REVISTA ELECTRO, Vol. 44 pp. 14-20, Oct 2022, Chihuahua, Chih. México
http:// electro.itchihuahua.edu.mx/revista
ISSN 1405-2172

OPERACIÓN DIRIGIDA POR MODELO EN SISTEMAS BASADOS EN PLC

Acosta Cano de los Ríos, José Eduardo; Ronquillo Tercero, Héctor Mario, Acosta Cano de los Ríos Pedro R., Chávez López Óscar Arturo
Tecnológico Nacional de México/I.T. Chihuahua
Laboratorio de Automática e Informática Industrial
Ave. Tecnológico No. 2909, 31310, Chihuahua, Chih. México.
(614)2026511
jose.ac@chihuahua.tecnm.mx, hmronquillotercero@gmail.com

RESUMEN.
Las empresas manufactureras se enfrentan a requerimientos de flexibilidad que impone el ambiente turbulento del mercado a su sistema de fabricación. El uso de equipos programables facilita hacer frente a los cambios requeridos en la lógica de operación de equipos de producción, situación que dio origen al controlador lógico programable (PLC). Es evidente que con los equipos basados en PLC implementar el cambio (en código) en lugar de realizarlo en hardware redujo considerablemente el esfuerzo para adaptar una nueva lógica de operación del equipo. Sin embargo, los avances en ingeniería de software no han sido reflejados en el ambiente del PLC de la manera en que se encuentra en el desarrollo de software para sistemas basados en computadora personal. En el presente artículo se muestra una manera de implementar el concepto de operación dirigida por modelo que permite desarrollar sistemas flexibles basados en PLC, mediante el uso del paradigma orientado a objeto y la técnica de modelado de particularidades identificada como iMRP y el concepto de operación dirigido por modelo planteado en el esquema genérico ArquiTAM. Se muestra la aplicación de un sistema genérico para el control de distintos sistemas de control utilizando la plataforma IDE CODESYS para la validación del concepto presentado.

ABSTRACT.
Manufacturing companies face flexibility requirements imposed on their manufacturing system by the turbulent market environment. The use of programmable equipment makes it easier to face the changes required in the operation logic of production equipment, A situation that gave rise to the programmable logic controller (PLC). It is evident that in plc-based equipment, implementing the change in code instead of doing it in hardware has considerably reduced the effort to adapt a new equipment operation logic. However, advances in software engineering area have not been reflected in the plc environment in the way that it is found in software development for personal computer-based systems. This article shows a way to implement the concept of model based operation system that allows the development of flexible plc-based systems, through the use of the object-oriented paradigm and the particularity modeling technique identified as imrp and the model-driven operation concept proposed in the article. arquitam generic schema. the application of a generic system for the control of different control systems using the ide platform codesys for the validation of the concept presented is shown.
Keywords: PLC software reusability, Model Based Operation, generic system, ArquiTAM, Low Code.

1. INTRODUCCIÓN.

Los entornos de manufactura están constantemente sometidos a cambios ya sea por el ciclo de vida del producto, equipos obsoletos, cambio de la demanda, etc. Es por ello que es necesaria la existencia de un alto grado de flexibilidad y reutilización de código en los PLC para poder atender los cambios que ocurren en su entorno de una manera más sencilla. En respuesta a ello han surgido intentos para apoyar a esta problemática como lo es la incorporación del Paradigma Orientado a Objetos en su programación mediante Bloques de Función [1]. Dentro de este ámbito en [2] se expresa que es posible reutilizar código mediante sistemas de control basados en modelos, pero estos modelos implican crear sistemas específicos donde cualquier cambio que ocurra en el sistema de control requiere de la modificación del modelo y por consiguiente su programación [3]. Otro problema presente en la programación de PLC ha sido la falta de cooperación por los distintos fabricantes en el mercado para fomentar la interoperabilidad de código entre las diferentes marcas existentes, que a pesar de apegarse al estándar IEC 61131-3 el código sigue siendo incompatible de un entorno a otro [4] [5].

La reutilización de código se ha trabajado mediante el desarrollo de sistemas informáticos basado en modelos (MDA) [2] **en** donde gráficamente se expone de la interacción entre sus elementos [8]. Tales herramientas también han evolucionado para su aplicación en sistemas basados en PLC, entre las que se encuentran las técnicas basadas en UML (SysML). Tales herramientas se basan en la generación de código a partir del modelo del sistema específico. Esta situación implica la modificación del modelo y consecuentemente del programa en caso de necesitar realizar un cambio en el sistema [3]. Es por

ello que resulta de utilidad una herramienta que simplifique las complejidades de los modelos actuales, así mismo que fomente la reutilización de código y sea flexible a los cambios del entorno (aplicaciones) de un PLC.

Por lo anteriormente descrito, en el presente trabajo se propone la aplicación del concepto de operación dirigida por modelo para el desarrollo de sistemas de control flexible basado en PLC en donde sea posible reutilizar código y abordar cambios en la lógica y/o en el entorno de control de manera sencilla mediante la técnica de modelado de particularidades iMRP. En la implementación de la solución se utiliza la plataforma de desarrollo CODESYS independiente de fabricante en donde se implementa el sistema genérico y se instancia la aplicación particular mediante la aplicación del concepto de operación dirigida por modelo. El fundamento básico de la solución es que el sistema de control basado en PLC interpreta (sin generación de código) el modelo (gráfico) de particularidades, tal como se describe en el esquema de referencia ArquiTAM [3]. El esquema ArquiTAM señala reutilizar el código del sistema en diferentes casos específicos de aplicación, mediante un sistema genérico (reutilizable) al cual se añaden únicamente las particularidades del caso específico a controlar (modelo de particularidades gráfico iMRP). Facilitando así la aplicación del sistema genérico en distintos casos específicos de control.

2. MARCO TEÓRICO.

2.1. Antecedentes.

Los sistemas de manufactura han evolucionado a lo largo del tiempo optando por sistemas genéricos flexibles, en lugar de los costosos sistemas personalizados. Así mismo, la tendencia ha sido realizar cambios en software en lugar del hardware [6]. Un soporte relevante en esta dirección es el concepto del Paradigma Orientado a Objetos, el cual ha incidido en la manera de programar permitiendo abstraer elementos de la realidad en objetos que facilitan reusabilidad, dividiendo un problema en subproblemas [7]. De igual manera esta evolución en la manera de programar ha impactado a los Controladores Lógicos Programables incorporando características del Paradigma Orientado a Objetos en forma de Bloques de Función. No obstante su incidencia en el desarrollo de sistemas de control basados en PLC, todavía en la mayoría de los casos implica un esfuerzo relativamente considerable el trabajo de adaptación de un sistema ante cambios en el entorno de control. Sin embargo, e otro aspecto del problema de adaptación se encuentra en que la mayoría de los fabricantes de PLC usan sus propios protocolos, impidiendo interoperabilidad, aún y cuando estén apegados al estándar IEC 61131-3 [4][5]. Existen intentos para solucionar esta problemática como es el estándar IEC 61499 pero este no ha sido ampliamente aceptado.

2.2. ArquiTAM.

ArquiTAM (Arquitectura de referencia del Taller Automático de Manufactura) es un esquema de referencia propuesto en [3] para el desarrollo de sistemas informáticos desde un modelo conceptual básico hasta un Sistema Particular. Se compone de tres niveles de abstracción, figura 1: Meta Modelo (Arquitectura de Referencia), Modelo de Referencia (Marco de Trabajo) y el Modelo Particular (caso específico de aplicación).

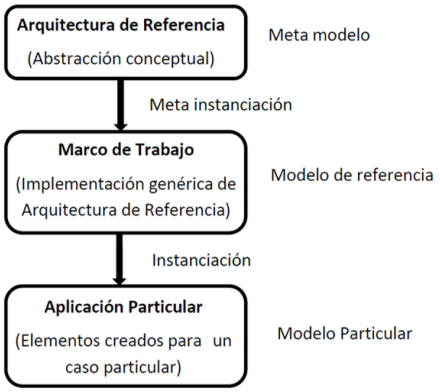

Figura 1 Estructura del esquema de referencia ArquiTAM [3].

La arquitectura de referencia, es un marco conceptual donde se definen requerimientos y posibles soluciones sin tomar en cuenta detalles de implementación (codificación) [3]. El nivel Modelo de Referencia, consiste en la implementación genérica en código del sistema con base en la abstracción (meta instancia) de conceptos y soluciones planteadas en la arquitectura de referencia (o Meta Modelo). Este modelo es un marco de trabajo (framework) en el que se definen componentes reutilizables (definiendo los elementos como clases) que permiten implementar de forma genérica código al momento de instanciar aplicaciones particulares en el sistema; en las clases implementadas se abstraen en objetos software (código) los recursos del sistema de producción. Por último, las aplicaciones particulares,el Modelo Particular, se define como la instanciación del Modelo de Referencia, reutilizando el código implementado en el framework además de las características particulares del caso específico de aplicación.
Es así que el Modelo Particular de un sistema específico se define como la suma de las generalidades (implementadas en el Modelo de Referencia) y las particularidades del sistema, ver figura 2, en donde las particularidades resultan ser más sencillas de modelar que el abstraer el Sistema Particular de manera completa. Así mismo, el modelo de particularidades será más sencillo de realizar así como de interpretar por un sistema informático, que el modelo particular completo, [3].

Figura 2 Simplificación de la interpretación del modelo mediante particularidades.

2.3. iMRP - Abstracción de las particularidades del piso de producción.

En el esquema ArquiTAM se plantea la interpretación del modelo de particularidades por el sistema genérico (framework) para su aplicación en el control de un caso específico. Así mismo, se describe la técnica de modelado, [3]. La técnica iMRP permite la abstracción de las particularidades del piso de producción incorporando los aspectos estáticos y dinámicos de un proceso de manufactura en un mismo modelo, a utilizar en el concepto de Operación Dirigida por Modelo, esto es, el modelo resulta interpretable por el sistema informático (framework).

El iMRP es una herramienta gráfica y se basa en constructores de tipo nodo y arco. Los nodos cuatro tipos de grafos que abstraen (clases) los cuatro tipos de recursos: Pieza, Operación, Equipo y Recurso Tiempo. Donde las interacciones entre estos elementos describen la estructura y dinámica de un proceso de producción. El grupo de nodos tipo pieza forman el grafo Producto en forma de BOM (por sus siglas en ingles Bill Of Materials), el grafo Equipo, nodos equipo, representa la manera en que se agrupan los equipos (como una estación de trabajo) y el grafo Operación comprende las posibles operaciones a realizar en una pieza (Alimentar/Almacenar, Manipular, Procesar y Transportar). Los grafos representan la parte estática (estructural) de cada tipo de elemento (recurso). Por otra parte, el Recurso Tiempo, el cual representa una actividad para la elaboración de un producto; definen la parte dinámica del proceso.

Como se describe en [3] el constructor tipo nodo del modelo de particularidades contiene atributos que comprenden: Tipo (tipo de clase de la que el nodo es abstraído), ID, GrafoID, Nombre, Nivel y variables para almacenar información (dependen del tipo de grafo). Mientras que los constructores arco, poseen atributos para representar las características particulares de cada asociación entre nodos, sus atributos dependen de los nodos que el arco asocie.

En la figura 3 se muestra un modelo de particularidades empleando la técnica de modelado iMRP [3], en el que se abstrae el grafo producto (en el que se representan las etapas de procesado del producto desde el estado inicial hasta el final de su proceso de fabricación). La secuencia de fabricación de las piezas se basa en al grafo producto, comenzando con los nodos hijos hasta llegar al nodo padre (primero el producto A hasta llegar al C). Cabe destacar que al aplicar más de un proceso sobre una misma pieza (pieza A, B o C) al nodo Recurso Tiempo en cuestión se le agrega un nodo hijo Recurso Tiempo y así sucesivamente si requiere de más procesos. Una descripción detallada de la técnica se muestra en [3].

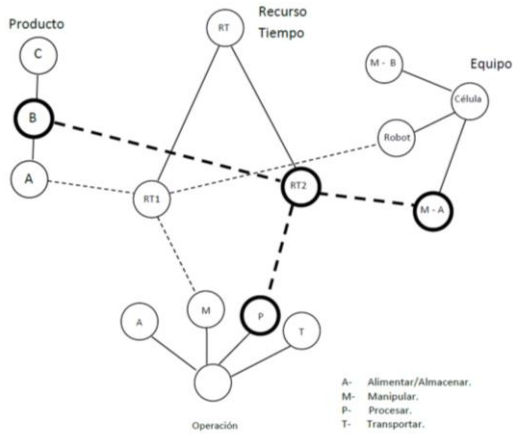

Figura 3 Representación de un modelo particular iMRP [3].

3. DESARROLLO

En la solución propuesta se trabaja con la plataforma CODESYS de desarrollo independiente de fabricante para sistemas PLC. En el presente proyecto se desarrolló un Editor iMRP y un Generador de Instrucciones, en VisualBasic.Net.. El Editor iMRP es una interfaz de usuario para la creación y edición de los modelos de particularidades iMRP. El editor permite graficar el modelo mediante nodos y arcos, así como incorporar la información referente a las particularidades de cada aplicación. Así mismo se implementó un programa genérico (framerwork o marco de trabajo) a operar en el PLC como interprete del modelo gráfico de particularidades representado en el PLC en forma matricial. Tal programa se implementó utilizando la plataforma CODESYS. En el presente trabajo el modelo de particularidades iMRP se adapta para ser incorporado en la Operación Dirigida por Modelo en sistemas basados en PLC, ya que los controladores lógicos programables automatizan procesos industriales, el recurso Producto es denominado recurso Fase. De la misma manera, los PLCs operan Actuadores por lo que el recurso Equipo es denominado recurso Actuador. En la figura 4 se muestra un grafo iMRP realizado en el entorno del Editor iMRP con las modificaciones mencionadas. En la figura 5 se muestran los atributos de un constructor tipo arco y nodo desde el Editor iMRP,

Por otro lado, el Generador de Instrucciones (operando en una PC) es una aplicación que se encarga de crear un archivo PLCOpenXML (para ser importado al PLC CODESYS que consiste en el sistema genérico mas el modelo de particularidades (especificaicón gráfica del programa de control) del proceso específico a controlar. El modelo de particularidades generado en por el generador de instrucciones junto con el modelo de referencia (framework) forman el modelo particular El Modelo Particular está conformado por el Modelo de Referencia (sistema genérico o framework) y el código del modelo de particularidades generado en PLCOpenXML.

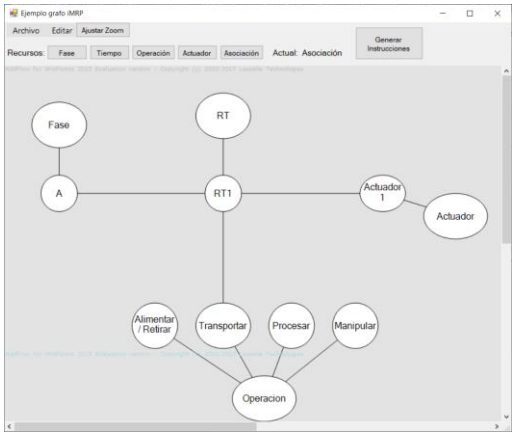

Figura 4 Ejemplo de un grafo iMRP adaptado para sistemas basados en PLC.

Figura 5 Atributos de un objeto arco y nodo.

Es importante destacar que el modelo de particularidades incluye la definición de la clase Actuador, la cual es añadida al Modelo Particular por incorporación dinámica de código en tiempo de ejecución de un objeto del tipo especificado en el atributo "tipo" del nodo actuador , como se observa en la figura 5. Así mismo, la clase Actuador proporciona flexibilidad a los modelos iMRP, ya que esta contiene información propia para el tipo de actuadores a emplear en el Modelo Particular. Las particularidades que añade la clase Actuador al Sistema Particular corresponden a la declaración de variables de entradas, salidas, temporizadores, contadores, entre otros (correspondientes a la aplicación particular que se esté modelando). El código asociado a la clase actuador para el PLC es escrito por el usuario (característica de programación Low Code) en un archivo de Microsoft Word que es anexado a la clase Actuador (DLL de la clase Actuador) por el generador de instrucciones. El código de la clase actuador resulta sencillo de programar (Texto estructurado) ya que sólo consiste en la operación básica del actuador respectivo sin tomar en cuenta la lógica de operación del sistema completo (lógica que es modelada gráficamente en iMRP).

En la figura 6 se expone un diagrama de bloques que explica el funcionamiento del sistema de control desarrollado para operar un sistema basado en PLC con base en el modelo de particularidades iMRP. El usuario crea el modelo de particularidades del sistema específico a controlar en el Editor iMRP para después utilizar la aplicación Generador Instrucciones donde el sistema informático es quien utiliza la información del modelo de particularidades (grafo iMRP) y lo añade el código genérico del Modelo de Referencia, conformando así el programa específico o Modelo Particular plasmado en un archivo PLCOpenXML. El formato del archivo de programa a utilizar depende del PLC a utilizar en el sistema. Para la solución propuesta se elige el formato PLCOpenXML para trabajar la información de la programación del PLC ya que ofrece características de interoperabilidad permitiendo importación y exportación de proyectos IEC 61131-3 entre entornos de programación de diferentes PLCs, como es el caso de CODESYS.

Figura 6 Diagrama de Bloques de la solución propuesta empleando Operación Dirigida por Modelo.

3.1. Modelo de Referencia

El sistema genérico que interpreta al modelo de particularidades es implementado en el PLC (entorno de desarrollo CODESYS lenguaje PLCOpenXML). Tal sistema es de aplicación genérica (para cualquier caso particular). En el Editor iMRP se modelan gráficamente las particularidades respecto a la lógica y estructura del sistema particular. En este modelo de particularidades se identifican las clases en donde se encuentra la codificación de la operación de cada uno de los actuadores. El sistema informático (Generador de Instrucciones) convierte a forma matricial el modelo gráfico de particularidades en código PLCOpenXML para incorporarlo al código principal del sistema genérico. Proceso a realizar con base en la identificación del archivo dll de cada actuador (iMRP) y el concepto de incorporación dinámica de código. De esta manera es generado el archivo con el programa completo para el control de un sistema especifico basado en PLC. o PLCOpenXML listo para cargarlo en ell PLC yrealizar la operación del sistema de control.

El código del Modelo del sistema (aplicación genérica) está integrado por un programa principal (PLC_PRG) y tres Bloques de Función (clases CFase, CRT y CActuador) para representar cada uno de los tres tipos de recursos del modelo de particularidades (iMRP), Figura 7. El PLC_PRG dentro del PLC es código para inicializar y gestionar la operación de cada uno de los objetos del modelo de particularidades (Fase, RT y Actuador). Este código es el encargado de interpretar el modelo

matricial (iMRP) de particularidades del control en particular. El modelo de particularidades en su representación matricial brinda al sistema genérico la información necesaria para realizar la Operación Dirigida por Modelo en el PLC (CODESYS).

Es así como el PLC_PRG manda a operar a los objetos de la clase CFase según la jerarquía presente en el modelo de particularidades, donde a su vez los objetos Fase en operación mandan a operar a los objetos RT (de la clase CRT) correspondientes a la fase, mientras que los objetos RT operan a los objetos Actuador correspondientes. Es importante destacar que el código del programa PLC-PRG y de las clases CFase y CRT es el mismo independientemente de la aplicación del sistema. Reduciendo así la programación del sistema para una aplicación particular a el proceso de modelado iMRP y el código del actuador. El código de la clase actuador correspondiente es incorporado por el usuario desde la clase Actuador en el Editor iMRP, dicho código representa la operación básica de cada objeto Actuador dependiendo del RT que lo opere. Teniendo en cuenta que una Fase puede estar asociada a varios RT, pero un RT sólo puede estar asociado a un Actuador.

3.2. Archivo PLCOpenXML

El código del Sistema Particular (figura 7), contenido en la aplicación es implementada en lenguaje PLCOpenXML por el Generador de Instrucciones que opera en la PC..

Al ejecutar el Generador de Instrucciones las particularidades del modelo iMRP son añadidas al Modelo de Referencia dentro de los POUS PLC_PRG y CActuador, para generar el archivo PLCOpenXML, En el programa principal (PLC_PRG) se añaden las particularidades correspondientes a la información de la construcción del modelo iMRP en forma de declaración de arreglos en XML, mientras que en el bloque de función CActuador el Generador de Instrucciones agrega la información de la clase Actuador incorporada al modelo iMRP, esto es, el código de operación escrito por el usuario así como la declaración de las variables que se utilizan en ese código de operación para la aplicación particular en específico. Por lo tanto, al ser particularidades esta es la única información que cambia de un Sistema Particular a otro, es decir, es la parte que cambia de un archivo PLCOpenXML a otro, simplificando así el desarrollo del programa de control.

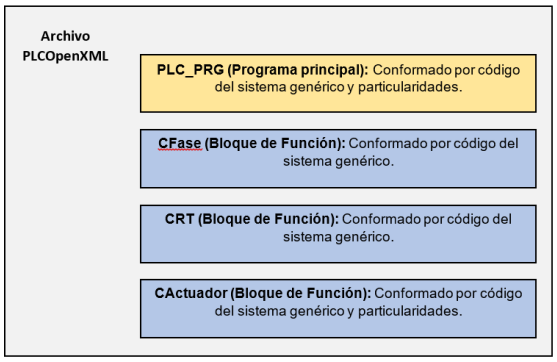

Figura 7 Contenido del archivo PLCOpenXML.

4. OPERACIÓN DIRIGIDA POR MODELO.

A manera de ilustrar la Operación Dirigida por Modelo, en la figura 8 se expone como ejemplo el sistema de control de un Silo, (simulado en la plataforma CODESYS). El proceso a controlar es el siguiente: cuando el usuario presione el botón START se debe transportar una caja y detenerse debajo de la tolva con la señal de un Sensor de Presencia, una vez debajo de la tolva el solenoide de la Válvula de Llenado se activa para empezar a llenar la caja, y después detener el llenado cuando el Sensor de Llenado ubicado en la tolva se active, por último, al estar llena la caja debe de ser transportada para retirarla de la tolva y del Sensor de Presencia. Al ser una simulación, los sensores de este sistema de control cuentan con un botón para ser activados de manera manual.

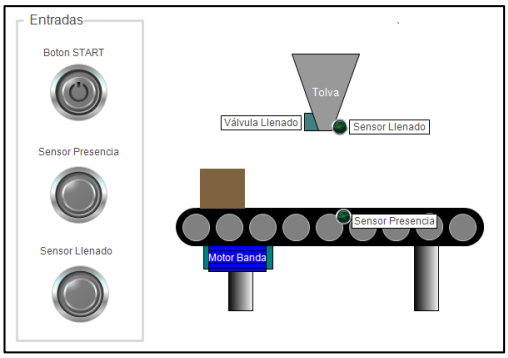

Figura 8 Ejemplo de la aplicación particular de un Silo.

En la figura 9 se muestra el modelo iMRP perteneciente al control de operación del silo. Los nodos actuador pueden ser una abstracción de actuadores físicos o variables internas que maneja el PLC para producir un resultado.

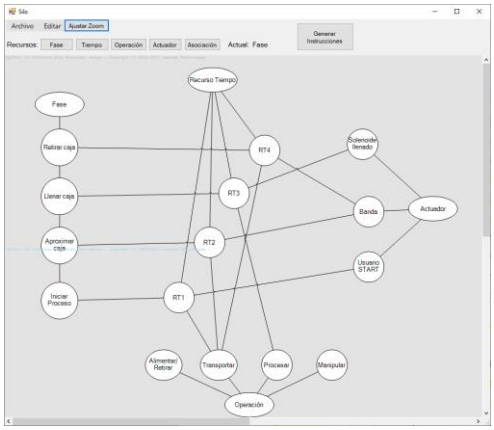

Figura 9 Ejemplo de un modelo iMRP de un Silo creado en el Editor iMRP.

De acuerdo a los requerimientos, el modelo iMRP de la figura 9 abstrae cuatro fases a desempeñar (cada una con su respectivo nodo RT) y tres actuadores a utilizar en el proceso. Los actuadores están conformados por: el "Usuario START" (el cual representa a una persona considerada como un actuador manual y realiza una operación de oprimir el botón START), la "Banda" (el motor de la banda) y el "Solenoide de llenado".

Al presiona el botón "Generar Instrucciones" en el Editor iMRP (aplicación en vb Net), se ejecuta la aplicación Generador de Instrucciones, que interpreta las particularidades de la aplicación particular y las incluye en el archivo PLCOpenXML, a ser cargado en el PLC. Entre estas particularidades se encuentra la declaración de los arreglos que describen el modelo gráfico de particularidades. En una hoja de Excel se escribe una tabla de manera automática estos arreglos y después se leen para hacer la declaración de dichos arreglos en dos dimensiones en el Modelo de Referencia (código del PLC).

En la figura 10, se muestra la tabla Fase escrita en Excel para describir la construcción del recurso Fase del modelo iMRP. La representación del grafo en la tabla se realiza de la siguiente manera. Un "1" significa que el nodo del renglón es el mismo que el de la columna (para indicar que es el nodo padre) y seguidamente de esa columna escribir de igual manera un "1" en las columnas de los nodos que sean hijos de ese nodo padre.

Fase	Retirar caja	Llenar caja	Aproximar caja	Iniciar Proceso
Retirar caja	1	1	0	0
Llenar caja	0	1	1	0
Aproximar caja	0	0	1	1
Iniciar Proceso	0	0	0	1

Figura 10 Tabla de construcción del recurso Fase escrita en Excel.

Ahora bien, el código para la operación de cada actuador es escrito por el usuario y se agrega como particularidades al sistema genérico en el POU CActuador (PCLOpenXML, por el Generador de Instrucciones). La operación referente a este ejemplo (Silo) se muestra en la fig. 11donde se utilizan las banderas bEnOp y bFinOp para indicar cuando se opera y se termina de operar un actuador.

```
// RT1.- Fase Asociada: Iniciar Proceso, Actuador Asociado: Usuario al
presionar el botón START
  1: bEnOp:= NOT(BotonStart);
           bFinOp:= BotonStart;
// RT2.- Fase Asociada: Aproximar caja, Actuador Asociado: Banda
   2: IF SensorPresencia = FALSE THEN
                MotorBanda:=TRUE;
                bFinOp:= FALSE;
                bEnOp:= TRUE;
          ELSE
                MotorBanda:= FALSE;
                bFinOp:= TRUE;
                bEnOp:= FALSE;
          END_IF

// RT3.- Fase Asociada: Llenar caja, Actuador Asociado: Solenoide Llenado
  3: IF SensorLleno = FALSE THEN
                ValvulaLlenado:= TRUE;
                bEnOp:= TRUE;
                bFinOp:= FALSE;
          ELSE
                ValvulaLlenado:= FALSE;
                bEnOp:= FALSE;
                bFinOp:= TRUE;
     END_IF
// RT4.- Fase Asociada: Retirar caja, Actuador Asociado: Banda
  4: IF SensorPresencia = TRUE THEN
                MotorBanda:=TRUE;
                bEnOp:= TRUE;
                bFinOP:= FALSE;
          ELSE
                MotorBanda:= FALSE;
                bEnOp:= FALSE;
                bFinOp:= TRUE;
          END_IF
```

Figura 11 Código de operación de actuadores del Silo.

5. VALIDACIÓN.

Para validar la aplicación del sistema genérico y el concepto de operación dirigida por modelo en diferentes casos específicos de control, el sistema se aplicó en diferentes casos de control mediante simulaciones y visualizaciones en el IDE CODESYS. Tales simulaciones consistieron en la implementación del los sistemas de control un silo, una cochera, un semáforo, un reactor químico, un elevador, figuras 8, 12,13,14 y 15. El sistema genérico es particularizado a cada uno de los sistemas de control específico mencionados utilizando el modelo gráfico de particularidades (iMRP). Posteriormente la operación del sistema es dirigida por el propio modelo de particularidades desarrollado para cada caso específico. Las modificaciones al código requeridas durante el desarrollo de cada aplicación resultaron mínimas, dado que el único código a modificar resulta ser el correspondiente a la rutina operación básica de la clase de cada actuador. Código que consiste en un número reducido de líneas, ya que la lógica de operación entre actuadores es descrita en el modelo gráfico de particularidades a interpretar por el sistema genérico.

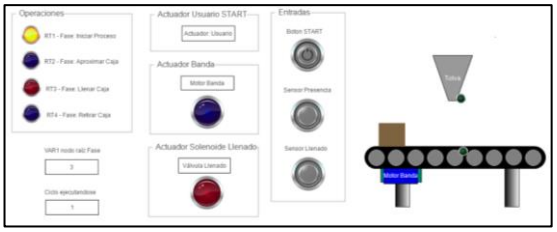

Figura 12 Visualización al operar el RT1 del Silo.

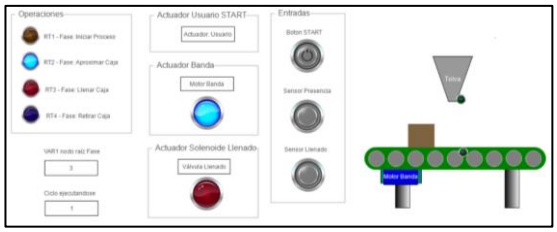

Figura 13 Visualización al operar el RT2 del Silo.

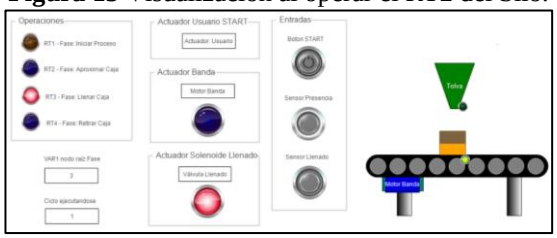

Figura 14 Visualización al operar el RT3 del Silo

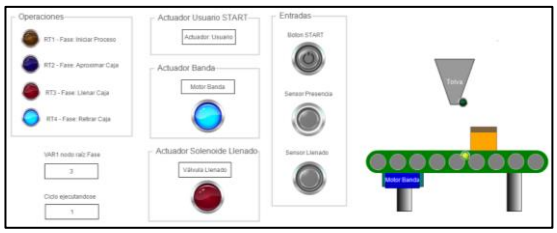

Figura 15 Visualización al operar el RT4 del Silo

6. CONCLUSIONES

La aplicación de un sistema genérico de control basado en PLC, en diferentes casos particulares es posible mediante el concepto de instanciación del modelo particular a partir del modelo de referencia y el modelo de particularidades, tal como se plantea en el esquema ArquiTAM.

El sistema genérico (modelo de referencia) de control desarrollado para sistemas basados en PLC funcionó de manera satisfactoria. En cada caso particular de aplicación el sistema genérico implementó de manera satisfactoria el sistema de control particular (modelo particular) y realizó la Operación Dirigida por Modelo, donde el modelo gráfico de particularidades (iMRP) dirige la secuencia del proceso y operación de sus actuadores. El código a escribir por el usuario se reduce a la escritura de la rutina de operación básica del actuador correspondiente, siendo ésta de un número muy reducido de líneas. Por tal situación es posible afirmar que la operación dirigida por modelo conduce a la realización del concepto de desarrollo de sistemas de bajo código (low code) en el ambiente de sistemas basados en controlador lógico programable (PLC).

7. REFERENCIAS.

[1] Simon, T., y Rösch, S. Comparing the object-oriented extension with the classical IEC 61131-3 regarding reusability and understandability - a case study, 2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA), (2015), Luxemburgo, pp. 1-8.

[2] Acosta, J. E., y Sastrón, C. F. PRODUCTION SHOP FLOOR SYSTEM MODELING WITH REUSABILITY FOCUS. IFAC Proceedings Volumes, 40(19), (2007), 12–17.

[3] Acosta, J.E. Esquema de Referencia para Acoplamiento Débil entre Sistema Informático y Equipo de Producción. Tesis Doctoral. Universidad Politécnica de Madrid, (2016).

[4] Estévez, E., Marcos, M., y Orive, D. Automatic generation of PLC automation projects from component-based models. The International Journal of Advanced Manufacturing Technology, 35(5-6), (2007), 527–540.

[5] Sunder, C., Zoitll, A., Christensen, J., Steininger, H., y Rritsche, J. Considering IEC 61131-3 and IEC 61499 in the context of component frameworks. 2008 6th IEEE International Conference on Industrial Informatics, Daejeon, 2008, pp. 277-282.

[6] Papakonstantinou, N., y Sierla, S. Generating an Object Oriented IEC 61131-3 software product line architecture from SysML. 2013 IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA), Cagliari, (2013).

[7] Prause, M., y Weigand, J. Industry 4.0 and Object-Oriented Development: Incremental and Architectural Change. Journal of Technology Management & Innovation, 11(2), (2016), 104–110.

[8] Flowers, R., y Edeki, C. Business Process Modeling Notation. International Journal of Computer Science and Mobile Computing, 3(2), (2013), 35-40.