Escuela Politécnica Superior de Jaén

Grado en Ingeniería Electrónica Industrial

UNIVERSIDAD DE JAÉN
Nombre del Centro

Trabajo Fin de Grado

GEMELO DIGITAL DE LA ESTACIÓN 8 DE CÉLULA DE FABRICACIÓN FLEXIBLE FMS200

Alumno: Óscar Espejo Quesada

Tutor: Prof. D. Alejandro Sánchez García
Prof. Dª. Elisabet Estévez Estévez
Dpto: Ing. Electrónica y Automática

Febrero, 2022

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Informática

Don Alejandro Sánchez Garcia y Dª Elisabet Estévez Estévez , tutores del Proyecto Fin de Carrera titulado: Gemelo Digital de la Estación 8 de Célula de Fabricación Flexible FMS200, que presenta Óscar Espejo Quesada, autoriza su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, Febrero de 2022

El alumno:

ÓSCAR ESPEJO QUESADA

Los tutores:

SANCHEZ GARCIA ALEJANDRO - 77321466C
Firmado digitalmente por SANCHEZ GARCIA ALEJANDRO - 77321466C
Fecha: 2022.02.22 13:37:27 +01'00'

ESTEVEZ ESTEVEZ ELISABET - 72578245C
Firmado digitalmente por ESTEVEZ ESTEVEZ ELISABET - 72578245C
Fecha: 2022.02.22 13:42:36 +01'00'

ALEJANDRO SÁNCHEZ GARCÍA Y ELISABET ESTEVEZ ESTEVEZ

RESUMEN

En este proyecto se ha llevado a cabo el Gemelo digital de la octava estación de la Célula de Fabricación Flexible FMS200.

En primer lugar, se ha desarrollado el modelado 3D de cada una de los elementos de la estación, tanto actuadores mecánicos como accesorios que incluye, de forma que se pueda hacer una representación lo más acorde con la realidad posible. La formación del conjunto virtualizado se ha realizado mediante el lenguaje de programación Processing, siendo importado dicho lenguaje mediante una librería dedicada en NetBeans. De este modo, se puede trabajar de forma sencilla aprovechando tanto las ventajas de Processing como las de Java para realizar el clonado de la estación de forma virtual.

Para llevar a cabo la lectura de los datos, se ha utilizado el formato XML como novedad al respecto los anteriores proyectos de la serie. De este modo se importa los datos fácilmente a través de un solo archivo que recoge todos los elementos.

Una vez preparado el entorno de programación, se ha dotado a cada clase de su funcionalidad específica a través de la adición de atributos que caracterizan los elementos que componen la máquina. Cada elemento será pilotado gracias a un PLC programado en el software TIA Portal y que se encargará de gestionar la funcionalidad.

Por otro lado, se ha llevado a cabo la comunicación entre el gemelo digital y un PLC virtual de Siemens, lo que da a lugar que permita ser controlado desde el autómata. Esto da lugar a la representación virtual de la estación FMS-208 totalmente funcional.

SUMARY

In this project, the Digital Twin of the eighth station of the FMS200 Flexible Manufacturing Cell has been carried out.

In the first place, the 3D modeling of each of the elements of the station has been developed, both mechanical actuators and accessories that it includes, so that a representation can be made as consistent with reality as possible.

The formation of the virtualized set has been carried out using the Processing programming language, being imported this language through a dedicated library in NetBeans. In this way, you can work in a simple way taking advantage of both the advantages of Processing and Java to perform the cloning of the station virtually. To carry out the reading of the data, the XML format has been used as a novelty in this regard the previous projects of the series. In this way, the data is easily imported through a single file that collects all the elements.

Once the programming environment has been prepared, each class has been endowed with its specific functionality through the addition of attributes that characterize the elements that make up the machine. Each element will be piloted thanks to a PLC programmed in the TIA Portal software and that will be responsible for managing the functionality.

On the other hand, the communication between the digital twin and a Siemens virtual PLC has been carried out, which gives rise to the fact that it can be controlled from the automaton. This results in the fully functional virtual representation of the FMS-208 station.

1 Contenido

FIGURAS

1. INTRODUCCIÓN

1.1 Motivación

Desde aproximadamente el año 2010, ha sido desarrollada la conocida cuarta revolución industrial, la cual está revolucionando la forma de vida tanto de las personas, como de la industria y su metodología de trabajo. Cada vez son más las fábricas que dan el paso a la automatización, diseñando procesos más automáticos e inteligentes que son capaces de actuar de forma independiente y ofrecen un resultado de proceso rápido y de buena calidad.

El término industria 4.0 surgió en Alemania para hacer referencia a la cuarta revolución industrial. Esta transición, está liderada por los sectores más competitivos como es el de la automoción, el agroalimentario, el farmacéutico y el de producción industrial, que andan en continua búsqueda de reducción de costes sin pérdida de la calidad del producto final. Para satisfacer la demanda, la respuesta es la digitalización de la fábrica, pero para ello es necesario definir las necesidades y capacidades, dominar el manejo de los datos y su conectividad, y adaptar adecuadamente la fábrica al entorno digital (Perasso, 2016). El llamado internet de las cosas juega un papel fundamental en esta etapa, además de la gestión de datos masivos (Big Data), la fabricación por adición o impresión 3D, la ciberseguridad y la simulación. Véase en la Figura 1 una ilustración que recoge las tecnologías utilizadas en estas nuevas industrias.

Figura 1. Tecnologías de la Industria 4.0

Un claro ejemplo de los objetivos de la industria 4.0 es la interconexión de los procesos de fabricación, es decir, evitar que cada una de las máquinas integren su propio lenguaje y hacer que todos los equipos de fabricación de se recojan en el mismo sistema de gestión de producción **(MES)** para conseguir una gestión totalmente automática y centralizada, que da a lugar a procesos más simples, menos errores y un aumento del volumen de producción (Silvia, 2020).

Un Sistema MES (*Manufacturing Execution System*) es un software enfocado al control de la producción, que monitoriza y documenta la gestión de la planta. Puede verse como el paso intermedio entre las decisiones de negocio y las decisiones de procesos. Véase en la Figura 2 un ejemplo de jerarquía industrial para integrar dicho sistema de control.

Figura 2. Pirámide de planificación industrial

En esta nueva etapa, ha entrado en juego una nueva herramienta muy potente, los llamados Gemelos Digitales, los cuales son capaces de simular de forma virtual cualquier proceso dando la oportunidad al usuario de anticipar y predecir los efectos y comportamientos de la máquina representada en cuestión siendo sometida a diferentes situaciones que pueden darse en un caso real. Teniendo la ventaja de poder hacer las pruebas necesarias en la simulación sin correr el riesgo de tener una avería en el real en el caso que haya un fallo en de hardware o de software.

En adición, una de las grandes ventajas que ofrece esta herramienta es la posibilidad de trabajar de forma telemática a distancia. Por lo tanto, tras un año 2020 azotado por una pandemia mundial que causó la paralización de casi toda la actividad presencial, surge la idea de poner en práctica dicha herramienta usando como ejemplo la Célula de Fabricación Flexible FMS200 en la Universidad de Jaén. Este método ofrece la posibilidad de hacer una simulación real del caso práctico físico en el laboratorio y da lugar a minimizar los contagios que provocan el colapso de los hospitales.

De esta forma, una vez realizado el entorno del gemelo digital, en la asignatura de Automática Avanzada, se podrá adaptar cada una de las estaciones a su correspondiente gemelo digital simplemente creando el diseño 3D correspondiente e

importándolos al espacio de trabajo. En el proyecto se busca la transversalidad hacia cualquier caso práctico, de modo que, no solo se podrá adaptar a las demás estaciones de la Célula de Fabricación Flexible FMS200, sino a cualquier otro proceso que incluya similares elementos mecánicos podrá ser implementado.

1.2 Estructura

Esta memoria descriptiva se ha realizado en el mismo orden de la realización del proyecto. En primer lugar, se ha desarrollado una breve introducción que resuma el porqué de este trabajo, en el que se introduce la situación actual en el mundo de la industria y la utilidad de desarrollar este trabajo. Además de, las novedades que se introducen en esta nueva actualización de la línea de proyectos de Gemelos Digitales de la Universidad de Jaén.

A continuación, se indicarán todas y cada una de las bases generales que han sido necesarias para el desarrollo de la actividad. En esta parte, se quiere demostrar la flexibilidad de la que se dota al proyecto para poderse adaptar a cualquier estación de la Célula de Fabricación Flexible FMS200 o a cualquier proceso relacionado con la automatización industrial. Para ello, se incluyen todos los aspectos detallados utilizados, como pueden ser librerías importadas, clases o estructuras de archivos donde se recogen los datos de los objetos.

Posteriormente, se particularizará el Gemelo Digital para el caso práctico de la estación 8, la cual se encarga del almacenaje de las piezas ya procesadas por todas y cada una de las estaciones anteriores. Este almacenaje se basa en un eje cartesiano formado por dos actuadores eléctricos que se combinarán para dar la función de paletizado al sistema.

Además, se incluirá una parte dedicada al manual del usuario. En esta parte se describe el funcionamiento del gemelo digital particularizado a la estación en cuestión, por lo que ofrece la posibilidad de que cualquier persona pueda ser capaz de controlar la simulación e interactuar con ella.

Por último, se añadirán las conclusiones que han sido extraídas del proceso de fabricación del proyecto y los anexos con la documentación pertinente.

1.3 Objetivos

Siguiendo la línea de los proyectos anteriores relacionados con la creación de gemelos digitales de otras estaciones de la Célula de Fabricación Flexible FMS200, se ha implementado una nueva estación más, pero con nuevas particularidades que ofrecerán a la serie unas nuevas mejoras que servirán para capítulos futuros.

Entre los objetivos de este proyecto, se encuentra la importación del procesado de archivos XML (Proveniente de la expresión *eXtensible Markup Language*). Hasta ahora, la importación de datos al sistema había sido realizada mediante formato YML, añadiendo así, tantos archivos como elementos contuviese el conjunto. Este sistema generaba un gran caos en la gestión de los datos, primero por tener un gran número de ficheros que procesar y segundo, por la estructura que debía tener.

Con este nuevo formato se consigue la unificación de todos los elementos que conforman la estación sean unificados en un mismo fichero, lo que mejora la eficiencia a la hora de ser ejecutado y la sensación de organización. Además, este formato ofrece una estructura sencilla, fácil de leer y se puede gestionar más cómodamente. También, NetBeans permite su apertura, por lo que se puede tener en el mismo IDE tanto el proyecto como el archivo desde donde se importan los datos.

Por otro lado, como nueva aportación a la serie de proyectos se ha añadido la creación de una nueva clase llamada “ActuadorElectrico”. Esta importación ha sido necesaria ya que, la estación numero 8 dispone de un eje cartesiano formado por dos actuadores lineales que funcionan con un motor eléctrico rotativo. Dichos motores no comparten funcionalidad con el resto de elementos ya implementados para las demás estaciones, por lo tanto, ha sido necesario la creación de una clase propia que caracterice el funcionamiento de dicho hardware.

Por último, un detalle que se considera de bastante utilidad a la hora de ejecutar un programa es la generación de un archivo ejecutable JAR. Este archivo da la oportunidad de desplazar el software a cualquier dispositivo que pueda ejecutar un archivo Java y hacer uso del gemelo digital implementado en el proyecto.

2 GEMELO DIGITAL

2.1 Introducción

En este apartado de la memoria se hará referencia a los aspectos generales característicos de la herramienta virtual tan utilizada en los últimos años. Haciendo referencia a las principales funciones de un gemelo digital, se desarrollan según tres usos fundamentales (Herranz, 2021):

- Prototipo de gemelo digital (DTP): Antes de crear un producto físico final, se realiza un gemelo para ver como quedaría realmente y cómo se comportaría.
- Instancia gemela digital (DTI): Una vez fabricado el producto, se emplea el clon para hacer las pruebas necesarias en diferentes escenarios de uso.
- Digital Twin Aggregate (DTA): Recopila información para determinar capacidades del proceso para realizar pronósticos y hacer pruebas de parámetros.

En primer lugar, antes de llevar a cabo la programación, se ha hecho una distinción entre los elementos físicos que intervienen. Posteriormente se han desarrollado una programación distinguida para cada clase en el lenguaje Java. Para ello se ha utilizado el software NetBeans. Véase en apartado 2.2 NetBeans

Por lo que se refiere a la movilidad del gemelo digital, se ha utilizado el “Algoritomo de Denavit-Hatenberg” (Parejo, 2008) como base principal del funcionamiento. Para ello se ha tratado un procedimiento para describir la estructura cinemática de una cadena articulada constituida por articulaciones con un solo grado de libertad. Se considerará un sistema de referencia local para cada una de las articulaciones, siendo la primera de ellas fija en inmóvil anclado en la base del robot. Por lo tanto, si se produce un movimiento en una de las articulaciones que tenga a otras como esclavas, el movimiento repercutirá en cada una de las hijas. Un ejemplo se encuentra en la Figura 3, una representación gráfica del comportamiento de un robot según la articulación que sea movilizada.

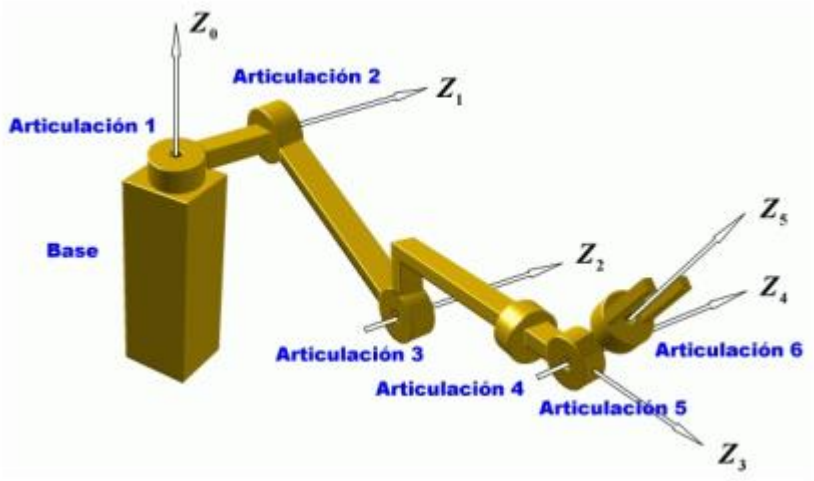


Figura 3. Algoritmo de Denavit-Hatenberg

2.2 Apache NetBeans

NetBeans comenzó en el año 1996 en República Checa cuando un grupo de estudiantes desarrollaron como proyecto el primer entorno de desarrollo integrado para Java llamado Xelfi, escrito en este mismo lenguaje. Posteriormente, este software fue lanzado por Sun MicroSystems en el año 2000. Posteriormente, fue adquirida por la empresa Oracle Corporation y donada a la asociación Apache, de ahí el nombre Apache NetBeans (Calendamaia, 2014).

Esta IDE (Integrated Development Enviroment) o entorno de desarrollo integrado de código abierto y gratuito sin restricciones de uso, hecho principalmente para el lenguaje de programación Java. Esta IDE incluye su propio editor de código, compilador y depurador, el cuál es muy útil a la hora de buscar un fallo en la programación ya que se puede seguir paso a paso cada una de las líneas del código del programa y comprobar el estado además del valor de las variables. Puede observarse su logo en la Figura 4.

Figura 4. Icono Apache NetBeans

2.2.1 Librerías importadas

Una funcionalidad muy usada en Java NetBeans es la importación de librerías. Son paquetes creados por terceros que podemos agregar a nuestros proyectos para enriquecer el sistema o aprovechar ciertas funcionalidades. Es una forma de simplificar el código y aumentar las posibilidades de programación (Univision, 2018). NetBeans ofrece un amplio catálogo de librerías para ser utilizadas, para ello se debe usar el comando “import”.

2.2.1.1 Processing

En el caso de este proyecto, se han utilizado varias librerías para adecuar y facilitar la programación del gemelo digital. La primera de ellas es la librería de Processing. Puede ser observado en la Figura 5 y la Figura 6.

```
import processing.core.*;
```

Figura 5. Importación de la librería

Figura 6. Icono de Processing

Una vez importada la librería al IDE, se puede hacer uso de todas las funciones de las que disponga Processing dando más versatilidad al código. Las principales son las funciones setup() y draw(), que dotan al sistema de la capacidad de ser cíclico al igual que el funcionamiento del PLC y de poder hacer representaciones 3D en una ventana emergente desde el software de Java.

2.2.1.2 Moka7

La librería de Moka7 (Snap7, s.f.), la cual simula el protocolo de comunicación con los PLCs S7 para dotar de la misma funcionalidad que la estación real al gemelo digital. Moka7 al ser una parte de Snap7 que es compatible con NetBeans, comparte la misma licencia, por lo que no tiene problemas de comunicación. Véase la Figura 7.

Figura 7. Icono de Moka7

Una vez incluida la librería en el proyecto, se generará un paquete en “Source Packages” llamado “Moka7”, la cual incluye todas las clases responsables de realizar la comunicación entre NetBeans y el PLC virtualizado. Véase la Figura 8.

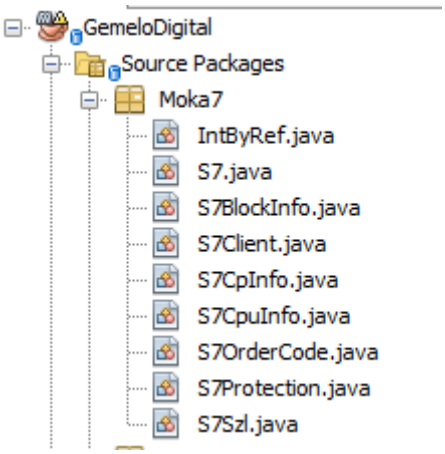


Figura 8. Paquete de Moka7 en NetBeans

2.2.2 Clases definidas

Como se ha mencionado anteriormente, se han gestionado diferentes clases de objetos según la función que cumplan físicamente. Todas han sido creadas en el mismo paquete del proyecto llamado con el nombre “com.grav.gemelodigital”. Véase en la Figura 9.

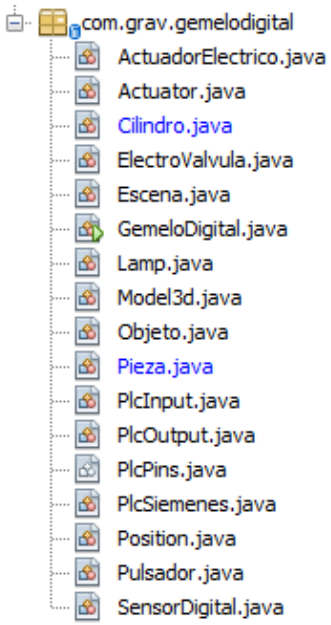


Figura 9. Paquete de clases Gemelo Digital

En la Figura 9. Paquete de clases Gemelo Digital se pueden visualizar la lista de clases que conforma el paquete. Para que el objetivo del proyecto sea logrado, se ha de describir los objetos virtuales lo más similares posible a la realidad. Para ello, se establece una relación entre elementos mostrada en la Figura 10.

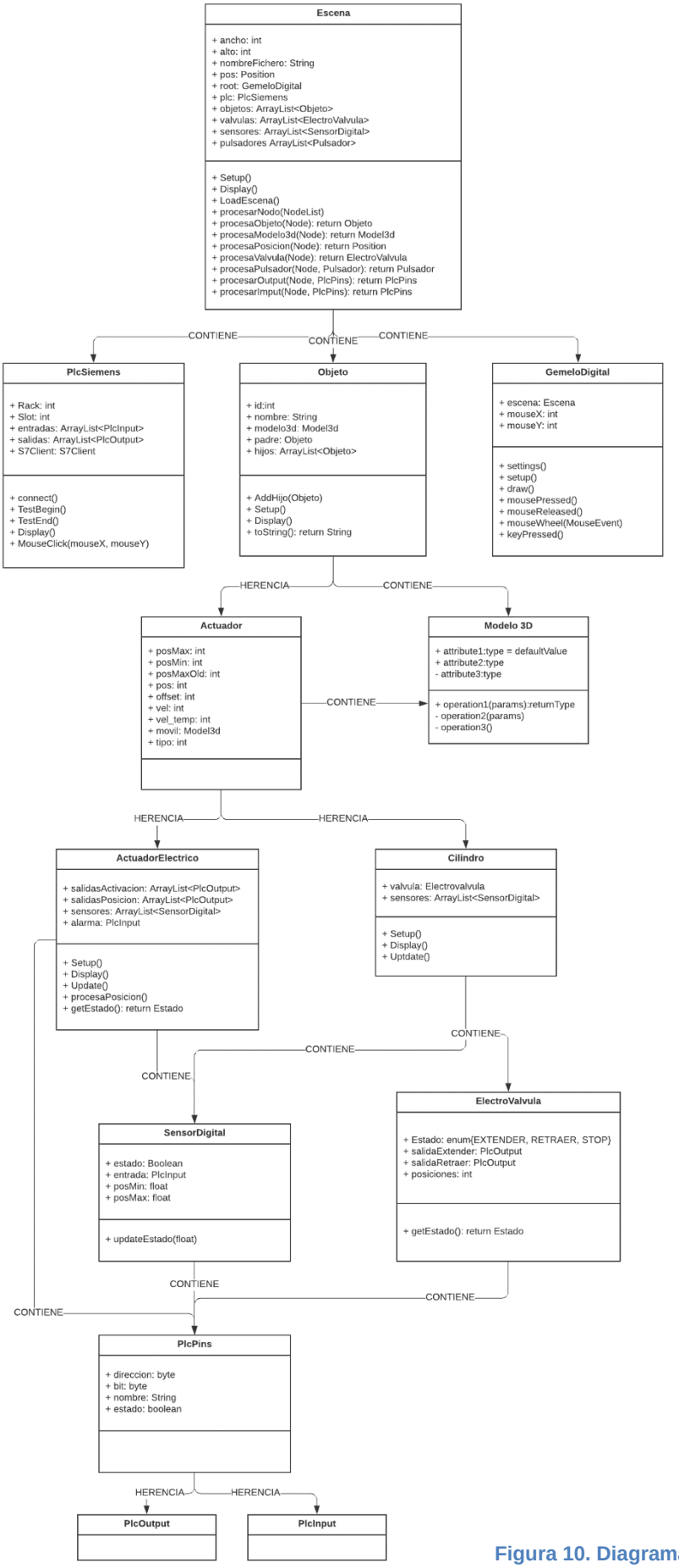


Figura 10. Diagrama UML de clases

A continuación, se van a describir los rasgos generales que conforman cada uno de los ítems creados en el archivo Java. Cada uno de ellos funcionan de forma autónoma e independiente, dando la facilidad de trabajar en un entorno donde se pueda controlar cada elemento por separado. Para ejemplificar este caso se puede nombrar la clase “SensorDigital”, que como se podrá observar, es capaz de establecer comunicación directa con el PLC a través de las entradas asignadas, de modo que no es necesario otra clase esclava que establezca la comunicación.

2.2.2.1 Gemelo Digital

La clase “GemeloDigital” es clase es la encargada de gestionar la interacción entre la interfaz gráfica y las funciones del gemelo como son la gestión de la vista o de la interacción con el teclado y ratón, además es una clase transversalizable, ya que dependiendo de la escena que sea configurada, se puede disponer de diferentes gemelos digitales. Se ve reflejado en la Figura 10 que ésta está compuesta por algunos atributos y métodos que serán descritos a continuación.

- **Atributos**

Los atributos implementados en esta clase son escasos. Simplemente se utiliza para cargar la escena correspondiente e interactuar con el ratón.

- **Métodos**

Los métodos utilizados son destinados a la creación de un nuevo objeto escena en el “settings()”, cargar la escena en la función “setup()” y de llamar a la representación en la función “display()”.

Los métodos restantes son utilizados para la interactuación con el gemelo. El “mousePressed()” se ejecuta cuando se hace clic izquierdo con el ratón. El “mouseReleased()” cuando se desactiva el clic del ratón anterior. El “mouseWheel()” se ejecuta cuando entra en acción la rueda del ratón. Y por último, el “keyPressed()” se da cuando se pulsan determinadas letras del teclado.

Por último, han sido importadas otras funciones que son de gran utilidad para la interacción con la interfaz gráfica. Entre ellas están la función de poder desplazar la representación en cualquiera de los ejes en función de la tecla pulsada. También, se

ofrece la posibilidad de girar la vista de representación del gemelo dando la posibilidad de dotar la escena de tres dimensiones.

2.2.2.2 Objeto

Véase en la Figura 10 que la clase llamada “Objeto” se sitúa a la cabeza de los elementos que intervienen en la representación del gemelo digital. “Objeto” es designada como padre de todos los elementos, se le atribuyen características generales que contiene características básicas de un objeto cualquiera. El resto de clases heredarán sus atributos y métodos para dotarlas de la posibilidad de ser consideradas objetos como tales, la función principal es la representación 3D de un modelo fijo.

- **Atributos**

El parámetro “id” atribuye un identificador a cada uno de los objetos de la escena, de manera que se podrán acceder de forma rápida y sencilla simplemente sabiendo el número de identificador del objeto que queremos.

El atributo “nombre” da simplemente un apelativo al objeto. Existe la posibilidad de variar este campo a elección del usuario.

El atributo “modelo3d” implementa la clase llamada “Model3d” en el objeto en cuestión, la cual se encarga de asignar a cada objeto un archivo OBJ a la hora de hacer la representación 3D de la escena.

PADRE

El ArrayList de objetos llamado “hijos” está dedicado a recoger todos los objetos que van a depender de este objeto creado. De este modo, todas las modificaciones que sufra el objeto principal, tendrán repercusión en los objetos que sean hijos de éste. La utilidad de este atributo está basada la concatenación de las articulaciones descrita en el “Algoritmo de Denavit-Hatenberg”. Véase la Figura 3.

- **Métodos**

Los métodos implementados en esta clase tienen carácter general que describen el funcionamiento de cualquier objeto, así como su representación. Posteriormente, cada clase implementa sus propios métodos específicos que definen su funcionalidad.

El método “AddHijo(Objeto)” se utiliza para añadir un objeto esclavo, por lo tanto, al someterse el objeto padre a alguna modificación en el espacio, el objeto hijo sufrirá una alteración de igual magnitud.

Los métodos “setup()” y “display()” son los usados siguiendo la línea de funcionamiento de Processing para llevar a cabo la representación 3D. En método “setup()” es ejecutado solamente una vez al iniciar el programa para cargar el archivo OBJ al proyecto y ser almacenado en la memoria del sistema. Después, será representado en la ejecución gracias al método cíclico “display()”. En cada ciclo Scan del controlador hará una llamada a dicha función. Es normalmente utilizada para actualizar el estado de la posición de los objetos.

El repetitivo uso del ciclo Scan es posible gracias a la continua ejecución del método “draw()” importado de la librería de Processing, que es ejecutadao en la clase “GemeloDigital”. En él, se llama al método “display()” de cada uno de los objetos del proyecto, lo que da a lugar que esté en continua actualización. Dentro de esta función, se sigue un proceso de representación característico del entorno de Processing. En primer lugar, se hace uso del “pushMatrix()” (Processing References) y el “popMatrix()” (Processing References), las cuales permiten generar una matriz de transformaciones en el sistema de representación entre las líneas de código que comprendan ambas funciones. En cada objeto se hace uso de un “pushMatrix()” y de un “popMatrix()”.

De este modo, una vez establecido el espacio de trabajo, haciendo uso de la función “translate()” se da la opción de desplazar el origen de representación al punto deseado para llevar a cabo la representación del objeto la posición adecuada. Además, mediante la función “rotate()” se puede modificar la orientación del origen de representación. En este caso, se han dispuesto todos los objetos con el origen en su dirección de expansión, de este modo se facilita la actualización de la posición en el proceso del movimiento, ya que solamente se ha de tratar el desplazamiento en una

coordenada. Para simplificar se recurre a la función “drawAxis()” para dibujar el origen de cada pieza, de este modo se puede observar el origen de representación de cada objeto y resulta más intuitivo a la hora de asignar el valor de las coordenadas en el espacio. Una vez que el origen está situado en la posición deseada, se representa el objeto a través de una llamada al método “shape()”. Reflejado en la Figura 11.

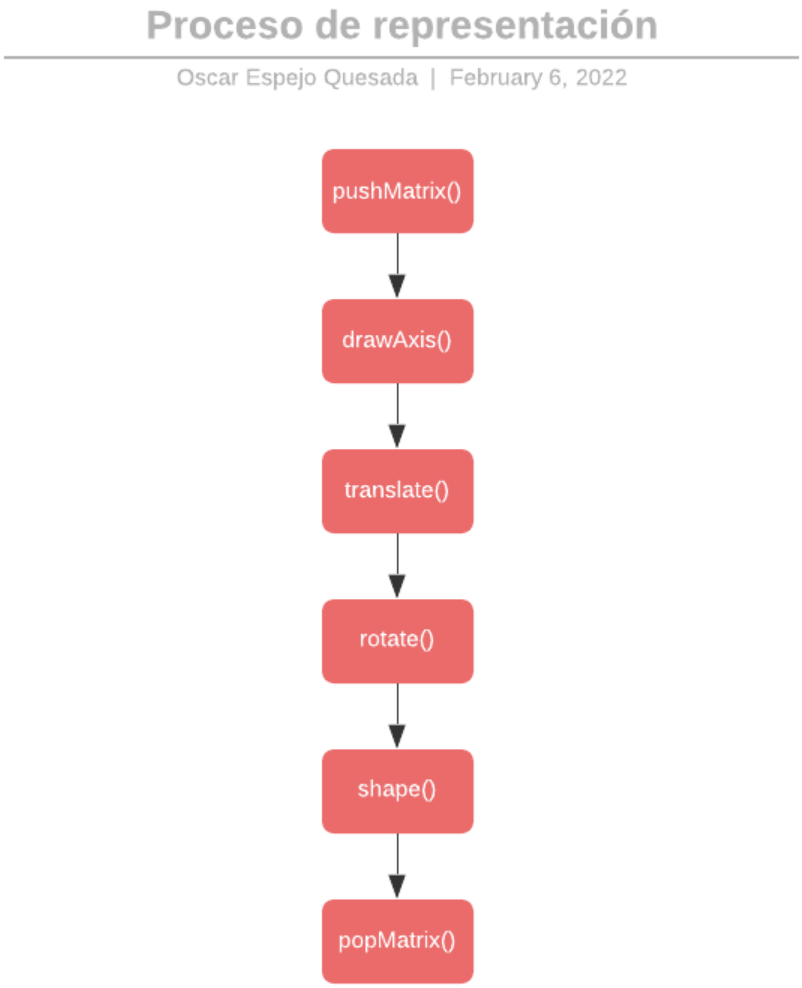


Figura 11. Proceso de representación de un objeto

Para realizar el montaje de la estación en el escenario de trabajo virtual se ha desarrollado un patrón de representación. En primer lugar, se representa el objeto base de la cadena. Posteriormente, se llama a la representación al objeto u objetos que tenga como hijos. Esto iniciará un bucle debido a que ese objeto hijo, cumple el mismo patrón que el padre, tendrá otro objeto hijo implementado, que volverá a ser llamado a representarse. El ciclo acabará cuando el último objeto que se llame a la representación no tenga ningún objeto esclavo. Así, se cumple el algoritmo de Denavit-Hatenberg (Parejo, 2008).

2.2.2.3 Actuador

La clase “Actuador” es una de las clases que heredan de la clase “Objeto”. La función principal designada a esta clase es actuar como la parte móvil de un objeto, en ella se incluyen atributos relacionados con el movimiento en el espacio de la pieza representada. Por lo tanto, a través de los atributos implementados en la clase padre más los de esta otra, será posible la creación de objetos dotados de funciones de movilidad como es el caso de un cilindro, el cual contiene una base que actúa como parte fija y un vástago que actúa como parte móvil. Haciendo referencia a la Figura 10 las propiedades implementadas en esta clase.

- **Atributos**

Los atributos que son implementados en esta clase se usarán en clases hijas para dotar de movilidad al conjunto. Incluyendo así, variables como posición mínima (“posMin”), posición máxima (“posMax”) o posición actual del objeto (“pos”), la velocidad viene determinada por la variable “vel” y “vel_temp” y la dimensión de la pieza para hacer el salto entre posiciones consecutivas del almacén se recoge de la variable “offset”. En adición, se importa un objeto 3D para hacer la representación del elemento móvil.

- **Métodos**

En esta clase no se implementan métodos.

2.2.2.4 Cilindro

En este caso, este objeto hereda de la clase “Actuador” (la cual hereda de clase “Objeto”), por lo tanto, la base del cilindro se representaría la base de forma fija gracias a los atributos de la clase “Objeto” mientras que el vástago, como parte móvil a través de los atributos heredados de la clase “Actuador”. Recordando que en la Figura 10 los atributos y métodos definidos para “Cilindro”.

- **Atributos**

En esta clase se incluyen dos atributos que siempre incluyen un cilindro real y controlan la actuación del objeto. Dichos elementos son sensores de final de carrera y electroválvulas. Véase en la Figura 10.

El atributo llamado “válvula”, es implementado desde la clase “Electroválvula”, es la encargada de dar la orden de expandir o retraer el actuador. Gracias a este atributo, se pueden determinar todo tipo de cilindros según las salidas que tenga asignadas la válvula.

Po otro lado, el siguiente atributo es llamado “sensores”, pertenece a la clase “SensorDigital”. Incluye todos los sensores que se incorporan al objeto, que son los encargados detectar los finales de carrera del vástago, tanto la retracción como la expansión. Normalmente, un objeto incluye dos sensores para notificar la posición de expandido y otro para la posición de retraído.

- **Métodos**

En primer lugar, esta clase incluye los métodos “setup()” y “display()” de la librería de Processing para realizar la representación.

Por otro lado, el método que da la funcionalidad de movilidad al cilindro es el método “Update()”, que es llamado desde la función “display()” en cada ciclo Scan.

Este método consulta el estado de la válvula contenida en el objeto creado de clase “Cilindro” y, según el estado que sea devuelto, se gestionará la orden de actuación. Si al consultar el estado de la válvula, ésta devuelve el estado “EXPANDIR” o “RETRAER”, la función actualizará la posición de representación de la parte móvil de forma progresiva y marcada por la velocidad del parámetro “vel” heredado de la clase “Actuador”. Dando constancia del proceso en la Figura 12.

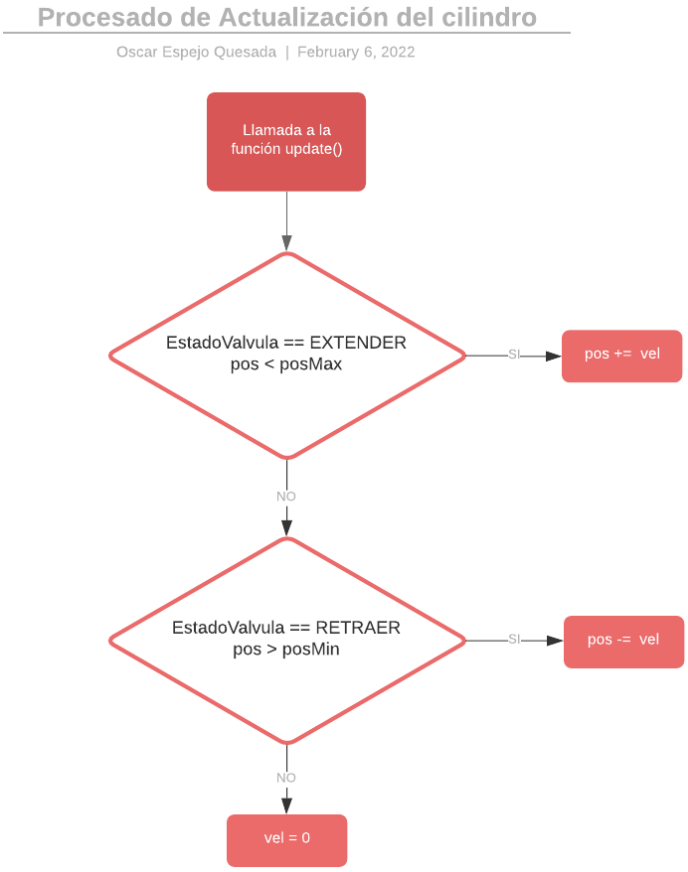


Figura 12. Proceso de actualización de posición de un cilindro

2.2.2.5 ElectroVálvula

La clase “ElectroValvula” se considera como un objeto, por lo tanto, hereda de dicha clase. La funcionalidad de dicha clase se basa en la gestión lógica de las salidas procedentes del PLC asociado al gemelo digital. Una vez que las señales son recibidas, se encarga de gestionarlas para dar la orden de expansión o retracción. La clase “Cilindro” implementa un atributo “Electroválvula” que gestiona su posición. Véase la Figura 10 donde se hace referencia a los atributos y métodos definidos.

- **Atributos**

El atributo “Estado” está basado en un enumerado que determina las diferentes posibilidades que se puede dar en un cilindro, la expansión o la retracción. Los valores de este enumerado serán recibidos por la clase cilindro para actuar.

Según las señales del PLC (Atributos de la clase “PlcOutput”) de las que disponga la electroválvula, se dan diferentes casuísticas que describen el

funcionamiento de los diferentes tipos de cilindros reales. Se puede dar el caso de ser monoestables, si la válvula es definida por una salida, y a su vez, pueden ser Normalmente Abiertos si dicha salida es la de retraer, o Normalmente Cerrados, si la salida asignada es la de expandir.

A su vez, se puede dar el caso ser asignadas dos salidas, una para extraer y otra para retraer. Este es el caso de una válvula biestable, que igualmente se pueden distinguir entre Normalmente Abiertas o Normalmente Cerradas dependiendo de la señal que esté activa por defecto.

Por último, el parámetro “posiciones” indica la cantidad de casos que se puede dar con la electroválvula. Si se indica el valor 2, se refiere a una válvula monoestable, ya que solamente puede realizar la acción de expandir o retraer. Asimismo, si es incluido el valor 3, se refiere a una biestable, ya que incorporaría el estado de estar parado en una posición intermedia.

- **Métodos**

El único método implementado en esta clase es el llamado “getEstado()”, cuya función es devolver el estado al cilindro que es reflejado por las salidas asignadas del PLC mediante un algoritmo de comprobación del estado de las señales. Reflejado el proceso de funcionamiento en la Figura 13.

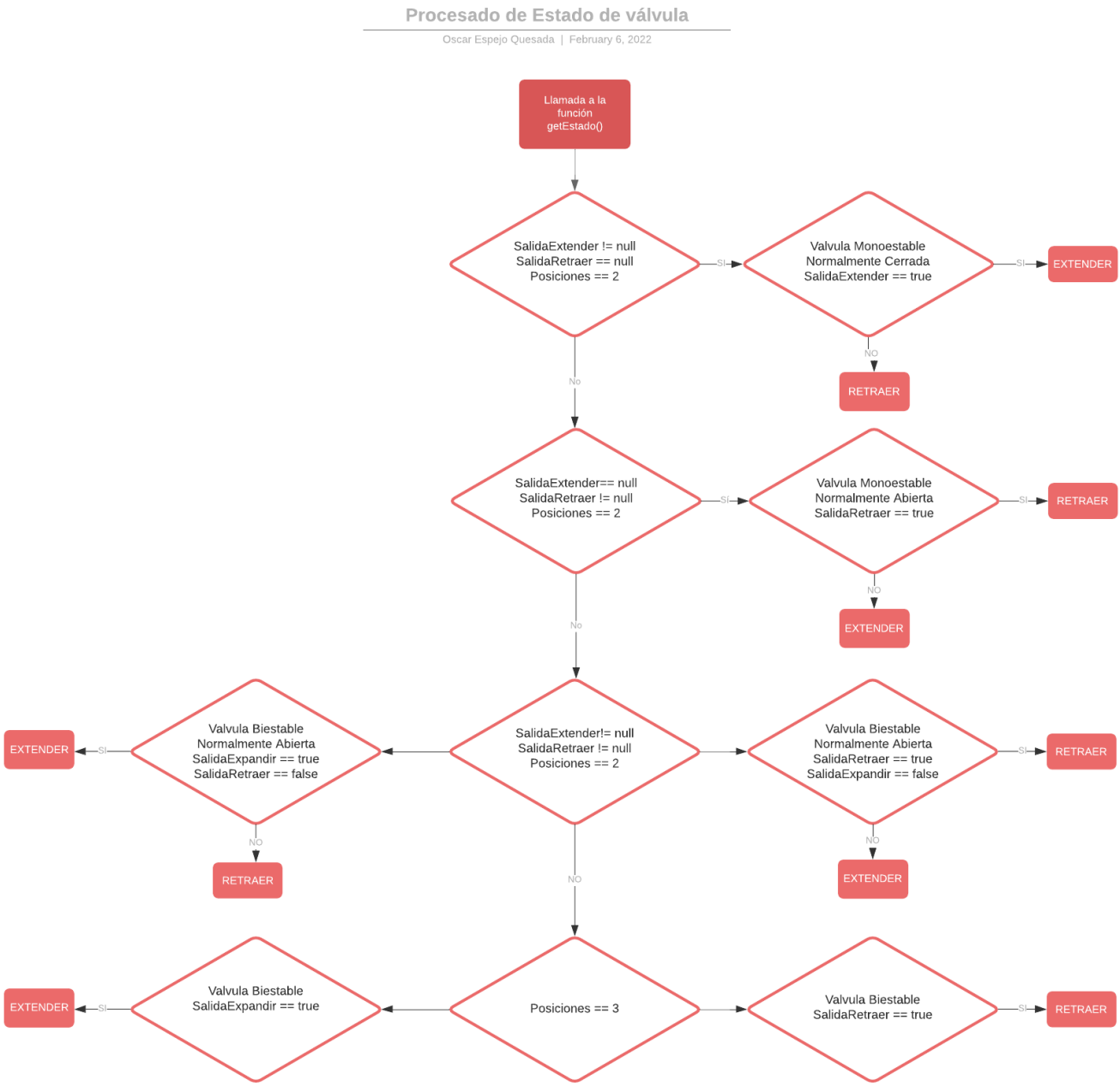


Figura 13. Proceso de cálculo de estado de una electroválvula

2.2.2.6 SensorDigital

Esta clase hereda directamente desde la clase “Objeto”, por lo tanto, se comporta como tal. Su función principal es actuar como finales de carrera en el movimiento de expansión o retracción de un actuador y dando lugar a la activación de la entrada correspondiente al PLC cuando se cumplen las condiciones indicadas.

- **Atributos**

Para simular la funcionalidad de un sensor físico, se incluyen los parámetros que tendría realmente (Véase la Figura 10), tales como el atributo “estado”, que consiste en una variable booleana que puede estar activada o desactivada según el estado del sensor. Asimismo, el atributo “entrada” define la dirección de la entrada al PLC que tendría en el conexionado de la estación. Y, por último, el intervalo en el que el sensor estará activo se define a través de las variables “posMin” y “posMax”.

- **Métodos**

El único método utilizado en esta clase tiene la función de comprobar la posición del actuador móvil y compararla con los valores máximos y mínimos, si el valor de la posición se encuentra dentro del intervalo, el sensor estará activo y, por tanto, la entrada al PLC asociada también. Tal como ilustra la Figura 14.

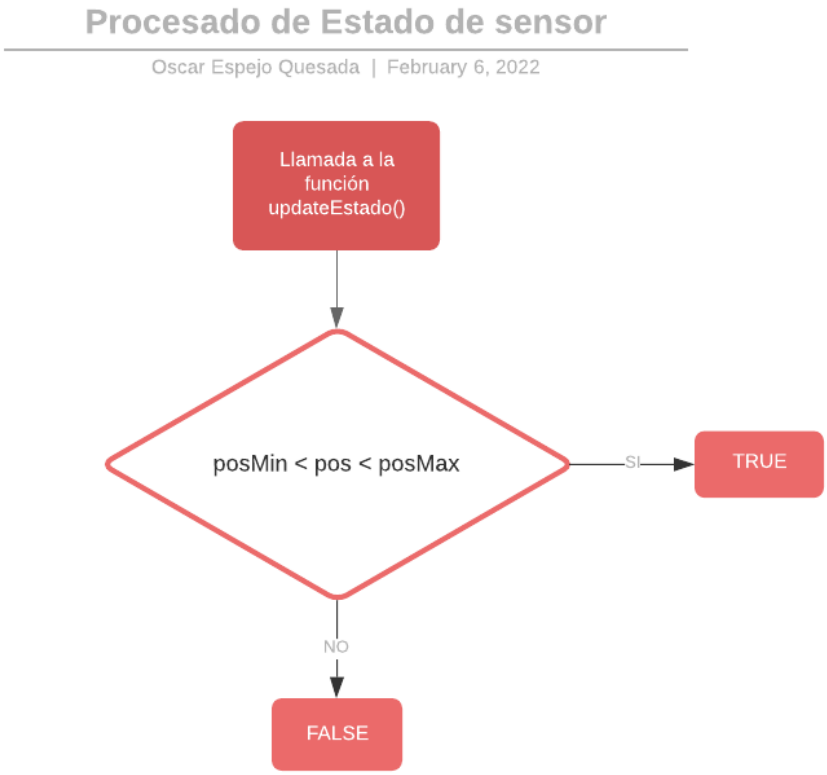


Figura 14. Procesado del estado de un sensor

2.2.2.7 ActuadorElectrico

"ActuadorElectrico" es una nueva clase creada con la intención de simular el caso del actuador eléctrico lineal, cuál no se puede desarrollar en el escenario como un cilindro común como se ha hecho hasta ahora. Por lo tanto, ha sido necesario crear la funcionalidad de dicho objeto.

Esta clase, al igual que la clase "Cilindro", hereda de "Actuador". Por lo tanto, seguirá la misma filosofía a la hora de ser representado, tendrá una base fija y una parte móvil que será la que realice el movimiento. Sin embargo, el modo de operación es algo más complejo, será necesario realizar un procedimiento de activación de señales conectadas directamente al PLC para que el dispositivo actúe.

- **Atributos**

En esta clase han sido necesarios cuatro atributos que describen el perfecto funcionamiento de la misma (Véase la Figura 10). Para ello, se dispone de dos vectores que van a contener las señales que le dan a dar movilidad al actuador, los

cuales son “salidasActivación” y “salidasPosicion”. Cada posición del vector va a contener una de las señales que manda el PLC a la controladora del robot.

En particular, en “salidasActivación” se puede encontrar las señales que activan el servomotor del robot (SERVO ON), el cual debe estar siempre activo para poder hacer efectiva la rotación del motor. Seguidamente, también incluye la señal de reseteo de la controladora en el caso de que entre en estado de alarma por algún fallo mecánico (RESET SERVO). Por último, se encuentra la señal que actúa como activación del movimiento (COMENZAR ROTACIÓN).

- **Métodos**

Al igual que en las otras clases, se repite el uso de las funciones del entorno de Processing llamadas “setup()” y “display()”. Dentro de esta última función se gestiona la respuesta del actuador al PLC manifestada por la señal llamada “Driver Ready” que es activada cuando el motor se enciende.

Para la simulación del proceso de actuación del controlador eléctrico se han desarrollado tres métodos, los cuales siguen una filosofía parecida a la de la clase “Cilindro” pero adaptada en hardware y software.

En primer lugar, el método “procesaPosicion()” que es primero que es llamado en el “display()” para calcular a qué posición ha de moverse el eje cartesiano a partir de las tres señales de codificación que manda el PLC. Este método es ejecutado cuando el gemelo digital detecta las señales de activación adecuadas. Tan como se recoge en la Figura 15.

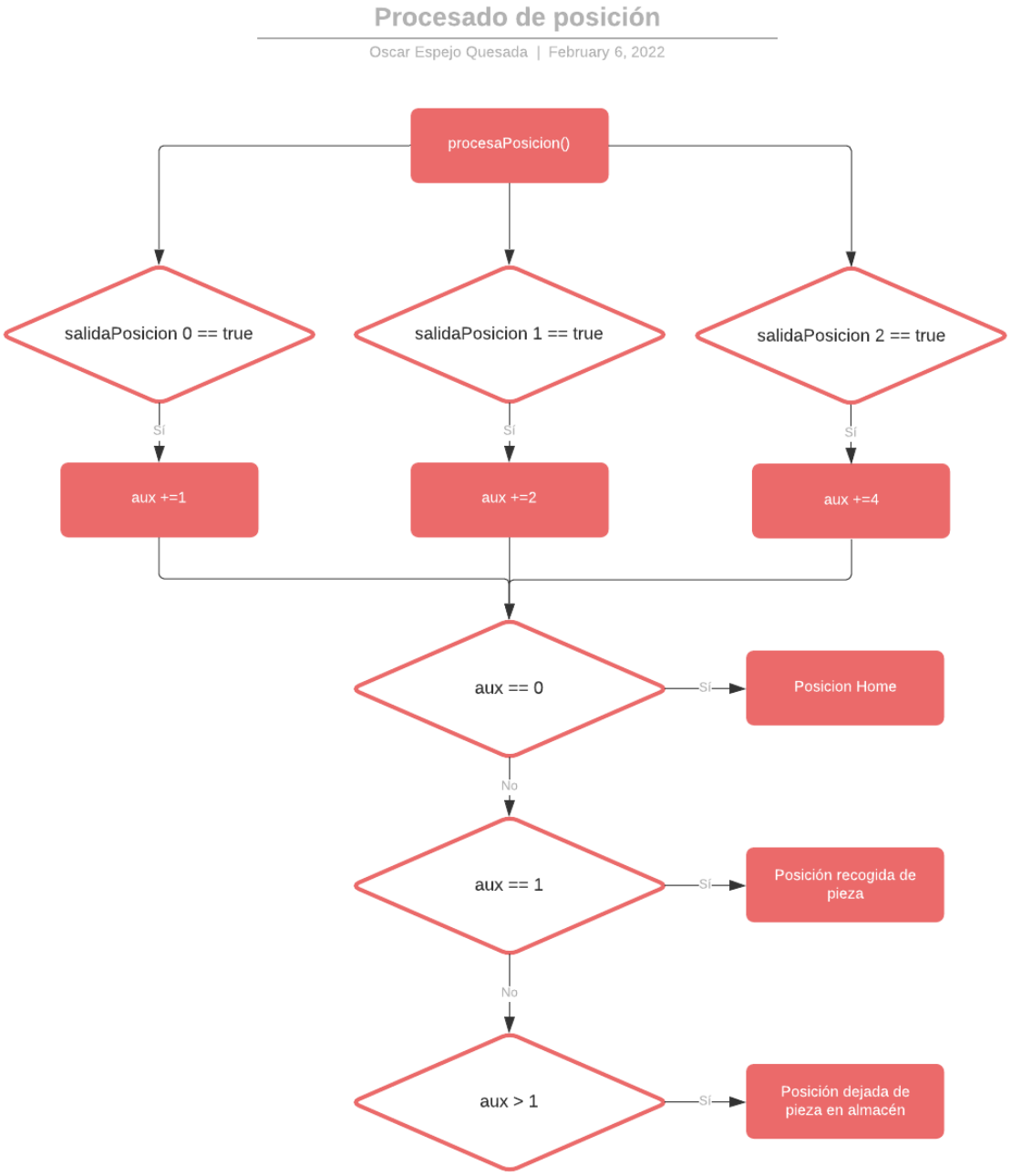


Figura 15. Procesado de posición

Para calcular a la posición a la que cada servomotor debe desplazarse, se utiliza una variable auxiliar para cada uno de ellos que se incrementa en función de la señal de codificación de posición que se reciba. Al funcionar en código binario, la señal 0 tendrá valor 1, la señal 1 tendrá valor 2 y la señal 2 tendrá valor 4. Véase un ejemplo de transformación de código binario a decimal en la Figura 16.

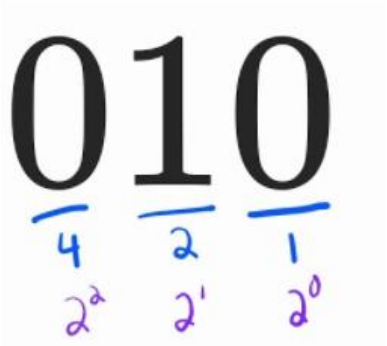


Figura 16. Transformación de código binario a decimal

En función del valor que tenga la variable auxiliar se recalculará la posición máxima de expansión del actuador eléctrico y de sus sensores correspondientes simulando así el desplazamiento del actuador real.

Seguidamente, otra función que es llamada en el “display()” es el “Update()”, la cual sigue la misma filosofía que la de clase “Cilindro”, la de actualizar la posición de la parte móvil del objeto hasta que sea alcanzado el objetivo tal y como recoge la Figura 17.

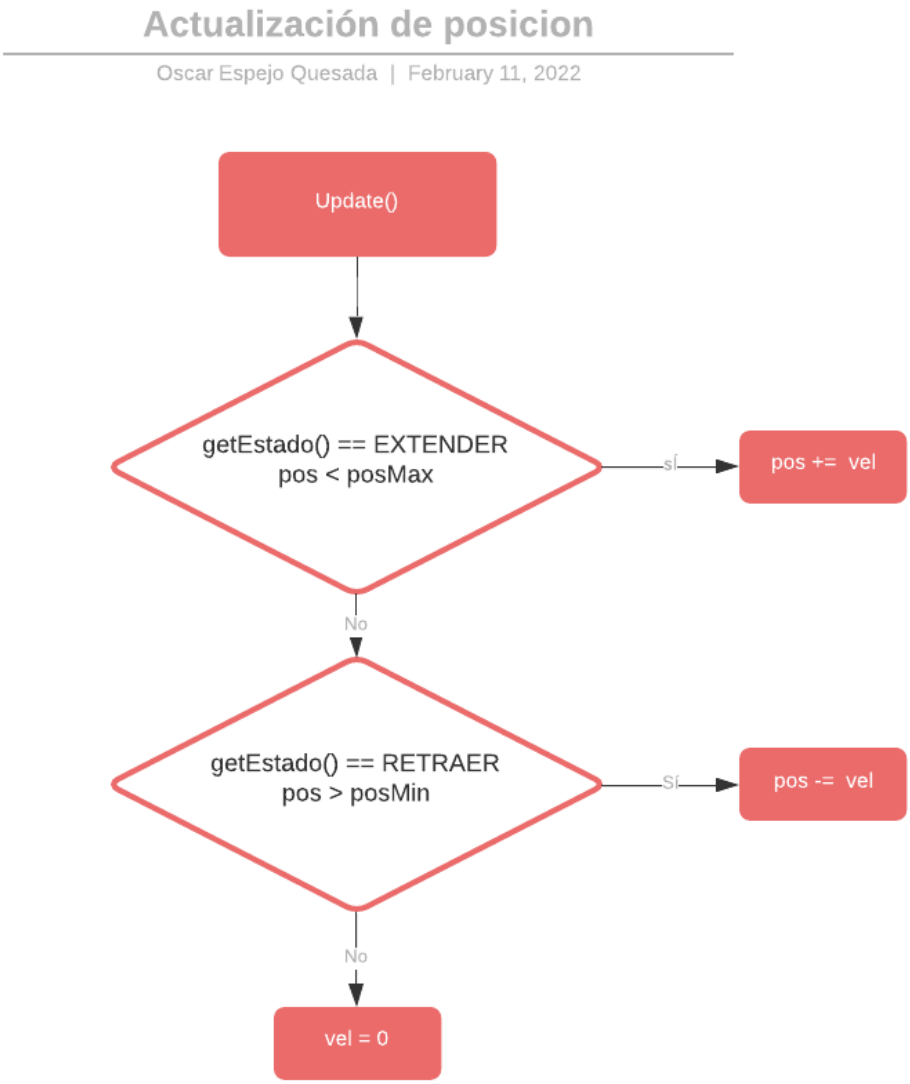


Figura 17. Proceso de actualización de la posición

El proceso es igual que el caso del objeto tipo “Cilindro”, si se obtiene un estado “EXTENDER” y la posición actual es menor que la posición máxima, la posición irá incrementando en cada ciclo un determinado valor marcado por el parámetro “vel” que es heredado de la clase “Actuador”. Además, en este método entra en juego el siguiente método utilizado, llamado “getEstado()”. Este se encarga de dar la orden de extender, retraer o estar parado dependiendo de las señales que reciba desde el PLC. El proceso se define en la Figura 18.

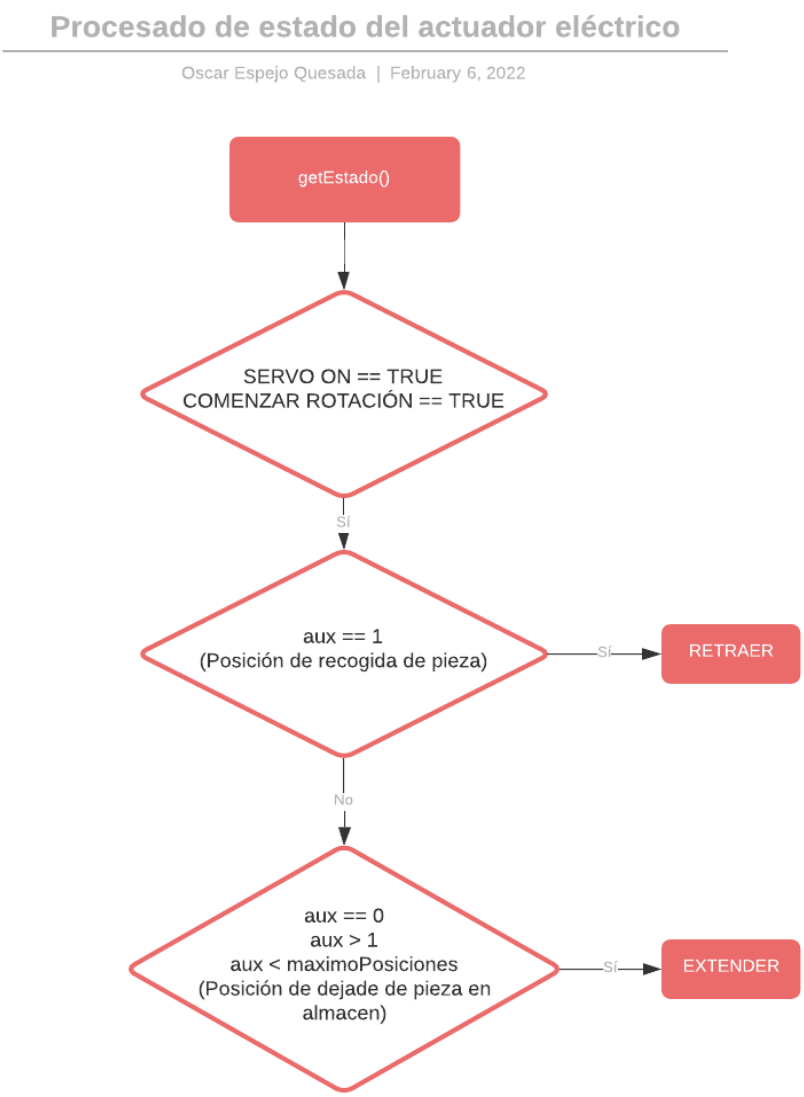


Figura 18. Procesado del estado del actuador eléctrico

En este caso, se comprueba que las salidas del PLC que marcan que el motor está activo y que se ha dado comienzo a la rotación del eje estén activas. Y una vez que se ha procesado la posición, el método decide si da la orden de expandir o retraer en función del valor de la variable auxiliar. En el caso que valga 1, significa que se tiene que desplazar a la posición de cogida de pieza en la cinta transportadora, por lo tanto, se considera un retraimiento, mientras que, si es el valor 0, debe hacer un home, que se sitúa en el máximo de cada eje por lo tanto debe devolver la orden de expandir a la posición máxima. Al igual que si es una de las posiciones del almacén, se considera una expansión.

2.2.3 Procesado XML

Una de las aportaciones que han sido incluidas en la serie de proyectos de gemelos digitales es el procesado de archivos de extensión XML para realizar la importación de datos de objetos pertenecientes a la escena, mejorando así, las versiones anteriormente creadas que habían sido basadas en archivos YML.

2.2.3.1 Introducción a XML

Los archivos XML son comúnmente utilizados principalmente para la transmisión de datos entre diferentes aplicaciones. Su estructura está basada en “elementos”, “etiquetas” y “atributos”.

Un elemento consiste en un bloque de construcción de un archivo XML. Es un componente que o bien comienza por una etiqueta de apertura y termina con una de cierre o que consiste en una única etiqueta. El contenido de un elemento es todo lo que se encuentra entre las etiquetas de apertura y cierre, incluso si estos son también elementos en cuyo caso se llaman elementos hijos (Mclibre, 2018).

Una etiqueta se encuentra delimitado por un carácter de apertura (“<”) y otro de cierre (“>”) y el contenido del mismo se encuentra entre ambas etiquetas. Véase en la Figura 19.

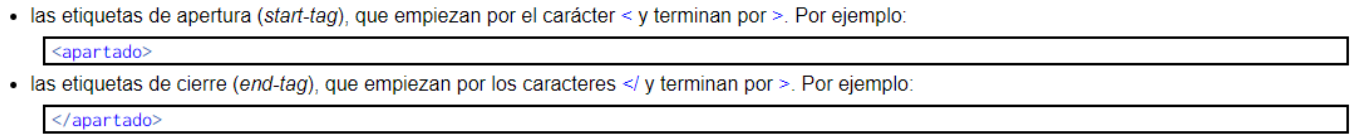


Figura 19. Ejemplo de etiqueta

Un atributo es componente de la etiqueta que consiste en una pareja nombre-valor. Se pueden encontrar en etiquetas de apertura, pero nunca en las de cierre. En una etiqueta no puede haber dos atributos con el mismo nombre. Reflejada su sintaxis en la Figura 20.

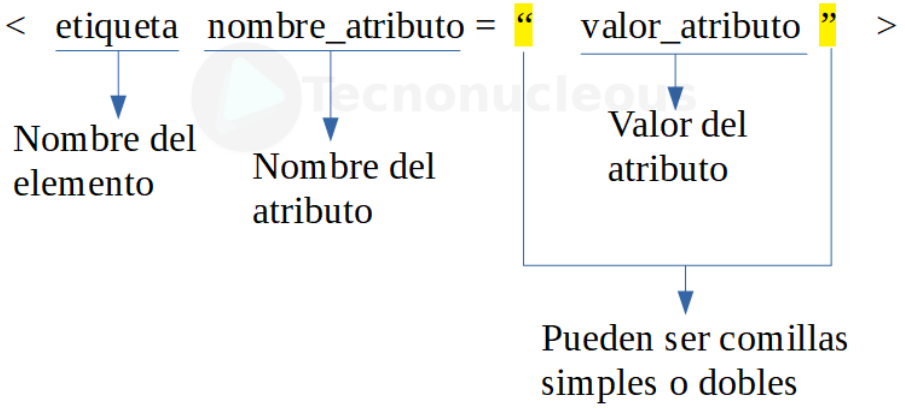


Figura 20. Estructura de un atributo

Existe una determinada forma de desarrollar un documento XML utilizando las etiquetas de apertura y cierre, y entre ellas, se indica su contenido en formato texto. Un elemento que no tiene contenido se denomina como elemento vacío. Para su utilización es necesario cumplir dos normas (Tutorialspoint, s.f.).

- **Norma 1**

Las etiquetas son sensibles a mayúsculas y minúsculas, por lo tanto, tanto en la etiqueta de apertura como en la de cierre se debe incluir el mismo texto como indica la Figura 21.

```
<address>This is wrong syntax</Address>
<address>This is correct syntax</address>
```

Figura 21. Ejemplo de sintaxis

- **Norma 2**

Las etiquetas deberán cerrarse en un orden apropiado, es decir, si existe una etiqueta dentro de un elemento, primero deberá cerrarse la etiqueta que esté situada más interior como se refleja en la Figura 22.

```
<outer_element>
   <internal_element>
      This tag is closed before the outer_element
   </internal_element>
</outer_element>
```

Figura 22. Ejemplo de orden de cierre de etiquetas

2.2.3.2 Ventajas y desventajas de utilizar XML

Como ventajas.

- Son una fácil forma de almacenar y leer datos por todos los navegadores y lenguajes de forma universal, ya que no pertenece a ninguna compañía (Fernandez, 2018).
- Se utilizan principalmente para exportar e importar bases de datos desde cualquier procedencia y únicamente basado en texto, por lo que, es un formato adecuado para este tipo de proyecto (Vaquero, s.f.).
- Ofrece la información de forma muy estructurada y ordenada, lo que conlleva que sea un lenguaje simplificado, fácil de leer y adaptado a cualquier aplicación y plataforma. Su formato es ideal para presentar datos porque tiene un fácil entendimiento debido a su estructura jerárquica y tabulaciones.
- Para ser utilizado en este proyecto es mucho más útil que el formato YML, ya que ofrece la oportunidad de recoger todos los elementos y parámetros que se deben ser incluidos en el escenario en un solo archivo, al contrario que dicho formato, que incluía un fichero por objeto representado. Por lo tanto, no genera incertidumbre a la hora de controlar diferentes archivos y se puede acceder más fácilmente. También da lugar a un menor peso de memoria en el sistema.
- Al contrario que YML, dichos archivos pueden ser abiertos por programas de procesado de bases de datos como XMLPad, que ofrece una interfaz gráfica para el trato de datos de este estilo.

Como desventajas.

- Normalmente, suele ser algo más costoso empezar desde cero a elaborar un archivo de este tipo ya que requiere una estructura muy específica. Sin embargo, conforme se avance en la realización será mucho más sencillo.
- Suelen ser archivos más grandes y pesados, lo que puede resultar que el tiempo de procesamiento sea algo más elevado que otros si es un archivo muy extenso.

2.2.3.3 Procesado de XML en el proyecto

Una vez elaborado el archivo XML donde se deben recoger todos los elementos que sean incorporados al gemelo digital se procede a importar dichos datos al proyecto una vez que se ejecuta el programa. Este procesado se realizará a través de la clase “Escena”, creada especialmente con el objetivo de desarrollar esa función. En ella se incluyen determinados métodos que son encargados de importar el archivo, descomponerlo en los diferentes objetos que lo conforman y procesar dichos datos de forma independiente. Véase los atributos y métodos definidos en esta clase en la Figura 10.

- **Atributos**

Esta clase “Escena” contiene una serie de atributos que han sido utilizados para realizar la importación de objetos que previamente han sido definidos en el fichero XML asociado. Tiene capacidad para acceder a cada uno de los objetos declarados y a sus elementos hijos asociados, por lo que se puede considerar la clase administrador del gemelo digital. Aunque disponga de acceso, cada clase controla sus elementos de forma independiente.

En primer lugar, los atributos “ancho” y “alto” definen la resolución que tendrá la ventana emergente que mostrará el escenario del proyecto. Según el dispositivo del que se disponga, puede ser útil bajar la resolución si la pantalla no soporta la predefinida por defecto.

La variable “pos” define la posición de origen del gemelo digital. Posteriormente, se podrá ajustar la vista al ejecutar el programa gracias a las funciones añadidas en la clase “GemeloDigital” para interactuar con la representación gráfica. Véase el apartado **Gemelo Digital**.

Los atributos “plc” y “root” hacen referencia a las clases “PlcSiemens” y “GemeloDigital” respectivamente, generando una unión entre ambas.

Por último, se incluyen los arrays de todos los objetos que intervienen en la simulación (objetos, válvulas, sensores y pulsadores).

- **Métodos**

Al igual que todas las demás clases, esta también incluye sus propios métodos de representación “setup()” y “display()”.

El método “loadEscena()” será utilizado para cargar el archivo XML en el proyecto, para ello se le debe suministrar la ruta en la que se encuentra. Una vez cargado el archivo, se descompone en los diferentes “nodos” que posteriormente se procesarán en el método “procesarNodo()” para convertirlos en objetos de la simulación.

En dicha función, se lee cada uno de los nodos, que corresponden a cada uno de los elementos del archivo XML. Están identificados gracias a la etiqueta que los limita, que al ser leído se identificará a qué tipo de objeto se refiere y se llamará al

método correspondiente para que lo procese. Véase la estructura de la función de procesado en la Figura 23.

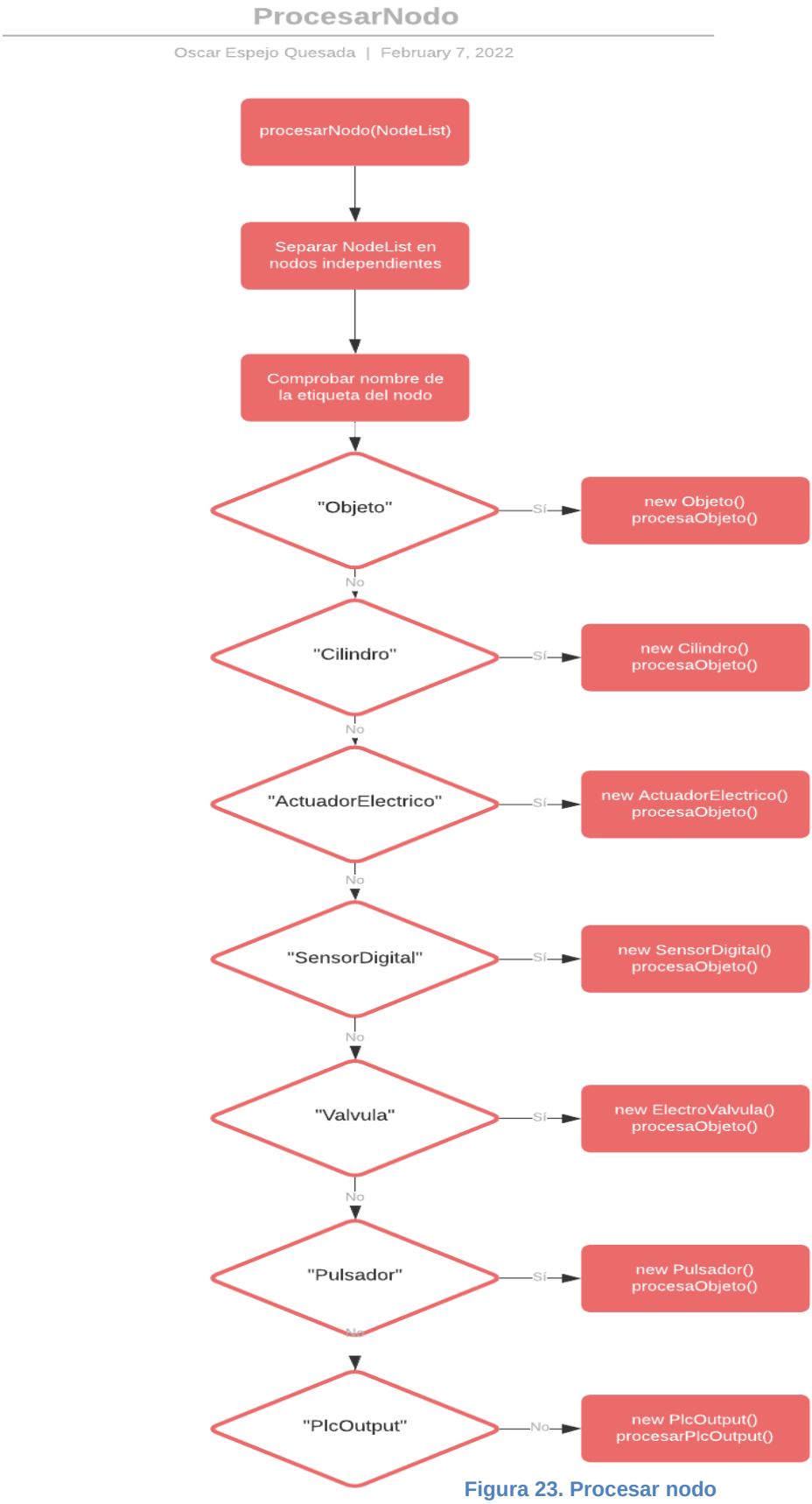


Figura 23. Procesar nodo

Al ser “Objeto” la clase padre casi la totalidad del resto, se tratarán como tales a todos los elementos que hereden de él, por tanto, serán procesados primeramente por esta función y posteriormente, se derivará a la función concreta de cada clase en el caso de que sea necesario. Se pueden observar los posibles métodos a los que puede ser derivado el objeto en la Figura 23. Este es el caso solamente de “ElectroValvula”, “model3d”, “PlcOutput” y “PlcInput”.

Por ejemplo, al leer una etiqueta que contenga el nombre “Valvula”, se ha considerado la herencia entre la clase “Objeto” y la “ElectroValvula”, por lo tanto, en primer lugar se procesará por el método procesaObjeto(), que será el encargado de asignar los atributos generales que comparten todos los objetos, y seguidamente, se llamará a la función “procesaValvula()” que será el encargado de asignar a los atributos especiales que no sean compartidos con la clase padre. Véase un ejemplo de código donde se crea un objeto “ElectroValvula” que será tratado como un objeto en la Figura 24. Esta estructura es la plantilla base de la cual se partirá a la hora de generar objetos tipo “ElectroValvula”.

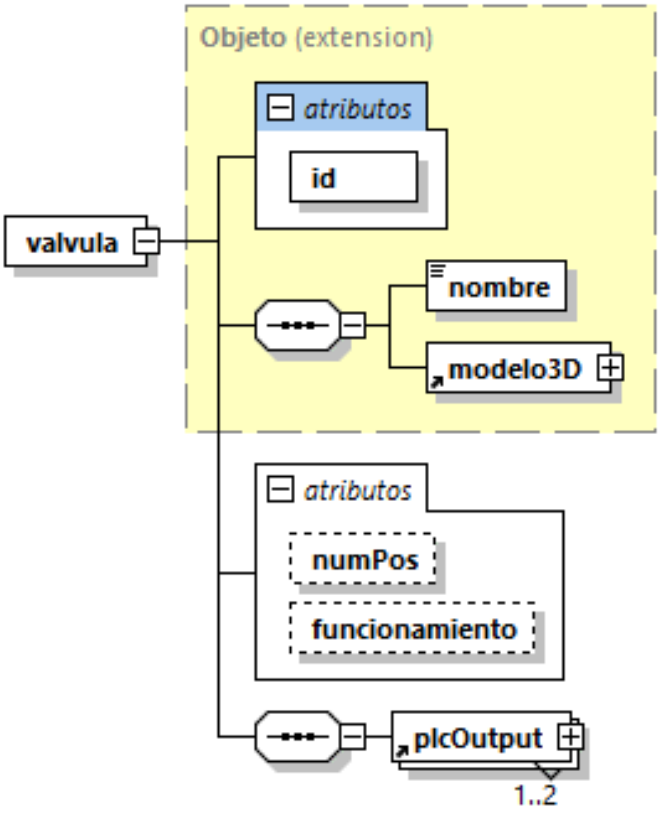


Figura 24. Plantilla de ejemplo para crear una válvula

A continuación, se describirá el método clave para llevar a cabo el desarrollo de la importación de datos desde un archivo XML. Dicho método es el “procesaObjeto()”. Su funcionamiento sigue una lógica determinada, primero recibe un elemento recibido entre dos etiquetas determinado como nodo como se ha indicado en la función anterior. Seguidamente, divide ese nodo en subnodos, que corresponderán con los parámetros que tendrá el objeto, y comprueba cada una de las etiquetas de los subnodos. Serán procesados uno a uno, e irá llamando a las funciones que correspondan según el valor que contenga la etiqueta.

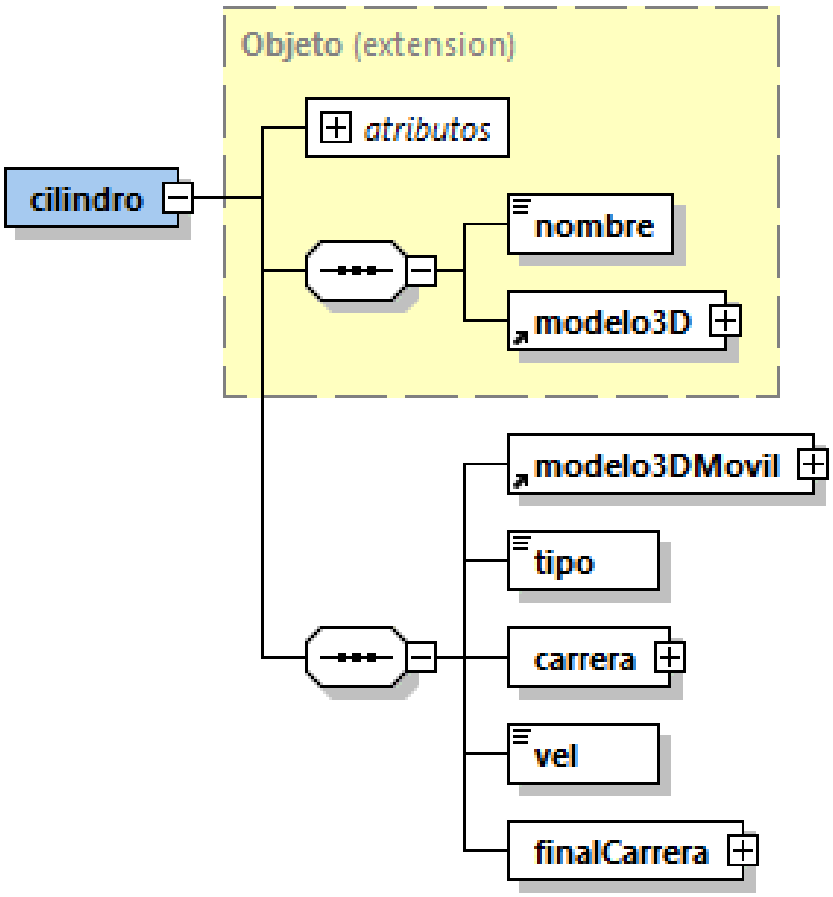


Figura 25. Plantilla ejemplo de creación de objeto

Recogida la referencia en la Figura 25 en XML, se da el caso de un nodo llamado “Cilindro”. Su contenido estaría delimitado entre las etiquetas de apertura y de cierre llamadas <element> y </element>, que posteriormente se llamarán <Cilindro> y </Cilindro>, por lo tanto, cada elemento que haya en su interior corresponde a un atributo, en este caso se incluyen los atributos de nombre y modelo3D propios de la clase “Objeto”, y además otros como “modelo3DMovil”, “tipo”, “carrera”, “vel” y

“finalCarrera”. Cada atributo se considerará un subnodo que será procesado según el nombre de la etiqueta que contenga. Por ejemplo, la etiqueta “<modelo3d>” y “<modelo3dMovil>” serán determinadas por “procesaModelo3d()”, o el caso de la etiqueta “<pulsador>” será por “procesaPulsador()”. Además, cuando se dé el caso en el que hay un objeto dentro del elemento que se está procesando se determinará como objeto hijo, como es este caso, que dentro del objeto “Cilindro” hay diferentes objetos como sensores o pulsadores. Véase el proceso de importación de la clase objeto en la Figura 26.

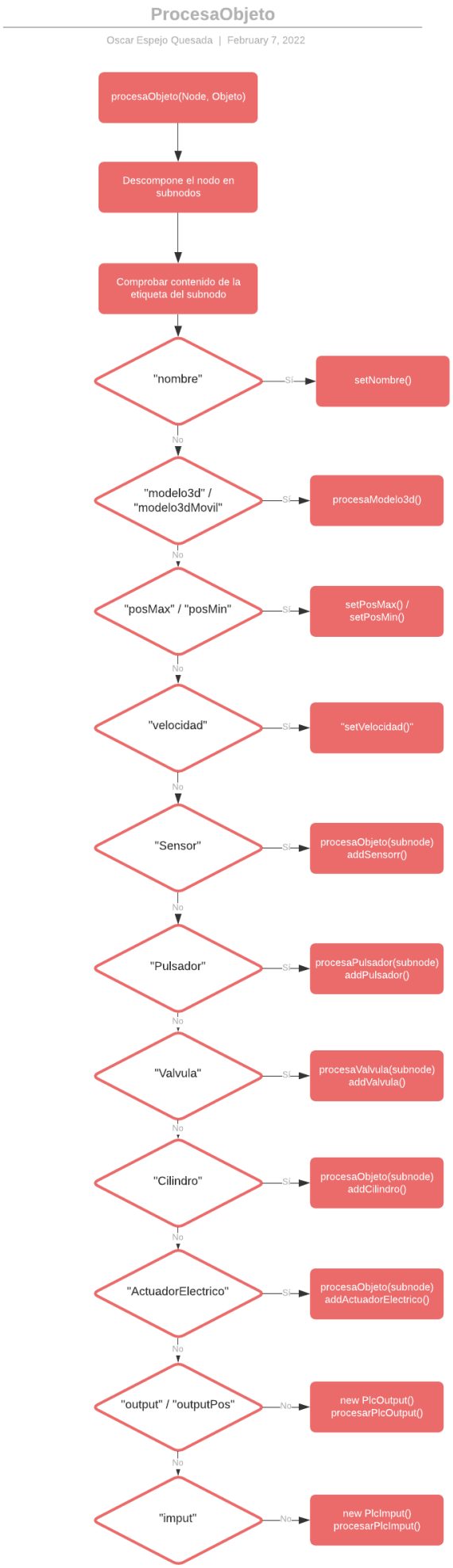


Figura 26. Procesador de Objeto

De este modo, gracias a este método se podrán recoger todos los parámetros que son generales para todas las clases (normalmente declarados en la clase “Objeto” y que comparten la mayoría de elementos implementados). En casos especiales en los que con solamente este método no pueda recoger todos los datos que se proporcionan, se solicitará la intervención de otro método dedicado especialmente para la clase que lo requiera. Dándose el caso para “Posicion”, “ElectroValvula”, “model3d”, “PlcOutput” y “PlcInput”.

Véase la Figura 27 donde es descrito el proceso que requiere un objeto tipo “ElectroValvula” para ser importado al sistema. Tiene la peculiaridad de tener la posibilidad de disponer de dos salidas del PLC llamadas “outputEx” y “outputRe”, haciendo referencia a la salida de expansión o a la de retracción respectivamente. Las cuales son utilizadas para poder pilotar la válvula. Además del número de posiciones que indica, conjunto al número de salidas asignadas, el tipo de válvula que se determina.

Los datos serán recogidos del fichero de texto indicado entre las etiquetas <valvula> y </valvula>.

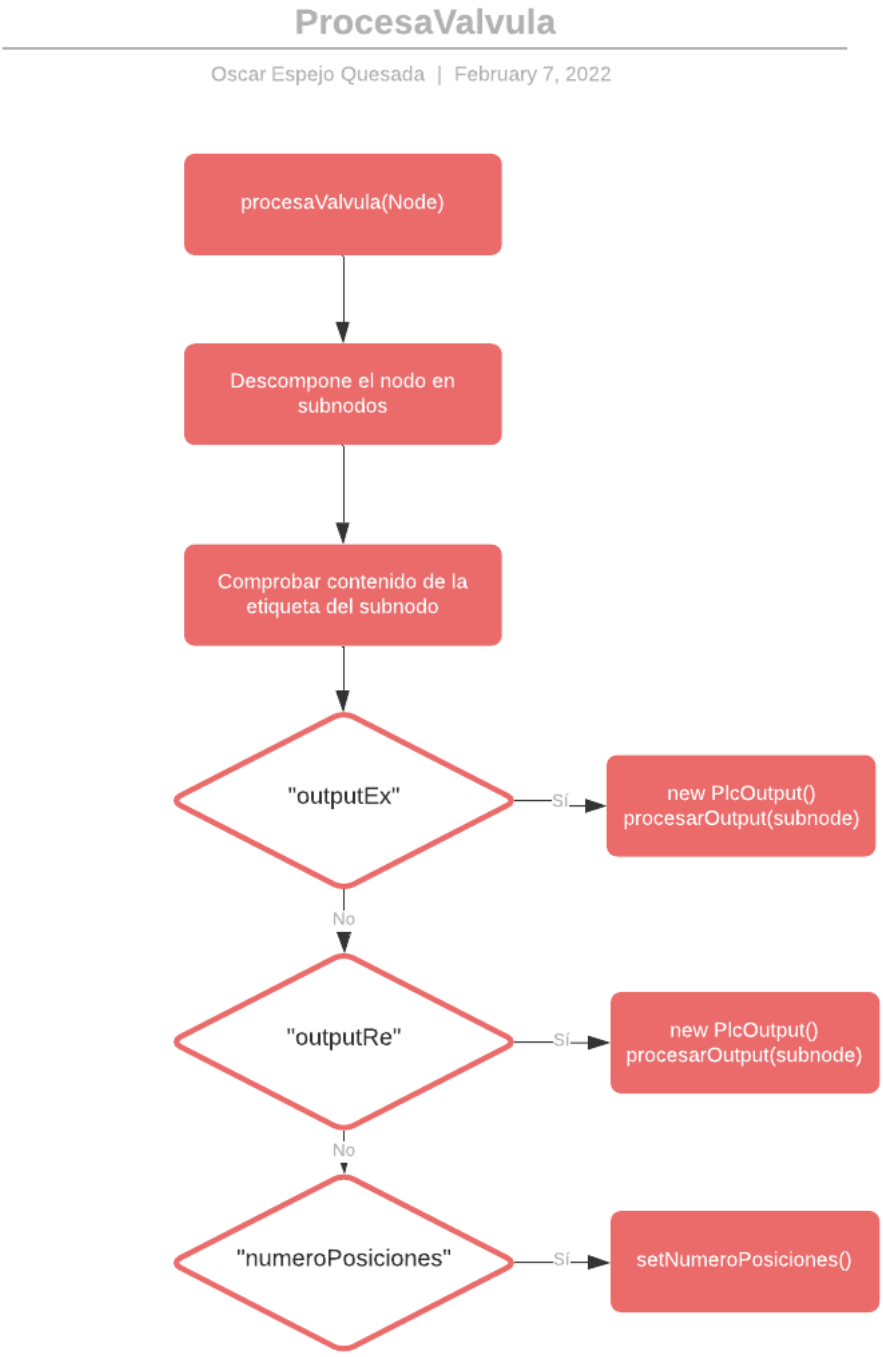


Figura 27. Procesar Válvula

Obsérvese en la Figura 28 el caso en el cual la etiqueta contenga el nombre “model3d” o “model3dMovil”, la cual se llamará al método definido para procesar modelos gráficos en el proyecto, determinado como “procesaModel3d()”. Es encargado de importar el nombre del archivo 3d que se ubica en la carpeta “config”

del proyecto, la posición a través de un método encargado de su procesado llamado “procesaPosicion()” y la escala a la que va a ser representado en la escena.

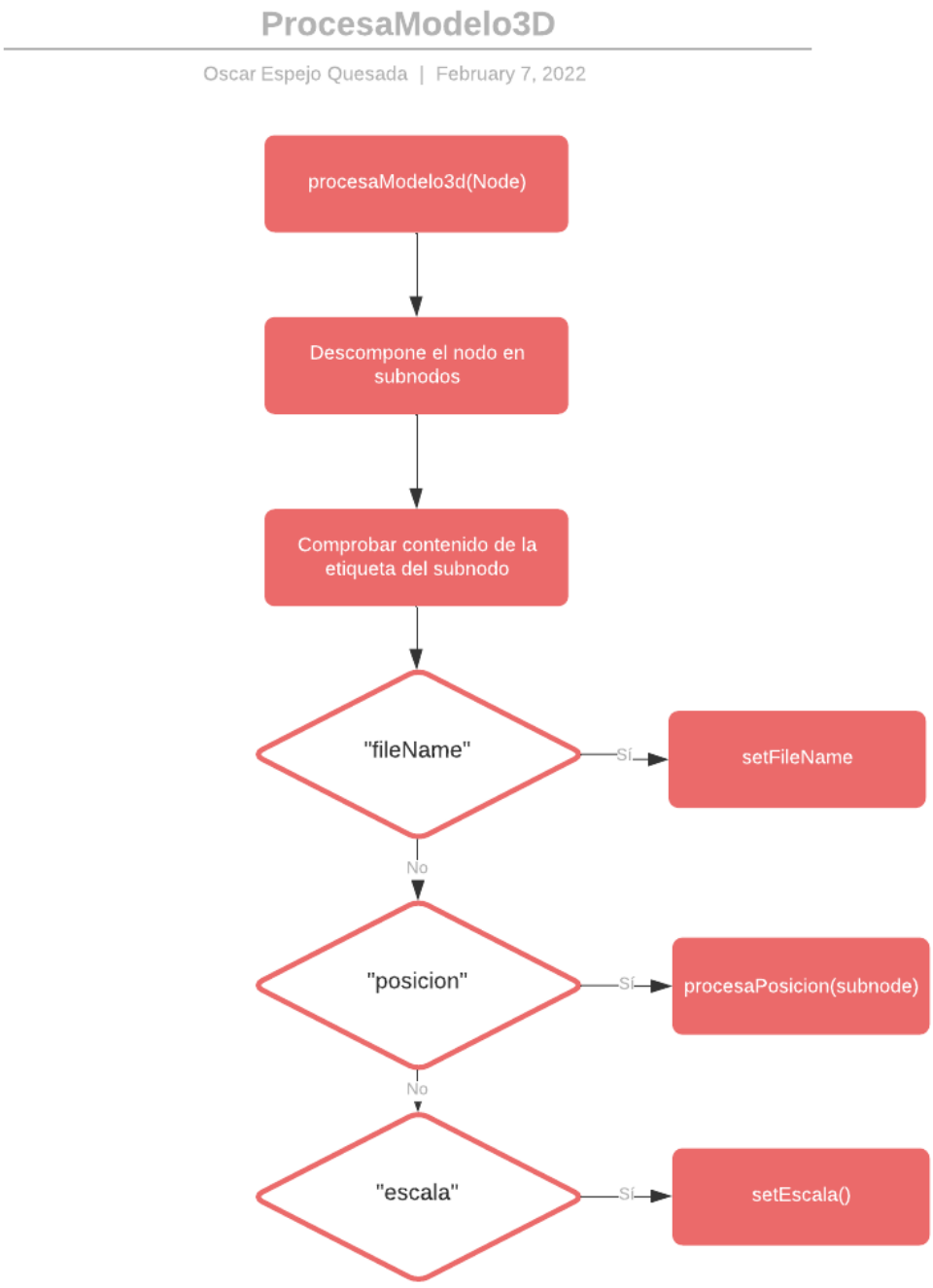


Figura 28. Procesar Modelo3D

Los datos serán recogidos del fichero de texto indicado entre las etiquetas con el nombre <modelo> y </modelo>. En la Figura 29 se puede observar un ejemplo de cómo habría que proceder para crear un parámetro de modelo 3D dentro de un objeto.

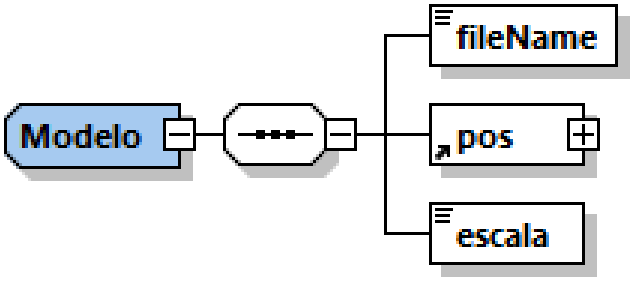


Figura 29. Plantilla ejemplo para crear Modelo3D

A continuación, se llevará a cabo el procesado de la clase “Posicion” ilustrado en la Figura 30. En dicha función se encarga de recoger todos los datos relacionados con la ubicación del origen de coordenadas del objeto que se va a representar. Está compuesto por tres coordenadas en los distintos ejes (x, y, z) y su rotación (rotación x, rotación y, rotación z).

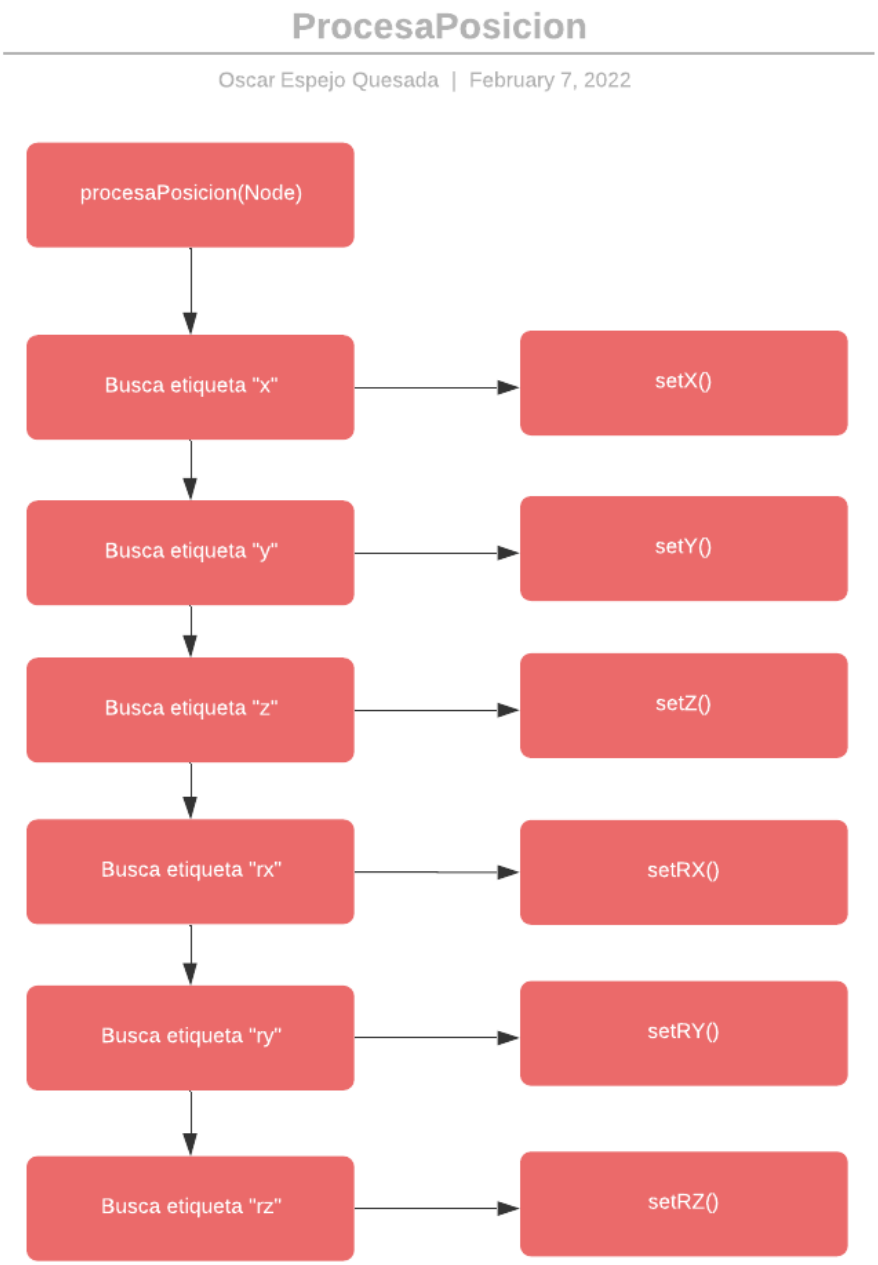


Figura 30. Procesar Posición

Los datos serán recogidos del fichero de texto indicado entre las etiquetas <pos> y </pos>. Véase la Figura 31 donde se observa qué elementos importa el procesado de un parámetro posición.

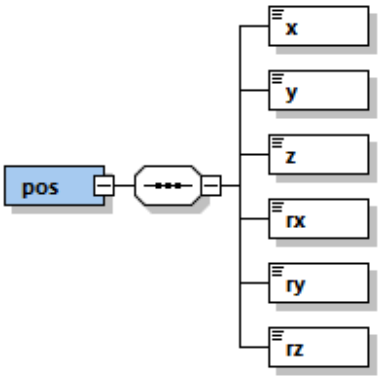


Figura 31. Plantilla ejemplo para crear un parámetro posición

Por último, para gestionar las entradas y salidas del PLC (plcOutput y plcInput) se ha desarrollado otro método que asigna los valores correspondientes para realizar una buena comunicación entre el gemelo digital y el controlador, debido a que los atributos leídos simulan un conexionado real de un circuito electrónico. Por eso mismo, se le atribuyen los parámetros del bit y el byte, que hacen referencia a la salida del controlador y se le asigna un nombre para poder identificarlo fácilmente en la lista de salidas y entradas. El procesado de la entrada sigue el mismo proceso que con la salida, con la única diferencia de que se incluye en el vector de entradas del PLC incluido en la Figura 32.

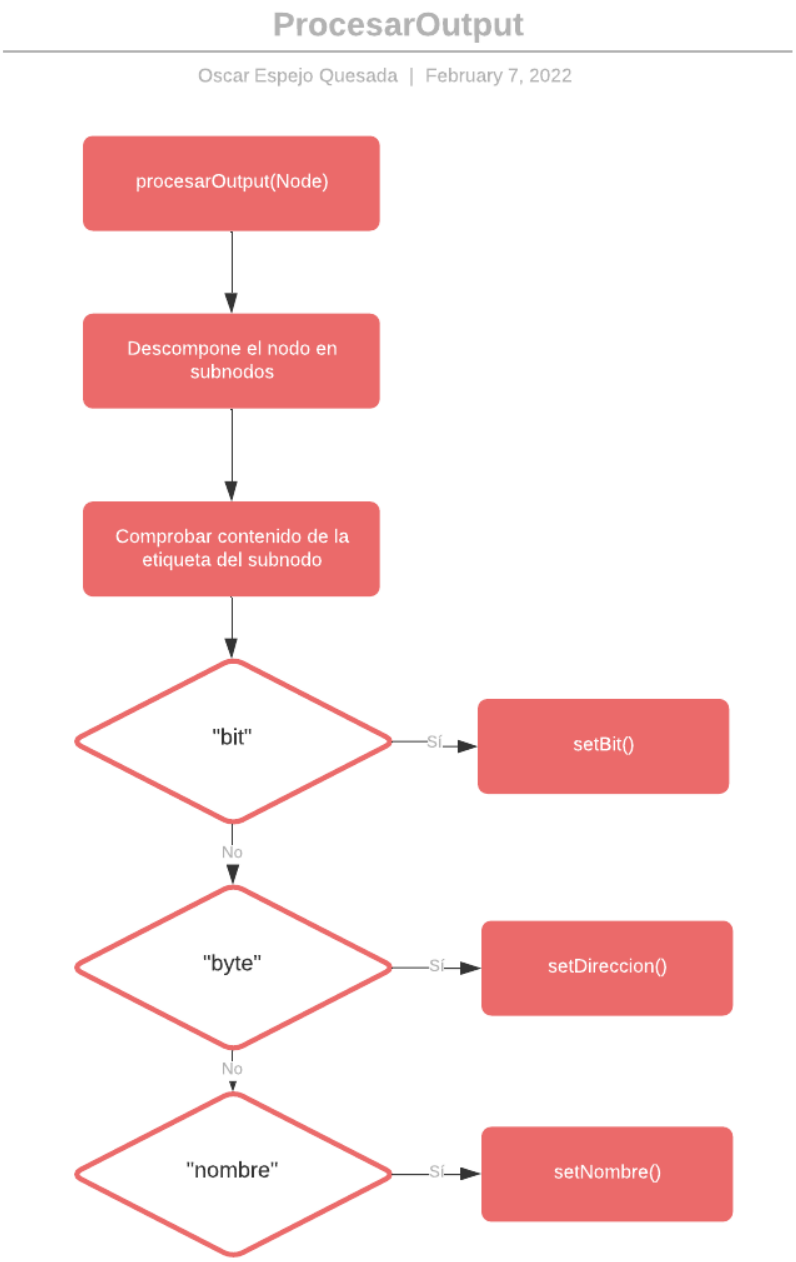


Figura 32. Procesar PlcOutput/PlcInput

Los datos serán recogidos del fichero de texto indicado entre las etiquetas <plcParam> y </plcParam>. Véase la Figura 33 para observar cómo será necesario crear una señal al PLC, ya sea entrada o salida indiferentemente.

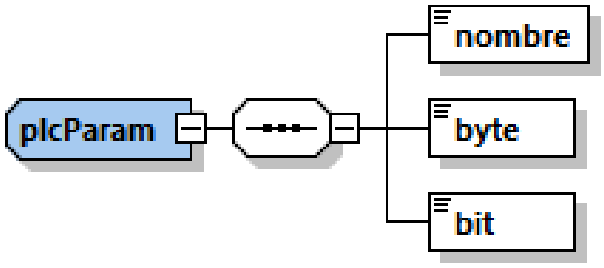


Figura 33. Plantilla ejemplo para crear parámetro de entrada o salida al PLC

2.2.4 Generación de un archivo ejecutable

Debido a la necesidad de que el gemelo digital sea una aplicación que pueda ser ejecutado en diferentes dispositivos, ha sido incluido un plugin en el archivo *pom.xml*, el cuál generará una carpeta llamada “dist” que incluye un archivo ejecutable (.Jar). Solamente sería necesario hacer una copia de la carpeta “config”, que contiene todos los archivos de modelado en tres dimensiones con extensión OBJ y de configuración de la escena del proyecto en la nueva carpeta generada. Véase en la Figura 34 como habría que proceder para añadir el plugin necesario en el archivo llamado “*pom.xml*” y véase en la Figura 35 el resultado de realizar el traspaso de la carpeta “config” y el archivo ejecutable generado gracias al plugin incluido.

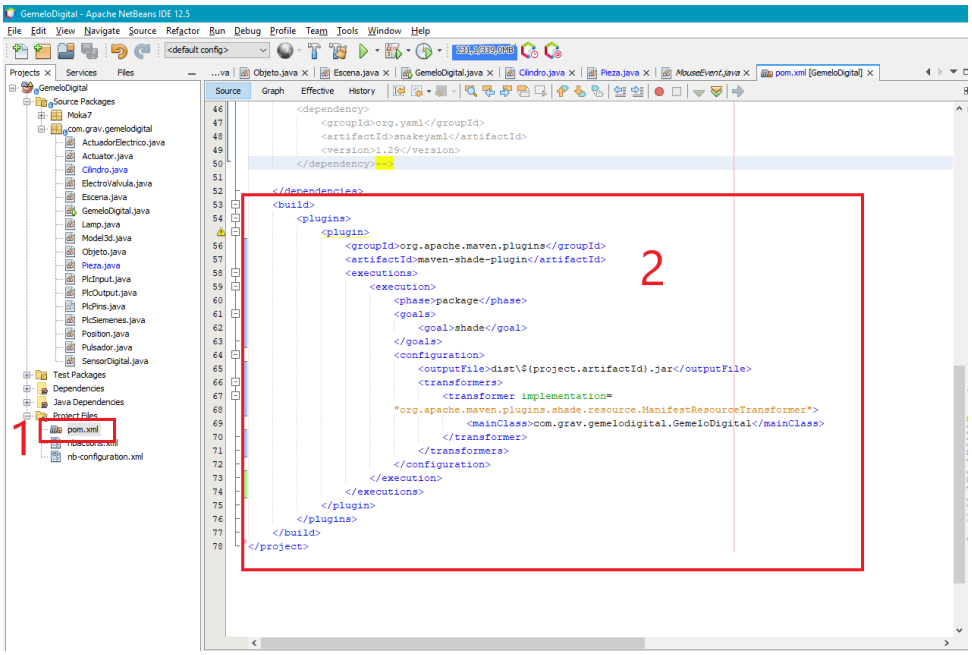


Figura 34. Adición de plugin en archivo pom.xml

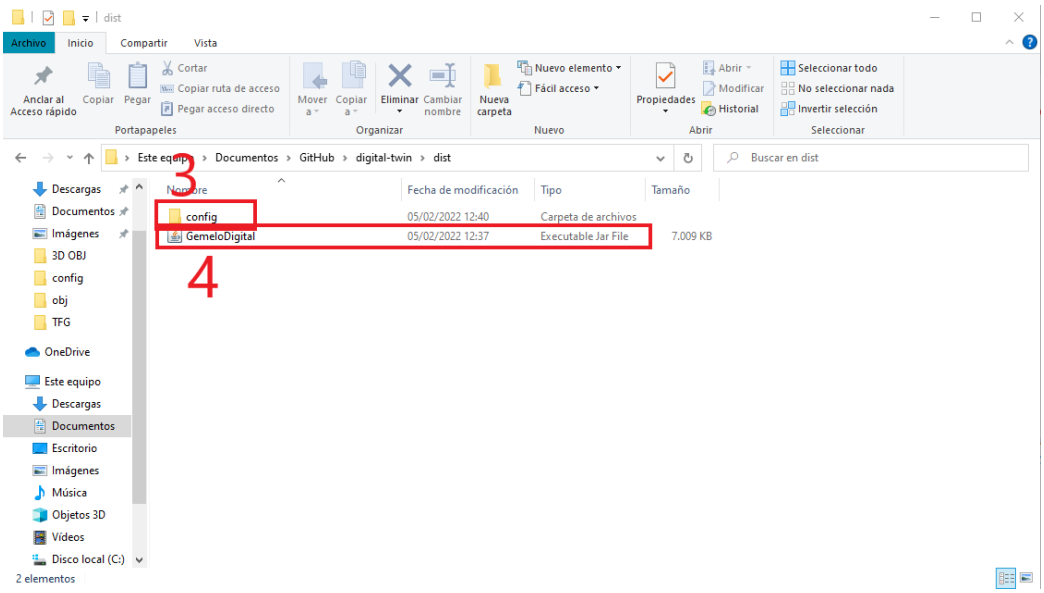


Figura 35. Adición de carpeta config y archivo ejecutable (.Jar)

2.3 TIA Portal V14

Siemens es una empresa fundada en Madrid en 1895 que se dedicó a investigar para introducir elementos innovadores y modernizar España para mejorar la calidad de vida de las personas (SIEMENS, SIEMENS, s.f.). Esta empresa está especializada en productos dedicados a la automatización (SIEMENS, SIEMENS Automatización, s.f.) y cuenta con su propio software para programar sus dispositivos. Dicho software de la empresa Siemens es en el que se ha llevado a cabo la programación del PLC,

es conocido como TIA Portal V14. Fue en el 2009 cuando la empresa Siemens, conocida en este país, introdujo al mercado una plataforma de uso muy intuitiva, para planificación de proyectos, programación y puesta en marcha de controles, dando la oportunidad de ofrecer una asistencia en las tareas de automatización. Este software supuso uno de los hechos más importantes en el desarrollo hacia la fábrica digital en la historia, ya que, debido a su potencial, supuso una gran oportunidad de desarrollo del sector de la automática industrial. En este mismo año, se lanzó una nueva generación de controladores, el "Simatic S7-1200 Basic Controller", el cual es utilizado en la gran mayoría de proyectos debido a su precio económico como a sus grandes aplicaciones.

Al ser un software oficial y privado, se tiene acceso a todos los productos del catálogo de Siemens. Su uso convence por su amplia funcionalidad y por ofrecer un entorno de ingeniería unificado que integra distintas aplicaciones software industrial en un mismo interfaz, que facilita enormemente el aprendizaje, la interconexión entre elementos y la operación. No importa si se trata de un PLC o un HMI, desde TIA Portal se puede gestionar el software de cualquier elemento hardware. Véase el icono del software en la Figura 36.

Figura 36. Icono TIA Portal V14

2.3.1 Conexión con PLCSIM y NetToPLCsim

Para llevar a cabo la comunicación entre el IDE de NetBeans y la programación de TIA Portal han sido necesarios realizar varios pasos mediante un simulador de PLC llamado PLCSIM y un creador de servidores para efectuar la conexión llamado NetToPLCsim. Véase el 6.3 que incluye un manual que detallan los pasos a seguir para realizar la conexión entre ambos programas.

El software PLCSIM está incluido en el paquete de TIA Portal, es muy útil a la hora de realizar una simulación de un proyecto cuando no se dispone de un PLC real ya que cumple la misma función. Una vez terminado el proyecto de automatización, se debe de cargar el software en el PLCSIM siguiendo el mismo protocolo que con un dispositivo físico.

Por otro lado, el NetToPLCsim cumple la función de enlace entre el simulador con el software ya incorporado y la ejecución de la representación 3D elaborada en NetBeans. Se encargará de enviar información entre ambas partes para que puedan interactuar entre ellas mediante la activación de las salidas por parte del simulador y la activación de entradas por parte de la representación.

3 Caso de uso

3.1 Hardware de la estación 8

Esta estación de la FMS200 contiene algunos de los elementos que son más comunes en las demás estaciones como pueden considerarse los cilindros neumáticos, sin embargo, tiene la peculiaridad de incorporar dos actuadores lineales para dotar de movimiento a la función de paletizado.

Su función a grandes rasgos es el almacenade de piezas que proceden de procesos de fabricación anteriores. Estas piezas llegan a la estación a través de una cinta transportadora que hace de nexo entre todas las demás, y a la cual, el robot se desplazará para recoger la pieza en cuestión. Posteriormente, una vez posicionada en la posición de inicio, actuará el cilindro extendiéndose para alcanzar la pieza, que por medio de vacío será capturada por las ventosas y se retraerá. Seguidamente, la controladora realiza los cálculos necesarios para saber la posición de dejada correcta en el almacén y se desplaza hasta ella. Una vez allí, el cilindro se expande y se desactiva el vacío para poder depositar la pieza. Se puede observar el diseño de la estación en la Figura 37.

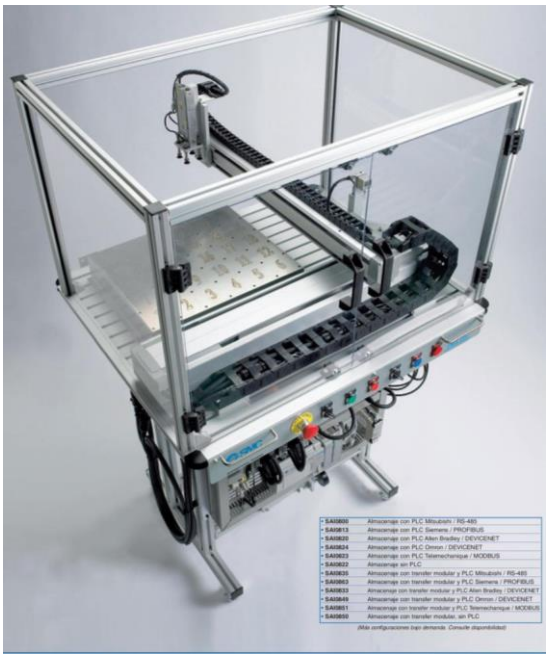


Figura 37. Estación 8 de la Célula de Fabricación Flexible FMS200

La estructura de esta estación está compuesta principalmente por los siguientes elementos mecánicos.

- **Actuadores eléctricos**

El principal elemento motor que interviene en esta estación son los actuadores eléctricos que incorpora para desplazarse entre las correspondientes posiciones de cogida y dejada de pieza. Contiene un motor rotativo que es el encargado de transmitir el movimiento a la parte móvil por medio de una correa. Este tipo de controlador es muy utilizado para estos tipos de servomotores, ya que ofrece un control exacto y sencillo del actuador. Además, no solo ofrece la posibilidad de ser controlado con el PLC, sino que esta marca cuenta con su propio software llamado MRZJW3-SETUP161E que dota de completa funcionalidad al equipo (Mitsubishi, s.f.)s. Se refleja en la Figura 38 una ilustración de la máquina físicamente.

Figura 38. Imagen del actuador eléctrico

La controladora tiene definidas cada una de las posiciones que intervienen en el proceso, que son determinadas a través de los bits de selección de programa de cada eje. Véase el ANEXO B: PLANOS ELÉCTRICOS. Por consiguiente, una vez que es conocida la codificación de cada posición, es importante tener en cuenta el protocolo de comunicación que debe seguirse para pilotar el actuador. Es importante seguir estrictamente ese orden, sino el robot entrará en estado de alarma y no permitirá su manipulación. Es un proceso difícil el de encontrar la secuencia que sigue el mismo de forma directa, para ello, se ha desarrollado una estructura de prueba y error a través de una serie de interruptores que al activarse cada uno de ellos simulan una salida del PLC real que controlan el robot. De esta forma, se puede ir probando las diferentes casuísticas de activación de señales hasta que se encuentra la adecuada fácilmente.

- **Cilindro neumático**

Esta estación incluye un cilindro SMC CXSM20-75 de simple efecto pilotado por una electroválvula conectada al sistema y conectado a la red de aire a presión. Su función en el sistema es bastante simple, expandir y retraer para coger o depositar la pieza según en la posición en la que se encuentre el eje cartesiano. La Figura 39 ilustra la pieza real.

Figura 39. Cilindro neumático con ventosa

- **Ventosas**

En esta estación se incluye unas ventosas situadas en el extremo móvil del cilindro conectadas a un sistema de vacío, cuya funcionalidad es la de activarse cuando el vástago se expande para así, poder absorber la pieza y desplazarla con el movimiento del actuador lineal hasta la posición de dejada. Una vez allí, se desactiva el vacío para depositarla en su posición de almacenaje. Al igual que el apartado anterior, se pueden observar las ventosas en la Figura 39. Cilindro neumático con ventosa.

- **PLC Siemens S7-1200**

Para lograr el objetivo de tener una correcta automatización de los procesos, la elección del hardware es fundamental a la hora de realizar el diseño. Se requiere un equipo que sea confiable y que permita cumplir las acciones requeridas.

En este caso se ha utilizado un PLC Siemens S7-1200, se trata de un dispositivo compacto comúnmente utilizado en la industria que proporciona una potente capacidad de cálculo (64 bits), con funciones integradas y fáciles de programar, pero con una gran precisión de actuación. También, incluye la posibilidad de establecer

comunicación a través de interfaz Ethernet con otros tipos de dispositivos como pueden ser pantallas HMI. Este PLC incorpora 14 entradas y 10 salidas que pueden ser directamente conexionadas en los puertos determinados. Mostrada la estructura del PLC en la Figura 40.

Figura 40. PLC Siemens S7-1200

- **Periférico SIMATIC ET 200SP**

En caso de que se requieran más cantidad, se podrá complementar con periféricos que recojan otras señales y lleguen al controlador principal que se encarga de gestionarlas. En el caso de esta estación, el conexionado está realizado en un periférico SIMATIC ET 200SP conectado al PLC. En la Figura 41 se refleja el dispositivo, en él tiene un módulo de 16 entradas DI y otro de 16 salidas DQ los cuales están conectados directamente a los elementos de la estación. Para realizar la conexión entre ambos se pueden conectar a través de un cable Ethernet y configurarlo en el listado de dispositivos simplemente añadiéndolo desde la biblioteca de Siemens y dándole una dirección IP.

Figura 41. SIMATIC ET 200SP

- **HMI SIMATIC Siemens KTP700 Basis**

Consiste en una pantalla táctil capaz de interactuar con la programación del sistema y permite visualizar de forma gráfica el funcionamiento del PLC. Véase laFigura 42. Dicho elemento lleva su programación independiente del programa principal, por lo tanto, puede actuar de forma independiente.

Figura 42. Pantalla HMI SIMATIC Siemens KTP700 Basis

- **Almacén**

Consiste en una simple bandeja numerada por posiciones en las que serán depositadas las piezas por el eje cartesiano de almacenaje. No incluye ninguna función automática como puede observarse en la Figura 43.

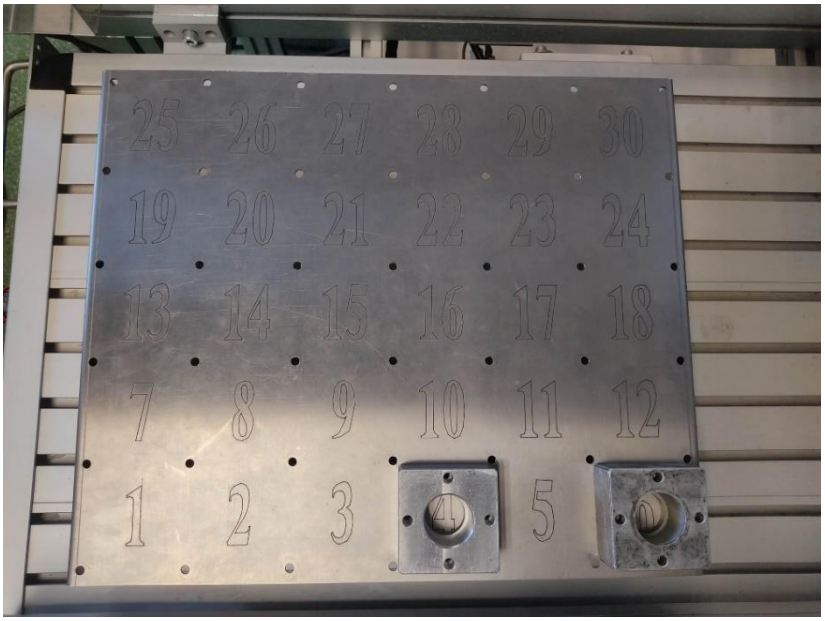


Figura 43. Almacén de piezas

3.2 Virtualización del hardware

En cuanto a la virtualización de los elementos componentes de la estación se ha llevado a cabo mediante un software de modelado en tres dimensiones conocido a gran escala mundial llamado Autodesk Inventor Profesional 2022 (Autodesk Inventor, 2022). Es un software que se creó en 1999 y desde entonces ha tenido un gran crecimiento en el mercado gracias a la cantidad de posibilidades que ofrece a la hora de modelar objetos tridimensionalmente. Representado su icono en la Figura 44.

Figura 44. Autodesk Inventor Profesional 2022

Este programa es muy útil a la hora de realizar los ensamblajes de los componentes 3D ya que tiene un entorno muy intuitivo y tiene la posibilidad de exportar el archivo a la extensión deseada, en especial para este proyecto, se ha trabajado con archivos OBJ. Además, gracias a la librería CAD de la empresa SMC (Enterprise, s.f.), proveedora de la Célula de Fabricación Flexible, se dispone de una gran cantidad de modelos que se incluyen en cada una de las estaciones disponibles para descargarlos, facilitando el proceso del proyecto.

Para realizar el montaje de la estación se ha llevado a cabo a través de diferentes ensamblajes. El ensamblaje inferior que incluye la mesa, el almacén y el actuador eléctrico inferior (Véase la Figura 45), el ensamblaje superior, que incluye el nexo entre actuadores, el actuador eléctrico superior y el indicador de vacío (Véase la Figura 46). El ensamblaje del cilindro se divide en parte fija y parte móvil (Véase la Figura 47 y la Figura 48). Por último, el diseño de la pieza individual (Véase la Figura 49).

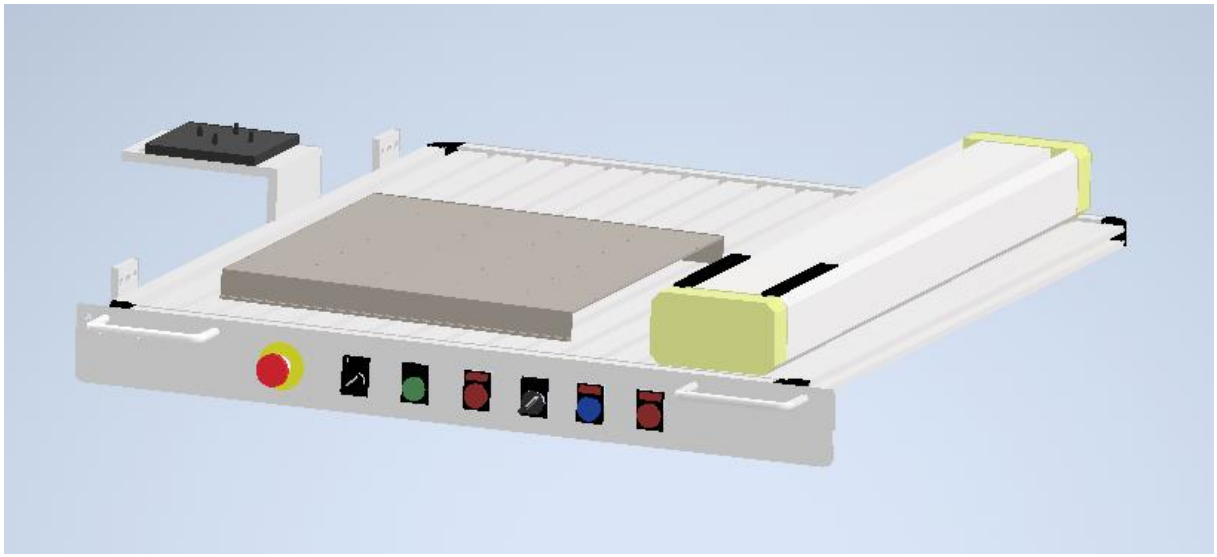


Figura 45. Ensamblaje Inferior

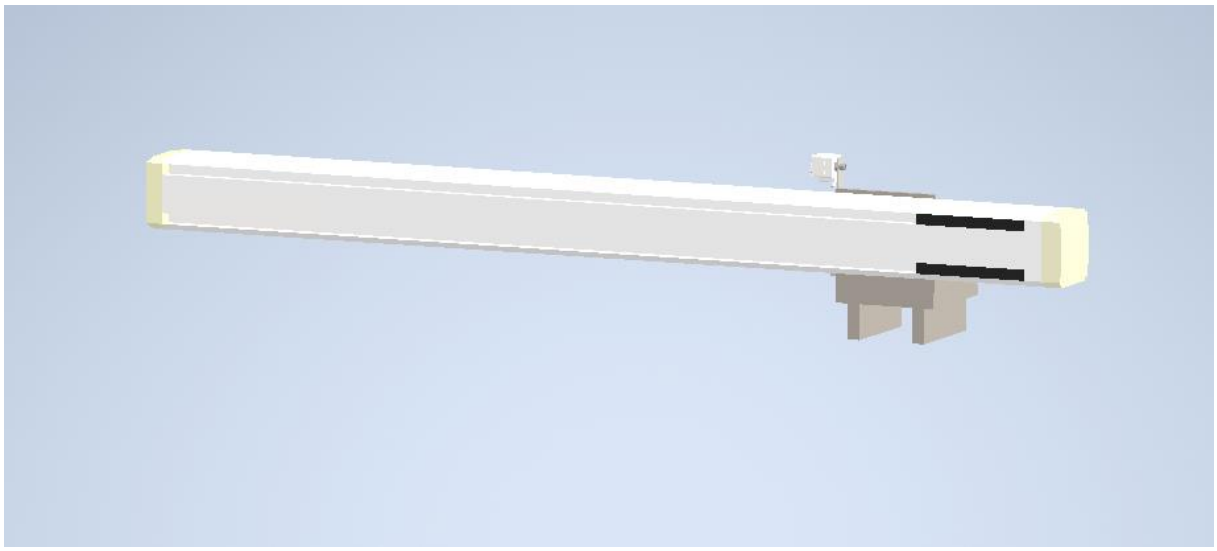


Figura 46. Ensamblaje superior

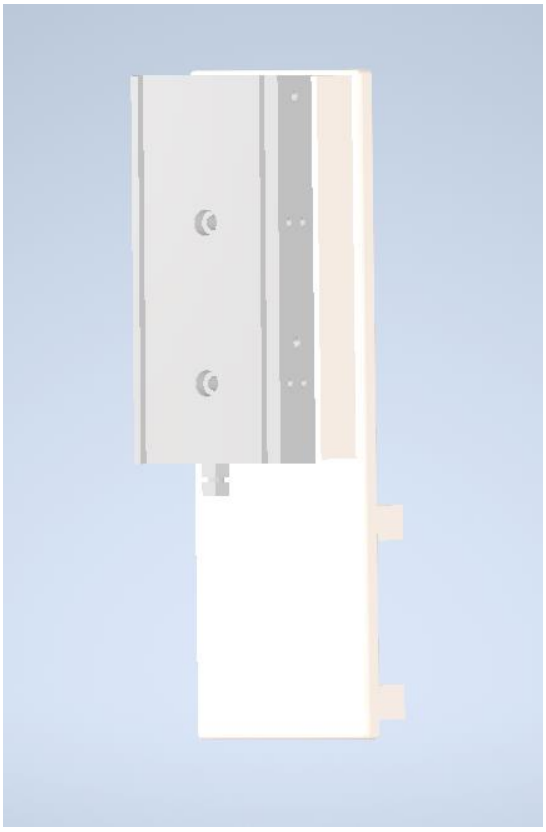


Figura 47. Ensamblaje del cilindro fijo

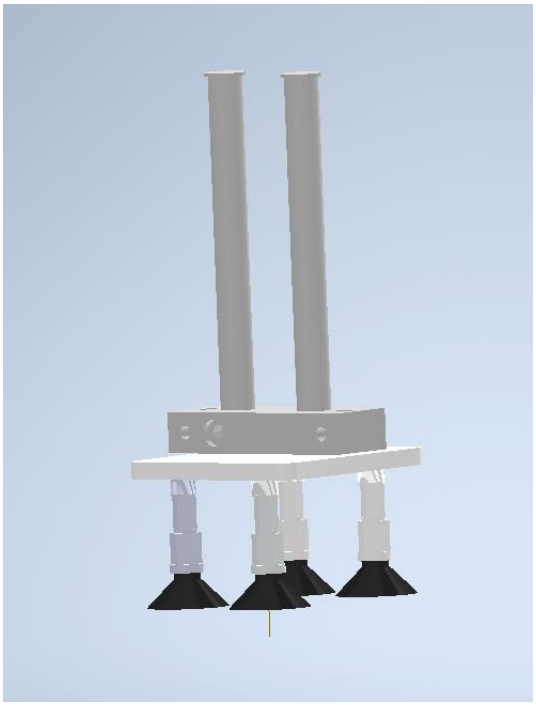


Figura 48. Ensamblaje del cilindro móvil

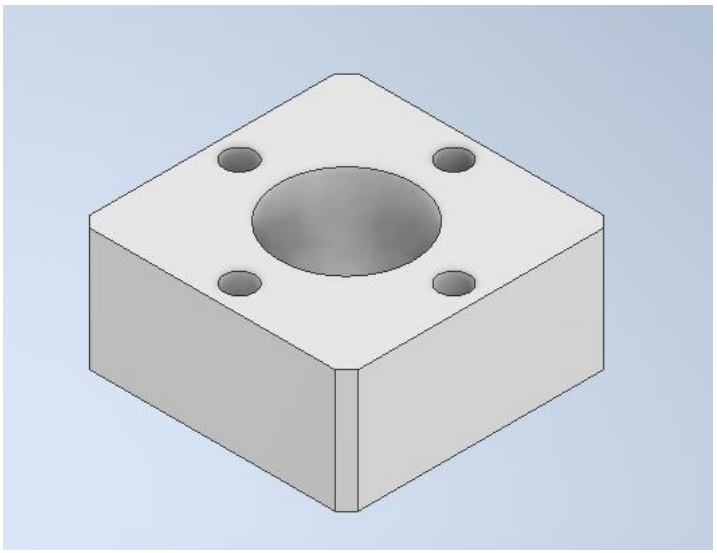


Figura 49. Pieza

3.3 Automatización del proceso

En cuanto a la automatización del proceso, se ha desarrollado en el software TIA Portal V14 (Véase el título TIA Portal V14) siguiendo los diagramas grafcet diseñados específicamente para la estación. En dicho diagrama se incluyen todos los casos de funcionamiento de la máquina que han sido considerados para cumplir el funcionamiento básico incluyendo tanto las operaciones internas del PLC como la interacción con el usuario.

En primer lugar, debe de ser considerada el conexionado eléctrico de las salidas conectadas al PLC de la máquina que vamos a desarrollar para poder comprender el proceso correctamente deben de ser activadas dependiendo de la etapa que se procese y deben de llamarse en un orden correcto para poder controlar adecuadamente el actuador eléctrico, dicha conexión eléctrica viene definida en la Tabla 1. Estas salidas del PLC están conectadas directamente a la controladora de los servomotores, por lo que se controlarán desde el PLC.

Tabla 1. Salidas del PLC

DIRECCIÓN	NOMBRE	UTILIDAD
124.3	RESET SERVO X	Introduce un reset en la controlara del eje X en caso estar en alarma por fallo mecánico
124.4	SERVO X ON	Activa el motor del servo del eje X
124.5	RESET SERVO Y	Introduce un reset en la controlara del eje Y en caso estar en alarma por fallo mecánico
124.6	SERVO Y ON	Activa el motor del servo del eje Y
124.7	SELECCIÓN DE PROGRAMA 0	Codificación del bit 0 del eje X
125.0	SELECCIÓN DE PROGRAMA 1	Codificación del bit 1 del eje X
125.1	SELECCIÓN DE PROGRAMA 2	Codificación del bit 2 del eje X
125.2	COMENZAR ROTACIÓN X	Señal que activa el movimiento en el eje X
125.3	SELECCIÓN DE PROGRAMA 0	Codificación del bit 0 del eje Y
125.4	SELECCIÓN DE PROGRAMA 1	Codificación del bit 1 del eje Y
125.5	SELECCIÓN DE PROGRAMA 2	Codificación del bit 2 del eje Y
125.6	COMENZAR ROTACIÓN Y	Señal que activa el movimiento en el eje Y

Véase en la Tabla 1. Salidas del PLC, se dispone de diferentes señales que controlan los servomotores tanto del eje x como del eje y. Para su correcto funcionamiento, se debe de seguir el siguiente proceso. Para comenzar, en caso de haber alarma en el sistema de control, se debe de activar la señal que manda un reseteo a la controladora para así, eliminar el estado de alarma. Una vez que el robot esté listo, debe de ser activada la salida “SERVO ON” de forma permanente para que esté en continuo funcionamiento. El robot devolverá una señal “*Driver Ready”* si todo está correctamente, en caso contrario mandará al PLC la señal de alarma. Si se ha arrancado por primera vez, para poder controlar el robot será necesario hacer un *Home* para referenciar el eje en el punto inicial. Por lo tanto, simplemente habría que

activar las salidas de rotación de los ejes (COMENZAR ROTACION) para que el robot se mueva hasta el origen. La referencia del robot se deberá realizar solamente una vez durante el proceso, excepto si se da el caso de alarma, que se deberá de repetir.

Posteriormente, una vez referenciado, para mover el robot a la posición deseada se debe de activar las salidas del PLC que codifican la posición final (Véase la Tabla 2. Matriz de posiciones del eje cartesiano) y activar las señales de "COMENZAR ROTACIÓN" de cada uno de los ejes. El actuador se desplazará, y cuando alcance su objetivo, activará la entrada del PLC "*Driver en Posicion"*. Las señales de activación de rotación y codificación de posición deben de ser desactivadas una vez que el movimiento se ha completado. Una vez hecho el ciclo, se procede de igual modo para llevarlo a la siguiente parada.

Se ha desarrollado un diagrama grafcet para cada uno de los elementos que deber ser controlados. Han sido utilizadas etapas según las condiciones que se cumplan a tiempo real.

3.3.1 Grafcet de START/STOP

Se desarrolla con la intención de dotar de funcionalidad a los botones de "START", "STOP" y "REARME" situados en la botonera de la estación. Al iniciarse la estación, comenzará activa la etapa "X51" como mecanismo de seguridad. Una vez que se pulse el botón "START" se saltará a la etapa "X50", etapa en la cual permitirá que comience el movimiento. En caso de ser pulsado los botones de "STOP" o "REARME" se volverá a la etapa anterior. Véase la Figura 50.

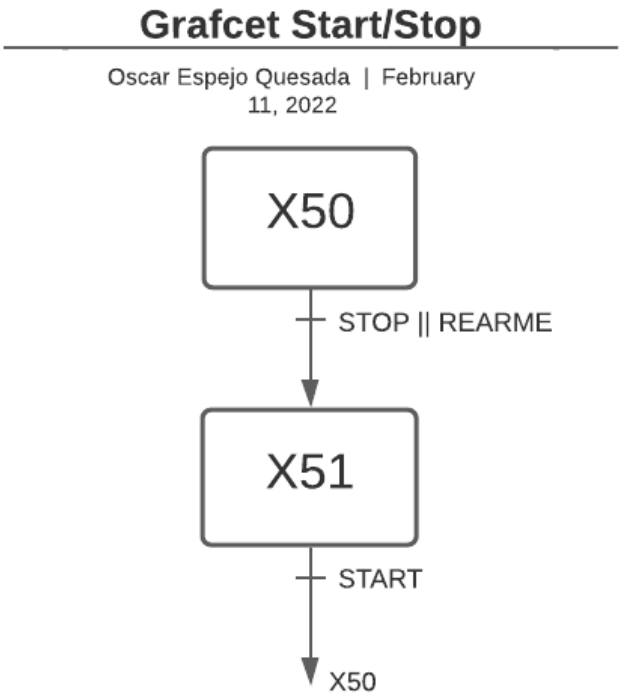


Figura 50. Grafcet START/STOP

3.3.2 Grafcet Automático/Manual

En la estación se ha incluido un interruptor que cumple la función de selección del modo automático, en el que se ha considerado que la estación funcione cíclicamente sin interrupción en el caso de situarse en la etapa “X60”, o, por el contrario, del modo manual, en la que solamente se ejecutará un ciclo si se sitúa en la etapa “X61”. Véase la Figura 51.

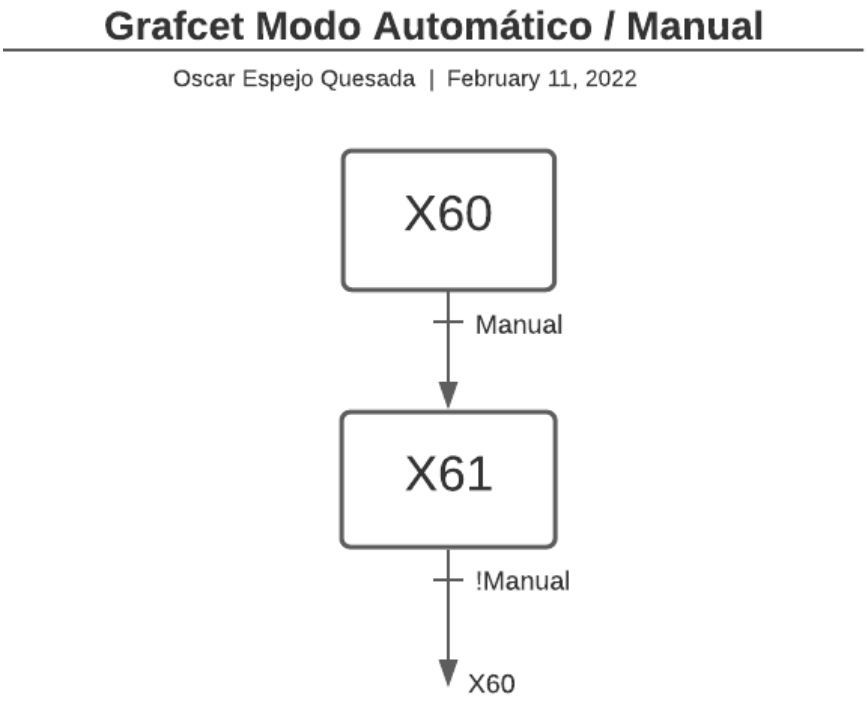


Figura 51. Grafcet Automático/Manual

3.3.3 Grafcet de control de vacío

Para realizar un control de la activación del sistema de vacío se ha considerado en un grafcet diferenciado al del proceso principal. Como etapa principal se sitúa la "X20" la cual es activada al iniciarse el programa, en ella el vacío estaría desactivado. Cuando el proceso principal llegue a la etapa "X4" significará que las ventosas del cilindro se han situado sobre la pieza y se saltará a la etapa "X21" automáticamente y, por tanto, se activará el vacío para agarrar la pieza. Véase la Figura 52.

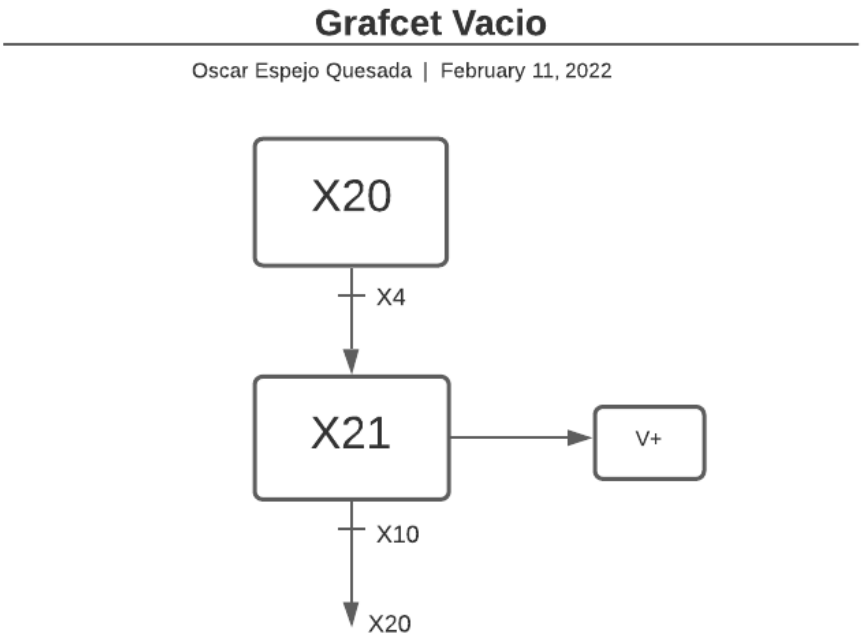


Figura 52. Grafcet de Vacío

3.3.4 Grafcet de cálculo de posición

Dicho diagrama se desarrolla con la intención de dividir el proceso principal del cálculo de posición, ya que supone un proceso de gran longitud por su gran número de etapas, tantas como posiciones definidas en el almacén de piezas. Una que se continúa hacia la etapa de la posición de dejada, se activarán unas determinadas salidas del PLC correspondiente a la codificación de la posición. El robot cartesiano funciona con una matriz de posiciones que ya están predefinidas en la controladora, por lo tanto, cada eje debe de mandar en código binario la posición a la que se desplaza a través de las señales DI0_X, DI1_X y DI2_X para el eje X y DI0_Y, DI1_Y y DI2_Y para el eje Y. Por lo tanto, realizando una adecuada combinación esas señales de ambos actuadores, se formará una matriz capaz de desplazarse a cada una de los puntos guardados. Al usar 3 bits, según el código binario se podría alcanzar hasta 7 posiciones, pero solamente se utilizan 6 en el eje X y 5 en el eje Y. Dicha combinación de señales viene definida en la Tabla 2.

.

Tabla 2. Matriz de posiciones del eje cartesiano

Numero	DI2_X	DI1_X	DI0_X	Salidas activas X	DI2_X	DI1_X	DI0_X	Salidas activas Y
Home	0	0	0		0	0	0	
Cogida de pieza	0	0	1	Q124.7	0	0	1	Q125,3
1	0	1	0	Q125,0	0	1	0	Q125,4
2	0	1	1	Q125,0 | Q124,7	0	1	1	Q125,3 | Q125,4
3	1	0	0	Q125,1	1	0	0	Q125,5
4	1	0	1	Q124,7 | Q125,1	1	0	1	Q125,3 | Q125,5
5	1	1	0	Q125,0 | Q125,1	1	1	0	Q125,4 | Q125,5
6	0	1	0	Q125,0	1	1	0	Q125,4
7	0	1	1	Q125,0 | Q124,7	1	1	0	Q125,3 | Q125,4
8	1	0	0	Q125,1	1	1	0	Q125,5
9	1	0	1	Q124,7 | Q125,1	1	1	0	Q125,3 | Q125,5
10	1	1	0	Q125,0 | Q125,1	1	1	0	Q125,4 | Q125,5
11	0	1	0	Q125,0	1	1	0	Q125,4
12	0	1	1	Q125,0 | Q124,7	1	1	0	Q125,3 | Q125,4
13	1	0	0	Q125,1	1	1	0	Q125,5
14	1	0	1	Q124,7 | Q125,1	1	1	0	Q125,3 | Q125,5
15	1	1	0	Q125,0 | Q125,1	1	1	0	Q125,4 | Q125,5
16	0	1	0	Q125,0	1	1	0	Q125,4
17	0	1	1	Q125,0 | Q124,7	1	1	0	Q125,3 | Q125,4
18	1	0	0	Q125,1	1	1	0	Q125,5
19	1	0	1	Q124,7 | Q125,1	1	1	0	Q125,3 | Q125,5
20	1	1	0	Q125,0 | Q125,1	1	1	0	Q125,4 | Q125,5
21	0	1	0	Q125,0	1	1	0	Q125,4
22	0	1	1	Q125,0 | Q124,7	1	1	0	Q125,3 | Q125,4
23	1	0	0	Q125,1	1	1	0	Q125,5
24	1	0	1	Q124,7 | Q125,1	1	1	0	Q125,3 | Q125,5
25	1	1	0	Q125,0 | Q125,1	1	1	0	Q125,4 | Q125,5
26	0	1	0	Q125,0	1	1	0	Q125,4
27	0	1	1	Q125,0 | Q124,7	1	1	0	Q125,3 | Q125,4
28	1	0	0	Q125,1	1	1	0	Q125,5
29	1	0	1	Q124,7 | Q125,1	1	1	0	Q125,3 | Q125,5
30	1	1	0	Q125,0 | Q125,1	1	1	0	Q125,4 | Q125,5

Su funcionamiento se basa en 31 etapas ilustradas en laFigura 53. Inicialmente, dicho diagrama se encuentra en la etapa principal llamada “X300”, la cual corresponde a la posición de cogida de pieza, de forma póstuma, cuando se alcanza la etapa correspondiente en el grafcet principal (“X6”), se realiza la comprobación del número de piezas que se han procesado hasta ese momento y dependiendo del resultado, dará paso a una etapa u otra. Las etapas que corresponden a las posiciones del almacén se comprenden entre la “X301” a la “X330”. Una vez dentro de dicha etapa, se procederá a hacer la llamada a la señal que comienza el movimiento en cada eje en el grafcet principal.

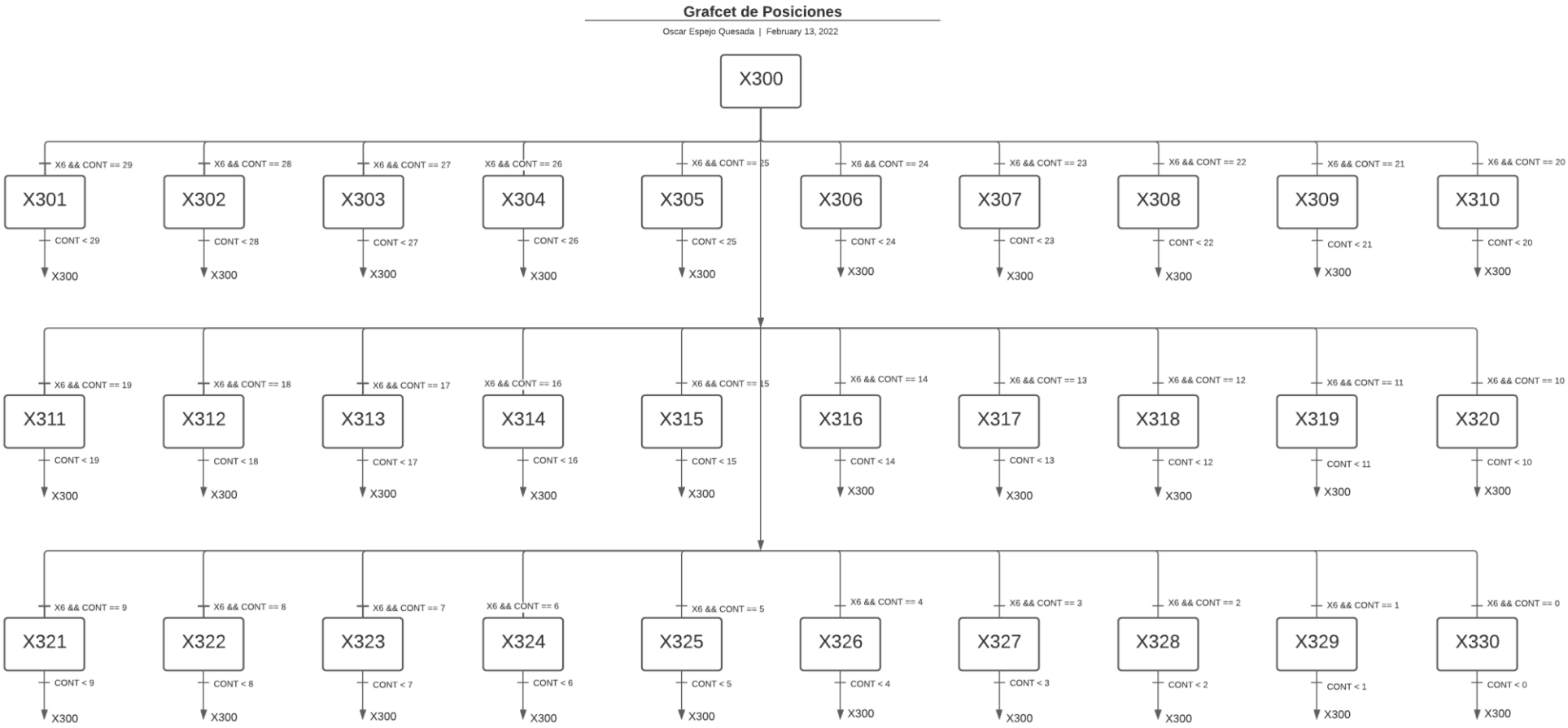


Figura 53. Grafcet posiciones

Por último, una vez el contador se ha decrementado en la etapa “X10” del proceso principal, se volverá a la etapa “X300” con la idea de volver a la posición de cogida de pieza finalizando así el ciclo. Véase la Figura 55.

3.3.5 Grafcet de rearme

En el diagrama Grafcet del proceso de rearme se describe tanto el proceso de inicialización al comenzar el funcionamiento del programa una vez cargado en el PLC, como el uso que se le asigna al botón de rearme de la máquina tal como indica la Figura 54.

Inicialmente, al arrancar el programa, la máquina se encuentra en estado de alarma porque es necesario realizar un home antes de comenzar la actuación. Por lo tanto, al comenzar el ciclo, aunque se active la etapa “X40” se saltará directamente a la etapa “X41” dado que se cumplen las condiciones de alarma. Una vez pulsado el botón de rearme en ese caso, será enviado un reset de error a la controladora que permitirá eliminar el estado de alarma. Posteriormente, cuando el servomotor envía la señal notificando que está listo para actuar, se salta a la siguiente etapa “X44” que activa el movimiento llevando el robot a la posición inicial de Home.

Por otro lado, se ha considerado darle funcionalidad al botón de rearme durante la ejecución del programa. Si mientras se desarrolla el ciclo, es pulsado dicho botón, se forzará a volver a la etapa “X0” dando lugar a llevar al robot a la posición de cogida de pieza y parándolo en dicha posición, además de reiniciar el contador al valor inicial, comenzando el proceso de paletizado desde la primera posición del almacén.

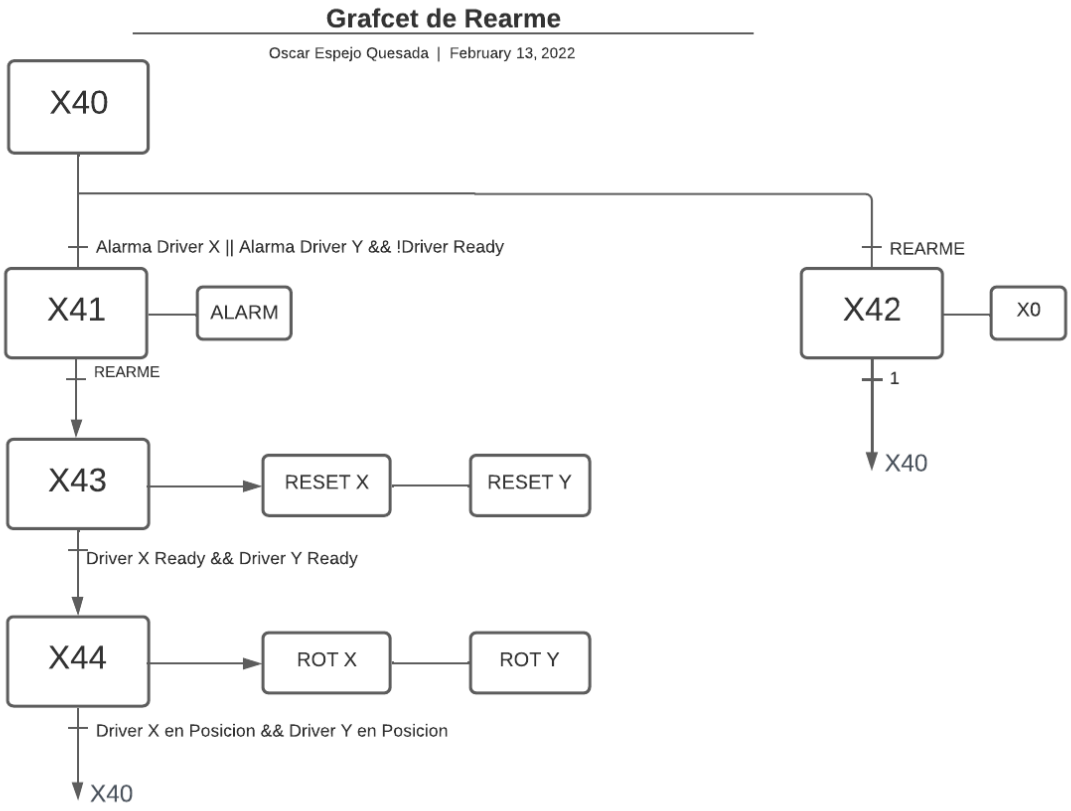


Figura 54. Grafcet de rearme

3.3.6 Grafcet principal

En el diagrama principal se definen cada uno de los pasos que deben de ser seguidos en el proceso (Véase la Figura 55). Al iniciar la máquina, comienza en la etapa “X0” en la cual simplemente se asigna el contador de piezas al valor máximo (29 piezas) para iniciar el almacenaje desde la primera posición, conforme avancen los ciclos, se irá decrementando este valor. En el caso de que el cilindro esté retraído y no haya ninguna alarma en el sistema de los servomotores, se adelanta a la etapa siguiente. En la “X1” se procederá a dar marcha a los motores. Una vez que se reciban las señales que indican que los servomotores están preparados para comenzar el funcionamiento y está activa la etapa “X50” (que indica si el robot está iniciado) se pasará a la siguiente etapa.

A continuación, en la siguiente etapa se indica al robot la posición número uno, que corresponde a aproximación a la cinta transportadora para comenzar el proceso de recogida de la pieza. Para ello, es indicada la posición a través de la correspondiente codificación y son activadas las salidas que comienzan la rotación del motor. A modo de mecanismo de seguridad, se añadir en esta etapa un pequeño temporizador que retrasa el paso a la siguiente etapa.

En este paso a la siguiente etapa entra en juego el uso del modo automático y modo manual. Para que el ciclo sea repetitivo y no tenga interrupciones debe ser indicado el modo automático, y, por el contrario, si se desea realizar solamente un ciclo de colocación, se debe indicar en el modo manual y posteriormente pulsar en el botón “START”. Si el servomotor se encuentra en posición uno mandado con anterioridad, se cumplen las etapas “X40” y “X50” y se ha activado el temporizador de seguridad, se procederá al salto a la siguiente etapa.

En la etapa “X3” simplemente manda la orden al cilindro de ser expandido. Esta acaba cuando se activa el final de carrera asociado “A1” y si la controladora no arroja ninguna alarma. Posteriormente, en la etapa “X4” se mantendrá la orden de expandir el vástago y se dará la orden de activar el vacío con intención de coger la pieza. Para hacer posible la cogida, es necesario que el proceso tenga la pausa necesaria para que el sistema de ventosas y vacío agarre firmemente a la pieza, es por ello, que se ha añadido un temporizador en dicha etapa.

Una vez finalizado el tiempo y estando la controladora libre de alarmas, se salta a la siguiente etapa “X5” en la cual se activa un temporizador que permite al cilindro retraerse con tiempo suficiente para evitar inestabilidad de la pieza. Por lo tanto, una vez retraído el cilindro y cumplido el tiempo de seguridad, se pasa a la siguiente etapa.

En la etapa “X6” se procede a hacer el cálculo de la posición a la que se va a dirigir el eje cartesiano. Véase la Figura 53 en el cual hace referencia a que dependiendo de la posición a la que se dirija, se activará una etapa distinguida entre los valores 301 y 330. Una vez terminado el procesado de la posición, comprueba que no siga activa la etapa “X300” para poder continuar.

El siguiente paso viene reflejado en la etapa “X7”, encargado de dar marcha a la posición definida con anterioridad mediante la activación de la rotación de los ejes. Se activa un temporizador de seguridad que da un tiempo de espera al circuito a que alcance la posición indicada para evitar saltos inesperados entre etapas. Ya que la señal que activa el robot una vez llega a una posición que se le indica es la llamada “Driver en Posición”. Esta salida permanecerá activa hasta que dé comienzo la orden de aproximarse a otra posición determinada, que será desactivada por el PLC. La ausencia de este temporizador puede suponer la confusión de la interpretación de esta salida si no ha sido desactivada rápidamente y se encontrase activa desde el ciclo anterior.

Una vez alcanzada la posición, cumplido el tiempo y dándose la ausencia de alarmas en el sistema, se pasa a la siguiente etapa “X8” en la que se extiende el vástago del cilindro para aproximarse a la posición de dejada en el almacén. En este caso se procederá de igual forma que en el conjunto de etapas del proceso de cogida de pieza. Una vez alcanzado el final de carrera “A1” se saltará a la etapa “X9” que activará un temporizador de seguridad de estabilidad de la pieza. Cumplido este tiempo, se desactivará el vacío y se retraerá el vástago del cilindro.

Además, se procede a añadir una unidad en el contador de piezas interno del sistema. Por lo tanto, una vez terminado el ciclo, se comprobará el número de piezas que hay en el almacén, en el caso de que no se haya alcanzado el número de piezas totales, se deberá continuar a la posición siguiente, por lo tanto, se reiniciará el ciclo en la etapa “X2”. En caso contrario, si el número de piezas es igual al máximo, se volverá a la etapa “X0”.

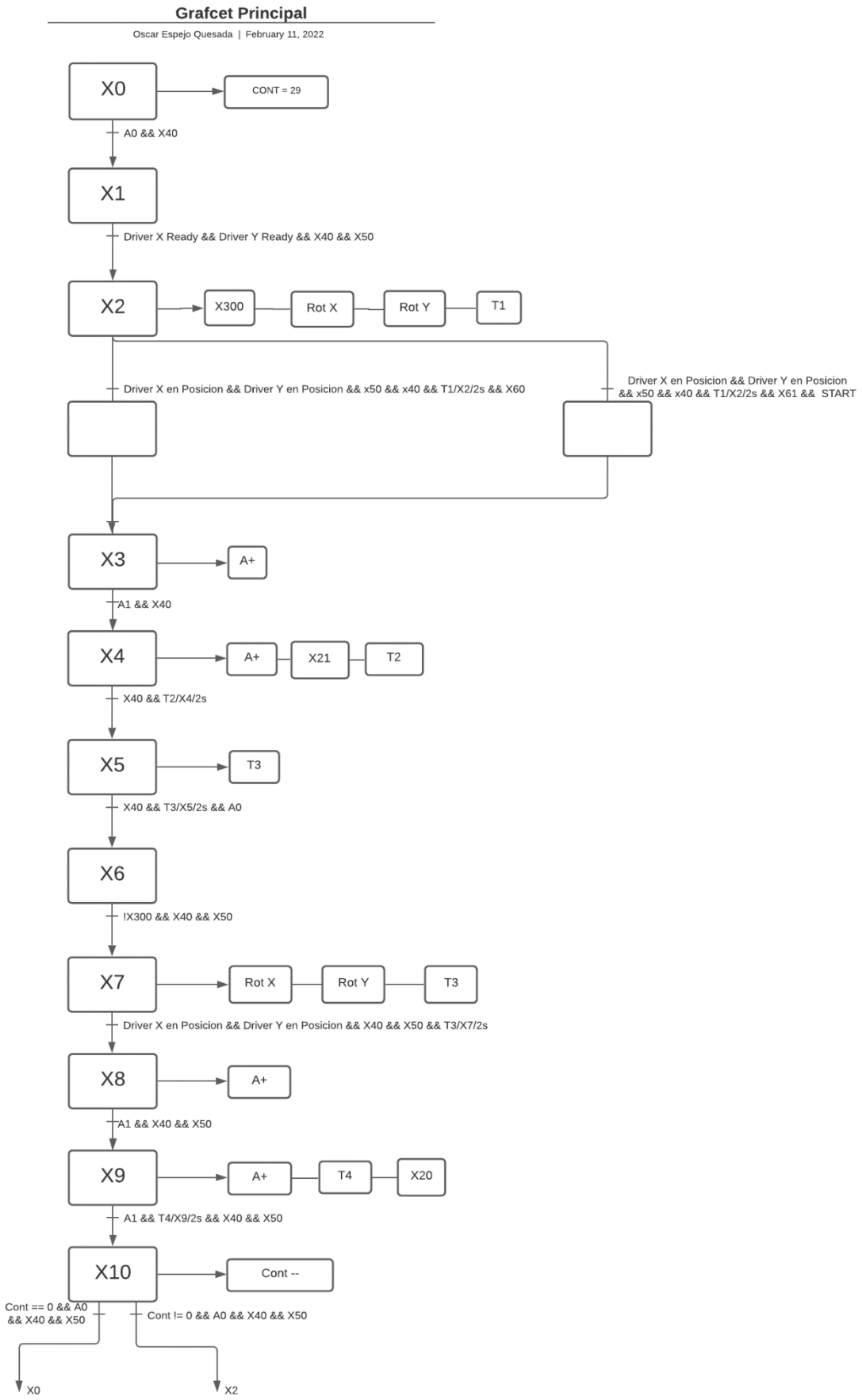


Figura 55. Grafcet principal

3.4 Comunicación

La comunicación entre el programa de automatización creado en TIA Portal, que incluye tanto las instrucciones del proceso automático como la del control de la pantalla HMI de forma gráfica, y el programa del gemelo digital realizado en NetBeans, en el cual se describen los parámetros y atributos de cada uno de los elementos que forman el sistema, se realizará a través de unas herramientas de simulación, aunque se podría realizar mediante un PLC real.

Para ello, será necesario descargar el programa en el simulador de PLC llamado PLCSIM, incluido en con el software de Siemens. Éste actuará de igual modo que un Siemens S7-1200 real. Su función será la de realizar la acción de activar o desactivar sus salidas en función de las entradas que reciba.

Por otro lado, será necesario crear un servidor que establezca la comunicación de señales entre el simulador de PLC y el gemelo digital. Esta herramienta es conocida como NetToPLCsim, el cual recogerá las direcciones IP tanto del simulador como del gemelo digital y actuará de intermediario entre ellos.

Por lo tanto, una vez establecida la comunicación, el gemelo enviará las señales al simulador, y éste responderá con otras señales que harán que actúe la representación 3D.

Todo el proceso de comunicación viene definido en el ANEXO C: REALIZAR CONEXIÓN ENTRE TIA PORTAL Y NETBEANS.

4 Manual de usuario

Con la finalidad de que cualquier usuario sea capaz de poder hacer uso del gemelo digital se ha creado este manual que detalla los pasos a seguir para la correcta iniciación y manipulación del gemelo digital.

Para particularizar en la aportación de este proyecto, se priorizará la importación de nuevos objetos al archivo XML a la escena de forma que se puedan incluir todos los elementos necesarios para conformar la estación.

En primer lugar, se deben de importar los parámetros del diseño de la representación que caracterizarán la ventana donde se visualice el gemelo digital. Para ello es necesario determinar el ancho, alto y posición inicial de origen dentro de la escena, tal como ilustra la Figura 56.

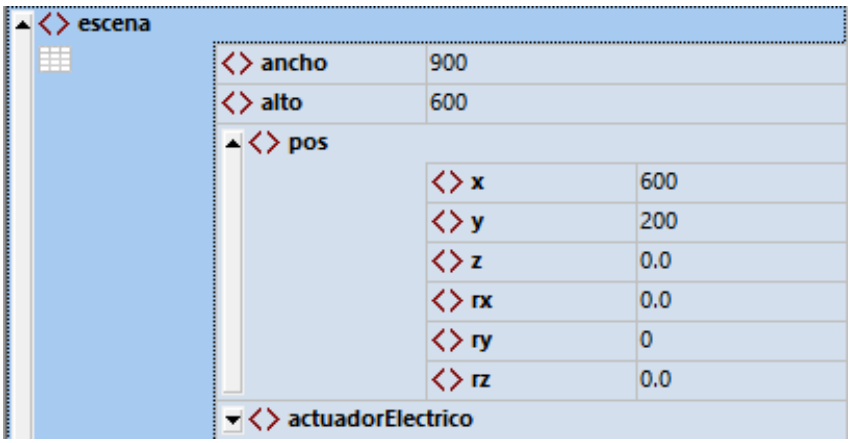


Figura 56. Creación de parámetros de escena

Posteriormente, será necesario importar los objetos que incluye la estación. Para ello en este caso se ha comenzado por la parte inferior formado por el ensamblaje de la mesa, el almacén y el actuador eléctrico, todo este conjunto se ha considerado un elemento “ActuadorElectrico” para simplificar el concepto de representación. Este objeto incluye en su interior otros objetos que son considerados hijos y dan lugar al funcionamiento del actuador combinando su funcionamiento como se indica en la Figura 57.

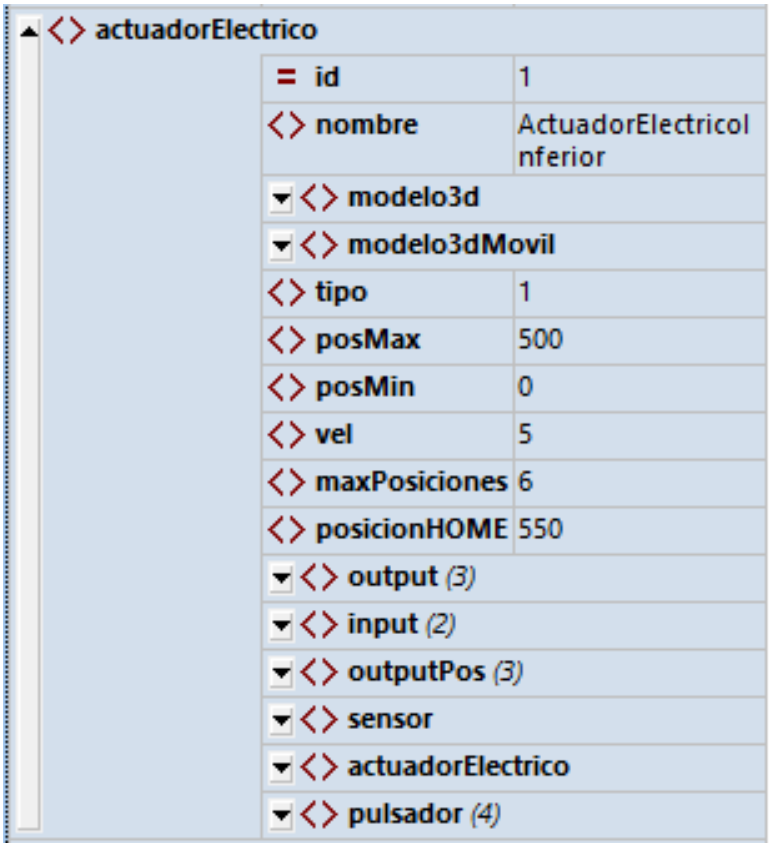


Figura 57. Creación de objeto "ActuadorElectrico"

En primer lugar, se describen los parámetros que incluye la clase "ActuadorElectrico" para funcionar de forma autónoma, en este caso son la posición máxima y mínima que puede recorrer el actuador en las posiciones guardadas del almacén, la velocidad a la que se desplazará el actuador, el número máximo de posiciones que permite guardar piezas en ese eje (Este almacén está formado por 5 filas y 6 columnas) y, por último, la posición de Home donde estará situado en la representación.

Se puede observar que posteriormente, en el interior del objeto creado anteriormente se pueden observar otros parámetros cuyo origen es otra clase del gemelo digital, son considerados parámetros hijos. Para crear dichos elementos hijos se deben de incluir dentro de las etiquetas de apertura y cierre del objeto "ActuadorElectrico". Por ejemplo, véase la Figura 58 para observar cómo se debería de importar un parámetro "Model3D" a un objeto. Sería necesario definir la posición que ocupará, el fichero OBJ al que corresponde y la escala a la que será representado.

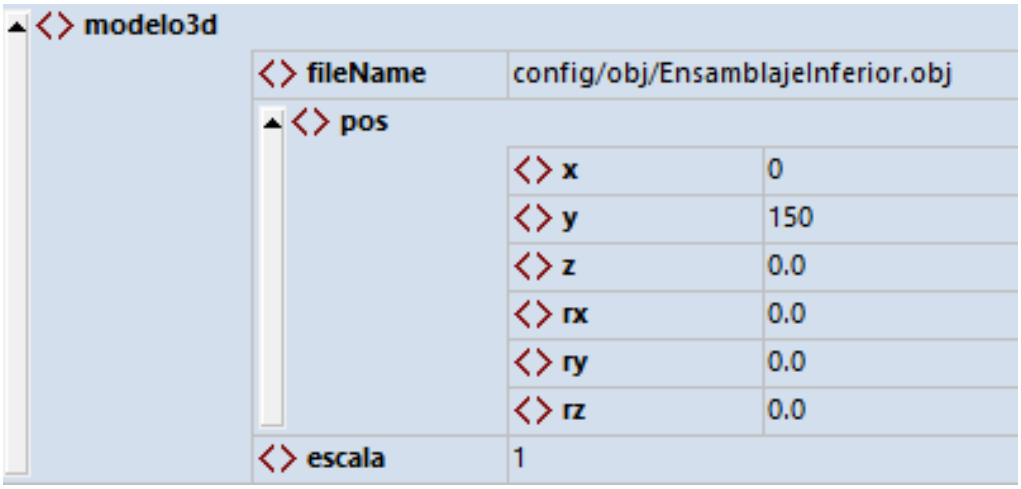


Figura 58. Ejemplo de modelo3D en XML

Particularizando para la clase “Posicion”. Véase la Figura 59 donde se puede observar qué parámetros puede incluir un atributo “pos”.

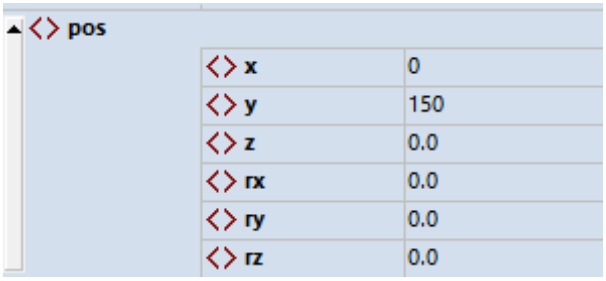


Figura 59. Creación de parámetro posición

En el caso de una etiqueta llamada “sensor” se debe de incluir el nombre, el modelo3D del objeto, la posición máxima y mínima a la cual estará activo el sensor digital y la señal del PLC a la que corresponde el conexionado eléctrico que actúa como entrada reflejada en Figura 60.

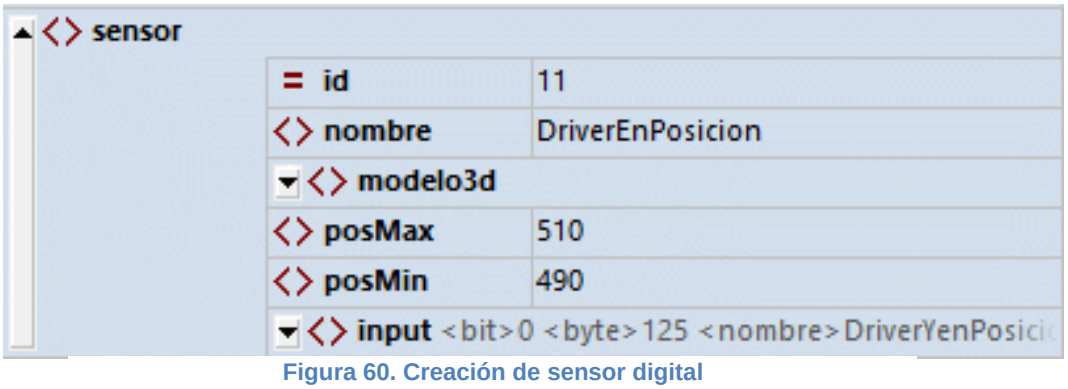


Figura 60. Creación de sensor digital

Por otro lado, para crear directamente conexiones al PLC, se hará con las clases “PlcOutput” o “PlcInput”, que corresponden a las etiquetas <Output> o <OutputPos> en el caso de salida y <Input> para el caso de entrada. El procesado de la información es el mismo asignando la señal a las entradas o salidas según corresponda, con la única diferencia de la etiqueta que contenga en el archivo XML como se puede observar en la Figura 61.

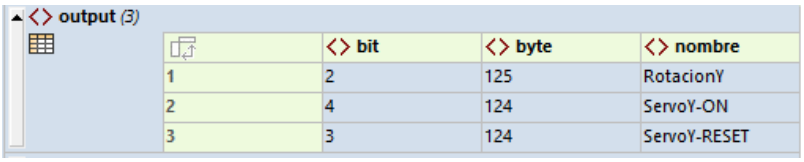


Figura 61. Creación de salida/entrada al PLC

Por último, haciendo referencia a la Figura 57. Creación de objeto “ActuadorElectrico”, se puede observar que el último elemento es otro objeto “ActuadorElectrico”, de este modo, dicho elemento dependerá siempre del elemento padre, por lo tanto, si el elemento padre realiza un movimiento, le afectará al hijo.

Finalmente, existe también la clase “ElectroValvula” que incorpora la clase “Cilindro”, solamente es utilizado una vez en el proyecto para el cilindro de la ventosa. Indicado cómo se debe de importar un objeto válvula en la Figura 62.

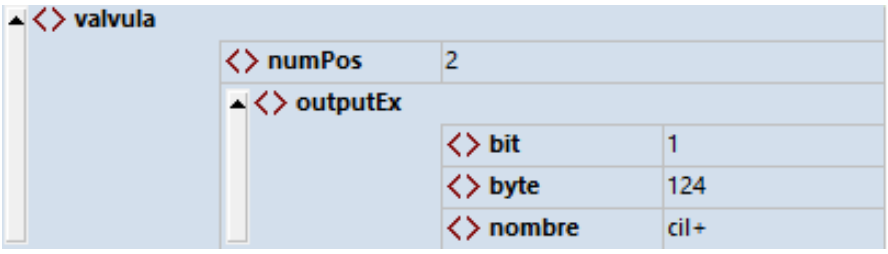


Figura 62. Creación de válvula

Las imágenes anteriores han sido extraídas desde un programa de visión de archivos XML llamado AltovaXMLSpy, el cual permite abrir y manejar este tipo de ficheros de una forma gráfica y sencilla, aporta más que abriendo el documento con el blog de notas.

5 Conclusiones

Una vez que concluido el proceso del proyecto del gemelo digital, se ha podido comprobar el gran beneficio que atribuye una herramienta como ésta tanto en al ámbito de la enseñanza como en el de la industria. Es por esa clara notoriedad de bien por el cual es cada vez más desarrollada a nivel mundial.

A pesar de haber importado diferentes mejoras a la línea de proyectos de la virtualización de las estaciones de la Célula de Fabricación Flexible FMS200, existe todavía, mucho más margen de mejora que serán incluidas conforme se avance en la investigación de dicho proyecto, dando lugar a la construcción de una gran herramienta flexible para cualquier sistema que se plantee ser virtualizado de forma eficiente y robusta.

Para ello, convendría seguir la línea de investigación en dicho proyecto para añadir funciones que ofrezcan la posibilidad de interactuar con el usuario que desea controlar la estación. Un ejemplo práctico que sería de gran utilizad sería la implementación de un sistema que asemeje la representación de la estación a un software de modelado en tres dimensiones, en el cual se pueda elegir a partir de un desplegable, los elementos que desean ser representados y poder importarlos al sistema de forma intuitiva mediante una opción dentro de la propia ejecución. O, por otro lado, poder situar los elementos en la posición deseada de forma intuitiva pudiendo interactuar así, con el ratón o el teclado. Hasta ahora, el mecanismo que se ha utilizado es basado prácticamente en prueba y error asignando coordenadas en el espacio al origen de cada una de los ensamblajes del sistema.

En conclusión, a pesar de denotarse una mejora desde el primer proyecto que se desarrolló de este conjunto de estaciones, queda mucho camino por recorrer y mucho margen de mejora que dará lugar a la creación de un software que supondrá un pilar de gran importancia pudiéndose adaptar tanto en educación como en la actual revolución industrial llamada Industria 4.0.

6 ANEXOS

6.1 ANEXO A: CREACIÓN DE PROYECTO EN TIA PORTAL V14

Cuando el software se inicia por primera vez, muestra una pantalla tal y como se puede observar en la Figura 63. En la cual da la opción de crear un proyecto nuevo, abrir uno ya creado o realizar una migración de un proyecto.

La opción Welcome Tour redirige a la página de Siemens donde se puede encontrar información sobre el uso del programa. Una vez abierto algún proyecto, podemos cambiar la vista del proyecto a una que ofrece la visualización de los hitos.

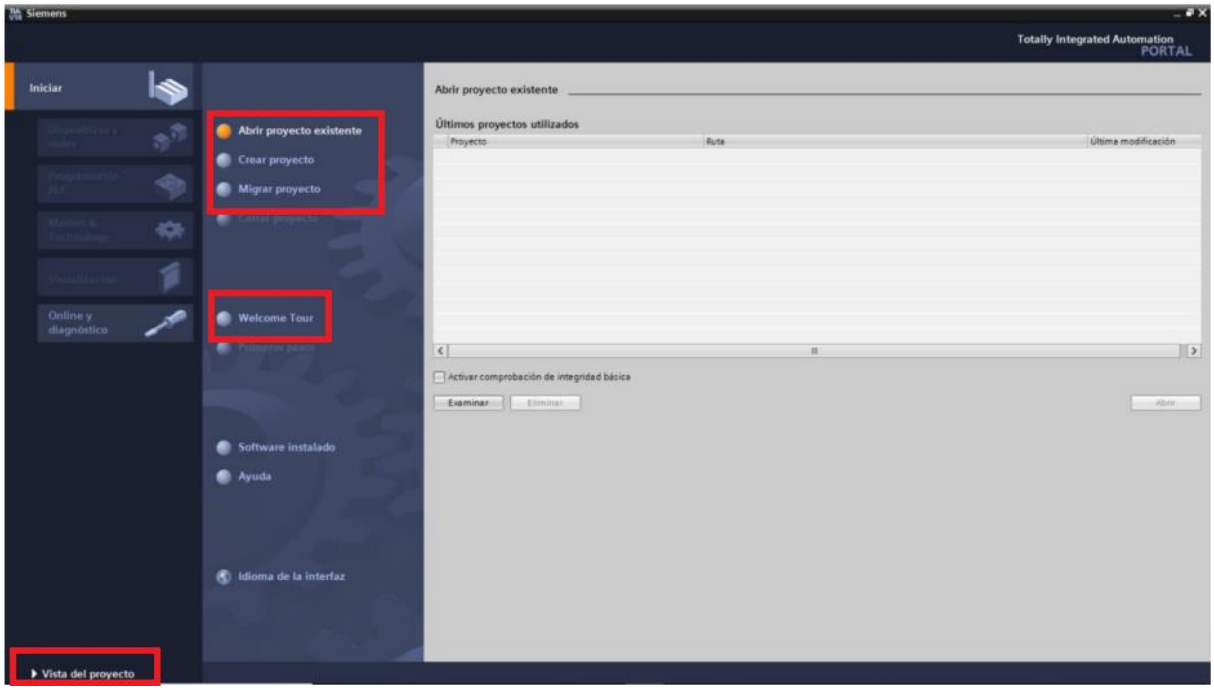


Figura 63. Pantalla de inicio de TIA Portal

Tal como se refleja en la Figura 64 una vez se ha creado y se ha entrado en la vista del proyecto, se da la opción de añadir dispositivos al entorno para comenzar a trabajar. Será necesario importar tantos dispositivos como tenga el sistema real.

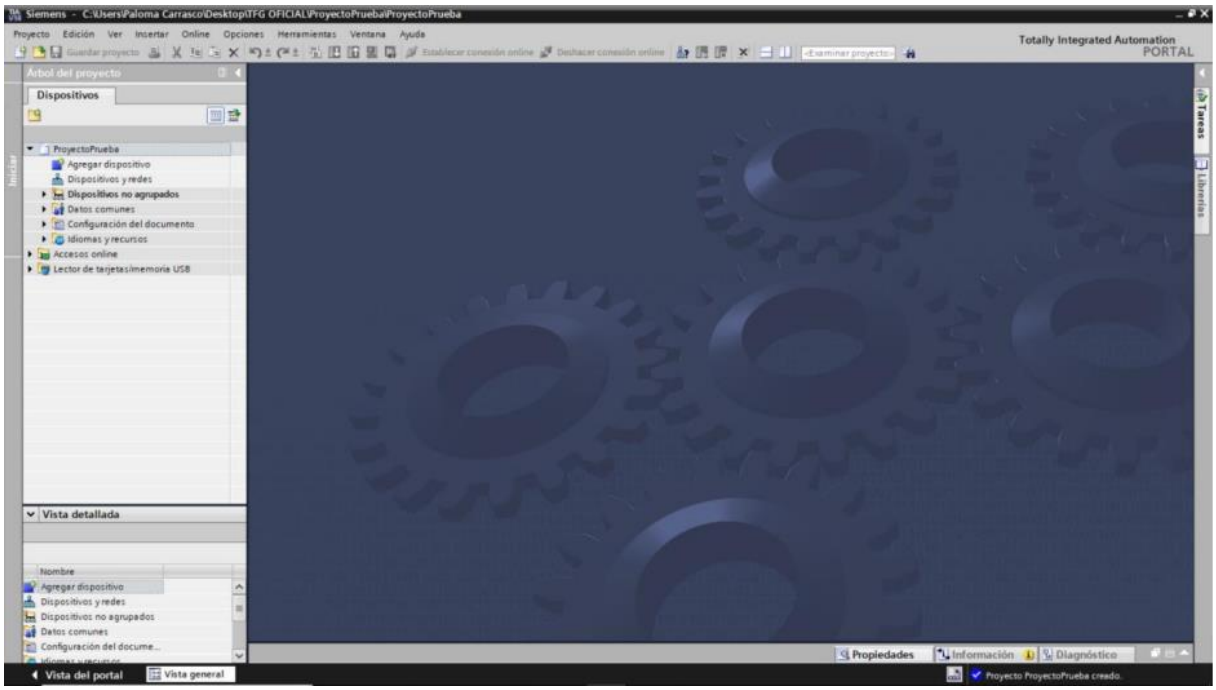


Figura 64. Vista de proyecto

Para añadir un dispositivo se debe clicar en agregar dispositivos y aparecerá una ventana con el catálogo de Siemens, solamente sería necesario buscar la referencia del dispositivo que queremos importar como se puede observar en la Figura 65.

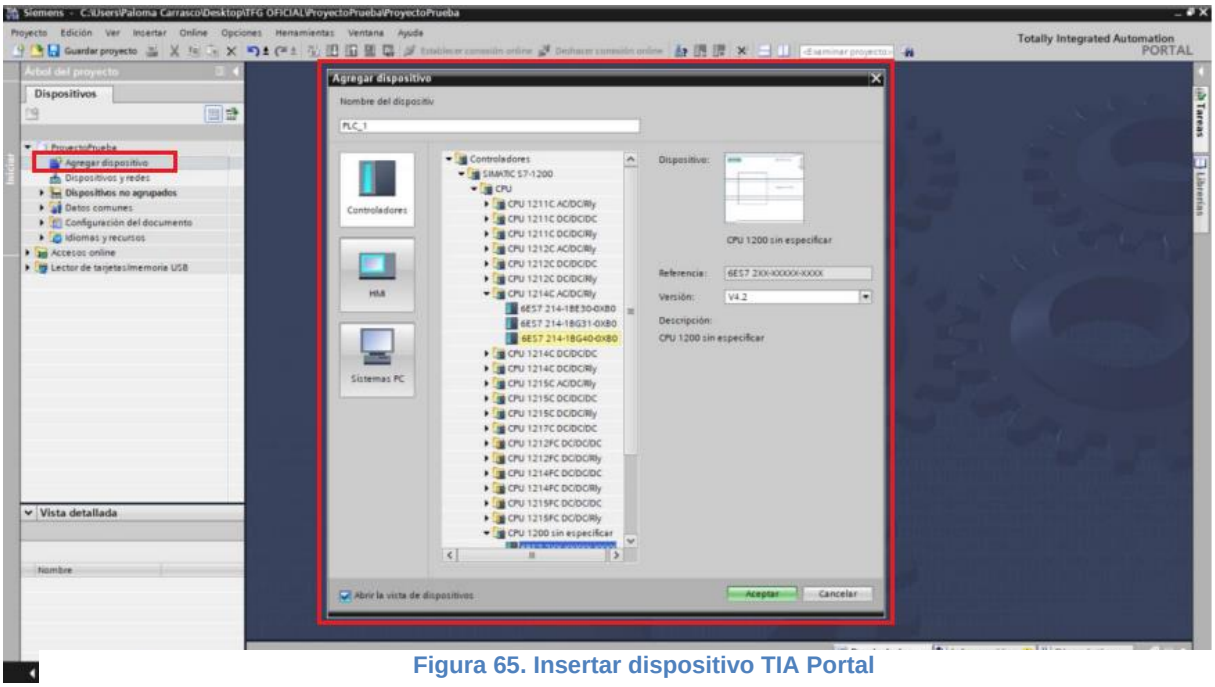


Figura 65. Insertar dispositivo TIA Portal

Una vez estructurado todo el hardware, en la ventana de árbol de proyecto situada a la izquierda de la interfaz gráfica aparecerá los dispositivos importados. Para comenzar la programación del PLC, se debe acceder clicando en dicho elemento y en la opción de bloques de programa. En esta carpeta se pueden observar que ya existen dos bloques creados por defecto, el Main[OB1] y el StartUp[OB100]. En el StartUp es la parte que se ejecutará solamente una vez al descargar el programa al PLC, por lo tanto, se debe de establecer en este bloque las variables que se desean que comiencen activas. Mientras que en el bloque Main, es un bloque que se ejecuta cíclicamente y en él se ejecutará el proceso del programa. Al abrir uno de los dos ficheros, se podrá observar la ventana mostrada en la Figura 66.

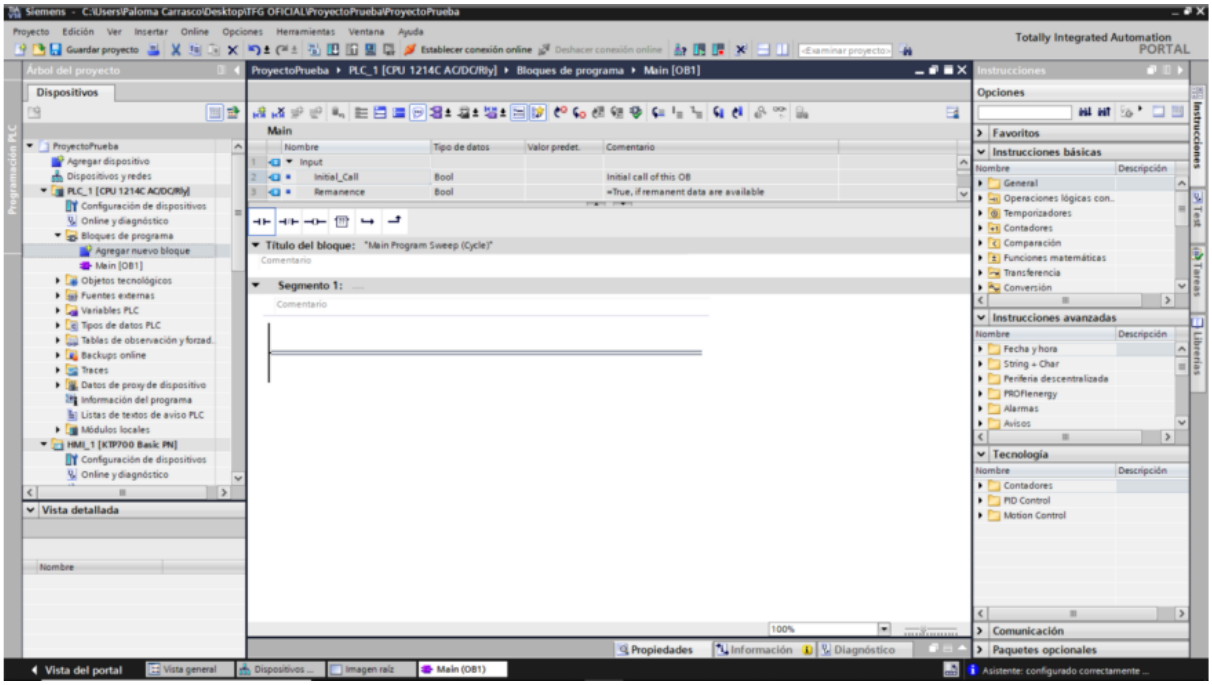


Figura 66. Programación Main[OB1] en Tia Portal

A continuación, se deben de añadir las condiciones necesarias utilizando los bloques funcionales definidos en la interfaz para ir desarrollando el programa deseado. Para ello se debe de hacer de forma manual clicando en el elemento elegido y arrastrando hasta la posición deseada. Los elementos que se usan principalmente son los mostrados en la Tabla 3.

Tabla 3. Elementos operacionales TIA Portal

Icono	Función
-| |-	Contacto normalmente abierto
-| / |-	Contacto normalmente cerrado
-( )-	Asignación
-(S)-	Activar salida
-(R)-	Desactivar salida
	Crear rama paralela
	Cerrar rama paralela
	Llamada a función predeterminada

Por otro lado, en la ventana situada a la derecha de la interfaz, podemos encontrar todo tipo de funciones predefinidas. Las utilizadas en este proyecto son las instrucciones básicas de comparación, contadores y temporizadores, que son indicadas en la Tabla 4, Tabla 5 y Tabla 6, respectivamente.

Tabla 4. Operadores de comparación TIA Portal

Símbolo	Función
CMP | == |	Igual
CMP | < > |	Diferente
CMP | > = |	Mayor o igual
CMP | < = |	Menor o igual
CMP | > |	Mayor
CMP | < |	Menor

Tabla 5. Temporizadores TIA Portal

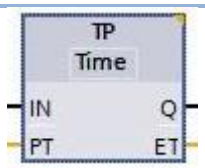
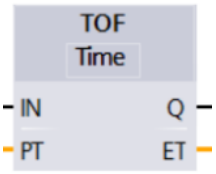
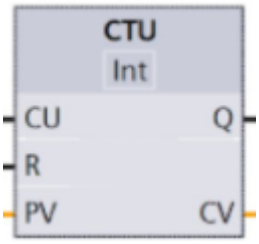
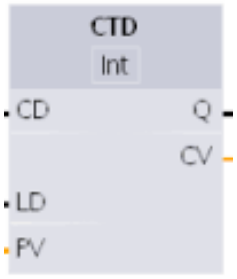
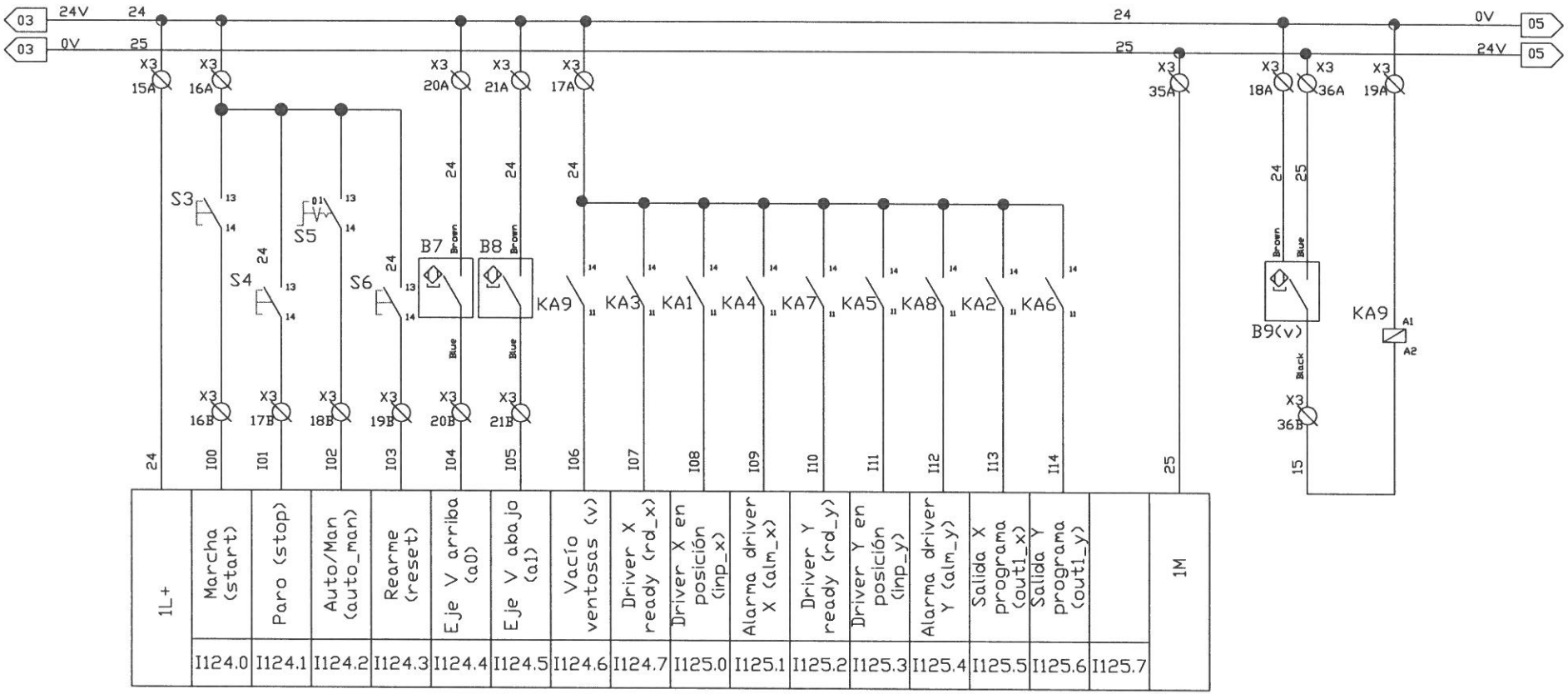
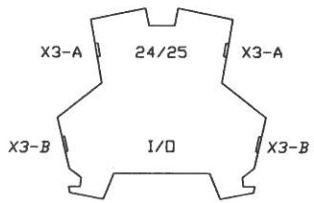
Símbolo	Función
	Temporizador que empieza a contar con el primer impulso de activación del proceso
	Temporizador que activa su salida cuando el tiempo finaliza
	Temporizador que desactiva su salida cuando el tiempo finaliza

Tabla 6. Contadores TIA Portal

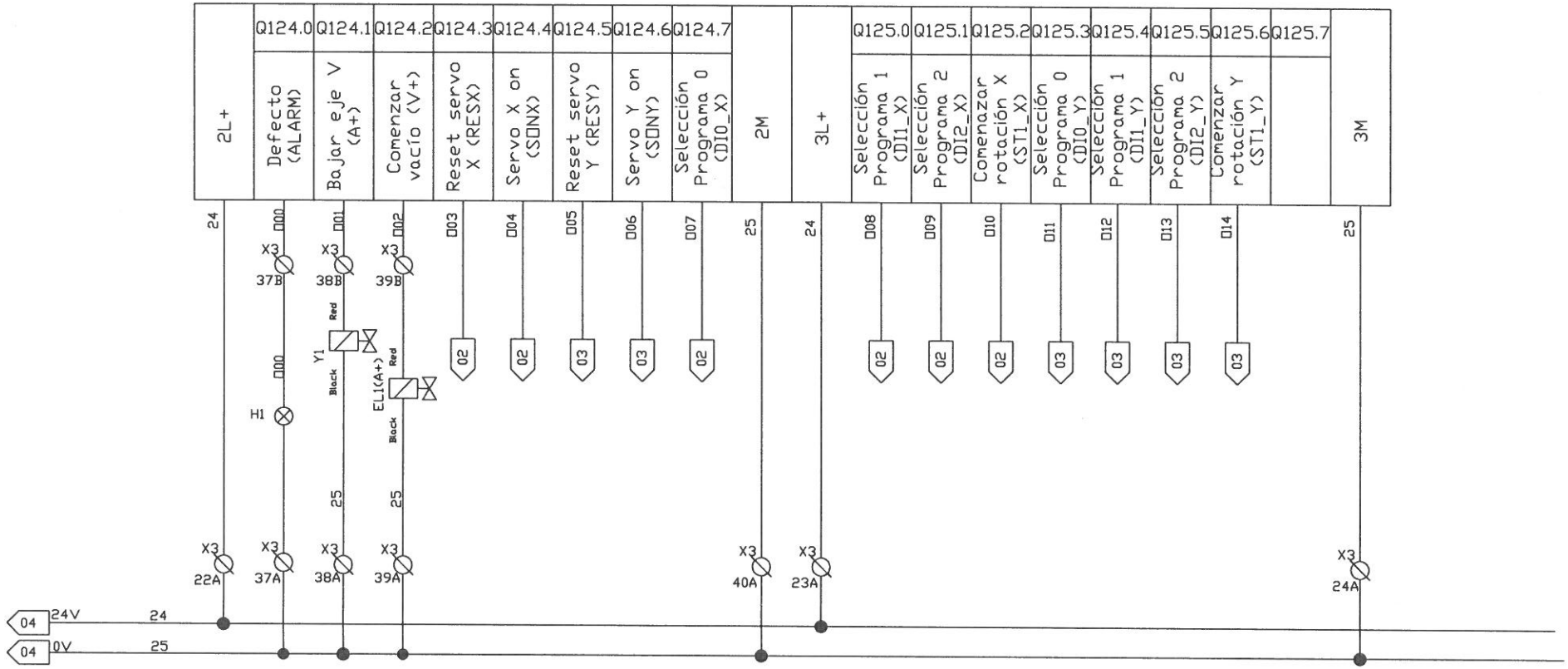
Símbolo	Función
	Contador que aumenta su valor cuando reciba un pulso por CU hasta que alcanza su valor máximo indicado en PV. Cuando termine se activará la salida Q. Para resetear el contador se mandará un pulso a la señal R.
	Contador que decrementa su valor cuando recibe un pulso por CD hasta que alcanza el valor mínimo indicado en PV. Cuando lo alcance, se activará la señal Q.

6.2 ANEXO B: PLANOS ELÉCTRICOS





Tolerancia General: Segun ISO-2768 m	Ref.	Fecha	Nombre	Modificacion	MATERIAL:
					TRATAMIENTO
					CANTIDAD:
SMC INTERNATIONAL TRAINING	FMS-200			DESIGNACION: CIRCUITO DE CONTROL EST. ALMACÉN-ENTRADAS	ESCALA:
					Fecha: 11-03-05
					Dibujado: J.R.Barreal
					Comprobado: Fdz. de Antona
					Nº DE PLANO: S/8-4



Tolerancia General: Segun ISO-2768 m	Ref.	Fecha	Nombre	Modificaciones	MATERIAL:	ESCALA:
					TRATAMIENTO	Fecha: 11-04-05
					CANTIDAD:	Dibujado: J.R.Barreal
						Comprobado: Fdz. de Antona

SMC INTERNATIONAL TRAINING

FMS-200

DESIGNACION: CIRCUITO DE CONTROL EST. ALMACÉN-SALIDAS

Nº DE PLANO: S/8-5

6.3 ANEXO C: REALIZAR CONEXIÓN ENTRE TIA PORTAL Y NETBEANS

En primer lugar, se ejecutarán TIA Portal, PLCSIM y NetToPLCsim. Es necesario que este último se ejecute con permisos de administrador. Una vez abierto el programa TIA Portal, es necesario abrir el proyecto que se va a comunicar con el gemelo digital y, para que se funcione, es necesario que solamente se importe al proyecto un solo PLC sin periféricos y activar la comunicación vía PUT/GET del interlocutor remoto. Para ello, se accederá a la pestaña de Dispositivos y redes como se indica en la Figura 67, después a la configuración del PLC importado al proyecto para poder visualizar la ventana donde aparece dicha opción.

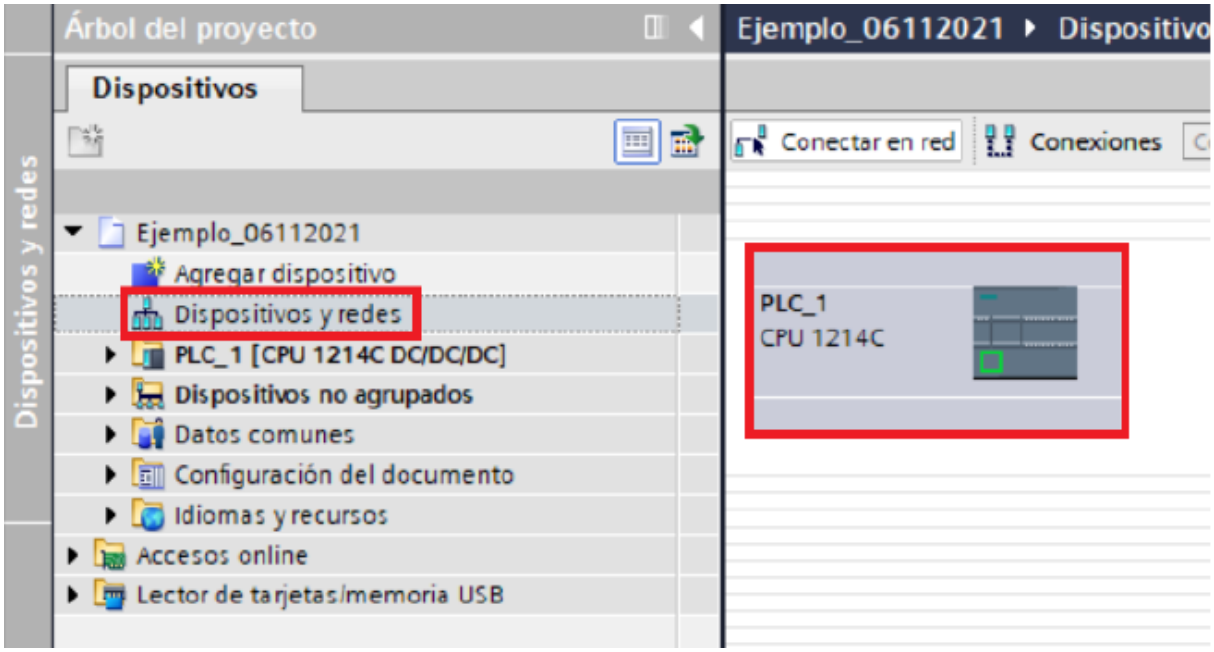


Figura 67. Acceso a dispositivos y redes

Una vez en esta ventana, se hará doble clic sobre el PLC y aparecerá la ventana de configuración en la parte inferior de la interfaz. Accediendo a la pestaña General y posteriormente a Protección y Seguridad se encontrará la opción de permitir el acceso al igual que en la Figura 68.

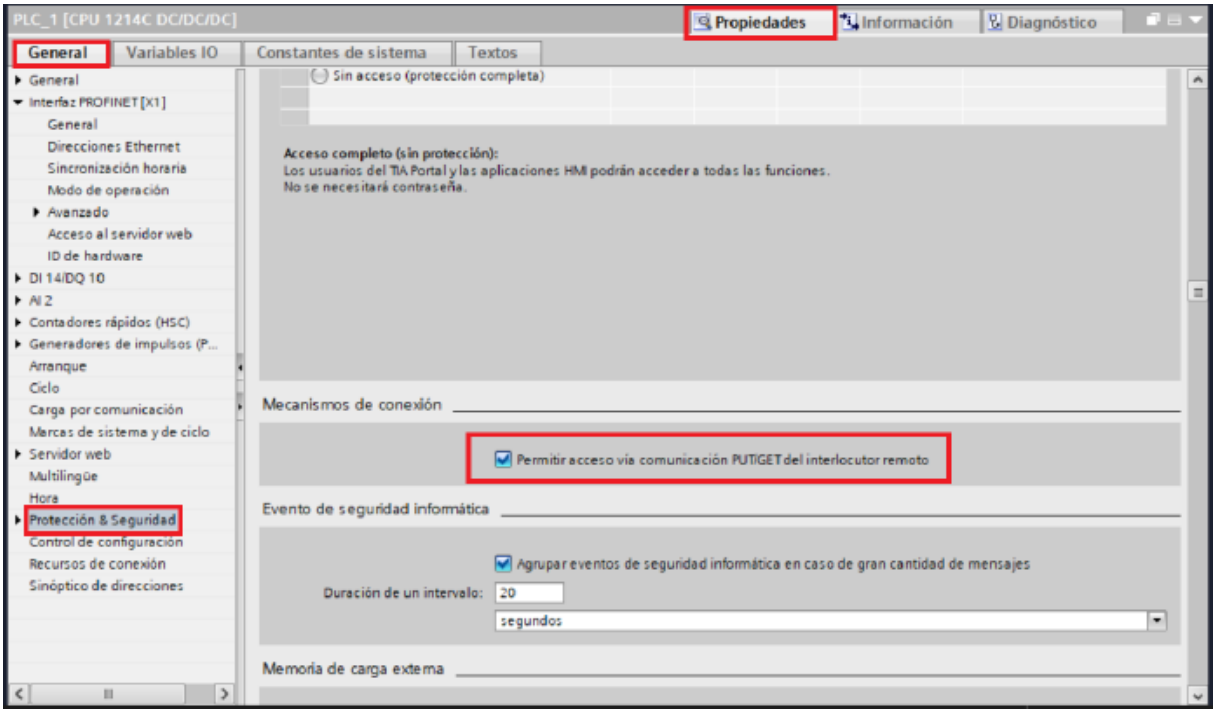


Figura 68.Activación de la comunicación PUT/GET

Una vez activado, se procederá a iniciar el simulador PLCSIM desde la misma interfaz de TIA Portal, para ello será necesario clicar sobre el símbolo que refleja la Figura 69.

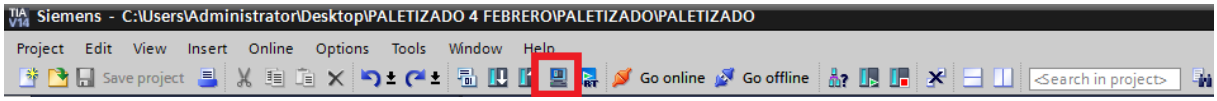


Figura 69.Icono de inicio del simulador PLCSIM

A continuación, aparecerá una ventana emergente que indicará el proceso de descarga del programa al PLC. El procedimiento es igual que si fuese un PLC real, tal como refleja la Figura 70.

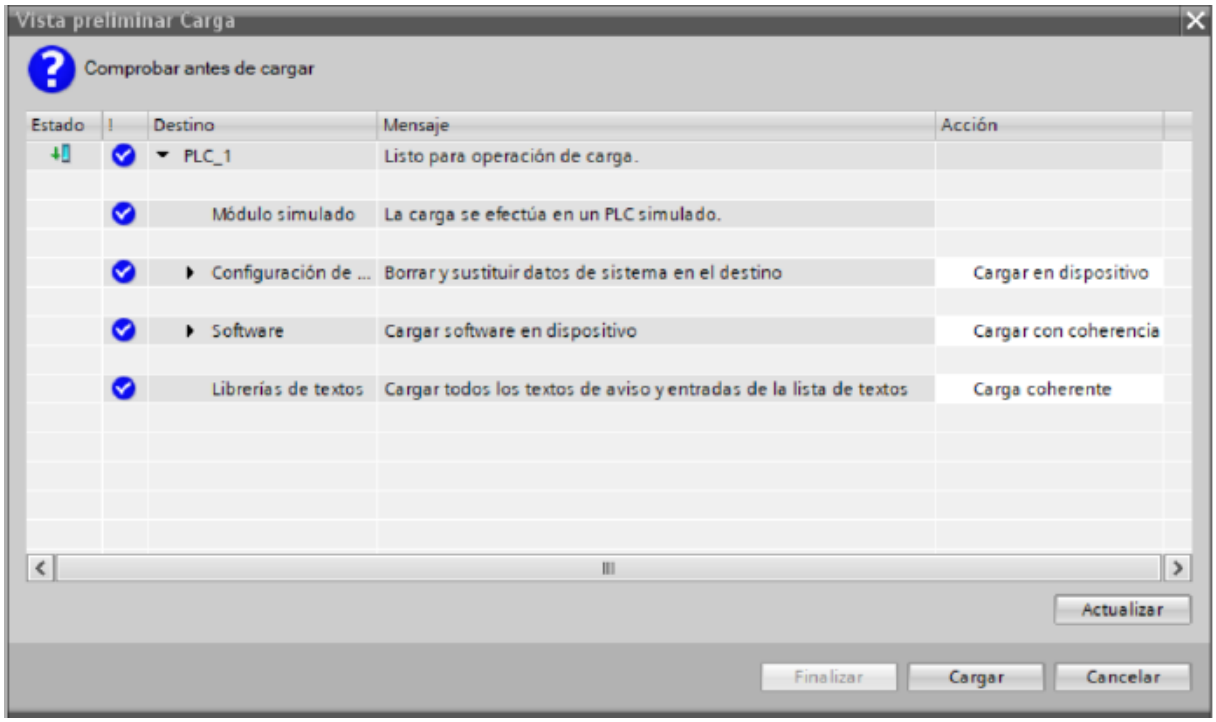


Figura 70. Carga del programa en el PLC virtual

Por último, una vez ejecutado el simulador, aparecerá una ventana que indica el estado del PLC, así como un panel de botones capaces de dar comienzo o parar la ejecución. Simulan las señales luminosas que tiene el PLC real. Véase la Figura 71.

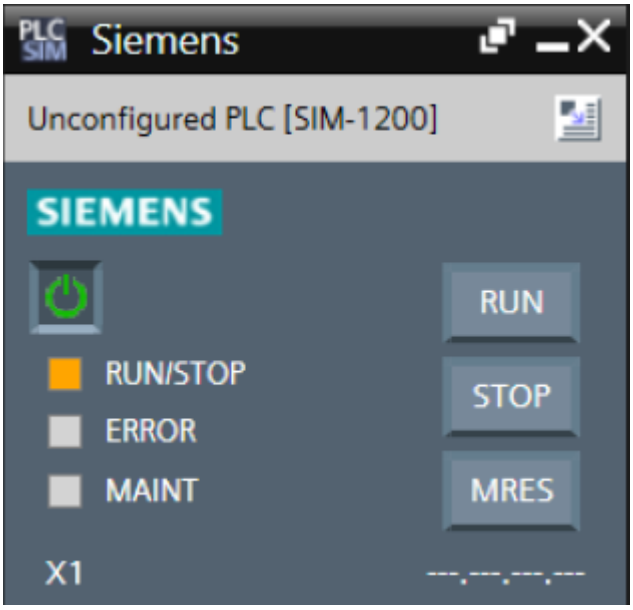


Figura 71. Simulador PLCSIM

El siguiente paso será ejecutar el programa NetToPLCsim. Es importante que tenga permisos de administrador para que funcione correctamente, para ello se debe de seguir el procedimiento indicado en la Figura 72, clic derecho sobre el archivo y ejecutar como administrador.

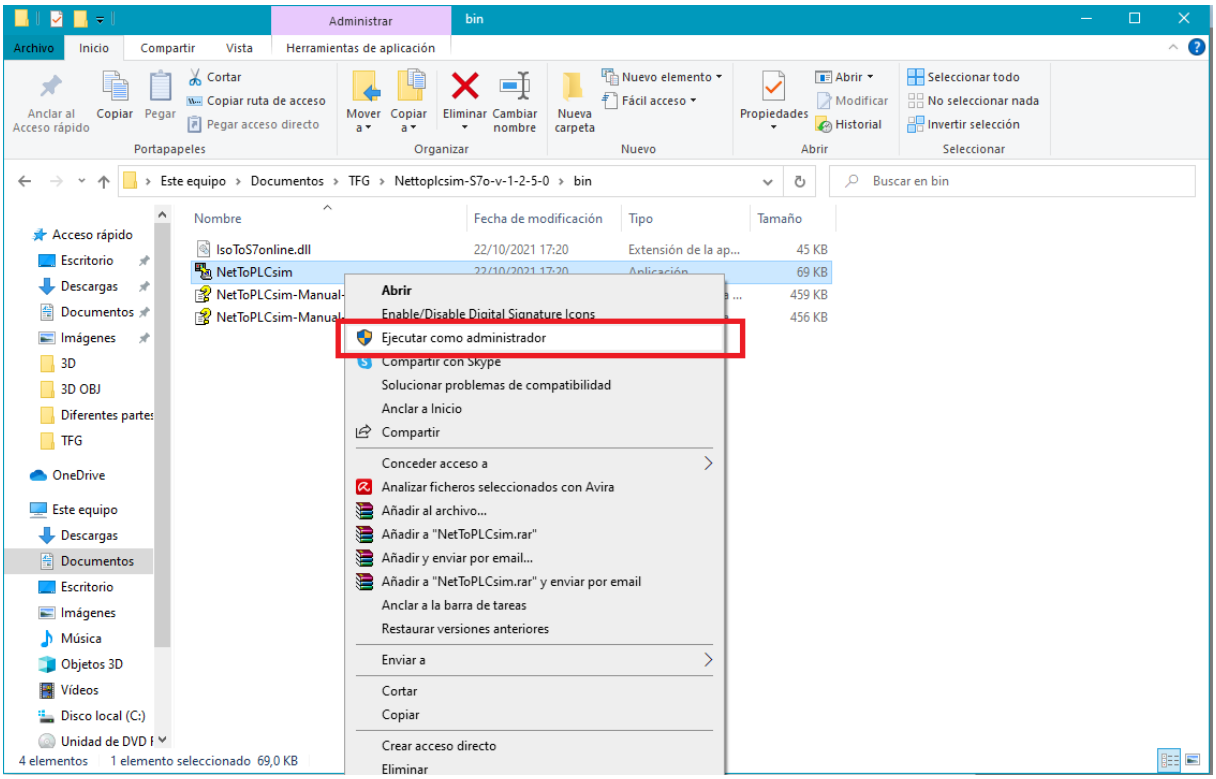


Figura 72.Ejecutar como administrador NetToPLCsim

Una vez abierto, aparecerá una ventana que indicará un aviso indicando que el puerto está siendo utilizado. Se parará la ejecución en el botón como la Figura 73.

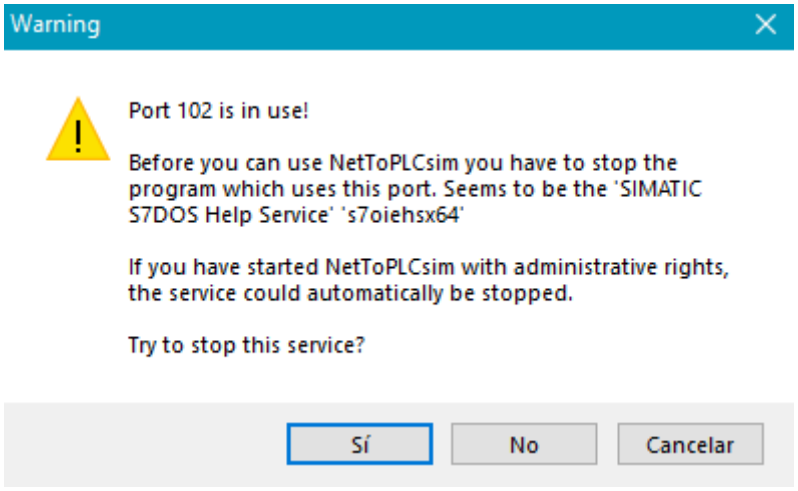


Figura 73. Mensaje de aviso del NetToPLCsim

A continuación, se debe de añadir un nuevo servidor haciendo clic sobre el botón Add de la ventana del programa. Véase la Figura 74.

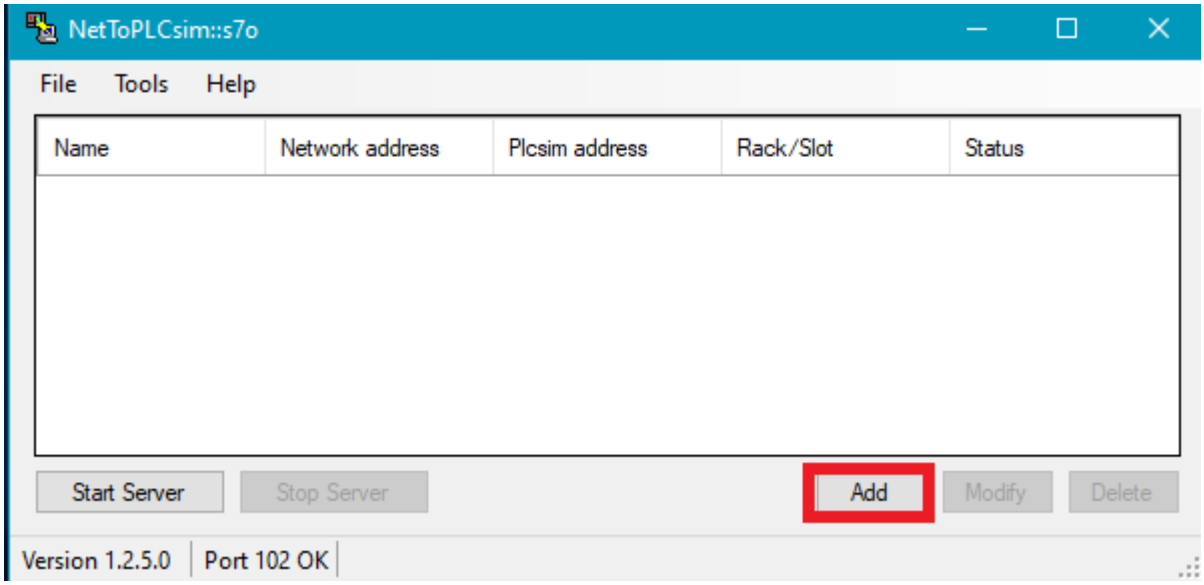


Figura 74. Añadir servidor

Posteriormente se introducirán los campos para terminar con la configuración del nuevo servidor. Se deben de introducir el nombre, la dirección IP tanto de la red como del simulador PLCSIM y el número de rack y slot. Se debe de tener en cuenta que, si se produce alguna modificación de alguno de los parámetros en el programa de TIA Portal, se debe de modificar el servidor. Haciendo uso de los tres puntos situados a la derecha del campo vacío aparecerá un desplegable que indica todas las posibilidades de conexión. Véase la Figura 75.

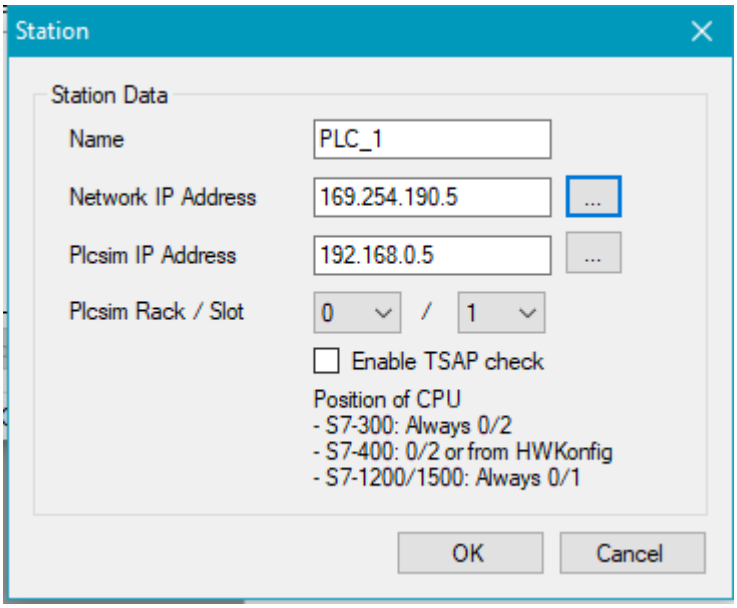


Figura 75.Configuración de los parámetros del servidor

Una vez creado, aparecerá en el listado de servidores. Para realizar la conexión se debe pulsar en Start Server. Véase la Figura 76.

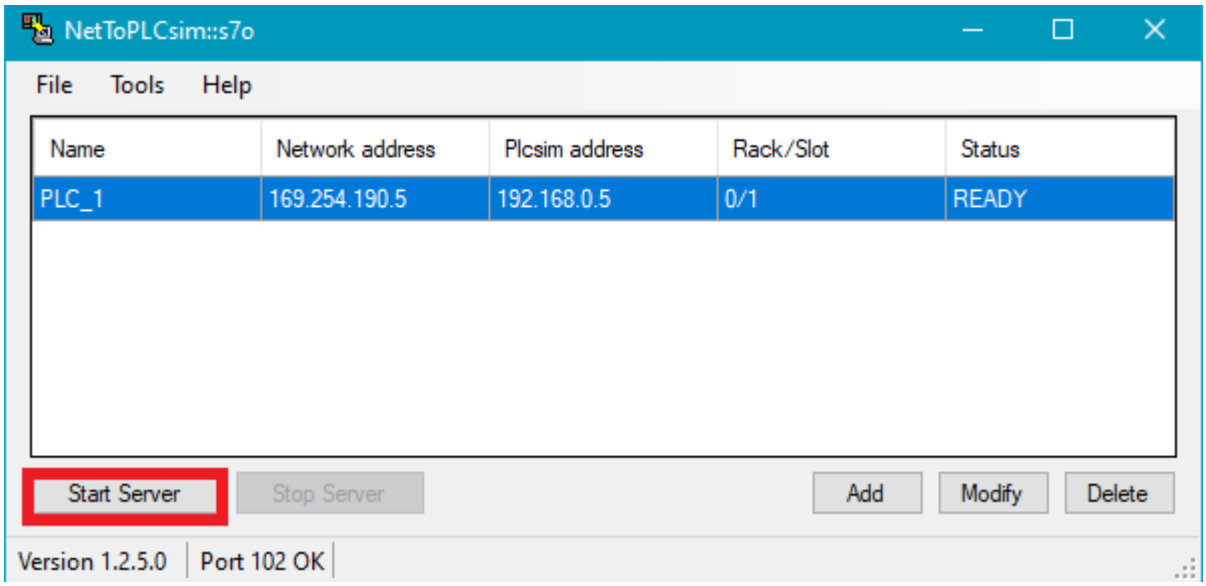


Figura 76. Lista de servidores y comienzo de la conexión

A continuación, se debe de abrir el archivo ejecutable generado, no es necesario abrir el programa NetBeans, compilar y ejecutar el código del programa. Haciendo doble clic en el elemento creado en la carpeta “dist” del proyecto que hace referencia la Figura 77.

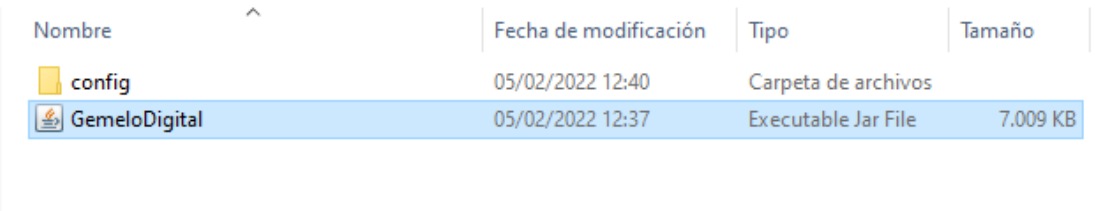


Figura 77.Archivo JAR del gemelo digital

Seguidamente, aparecerá una ventana emergente que corresponde al software de Processing y en el cual aparecerá la representación del gemelo digital cargado en el archivo XML. Al ejecutarse por primera vez, aparecerá la representación desde una vista muy cercana a los objetos como se ve reflejado en la Figura 78.

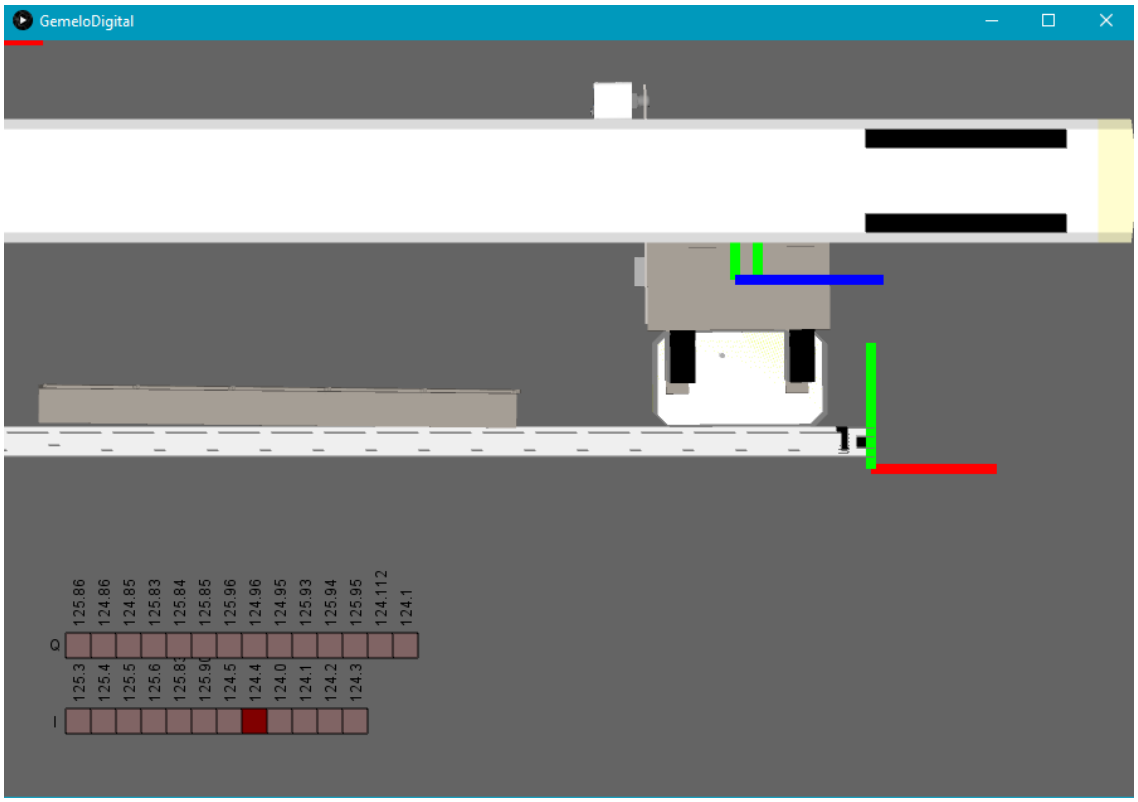


Figura 78.Primera vista de al ejecutar

Sin embargo, existe la posibilidad de desplazarlo y alejar la representación para adecuarla a la vista más adecuada. Estas acciones se podrán realizar a través de las letras de teclado “A” para desplazar a la izquierda, “W” para desplazar hacia arriba, “D” para desplazar hacia la derecha y “X” para desplazar hacia abajo. Con la rueda el ratón se aumenta o decrementa el zoom de la representación. Por último, para girar la vista se pulsará el botón CRL más hacer clic con el ratón y desplazar hacia la dirección que se desee de igual modo que en la Figura 79.

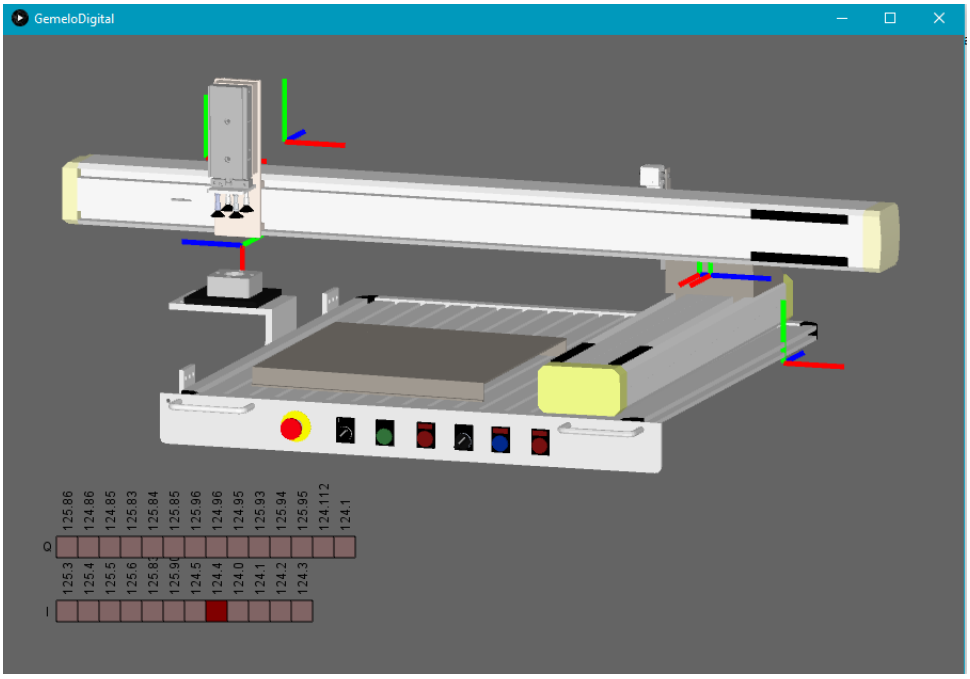


Figura 79. Vista adecuada de la representación

Una vez centrada la estación en la vista deseada, se procederá a comenzar con el funcionamiento del gemelo digital. Para ello se han de copiar los pasos a seguir igual que con la máquina.

En primer lugar, se arranca con la controladora en estado de alarma (se verá reflejado en la salida luminosa de alarma) debido a que la referencia no se ha ejecutado, para ello será necesario hacer un Home al eje cartesiano pulsando el botón REARME. Una vez pulsado, los motores comenzarán el movimiento hasta la posición inicial de referencia y se quedará ahí parado.

Seguidamente, para comenzar el funcionamiento se puede llevar a cabo mediante dos modos de actuación, en modo automático, en el cual comenzará el

funcionamiento de la máquina de forma continua sin ningún tipo de interrupción, o, por otro lado, en modo manual, que solamente realizará un ciclo de deposición. Para seleccionar dicho modo se deberá accionar el selector AUTOMÁTICO/MANUAL. En cualquiera de los casos, será necesario pulsar el botón START para comenzar el movimiento hacia la posición de recogida de pieza, donde se quedará parado.

Una vez comenzado, el robot procederá a moverse a la posición de recogida de pieza, expandir el cilindro hasta su final de carrera, activar el vacío y retraerse. Posteriormente, se desplazará a la posición calculada o indicada para depositarla mediante el mismo proceso.

Una vez iniciado el ciclo manual, se podrá actuar de dos maneras. La primera de ellas es simplemente mediante la pulsación del botón START una vez que está parado en la posición de recogida, que actuará siguiendo el contador interno de piezas. Otra opción es mediante la pantalla HMI asociada al proyecto, en la cual se podrá observar cada una de las posiciones de dejada del almacén, por lo tanto, simplemente pulsando en cualquiera de las posiciones, el ciclo se activará y el robot se desplazará hasta la posición pulsada. Véase la Figura 80 donde se puede observar dichas posiciones y un campo numérico que indica el número de piezas que refleja el contador interno.

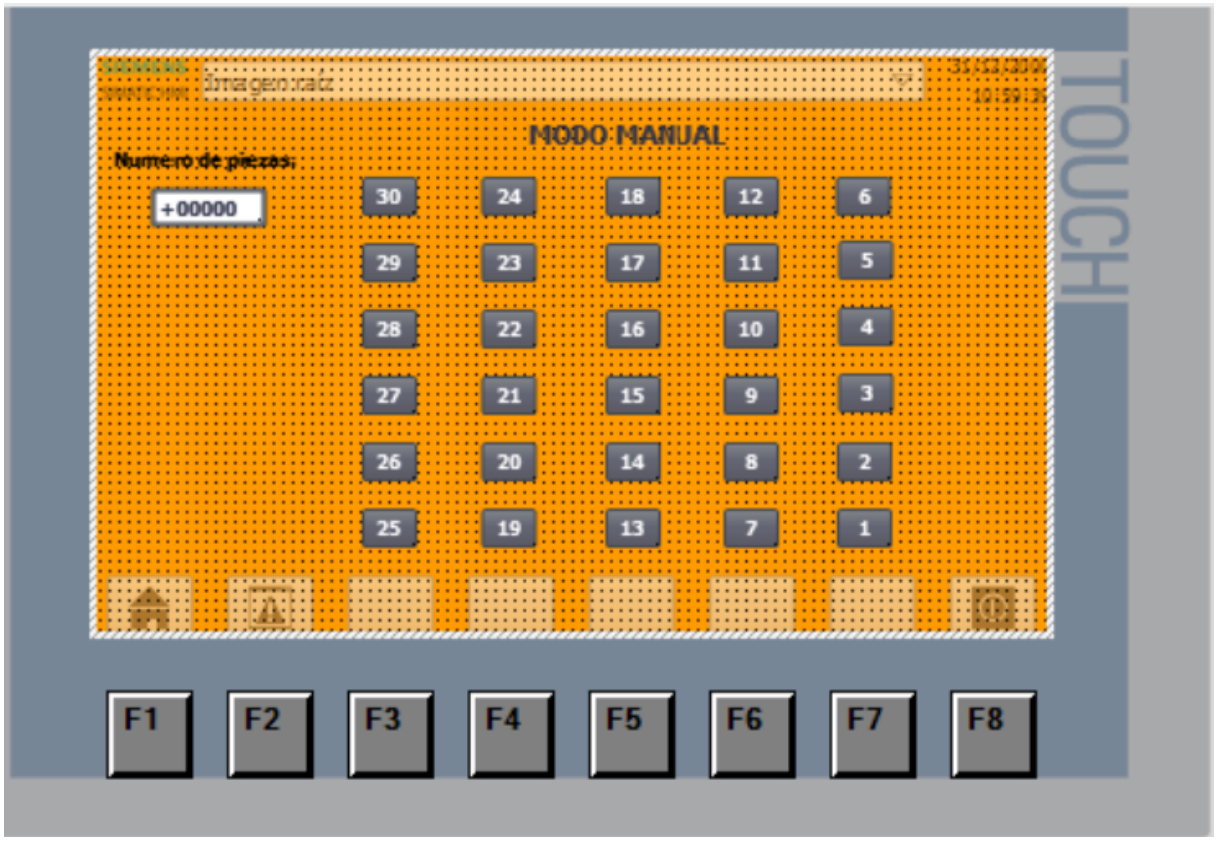


Figura 80. Pantalla HMI del proceso

6.4 ANEXO D: VIDEO EXPLICATIVO

En el anterior código QR se recoge un vídeo del proceso que es necesario seguir para ejecutar e interactuar con la estación a modo de ejemplo. En él se ve las diferentes posibilidades de movimiento que existen tanto de forma manual jugando con la activación de las diferentes salidas desde el panel de control, o interactuando mediante los botones de la estación. Véase en la Figura 78. Código QR del vídeo del proceso donde se incluye el acceso al video que describe el proceso.

Figura 78. Código QR del vídeo del proceso

7 Bibliografía

Autodesk Inventor. (2022). Obtenido de https://www.autodesk.es/products/inventor/new-features

Benkler, Y. (2012). *El Pingüino y el Leviatán: Por qué la cooperación es nuestra arma más valiosa para mejorar el bienestar de la sociedad.*

Calendamaia. (9 de Enero de 2014). *Genbeta*. Obtenido de Desarrollo de NetBeans: https://www.genbeta.com/desarrollo/netbeans-1

Enterprise, S. (s.f.). *SMC* . Obtenido de Librería CAD: https://www.smc.eu/es-es/productos/herramientas-de-ingenieria/libreria-cad-smc

Fernandez, D. P. (9 de Noviembre de 2018). *Tecnonucleous*. Obtenido de https://tecnonucleous.com/2018/11/09/que-es-xml/

Herranz, A. (26 de Mayo de 2021). *Digital twins: qué son, para qué sirven y cuáles son los beneficios y problemas de los gemelos digitales*. Obtenido de https://www.xataka.com/pro/digital-twins-que-sirven-cuales-beneficios-problemas-gemelos-digitales#:~:text=Un%20gemelo%20digital%20se%20crea,de%20an%C3%A1lisis%2C%20monitorizaci%C3%B3n%20y%20predicci%C3%B3n.

Mclibre. (1 de Mayo de 2018). Obtenido de https://www.mclibre.org/consultar/xml/lecciones/xml-conceptos-basicos.html

Mitsubishi. (s.f.). Obtenido de Servo Configuration software SETUP 161E.

Parejo, J. C. (Marzo de 2008). Obtenido de https://personal.us.es/jcortes/Material/Material_archivos/Articulos%20PDF/RepresentDH.pdf

Perasso, V. (12 de Octubre de 2016). Qué es la cuarta revolución industrial (y por qué debería preocuparnos). *BBC*, págs. https://www.bbc.com/mundo/noticias-37631834.

Processing. (s.f.). Obtenido de https://processing.org/examples/mousefunctions.html

Processing Reference. (s.f.). Obtenido de https://processing.org/reference/mouseWheel_.html

Processing References. (s.f.). Obtenido de https://processing.org/reference/pushMatrix_.html

Processing References. (s.f.). Obtenido de https://processing.org/reference/popMatrix_.html

SIEMENS. (s.f.). *SIEMENS*. Obtenido de https://new.siemens.com/es/es/empresa/historia/125-aniversario.html

SIEMENS. (s.f.). *SIEMENS Automatización*. Obtenido de https://new.siemens.com/es/es/productos/automatizacion.html

Silvia. (28 de Febrero de 2020). *¿Qué es la Industria 4.0?* | *7 Ejemplos Prácticos.* Obtenido de Trebolgroup: https://www.trebolgroup.com/que-es-la-industria-4-0-7-ejemplos-practicos/

SMC. (s.f.). *Sistema didáctico modular de ensamblaje flexible.* Obtenido de http://www1.udistrital.edu.co:8080/documents/138588/a7d694f2-2780-4774-ad28-a0f0b8ee6631

Snap7. (s.f.). Obtenido de Moka7: http://snap7.sourceforge.net/

Tutorialspoint. (s.f.). Obtenido de https://www.tutorialspoint.com/es/xml/xml_tags.htm

Univision. (2 de Abril de 2018). Obtenido de Uso de librerías y manejo de bases de datos en NetBeans: https://www.univision.com/explora/uso-de-librerias-y-manejo-de-bases-de-datos-en-netbeans

Vaquero, B. A. (s.f.). Obtenido de https://borjaarandavaquero.com/que-es/xml/

Velasco, J. G. (2009). *Energías renovables.* Editorial Reverte.