Artículo Científico / Scientific Paper

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR
INGENIUS
Revista de Ciencia y Tecnología
https://doi.org/10.17163/ings.n18.2017.05
pISSN: 1390-650X / eISSN: 1390-860X

Implementación de sistemas distribuidos de bajo costo bajo norma IEC-61499, en la estación de clasificación y manipulación del MPS 500

Implementation of Low Cost Distributed System Under IEC-61499 Standard, in Classification and Handling Station of MPS-500

Gustavo Caiza[1,2,*], Marcelo García[3]

Resumen

El desarrollo de las tecnologías, los requerimientos de nuevas prestaciones y las limitantes dadas por los fabricantes de *software* y *hardware* para la automatización ha hecho que nuevos estándares sean desarrollados. El presente trabajo tiene como propósito la implantación de un sistema distribuido de bajo costo, usando la norma IEC-61499 en la estación de clasificación y manipulación del MPS-500. En estos módulos se pueden dar problemas de aplicaciones industriales reales. IEC-61499 estandariza un entorno de programación para sistemas distribuidos y tiene un alto nivel de versatilidad para el diseño de sistemas, pues combina *software* y *hardware* de manera independiente. Se utilizó el *software* 4DIAC y se crearon nuevos bloques de función (FB) para el control del proceso y el *runtime* FORTE fue ejecutado en una tarjeta Raspberry Pi. La norma IEC-61499 permite al diseñador del sistema crear un modelo de ejecución de bloques de función conducido por eventos, de esta manera, puede dar prioridad a la orden de ejecución y modificar fácilmente algún parámetro de la planta.

Palabras clave: automatización, estándar IEC-61499, Runtime FORTE, sistema distribuido.

Abstract

The development of new technologies, requirements for new benefits and the limitations given by manufacturers of automation software and hardware have promoted the development of new standards. The aim of this paper is the implementation of low-cost distributed systems under IEC-61499 standard, and will be implemented at the classification and handling station of the FESTO MPS-500, in these modules it is possible to use practical problems of real industrial applications. The IEC-61499 standard standardizes the environment of programming for distributed systems. It has a high versatility for systems' design because it allows the use of independent software from the hardware that is being used. In implementation the process, we used the software 4DIAC, where new function blocks (FBs) for process control were created. The FORTE runtime was executed on a raspberry Pi card. The IEC-61499 standard allows the system designer to create an event-driven function-blocks execution model, so that the designer can prioritize the execution order and easily modify some parameters of the plant.

Keywords: Automation, Standard IEC-61499, Runtime FORTE, Distributed System.

[1] Instituto de Posgrado y Educación Continua, Escuela Superior Politécnica de Chimborazo – Ecuador.
[2,*] Carrera de Ingeniería Electrónica, Grupo de Investigación en Electrónica y Telemática (GIETEC), Universidad Politécnica Salesiana – Ecuador. Autor para correspondencia ✉: gcaiza@ups.edu.ec
[3] Carrera de Ingeniería Industrial en Procesos de Automatización, Universidad Técnica de Ambato – Ecuador

Recibido: 07-04-2017, aprobado tras revisión: 29-05-2017

Forma sugerida de citación: Caiza, G.; García, M. (2017). «Implementación de sistemas distribuidos de bajo costo bajo norma IEC-61499, en la estación de clasificación y manipulación del MPS 500». Ingenius. N.°18, (julio-diciembre). pp. 40-46. ISSN: 1390-650X.

1. Introducción

Durante los últimos años, los procesos industriales de automatización a nivel mundial están cambiando, debido a la demanda que requiere mayores prestaciones en los procesos. La mayor parte de procesos industriales utilizan los controladores lógicos programables (PLC), y en estos sistemas las aplicaciones ya instaladas son difíciles de integrar y expandir [1].

La IEC-61131 ha sido la principal norma en el ámbito de la automatización industrial, pues estandariza los lenguajes de programación en este campo [2]. El estándar IEC-61131 no establece los medios para crear nuevos recursos de configuración, además, la semántica está definida de manera ambigua en el apartado IEC 61131-3 por lo que al realizar la implementación las herramientas de *software* interpretan el estándar de diferente manera [3]. Posteriormente el desarrollo de la tecnología ha hecho que se realicen o complementen nuevos estándares, como fue el IEC-61499 el mismo que normaliza un entorno de programación para los sistemas de control distribuido [4].

La IEC-61499 tiene un alto nivel para el diseño de sistemas, y permite combinar componentes de *software* independientemente del *hardware* utilizado. La unidad central de este estándar es el bloque de función (FB) que puede organizarse de manera jerárquica y reutilizarse cuando sea necesario [5]. El FB permite encapsular algoritmos de control o comunicación y estos pueden ser escritos en varios lenguajes de programación: Java, C++, Phyton, etc. [2]. El bloque de función está basado en la norma IEC61131-3 [5].

Durante las últimas décadas el concepto de bloque de función ha estado más presente en la programación de sistemas de automatización y control industrial, el cual encapsula datos y código con sus respectivas entradas y salidas [6].

La norma IEC-61499 separa el flujo de datos y el flujo de eventos dentro de una aplicación, es decir, el FB tiene un flujo de datos de entrada y salida que son activados por eventos, brindando un diseño más óptimo para la utilización de recursos y control de una red de equipos [7].

Con el desarrollo de nuevos estándares se pretende que la industria pueda tener otras opciones para automatizar un proceso y así crear sistemas más flexibles, ya que el *software* no estará limitado o bloqueado por el proveedor con la consiguiente reducción, en gran medida, de los costos de puesta en marcha [8].

Con el presente trabajo se va a implantar un sistema distribuido de bajo costo bajo norma IEC-61499 utilizando la tarjeta Raspberry Pi 3B y será usado en la estación de clasificación y manipulación del MPS 500 de la Universidad Politécnica Salesiana. Se realiza la implementación del estándar IEC-61499 para promover el desarrollo de sistemas heterogéneos, que no estén limitados por el fabricante tanto en *software* como en *hardware* y, además, probar los alcances que se puede tener al utilizar el estándar.

El proceso consiste en colocar una línea de clasificación final y se utiliza las estaciones del MPS-500 debido a que se puede aplicar procesos prácticos y reales de automatización industrial. Utilizando el *software* 4DIAC se realiza el diseño y programación de nuevos bloques de función que serán ejecutados en el *runtime* FORTE. Estos bloques de función son específicos para automatizar el control de sensores y actuadores de las estaciones. De esta manera, se busca tener nuevas opciones para el desarrollo de aplicaciones de control distribuido, reducir el costo de implementación y contar con sistemas flexibles y reconfigurables.

El artículo está organizado de la siguiente manera: en la sección II se describe los conceptos básicos de la norma IEC-61499 y la metodología utilizada; en la sección III se muestra el diseño del sistema de control distribuido y, finalmente, en la sección IV se presentan las conclusiones.

2. Materiales y métodos

2.1. Estándar IEC-61499

Esta norma define una metodología para el desarrollo de sistemas de control distribuidos, la arquitectura está organizada de manera jerárquica, destinada a la portabilidad, interoperabilidad, reutilización y reconfiguración de aplicaciones distribuidas, además, proporciona un modelo genérico que incluye procesos y redes de comunicación como un medio para dispositivos embebidos, recursos y aplicaciones [8].

Se puede tener varios dispositivos compatibles con la norma IEC-61499, los cuales puedan interactuar entre sí y no dependen del fabricante [7].

2.1.1. Modelos de la norma IEC-61499

El estándar IEC 61499 define una arquitectura y unos modelos de *software* para el desarrollo de aplicaciones reconfigurables de control distribuido que se muestran a continuación.

Modelo de recurso: Representa un entorno para la ejecución independiente de las aplicaciones. Los recursos tienen acceso a las interfaces de comunicación y proceso del dispositivo [9]. En la Figura 1 se muestran los procesos que cumple un modelo de recurso.

Modelo de dispositivo: Es la parte física en la cual se va a ejecutar el programa, este dispositivo presenta sus propias características (interfaces de comunicación, periféricos de entrada y salida, etc.) de acuerdo con el fabricante. Estos dispositivos pueden ser conectados a más de un segmento ya que posee dos interfaces: la de proceso y la de comunicación [3].

Modelo de sistema: El sistema es el elemento de mayor nivel el cual engloba todos los dispositivos y apli-

caciones y su respectiva relación entre estos mediante alguna red de comunicación [9].

Modelo de aplicación: Es un conjunto de bloques de función interconectados para una aplicación. Las aplicaciones se organizan mediante redes de FB que a su vez son distribuidas en diferentes recursos [9].

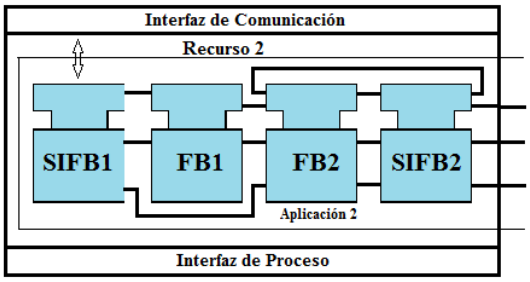

Figura 1. Modelo de recurso IEC-61499 [9].

2.1.2. Bloque de función

El bloque de función proporciona una interfaz gráfica para eventos de entrada y salida como lo muestra la Figura 2, tiene funciones internas que procesan las entradas y de acuerdo con el código interno genera las salidas.

Los bloques de función incluyen gráficos de control de ejecución basados en eventos ECC (Execution Control Chart), que son máquinas de estado. Estos elementos son estados, transiciones y acciones activadas por eventos que controlan la ejecución de los algoritmos que se encuentran asociados al FB. Una ECC puede desencadenar la ejecución de algoritmos dependiendo del evento dado, estas características permiten la fabricación de sistemas flexibles y reconfigurables en la automatización [1].

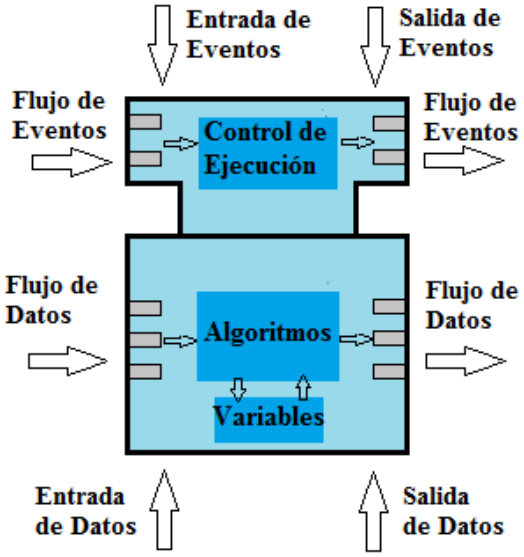

Figura 2. Bloque de función [2].

Los bloques de función pueden ser de tres tipos y se detallan a continuación:

Bloque de función básico: Contiene una máquina de estado que controla la ejecución interna llamada ECC.

Bloque de función compuesto: Puede contener otros bloques de función, por lo tanto, los bloques de función compuesta permiten metodologías de diseño modular [10].

Bloque de función de interfaz de servicio: Es la interfaz entre las aplicaciones y el hardware de los sistemas embebidos, es decir, vincula los recursos para los subsistemas de procesos y comunicaciones [11].

2.2. *Software*

2.2.1. 4DIAC-IDE

Es una herramienta de *software* que se distribuye como un conjunto de complementos para el entorno de desarrollo integrado (IDE) de eclipse. Se basa en el estándar IEC 61499-Bloques de función, el mismo que es un modelo de referencia para la automatización y control distribuido bajo licencia publica Eclipse. El objetivo de la iniciativa 4DIAC (*Framework for Distributed Industrial Automation and Control*) es proporcionar un estándar abierto y gratuito para la automatización y también fomentar la investigación [12]. Proporciona un entorno de ingeniería extensible sobre la base de la norma IEC-61499 para aplicaciones de control distribuido, las mismas pueden ser descargadas en dispositivos de campo que soporten esta norma y, además, permite una fácil integración de otros plugins [13].

2.2.2. FORTE

El entorno de ejecución de 4DIAC-RTE, FORTE es una pequeña implementación portable de un entorno de ejecución IEC-61499. Está enfocado a pequeños dispositivos (16/32 bits) instrumentado en C++. FORTE proporciona una infraestructura de comunicación flexible a través de las llamadas capas de comunicación. Además, proporciona una arquitectura escalable que permite adaptarse al diseño de la aplicación [14].

2.3. *Hardware*

2.3.1. Raspberry Pi

Es un pequeño ordenador de placa reducida fabricado en Reino Unido. La primera versión fue lanzada al mercado en el año 2012 con el propósito de inspirar la enseñanza de ciencias de computación [15].

La tarjeta tiene pines GPIO (*General Purpose Input/Output*) de entrada y salida de propósito general y mediante estos se tiene una interfaz con el mundo exterior. Estos pines son una interfaz física. Tiene 40 pines de los cuales 26 son GPIO y los otros son de

alimentación de 3,3 V, 5 V y tierra, además, dos pines de ID EEPROM [16].

En el presente proyecto se va a utilizar la tarjeta Raspberry Pi 3B debido a sus altas y mejoradas prestaciones respecto a los modelos anteriores.

2.3.2. MPS 500

El MPS-500 (*Modular Production System*) es una estación de producción flexible-compatible, modular y versátil de marca FESTO. Forma la base para la formación técnica en general, utilizando problemas prácticos de aplicaciones reales. Permite la interacción de la mecánica, neumática, ingeniería eléctrica, tecnología de control y las interfaces de comunicación [17].

El módulo consta de seis estaciones como se observa en la Figura 3, cada estación es controlada por un PLC S7-300. En el proyecto se retira los dos PLC de las estaciones y se realiza el control de una aplicación real mediante la tarjeta Raspberry Pi bajo el estándar IEC-61499, y así bajar los costos de implementación de un sistema de control distribuido y ver los alcances que ofrece el estándar al ser instalado en un proceso de automatización real.

Figura 3. MPS-500 [17].

Estación de manipulación. La estación es de control electroneumático y está equipada con un manipulador flexible de dos ejes («x», «y»). Este manipulador cuenta con sensores: óptico, de proximidad magnético, reflectivo y actuadores (pinzas paralelas, cilindro plano, actuador lineal horizontal).

Estación de clasificación. La estación tiene tres rampas para clasificar las piezas por colores (negra, roja, plateada) y una rampa para desechar. Están presentes los sensores de retrorreflexión, de reflexión directa, inductivo y de presencia; los actuadores son el cilindro de carrera corta, el cilindro compacto, el motor dc y los desviadores.

3. Diseño del sistema de control distribuido

En esta sección se muestra el diseño de la aplicación de control mediante el estándar IEC-61499, para lo cual se utiliza la estación de manipulación y clasificación. El proceso consiste en una línea de clasificación final, para ello la primera estación provee piezas al azar de color rojo, negro o plateado y las transporta hacia la estación de clasificación. Esta recibe las piezas y activa el sensor de presencia, el cual inicia el proceso y posteriormente los otros sensores detectan el color; cuando el color de la pieza es detectado activa los actuadores y transporta hacia las rampas seleccionas. En la Figura 4 se muestra de manera general el diseño del sistema de control distribuido.

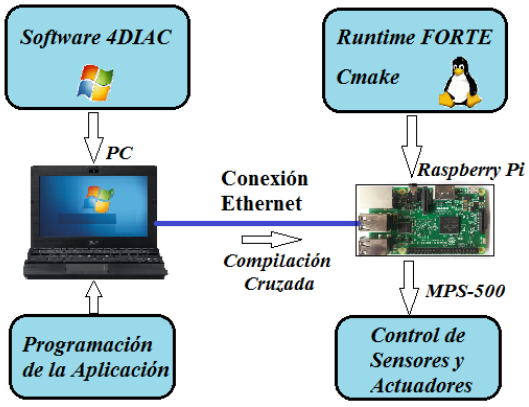

Figura 4. Diseño del sistema de control distribuido.

La aplicación fue desarrollada bajo las siguientes características:

- *Software* de programación: 4DIAC 1.7.3
- *Runtime:* Forte 1.7.3
- Ordenador: *Laptop* / Windows 7
- Sistema embebido: Raspberry Pi 3B/Linux.

3.1. Desarrollo de bloques de función

Se realiza el desarrollo de nuevos bloques de función con el objetivo de editar y diseñar según las variables de entrada y salida que requiera el proceso. Con el diseño de nuevos FB se libera de las limitantes ya establecidas por el creador del *software*. Para desarrollar dichos FB se utiliza el *software* 4DIAC-IDE en el cual se programan utilizando un editor de lenguaje C++ y posteriormente estos archivos se enlazan con el *runtime* FORTE. En la Figura 5 se muestra el diagrama de bloques en el cual están los pasos para desarrollar los bloques de función.

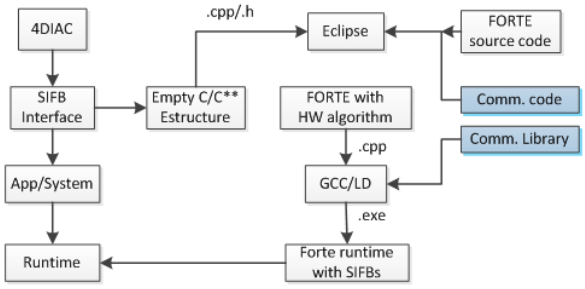

Figura 5. Desarrollo general de bloques de función [11].

3.1.1. Creación y programación de FB

Primero, se crean nuevos bloques de función para manipular las entradas y salidas digitales de la tarjeta.

Bloques de función para los sensores: Se desarrollaron nueve FB para tomar los datos de los sensores de las estaciones de manipulación y clasificación. Los sensores para la estación de manipulación son los siguientes: izquierda, derecha, abajo, arriba, pinza. Los sensores para la estación de clasificación son: inductivo, presencia, color y reflectivo. La configuración general se describe a continuación:

- PIN (INT). Sirve para asignar el número de pin de manera externa a cada sensor.
- VALUE1 (BOOL). Sirve para enviar el estado del sensor al siguiente FB.
- INIT (Event). Envía eventos de manera continua para que el FB envié los datos del estado del sensor cada 100 ms.
- REQ (Event). Envía un solo evento que está programado para que inicialice las librerías que se utiliza en el algoritmo.

En la Figura 6 se muestra el bloque de función creado para el sensor inductivo.

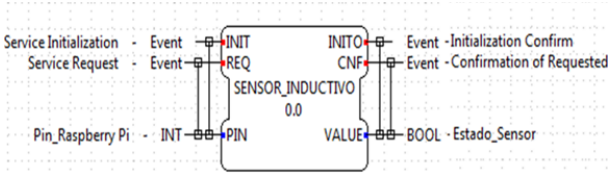

Figura 6. Bloque de función del sensor inductivo.

Bloques de función para los actuadores: Se crean ocho FB para control de actuadores. Los actuadores para la estación de manipulación son brazo izquierda, brazo derecha, brazo abajo y abre pinza. Los actuadores para la estación de clasificación son motor, pistón de espera, pistón rampa uno y pistón rampa dos. La configuración de los FB se describe a continuación:

- PIN (INT). Sirve para asignar el número de pin a utilizar por cada actuador de manera externa.
- VALUE (BOOL). Recibe el dato para activar o desactivar el actuador seleccionado.

Bloques de función para controlar la estación de manipulación: Este FB sirve para programar el algoritmo que controla la estación. El bloque de función toma el estado de los sensores del brazo realiza el algoritmo programado y activa las salidas de los actuadores. La configuración del FB se describe a continuación:

- VALUE1 (BOOL). Recibe el estado del sensor y luego se programa el algoritmo de control.
- ACT1 (BOOL). Envía el dato según el algoritmo programado para activar o desactivar el actuador asignado.

En la Figura 7 se muestra un FB creado para el control del brazo.

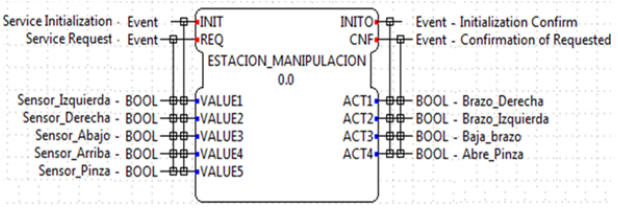

Figura 7. Bloque de función de la estación de manipulación.

Bloque de función para controlar la estación de clasificación: Este FB sirve para crear el algoritmo que controla la estación de clasificación. El FB toma los estados de los sensores realiza el algoritmo y activa los actuadores de la estación.

- VALUE1 (BOOL). Recibe el estado del sensor asignado para realizar la clasificación de piezas.
- ACT1 (BOOL). Envía el dato para activar o desactivar el actuador de la estación.

3.2. Diseño del acondicionamiento

Para realizar la etapa de acondicionamiento se revisan los voltajes y corrientes con los que trabajan los sensores, actuadores y también los pines GPIO de la tarjeta.

3.2.1. Acondicionamiento de la señal para sensores y actuadores

Se tiene que los sensores del MPS 500 trabajan con un voltaje de 24 V y los pines de entrada de la Raspberry Pi aceptan un voltaje de 3,3 V. Para acondicionar la señal se utilizara el *driver* CNY74-4 que es un optoacoplador que consta de un led infrarrojo y un fototransistor y permite realizar el acoplamiento de tierras.

3.2.2. Acondicionamiento de la señal para actuadores

Los actuadores del MPS 500 trabajan con un voltaje de 24 V por lo cual se debe acoplar los pines de salida de la tarjeta Raspberry Pi (3,3 V). La señal para los actuadores se acondiciona con el *driver* L293D que es un puente H con alimentación independiente para los actuadores y permite tener el control de cuatro salidas.

3.3. Diseño de la aplicación de control

Se realiza el diseño de la aplicación de control para cada estación y posteriormente se descarga en el *runtime* FORTE. Se debe configurar el sistema a utilizar y la aplicación en el *software* 4DIAC. Se conectan los FB de los sensores para adquirir los datos y enviar al proceso, de esta manera, el FB según el estado de las entradas y el algoritmo programado active los actuadores de la estación de manipulación como lo muestra la Figura 8.

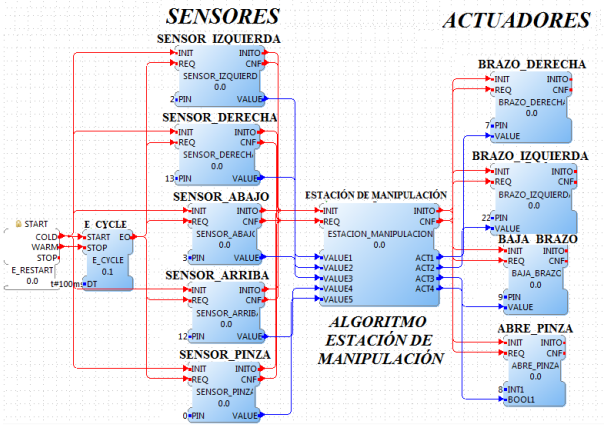

Figura 8. Arquitectura estación de manipulación.

De igual manera, se procede a enlazar los FB anteriormente diseñados para crear la aplicación de control en la estación de clasificación como lo muestra la Figura 9.

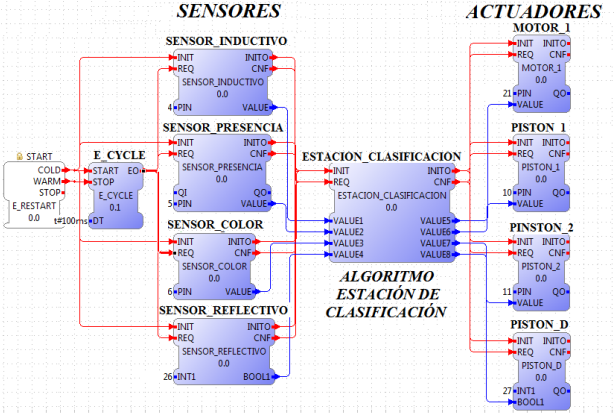

Figura 9. Arquitectura estación de clasificación.

Finalmente, se conectan los sensores y actuadores de las estaciones a la etapa de acondicionamiento y se descarga la aplicación mediante un cable Ethernet en la tarjeta Raspberry Pi. Se observa las nuevas alternativas que se pueden dar a los procesos de automatización gracias a los nuevos estándares.

4. Conclusiones

Mediante la implementación del estándar IEC-61499 para el desarrollo de aplicaciones de control distribuidas y el diseño de nuevos FB para el control, se observó el funcionamiento del proceso en donde se verifican los alcances y control que se pueden dar a un sistema, debido a que permite automatizar y modificar de manera fácil algún parámetro de la planta creando así sistemas reconfigurables y escalables. Al ejecutar el *runtime* FORTE en la Raspberry Pi y tener los otros programas en un ordenador diferente, evita el consumo de recursos sobre la tarjeta, mejorando así el tiempo de respuesta en el proceso. Esto se logra mediante la compilación cruzada donde es posible ejecutar los archivos creados de Windows dentro de la plataforma Linux. Además, al utilizar la tarjeta Raspberry Pi sobre una plataforma Debian/Linux mejora la flexibilidad y escalabilidad en el proceso, ya que utiliza la librería Open CV que es código abierto. El IEC-61499 es compatible con otros estándares de automatización, además, con varios dispositivos existentes en el mercado, por lo cual se pueden desarrollar diversas aplicaciones de control de procesos reduciendo el costo de su funcionamiento. En este sistema se automatizó el proceso con un sistema de bajo coste mediante la utilización de la tarjeta Raspberry Pi.

Agradecimientos

Los autores agradecen a la Escuela Superior Politécnica de Chimborazo (ESPOCH) por el apoyo brindado al desarrollo del proyecto, así como, a la Universidad Politécnica Salesiana (UPS) por permitir el uso de sus laboratorios y a la Universidad Técnica de Ambato (UTA) bajo el proyecto CONIN-P-107-2016

Referencias

[1] D. Ivanova, G. Frey, and I. Batchkova, "Intelligent component based batch control using iec61499 and ansi/isa s88," in *2008 4th International IEEE Conference Intelligent Systems*, vol. 1, Sept 2008, pp. 444–449. [Online]. Available: https://doi.org/10.1109/IS.2008.4670424

[2] M. V. García, F. Pérez, I. Calvo, F. López, and G. Morán, "Desarrollo de CPPS sobre IEC-61499 basado en dispositivos de bajo coste," *XXXVI Jornadas de Automática*, pp. 230–237, 2015.

[3] M. V. García and F. P. González, *Implementación de sistemas empotrados de control distribuido bajo el estándar IEC-61499*, Trabajo fin de Master, Universidad del País Vasco, 2013.

[4] M. de Sousa, "Analyzing the compatibility between isa 88 and iec 61499," in *2010*

IEEE 15th Conference on Emerging Technologies Factory Automation (ETFA 2010), Sept 2010, pp. 1–8. [Online]. Available: https://doi.org/10.1109/ETFA.2010.5641308

[5] M. Trejo-Hernández and E. López-Mellado, "Specification of manufacturing systems controllers using the standard iec61499," in *CONIELECOMP 2013, 23rd International Conference on Electronics, Communications and Computing*, March 2013, pp. 179–184. [Online]. Available: https://doi.org/10.1109/CONIELECOMP.2013.6525782

[6] E. Querol, J. A. Romero, A. M. Estruch, and F. Romero., "Norma IEC-61499 para el control distribuido aplicación al CNC," *Jornadas de Automatica*, pp. 3–5, 2014.

[7] M. van der Linde, "Using iec61499 to achieve smart grid automation through interconnected distribution reclosers," in *2016 Australasian Universities Power Engineering Conference (AUPEC)*, Sept 2016, pp. 1–6. [Online]. Available: https://doi.org/10.1109/AUPEC.2016.7749340

[8] S. Sierla, J. Christensen, K. Koskinen, and J. Peltola, "Educational approaches for the industrial acceptance of iec 61499," in *2007 IEEE Conference on Emerging Technologies and Factory Automation (EFTA 2007)*, Sept 2007, pp. 482–489. [Online]. Available: https://doi.org/10.1109/EFTA.2007.4416807

[9] C. Catalán., "Modelos y plataforma IEC61499 adaptados al control distribuido de máquinas herramientas en sistemas de fabricación ágil," Ph.D. dissertation, Universidad de Zaragoza, 2016.

[10] K. Thramboulidis, S. Sierla, N. Papakonstantinou, and K. Koskinen, "An IEC 61499 based approach for distributed batch process control," in *Industrial Informatics, 2007 5th IEEE International Conference on*, vol. 1. IEEE, 2007, pp. 177–182. [Online]. Available: https://doi.org/10.1109/INDIN.2007.4384752

[11] I. Calvo, F. Pérez, I. Etxeberria, and G. Morán, "Control communications with DDS using IEC61499 service interface function blocks," in *2010 IEEE 15th Conference on Emerging Technologies Factory Automation (ETFA 2010)*, Sept 2010, pp. 1–4. [Online]. Available: https://doi.org/10.1109/ETFA.2010.5641207

[12] A. Zoitl, T. Strasser, and G. Ebenhofer, "Developing modular reusable iec 61499 control applications with 4diac," in *2013 11th IEEE International Conference on Industrial Informatics (INDIN)*, July 2013, pp. 358–363. [Online]. Available: https://doi.org/10.1109/INDIN.2013.6622910

[13] J. H. Christensen, T. Strasser, V. Vyatkin, and A. Zoitl., "The IEC 61499 Function Block Standard?: Software Tools and Runtime Platforms," *Presented at ISA Automation*, 2012.

[14] Eclipse. (2017) 4diac. [Online]. Available: https://goo.gl/6Z55em

[15] R. P. Foundation. (2017) Raspberry pi. [Online]. Available: https://goo.gl/4QSy6Y

[16] ——. (2017) Gpio: Raspberry pi 3b. [Online]. Available: https://goo.gl/ZANVso

[17] Festo. (2017) MPS 500-FMS. [Online]. Available: https://goo.gl/l5qHJo