XLIV Jornadas de Automática 2023

Gemelo funcional para pruebas del software de sistemas de producción automatizados

Álvarez, M.L. *, Sarachaga, I., Burgos, A., Iriondo, N.

Departamento de Ingeniería de Sistemas y Automática, Escuela de Ingeniería de Bilbao, UPV/EHU

To cite this article: Álvarez, M.L., Sarachaga, I., Burgos, A., Iriondo, N. 2023. Functional twin for software testing of automated production systems. XLIV Jornadas de Automática, 831-836. https://doi.org/10.17979/spudc.9788497498609.831

Resumen

En el ámbito del paradigma de la Industria 4.0, el desarrollo del software de control de los nuevos sistemas de producción automatizados (aPS) se ha convertido en un proceso cada vez más complejo donde toman mayor importancia las pruebas del software. Este trabajo contribuye con una propuesta de desarrollo de gemelos funcionales, construidos a partir de componentes funcionales básicos reutilizables correspondientes a los módulos de control identificados en una unidad de trabajo de un aPS. Este gemelo funcional es una representación virtual de la funcionalidad del aPS y sirve como herramienta de pruebas para el software de control. Su construcción se realiza en el mismo entorno en el que se desarrolla el software de control siguiendo el estándar IEC 61131-3, sin requerir herramientas comerciales de desarrollo de plantas virtuales.

Palabras clave: Sistemas de producción automatizados, Control de plantas de fabricación, Implementación digital, Pruebas de software.

Functional twin for software testing of automated production systems

Abstract

In the context of the Industry 4.0 paradigm, the development of control software for new automated production systems (aPS) has become an increasingly complex process where software testing plays a significant role. This work presents a proposal for the development of functional twins, built from reusable basic functional components corresponding to the control modules identified in a work unit of an aPS. This functional twin is a virtual representation of the aPS functionality and serves as a testing tool for the control software. Its construction takes place in the same environment where the control software is developed following the IEC 61131-3 standard, without requiring commercial tools for the development of virtual plants.

Keywords: Automated production systems, Manufacturing plant control, Digital implementation, Software test.

1. Introducción

En la actualidad se puede apreciar un incremento notable en el grado de complejidad del software de control de los sistemas de producción automatizados (aPS) para afrontar la innovación y los retos tecnológicos de la Industria 4.0 (IEC, 2017), permitiendo su integración en un entorno global interconectado. Su desarrollo es un proceso complejo, ya que se integran las tecnologías de la información con las tecnologías de operación del entorno industrial con objeto de transformar información en acciones de dispositivos de un modo controlado, afectando a su coste, tiempo, calidad y flexibilidad.

Las pruebas constituyen un aspecto fundamental en el ciclo de desarrollo, dado que van a evidenciar que el software y los productos asociados cumplen con los requisitos (funcionales y

no funcionales), satisfacen los estándares, prácticas y convenciones, y responden a sus objetivos comerciales y las expectativas de las partes interesadas.

Una herramienta de pruebas muy potente para el software de control de un aPS es emplear plantas virtuales que constituyen uno de los pilares de la digitalización en la industria. En ese contexto, el concepto de gemelo digital (DT - Digital Twin) está muy extendido con diferentes objetivos y niveles de aplicación, bien para conectarlo con el aPS o bien para simularlo. Una revisión de la literatura sobre DT en fabricación se recoge en (Kritzinger et al., 2018), proponiendo una clasificación en función del nivel de integración del DT con el aPS: Digital Model, Digital Shadow y Digital Twin.

En lo que respecta a las pruebas de software de control, (Orive et al., 2019) presentan una metodología de desarrollo de un gemelo digital en la herramienta NX-MCD de Siemens PLM. La inclusión de fallos de proceso al gemelo digital permite testear la reacción del sistema de control en una puesta en marcha virtual. De acuerdo a la clasificación citada, dicho gemelo se correspondería con un Modelo Digital ya que es una representación funcional del proceso real que no utiliza ninguna forma de intercambio automatizada de datos con el proceso real, únicamente lo simula.

En este trabajo se presenta un método de desarrollo que permite construir gemelos con distintos niveles de granularidad, desde un dispositivo de control hasta un aPS. Para su construcción no se requiere de herramientas comerciales de desarrollo de plantas virtuales, sino que se implementa siguiendo el estándar IEC 61131-3 en el mismo entorno en el que se desarrolla el software control. Denominado Gemelo Funcional (GF), se concibe a partir de Componentes Funcionales Básicos (CFB) de mayor nivel de granularidad. El GF permite simular el comportamiento del aPS, realizando la misma función que un Modelo Digital.

La estructura del artículo es la siguiente: el apartado 2 presenta los antecedentes abordando las fases de pruebas del ciclo de desarrollo de software de control de aPS propuesto en la metodología MeiA. El apartado 3 presenta la metodología de desarrollo de CFB, así como de los GF. En el apartado 4 se aplica la metodología a la creación de un GF para una unidad de trabajo de una célula de ensamblado. El último apartado presenta las conclusiones del trabajo.

2. Antecedentes

En este trabajo el sistema de control del aPS se concibe como un conjunto de componentes MeiA (Iriondo et al., 2022) que ofrecen su funcionalidad a través de una interfaz que oculta todos los aspectos de implementación. Como ciclo de desarrollo para su construcción se adopta el modelo en V de la metodología MeiA 4.0 (Burgos et al., 2021), que establece la secuencia de los procesos propuestos en el estándar ISO/IEC/IEEE 12207 “Procesos del ciclo de vida del software” (ISO/IEC/IEEE, 2017). Estos procesos guían la transformación de los requisitos en proyectos de automatización de las unidades de trabajo del aPS, según el estándar IEC 61131-3 (IEC, 2010), que proporcionan diferentes modos de mando e interconexión con otros componentes MeiA.

Las pruebas se realizan en paralelo con el proceso de desarrollo. En la rama descendente del modelo en V se aplican pruebas estáticas sobre los entregables para detectar los errores en las primeras etapas del proceso de desarrollo de software. Una vez desarrollado el código, en la rama ascendente del modelo en V, también se pueden aplicar técnicas dinámicas. Así, se establecen las siguientes fases de pruebas: pruebas unitarias, pruebas de integración, y pruebas de verificación y validación. No olvidar que las técnicas dinámicas también incluyen pruebas de regresión para garantizar que no se han introducido nuevos errores como resultado de modificaciones, actualizaciones o corrección de fallos en las pruebas comentadas.

Estas fases de pruebas están relacionadas con las fases de la rama izquierda al mismo nivel en las que se definen las pruebas a realizar. Por ejemplo, en la fase de diseño detallado también se realiza el detalle de las pruebas unitarias que se ejecutarán en la fase de pruebas unitarias (al mismo nivel).

Las pruebas unitarias permiten comprobar el correcto funcionamiento de cada uno de los POUs programados por separado. A cada uno de ellos se le realizan dos tipos de pruebas: estructurales y funcionales. En las pruebas estructurales de la parte secuencial, se testea que la activación y desactivación de cada una de las etapas es correcta, probándose todos los posibles caminos de evolución y cerciorándose de que se recorren adecuadamente. En las pruebas funcionales de la parte combinacional se testea que las acciones asociadas a las etapas se realizan correctamente al estar éstas activadas.

Como herramienta para realizar las pruebas unitarias se proponen los sistemas de simulación que ofrecen los IDE de programación de PLCs, que permiten la activación de las entradas al software (salidas del aPS) y la visualización del resto de señales. Otra opción es emplear el HMI o pantalla de explotación como herramienta de pruebas que permite activar las entradas al SW a través de pulsadores que representan los sensores y los controles de mando. Para las pruebas unitarias estas dos opciones pueden ser suficiente, no obstante, también puede ser interesante disponer de un componente software que simule la funcionalidad de los módulos más simples que constituyen la unidad de trabajo para realizar las pruebas funcionales.

Las pruebas de integración permiten comprobar la correcta coordinación y sincronización entre los POU. Para ello, por una parte, se realizan pruebas de sincronización que permiten testear las señales de coordinación entre los distintos POU y, por otra parte, pruebas de operaciones, en las que se deberá poner la unidad de trabajo en todos los estados identificados y validar todas las operaciones involucradas en cada uno de ellos (arranque, ejecución y parada correcta de cada operación).

Como herramienta para realizar las pruebas de integración se propone el uso de componentes software cuya funcionalidad es simular el comportamiento de una composición de módulos más simples hasta el nivel de unidad de trabajo del aPS. Estos componentes software con menor grado de granularidad se crean a partir de los componentes software que simulan la funcionalidad de los módulos más simples que constituyen la unidad de trabajo.

Por último, las pruebas de verificación y validación permiten comprobar que el componente MeiA cumple con los

requisitos funcionales y no funcionales especificados, así como con las expectativas del cliente. Como herramienta para realizar estas pruebas también se propone el uso de un componente software que simule la funcionalidad de la unidad de trabajo permitiendo la inserción de fallos, muy interesante cuando se ejecuten operaciones en paralelo. También se podría utilizar un Digital Twin a modo de modelo virtual, para realizar la puesta en marcha virtual que permite probar el sistema de control en simulación como el presentado en (Iriondo et al., 2020).

3. Gemelo Funcional

En un sistema automatizado se distingue la parte de mando, que incluye el software de control, y la parte operativa, que engloba el aPS a controlar, el panel de operación y el HMI (Figura 1). El software de control recibe información de los sensores del aPS, de los controles situados en los paneles de operación y del HMI. Además, el software de control activa los accionamientos que realizan las operaciones en el aPS, así como los elementos de visualización y de aviso en el panel operación y en el HMI.

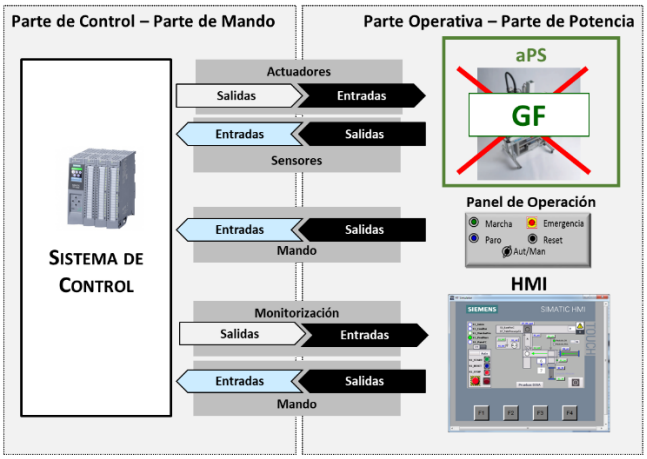

Figura 1: Estructura de un sistema automatizado

En este contexto, el GF es un componente software que simula el comportamiento del aPS, respondiendo a las órdenes del software de control (entradas al aPS) con la activación de los sensores (salidas del aPS). El GF sirve como herramienta de pruebas para los sistemas de control.

El GF está constituido por un conjunto de componentes software que simulan la funcionalidad de los módulos de control o de equipo del Modelo Físico del aPS según el estándar IEC 61512 (IEC, 2002). Ejemplos de estos componentes software, denominados CFB, se asocian a cilindros, pinzas, cintas transportadoras, elevadores, ventosas, almacenes, garras, mesas giratorias, etc.

Estos CFB se combinan para formar componentes software que simulan elementos de nivel superior en la jerarquía del modelo físico, como por ejemplo: módulos de control o módulos de equipo compuestos, unidades de trabajo y células de proceso (Figura 2).

En resumen, el GF se construye a nivel de unidad de trabajo a partir de CFB reutilizables, de acuerdo a las distintas estructuras organizativas que dependerán de las particularidades del aPS a simular. Por tanto, en las siguientes subsecciones se describen tanto el proceso de desarrollo de un CFB como el proceso para combinarlos con objeto de construir el GF a nivel de unidad de trabajo.

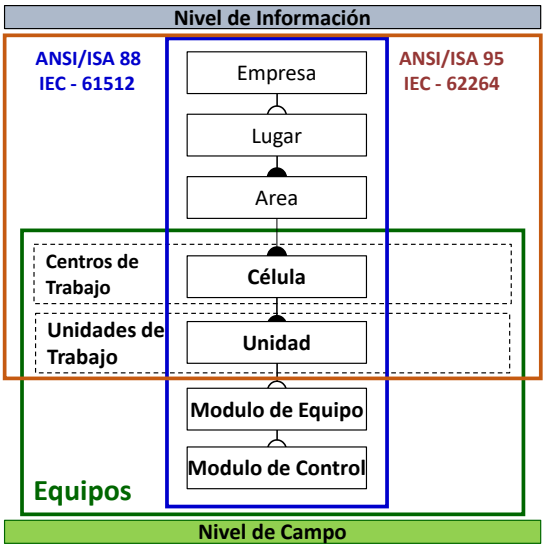

Figura 2: Modelo Físico para Sistemas Discretos

3.1. Desarrollo de un CFB

La metodología para el desarrollo de un CFB consta de los siguientes pasos:

1. Identificación de las señales de entrada y salida.
2. Análisis de la funcionalidad del módulo de control o módulo de equipo en base a las operaciones representadas en su Modelo de Proceso del estándar IEC 61512.
3. Análisis dinámico para establecer los parámetros de tiempo necesarios para simular su comportamiento. Dado que la respuesta de un sistema ante una variación en sus entradas no es instantánea, se deben establecer los tiempos de respuesta, como pueden ser: tiempos de reacción, tiempos muertos, constantes de tiempo, tiempos de pico o tiempos de establecimiento. Estos tiempos pueden ser adaptados para la realización de las pruebas.
4. Diseño detallado de la funcionalidad del CFB en lenguaje de modelado GRAFCET.
5. Análisis de fallos del módulo (opcional). Si el CFB se va a preparar para simular fallos, se analizan los posibles fallos que pueden producirse en el funcionamiento del módulo.
6. Diseño detallado de la funcionalidad con fallos del CFB (opcional). Para simular el comportamiento ante fallos, se añaden dos señales que permiten seleccionar el tipo de fallo (*CodFallo*) y activar la simulación con fallos (*ActivarFallo*).
7. Implementación del CFB siguiendo el estándar IEC 61131-3. Cada componente se implementa como un POU de tipo FB en lenguaje estructurado (ST) para que los componentes tengan el mayor grado de reutilización. La parte secuencial de los Grafcets se implementa en lenguaje ST siguiendo el método propuesto en (Burgos et al., 2020) y la parte combinacional se programa tras la parte secuencial.
8. Pruebas del CFB tanto estructurales (caja blanca) como funcionales (caja negra).
9. Documentación del CFB tomando como ejemplo la plantilla de la Tabla 1.
10. Inclusión del CFB en la librería.

Tabla 1: Plantilla ejemplo para documentar un CFB

Id CFB	Identificación del CFB
Metadatos	Datos sobre el hardware y software compatible, versión del lenguaje, actualizaciones del CFB,…
Palabras clave	Términos de búsqueda del CFB
Descripción Funcional	Descripción del funcionamiento del CFB
Modelo de Proceso	Operaciones a nivel de acciones
Módulo Físico	Módulo de Control o Módulo de Equipo simple o compuesto
Fallos	Identificación y descripción de fallos
Señales	Entradas (acciones), salidas (sensores), temporizaciones, activación fallos,…
Interface	Parametrización de las señales
Diseño Detallado	GRAFCET
HMI	Representación del elemento en el HMI
Esquemas	Eléctricos, neumáticos, etc.
Referencias a módulos comerciales	Referencias técnicas de fabricantes

A continuación, se aplica la metodología para el desarrollo del CFB de un módulo de control que consta de un cilindro de doble efecto con válvula 5/3, muelle mecánico y centro cerrado, y de dos sensores de final de recorrido (Figura 3).

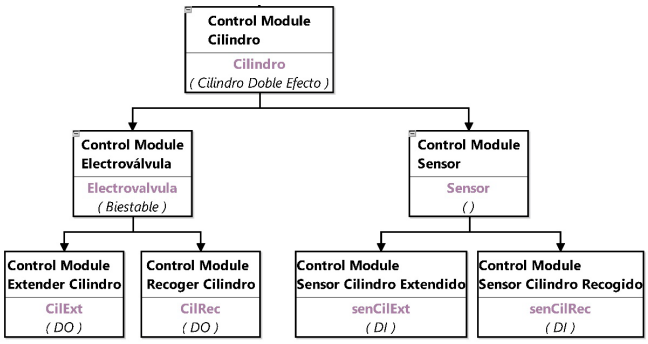

Figura 3: Modelo Físico del módulo de control del cilindro

1. *Identificación de señales de entrada y salida*

Las entradas al CFB serán las señales de Extender (*CilExt*) y Recoger (*CilRec*) asociadas a la válvula biestable. Las salidas serán las señales de los sensores que detectan si el cilindro se encuentra extendido (*senCilExt*) o recogido (*senCilRec*).

2. *Análisis de la funcionalidad del cilindro*

Cuando se activa la señal de *Extender* del cilindro y éste no se encuentra *extendido*, comienza la operación de extender. Cuando se activa la señal de *Recoger* del cilindro y éste no se encuentra *recogido*, comienza la operación de recoger. Los sensores de final de recorrido se activarán y desactivarán en función de la posición del cilindro. Si la señal de *Extender* o *Recoger* se desactiva en mitad del recorrido, el cilindro se detendrá en una posición intermedia (ninguno de los sensores de final de recorrido estará activado).

3. *Análisis dinámico del cilindro*

Para simular el comportamiento del cilindro se identifican tres parámetros de tiempo:

- el tiempo que transcurre desde que se da la orden de Extender o Recoger y se desactiva el sensor correspondiente que señala su posición de recogido o extendido (*TiempoQuitarSensor*),
- el tiempo que tarda en realizar el recorrido de avance (*TiempoExtender*) y
- el tiempo que tarda en realizar el recorrido de retroceso (*TiempoRecoger*).

4. *Diseño detallado de la funcionalidad del CFB*

En la Figura 4 se muestra el grafcet del diseño detallado de la funcionalidad incluidos los fallos.

5. *Análisis de Fallos (opcional)*

En la Tabla 2 se describen cuatro posibles situaciones de fallo en el cilindro.

Tabla 2: Situaciones de fallo del cilindro

Código fallo	**Descripción**	**Detección**
Fallo 1	Cilindro atascado o fallo del sensor de cilindro extendido	Tras la orden de Extender y transcurrido el tiempo de avance, la señal de sensor extendido no se activa
Fallo 2	Cilindro atascado o fallo del sensor de cilindro recogido	Tras la orden de Recoger y transcurrido el tiempo de retroceso, la señal de sensor recogido no se activa
Fallo 3	La electroválvula no responde a la señal de Extender	Tras la orden de Extender, el sensor de recogido no se desactiva y el sensor de extendido no se activa transcurrido el tiempo de avance
Fallo 4	La electroválvula no responde a la señal de Recoger	Tras la orden de Recoger, el sensor de extendido no se desactiva y el sensor de recogido no se activa transcurrido el tiempo de retroceso

6. *Diseño detallado de la funcionalidad con fallos del CFB s (opcional)*

En la Figura 4 se muestra el grafcet correspondiente.

7. *Implementación del componente siguiendo el estándar IEC-61131-3*

El CFB se ha implementado en un POU tipo FB.

Por último, una vez realizadas las fases 8 y 9 de pruebas y documentación, el CFB se incluye en la librería de componentes reutilizables. Estos componentes son los elementos básicos a partir de los cuales se construirá el GF.

3.2. Desarrollo de un GF

El desarrollo de un GF se realiza de acuerdo a los siguientes pasos:

1. Identificación de los módulos de equipo o de control de la unidad de trabajo en el Modelo Físico del aPS.
2. Búsqueda de los CFB, correspondientes a los módulos identificados, en la librería de componentes. En caso de no existir el CFB, se deberá desarrollar siguiendo los pasos descritos en el apartado 4.1 Desarrollo de un CFB.
3. Implementación del GF siguiendo el estándar IEC 61131-3. El GF se implementa como un POU de tipo FB en lenguaje estructurado (ST) o de contactos (LD). En la *Interfaz de Bloque* se definen las entradas y salidas del GF. Las entradas se corresponden con las salidas del software de control que activan los actuadores para ejecutar las órdenes de control. Las entradas/salidas del GF se

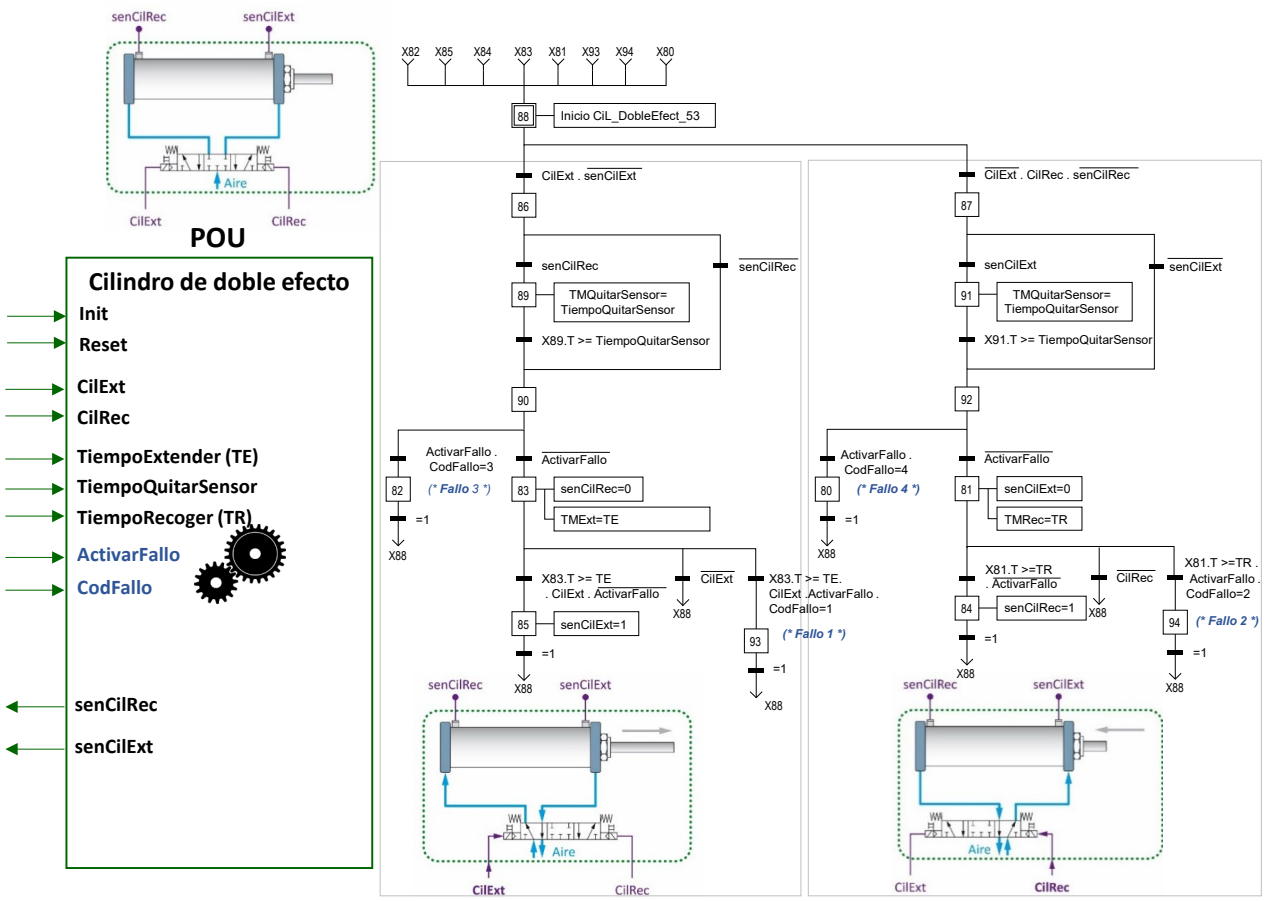

Figura 4: Diseño detallado CFB con fallos para el cilindro

corresponden con las entradas del software de control que informan de la situación de la unidad del aPS. En el *Cuerpo de Bloque* se instancian los FB de los CFB identificados y se conectan con las entradas y salidas del GF.

4. Modificación de las variables de entradas (%I) y salidas (%Q) a marcas de memoria (%M), permitiendo que las actualice el GF sin ser modificadas en cada ciclo de programa. Como alternativa, se pueden crear nuevas variables de memoria para las entradas y salidas, con objeto de preparar el software de control para conmutar entre el funcionamiento con el GF o con el aPS. En ambas opciones, el GF dispone de una señal de activación.
5. Vinculación del GF con el software de control. Una vez realizada la instancia del FB del GF, se realiza la conexión con el software de control. Las entradas del GF se conectan con las salidas generadas por el software de control para actuar sobre el aPS. Las variables de entrada/salida se conectan con las entradas simuladas del software de control.

4. Caso de estudio

El caso de estudio aborda el desarrollo de un GF para realizar las pruebas del software de control de una de las unidades de la célula de ensamblaje DISA FMS200. Concretamente, la unidad de trabajo denominada *S1 - Ubicación Base,* que realiza el proceso de colocación correcta de la base sobre el palet situado en la posición de carga de la unidad. Las bases se alimentan de un almacén vertical.

Para la realización del GF de la unidad se seguirán los pasos descritos en el apartado 3.2.

1. *Identificación de los módulos de equipo o de control de la unidad de trabajo en el Modelo Físico del aPS.*

En el modelo físico de la Figura 5 podemos observar que la unidad *S1 - Ubicación Base* consta de dos módulos de equipo: *Manipulador* y *Alimentador de Bases.* El *Manipulador* está constituido por los siguientes módulos de control: un cilindro de simple efecto, un cilindro de doble efecto (el utilizado como ejemplo para desarrollar el CFB) y una ventosa. El *Alimentador de Bases* está formado por cuatro cilindros de simple efecto y un almacén por gravedad.

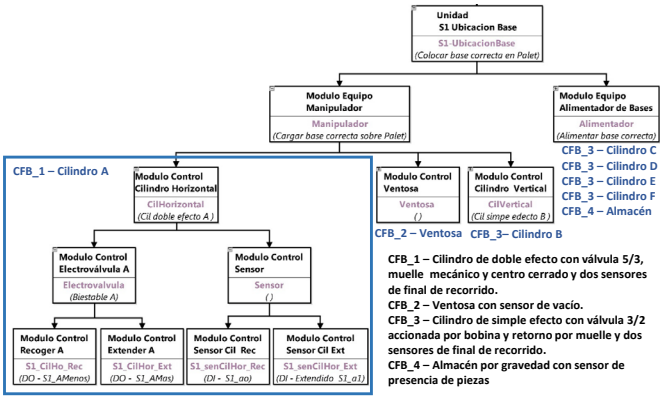

Figura 5: Modelo Físico de la Unidad S1 – Ubicación Base

2. *Búsqueda de los CFB en la librería de componentes.*

Los CBF asociados a los módulos de control identificados se encuentran en la librería de componentes. En concreto para la estación *S1 - Ubicación Base* serán necesarios cuatro tipos de CFB, tal y como se muestra en la Figura 6.

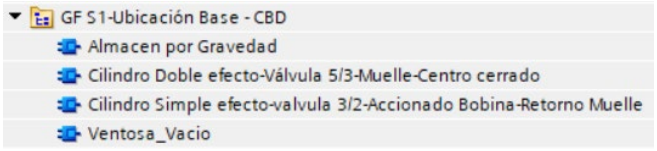

Figura 6: CFB seleccionados de la librería

3. *Implementación del GF según el estándar IEC 61131-3.*

Tras la creación del FB y la definición de la interface de bloque, en el cuerpo del FB se instancian los CFB identificados y se conectan. Este desarrollo se ha realizado en *Tia Portal* (Figura 7).

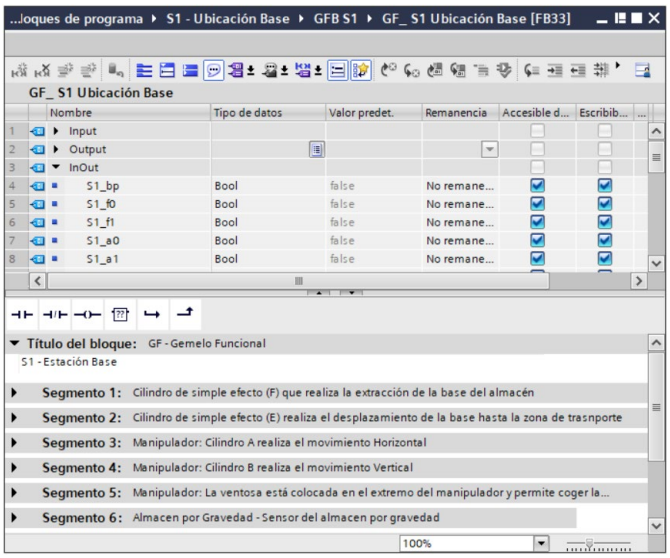

Figura 7: Implementación del FB del GF en Tia Portal

4. *Modificación de las variables de entradas y salidas a marcas de memoria y creación de la señal de activación.*
5. *Vinculación del GF con el software de control.*

En la Figura 8 se puede apreciar el GF en funcionamiento.

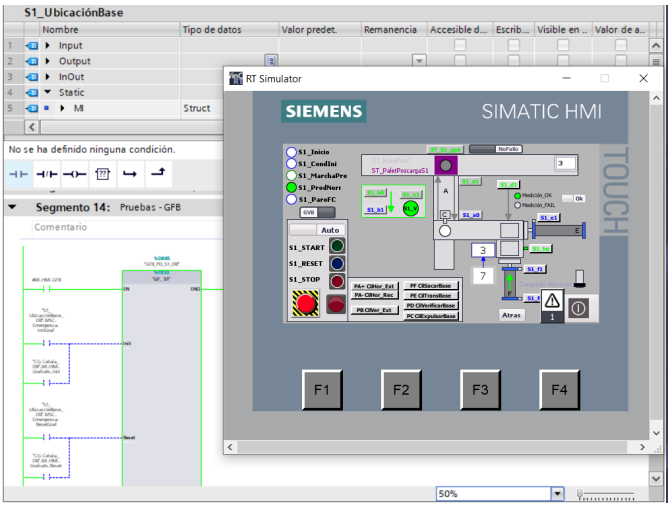

Figura 8: GF en ejecución

5. Conclusiones

La correcta realización de las pruebas del software de control de un aPS con las herramientas convencionales requiere un profundo conocimiento del comportamiento del aPS en cuestión, ya que se precisa activar y desactivar los sensores en función de las acciones realizadas por medio de los actuadores. El esfuerzo aumenta cuando el proceso requiere operaciones en paralelo.

En este sentido, el GF simula el comportamiento del aPS activando y desactivando los sensores tal y como lo haría el sistema físico. Por lo tanto, reduce los errores que se comenten al realizar una simulación manual con las herramientas convencionales que dependen, además, de la destreza del personal encargado de su ejecución.

Destacar también la reducción en el tiempo dedicado a las pruebas, la facilidad de construcción sin herramienta comerciales de desarrollo de plantas virtuales, lo cual revierte en el aspecto económico, así como la reutilización y portabilidad dado que sigue el estándar IEC 61131-3 ampliamente aceptado en el área de automatización.

Referencias

Burgos, A., Iriondo, N., Álvarez, M.L., Sarachaga, I., 2021. MeiA 4.0 para abordar los retos actuales de formación en automatización, XLII Jornadas de Automática, 1-3 Sep., Castellón, España, pp. 240-247.

Burgos, Álvarez, M.L., A., Iriondo, N., Sarachaga, I., 2020. Metodología para la transformación de diseños en GRAFCET a código IEC 61131-3, Información Tecnológica 31(6).
DOI: 10.4067/S0718-07642020000600133

ISO/IEC/IEEE 12207, 2017. Systems and software engineering – Software life cycle processes. International Organization for Standardisation.

IEC PAS 63088, 2017. Smart manufacturing – Reference architecture model industry 4.0 (RAMI 4.0).

IEC 61131-3, 2013. IEC 61131-3, Programmable Controllers, Part 3: Programming Languages.

IEC 61512-1, 2002. Batch Control-Part 1: Models and terminology.

IEC 62264, 2013. Enterprise-control system Integration-Part 1: Models and terminology.

Iriondo, N., Álvarez, M.L., Sarachaga, I., Burgos, A., 2022. Unidades de control encapsuladas para sistemas de automatización, XLIII Jornadas de Automática, 7-9 Sep., Logroño, España, pp. 892-899.

Iriondo, N., Orive, D., Casquero, O., Marcos, M., 2020. A proposal to introduce digitalization technologies within the automation learning process, IFAC-PapersOnLine, 53(2), pp. 17592-17597.
DOI: 10.1016/j.ifacol.2020.12.2674

Kritzinger, W., Karner, M., Traar, G., Henjes, J., & Sihn, W., 2018. Digital Twin in manufacturing: A categorical literature review. IFAC-PapersOnLine, 51(11), pp. 1016-1022.
DOI: 10.1016/j.ifacol.2018.08.474

Orive, D., Iriondo, N., Burgos, A., Sarachaga, I., Alvarez, M.L., Marcos, M. 2019. Fault injection in Digital Twins as a means to test the response to process faults at virtual commissioning. 24th IEEE Conference on Emerging Technologies and Factory Automation (ETFA), 1230-1234.
DOI: 10.1109/ETFA.2019.8869334