CEA Comité Español de Automática
XLVII JORNADAS DE AUTOMÁTICA CÓRDOBA 2026
UNIVERSIDADE DA CORUÑA

Jornadas de Automática

Generando componentes MeiA a partir de sus modelos de información

Ekaitz Hurtado, María Luz Álvarez*, Nagore Iriondo, Aintzane Armentia,
Arantzazu Burgos, Isabel Sarachaga

Departamento de Ingeniería de Sistemas y Automática, Escuela de Ingeniería de Bilbao,
Universidad del País Vasco / Euskal Herriko Unibertsitatea UPV/EHU

To cite this article: Hurtado, E., Álvarez, M.L.*, Iriondo, N., Armentia, A., Burgos, A., Sarachaga, I. 2026. Generating MeiA components from their information models. Jornadas de Automática, 47. https://doi.org/10.17979/ja-cea.2026.47.13656

Resumen

La transformación digital en la Industria 4.0 exige integrar eficazmente las tecnologías de la información (IT) y operativas (OT), un reto clave debido a sus diferencias y a la baja tasa de éxito de proyectos actuales. Para abordar esta problemática, se propone el uso de componentes MeiA en sistemas de producción automatizados, que estructuran el software de control mediante dos modelos: el modelo de integración (MI) y el modelo del sistema de control (MSC). Estos modelos, basados en estándares IEC 61512 e IEC 62264, facilitan la interoperabilidad IT-OT y la conexión entre niveles de planta y gestión empresarial. El trabajo presenta una herramienta que permite modelar, validar y transformar estos modelos en el esqueleto de un proyecto de automatización, utilizando tecnologías abiertas como Sirius, XML y Java. El proceso automatiza tareas complejas, mejora la calidad del software y reduce errores y tiempos de desarrollo, generando sistemas modulares, mantenibles y adaptables a futuros cambios industriales.

Palabras clave: Ingeniería Dirigida por Modelos, Sistemas de Fabricación Inteligentes, Control de planta de fabricación

Generating MeiA components from their information models

Abstract

Digital transformation in Industry 4.0 requires effective integration of Information Technology (IT) and Operational Technology (OT), a key challenge due to their differences and the low success rate of current projects. To address this issue, the use of MeiA components in automated production systems is proposed. These components structure control software through two models: the Integration Model (MI) and the Control System Model (MSC). Based on IEC 61512 and IEC 62264 standards, these models enable IT-OT interoperability and facilitate connectivity between plant-level operations and enterprise management systems. This work presents a tool that allows modeling, validation, and transformation of these models into the skeleton of an automation project, using open technologies such as Sirius, XML, and Java. The process automates complex tasks, improves software quality, and reduces errors and development time, resulting in modular, maintainable systems that can adapt to future industrial changes.

Keywords: Model Driven Engineering, Intelligent Manufacturing Systems, Manufacturing plant control.

1. Introducción

Los retos actuales de la fabricación precisan una fuerte inversión en transformación digital para abordar la Industria 4.0 (I4.0). Sin embargo, la tasa de éxito de los proyectos desarrollados es inferior a la deseada (Cochran et al., 2022), siendo la escasa integración entre las tecnologías de la información (TI) y las tecnologías operativas (TO) uno de los puntos débiles a abordar (Vogel-Heuser, et al., 2021). Estos dominios tienen características esencialmente diferentes con requisitos contradictorios para las partes interesadas, así que dicha integración tiene un gran impacto en la interoperabilidad entre el nivel de planta y los sistemas de nivel estratégico empresarial (Behrendt et al., 2023).

Muchos trabajos se orientan hacia los niveles superiores, mientras que en menor medida abordan la incorporación de

*Autor para correspondencia: marialuz.alvarez@ehu.eus

los avances en OT en el desarrollo de los Sistemas de Producción Automatizados (aPS) (Díaz *et al*., 2020). No obstante, es precisamente dicha incorporación la que proporciona sistemas modulares con modelos de información bien estructurados para el nivel de planta que permiten conectar con éxito datos y procesos desde la planta de fabricación hasta los sistemas de nivel estratégico. En este sentido, la metodología MeiA (Sarachaga et al., 2026) concibe el software de control de un aPS como un conjunto de componentes que interoperan para alcanzar los objetivos de producción. La arquitectura de un componente MeiA proporciona dos modelos de información: el Modelo de Integración (MI) que organiza el estado de la unidad de trabajo y sus conexiones con el entorno, y el Modelo del Sistema de Control (MCS) que contiene el estado de su software de control. Estos modelos de información facilitan la interoperabilidad entre componentes y la integración con los niveles estratégicos de la empresa, garantizando un acceso restringido y seguro.

Por otra parte, la normalización es un aspecto clave para garantizar la interoperabilidad entre los niveles jerárquicos de una empresa según RAMI 4.0 (IEC, 2017), que adopta las normas IEC 61512 (IEC, 2002) e IEC 62264 (IEC, 2013). La primera proporciona un conjunto de modelos que promueven un enfoque de diseño modular para los sistemas de control, mientras la segunda establece modelos de datos e interfaces normalizados para integrar las soluciones de nivel empresarial y los sistemas de control modulares. Por ello, los modelos de información de un componente MeiA se construyen a partir de modelos IEC 61512, facilitando la integración IT-OT con aplicaciones empresariales según IEC 62264.

En este contexto, con objeto de gestionar la complejidad creciente que exige la automatización industrial moderna y reducir la dependencia de conocimientos técnicos específicos, se precisan herramientas que automaticen la implementación de estos componentes MeiA a cargo del control de las unidades de trabajo y también del control de las células de planta que conectarán con los sistemas de nivel estratégico de la empresa. Así, en este trabajo se propone una herramienta que aborda el modelado de los modelos de información y su validación como paso previo a la generación automática del esqueleto del proyecto de automatización que implementa un componente MeiA.

El artículo se estructura en una introducción que presenta el problema de la integración IT-OT y la propuesta basada en componentes MeiA. A continuación, se describe el escenario de desarrollo y las herramientas utilizadas para modelar, validar y transformar los modelos de información. Después, se detallan los modelos MI y MSC y las herramientas asociadas, seguidas de la explicación del proceso de validación y generación automática del proyecto. Finalmente, se incluye un ejemplo de aplicación y se concluye destacando las ventajas de la solución propuesta.

2. Escenario de desarrollo

La generación del esqueleto del proyecto de automatización que implementa un componente MeiA de tres pasos:

1. Generación de los modelos MI y MSC definidos en las fases de análisis y diseño del componente.
2. Validación de dichos modelos y su preparación para una posterior transformación.
3. Transformación de los modelos para generar el esqueleto del proyecto de automatización.

En la Figura 1 se muestra el escenario con las herramientas desarrolladas para realizar cada paso: dos editores gráficos para modelado, uno para cada modelo de información, y la aplicación MeiA-GENXX que integra las herramientas de validación y transformación.

En la selección de la tecnología para generar los lenguajes específicos de dominio (DSL) de los modelos y los editores gráficos se han evaluado seis herramientas considerando siete criterios técnicos y prácticos: el tipo de herramienta (abierta o comercial), el lenguaje de meta-modelado, el entorno de trabajo (estándar o propietario), la complejidad de uso, la accesibilidad al código, los formatos de almacenamiento y la facilidad de manipulación de los modelos.

Como resultado del análisis, se ha seleccionado Sirius, ya que se trata de una herramienta de código abierto, desarrollada sobre la plataforma Eclipse y construida a partir de EMF (Eclipse Modeling Framework) y GMF (Graphical Modeling Framework) (Wegert and Shatalin, 2008. Sirius permite definir lenguajes específicos de dominio y generar editores gráficos sin necesidad de poseer un conocimiento profundo de las tecnologías subyacentes. Además, utiliza el lenguaje Java para generar el código del editor a partir de los modelos definidos. El editor gráfico almacena automáticamente los modelos en formato XMI (XML Metadata Interchange) durante la definición gráfica del mismo (Zisman, 2000), proporcionando las herramientas necesarias para editar, modificar y validar los modelos de información de forma eficiente.

En la implementación de la solución se ha empleado el entorno Obeo Designer (Obeo, 2026), que integra Sirius y proporciona herramientas avanzadas para la definición y despliegue de editores gráficos personalizados, facilitando así el desarrollo de la herramienta de modelado propuesta.

La adopción del entorno Sirius conlleva la utilización de XML (eXtensible Markup Language) (W3C, 2008) y Java como tecnologías base en las etapas de validación y transformación de modelos, a partir de las cuales se genera el esqueleto del proyecto de automatización. XML es un lenguaje de marcado basado en texto orientado a la representación estructurada e interoperable de datos, ampliamente utilizado para el intercambio de información entre sistemas heterogéneos gracias a su independencia de plataforma. En su ecosistema, XML Schema (XSD) permite definir la estructura y restricciones de los documentos (W3C, 2012), XPath facilita la navegación y selección de nodos (W3C, 2017a), y XSLT posibilita la transformación de documentos XML hacia otros formatos, soportando procesos Model-to-Model (M2M) y Model-to-Text (M2T) (W3C, 2017b). Asimismo, APIs como DOM y SAX ofrecen distintos

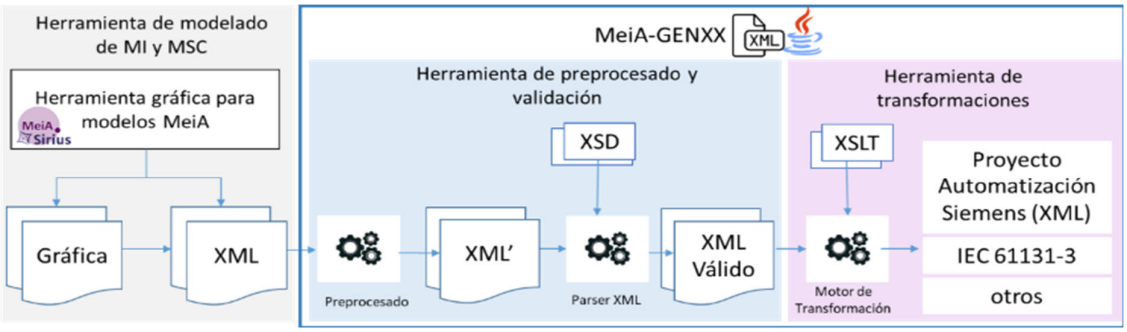

Figura 1: Escenario de desarrollo

enfoques para el procesamiento de XML, mientras que en entornos Java estas capacidades se integran mediante JAXP (Java API for XML Processing), que incorpora motores de validación y transformación (Wu et al., 2019). En el contexto de Eclipse, este conjunto de tecnologías se utiliza como base para la gestión, validación y transformación de modelos, proporcionando un motor flexible y estandarizado para el tratamiento de información estructurada dentro de herramientas de modelado.

En conjunto, la selección de estas tecnologías responde a criterios de apertura, madurez tecnológica, facilidad de uso y capacidad de integración, posicionándose como una solución adecuada para el desarrollo de herramientas de modelado en el contexto planteado.

En los siguientes subapartados se presentan los modelos de información del componente MeiA y las etapas de desarrollo de las herramientas que permiten automatizar la generación del esqueleto del proyecto de automatización.

2.1. Modelos de información del componente MeiA

Un componente MeiA encapsula el software de control de una entidad de un aPS, ofreciendo su funcionalidad a través de una interfaz que oculta todos los aspectos de implementación y le permite operar con otros componentes (Iriondo et al., 2022).

La arquitectura del componente organiza los datos en dos modelos de información manipulados por los procedimientos que realizan las operaciones de control.

En la Figura 2 se muestra el metamodelo del modelo MI, formado por el modelo físico y por un conjunto de módulos conectados. El modelo físico está organizado de manera jerárquica, según el estándar IEC 61512, en unidades, módulos de control y módulos de equipo, con las señales asociadas a las entradas y salidas del sistema de control.

Por otro lado, los módulos conectados representan las conexiones del componente con el entorno y contienen las señales o los grupos de señales con las peticiones de servicio y los comandos que pueden solicitar al componente.

Todas las señales se caracterizan por su tipo y por su rol dentro del sistema (entrada, salida, petición de servicio y comando).

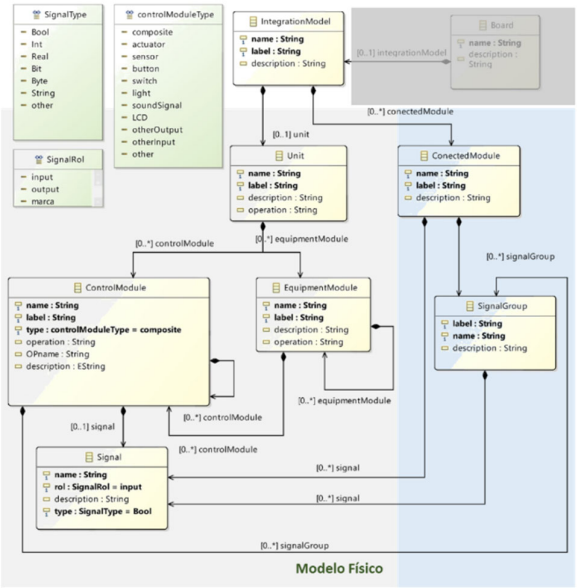

Figura 2: Metamodelo del modelo MI

En la Figura 3 se presenta el metamodelo del modelo MSC formado por elementos de control organizados jerárquicamente según las formas de mando, que pueden tener asociados procedimientos y sus señales de solicitud, repuesta y estado.

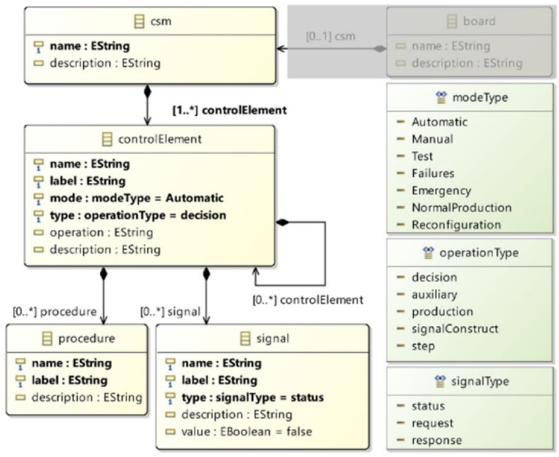

Figura 3: Metamodelo del modelo MSC

2.2. Herramienta de modelado

Los dos editores gráficos de los modelos de información se han construido siguiendo los pasos que se describen a continuación:

1.- Definir el DSL del modelo de información.

Crear un proyecto EMF en Eclipse (meia.integration), en el cual se diseña el DSL correspondiente mediante el lenguaje ecore. Dicho lenguaje permite definir metamodelos, guardarlos en un archivo con extensión .ecore y validarlos. La Figura 4 muestra el archivo .ecore para crear modelos MI.

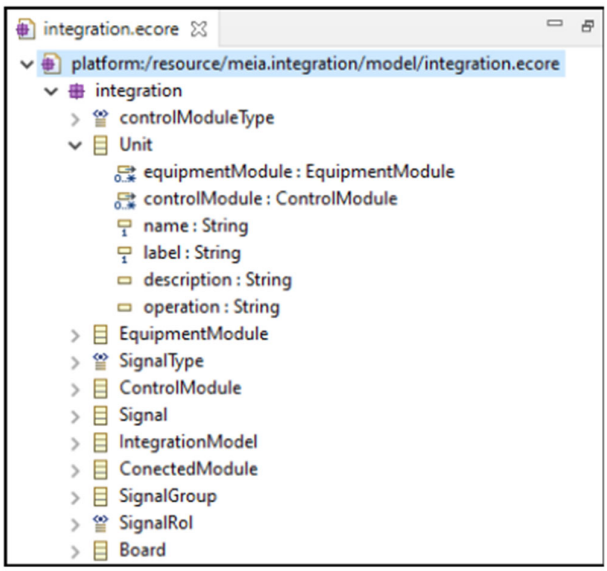

Figura 4: Archivo .ecore para crear modelos MI

2.- Generar el modelo EMF.

Generar las clases Java necesarias para instanciar y manipular los elementos definidos en el metamodelo con Sirius. Se generan lo proyectos meia.integration.edit y meia.integration.editor.

3.- Diseñar la representación gráfica.

Crear un Viewpoint Specification Project (meia.integration.project.design), el cual incluirá el archivo .odesign. En este archivo se definen las representaciones gráficas de los elementos del metamodelo, así como la paleta de herramientas que permitirá su manipulación (Figura 5).

4.- Distribuir e instalar el editor gráfico.

Crear un proyecto de tipo Feature que incluya todos los proyectos mencionados (meia.integration y sus asociados .edit,.editor y .project.design).

A continuación, crear un proyecto de tipo Update Site, que sirve para distribuir e instalar fácilmente los plugins en Eclipse y exportarlo como un archivo comprimido. Finalmente, el archivo exportado se agrega utilizando la opción Install New Software de eclipse.

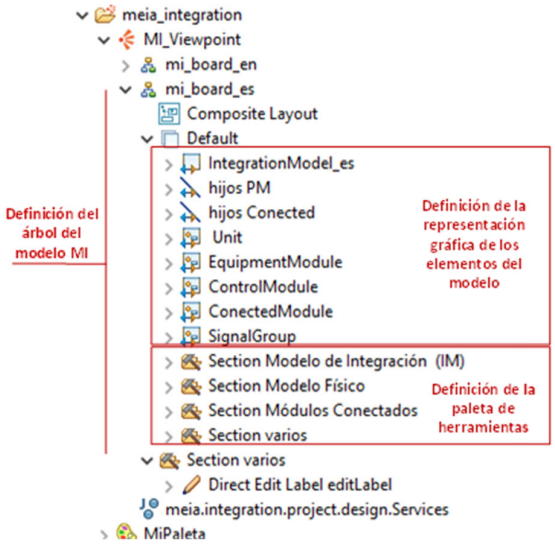

Figura 5: Archivo .odesign para el editor gráfico de modelos MI

La creación de un modelo MI o MSC con su editor gráfico precisa crear un proyecto de modelado, abrir el archivo de modelo en Eclipse, seleccionar el Viewpoint correspondiente, que activará las representaciones específicas diseñadas, y crear y editar el modelo usando dichas representaciones. Los modelos son almacenados en formato XMI. En la Figura 6 se presenta el editor gráfico para el modelo MI.

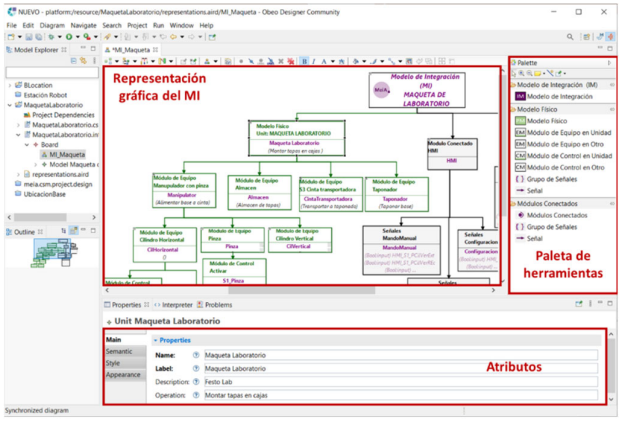

Figura 6: Editor gráfico de modelos MI.

2.3. Herramientas de preprocesado y validación

Como se muestra en el escenario de la Figura 1, esta herramienta corresponde a la primera etapa de la aplicación MeiA-GENXX. Antes de llevar a cabo la transformación de los modelos, se realiza un preprocesamiento que elimina la información no relevante contenida en los archivos XML generados por la herramienta de modelado. Asimismo, se realiza una validación estructural utilizando los esquemas XML (.xsd) asociados a los modelos de información, con el fin de asegurar que estén correctamente construidos. La

aplicación desarrollada en Java permite identificar los errores y editar los modelos para facilitar su corrección.

2.4. Herramienta de transformación

La segunda etapa de la aplicación MeiA-GENXX corresponde a la herramienta de transformación de los modelos que genera el esqueleto del proyecto de automatización del componente MeiA utilizando las transformaciones XSLT necesarias. El proceso de transformación puede comenzar con el procesado del modelo MI o del MSC, que creará el FB del componente. Además, a partir del MI, se crearán las entradas/salidas de su interfaz y la estructura estática para la información almacenada en el MI. Por otro lado, desde el MSC se crearán la estructura estática para la información almacenada en este modelo, el esqueleto de todos los FB correspondientes a los procedimientos y su instanciación dentro del FB del componente. En la Figura 7 se muestra un esquema de las reglas de trasformación utilizadas para la construcción del esqueleto del proyecto de automatización, que se describen a continuación.

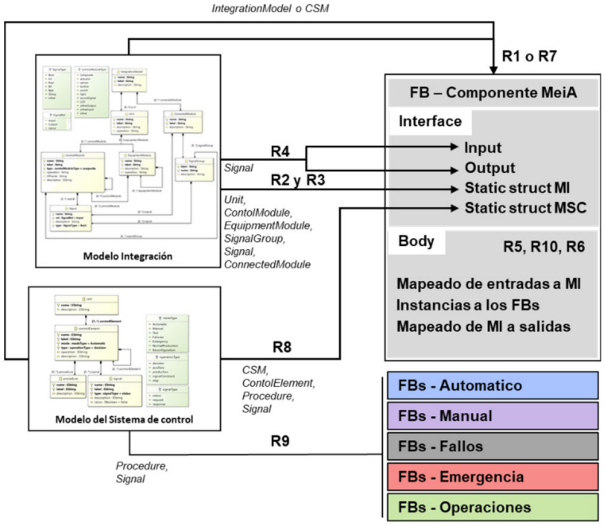

Figura 7: Esquema de las reglas de trasformación.

Las reglas de transformación definidas para el modelo MI son las siguientes:

R1.- Un elemento de tipo IntegrationModel crea un POU de tipo FB para el componente, siempre y cuando no haya sido creado previamente a partir de las transformaciones del modelo MSC. El interfaz del POU incluye su MI como variable estática de tipo estructura con identificador MI.

R2.- Un elemento de tipo Unit crea el modelo físico en la variable estática MI con el identificador ModeloFisico. Los elementos ControlModule, EquipmentModule y SignalGroup, que cuelgan del elemento Unit, se transforman en estructuras del modelo físico, siguiendo la misma jerarquía, con el identificador indicado en el atributo label del elemento. Los elementos Signal identificados al final de la jerarquía se convierten en variables con el identificador del atributo name.

R3.- Los elementos de tipo ConectedModule con sus elementos SignalGroup y Signal asociados se transforman, de acuerdo con su jerarquía, en otras estructuras de la variable estática MI al mismo nivel que el modelo físico. Al igual que en la regla anterior, los elementos Signal se identifican con el atributo name y el resto de los elementos con el atributo label.

R4.- Los elementos de tipo Signal del modelo físico se transforman en las entradas y salidas del interfaz del FB. Estos elementos serán entradas o salidas en función de su atributo rol, mientras que su nombre y su tipo estarán determinados por los atributos name y type.

R5.- En el cuerpo del FB del componente se mapean las entradas de su interfaz con las correspondientes variables de la estructura del modelo físico.

R6.- En el cuerpo del FB del componente se mapean las variables de la estructura del modelo físico con las correspondientes salidas de su interfaz.

Las reglas de transformación definidas para el modelo MSC, que permiten completar el esqueleto del proyecto de automatización del componente, son las siguientes:

R7.- El elemento raíz del modelo (CSM) crea un POU de tipo FB para el componente, siempre y cuando no haya sido creado previamente a partir de las transformaciones del modelo MI. El interfaz del POU incluye su MSC como variable estática de tipo estructura con identificador MSC.

R8.- Los elementos csm, controlElement y procedure se transforman en estructuras en la variable estática MSC, siguiendo la misma jerarquía, con el identificador indicado en el atributo label del elemento. Los elementos Signal identificados al final de la jerarquía se convierten en variables con el identificador del atributo name.

R9.- Un elemento de tipo procedure se transforma en un FB con identificador igual a su atributo label. Las entradas y salidas de su interfaz se obtienen de sus elementos signal asociados.

R10.- Los FB de los procedimientos se instancian dentro del cuerpo del FB del componente.

3. Ejemplo de aplicación

Como ejemplo para la generación de un componente MeiA, se ha utilizado la maqueta de aprendizaje MecLab de Festo que coloca una tapa sobre una base. Consta de un brazo manipulador que introduce las bases sobre una cinta transportadora y un sensor para identificar el material de la pieza. Si el material no es el correcto, la base es descartada. En caso contrario, la base se traslada a la zona de taponado en la cual se coloca la tapa que se extrae de un almacén.

En primer lugar, se generan los modelos MI y MSC mediante la herramienta gráfica mostrada en la Figura 6. A continuación, los documentos XML generados correspondientes a dichos modelos se introducen en la herramienta MeiA_GENXX, que los preprocesa y valida con el motor XSD, permitiendo su modificación en caso de que sea necesario. Finalmente, se configura la transformación, seleccionando el formato de destino, así como el fichero de

salida. El documento XML generado se importa en la plataforma de desarrollo del proyecto de automatización correspondiente. La Figura 8 muestra la interfaz gráfica de la herramienta MeiA_GENXX.

En este trabajo, se genera código para un PLC de Siemens con CPU 1512C-1PN, que incorpora el lenguaje gráfico GRAPH equivalente a SFC. El documento XML generado por la herramienta se importa en TIA Portal mediante la aplicación Openness Scripter.

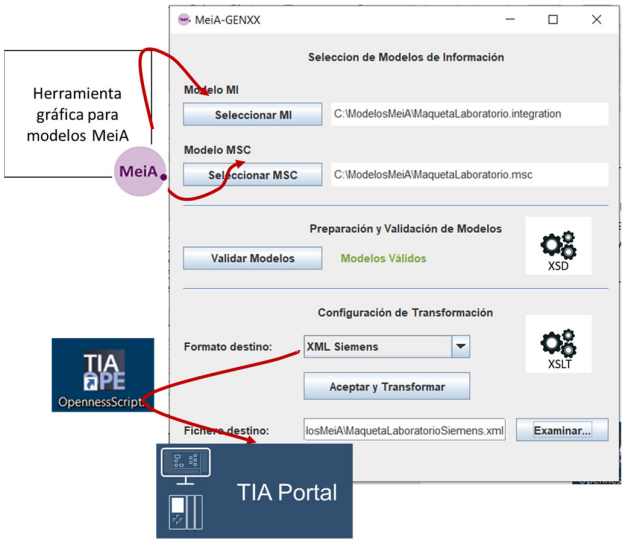

Figura 8. Interfaz gráfica de MeiA_GENXX.

A modo de ejemplo, en la Figura 9 se presenta la regla de transformación R4 utilizada por el motor de transformación XSLT, donde el elemento *signal* es transformado en un elemento de tipo *input* dentro de la interfaz del bloque del componente.

Figura 9. Regla de transformación R4

4. Conclusiones

La generación automática de los proyectos de automatización permite combinar metodologías, estándares y buenas prácticas de ingeniería sin exigir un conocimiento profundo de cada tecnología.

El resultado es un software más modular que reutiliza diseños, minimiza errores humanos, mejora la calidad del software, acorta tiempos de desarrollo y de puesta en marcha, reduce costes y facilita su mantenibilidad, documentación y adaptación a cambios futuros.

Agradecimientos

Este trabajo ha recibido el apoyo del proyecto ELKARTEK KK-2025/00035.

Referencias

Behrendt, S., Martin, M., Puchta, A., Ströbel, R., May, M. C., Gönnheimer, P., Fleischer, J., Lanza, G., 2023. Software-Defined Manufacturing for the Entire Life Cycle at Different Levels of Production. In: Kiefl, N., Wulle, F., Ackermann, C., Holder, D. (Ed.) Advances in Automotive Production Technology – Towards Software-Defined Manufacturing and Resilient Supply Chains. Springer, Cham.
DOI: 10.1007/978-3-031-27933-1_3

Cochran, D. S., Smith, J., Fitch, J., 2022. MSDD 10.0: a design pattern for sustainable manufacturing systems. Production and Manufacturing Research 10 (1), 964–989.
DOI: 10.1080/21693277.2022.2153184

Diaz, R. A. C., Ghita, M., Copot, D., Birs, I. R., Muresan, C., Ionescu, C., 2020. Context aware control systems: An engineering applications perspective, IEEE Access 8, 215550–215569.
DOI: 10.1109/access.2020.3041357.

IEC, 2017. PAS smart manufacturing -reference architecture model industry 4.0 (RAMI 4.0). Technical Report, IEC.
https://webstore.iec.ch/publication/30082

IEC 61512-1, 2002. Batch control Part 1: Models and terminology. Technical Report, IEC. https://webstore.iec.ch/publication/5528

IEC 62264-1, 2013. Enterprise-control system integration - Part 1: Models and terminology. Technical Report, IEC.
https://www.iso.org/standard/57308.html.

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.

Obeo, 2026. Obeo Designer. https://www.obeosoft.com/en/products/obeo-designer/. Accessed 12 May 2026.

Sarachaga, I., Alvarez, M. L., Iriondo, N., Burgos, A., Armentia, A., Casquero, O., Hurtado, E., 2026. Metodología MeiA innovación en el desarrollo de software de control para sistemas de producción automatizados. EHU (Ed.). Leioa.
ISBN: 978-84-1319-749-4
http://hdl.handle.net/10810/78349

Vogel-Heuser, B., Fischer, J., Neumann, E. M.. Kreiner, M., 2021. Success factors for the design of field-level control code in machine and plant manufacturing - an industrial survey. Research Square.
DOI: 10.21203/rs.3.rs-168613/v1

W3C, 2008. Extensible Markup Language (XML) 1.0 (Fifth Edition).
https://www.w3.org/TR/2008/REC-xml-20081126/

W3C, 2012. XML Schema Definition Language (XSD) 1.1 Part 1: Structures.
https://www.w3.org/TR/2012/REC-xmlschema11-1-20120405/

W3C, 2017a. XML Path Language (XPath) 3.1.
https://www.w3.org/TR/2017/REC-xpath-31-20170321/

W3C, 2017b. XSL Transformations (XSLT) Version 3.0.
https://www.w3.org/TR/2017/REC-xslt-30-20170608/

Wegert, V. and Shatalin, A., 2008. Integrating EMF and GMF Generated Editors. Eclipse.org.
https://www.eclipse.org/articles/Article-Integrating-EMF-GMF-Editors/index.html

Wu, D., Chau, K. T., Wang, J., Pan, C., 2019. A comparative study on performance of XML parser APIs (DOM and SAX) in parsing efficiency. In Proceedings of the 3rd International Conference on Cryptography, Security and Privacy, 88–92.
DOI: 0.1145/3309074.3309124

Zisman, A., 2000. An overview of xml. Computing & Control Engineering Journal, 11(4),165–