Studio (2019). https://visualstudio. microsoft. com.

Vimos, V., Sacoto Cabrera, E. J. (2018). Results of the implementation of a sensor network based on Arduino devices and multiplatform applications using the standard OPC UA. IEEE Latin America Transactions, 16(9), 2496-2502. https://doi.org/10.1109/TLA.2018.8789574.

VNC Connect. (2021). RealVNC. https://www.realvnc.com/es/connect/benefits/

Wang, L. (2020). PID Control System Design and Automatic Tuning using MATLAB/Simulink. https://books.google.com.mx/books?id=1NXKDwAAQBAJ

Arduino (2018). What is Arduino?. https://www.arduino.cc/en/Guide/Introduction/

OPC Foundation. (2017). What is OPC? https://opcfoundation.org/about/what-is-opc/

Lu,Y. (2017). Industry 4.0: A survey on technologies, applications and open research issues. J. Ind. Inf. Integr. 6, 1–10.

Charania, Z. (2021). Opportunities For A Hardware-Based OPC UA Server Implementation In Industry 4.0. IECON 2021 – 47th Annual Conference of the IEEE Industrial Electronics Society, pp. 1-6. https://doi.org/10.1109/IECON48115.2021.9589043