XXXVIII Jornadas de Automática
https://doi.org/10.17979/spudc.9788497497749.0913

GENERACIÓN AUTOMÁTICA DEL PROYECTO DE AUTOMATIZACIÓN TIA PORTAL PARA MÁQUINAS MODULARES

Darío Orive, Aintzane Armentia, Eneko Fernández, Marga Marcos.
Departamento de Ingeniería de Sistemas y Automática
Universidad del País Vasco / Euskal Herriko
Unibertsitatea

dario.orive@ehu.eus, aintzane.armentia@ehu.eus, eneko.fernandez.alonso@gmail.com,
marga.marcos@ehu.es

Resumen

Uno de los retos de la Industry 4.0 es disponer de sistemas de fabricación más flexibles y eficaces, que permitan fabricar productos cada vez más personalizados, adaptados a la demanda de los clientes. Esto unido a que esta flexibilidad se logre a un precio competitivo ha llevado al concepto de máquina modular. La máquina se concibe como un conjunto de componentes mecatrónicos que incluye una parte física (mecánica, eléctrica y electrónica), y el software y hardware necesario para controlarla. Estos componentes pueden fabricarse en serie. Diferentes combinaciones de componentes mecatrónicos permiten fabricar máquinas que se adaptan a las demandas del cliente.

Este artículo propone un modelo para la maquina modular a partir cual es posible generar de forma automática el proyecto TIA Portal, utilizando tecnologías XML.

Palabras Clave: *Industry 4.0, Modularity, Automation System, Automatic code generation, Flexible manufacturing systems.*

1. Introducción

Es de sobra conocida la tendencia generalizada a introducir mayores niveles de automatización en los procesos de fabricación, persiguiendo como objetivo conseguir productos de mejor calidad y precios más competitivos. Los procesos de fabricación cada día son más complejos, con tiempos de cadencia de producción cortos entre series de producción pequeñas, requiriendo además un alto nivel de flexibilidad para adaptarse a las necesidades del mercado[1].

Otro requisito demandado es el de acortar el tiempo necesario para modificar la línea de producción con el objetivo de reducir los tiempos de parada de la línea.

También se prevé que los procesos de fabricación tengan que ser más flexibles para adaptarse a fabricar lotes de productos mucho más pequeños que los actuales, y más personalizados a las necesidades de los clientes finales. Todas estas adaptaciones y cambios en los procesos de fabricación deben realizarse de forma ágil, rápida y fiable.

Estos son solo algunos de los retos de lo que se ha denominado en Europa la Industry 4.0 o lo que otros denominan como 4ª revolución industrial [2]. Son muchos los objetivos planteados bajo este paradigma, tanto de tipo tecnológico como de tipo económico. Sin embargo, todos ellos se caracterizan por la introducción de modernas tecnologías como elemento habilitador de la competitividad. Entre sus objetivos destacan la digitalización en todas las fases del ciclo de vida del producto, así como la extracción de conocimiento adquiriendo y procesando información a lo largo del ciclo de vida. La acción combinada de ambas es la base para construir sistemas de fabricación más flexibles y eficientes que los actuales.

El diseño modular de sistemas de fabricación [3] es una idea que se ha propuesto en diferentes trabajos. Así, [4] focaliza en aspectos de la granularidad de los módulos para conseguir altos grados de reutilización con mínimo esfuerzo. En [5] se propone la definición y diseño de un sistema de automatización capaz de asegurar su disponibilidad mediante técnicas y tecnologías de modelado y tecnología multi-agente. [6] estudia la problemática del control de versiones de software. En [7] se analiza aspectos de cómo generar el programa de control de sistemas

modulares, en un entorno multidisciplinar de tres formas diferentes; basada en librería central, en base a parámetros y basada en plantillas. En estos trabajos, la modularidad se utiliza para facilitar el diseño del sistema de control. Pero en prácticamente todos ellos, bien a través del modelado, bien a través de la codificación, existen procesos manuales que son propensos a error. Sólo en [8] se propone una generación automática de código modular. Ahora bien, los módulos físicos se definen de forma abstracta y es el usuario el que debe modelar la máquina, definiendo todos sus componentes.

Esta es precisamente la aportación de este trabajo: a través de la definición de la máquina modular en base a componentes mecatrónicos reutilizables, permite automatizar la definición de nuevas máquinas, asegurando la corrección de la definición y generando el proyecto de automatización (hardware y software) de la máquina completa. De esta forma se eliminan tareas repetitivas (como cambio de nombres de variables o asignación de direcciones de entrada/salida) y propensas a error. Concretamente se realiza para Controladores Lógicos Programables desde la herramienta TIA Portal de Siemens utilizando el API TIA Portal Openness que facilita la manipulación automática de proyectos completos.

La estructura del artículo es la siguiente: En el apartado 2 se propone el modelado de la máquina modular desde el punto de vista funcional y de control (hardware-software), cuya implementación en casos concretos plasmará el grado de granularidad adecuado. Se ilustra con una propuesta concreta de máquina que es la que se utiliza en el resto de apartados. El apartado 3 se dedica a la generación automática del proyecto TIA Portal. Se presenta la arquitectura general de la aplicación de generación así como sus módulos principales. En el apartado 4 se presenta el caso de estudio de una máquina de ensayos de fugas. El artículo finaliza con el apartado de conclusiones y trabajos futuros.

2. Modelado modular de máquinas.

El modelado de la máquina modular que se propone se basa en dos vistas: La funcional, en la que la máquina está definida por un conjunto de módulos, de forma que un módulo puede estar a su vez formado por otros formando una jerarquía, y la de implementación que define el hardware y el software de control de la máquina.

A continuación se presentan ambas vistas y se ilustran para un caso concreto de jerarquía.

2.1 Vista funcional.

La vista funcional define los componentes que pueden conformar una máquina. Dado que esta vista define componentes físicos, se propone una estructura jerárquica que permite al diseñador establecer la granularidad deseada para definir los módulos físicos. De esta forma, un módulo funcional puede estar formado por otros módulos y dispositivos de entrada/salida que representan el conjunto de señales de los sensores y actuadores que se utilizan para controlar ese componente. El último nivel de la jerarquía de cada rama únicamente contiene dispositivos de entrada/salida. El meta-modelo de esta vista se ilustra en la Figura 1 mediante un diagrama de clases UML.

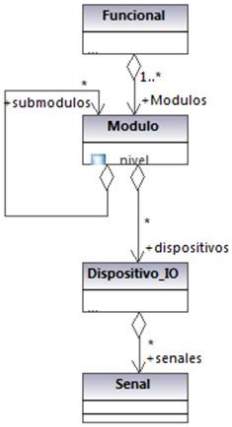

Figura 1 Meta-Modelo de la Vista Funcional

El meta-modelo se completa con dos reglas de composición: un módulo sólo puede contener módulos de un nivel inferior y los módulos de último nivel de la jerarquía sólo pueden contener dispositivos.

Para ilustrar el uso de esta vista, supongamos una ingeniería que diseña y monta máquinas que están formadas por un conjunto de bastidores sobre los que se ubican estaciones que pueden realizar operaciones de fabricación. Además, se contempla la existencia de módulos botonera (para el mando de la máquina, transporte (módulo opcional que puede existir para transportar la pieza entre máquinas), HMI y armario (que agrupa el resto de señales necesarias para el buen funcionamiento de la máquina). Siguiendo el meta-modelo de la Figura 1, es posible definir el meta-modelo de este tipo de máquinas definiendo los módulos básicos y sus relaciones. El meta-modelo resultante se ilustra en la Figura 2.

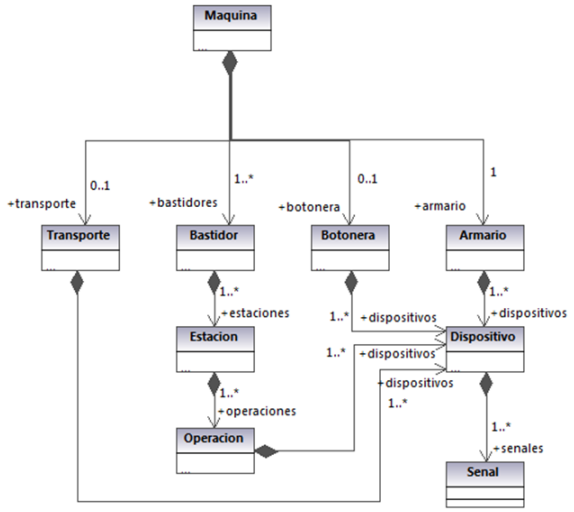

Figura 2 Meta-Modelo Funcional de un tipo de máquina

2.2 Vista de Implementación

La vista de implementación sigue el modelo de proyectos de automatización de TIA Portal (Siemens) que define los elementos del meta-modelo de la Figura 3.

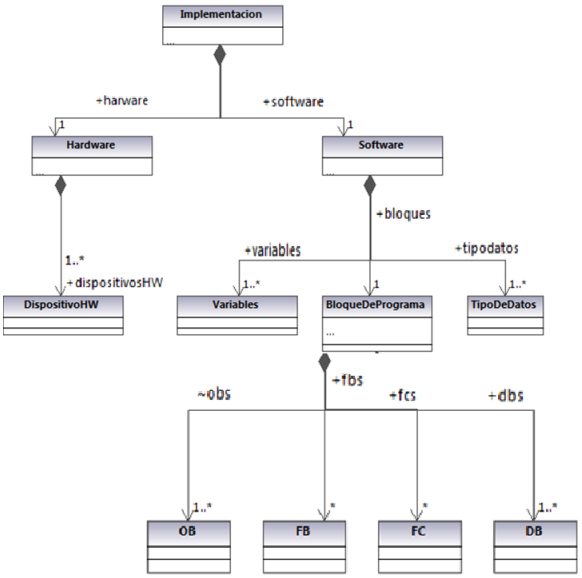

Figura 3 Meta-Modelo de la implementación

Para ilustrar la vista de implementación, la Figura 4 representa las propiedades de dispositivoHW. Como se puede observar, en este caso se ha caracterizado un dispositivo de una red Profinet. Cada dispositivo se caracteriza por un nombre, el nombre Profinet y la dirección IP (necesarios para configurar la red) así como la dirección inicial de señales de E/S.

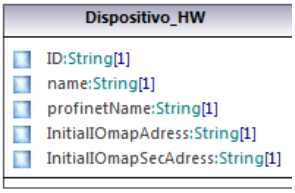

Figura 4 Dispositivo_HW

La máquina modular está formada por las dos vistas y las relaciones entre ellas. Concretamente, existe una relación entre los dispositivos hardware de la vista de implementación y el dispositivo IO de la vista funcional. Por otro lado, cada señal de la vista funcional corresponde a una variable y una dirección física en la tabla de variables. Por otro lado, los módulos de la vista funcional se mapean a boques de programa.

3. Generación automática del proyecto TIA Portal

La Figura 5 representa la arquitectura de alto nivel de la aplicación de generación.

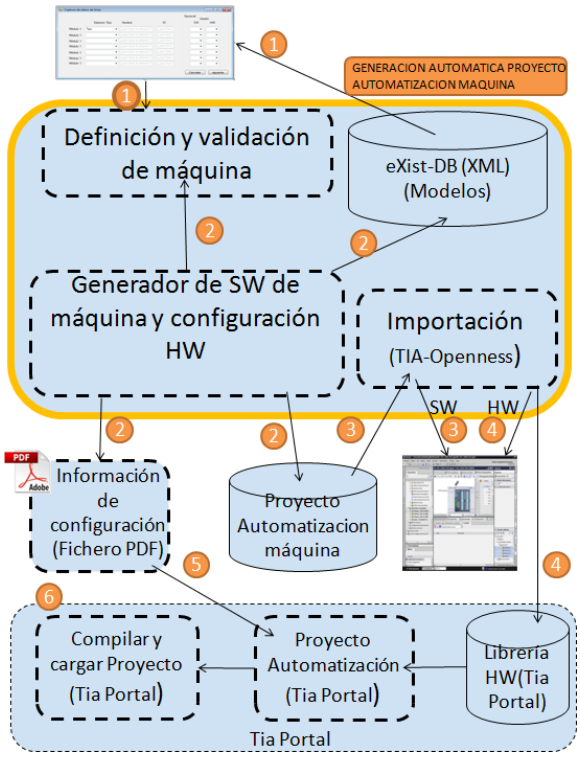

Figura 5 Arquitectura general

La aplicación de generación de un proyecto de automatización está dividida en una serie de fases: se comienza definiendo una nueva máquina que está compuesta por un conjunto de módulos con diferentes funcionalidades. Por cada módulo que compone la máquina (y que puede contener una jerarquía de módulos) se dispone de un proyecto TIA

Portal que corresponde a su modelo de implementación. La aplicación debe resolver etiquetas de código, variables y direcciones Profinet duplicadas y generar el código final.

Se ha definido una base de datos orientada a modelos que contiene todos los proyectos de automatización tipo correspondiente a módulos funcionales, así como las nuevas máquinas definidas.

La base de datos elegida ha sido eXist-db [9], que es del tipo NoSQl, y XML nativa lo que la hace adecuada para el manejo y almacenamiento de modelos en formato XML, que es el formato en el que se almacena la información de los modelos.

Una de las características de esta base de datos es que no se necesita especificar un XML schema de la información que se almacena, lo cual da mucha flexibilidad para introducir diferentes tipos de documentos, y para futuras ampliaciones. Esto no significa que no se puedan hacer validación de los documentos XML que se introduzcan.

eXist-DB es independiente de la plataforma, dado que está basada en java, al igual que su API. Esta API permite el uso de lenguajes query como Xpaht o Xquery. También permite la administración de usuarios, permisos, indizado…

La administración de los datos se consigue por medio de colecciones de documentos jerárquica, por los que se pueden realizar las acciones sobre conjuntos de documentos. Para actualizar o borrar documentos se puede hacer uso del lenguaje XUpdate, que nos permite actualizar el documento entero o la elección de los nodos a actualizar. También provee de mecanismos de copias de seguridad y restauración (Backup/restore). El motor de la base de datos tiene funciones básicas de seguridad, como pude ser el control de acceso mediante contraseña de los usuarios a los grupos que pertenezcan.

La elección de una base de datos XML nativa se debe a que la herramienta TIA Portal ofrece TIA Portal Openness, un API que permite acceder a la estructura de la herramienta TIA Portal, de forma remota y modificar el contenido de la misma. De esta forma se pueden generar proyectos de automatización utilizando objetos almacenados en librerías u objetos externos importados en XML.

También, se puede acceder a los datos de proyectos para su posterior procesamiento, extraer datos estadísticos, realizar copias de seguridad o actualizar el contenido de los proyectos.

Entre las funcionalidades ofrecidas por TIA Portal Openness, caben destacar las siguientes:

- *Manipulación* de ciertos objetos de un proyecto, p.e. (carpetas, hardware, bloques de programa, variables, tipos de variables, …), tanto del PLC, como del HMI.
- *Utilización* de librerías, tanto del proyecto, como librerías globales.
- *Exportación* e *importación* de información relativa al software de proyectos.
- *Ejecución* de comandos relacionados con tareas del TIA Portal, como por ejemplo compilar un proyecto ya generado y cargar el proyecto en el PLC.

Por otro lado, un proyecto de automatización TIA Portal contiene toda la información relacionada con todos los controladores del proyecto. Por cada controlador se genera una estructura de directorios, tal y como se ilustra en la Figura 6. En ella se puede observar que la estructura de carpetas relativas al software coincide con la del meta-modelo de la Figura 3: la carpeta de “bloques de programa”, en la cual se organizan todos los bloques de programa del PLC (OB, FB, FC, DB), la carpeta “variables PLC”, donde están declaradas las variables globales del programa del PLC y la carpeta de “tipos de dato de PLC”, en la que se pueden definir tipos de dato de usuario específicos para la aplicación. Estas carpetas se generan automáticamente vacías en TIA Portal cuando se inserta un PLC y se va introduciendo el contenido a medida que se va desarrollando el proyecto.

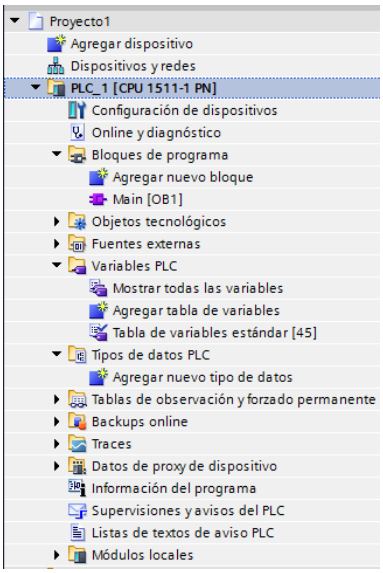

Figura 6 Árbol de directorios de PLC en TIA Portal

A continuación se describen los pasos que ejecuta la aplicación de generación:

3.1 Captura de datos de definición de la máquina

Se ha desarrollado una aplicación que permite, que el operador defina las características de la maquina. Dispone de un interfaz de usuario que accede a la

base de datos de la que se extraen los proyectos tipo definidos, que pueden utilizarse en la composición de la máquina. Esto corresponde al paso 1 de la Figura 5.

Para los tipos de máquinas definidos en la Figura 2 la aplicación ofrece los componentes mecatrónicos que son los de primer nivel: Bastidor, Transporte, Botonera y Armario.

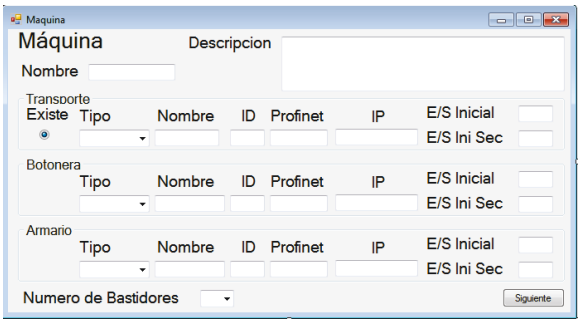

Figura 7 Interfaz de definición de la máquina.

Base de datos orientada a modelos

Una vez definida la máquina a partir de módulos funcionales, la aplicación accede a la base de datos orientada a modelos que contiene los modelos de implementación que siguen la estructura TIA Portal correspondientes a los módulos funcionales tipo, paso 2 de la Figura 5.

La Figura 8 representa las colecciones de la base datos para el caso de la máquina tipo definida en la sección 2. Contiene los modelos de implementación (bastidor, transporte, botonera, armario y HMI) de los que se van a componer las máquinas. Cada modelo representa la información relativa a un proyecto de automatización con toda la estructura hardware y software, siguiendo el meta-modelo de la Figura 3. La estructura sigue el Schema XML de la Figura 9. Además, los proyectos de automatización de las nuevas máquinas definidas, se almacenan como proyectos máquinas tipo para su posterior reutilización.

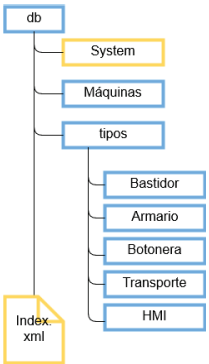

Figura 8 Estructura de la base de datos de modelos de implementación

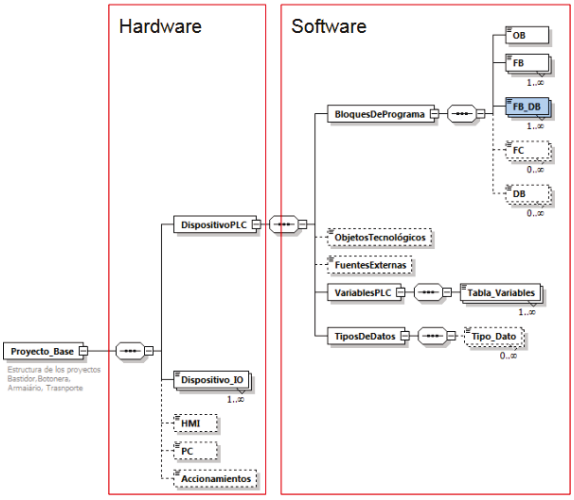

Figura 9 Schema de proyecto tipo

3.2 Generación del Proyecto Máquina

El proyecto Máquina se genera a partir de los proyectos tipo de los módulos que la componen.

Esta parte de la aplicación se encarga de adaptar el código y los datos de cada proyecto tipo, para asegurar que las instanciaciones en el código generado sean correctas, paso 2 de la Figura 5.

Esta operación se realiza en las siguientes fases:

- Fase 1. Se dispone del archivo de definición de la máquina en formato XML, generado en el paso 1. La aplicación extrae de él la información necesaria para seleccionar de la base de datos los proyectos tipo con los que se va a construir la máquina.
- Fase 2. Adaptación de las variables globales (Tipos de datos, DB globales, Tablas de variables) de aquéllos módulos tipo con más de una instancia. A modo de ejemplo, la Figura 10 representa las modificaciones que habría que realizar en una tabla de variables en cada instancia del mismo módulo.
- Fase 3. Modificación de los nombres de los parámetros actuales que se utilizan en las instanciaciones de los módulos de programa, haciendo uso de las variables declaradas en las tablas de variables.
- Fase 4. Generación del modelo de implementación del proyecto de la máquina integrando en los OBs del mismo tipo de ejecución, el contenido de los OBs de cada módulo tipo. En el modelo final solo existe un OB de cada tipo de ejecución que integrará el código de todos los proyectos tipo. En la Figura 11 se indica la forma de proceder para la generación de los OBs.

Figura 10 Adaptación de la tabla de variables

- Fase 5. Generación de un documento PDF con la documentación necesaria para la configuración del hardware y del sistema de comunicaciones de la maquina.

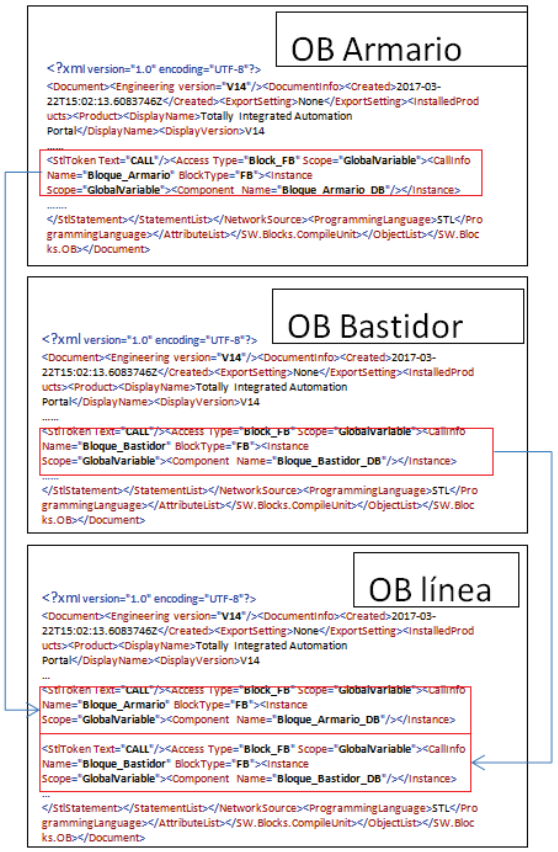

Figura 11 Generación del OB1 del proyecto máquina

3.3 Generación del proyecto en TIA Portal

El modelo de implementación de la máquina generado siguiendo el procedimiento descrito en el apartado 3.3 se importa a la herramienta TIA Portal utilizando las funciones del API TIA Portal Openness, paso 3 de la Figura 5.

La importación se realiza en dos fases:

- Fase 1. Utiliza la función de grupos para crear bloques, variables y tipos en una estructura de carpetas. La estructura de carpetas del proyecto generado en TIA Portal sigue la estructura del modelo funcional. Ver Figura 13.

- Fase 2. Importa el software del proyecto generado descrito en el apartado 3.3 a las diferentes carpetas del proyecto.

El hardware relacionado con los proyectos tipo se encuentra almacenado en la librería global de TIA Portal. Para completar el proyecto de automatización que se está generando, es necesario importar en el mismo el hardware de cada componente mecatrónico, paso 4 de la Figura 5.

TIA Portal Openness dispone de funciones que permite importar hardware almacenado en librerías TIA Portal al proyecto de automatización generado.

El operador configurará el hardware y las comunicaciones de forma manual en el entorno TIA Portal. La información necesaria para realizar esta configuración la genera la aplicación, a partir del proyecto generado, en un fichero PDF. Corresponde al paso 5 de la Figura 5.

Una vez completados los pasos anteriormente indicados se tendrá el proyecto completo generado en TIA Portal. Quedaría por último la compilación del mismo y la carga en el PLC, paso 6 de la Figura 5.

4. Caso de estudio

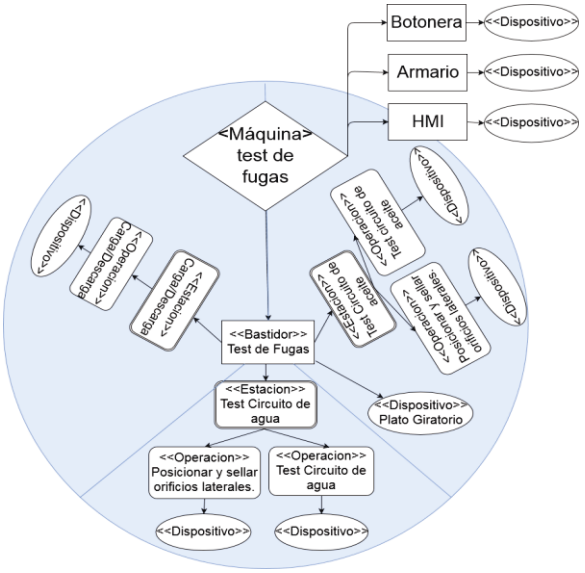

Figura 12 Diagrama funcional caso de estudio "Test de fugas"

El caso de estudio se ha desarrollado para la generación del proyecto de automatización de una máquina de testeo de fugas, utilizada en la fabricación de bloques y culatas de motores de combustión en el sector del automóvil, con la estructura de la Figura 12.

Aunque este tipo de máquinas pueden tener diferente formas constructivas, en muchos casos la máquina está formada por un plato giratorio, con varias posiciones (3 o 4), en las que se realizan las operaciones sobre las piezas. Estas máquinas pueden estar varias juntas formando una línea de procesando de piezas, iguales o diferentes, en un sistema de producción general, o pueden estar insertadas en una línea de producción con otras máquinas que realizan otras operaciones sobre el producto que se está fabricando. La máquina consta de los elementos componentes mecatrónicos siguientes.

Plato divisor (Bastidor): Es un plato giratorio de 3 posiciones desfasadas 120º, accionado por un servo-accionamiento eléctrico en posición, que dispone de señales E/S analógicas y digitales y tiene un FB asociado de control. En una posición se realiza la carga y descarga de las piezas, en la segunda posición se realiza el test del circuito de agua y en la tercera posición se realiza el test del circuito de aceite.

Cilindro hidráulico (Operación): El cilindro hidráulico eleva la pieza hasta la posición donde se encuentra un bloque mecánico, sobre el que se encuentran los sensores y actuadores del "sistema de inyección de aire y registro de valores". Tiene un dispositivo IO con un FB para su control, por cada pieza y por cada estación de la máquina.

Sistema de inyección de aire y registro (Operación): Este sistema consta de sensores de medida y actuadores y dispone de un sistema de control propio que realiza el control del testeo sobre la pieza. Dispone de un FB por cada tipo de pieza y por estación. La comunicación con el PLC es en Profinet

Transporte: la maquina utilizada como caso de estudio no tiene transporte.

Sistema de alimentación: La alimentación de la máquina se hace de forma manual. Las señales están conectadas a un dispositivo I/O y se procesan mediante un FB.

Interfaz de operador (HMI): La máquina dispone de un interfaz de operador HMI desde donde se puede dar órdenes al sistema de control.

Botonera (Botonera): La máquina dispone de un conjunto de pulsadores y dispositivos de puesta en marcha y paros de la máquina. Las señales implicadas se controlan mediante un FB de control de señalizaciones y se capturan y transmiten a través de un dispositivo I/O.

La Figura 13 presenta la estructura del proyecto generado, que como se puede observar sigue la estructura del modelo funcional de la Figura 3.

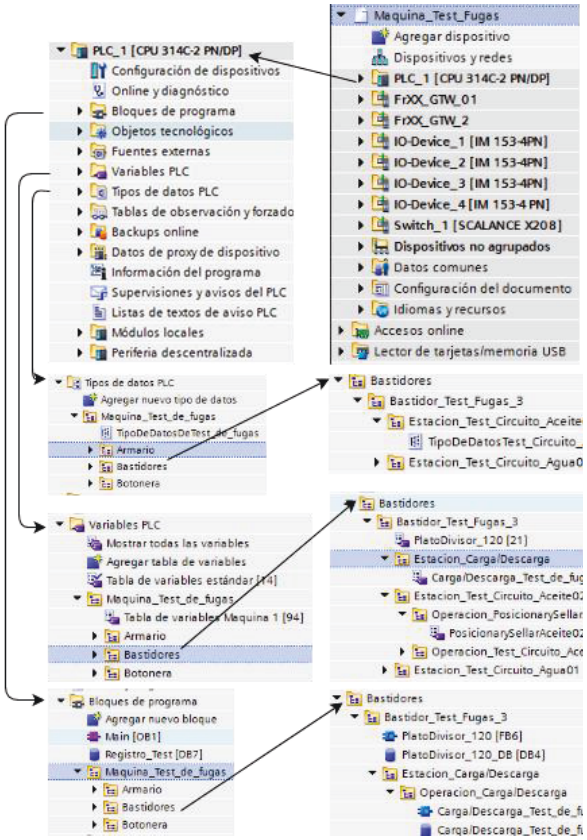

Figura 13 Resultado generación de código en TIA Portal

5. Conclusiones y trabajos futuros

La arquitectura desarrollada y descrita en este articulo permite generar proyectos de automatización de maquinas modulares a partir de proyectos tipo de forma automática, evitando todos los inconvenientes que presenta hacerlo de forma manual. Con lo que se consigue un mayor grado de de reutilización de los desarrollos realizados.

Para poder utilizar esta arquitectura es necesario un diseño modular de todos los proyectos tipo de la máquina desde su concepción.

Es una arquitectura abierta sobre la que siempre se pondrán añadir proyectos tipo básicos que se pueden ir desarrollando para futuras máquinas. Este desarrollo se realiza una única vez y se puede utilizar en muchas máquinas.

La arquitectura desarrollada es fuertemente dependiente de:

- Plataforma en la que se va a generar el proyecto de automatización en este caso TIA Portal
- El meta-modelo de las máquinas que construye cada empresa.

En un principio se ha definido una arquitectura que contempla el nivel máquina, pero esta arquitectura podría ampliarse añadiendo mas niveles, como podría ser la línea completa de un proceso de fabricación.

Agradecimientos
Este trabajo se ha financiado en parte bajo el Proyecto DPI2015-68602-R (MINECO/FEDER, UE), por la Universidad del País Vasco (UPV/EHU) con el proyecto PPG17/56, y por el Gobierno Vasco (GV/EJ) bajo el reconocido grupo de investigación IT914-16.

Referencias

[1] B. Prasad, "Analysis of pricing strategies for new product introduction," *Pricing Strateg. Pract.*, vol. 5, no. 4, pp. 132–141, 1997.

[2] Plattform Industrie 4.0, "What is Industrie 4.0?," pp. 4–5, 2017.

[3] K. Ulrich, "Fundamentals of product modularity," in *Management of Design*, Springer, 1994, pp. 219–231.

[4] C. Maga, N. Jazdi, and P. Göhner, "Reusable models in industrial automation: experiences in defining appropriate levels of granularity," *IFAC Proc. Vol.*, vol. 44, no. 1, pp. 9145–9150, 2011.

[5] R. Priego, A. Armentia, E. Estévez, D. Orive, N. Iriondo, and M. Marcos, "Implementación de mde para la generación de sistemas de automatización reconfigurables," *XXXVI Jornadas de Automática*, pp. 166–173, 2015.

[6] B. Vogel-Heuser, A. Fay, I. Schaefer, and M. Tichy, "Evolution of software in automated production systems: Challenges and research directions," *J. Syst. Softw.*, vol. 110, pp. 54–84, 2015.

[7] S. Feldmann and B. Vogel-Heuser, "Interdisciplinary product lines to support the engineering in the machine manufacturing domain," *Int. J. Prod. Res.*, vol. 55, no. 13, pp. 3701–3714, Jul. 2017.

[8] E. Estévez, M. Marcos, and D. Orive, "Automatic generation of PLC automation projects from component-based models," *Int. J. Adv. Manuf. Technol.*, vol. 35, no. 5–6, pp. 527–540, 2007.

[9] E. Siegel and A. Retter, *eXist A NoSQL Document Database and Application Platform*. O'Reilly Media, 2014.