Universidad Zaragoza
1542

Trabajo Fin de Grado

Automatización de un proceso de fabricación flexible utilizando un robot colaborativo

Automation of a flexible manufacturing process using a collaborative robot

Autor

Alfonso Colás Fraile

Director

Pedro Pablo Huerta Abad

Escuela Universitaria Politécnica La Almunia

2021

Escuela Universitaria Politécnica - La Almunia
Centro adscrito
Universidad Zaragoza

ESCUELA UNIVERSITARIA POLITÉCNICA
DE LA ALMUNIA DE DOÑA GODINA (ZARAGOZA)

MEMORIA

Automatización de un proceso de fabricación flexible utilizando un robot colaborativo

Automation of a flexible manufacturing process using a collaborative robot

424. 20. 16

Autor: Alfonso Colás Fraile

Director: Pedro Pablo Huerta Abad

Fecha: 18 de Septiembre de 2021

INDICE BREVE

INDICE DE CONTENIDO

INDICE DE ILUSTRACIONES

INDICE DE TABLAS

1. RESUMEN

Durante estos últimos años, se está produciendo la cuarta revolución industrial, que busca transformar la empresa en una organización inteligente para optimizar los resultados de negocio. Para ello las empresas deben poder adaptarse de forma rápida a las necesidades del mercado. Esto se consigue usando sistemas de fabricación flexible, los cuales permiten la producción de forma automática de series más cortas de producto, pudiendo adaptarse a la demanda sin tener un coste económico muy elevado.

En este proyecto se resuelve el cambio de hábitos de los consumidores creando una pequeña línea de producción flexible, en la cual un operario mediante un terminal programable o HMI decide si la operación a realizar es el despaletizado o el paletizado. Si se realiza este último se deberá seleccionar el tipo de palet a rellenar y las diferentes piezas que usará en cada posición. Esta información se transfiere al PLC, el cual la procesa y la envía al Robot Colaborativo o Cobot. Este último, mediante visión artificial localiza el palet, comprueba el estado del mismo y realiza la operación ordenada por el operario.

1.1. PALABRAS CLAVE

Interface hombre-máquina (HMI), Controlador lógico programable (PLC), Robot colaborativo (Cobot) y sistema de fabricación flexible (FMS).

2. ABSTRACT

During these last years, the fourth industrial revolution is taking place, which seeks to transform the company into an intelligent organization to optimize business results. For this, companies must be able to adapt quickly to market needs. This is achieved using flexible manufacturing systems, which allow the automatic production of shorter series of products, being able to adapt to demand without having a very high economic cost.

This project solves the change in consumer habits by creating a small flexible production line, in which an operator through a programmable terminal or HMI decides whether the operation to be carried out is a depalletized or a palletized. If the latter is carried out, the type of pallet to be filled and the different pieces that will be used in each position must be selected. This information is transferred to the PLC, which processes it and sends it to the Collaborative Robot or Cobot. The latter, using artificial vision, locates the pallet, checks its status and performs the operation ordered by the operator.

3. INTRODUCCIÓN

En la actualidad se está produciendo la cuarta revolución industrial, igual que a finales del siglo XVII la máquina de vapor representó una revolución industrial. Esta vez, serán los robots integrados en sistemas ciberfísicos los responsables de una transformación radical. Esta revolución no se define por un conjunto de tecnologías emergentes en sí mismas, sino por la transición hacia nuevos sistemas que están construidos sobre la infraestructura de la revolución digital. El principio básico es que las empresas podrán crear redes inteligentes que se controlaran a sí mismas, a lo largo de toda la cadena de valor.[1]

Todo esto junto con la variabilidad de las demandas en el mercado ha provocado que los sistemas de fabricación flexibles (FMS) sean una necesidad en muchas empresas. Los podemos definir como un sistema de fabricación diseñado para que las áreas de producción puedan ser modificadas a menudo, ajustando la mano de obra y los materiales, consiguiendo así satisfacer mejor la demanda del mercado. Hay varias categorías de flexibilidad.

La primera categoría se basa en la flexibilidad de la máquina, la cual cubre la capacidad del sistema para ser modificado, consiguiendo así producir nuevos tipos de productos, y la capacidad de cambiar el orden de operaciones que se ejecutan en una pieza. La segunda categoría se denomina flexibilidad de asignación, que consiste en la capacidad de utilizar varias máquinas para realizar la misma operación en una parte, así como la capacidad del sistema para absorber los cambios a gran escala, como en volumen o capacidad.

Las principales ventajas de un FMS es su alta flexibilidad en la gestión de los recursos de fabricación como el tiempo y el esfuerzo necesario para fabricar un nuevo producto. La mejor aplicación de un FMS se encuentra en la producción de pequeños lotes de productos.[2]

En este proyecto se automatiza una pequeña línea de producción flexible compuesta por un HMI, un PLC y un Cobot, para realizar el paletizado y despaletizado de diferentes tipos de piezas y con diferentes pallets. Con ello pretendemos resolver el problema de la personalización de productos para ajustarlos a los cambios de hábitos de los consumidores.

Para ello un operario deberá de realizar la selección del proceso en el HMI, indicando si desea realizar un paletizado o despaletizado.

En primer lugar explicaré la selección del despaletizado, la cual es la más sencilla. El Cobot mediante la visón realiza una foto del pallet y comprueba los huecos llenos y el tipo de pieza que los ocupa, y después realiza el despaletizado, dejando las piezas sobre una caja colocada en un punto predefinido. El despaletizado se realiza por líneas, debido a que el picking de las piezas es mucho más preciso y da menos errores.

En segundo lugar explicaré la selección del paletizado, el operario tiene que seleccionar el tipo de pallet, ya sea de doce o de trece posiciones. Después debe de seleccionar la pieza que será colocada en cada posición, en función de su forma y su color. Una vez seleccionado, el cobot realiza una foto del pallet para comprobar que está vacío, sino es así, se realiza un despaletizado. En el siguiente paso realiza una foto a las piezas y comprueba la forma y el color de cada una de ellas. Con estos datos va cogiendo y dejando las piezas necesarias en cada posición del palet.

Con todo ello tenemos una línea de producción completa y flexible, adaptable a las necesidades de los consumidores y usando un robot colaborativo, con todas ventajas que esto conlleva, ya que podemos trabajar cerca suyo e incluso realizar operaciones en común, para por ejemplo, retirarle un pallet completado por otro nuevo.

4. FUNDAMENTOS TEÓRICOS Y ESTADO DEL ARTE

4.1. INDUSTRIA 4.0

El término industria 4.0 se refiere a un nuevo modelo de organización y de control de la cadena de valor a través del ciclo de vida del producto y a lo largo de los sistemas de fabricación apoyado y hecho posible por las tecnologías de la información.

El término industria 4.0 se utiliza de manera generalizada en Europa, si bien se acuñó en Alemania. También es habitual referirse a este concepto con términos como "Fábrica Inteligente" o "Internet industrial". En definitiva se trata de la aplicación a la industria del modelo "Internet de las cosas" (IoT). Todos estos términos tienen en común el reconocimiento de que los procesos de fabricación se encuentran en un proceso de transformación digital, una revolución industrial producida por el avance de las tecnologías de la información y, particularmente, de la informática y el software.[3]

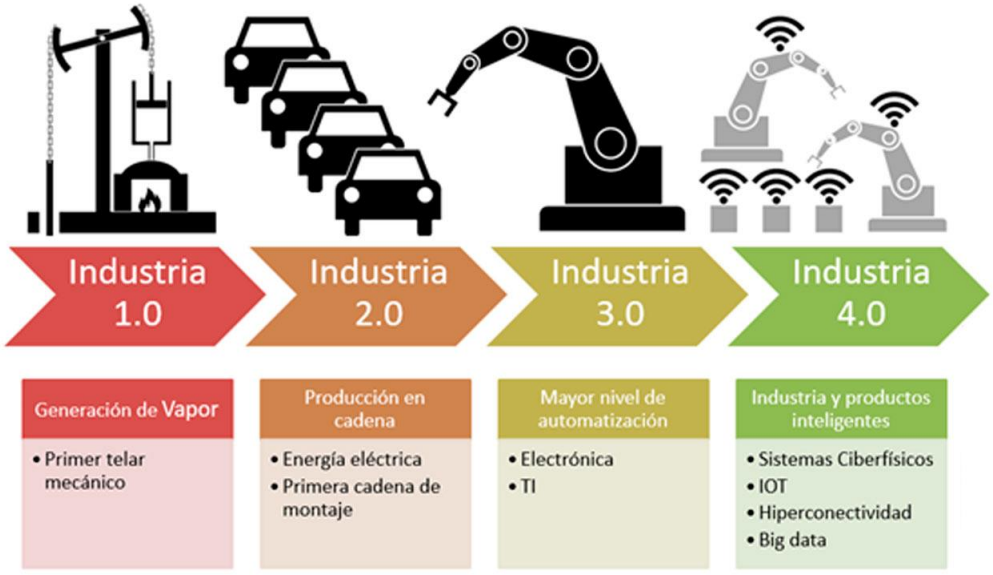


Ilustración 1. Diagrama evolución de la industria. [3]

4.1.1. Cuarta revolución industrial

n la primera Revolución Industrial, entre los siglos XVIII y XIX, se mecanizaron los procesos de producción, transformando la economía agraria y artesanal en otra liderada por la industria. En la segunda transición, en el siglo XX, trajo la producción en serie, con la aparición de fábricas y líneas de montaje que permitieron fabricar productos para el gran consumo. El final del Siglo XX trae una nueva transformación, el despliegue de la electrónica y la informática en los procesos industriales permitió automatizar las líneas de producción y que las máquinas reemplazaran a las personas en tareas repetitivas. Dos décadas de vertiginosos avances en la tecnología de Internet han producido un impacto radical en la economía y en la sociedad. La convergencia de las tecnologías de la información con la sensórica y la robótica están transformado el internet tradicional, información y personas, en el internet de las cosas. Y este nuevo escenario aplicado a la industria ha producido un impacto disruptivo en ésta, abriendo un escenario de enormes oportunidades basado en el aprovechamiento de la informática.[3]

4.1.2. Pilares de la inteligencia en la industria 4.0

4.1.2.1. Soluciones inteligentes

Los productos inteligentes se caracterizan por disponer de electrónica, software embebido y conectividad lo que, en conjunto, le dotan de nuevas características, capacidades y funciones. Se les denomina sistemas ciber-físicos (CPS) y son los "habitantes" del ecosistema de la Internet de las cosas (IoT). La conectividad les proporciona capacidad de comunicación máquina a máquina (M2M) e interacción con humanos. El software les permite auto-gestionarse y tomar decisiones descentralizadas. Equipados con sensores captan información sobre su entorno y sobre su propio uso y estado, datos que pueden proporcionar a quien lo fabricó o gestiona su servicio.

Estos mismos elementos se aplican no sólo a los productos sino a las máquinas que los fabrican, los sistemas de producción ciber-físicos(CPPS), que conforman la "Fábrica Inteligente". Son máquinas con gran capacidad de comunicación M2M que ofrecen personalización, adaptación al entorno y a tareas nuevas. Gracias a su autogestión, productos y máquinas inteligentes se vuelven invisibles a los operadores y sólo precisan atención cuando precisan mantenimiento.

Además, la comunicación M2M les permite auto-configurarse para adaptar su funcionalidad en tiempo real a las necesidades del cliente a lo largo de su ciclo de vida. Con ello se hace posible mejorar la experiencia del usuario, intensificar la interacción con el cliente y generar nuevos servicios añadidos.

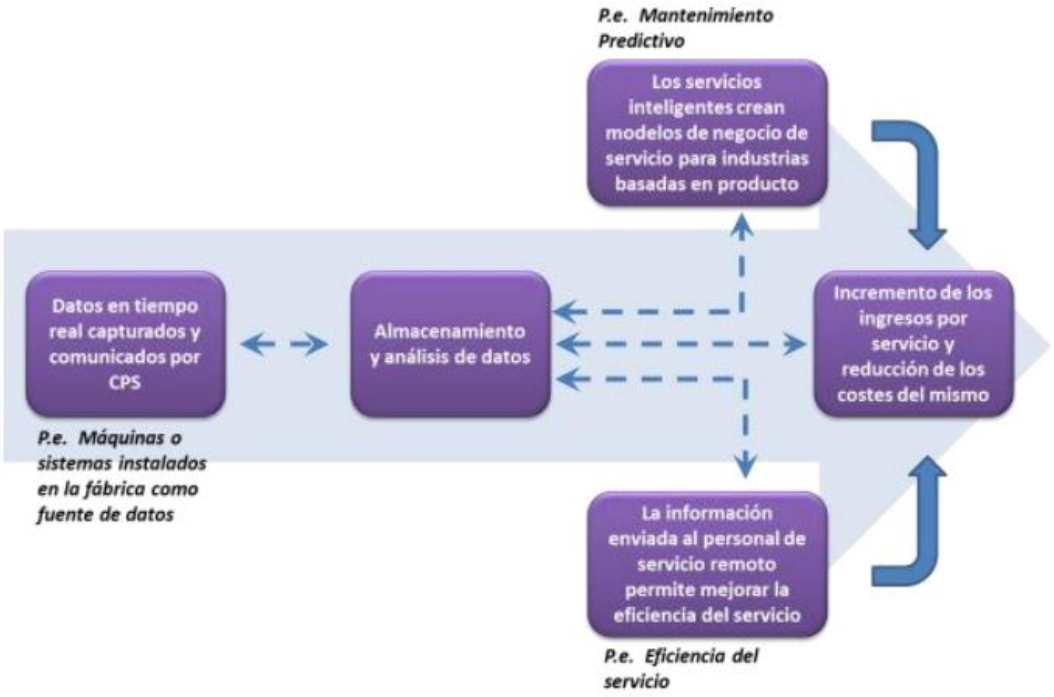


Ilustración 2. Servicio inteligente. [3]

Los servicios inteligentes permiten ofrecer servicios innovadores y establecer nuevos modelos de negocio, por ejemplo modelos de pago por uso o servicio. La comunicación con el fabricante, la recogida de grandes cantidades de datos y su análisis es la base para generar nuevas ofertas de servicios y optimizar los modelos existentes. Los modelos analíticos aplicados a esos datos (Big Data) pueden automatizar la toma de decisiones. Por ejemplo, predecir el momento en que un sistema requerirá mantenimiento. Los fabricantes podrán aprovechar combinaciones innovadoras de servicios inteligentes para incrementar su creación de valor, aguas arriba o abajo de la cadena de valor.[3]

4.1.2.2. Innovación inteligente

La conectividad permite extender la innovación a toda la empresa apoyándose en la información que fluye desde y hacia la fábrica. Apoyándose en soluciones informáticas como comunidades virtuales o herramientas PLM ("Product Life Management") colaborativas, los procesos de innovación se abrirán a socios y clientes, potenciándose la orientación al cliente de la industria. La colaboración con clientes y socios acelerará el flujo de innovación y reducirá los tiempos de comercialización.

La innovación a lo largo del Ciclo de Vida del producto inteligente y conectado combina la capacidad analítica de las herramientas informáticas con los datos, cada vez más ricos, proporcionados por el producto inteligente a lo largo de su ciclo de vida. Combinando los datos recogidos del producto inteligente (CPS), de las máquinas (CPPS) y de los clientes se tomarán decisiones para optimizar la fabricación, los servicios y la experiencia del cliente. En la base de todo ello estarán sistemas PLM avanzados, interconectados y con sistemas de análisis y visualización potentes e intuitivos.[3]

4.1.2.3. Cadenas de suministro inteligente

Las cadenas de suministro inteligentes estarán altamente automatizadas e integradas y, de nuevo, serán posibles gracias a la integración del software y las comunicaciones en la industria.

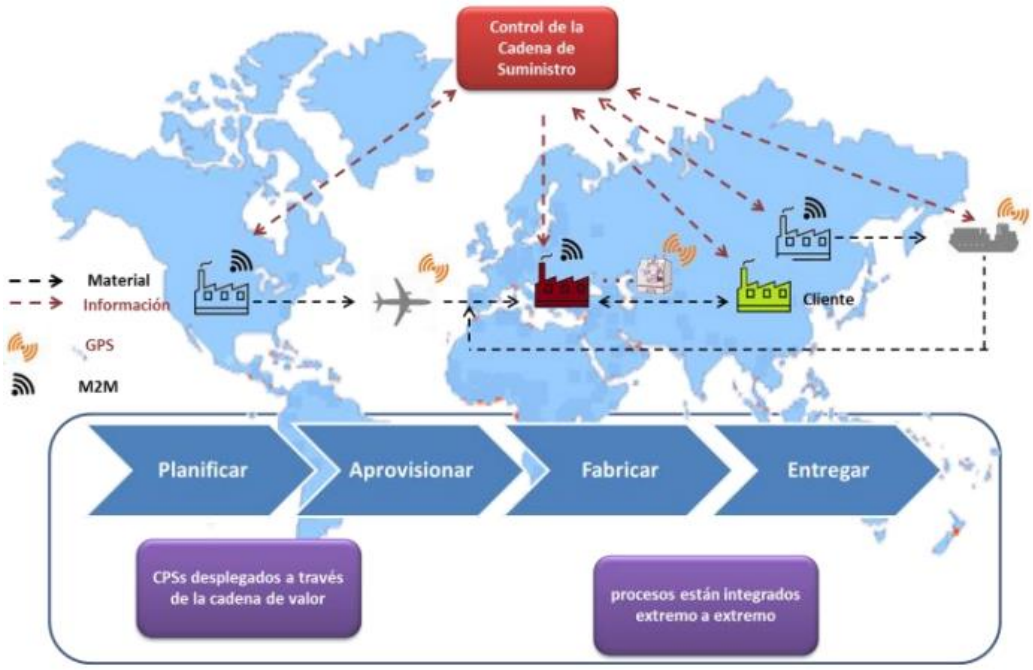


Ilustración 3. Cadena de suministro conectada. [3]

En lugar de la integración horizontal habitual en la industria hoy, la colaboración entre empresas en la Industria 4.0 se basará en configuraciones "ad-hoc" para ofrecer soluciones a medida de cada cliente. Usando redes de colaboración ágiles la industria puede aprovechar las oportunidades de un mercado globalizado de habilidades y capacidades. Por ejemplo, un fabricante podrá decidir con flexibilidad qué externalizar o hacer "in house", podrá trabajar con proveedores de servicios de ingeniería a través de plataformas CAD compartidas o asignar órdenes de producción al proveedor con más capacidad libre disponible en cada momento.

La base para estas redes son entornos de producción y plataformas de ingeniería conectadas en red junto con interfaces entre empresas. También en este aspecto la base es la informática y el software será decisivo y buena muestra de ello es el liderazgo de SAP en el impulso de la industria 4.0.

La cadena de suministro conectada es otra pieza central en toda estrategia de Industria 4.0. Para gestionar la creciente complejidad de las cadenas de suministro, los flujos físicos se replican en plataformas digitales.

Esta imagen virtual de la red de suministro se crea a través de materiales y piezas etiquetadas con RFID. A lo largo de la cadena de suministro, los CPS generan datos en tiempo real sobre su posición y estado. Esta digitalización permite automatizar los procesos de la cadena de suministro e identificar al producto a lo largo del proceso de producción permitiendo al fabricante ser más sensible a cambios en los pedidos. La visibilidad de los movimientos de la red de suministro proporciona transparencia. Permite reconocer ineficiencias y riesgos, aumentarla robustez y la capacidad de respuesta a incidencias, incrementar la fiabilidad y disminuir los costes.[3]

4.1.2.4. Fábrica inteligente

La fábrica inteligente es el cuarto pilar de la industria 4.0. Está formada por unidades de producción inteligentes (CPPS) vinculadas al ecosistema de fabricación, del que conocen su estado y limitaciones. Como cada módulo es capaz de obtener la información que necesita, la fábrica se convierte en una red de agentes que toman decisiones optimizadas a nivel local. La producción podría organizarse según un modelo de oferta-demanda donde la capacidad de los sistemas es la oferta y la demanda surge de las órdenes que deben atenderse. Cada CPPS podría decidir su programa de producción (en base a su tiempo de procesamiento, las fechas de entrega u objetivos de beneficio o sostenibilidad).

Este Control de Producción descentralizado ofrece la posibilidad de fabricar cada producto de manera individual sin costes adicionales y con fechas de entrega de gran fiabilidad. Además, la captura masiva de datos relacionados con la producción y su análisis permitirán alcanzar niveles desconocidos hasta el momento de productividad y calidad del producto.[3]

Ilustración 4. Industria 4.0.[4]

4.2. SISTEMAS DE FABRICACIÓN FLEXIBLES (FMS)

Para poder tener una mejor perspectiva de qué es una línea de producción flexible debemos recordar que la fabricación de piezas mecánicas sometidas a varios procesos, involucraban complejos sistemas de control y producción, tales como, proveer materias primas, materiales y órdenes de trabajo, entre otros. Uno de los principales problemas consistía en el cambio y ajuste de herramientas de trabajo, lo que evidentemente imposibilitaba poder obtener altos índices de productividad, debido a los tiempos de recambio de piezas, cambios de formato de máquinas, ajuste y reprogramación de proceso de máquina.

Con la introducción de nuevos sistemas de control, gracias a los avances de la informática, se obtuvieron mejoras en la eficiencia de la fabricación, tales como el diseño del producto, la maquinaria, la planificación del proceso, la disponibilidad de materiales, el control de la producción, la automatización, etc.

Derivado de lo anterior, estamos en mejor posición para afirmar que la fabricación flexible no es simplemente un concepto aislado sino más bien es la conjunción de tecnología y esfuerzo humano integrados indudablemente en un equipo seleccionado de alta tecnología. Su finalidad es responder de forma flexible a cualquier cambio que se presente, es decir adaptándose al cambio rápidamente.

Este concepto nace de la necesidad continua de mejora que se establece en empresas Japonesas como Toyota, en donde logra imponerse, no solo un modelo de fabricación, sino un proceso con excelentes resultados en la fabricación.[5]

4.2.1. Objetivos

Los objetivos que se desean alcanzar con un sistema de fabricación flexible son los siguientes:

- Una marcada tendencia en la reducción de los costos de fabricación, al eliminar operaciones innecesarias, transporte materiales y producto terminado, desperdicio de materiales y disminución efectiva de piezas defectuosas.
- Un incremento sustancial de los indicadores de productividad, al incrementar los volúmenes de fabricación significativamente.
- Un grado significativo de calidad del producto terminado.
- Mejora el grado de satisfacción del cliente, al proporcionársele un producto de alta calidad y en tiempo.
- Reduce significativamente el espacio necesario o área de trabajo necesaria para la operación de equipo y maquinaria.
- Puede llevarse fácilmente al sistema JIT (Justo a tiempo) lo que puede eliminar o disminuir significativamente las áreas de almacenamiento de producto terminado y materias primas.[5]

4.3. CONTROLADOR LÓGICO PROGRAMABLE (PLC)

4.3.1. Definición y principales características

Un controlador lógico programable, más conocido por sus siglas en inglés PLC (Programmable Logic Controller), es un equipo electrónico, que utiliza memoria programable para guardar instrucciones para la implementación de determinadas funciones, como operaciones lógicas, secuencias de acciones, especificaciones temporales, contadores y cálculos para el control mediante módulos de entradas y salidas analógicos o digitales sobre diferentes tipos de máquinas y de procesos.[6]

Ilustración 5. PLC NX1.[7]

El campo de aplicación de los PLCs es muy diverso e incluye diversos tipos de industrias (ej. automoción, aeroespacial, construcción, etc.), así como de maquinaria. A diferencia de los ordenadores de propósito general, el PLC está diseñado para múltiples señales de entrada y de salida, amplios rangos de temperatura, inmunidad al ruido eléctrico y resistencia a la vibración y al impacto. Los programas para el control de funcionamiento de la máquina se suelen almacenar en memorias protegidas por baterías o en memorias no volátiles.

Dentro de las ventajas que estos equipos poseen se encuentran que, gracias a ellos, es posible realizar operaciones en tiempo real, debido a su reducido tiempo de reacción. Además, son dispositivos que se adaptan fácilmente a nuevas tareas debido a su flexibilidad a la hora de programarlos, reduciendo así los costos adicionales a la hora de elaborar proyectos. Permiten también una comunicación de alta velocidad con otro tipo de controladores y ordenadores e incluso permiten realizar las operaciones en red. Como ya se ha mencionado previamente, tienen una construcción estable al estar diseñados para poder funcionar en condiciones adversas de vibraciones, temperatura, humedad y ruidos. Son fácilmente programables por medio de lenguajes de programación bastante comprensibles. Sin embargo, presentan ciertas desventajas como la necesidad de contar con técnicos cualificados para ocuparse de su buen funcionamiento.[6]

4.3.2. Estructura general de los PLCs

El siguiente diagrama de flujo es una representación de los diferentes componentes y la estructura de la que está compuesta un controlador lógico programable.

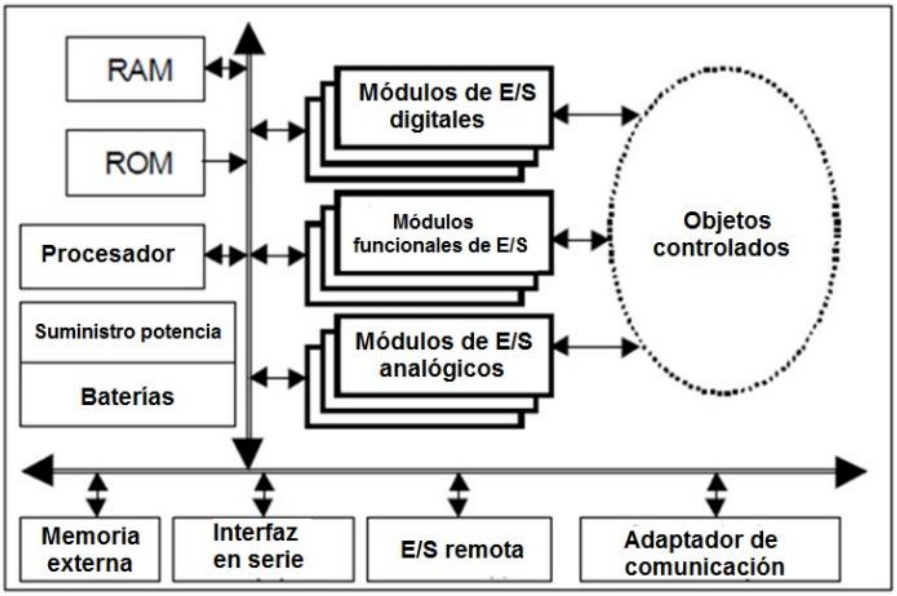


Ilustración 6. Diagrama generalizado de un PLC. [6]

Como puede observarse en la figura, para que el sistema funcione es necesario que exista un suministro de energía cuyo propósito principal es garantizar los voltajes de operación internos del controlador y sus bloques. Los valores más frecuentemente utilizados son ±5V, ±12V y ±24V y existen principalmente dos módulos de suministro de potencia: los que utilizan un voltaje de entrada de la red de trabajo y los que utilizan fuentes de alimentación para el control de los objetos.

La parte principal es la denominada "unidad central de procesamiento" o CPU que contiene la parte de procesamiento del controlador y está basada en un microprocesador que permite realizar operaciones aritméticas y operaciones lógicas para realizar diferentes funciones. Además, la CPU, testea continuamente su funcionamiento para detectar errores en tiempo real.

La transferencia de datos y/o direcciones en los PLCs es posible gracias a cuatro tipos de buses diferentes:

- Bus de datos, para la transferencia de datos de los componentes individuales Controladores Lógicos Programables (PLCs).
- Bus de direcciones, para aquellas transferencias entre celdas donde se habían guardado datos.
- Bus de control, para las señales de control de los componentes internos.
- Bus de sistema, para conectar los puertos con los módulos de E/S.

El lugar donde se guardan los datos y las instrucciones es la memoria que se divide en memoria permanente, PM, y memoria operacional, conocida como memoria de acceso aleatorio o RAM. La primera, la PM, se basa en las ROM, EPROM, EEPROM o FLASH; es donde se ejecuta el sistema operativo del PLC y puede ser reemplazada. Sin embargo, la RAM, es donde se guarda y ejecuta el programa en cuestión utilizado y es la de tipo SRAM la que se utiliza habitualmente. La condición común para las entradas de dos componentes digitales de un PLC se guarda en una parte de la RAM y se denomina tabla PII o entrada imagen de proceso. La salida controlada, o el último valor de la salida calculada por las funciones lógicas, se guardan en la parte de la RAM denominada tabla PIO, salida de la imagen del proceso.

Finalmente, los módulos de E/S, son aquellos módulos de señal (SM) que coordinan la entrada y salida de las señales, con aquellas internas del PLC. Estas señales pueden ser digitales (DI, DO) y analógicas (AI, AO), y provienen o van a dispositivos como sensores, interruptores, actuadores, etc.[6]

4.3.2.1. Componentes del hardware

Una PLC puede contener un casete con una vía en la que se encuentran diversos tipos de módulos, como puede observarse en la siguiente figura, correspondiente a una PLC de la empresa Siemens.[6]

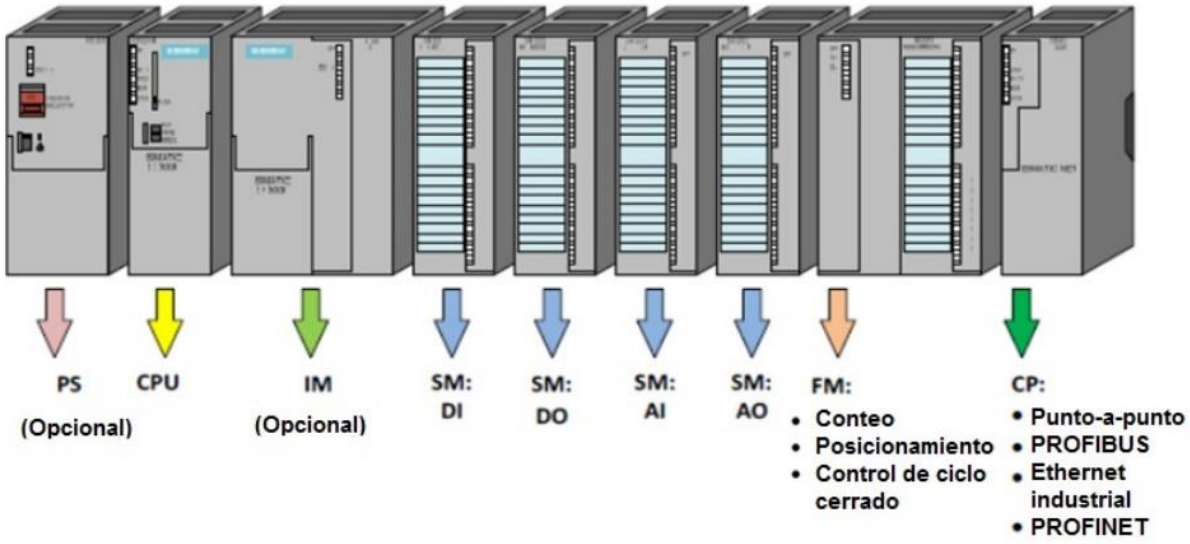


Ilustración 7. Organización modular del PLC Siemens S7-300. [6]

Como puede observarse en la figura, el PLC dispone de los siguientes módulos que, aunque en este tipo no puede ser intercambiada, esto sí es posible para PLCs de otras compañías. Los módulos más importantes son:

- Módulo de interfaz (IM), conecta diferentes modulos individuales con un único PLC.
- Módulo funcional (FM), procesamiento complejo en tiempo-crítico de procesos independientes de la CPU, por ejemplo, conteo rápido.
- Regulador PID o control de la posición.
- Procesador de comunicaciones (CP), conecta el PLC en una red de trabajo industrial, ej. Industrial Ethernet, PROFIBUS, AS – interfaz, conexión serie punto-a-punto.
- Interfaz hombre-máquina (HMI), ej. panel de operaciones.
- Entradas/salidas remotas.
- Módulos de señal de alta-velocidad.

- Cada módulo de PLC tiene su propia interfaz-HIM básica, utilizada para la visualización de los errores y las condiciones de comunicación, la batería, entradas/salidas, operación de los PLC, etc. Pequeños displays de cristal líquido (LCD) o diodos LED también se utilizan como interfaz-HMI.[6]

4.4. INTERFACE HOMBRE-MÁQUINA (HMI)

Un sistema HMI, que es abreviatura en ingles de Human Machine Interface, es una interfaz de usuario o panel de control que conecta a las personas con una máquina, sistema o dispositivo. Aunque el término puede aplicarse técnicamente a cualquier pantalla que permita al usuario interactuar con un dispositivo, la HMI se utiliza más comúnmente en el contexto de los procesos industriales que controlan y monitorean máquinas de producción. Por otro lado, HMI es la abreviatura de Human Machine Interface. De igual manera, entre profesionales se suele emplear su traducción al castellano Interfaz Hombre-Máquina para denominar a este tipo de paneles de operador.

Las pantallas HMI se utilizan para optimizar un proceso industrial digitalizando y centralizando los datos. De esta manera, los operadores pueden ver información importante en gráficos, cuadros de mando digitales o ver y gestionar alarmas.

La interfaz hombre-máquina se comunica con los controladores lógicos programables (PLC) y los sensores de entrada/salida para obtener y mostrar información para que los usuarios la vean. Del mismo modo, pueden utilizarse para una sola función, como el monitoreo y el seguimiento, o para realizar operaciones más sofisticadas, como el apagado de máquinas o el aumento de la velocidad de producción, dependiendo de cómo se implementen.[8]

Ilustración 8. HMI NA Omron.[9]

4.5. ROBOT COLABORATIVO (COBOT)

Los robots colaborativos, también conocidos como Cobots, han sido diseñados para trabajar junto a operarios humanos, ayudando a estos en las tareas que resultan más pesadas y de mayor precisión. Gracias a los contenidos precios de los cobots, su alta adaptabilidad a cualquier tarea, y su multitud de elementos "plug and play", están logrando que la pequeña-mediana empresa comience a adoptar esta tecnología. Por todo ello, los analistas creen que este campo experimentará un gran crecimiento en un futuro próximo.

Existen muchas razones por las cuales han aparecido los robots colaborativos, ya que pueden estar colocados junto a humanos en las mismas líneas de montaje empleando pequeños espacios, las medidas de seguridad necesarias son mucho menores que en robots simples, son muy flexibles para trabajar en remesas cortas sin necesidad de perder mucho tiempo en programarlos y son idóneos para reemplazar a operarios en tareas repetitivas o con problemas de ergonomía.

Los robots industriales poseen jaulas de seguridad para proteger a los humanos de estas máquinas. Sin embargo, los cobots, que existen en multitud de tamaños, poseen unas medidas de seguridad que pueden no requerir el uso de jaulas, para ello tienen sensores de seguridad, su geometría es redondeada y suave para evitar lesiones, etc. Pero su principal característica es que en cada una de las articulaciones del brazo robótico existen sensores diseñados para detectar fuerzas de impacto, por pequeñas que sean, y reaccionar rápidamente. [10]

Ilustración 9. Espacio colaborativo de trabajo.[11]

Para aumentar la seguridad de los Cobots se implementa el monitoreo de fuerza y velocidad, ya que cuando estos detectan a una persona dentro de su zona de trabajo, el cobot pasa a modo colaborativo reduciendo la velocidad y la fuerza, creando así un espacio de trabajo seguro.

Las normas de seguridad ISO 10218-1, ISO 10218-2 y la especificación técnica ISO TS-15066 define las funciones de seguridad y desempeño del robot colaborativo. Bajo TS-15066, la fuerza y el monitoreo de velocidad del cobot se establece en base a los datos de la aplicación, área de contacto humano y peligros del espacio de trabajo. El contacto humano se define en dos tipos: transitorio y cuasi estático. El primero se refiere al contacto que no es de sujeción, mientras que el último involucra situaciones que pueden causar agarre de una parte del cuerpo.

Los robots colaborativos realizan tareas automatizadas alrededor de otros equipos que potencialmente son peligrosos y pueden causar daño. El área en el que un cobot opera, incluyendo cualquier herramienta o equipo adicional, se conoce como el espacio colaborativo de trabajo. Según lo definido por ISO 10218 / ANSI RIA 15.06, este es el espacio dentro del área protegida donde el robot y el humano pueden realizar tareas simultáneamente durante las operaciones de producción. De manera similar, TS 15066 lo define como el área dentro del espacio operativo donde el robot puede realizar tareas al mismo tiempo que un operador humano durante la producción.

Por lo tanto no solo es importante evaluar el peligro del cobot, sino de todos los complementos que posea, por ejemplo, un gripper. Por ello el espacio colaborativo debe de estar claramente marcado. [12]

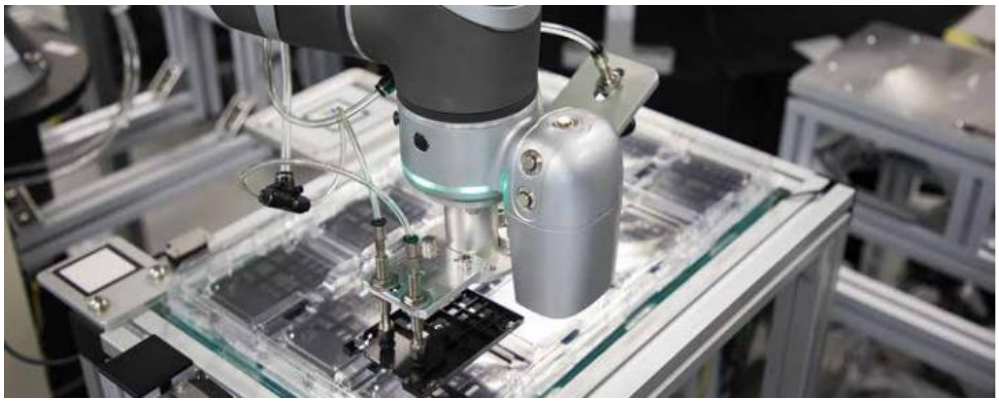


Ilustración 10. Cobot Omron TM.[13]

Los robots colaborativos están ganando popularidad debido a la reducción de precio de los sensores, lo cual influye directamente en el precio de los cobots. Esto permite acceder a este servicio a empresas de cualquier tamaño, sumado a su facilidad de uso y programación, lo convierte en un producto muy atractivo.[10]

4.6. REDES INDUSTRIALES

Las comunicaciones industriales son aquellas que permiten el flujo de información del controlador a los diferentes dispositivos a lo largo del proceso de producción: detectores, actuadores, sensores entre otros. Dada la gran variedad de sistemas de comunicación entre equipos industriales, de los cuales la mayoría son cerrados, se ha optado por el desarrollo de un entorno que permita la implementación de protocolos de especificaciones conocidas, en un sistema de comunicación completo, desde el medio físico hasta el nivel más alto de red, con el conocido modelo CIM (Computer Integrated Manufacturing). En la industria este concepto corresponde a una estructura piramidal jerarquizada, produciéndose en la cúspide decisiones de política empresarial. En la base lo que se pretende es que las denominadas islas de automatización (autómatas programables, máquinas de control numérico, robots) se integren en un sistema de control jerarquizado y distribuido que permita la conversión de decisiones de política empresarial, en operaciones de control de bajo nivel.[14]

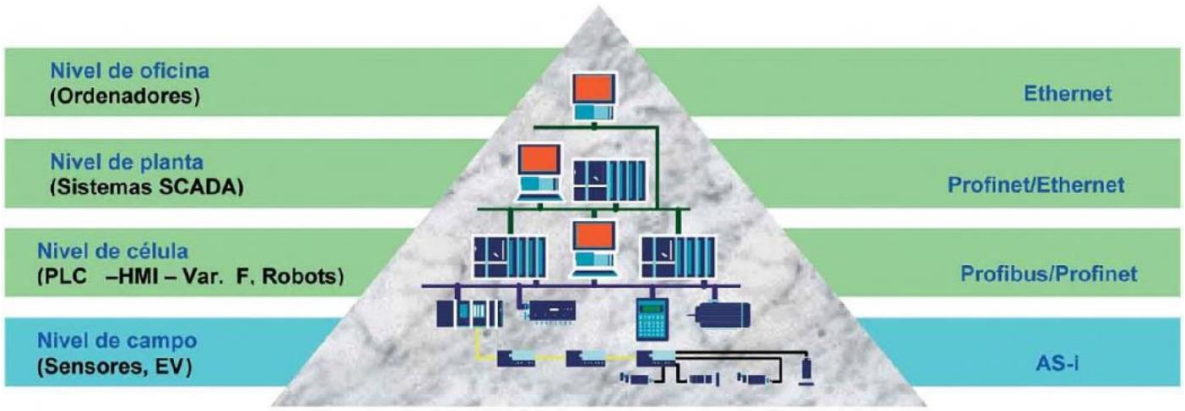


Ilustración 11. Pirámide modelo CIM. [14]

A continuación vamos a explicar las principales redes de comunicación industriales estandarizadas.

4.6.1. AS-Interface

El bus AS-Interface o Interfaz de Actuador/Sensor, es un sistema de enlace para el nivel más bajo del procesos en instalaciones de automatización. Los mazos de cables utilizados hasta ahora en este nivel son reemplazados por un único cable eléctrico, el cable AS-i. Por medio del cable AS-i y del maestro AS-i se acoplan sensores y actuadores binarios de la categoría más simple a las unidades de control a través de módulos AS-i en el nivel de campo.

AS-Interface presenta varias características fundamentales, las cuales son las siguientes:

- AS-Interface es idóneo para la conexión de actuadores y sensores binarios. A través del cable AS-i tienen lugar tanto el intercambio de datos entre sensores/actuadores (esclavos AS i) y el maestro AS-i como la alimentación eléctrica de los sensores y los actuadores.
- Cableado sencillo y económico; montaje fácil con técnica de perforación de aislamiento; gran flexibilidad gracias al cableado tipo árbol.
- Reacción rápida: el maestro AS-i necesita como máximo 5 ms para el intercambio de datos cíclico con hasta 31 estaciones conectadas.
- Las estaciones (esclavos AS-i) conectadas al cable AS-i pueden ser sensores/ actuadotes con conexión AS-i integrada o módulos AS-i, a cada uno de los cuales se pueden conectar hasta ocho sensores/actuadores binarios convencionales.
- Con módulos AS-i estándar pueden funcionar hasta 124 actuadores y 124 sensores conectados al cable AS-i. • Si se utilizan módulos AS-i con un espacio de direcciones ampliado, es posible la operación de hasta 186 actuadores y 248 sensores con un maestro extendido.[15]

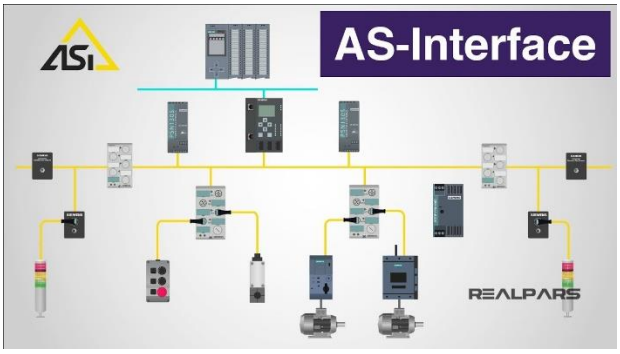


Ilustración 12. Protocolo AS-I. [15]

4.6.2. Profibus

La comunicación por protocolo Profibus o Process Field Bus se fundamentan en la funcionalidad, rapidez, seguridad, compatibilidad, versatilidad y sencillez de instalación. La conexión entre controladores, los instrumentos de medición y los elementos finales de control, se hace solamente por un solo cable para una comunicación en serie. Las principales características de esta red son las siguientes:

- Bus de campo abierto y universal
- Permite comunicación rápida con dispositivos inteligentes ubicados en forma descentralizada (Profibus DP).
- Posibilita tanto la comunicación, como la alimentación de dispositivos transmisores de señal y elementos finales de control (Profibus PA).
- Su estructura es sencilla, fiable y robusta.
- Tiene una plataforma con un sistema modular que posibilita aumentar la estructura a la medida de las necesidades de cada industria en particular.
- Está soportado en normas internacionales IEC 61158 e IEC 61754 [IEC: International electrotechnical Commission], EN 50170, EN 50250 y la norma Alemana DIN 19 245.
- Profibus puede soportar hasta 12 Mbits/s como velocidad de transmisión de datos.
- Entre los dispositivos que se pueden integrar a la red Profibus, se destacan: dispositivos inteligentes de campo, variadores de velocidad, elementos finales de control, PLC's, interfaces hombre máquina, controladores de proceso y estaciones periféricas.

Para poder acceder a una red Profibus, los equipos involucrados deben disponer de unas herramientas de ingeniería, parametrización, puesta en funcionamiento, diagnóstico, gestión de archivos y mantenimiento, así como una descripción detallada de cada uno de los equipos.[16]

4.6.3. Profinet

PROFINET es la solución de Ethernet industrial más avanzada del mundo. Es un protocolo de comunicación para intercambiar datos entre controladores y dispositivos. Los controladores pueden ser PLC, DCS o PAC. Los dispositivos pueden ser bloques de I/O, sistemas de visión, lectores RFID, accionadores, instrumentos de procesos e incluso otros controladores.

PROFINET es compatible, y aprovecha todas las características, del Ethernet de oficina. Sin embargo, hay diferencias, el Ethernet de oficina no tiene la capacidad de ofrecer rendimiento en tiempo real necesario para automatización industrial. Aparte, el Ethernet de oficina tiene menos capacidades para soportar los difíciles entornos industriales.

PROFINET puede funcionar en entornos industriales difíciles, y tiene la capacidad de ofrecer la velocidad de precisión que requieren las plantas de fabricación. También puede brindar funciones adicionales, por ejemplo, seguridad funcional, administración de energía e integración a TI. Estas funciones adicionales se pueden utilizar en combinación con las funciones de control y monitoreo. Los usuarios pueden elegir en cuáles dispositivos desean utilizarlas.

Estas son algunas ventajas que trae la implementación de PROFINET en aplicaciones industriales:

- Arquitecturas altamente escalables.
- Acceso a dispositivos en campo sobre la red.
- Mantenimiento y servicio desde cualquier lugar (incluso por Internet).
- Los mejores diagnósticos de su clase.
- Menores costos de la producción y monitoreo de calidad de los datos.[17]

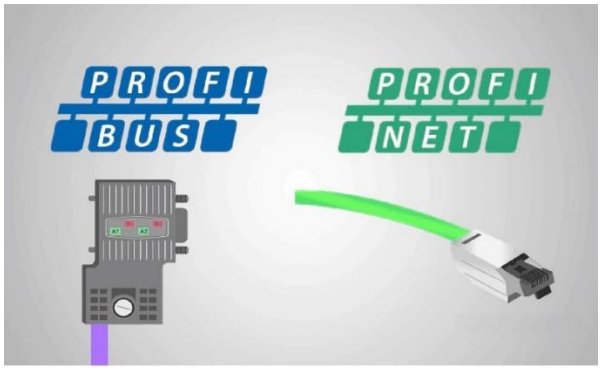


Ilustración 13. Logotipos Profibus y Profinet.[18]

4.6.4. Ethernet/IP

Ethernet/IP es un protocolo de red en niveles para aplicaciones de automatización industrial. Basado en los protocolos estándar TCP/IP, utiliza los ya bastante conocidos hardware y software Ethernet para establecer un nivel de protocolo para configurar, acceder y controlar dispositivos de automatización industrial. Ethernet/IP clasifica los nodos de acuerdo a los tipos de dispositivos preestablecidos, con sus actuaciones específicas. El protocolo de red Ethernet/IP está basado en el Protocolo de Control e Información (Control and Information Protocol - CIP) utilizado en DeviceNet y ControlNet. Basados en esos protocolos, Ethernet/IP ofrece un sistema integrado completo, enterizo, desde la planta industrial hasta la red central de la empresa.

Ethernet/IP utiliza todos los protocolos del Ethernet tradicional, incluso el Protocolo de Control de Transmisión (TCP), el Protocolo Internet (IP) y las tecnologías de acceso mediático y señalización disponibles en todas las tarjetas de interfaz de red (NICs) Ethernet. Al basarse en los estándares tecnológicos Ethernet, el Ethernet/IP garantiza la compatibilidad de funcionamiento con todos los dispositivos del estándar Ethernet/IP utilizados en la actualidad. Y lo mejor es que al apoyarse en los estándares de esa plataforma tecnológica, el Ethernet/IP, con toda la seguridad, evolucionará junto con la evolución de la tecnología Ethernet.

Las entidades que desarrollan el Ethernet/IP están trabajando juntas en la producción de un estándar completo y consistente. Esos trabajos se están conformando con la participación de varios fabricantes, lo que abarca la definición de especificaciones mediante la aplicación de pruebas exhaustivas en laboratorios certificados.[19]

EtherNet/IP es bastante fácil de implementar y es compatible con los switches Ethernet estándar para automatización industrial. Sin embargo, la forma básica de EtherNet/IP es no determinística y, por lo tanto, no es adecuada para aplicaciones industriales de tiempo real estricto.

La transmisión de datos vía EtherNet/IP mediante TCP y UDP (User Datagram Protocol) son también los protocolos de comunicación subyacentes de la Internet y de muchas redes privadas. EtherNet/IP emplea un puerto TCP para lo que se denomina mensajería explícita. Tal mensajería es cuando el sistema envía datos a un cliente en respuesta a una solicitud específica de esos datos. Utiliza TCP/IP, o sea un protocolo orientado a conexión que gestiona explícitamente enlaces entre clientes y servidores.[20]

4.7. VISIÓN ARTIFICIAL

La visión artificial industrial es una de las tecnologías que marca la diferencia en ciertas tareas esenciales en la producción industrial. Estos sistemas de visión artificial aplicados a la robótica facilitan el dar solución en fases industriales tan decisivas como los controles de calidad o la detección de la posición de diversos productos.

La visión artificial es una tecnología industrial aplicable a diferentes sectores y fases de producción. Es de los métodos automatizados e inteligentes más efectivo e innovador para adquirir, procesar y analizar imágenes en los procesos de producción.

El procesamiento de estas imágenes con los paquetes de software que llevan instalados los sistemas y cámaras visión artificial tienen el objetivo de producir información que las máquinas automatizadas pueden emplear y corregir posibles errores en las líneas de montaje.

De esta manera, son los sistemas de visión artificial aplicados a la robótica los que se encargan de realizar procesos de inspección de muestreo, supervisión y controles de calidad, eximiendo a los empleados de realizar tareas repetitivas y donde existe más margen de error y menos exactitud analítica que la que ofrece el procesamiento de una imagen.

Por tanto, la base de la visión artificial consiste en el empleo de imágenes captadas por los sistemas de visión, que son procesados por el software que lleva el mismo sistema y, con todo ello, medir, contar, seleccionar o identificar productos y si están o no defectuosos.

Según el procesamiento de estos datos, el sistema robotizado puede tomar decisiones automatizadas e inteligentes para revertir los errores, desechar productos y todo ello, siguiendo los parámetros con los que el sistema ha sido programado.[21]

Ilustración 14. Cámara de inspección Omron F420-F.[22]

5. MÉTODO OPERATIVO

A la hora de desarrollar este proyecto, lo primero que tenemos que tener en cuenta será el estudio del proceso a automatizar, estableciendo las necesidades que pueda tener y eligiendo las funcionalidades que mejorarían la productividad y la flexibilidad del proceso.

Una vez finalizado el estudio del proceso, realizamos la elección de los diferentes elementos con los que lo implementaremos. Hablando tanto de los componentes como de los diferentes paquetes de software de programación y sus tipos de comunicación. También realizamos las piezas para el pick and place, y seleccionamos los dos tipos de pallets que vamos a utilizar.

Una vez completado todo el previo anterior, se realizará la primera prueba de comunicación entre los componentes, mediante un pequeño proyecto. Debido a que era la primera vez que tenía contacto con un robot, con un HMI y con un PLC de última generación.

Por último, trataremos las diferentes partes que construyen el sistema final, hablando de su esquema general, del desarrollo por separado del programa de cada elemento, de su comunicación final y de su puesta en funcionamiento de forma física, que es en lo que realmente se basa este proyecto.

Ilustración 15. Paletizado con Omron TM.[23]

5.1. ESTUDIO DEL PROCESO A AUTOMATIZAR

El proceso flexible a automatizar se basa en la realización de un paletizado y despaletizado de precisión, pudiendo seleccionar, por forma y color, las piezas que deben de rellenar cada una de las posiciones de los diferentes pallets.

Para la automatización de un proceso productivo debemos seguir siempre unos pasos preestablecidos. La primera fase es la de captación de datos, convertir los datos en información de valor y realizar la toma de decisión. Debido a que la propuesta de este trabajo es un proceso simple, solo hubo un tema determinante a la hora de la automatización, y era el uso de un Cobot, por su facilidad a la hora de realizar la exposición física del proyecto.

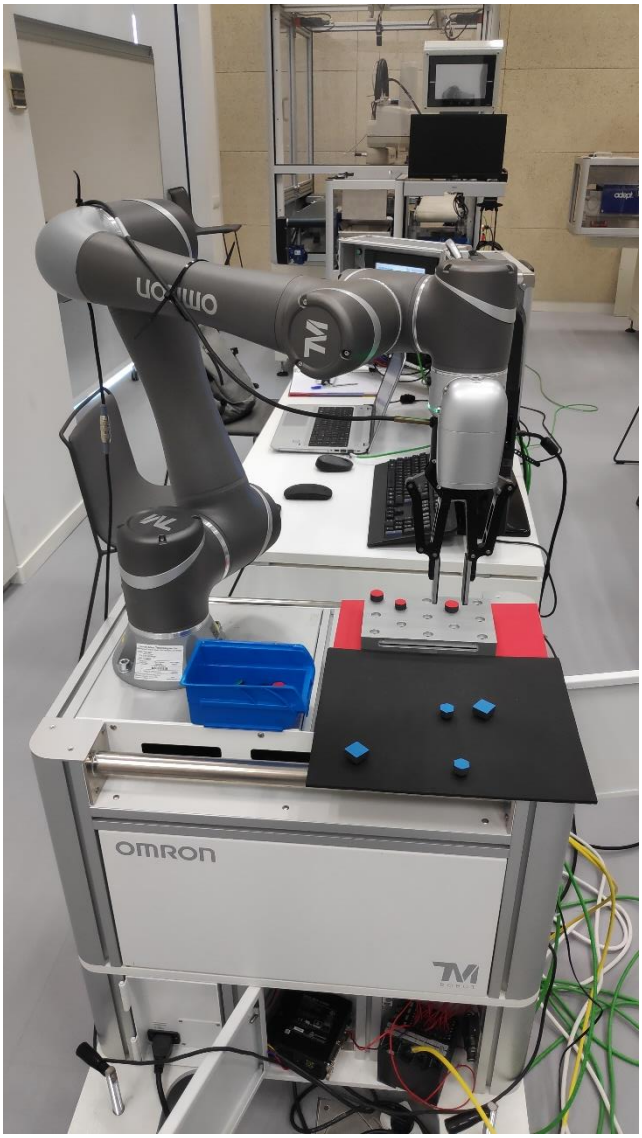


Ilustración 16. Despaletizado mediante cobot TM.

5.2. ELECCIÓN DE COMPONENTES

Los componentes que tendremos que elegir serán un PLC, un HMI y un Cobot. El PLC es el encargado de realizar la comunicación entre los otros dos componentes, aparte de ejecutar el programa principal de nuestro proyecto. El HMI debe de ofrecer al usuario una interfaz cómoda e intuitiva, para poder seleccionar las diferentes tareas que se pueden realizar en nuestro proyecto. Y por último el Cobot para poder hacer el pick and place, siendo capaz de coger piezas de diferentes formas, y diferenciarlas mediante visión según el color y la forma. También debe de localizar el Pallet y comprobar que este vacío y si no lo está, tener la capacidad de vaciarlo, todo esto mediante la visión artificial. Y por último tiene que ser seguro para poder realizar la demostración presencial sin tener problemas de seguridad.

Todos los componentes de este proyecto fueron cedidos por la empresa Omron, por la tanto la selección de componentes ha sido realizada solo dentro de su catálogo de producto.

Comenzamos seleccionado el robot colaborativo Omron TM 5-900, el cual era ideal para la realización de la demo. Después seleccionamos el PLC NX102, ya que es la única serie que pose bloques de función para la comunicación directa entre el Cobot de Omron y el PLC. Y por último seleccionamos la pantalla HMI, la cual se trata de una pantalla de la serie NA de nueve pulgadas, ya que la programación de esta serie va integrada en el software de programación del PLC, facilitando mucho la comunicación y la programación del HMI. Se escogió de nueve pulgadas ya que es la más vendida en el mercado, y es lo suficientemente grande para realizar una interface intuitiva y representativa del proceso que se está realizando.

5.2.1. PLC NX102

El controlador lógico programable que usaremos para llevar a cabo el proyecto será el NX102, de la marca Omron. La serie NX1 ofrece funciones de lógica, motion e información. El NX1 une los mundos de producción e IT, minimizando la ingeniería y el mantenimiento y suprimiendo el middleware.

Esta serie está preparada para el internet de las cosas (IoT), debido a su servidor OPC UA, que es un servidor estandarizado para las fábricas inteligentes, y la compatibilidad con estructuras. Pose también conexión directa con las siguientes bases de datos: cliente SQL para servidor Microsoft SQL, Oracle, IBM DB2, MySQL, Firebird, PostgreSQL.

La memoria de esta serie para el procesamiento de datos es muy amplia, ya que posee 33,5 MB para variables. Tiene un tiempo de ciclo a partir de un milisegundo, la capacidad de ser un controlador multitarea, no posee batería, un solo ciclo sincronizado para entradas y salidas locales y remotas, dispositivos Ethercat y motion control.

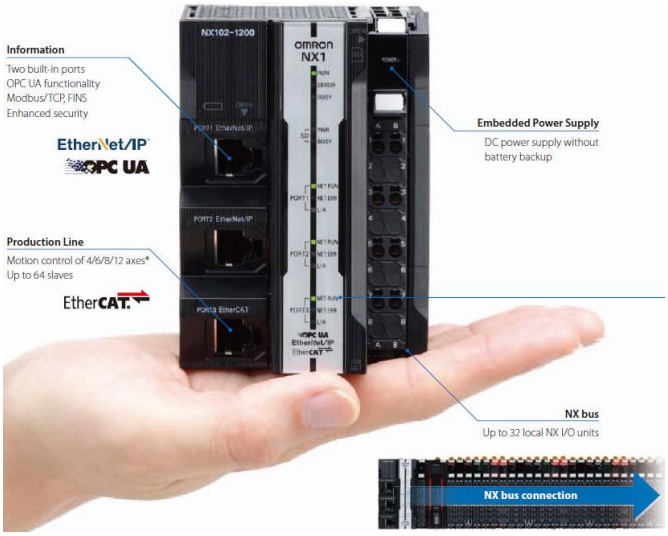


Ilustración 17. PLC de la serie NX1.[24]

Por ultimo posee una conectividad muy amplia, ya que tiene dos puertos Ethernet y un puerto Ethercat, puedes conectarla hasta 32 unidades NX locales, incluidas las de seguridad y el maestro IO-Link. Es compatible con hasta 64 nodos Ethercat, equivalente a 400 unidades NX. Y posee un sistema de seguridad de anillo Ethercat en la que es capaz de mantener las comunicaciones y el control en caso de que se rompa un cable o falle un dispositivo.

Todo esto se traduce en que El NX1 puede utilizar información, tomar medidas de seguridad y controlar la calidad, y al mismo tiempo, mejorar la eficiencia de la producción mediante un control de alta velocidad y alta precisión. Esto contribuye a la mejora continua de la productividad.

El NX1 es el primero en el mundo en integrar dos redes abiertas diferentes: Ethernet IP para control escalable en líneas de producción y Ethercat para control rápido y fiable en máquinas. Además, integra el control de seguridad de las maquinas en líneas que requieren tiempos de ciclo rápidos. Esta integración le permite estandarizar máquinas y construir líneas flexibles.

El controlador NX1 integra entradas, lógica, salidas, seguridad y robótica, ofreciendo una amplia variedad de aplicaciones que aprovechan la información para impulsar la productividad y las medidas de calidad y seguridad.[25]

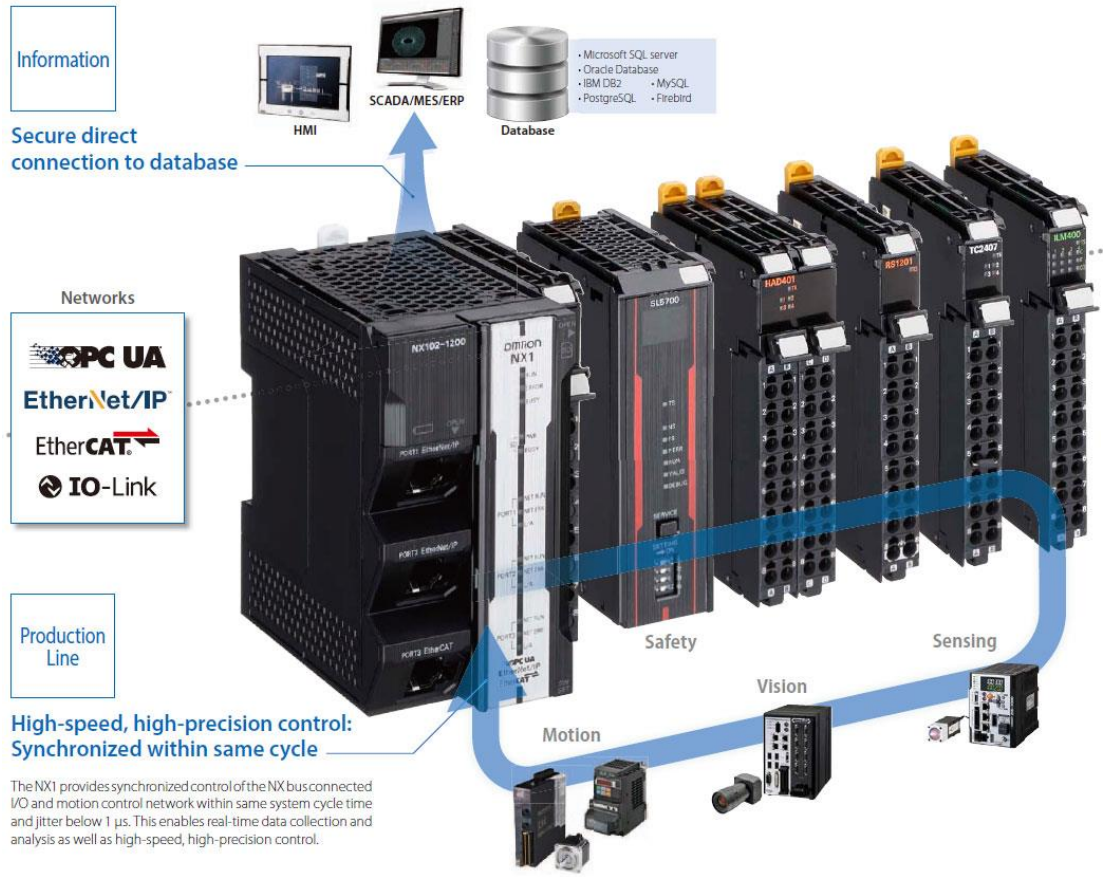


Ilustración 18. Esquema módulos NX1. [25]

5.2.1.1. Sysmac Studio

El software de automatización Sysmac Studio proporciona un entorno de desarrollo integrado para configurar, programar, depurar y mantener los controladores de la serie SYSMAC NJ/NX/NY y otros controladores de automatización de máquinas, así como esclavos EtherCAT.

Sysmac Studio proporciona un entorno para programar con variables. No hay necesidad de preocuparse por las direcciones de memoria. Esto elimina la necesidad de esperar las definiciones de direcciones de memoria para el hardware antes del inicio del desarrollo del software. El hardware y el software se pueden diseñar de forma independiente y desarrollar en paralelo. Las POU (program organization units) que incluyen programas, funciones y bloques de funciones se pueden usar para diseñar programación que no dependa de ningún sistema específico. Esto aumenta la reutilización de la programación.

Sysmac Studio proporciona un entorno para la programación con variables y POU. La programación está diseñada con POU (programas, funciones y bloques de funciones). A continuación, los programas se asignan a tareas y se define el orden de ejecución del programa. Esto reduce la interdependencia de los programas y, por lo tanto, permite que más de un programador trabaje fácilmente al mismo tiempo. Las asignaciones de variables al hardware y las definiciones de las relaciones entre la información que se comparte entre diferentes programas se pueden configurar en cualquier momento.

Ilustración 19. Inicio Sysmac Studio.[26]

Sysmac Studio se basa en el estándar internacional IEC 61131-3. Proporciona un entorno de programación de vanguardia basado en el diagrama de relés y los lenguajes de programación de texto estructurado y en POU, que incluyen programas, funciones y bloques de funciones.

Sysmac Studio impone la menor cantidad posible de restricciones a los procedimientos de diseño, para permitirle comenzar el trabajo de diseño desde cualquier parte del sistema. Esto proporciona una operación fácil de usar para un trabajo de diseño flexible en el que incluso los errores en la configuración y los procedimientos se pueden corregir de inmediato o dejarlos hasta que se finalice el proyecto, siempre que no den lugar a accidentes graves. Sysmac Studio está diseñado para lograr una funcionalidad óptima y facilidad de operación para combinar controladores de automatización de máquinas, como los de la serie NJ/NX/NY, con esclavos EtherCAT compatibles con Sysmac.

Sysmac Studio proporciona funciones completas para depurar el control de secuencia, como cambiar los valores actuales y cambiar la programación en línea. También proporciona funciones de depuración con simulaciones de motion control, como mostrar los resultados de las trazas en 2D o 3D y mostrar las trazas en dispositivos virtuales. Estas funciones permiten depurar imágenes más cercanas a los dispositivos físicos.

Además, se admite la depuración a través de la simulación del sistema de visión con un sensor de visión. Con Vision Edition, puede realizar simulaciones para sensores de visión individuales. Con la Standard Edition, también puede realizar una simulación integrada con un controlador.

Sysmac Studio le permite verificar el estado del controlador en una lista de estado. Las funciones de resolución de problemas le permiten verificar fácilmente los detalles de los errores y las correcciones de los errores del controlador. También puede asignar errores definidos por el usuario de la misma manera que se asignan los errores del controlador. [27]

5.2.1.1.1. Diseño básico de un sistema.

En este apartado se explica de forma esquematiza el proceso para crear un sistema en Sysmac Studio desde cero, aprovechando las características de la programación IEC 61131-3. Este procedimiento es adecuado para el diseño de programas de usuario en un escenario de desarrollo, en el que las especificaciones de hardware y software se determinan en fases. Los procedimientos a realizar se describen a continuación:

Creación de un proyecto

- 1. Inicio de Sysmac Studio
- 2. Crear un proyecto

Diseño del prgrama de usuario

- 1. Registro de variables globales: registre las variables definidas por el usuario que se utilizan en más de un programa en el Variables globales
- 2. Registro de POU: registre los programas, funciones y bloques de funciones como POU
- 3. Registro de variables locales: registre las variables definidas por el usuario que se utilizan en una sola POU en las Variables locales para ese POU
- 4. Registro de variables de Axis: registre las variables que se utilizan en el programa de usuario en la configuración del Axis
- 5. Registro de variables de grupo de Axis: registre las variables de grupo de Axis que se utilizan en programas que controlan Axis interpolados en la configuración del grupo de Axis.
- 6. Registro Cam Data Variables: registre las Cam Data Variables que se utilizan en el programa de usuario en la Cam Data Settings
- 7. Creación de algoritmos de POU

Diseño de tareas

- 1. Registro de tareas: registre las tareas en Tareas
- 2. Diseñar Tareas: Diseñar el funcionamiento de las tareas
- 3. Tareas de edición: asigne programas a las tareas
- 4. Tareas de edición: Diseñe las E / S que están controladas por las tareas y las variables que se comparten entre las tareas

Depuración del programa sin conexión

- 1. Depuración mediante simulación: inicie el simulador y depure el programa de usuario

Preparativos para la depuración en línea

- 1. Asignación de variables de Axis y dispositivos de E / S
- 2. Asignación de variables y E / S reales

Depuración de programas en línea

- 1. Transferencia de datos al controlador: transfiera el programa de usuario, la configuración del sistema y la información variable al controlador
- 2. Prueba de funcionamiento del Axis: Utilice la prueba de funcionamiento de MC para comprobar el cableado
- 3. Comprobación de E / S: utilice el mapa de E / S para comprobar el estado de E / S entre los dispositivos de E / S internos y los dispositivos de E / S externos, y el cableado a los dispositivos de E / S externos
- 4. Verificación de asignaciones: use una página de la pestaña Ver para verificar las asignaciones entre las variables definidas por el usuario y dispositivos de E / S

Operación de testeo

5.2.2. HMI NA 9”

El HMI seleccionado para la realización de este proyecto ha sido la serie NA de Omron, exactamente la de nueve pulgadas, debido a que es la más común en estas aplicaciones.

La serie NA transforma los datos de la máquina en información, la cual muestra por pantalla, y controla los dispositivos según los requisitos en los sitios de fabricación de FA. La serie NA permite un control y una supervisión más rápidos y eficientes. Con una pantalla ancha que muestra 16.770.000 colores. La HMI que es dinámica, intuitiva y predictiva hace que las máquinas industriales sean más atractivas y competitivas.

Ilustración 20. Esquema control serie NA. [28]

Los terminales programables de la serie NA son dispositivos Sysmac que puede utilizar junto con los controladores de automatización de máquinas de las series NJ/NX/NY y el software de automatización Sysmac Studio para lograr una funcionalidad óptima y facilidad de operación. Esta es su principal ventaja, ya que la programación del HMI se realiza en el mismo software y en el mismo proyecto que la programación del controlador, lo que acelera y facilita el desarrollo. Gracias a esto se pueden realizar simulaciones conjuntas del HMI y del controlador para verificar el funcionamiento del dispositivo sin necesidad de disponer en todo momento de los componentes.

Método operativo

El solucionador de problemas de la serie NA permite supervisar y emitir directamente los errores y eventos de los controladores NJ/NX/NY, así como los errores y eventos definidos por el usuario. A su vez se ha aumentado la seguridad, ya que hay diferentes niveles de acceso por contraseña, para que cada tipo de personal solo pueda acceder al nivel que le corresponda.

La serie NA tiene la capacidad de acceso remoto, ya que puedes visualizar y utilizar el HMI instalada en centros de producción usando una Tablet mediante Ethernet o Wifi, este acceso se puede administrar y limitar, para garantizar la accesibilidad y evitar el funcionamiento accidental y las fugas de información.[28]

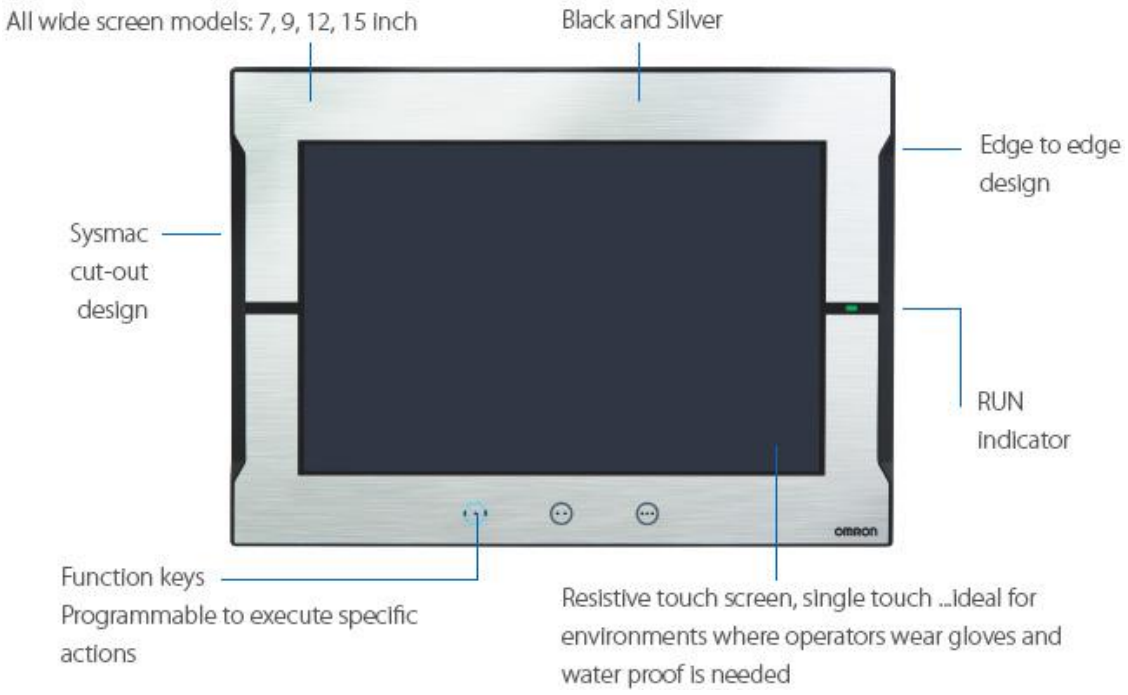


Ilustración 21. Composición de la serie NA. [29]

La serie NA posee un panel de alta resolución, en mi caso de 9 pulgadas, junto a tres botones programables, que para mí proyecto no son usados. La comunicación con él se puede realizar a través de los dos puertos Ethernet, a los cuales, se les puede conectar un controlador de las series NJ/NX/NY o un ordenador para realzar la programación. Otra opción es la introducción de una tarjeta de memoria SD para transferir directamente el proyecto que se ha debido de crear previamente en Sysmac Studio o para la transferencia de datos de registro de la unidad NA.[29]

5.2.2.1. Sysmac Studio

Para la programación de la serie NA se usa, como ya hemos comentado antes, el programa Sysmac Studio. Debido a que nuestro controlador es de la serie NX, y se deben de programar los dos dispositivos en el mismo proyecto para simplificar la programación, ya que puedes compartir las variables globales. Todo lo que tiene que hacer para especificar la memoria en el controlador es especificar las variables del controlador. Esto le permite crear objetos que no dependen de dispositivos o mapas de memoria específicos. A su vez, hace que los objetos sean mucho más reutilizables que con los PT anteriores.

Para la programación de objetos muy complicados se puede utilizar Visual Basic de Microsoft para programar funciones avanzadas que no puede lograr con objetos estándar. Pero esto no es necesario usarlo para el HMI que se creado en este proyecto.

Para la mayor seguridad del proyecto se deben de establecer niveles de seguridad, para los cuales hace falta una contraseña o un número de identificación personal, con eso nos aseguramos de que los operarios no puedan acceder a niveles superiores, para los cuales no están cualificados, poniendo en riesgo su seguridad. [29]

Ilustración 22. HMI de la serie NA. [30]

Por último, debemos de decir que Sysmac Studio proporciona un entorno de desarrollo integrado que cubre no solo el terminal programable de la serie NA, sino también el controlador y los dispositivos en EtherCAT. Puede utilizar procedimientos coherentes para todos los dispositivos independientemente de las diferencias en los dispositivos. Sysmac Studio admite todas las fases de la aplicación del controlador, desde la creación de la página y el diseño de la secuencia hasta la depuración, las simulaciones, la puesta en servicio y los cambios durante la operación. Junto a todo esto, también se puede realizar la simulación del HMI, a la vez que realizas la depuración en línea con un controlador virtual de las series NJ/NX/NY. [29]

5.2.2.1.1. Proyecto NA

Un proyecto de la serie NA creado en Sysmac estudio contiene los datos visualizados en la ilustración número 21. Un proyecto posee cuatro partes importantes, las páginas, las cuales están formados por objetos y tienen la capacidad de realizar acciones y eventos. Por otro lado, las variables globales y los eventos globales que se desarrollan en todo el proyecto. Y por último, las subrutinas que se inicializan cuando ocurren eventos, estas subrutinas se deben de programar en lenguaje estructurado, haciendo que sean complejas y poco intuitivas de crear.

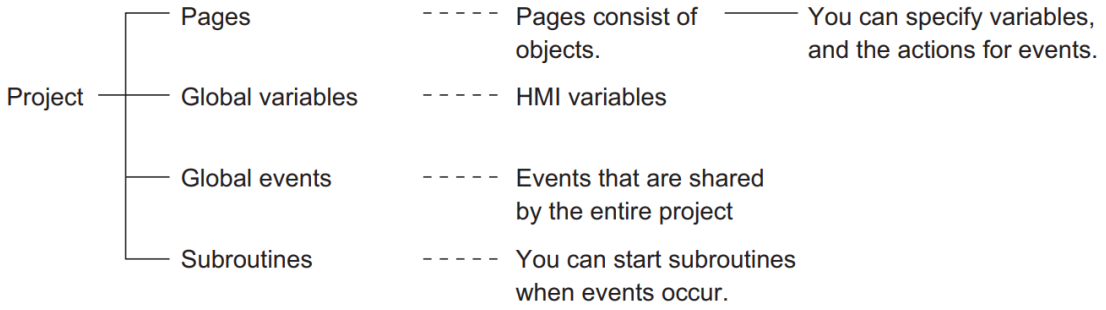


Ilustración 23. Componentes de un HMI. [29]

Además de todo esto, existen datos compartidos por todo el proyecto como las alarmas de usuario, los registros de datos, las recetas y los recursos.

5.2.2.1.2. Paginas NA

A continuación podemos observan en la ilustración 22 los datos que se le deben de asignar a cada página, que son su tamaño y su plantilla.

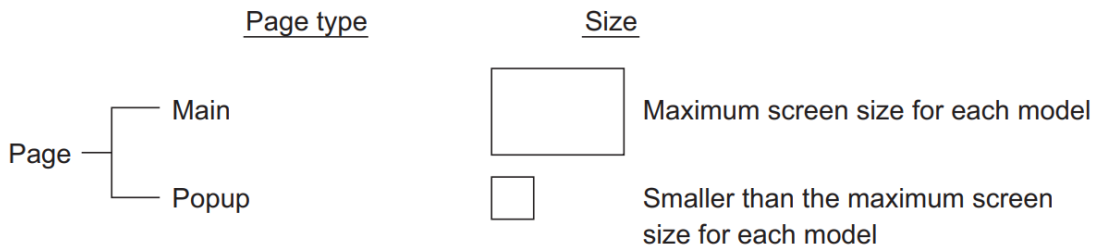


Ilustración 24. Componentes de una página. [29]

5.2.2.1.3. Objetos NA

Los objetos de la serie NA son todos los componentes de una página, y se les pueden asignar las características vistas en la ilustración 23. Primero debemos de seleccionar el tipo de objeto y darle unas propiedades estáticas. Una vez realizado esto le podemos asignar unas animaciones, que ocurran cuando una condición se cumpla. Y a su vez se les pueden asignar unos eventos y acciones para que el objeto realice algo específico cuando el evento ocurra.

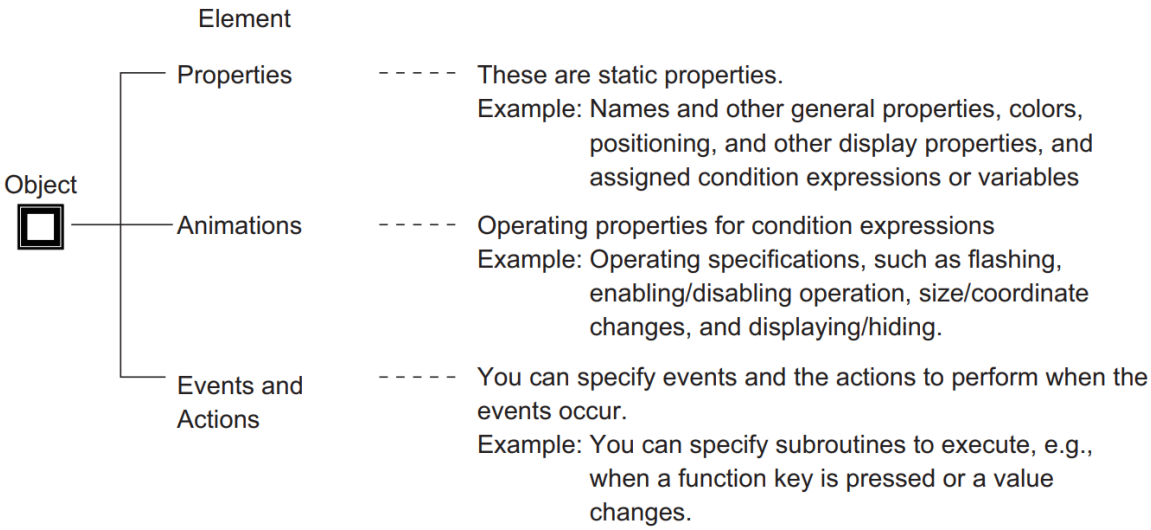


Ilustración 25. Componentes de un objeto. [29]

5.2.2.1.4. Memoria NA

La serie NA posee memoria propia, en la cual debes de crear el tipo de variable según el uso que le vallas a dar. Para especificar la memoria en un controlador o PLC, debes de usar variables globales, y estas variables globales se deben de asignar a un dispositivo previamente conectado. Estas variables se registran en variables de dispositivo en el proyecto HMI, y se pueden usar los siguientes métodos:

- Las variables de los dispositivos conectados que están registrados en el mismo proyecto se registran automáticamente.
- Puede copiar y pegar variables de otro proyecto usando el portapapeles.
- Puede importar variables desde el dispositivo externo conectado.

Por otro lado, las variables de los dispositivos se asignan a las variables globales del HMI y las variables globales asignadas se especifican en las propiedades de los objetos.

Todo ello se puede ver reflejado en la siguiente imagen:

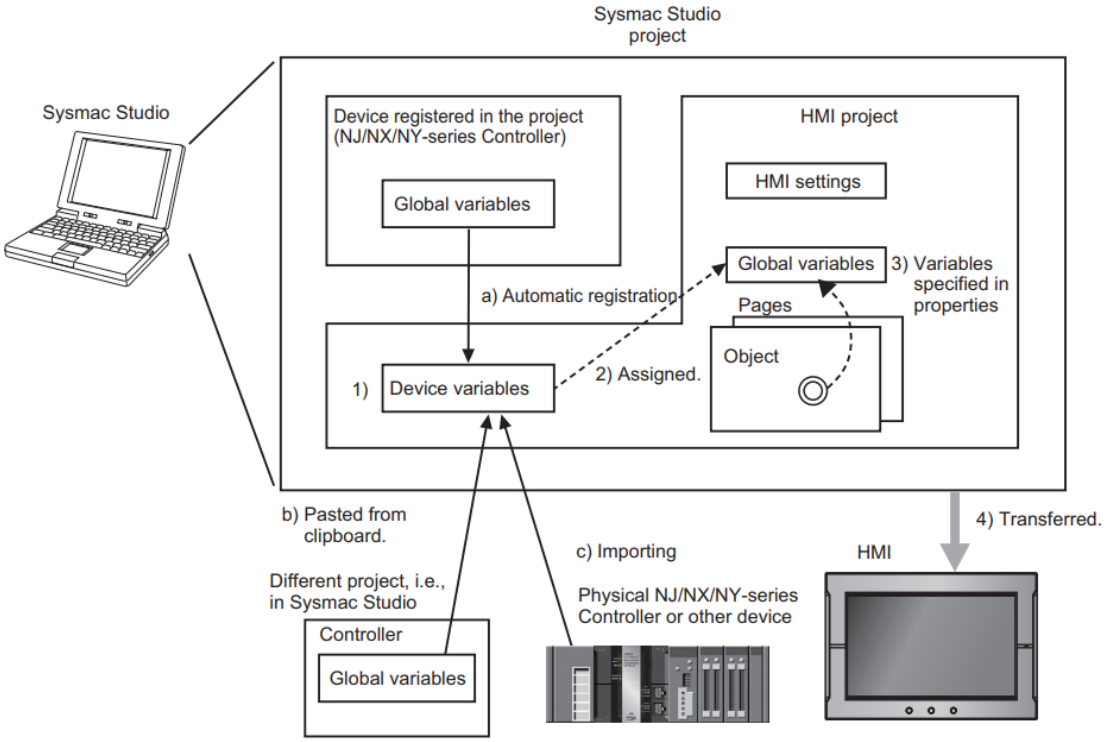


Ilustración 26. Registro de variables en la serie NA. [29]

5.2.2.1.5. Eventos

Los eventos son los desencadenantes de las acciones, es decir, para que una acción se produzca debe de haber un evento que la active, a su vez, las acciones son varias operaciones que se pueden asignar directamente a eventos, y estos eventos ocurren cuando el estado de la página común o el estado del objeto cumple ciertas condiciones. Los eventos se clasifican en tres grupos como se muestra a continuación.

	Group	Description
Events	Global events	Events that occur for shared project status.
	Page and object events	Events that occur for specific page or object status.
	User alarm events	Events that occur for user alarm status.

Ilustración 27. Clasificación de los eventos en la serie NA. [29]

La clasificación de eventos posee tres grandes grupos, que son, los eventos globales que son usados para el estado del proyecto compartido, los eventos de objetos y páginas, que se ocurren para el estado especifico de un objeto o una página, y por último los eventos de alarmas de usuario, que son usados para dar alertas y alarmas por pantalla para el operario.

5.2.2.1.6. Subrutinas

Las subrutinas son pequeños programas realizados en texto estructurado que crean acciones y eventos. Para ejecutarlas se debe de dar un evento global, un evento de página o de objetos, o un evento de alarma de usuario.

Las subrutinas pueden ser de dos tipos, globales o de página, y deben de ser creadas usando Visual Basic para redactaras, este método es muy complejo, ya que no existen ayudas a la programación y los errores de programa no son reportados en su mayoría.

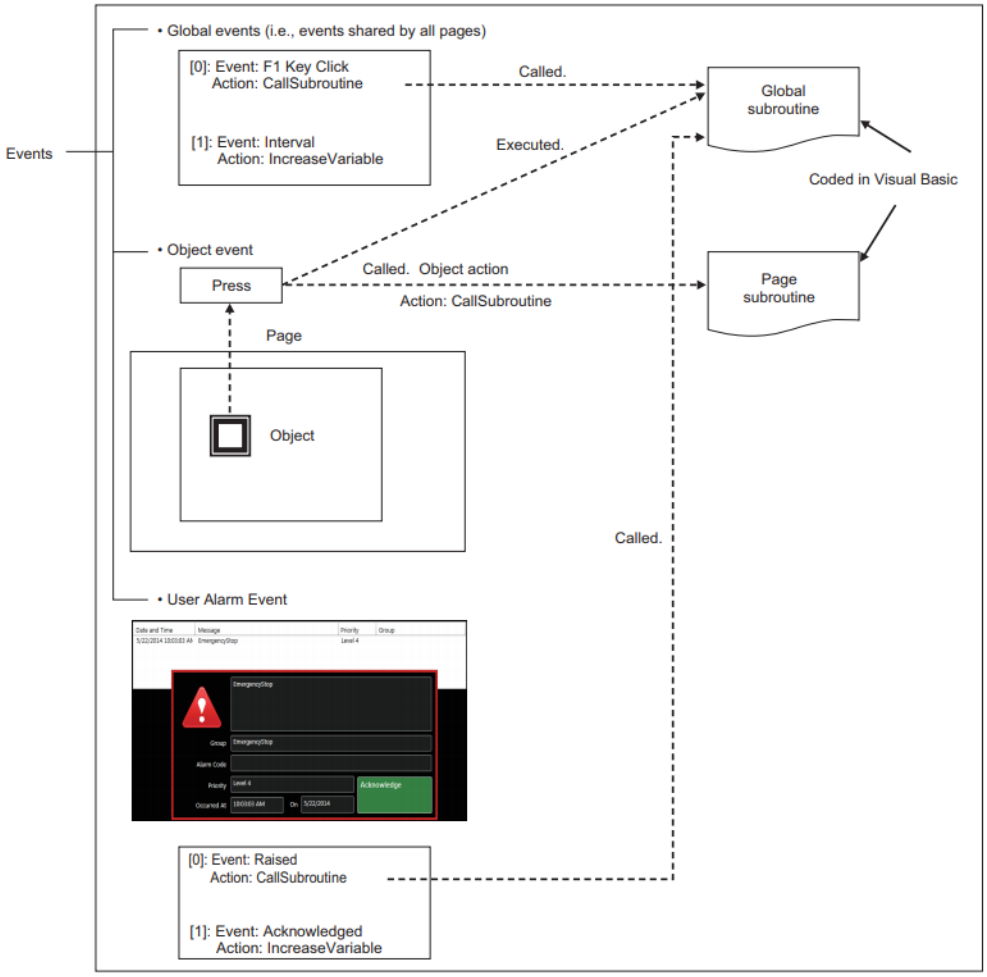


Ilustración 28. Uso de las subrutinas en la serie NA. [29]

5.2.3. Cobot TM 5-900

La serie TM es la última incorporación en la parte robótica de la empresa Omron, se trata de robots colaborativos creados por la empresa "TechMan Robot". Son una serie de brazos robóticos de 6 ejes con función de limitación de potencia y fuerza, que presenta una programación simple, capacidades de visión integradas e innovadoras, junto con la última funcionalidad de seguridad.

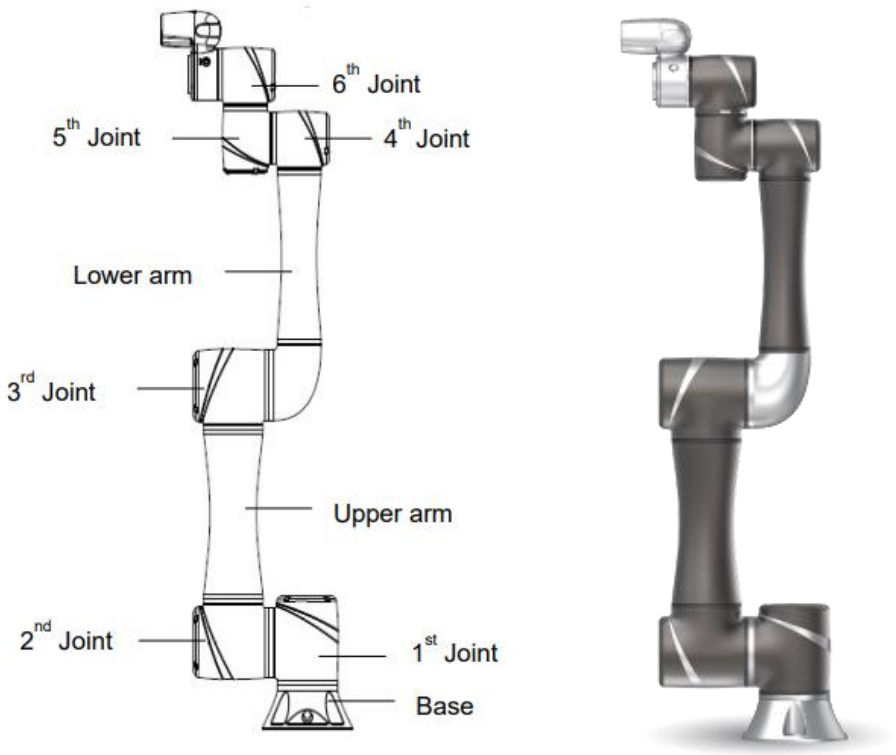


Ilustración 29. Ejes TM5-900. [31]

Esta serie está diseñada para poder trabajar tanto con humanos como con otras máquinas, teniendo unos sistemas de seguridad y una visión integrada de muy alta calidad. A la vez que es transportable, permite arranques rápidos y adaptabilidad para realizar casi cualquier tarea donde sea necesario, aumentando la productividad al realizar tareas repetitivas.

Tal y como se puede ver en la siguiente imagen las aplicaciones de este robot son muy diversas, solo realizando un cambio en el complemento, ya sea una pinza, una ventosa, un atornillador, una ventosa,…[32]

Ilustración 30. Funcionalidad Omron TM.[33]

Esta serie también ha sido creada para la instalación sobre un AGV, consiguiendo así un brazo robot móvil colaborativo. A este tipo de robots se les denomina MoMa, que viene de las siglas en ingles mobile manipulator, y en Japón ya están siendo implementados en los procesos de fabricación por islas.[34]

Ilustración 31. MoMa Omron.[35]

El robot colaborativo OMRON TM cuenta con un sistema de visión incorporado. La cámara integrada localiza objetos en un amplio campo de visión y la luz de mejora de la imagen, permite el reconocimiento de objetos en casi cualquier condición.

El sistema de visión mejora la confiabilidad, la consistencia y la ubicación de alta precisión, e incluye funciones como la coincidencia de patrones, la lectura de códigos de barras y la clasificación de colores que permiten la inspección, las mediciones y la clasificación sin costos ni esfuerzos adicionales.[36]

Ilustración 32. Visión integrada Omron TM. [36]

La parte más importante de un Cobot es la seguridad, la serie TM de Omron posee tecnologías como el force feedback, los servomotores de baja inercia, los actuadores elásticos y la tecnología de detección de colisiones, que limitan su potencia y su fuerza a niveles adecuados para el contacto. Todos ellos controlados por un software para aumentar la seguridad lo máximo posible y promover un lugar de trabajo más seguro.[32]

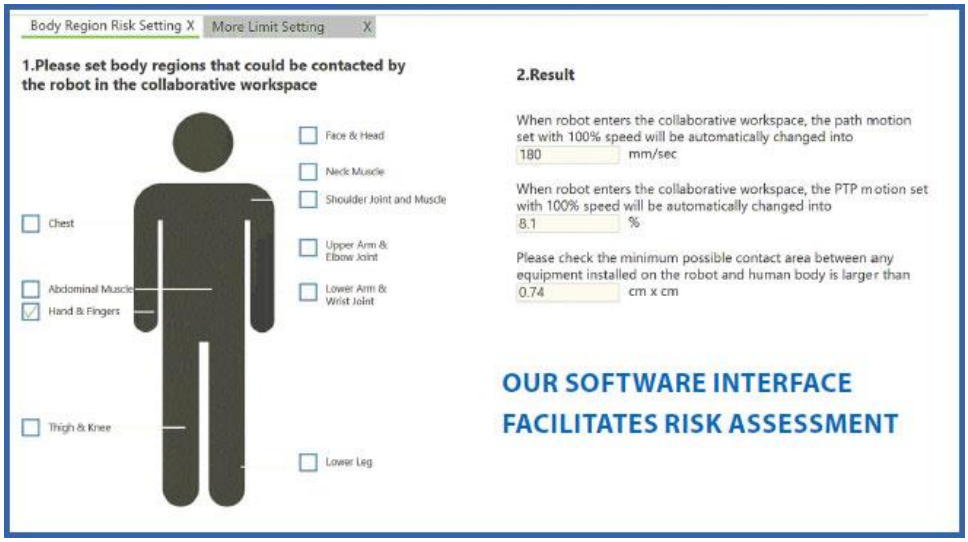


Ilustración 33. Interface del software de seguridad. [32]

Dentro de la serie TM, el robot que se ha utilizado para este proyecto es el TM 5-900, el cual posee las siguientes características:

- Alcance (mm): 900
- Carga máxima (kg): 4
- Velocidad máxima (m/s): 1,4 [36]

5.2.3.1. Funciones de seguridad e interface

El sistema de control del Techman Robot presenta una serie de funciones integradas relacionadas con la seguridad y proporciona una interfaz para la conexión con dispositivos de seguridad externos.

Para el funcionamiento correcto del cobot debe de tener unas características ambientales de entre 0°C y 50°C, una humedad relativa menor del 85%, además de que el cobot debe de estar protegido contra golpes y vibraciones. En su máxima velocidad este genera una contaminación acústica de 49,3 dB, lo que no obliga a los trabajadores a usar protección acústica.

Antes de realizar la instalación o el uso del producto se debe de realizar una evaluación de riesgos en función de las condiciones de uso, que en el caso de este proyecto ha sido muy simple, ya que a la hora de que el cobot se ponga en funcionamiento no es necesario ningún operario cerca.

Las paradas de emergencia se pueden realizar en cualquier momento activando la seta de emergencia, tanto la instalada en el mando del cobot, como la instalada dentro de la maleta de demostración, junto al HMI. Cuando el robot se detiene, el usuario debe asegurarse de que se eliminen todas las condiciones de falla antes de reiniciar manualmente el robot. Cuando el usuario presiona el interruptor de emergencia, el producto TM Robot desconectará la energía del robot y activará el freno después de que se detenga el movimiento del robot. La seta de emergencia instalada en la pantalla HMI debe de cumplir la normativa ISO 60204-1.[31]

5.2.3.2. Hardware

En este apartado expondremos la interface mecánica del TM Robot System.

Ilustración 34. Componentes del hardware. [31]

5.2.3.2.1. Brazo robótico

Comenzamos mostrando las dimensiones exactas que posee el brazo robótico, en los esquemas de a continuación:

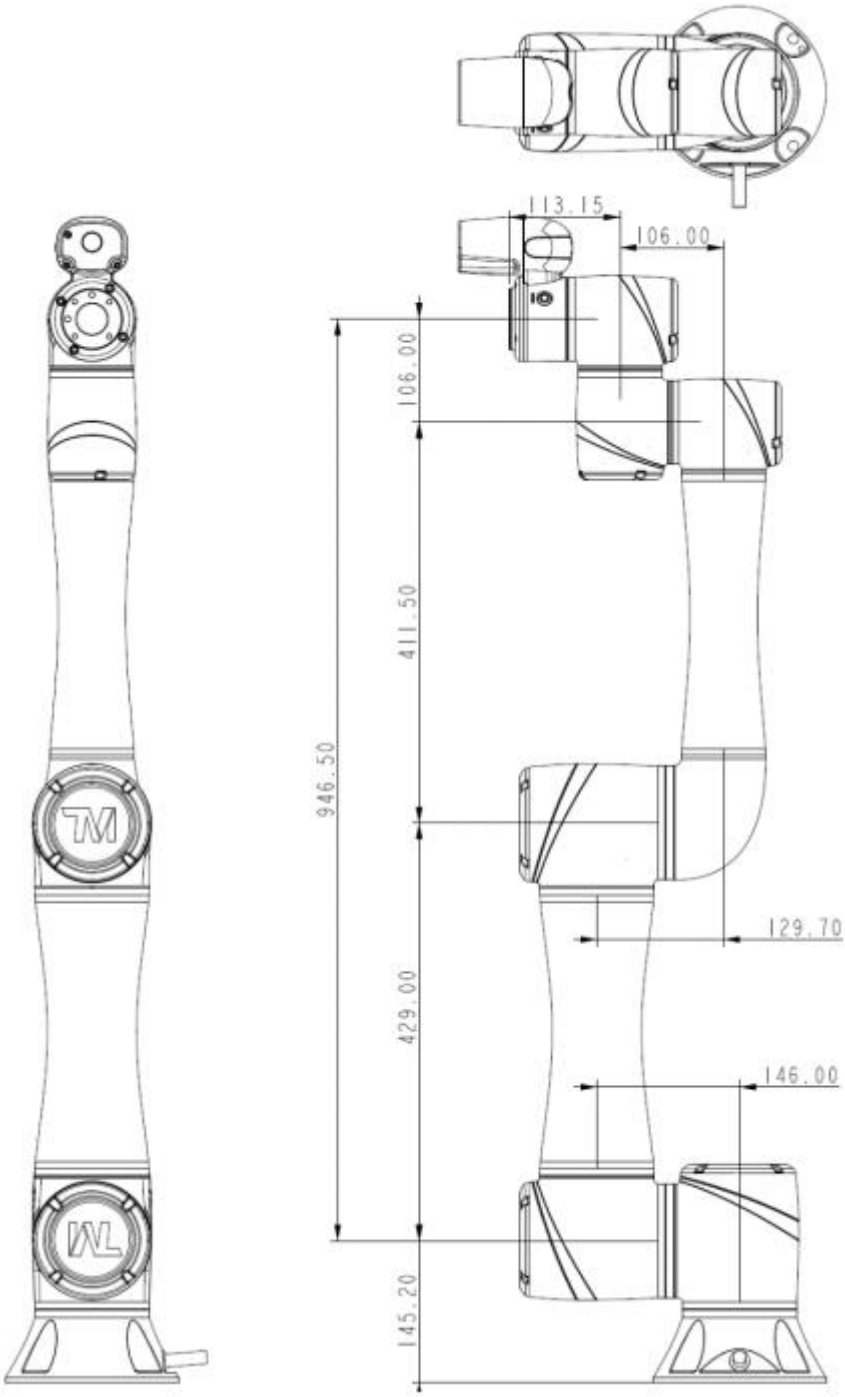


Ilustración 35. Dimensiones TM5-900. [31]

Una vez vistas las dimensiones totales del cobot, es muy importante saber cuál es el área del trabajo en la que puede actuar, y cuáles son las zonas donde se pueden situar operarios y los riesgos que posee entrar en la zona de trabajo. Eso se ve representado en los siguientes esquemas:

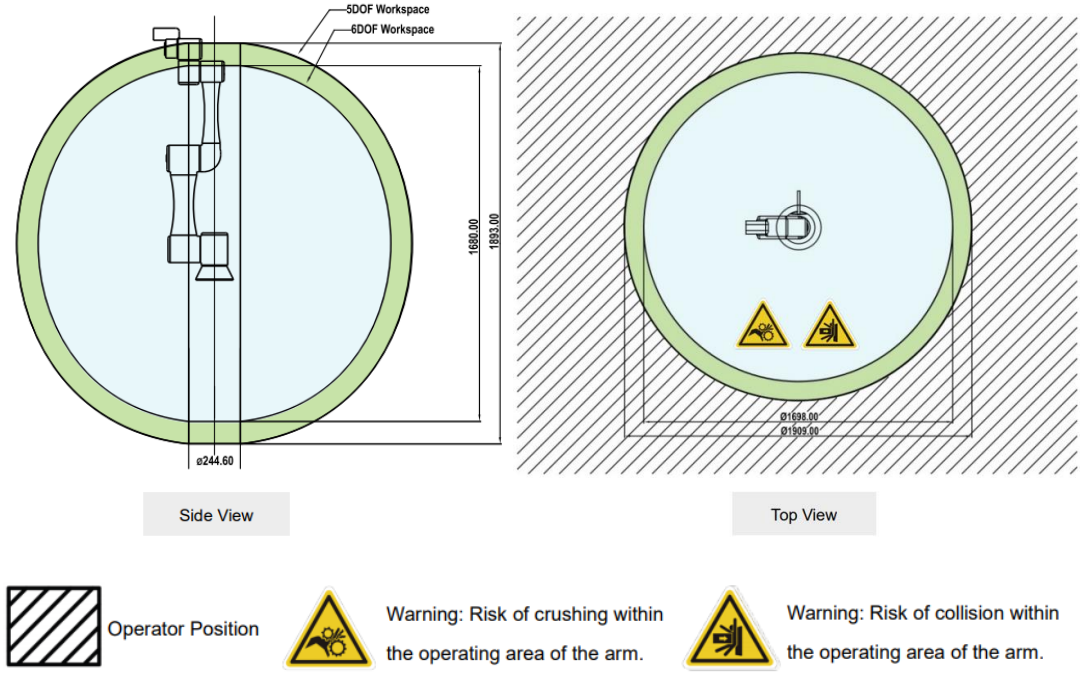


Ilustración 36. Rango de movimiento TM5-900. [31]

Por otro lado, la carga útil máxima permitida del brazo robótico está relacionada con el desplazamiento del centro de gravedad, que se define como la distancia desde el punto central del cobot hasta el centro de gravedad de la carga útil. En la siguiente figura vemos el grafico de los pesos soportados.

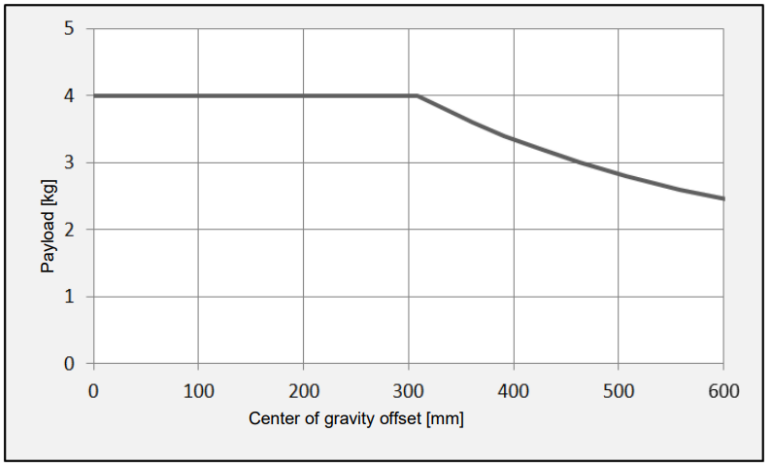


Ilustración 37. Carga máxima TM5-900. [31]

Por último, vamos a hablar del módulo del extremo del robot, desde el cual se pueden realizar multitud de funciones usando sus botones y sus complementos. En la foto de a continuación se pueden observar la posición de ellos.

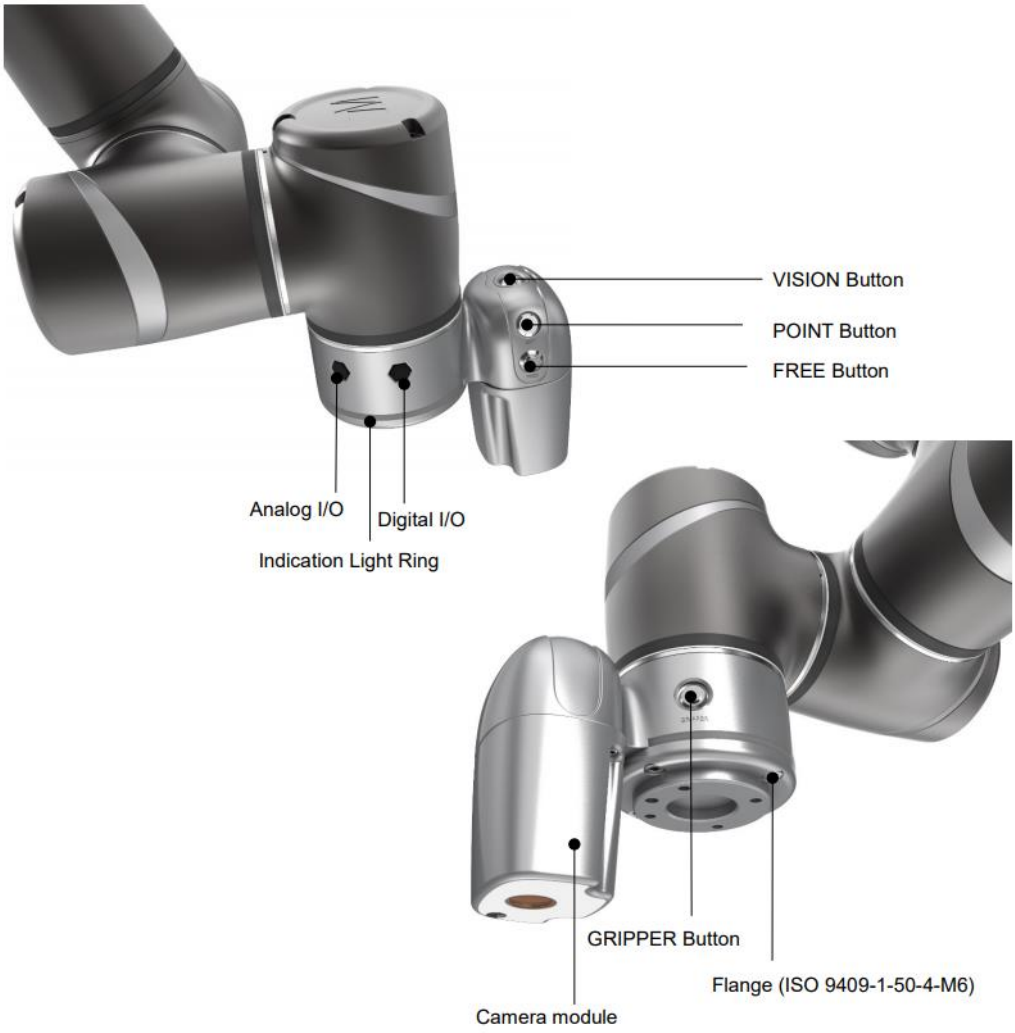


Ilustración 38. Componentes del TM5-900. [31]

Estos botones sirven en su mayoría para simplificar la programación y hacerla más intuitiva, pero a la hora de realizar una aplicación de precisión como la que se realiza en este proyecto dejan de ser útiles. El FREE button sirve para mover el robot manualmente a la posición deseada, el POINT button sirve para crear un punto en la programación, el VISION button sirve para crear una tarea de visión en la programación, y el GRIPPER button sirve para crear una tarea del complemento instalado, en la programación.

Por otro lado, se encuentran la cámara y un aro de luz led que indica mediante colores y parpadeos los diferentes modos del robot. Y por último, se encuentran dos módulos de entradas y salidas digitales, uno analógico y otro digital.

5.2.3.2.2. Controlador

Para el controlador de la serie TM5 solo se debe de tener en cuenta que es necesario un espacio de libre de 5cm a ambos lado, para su refrigeración por aire.

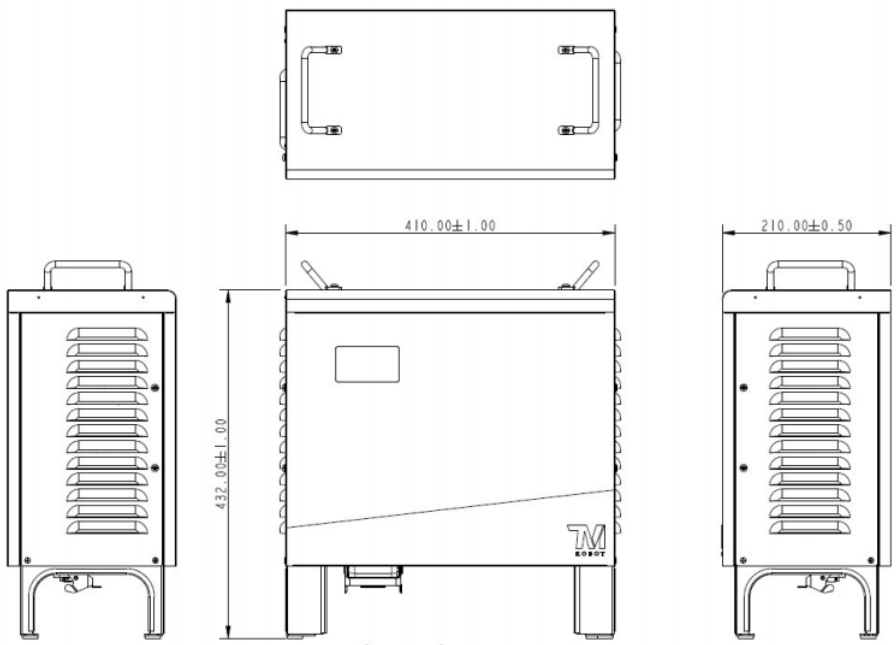


Ilustración 39. Dimensiones controlador. [31]

5.2.3.2.3. Mando del robot

El mando del robot posee 6 botones, 3 luces de indicación, una seta de emergencia y un código QR para su identificación. Las funciones de los diferentes elementos son las siguientes.

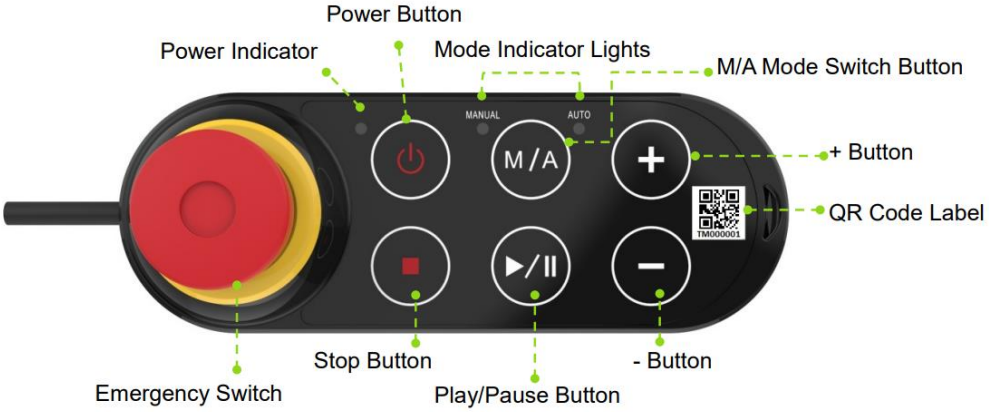


Ilustración 40. Mando TM5-900. [31]

5.2.3.3. Interface eléctrica

En este apartado vamos a ver toda la interface eléctrica tanto del brazo como del controlador. En la siguiente foto podemos ver un esquema de los tipos de conectores que posee y su posicionamiento.

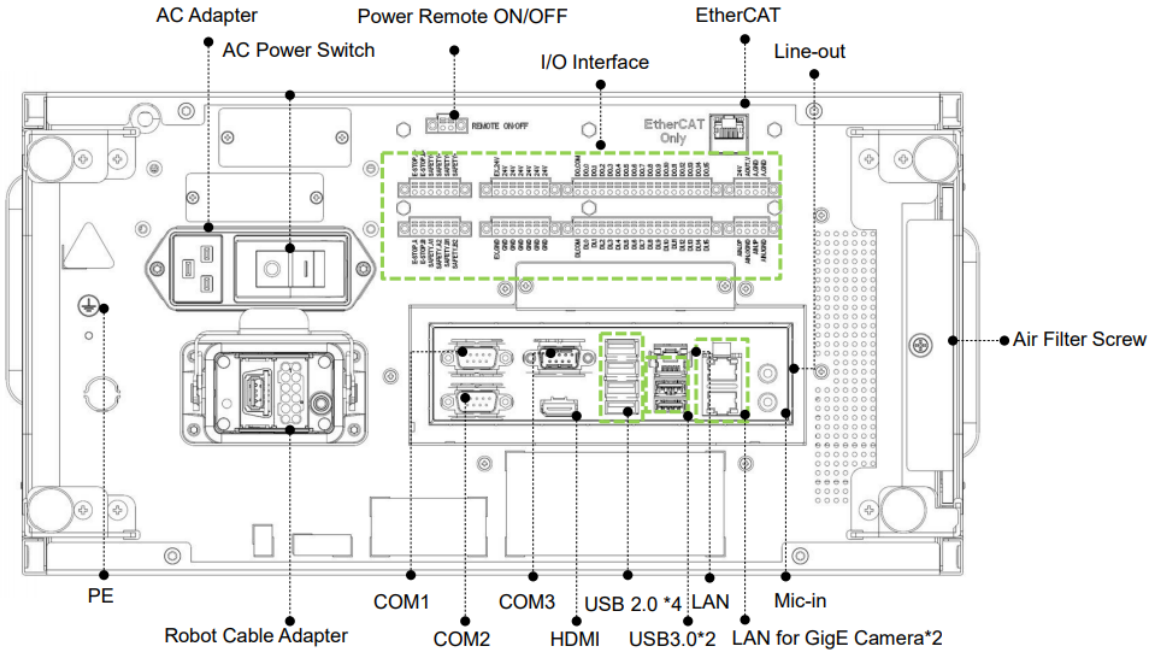


Ilustración 41. Esquema conexiones TM5-900. [31]

5.2.3.3.1. Conector de seguridad

Comenzamos explicando el conexionado de seguridad, el cual proporciona puertos de extensión para la parada de emergencia y dos puertos de protección. Los cuales tienen las siguientes características:

1. E-STOP es un contacto N.C. (normalmente cerrado). Cuando cualquier interruptor E-STOP conectado está abierto, el robot entra en el estado de parada de emergencia.
2. Safeguard A Port es un contacto N.C. (normalmente cerrado). Cuando el interruptor de seguridad A está abierto, el robot entra en el estado de pausa de seguridad.
3. Safeguard B Port es un contacto N.C. (normalmente cerrado). Cuando el interruptor de seguridad B está ABIERTO, el robot entra en el estado del modo colaborativo de protección.

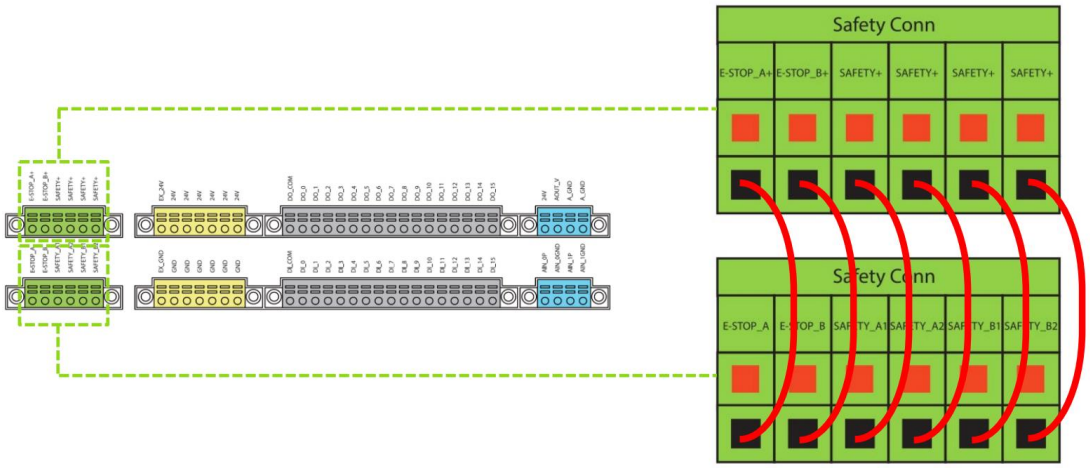


Ilustración 42. Conexionado de seguridad TM5-900. [31]

5.2.3.3.2. Conector de alimentación

En segundo lugar hablamos del conexionado de la alimentación, el cual posee las siguientes características:

1. Durante el arranque, la caja de control buscará una entrada externa de 24V. Si no se encuentra ninguno, cambiará al suministro interno de 24V.
2. La caja de control en sí ofrece una salida de 24V1.5A (24_EX). Si la carga de 24 V supera los 1,5 A, entra en modo seguro y desactiva la salida de 24 V.
3. EX24V proporciona un puerto de entrada externo de 24V. Si la carga supera los 1,5 A, se puede utilizar una fuente de alimentación externa. La carga en EX24V no debe exceder los 3.5A.

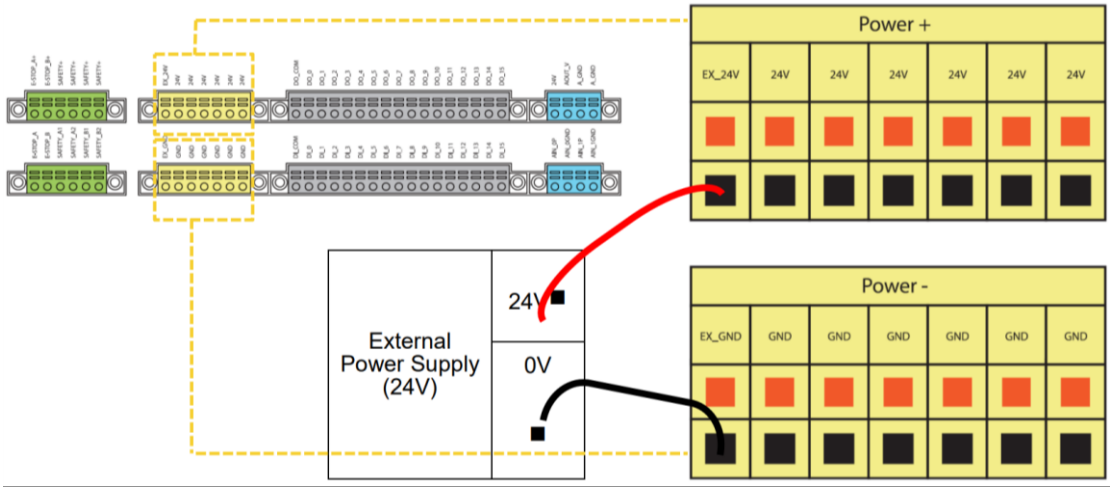


Ilustración 43. Conexionado alimentación TM5-900. [31]

5.2.3.3.3. Entradas y salidas digitales

Las entradas y salidas digitales que posee el controlador son 16 de cada tipo. Tal y como se puede ver en la siguiente imagen.

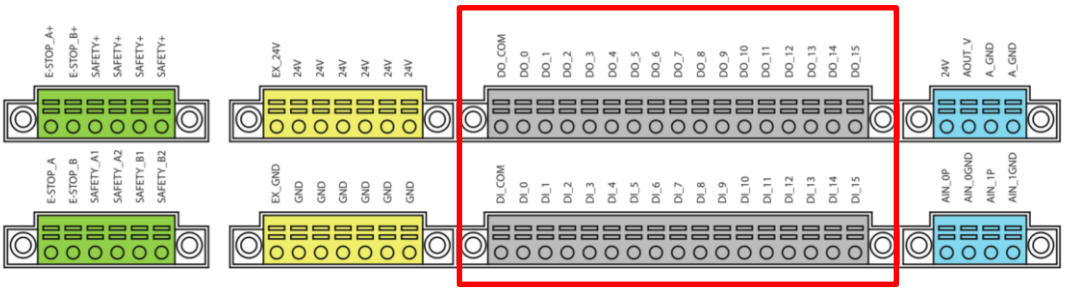


Ilustración 44. Conexionado I/O digitales TM5-900. [31]

Según la conexión realizada se pueden usar como tipo sinking, las cuales proporcionan una conexión a tierra para la carga, o sourcing, las cuales proporcionan una fuente de voltaje para la carga, esto se consigue de manera muy sencilla.

5.2.3.3.4. Entradas y salidas analógicas

Por ultimo explicamos las entradas y salidas analógicas, las cuales aceptan un voltaje desde los -10V a los +10V. Y el conexionado es el que se puede ver en las siguientes imágenes.

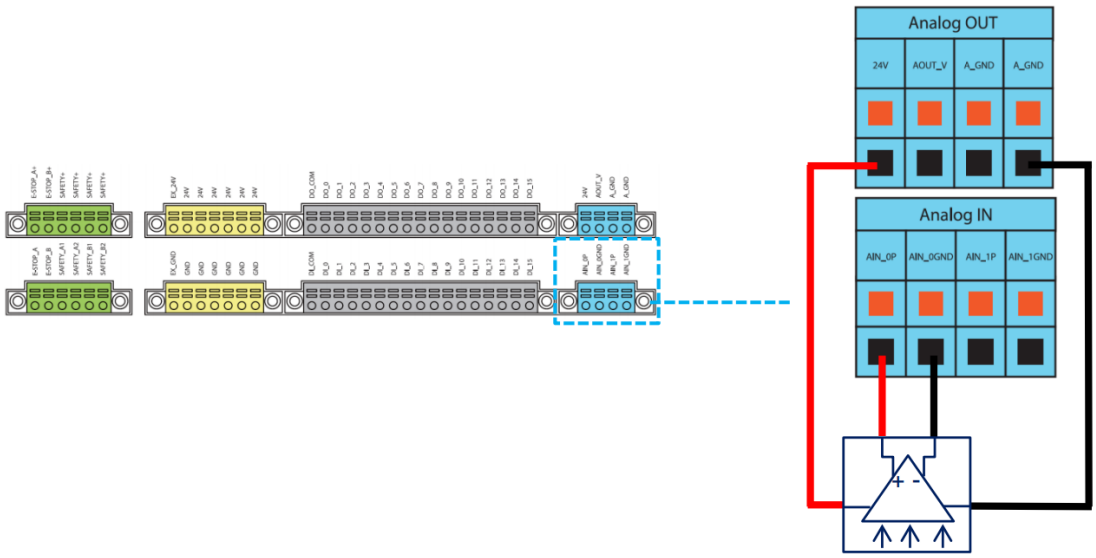


Ilustración 45. Conexionado entradas analógicas TM5-900. [31]

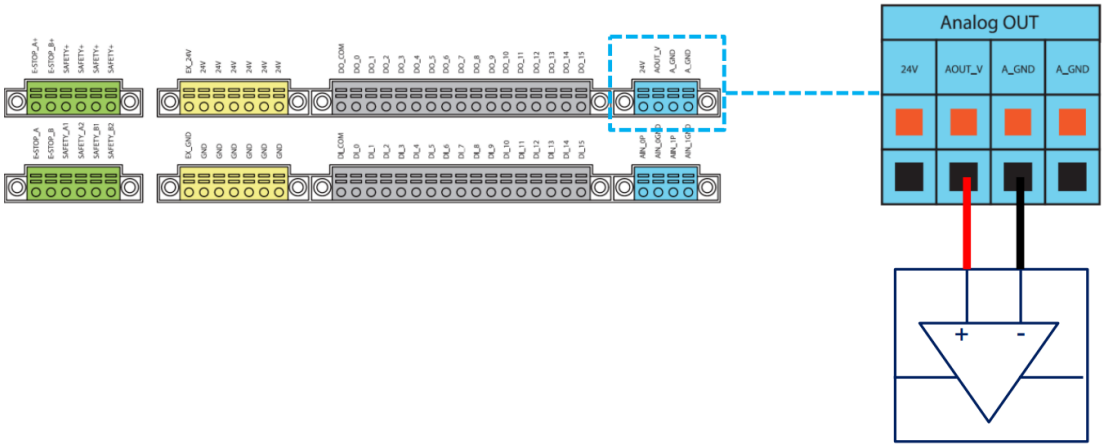


Ilustración 46. Conexionado salidas analógicas TM5-900. [31]

5.2.3.3.5. Interface de potencia

Para la conexión de potencia del controlador se utiliza un cable de alimentación con un conector IEC. Tal y como se puede ver en la siguiente imagen. La alimentación dada es de 24V a 200W, pero puede soportar como máximo 60V a 1500W.

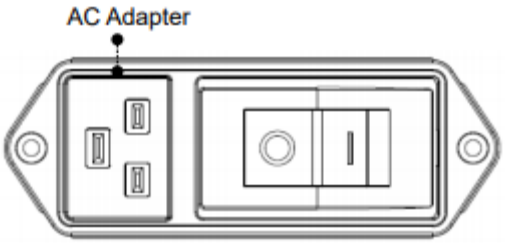


Ilustración 47. Conexionado de alimentación del controlador. [31]

La conexión de potencia del brazo robótico viene dada por el controlador, y se conecta a él mediante el conector mostrado en la siguiente ilustración. [31]

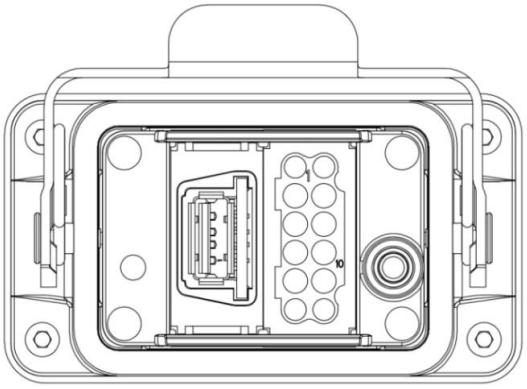


Ilustración 48. Conexionado de potencia del brazo robótico. [31]

5.2.3.4. TMflow

TMflow es el programa usado para la programación de la serie TM. TMflow es un programa que usa una interface gráfica, y su función es proporcionar a los usuarios una interfaz completa, conveniente y simple para entornos de programación lógica y de movimiento de robots. A través de un ordenador o una tablet, los usuarios pueden simplemente administrar y configurar los parámetros del robot y usar el diagrama de flujo gráfico para planificar el movimiento del robot y la lógica del proceso. Todo esto con la opción de administrar varios robots desde un solo dispositivo Windows, ya sea una tablet o un ordenador. [37]

Ilustración 49. Interface de programación TMflow.[38]

5.2.4. Pinza 2F-140

La pinza seleccionada para este proyecto es de la marca ROBOTIQ, y se trata del modelo 2F-140, la cual posee dos dedos articulados, cada uno de ellos con dos falanges. La pinza de agarre puede enganchar hasta cinco puntos de contacto con un objeto, dos en cada una de las falanges más la palma. Los dedos son under-actuated, lo que significa que tienen menos motores que el número total de articulaciones. Esta configuración permite que los dedos se adapten automáticamente a la forma del objeto que agarran y también simplifica el control de la pinza y el tipo agarre. Posee una apertura de pinza de 140 mm, más que de sobras para la realización del proyecto.[39]

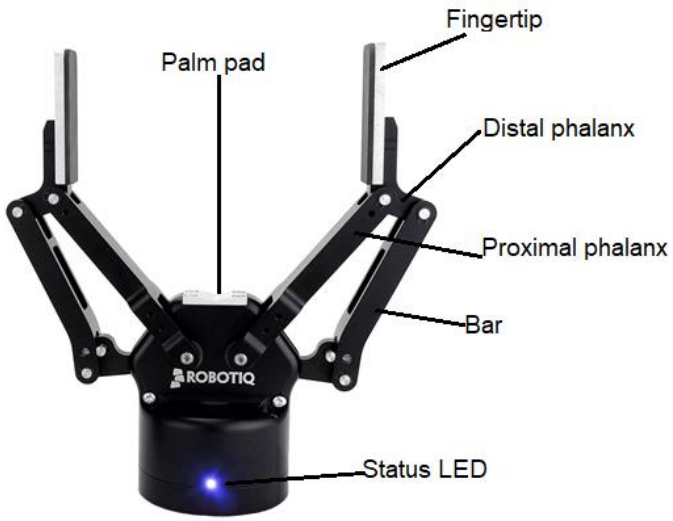


Ilustración 50. Partes de la pinza 2F-140. [40]

El Status LED sirve para saber el estado de la pinza, y según el color y el parpadeo significan una cosa:

- Azul/rojo fijo al arrancar
- Azul fijo cuando se enciende sin errores (mientras la comunicación está activa)
- Rojo fijo si se produce una falla menor, consulte los detalles de estado en la sección Control.
- Parpadeando en rojo/azul si ocurre una falla mayor, vea los detalles de estado en la sección Control.

Este tipo de pinza solo posee un actuador para abrir y cerrar los dedos, por lo que estos se adaptan automáticamente a la forma del objeto manipulado. Ya sea un agarre paralelo o un agarre envolvente, tal y como se muestra en la siguiente ilustración.

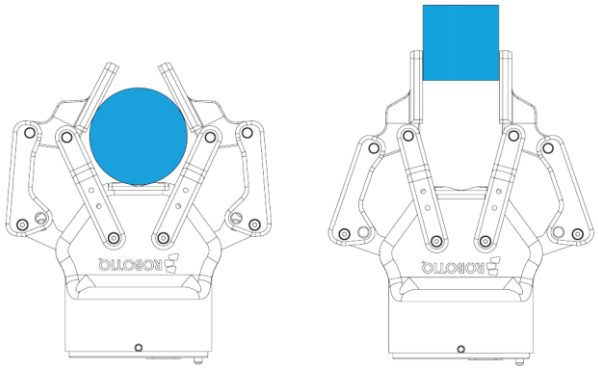


Ilustración 51. Adaptación de la pinza 2F-140. [38]

Este tipo de pinza permite tanto el agarre externo como el interno, ya que puede aplicar presión en las dos direcciones, como se puede ver en la siguiente ilustración.

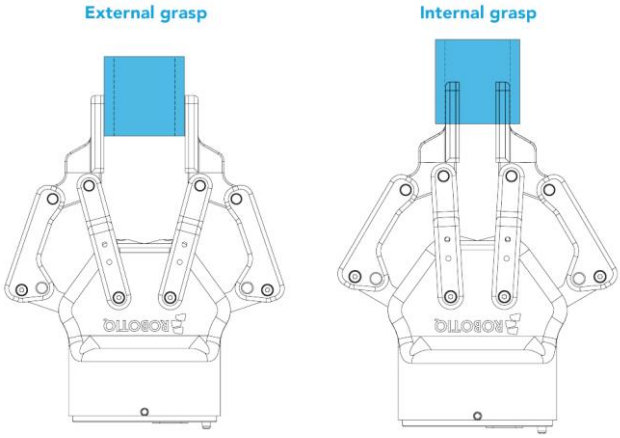


Ilustración 52. Tipos de agarre pinza 2F-140. [38]

La pinza se alimenta y controla directamente a través de un único cable de dispositivo que lleva un suministro de 24 VCC y comunicación Modbus RTU a través de RS-485. La propia pinza tiene detectores de presión, para saber cuándo ha sido cogido el objeto y el estado del mismo durante todo el proceso. También es necesario realizar ajustes de presión, apertura y cerrado, dependiendo siempre del proceso a realizar.

Para la instalación de este gripper es necesario de realizar un proceso muy simple, ya que este tipo de robots están creados para realzar cambios en sus procesos rápidamente. Primero debemos de atornillar una base donde va conectado el cable, y esta lleva unos pines para poder conectar la pinza. Después debemos de atornillar sobre esta base la pinza, haciendo coincidir los pines mencionados anteriormente. Y por último debemos de poner el protector de seguridad de la muñeca. Este proceso lo podemos ver en la siguiente ilustración.

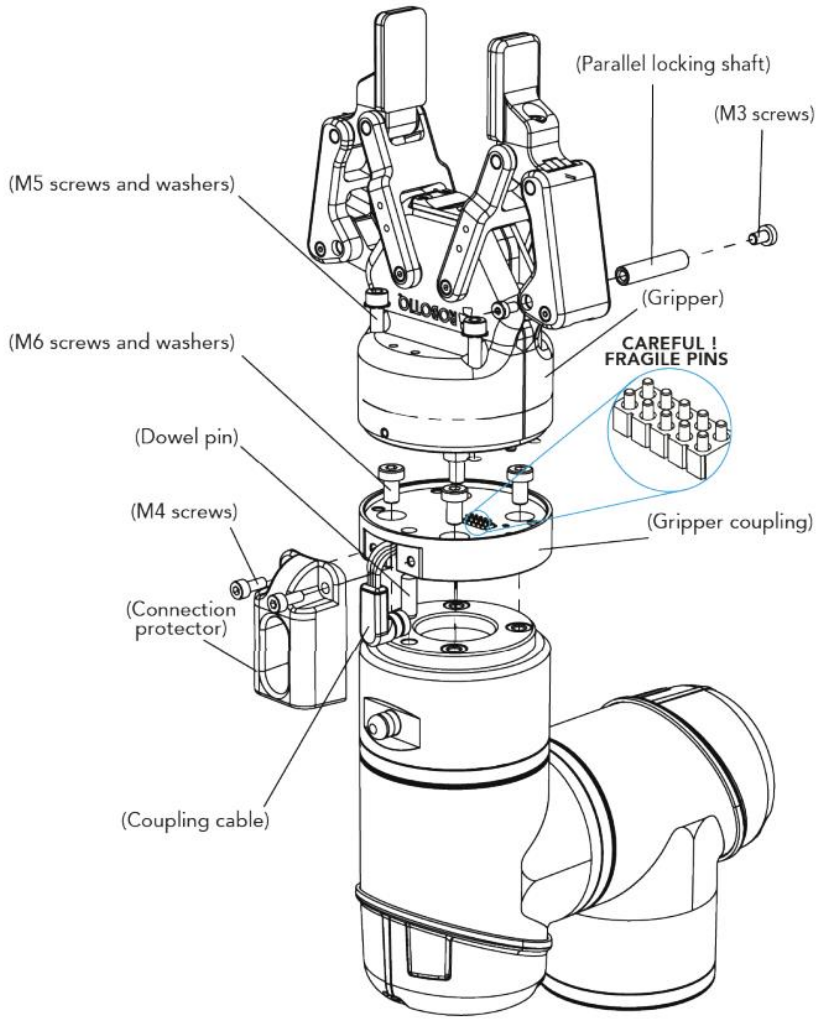


Ilustración 53. Instalación de la pinza 2F-140. [38]

La pinza interactúa con su acoplamiento a través de un conector de clavija de 10 resortes ubicado en su superficie exterior. [39]

Por ultimo para el uso de esta pinza en el programa TMflow debes de descargarte unos function-blocks, para poder realizar la apertura, el cerrado y la configuración.

5.3. DESARROLLO EN SYSMAC STUDIO

5.3.1. Introducción y configuración de Sysmac Studio

Para comenzar a programar en Sysmac Studio se debe de poseer la licencia del programa, esta se puede adquirir en la página web de Omron. Una vez instalado y validada la licencia se abre el programa y nos aparece la ventana mostrada en la siguiente ilustración.

Ilustración 54. Ventana de inicio Sysmac Studio.

En esta ventana se nos ofrecen las opciones de crear nuevo proyecto, abrir un proyecto, importar, exportar o conectar con un dispositivo. En primera instancia seleccionaremos "Nuevo proyecto", lo que saca por pantalla la siguiente ventana emergente.

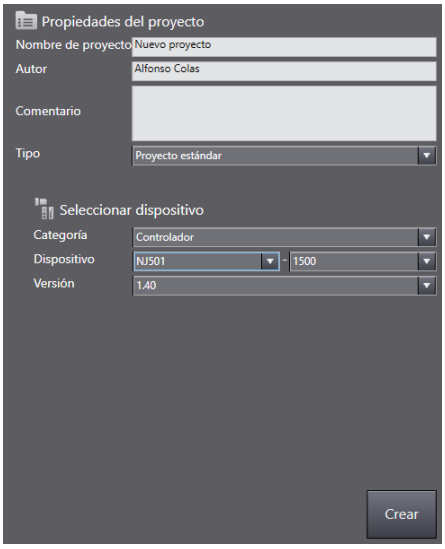


Ilustración 55. Crear nuevo proyecto en Sysmac Studio.

En la anterior ventana debemos de realizar la selección de las características de nuestro proyecto. Dentro del desplegable "Tipo" podemos seleccionar los siguientes tipos de proyecto a crear:

- Proyecto estándar: Simplemente para crear un proyecto para una aplicación.
- Proyecto de biblioteca: Se usa cuando tenemos un proyecto con un determinado tipo de controlador de una determinada configuración en la que vamos a realizar varios proyectos usando dicha configuración.
- Proyecto IAG: Esta opción nos permite la posibilidad de creación de un entorno de desarrollo muy similar al de una página HMI estándar con algunas excepciones para navegar por las pantallas.

Para nuestra aplicación debemos de seleccionar el "Proyecto estándar".

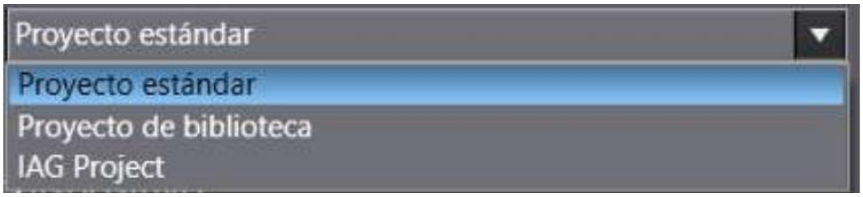


Ilustración 56. Tipos de proyecto Sysmac Studio.

El siguiente paso será seleccionar el tipo de dispositivo con el cual vamos a trabajar, en nuestra aplicación concreta empezaremos seleccionando la pestaña del "Controlador", ya que el HMI lo añadiremos posteriormente.

Ilustración 57. Selección del dispositivo en Sysmac Studio.

Posteriormente, en los desplegables "Dispositivo" y "Versión" seleccionaremos el modelo de nuestra CPU y la versión de la misma, dicha información la podremos encontrar en una etiqueta adhesiva que se encuentra en la sección trasera de nuestro controlador o bien en el datasheet del mismo. La configuración inicial en nuestro caso quedaría de la siguiente manera, seleccionamos nuestro modelo de CPU y la versión.

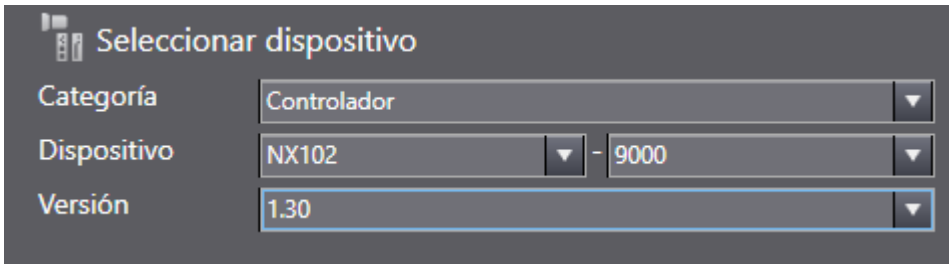


Ilustración 58. Selección de dispositivo en Sysmac Studio.

Tras haber creado el proyecto se abrirá la ventana de aplicación de Sysmac Studio, en la cual podemos diferenciar dos desplegables principales, tendremos por una parte el desplegable "Configuraciones y ajustes" y por otra parte "Programación".

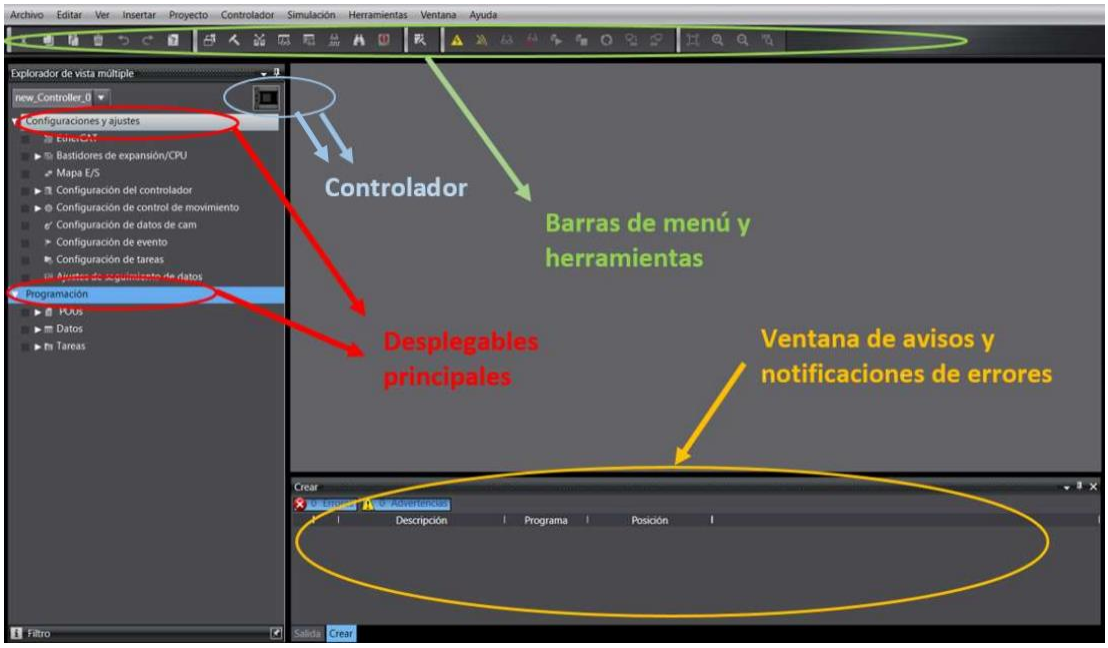


Ilustración 59. Ventana de aplicación de Sysmac Studio.

Para realizar los guardados de seguridad deberemos de realizar la exportación del proyecto desde la barra de menú, en la opción "Archivo" y seleccionamos en su desplegable el botón "Exportar".

Una vez presentada la ventana de aplicación pasamos a detallar toda la parte de configuración, cabe decir que existen multitud de configuraciones y protocolos de comunicación como puede ser Ethercat en el que se presenta un esquema de conexiones de los distintos dispositivos que tenemos conectados a través del puerto Ethercat, simplemente con la conexión de los diversos componentes a través de la red propuesta podemos traernos la configuración de dichos componentes, es decir, detecta la configuración de las unidades de periferia automáticamente, pero para nuestra aplicación no hemos hecho uso de la red Ethercat.

5.3.1.1. Bastidores de expansión

El primer paso será configurar el bastidor de expansión de la CPU, en esta opción integraremos todas las tarjetas o dispositivos entrada o salida de los cuales vamos a hacer uso en nuestra aplicación, podemos insertarlos uno a uno o podemos directamente seleccionar la opción de que el controlador reconozca directamente los dispositivos conectados a la CPU mediante una prueba de conectividad del bus NX.

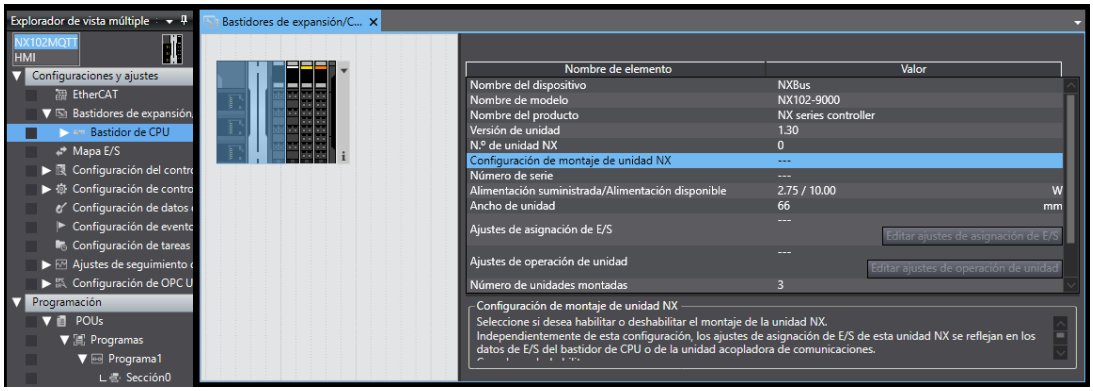


Ilustración 60. Bastidores CPU en Sysmac Studio.

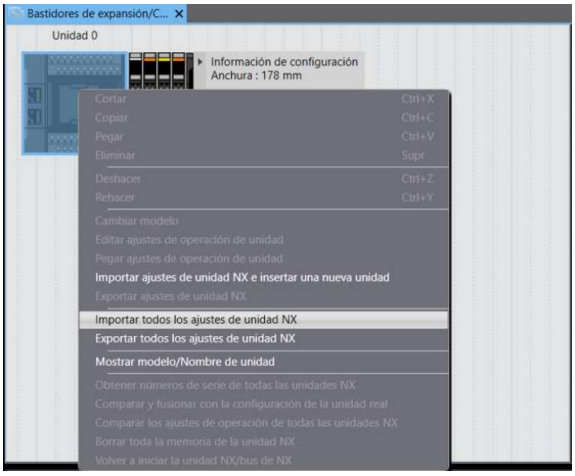


Ilustración 61. Importar bastidores automáticamente en Sysmac Studio.

5.3.1.2. Mapa de entradas y salidas

En este apartado de la configuración será donde determinemos e identifiquemos las variables de cada dispositivo, en Sysmac Studio las variables están separadas en distintos niveles pertenecientes a los dispositivos entrada/salida que estén conectados a nuestra CPU, de tal manera que la propia CPU tendrá una serie de entradas y salidas digitales correspondientes a cada uno de los compartimentos de conexión, por otra parte los periféricos conectados a la CPU, que en este caso son las tarjetas de ampliación tendrán sus propias variables internas asignadas a cada una de las entradas del periférico, a las cuales podremos asignar un nombre a nuestro parecer. Las variables aquí descritas son variables que se les asignan a un estado de señal correspondiente a una salida o entrada, sin embargo, podemos utilizar otro tipo de variables que definiremos posteriormente en el apartado de "Programación" para almacenar resultados de operaciones o cualquier tipo de información de interés.

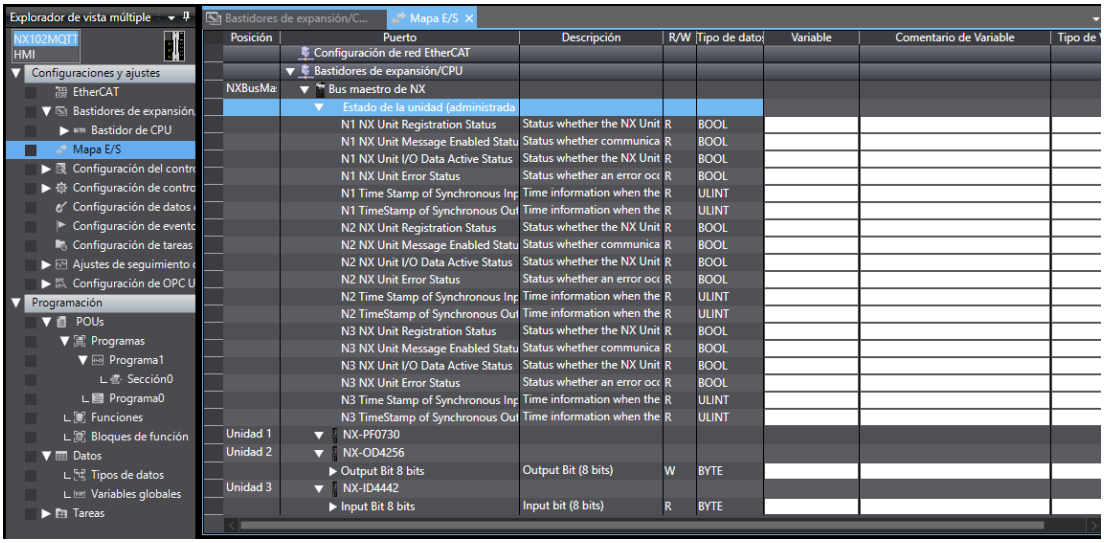


Ilustración 62. Configuración de variables en periféricos en Sysmac Studio.

5.3.1.3. Configuración del controlador

En esta sección se realizarán ajustes a nivel de comunicaciones; como puede ser la configuración del puerto Ethernet/IP, la configuración de operación; donde básicamente definiremos las configuraciones de arranque de nuestra CPU, por ejemplo, si queremos que arranque en modo "PROGRAM" el PLC ejecutará el programa, leyendo datos de las entradas y salidas conectadas a la CPU pero no se conectará On-line con el PLC, esto es muy útil a la hora de detectar algún tipo de error que podamos cometer a la hora de configurar las comunicaciones o las conexiones con sensores o dispositivos que hayamos anexionado a nuestra CPU.

Por otra parte, si decimos arrancar la CPU en modo "RUN", tras iniciarse el sistema se compilará toda la información y se llevará al PLC, de tal manera que el sistema no permanecerá en esa especie de "modo espera" en el cual se encontraba cuando definíamos las condiciones de arranque en modo "PROGRAM", tras arrancar en "RUN" se transmitirá todas las acciones de control al PLC y el programa se transmitirá a dicha terminal.

5.3.1.3.1. Configuración de operación

Para este proyecto dejamos todos las configuración tal y como vienen predeterminadas.

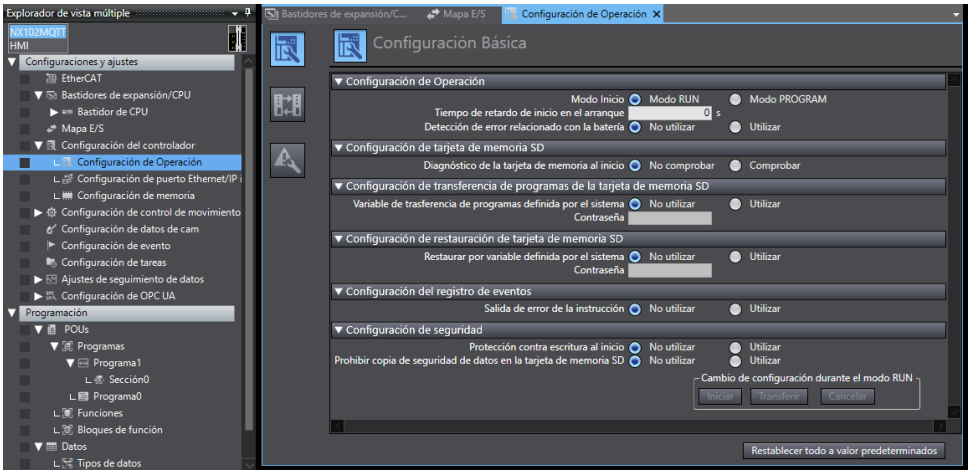


Ilustración 63. Configuración de operación en Sysmac Studio.

Dentro de este apartado existe una parte en la cual podemos definir los tipos de eventos de nuestro sistema, es decir, podemos clasificar las situaciones que se produzcan en nuestro controlador aportando un grado de significación y detallando cuales son las labores automáticas que debe realizar el mismo. Simplemente comentamos esta posibilidad pero no es objeto de estudio para este trabajo.

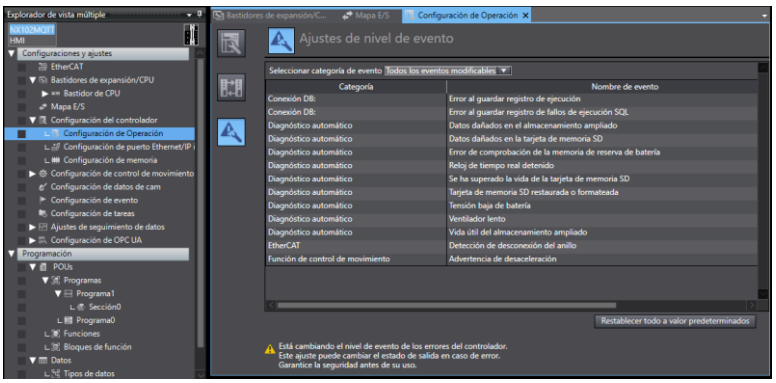


Ilustración 64. Configuración de eventos en Sysmac Studio.

5.3.1.3.2. Configuración de puerto Ethernet/IP integrado

En esta sección concretaremos la dirección IP que poseerá nuestro controlador dentro de la red para que sea posible la comunicación entre los dispositivos. También indicamos en el puerto en el que va a ser conectado el cable Ethernet.

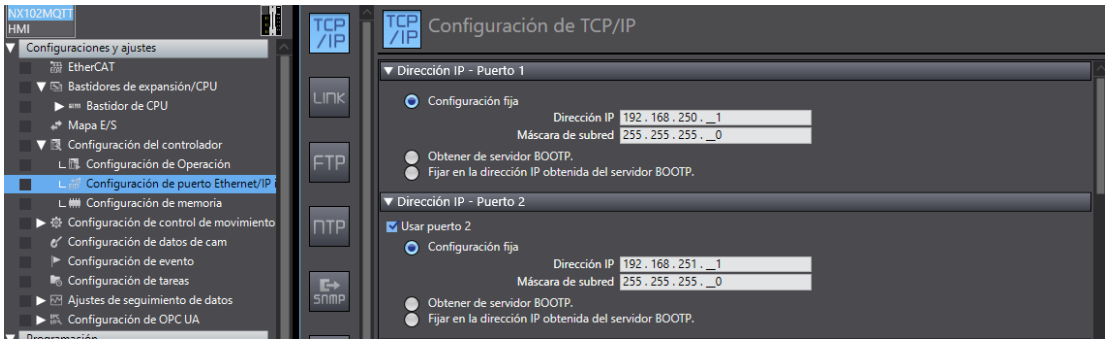


Ilustración 65. Configuración Ethernet/IP en Sysmac Studio.

En nuestro caso utilizaremos la dirección IP 192.168.250.001 y la máscara de subred será creada automáticamente. Esta dirección es usada ya que se encuentra dentro del rango de direcciones IPv4 para uso interno, dentro de una red cerrada. Cada elemento que añadamos al proyecto estará en esta red, y solo modificaremos los tres últimos dígitos. El ordenador portátil usado para la programación usara la dirección IP 192.168.250.040 para poder estar dentro de la red cerrada.

5.3.1.3.3. Configuración de la memoria

En versiones antiguas tanto del programa como de la CPU, la memoria estaba separada en una serie de áreas que determinaban el tipo de instrucciones realizadas por la correspondiente CPU, actualmente con Sysmac Studio no se realiza esa segmentación de la memoria en áreas, sino que es todo mucho más sencillo e intuitivo, sin embargo si queremos realizar comunicaciones con otro PLC de una serie más antigua, es necesario realizar la susodicha separación de la memoria en áreas para que pueda realizarse con satisfacción el intercambio de información entre las dos CPUs. En nuestra aplicación no usaremos estas áreas, por lo que la configuración será la visualizada e la siguiente ilustración.

Configuración de memoria para unidades de la serie CJ

Activar	Tipo de área	Tamaño (número de canales	Retener
☐	CIO	6144	Sin retención
☐	WR	512	Sin retención
☐	HR	512	Con retención
☐	DM	32768	Con retención
☐	EM0	32768	Con retención
☐	EM1	32768	Con retención
☐	EM2	32768	Con retención
☐	EM3	32768	Con retención
☐	EM4	32768	Con retención
☐	EM5	32768	Con retención
☐	EM6	32768	Con retención
☐	EM7	32768	Con retención
☐	EM8	32768	Con retención
☐	EM9	32768	Con retención
☐	EMA	32768	Con retención
☐	EMB	32768	Con retención
☐	EMC	32768	Con retención
☐	EMD	32768	Con retención
☐	EME	32768	Con retención
☐	EMF	32768	Con retención
☐	EM10	32768	Con retención
☐	EM11	32768	Con retención
☐	EM12	32768	Con retención
☐	EM13	32768	Con retención
☐	EM14	32768	Con retención
☐	EM15	32768	Con retención
☐	EM16	32768	Con retención
☐	EM17	32768	Con retención
☐	EM18	32768	Con retención

Ilustración 66. Configuración de las áreas de la memoria en Sysmac Studio.

5.3.1.3.4. Ajuste de seguimiento de datos

Esta parte de la configuración está destinada a la captación de datos y valores de las variables durante el proceso, de tal manera que podemos realizar gráficas de diagnóstico o meramente gráficos en los que observemos la evolución de una cierta variable a lo largo del tiempo, es de gran utilidad a la hora de llevar un control visual de la instalación. Podemos monitorizar la gráfica en todo momento, incorporando diferentes lecturas superpuestas sobre la misma gráfica, en la misma sección se pueden realizar infinidad de configuraciones que nos permitan una monitorización óptima de las variables del proceso, podemos ajustar los rangos máximos y mínimos de la escala, obtener el valor medio de una variable, registrar máximos o mínimos, modificar el tiempo de muestreo etc. , a su vez también podemos guardar ese resultado de supervisión para la realización de posteriores análisis.

Debido a que nuestro proyecto no recibe datos directamente de ningún sensor, no vamos a utilizar esta herramienta.

5.3.1.3.5. Configuración de comunicaciones

Esta opción se encuentra en la parte superior del menú de herramientas que encontramos en la ventana de aplicación nada más iniciar Sysmac Studio concretamente en "Controlador", en este apartado podemos configurar el método de comunicación que tendremos con el PLC.

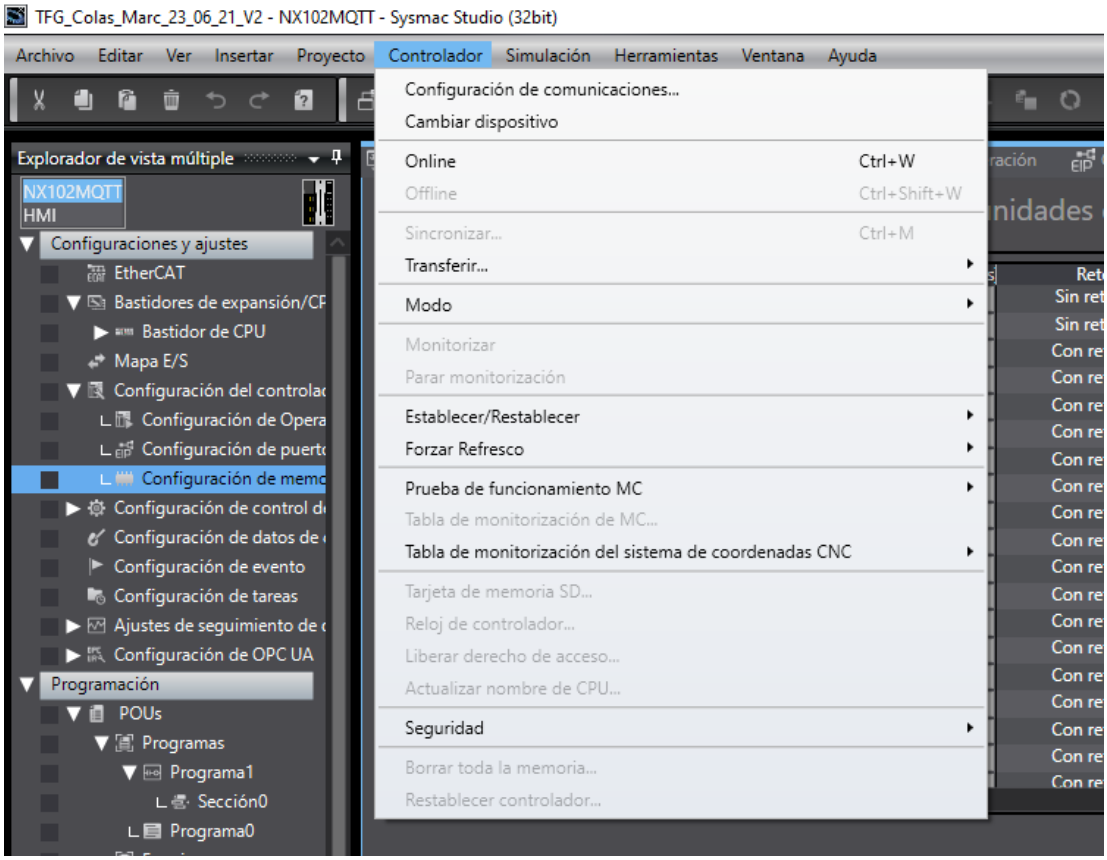


Ilustración 67. Configuración de comunicaciones en Sysmac Studio.

Si accedemos a la opción "Configuración de comunicaciones", aparecerá un menú emergente en el cual podemos detallar el tipo de conexión que se llevará a cabo para conectarse con el controlador. Existen diferentes tipos de conectividad, en nuestro caso hemos conectado directamente la CPU a través de Ethernet con el ordenador.

Una vez hemos determinado el modo de conexión, podemos realizar una prueba de conectividad a través de la opción "Prueba de comunicaciones Ethernet", en el recuadro inmediatamente inferior nos aparecerá el resultado de dicha prueba.

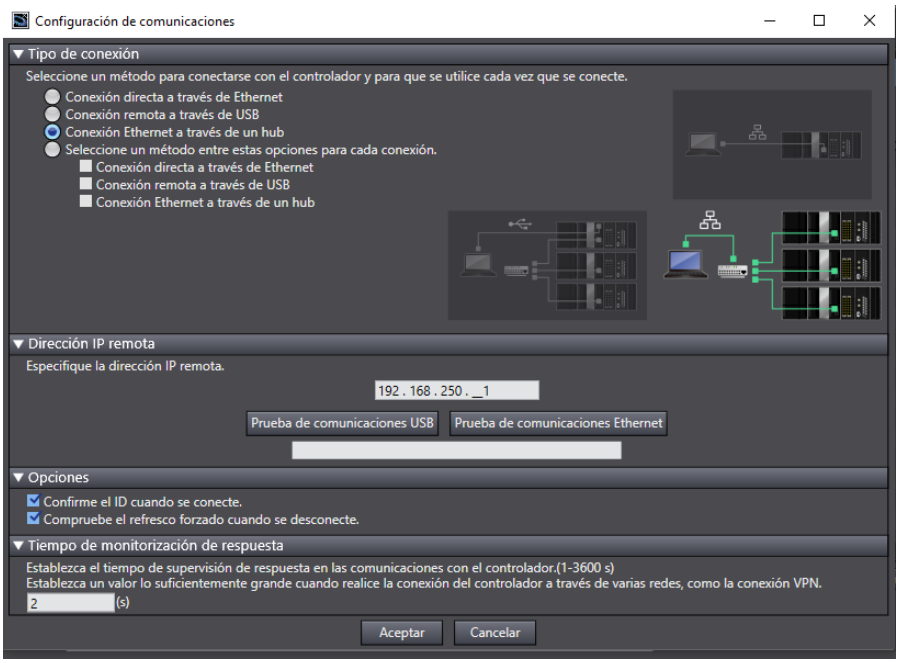


Ilustración 68. Prueba de conexión mediante Ethernet en Sysmac Studio.

5.3.1.3.6. Transferencia de datos al controlador

Una vez hallamos terminado de elaborar nuestro programa el paso siguiente es transferir dicho programa al controlador, para ello seguiremos los pasos a continuación indicados.

Comprobamos que la compilación de todos los programas no ha causado ningún error, para ello accedemos a la opción "Proyecto", la cual se encuentra en la parte superior del menú de herramientas junto a la opción "Controlador", una vez hallamos accedido a esta opción seleccionamos "Comprobar todos los programas", Sysmac Studio se encargará de compilar todo el código y nos indicará si existe algún error.

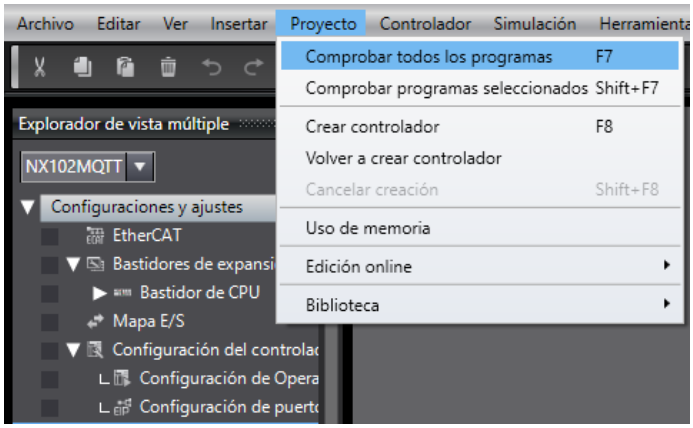


Ilustración 69. Comprobación de programas en Sysmac Studio.

Paso siguiente a la comprobación, en la misma sección de "Proyecto", seleccionamos "Crear controlador", de esta manera se transmitirán los datos a la CPU, pero no se llevará el sistema a "ONLINE".

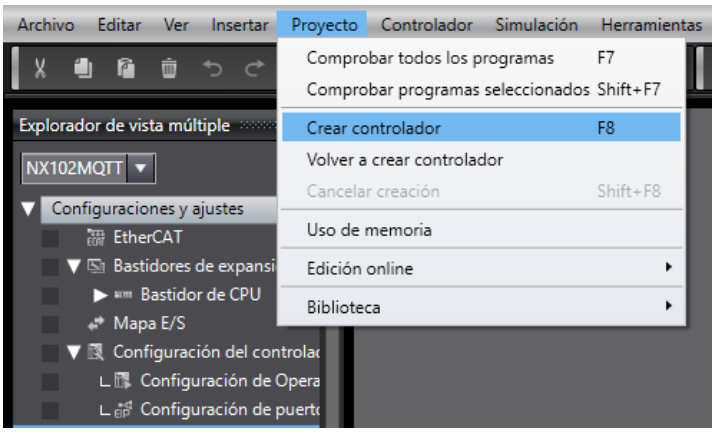


Ilustración 70. Crear controlador en Sysmac Studio.

A continuación dentro del menú "Controlador", seleccionamos "Transferir" -- "transferir datos al controlador" y por último le damos a "Sincronizar".

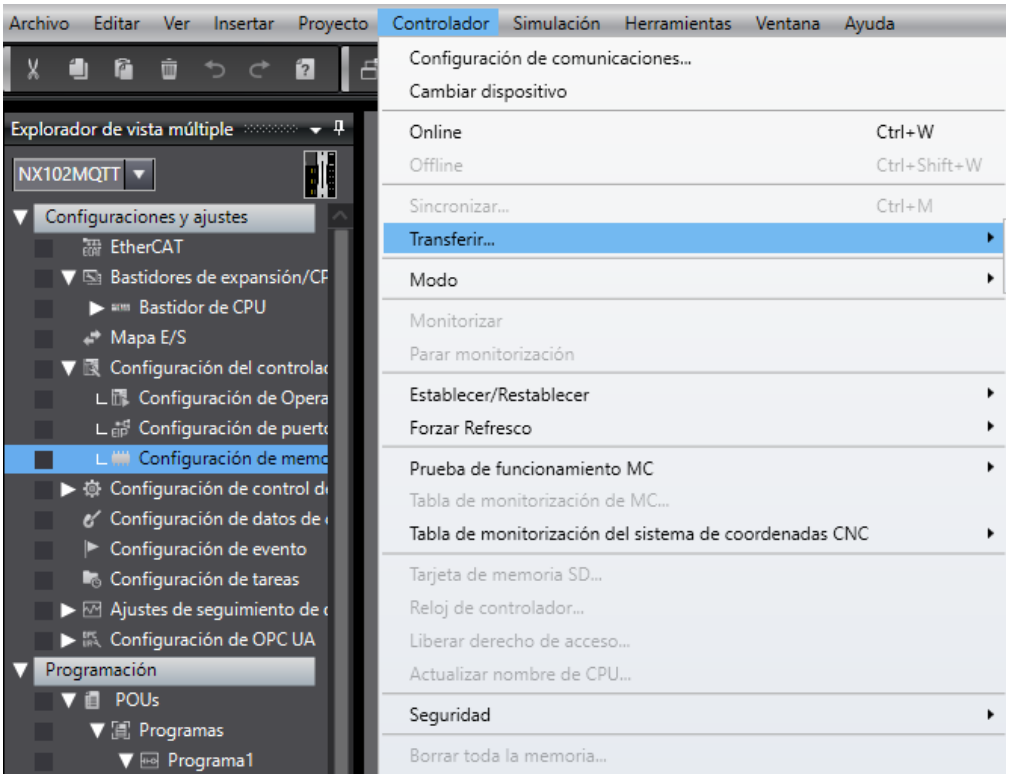


Ilustración 71. Sincronizar y transferir al controlador en Sysmac Studio.

Es importante que antes de llevar el controlador a estado online, realicemos una simulación que nos permita visualizar el flujo de código y nos permita detectar si existe alguna incongruencia o problema en el proceso, de esta manera podremos corregirlo antes de conectar con el controlador. Para ello, accedemos a "Simulación" y seleccionamos la opción "Ejecutar en modo PROGRAM".

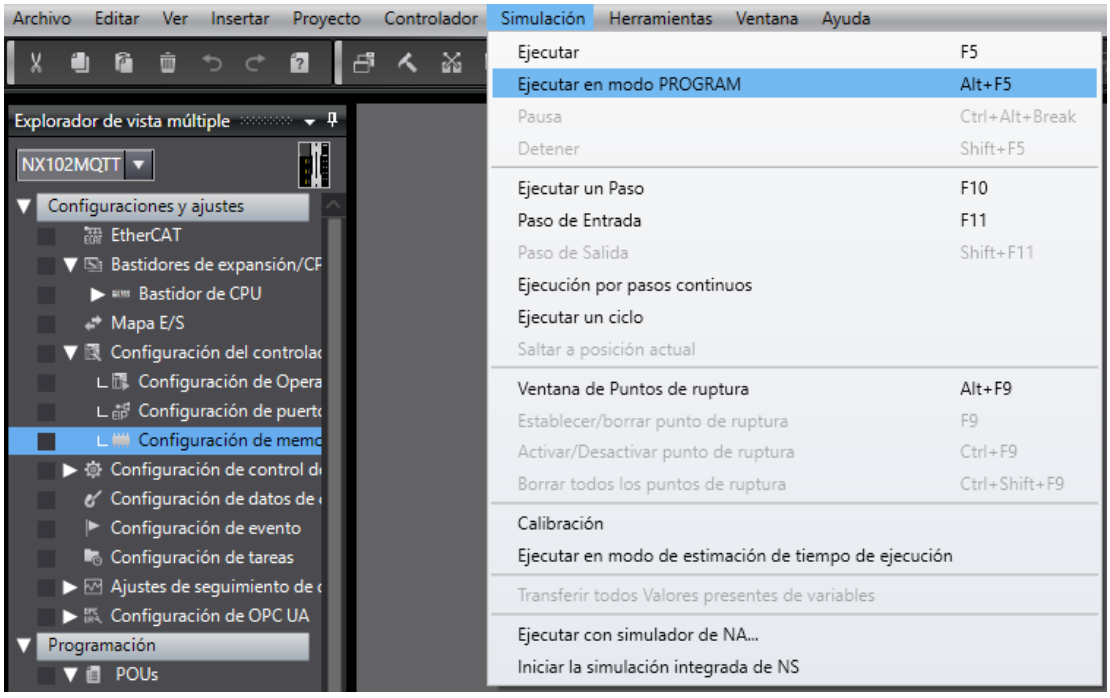


Ilustración 72. Simulación en modo PROGRAM en Sysmac Studio.

El siguiente paso será acceder al modo "ONLINE", los datos se cargarán en el controlador y dará lugar la tarea de control. Para ello, nos dirigimos a la barra de herramientas y seleccionamos la opción "ONLINE", o bien accedemos a la pestaña de "Controlador" y seleccionamos la opción anteriormente dicha.

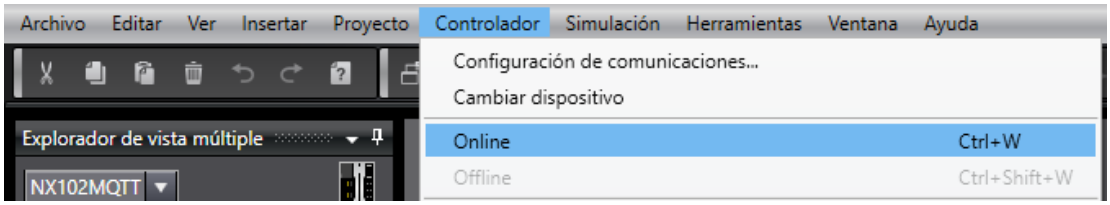


Ilustración 73. Modo Online en Sysmac Studio.

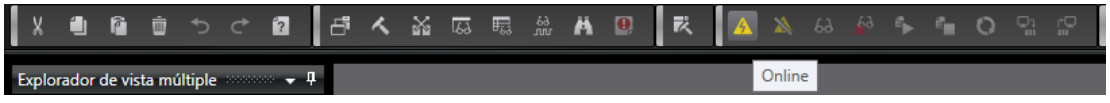


Ilustración 74. Modo Online de acceso rápido en Sysmac Studio.

Una opción muy interesante a la hora de interactuar con el código en el modo de simulación es la creación de una "página de vigilancia", gracias a esta opción podemos modificar el valor de las variables del proceso, simulando un tipo de comportamiento para comprobar que nuestro programa realiza las secuencias que nosotros esperamos, podemos verificar si existirá algún punto singular en el programa con anterioridad, esto nos permitirá anticiparnos a errores de programación, es coherente que estemos seguros de que no existirá ningún tipo de problema cuando el sistema de control esté operativo. Para ello accedemos al desplegable "Ver" y seleccionamos la opción "Página de Vigilancia".

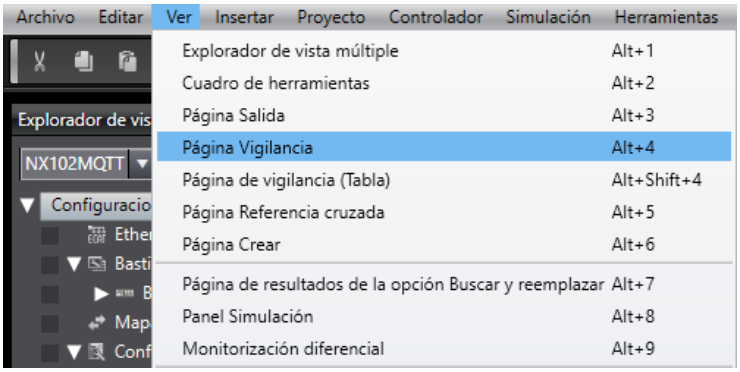


Ilustración 75. Insertar página de vigilancia en Sysmac Studio.

En la parte inferior de la ventana de aplicación aparecerá una tabla que mostramos a continuación:

Ilustración 76. Visualización de la página de vigilancia en Sysmac Studio.

Podemos observar que se divide la información en 8 columnas, comprendidas por; nombre de la variable que estamos "vigilando", "valor online" es simplemente el valor que toma dicha variable durante la ejecución del programa, en la columna modificar podremos forzar el valor de la variable de estudio a uno de nuestro interés, esto solo será posible si nos encontramos en modo simulación o modo "PROGRAM", las demás columnas constituyentes de la tabla nos dan información sobre el tipo de dato de la variable y si está relacionada con una entrada o salida.

Para añadir una variable a la página de vigilancia simplemente añadimos el nombre del dispositivo del que queremos visualizar la variable en la columna "Nombre del dispositivo", y la referencia de la variable en la columna "nombre", el programa directamente nos busca las variables creadas que existen por ese nombre y nos permite integrarlas en la ventana de vigilancia. Lógicamente debemos haber creado con anterioridad la variable para poder introducirla en la tabla.

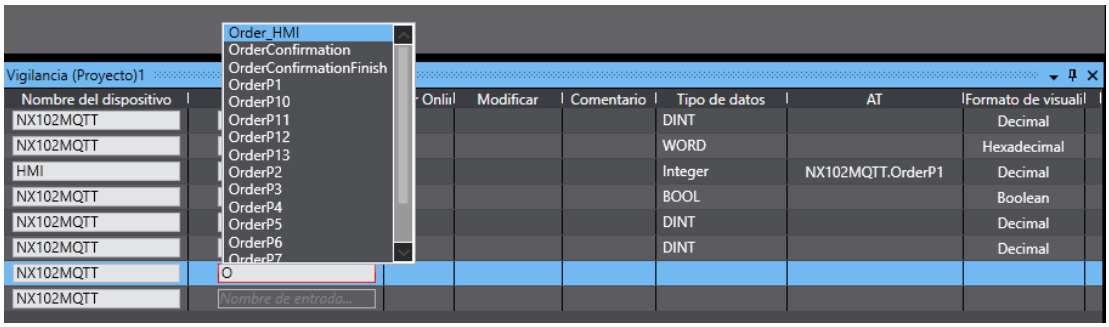


Ilustración 77. Insertar variables en la página de vigilancia en Sysmac Studio.

5.3.2. Programación del PLC en Sysmac Studio

El otro apartado principal en el que se divide el menú de la ventana de aplicación de Sysmac Studio es el apartado de "Programación", en dicha sección introduciremos la manera de crear un nuevo programa, cómo definir variables globales, creación de bloques de función, incorporación de archivos de texto estructurado y asignación de tareas principales en el proceso.

5.3.2.1. Creación de variables

Podemos crear variables a las cuales podemos acceder desde diferentes POUs (programas, funciones, bloques…). Para ello accedemos al menú de variables globales, situado en la parte inferior del desplegable de "Programación", dentro del apartado de "Datos", accedemos a "Variables globales". Una vez accedemos aparecerá una tabla emergente donde contenemos todas las variables globales que hemos definido en nuestra aplicación, para crear una nueva variable simplemente pulsamos con el botón derecho del ratón y seleccionamos la opción "Crear nuevo / insert", posteriormente especificamos el tipo de dato de la variable y a continuación un estado inicial de la misma, aunque esta última premisa no es obligatoria.

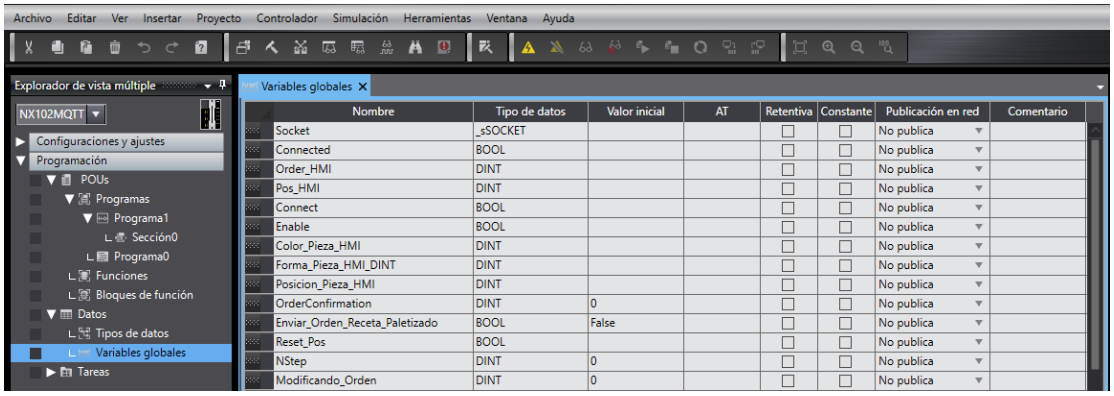


Ilustración 78. Visualización de variables globales en Sysmac Studio.

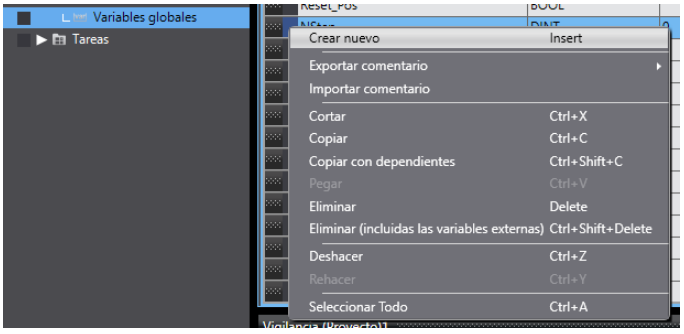


Ilustración 79. Creación de variables globales en Sysmac Studio.

5.3.2.2. Creación de programa

Para la creación de un programa simplemente desplegamos la opción "POUs", dentro del apartado "Programas" hacemos click derecho y seleccionamos "Añadir". Una vez aquí debemos de seleccionar el tipo de programa que queremos agregar.

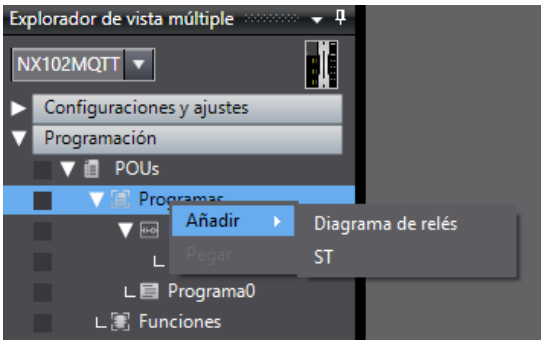


Ilustración 80. Creación de un programa en Sysmac Studio.

Comenzamos explicando la creación de un "Diagrama de relés", en el cual debes de añadir una "sección" para comenzar.

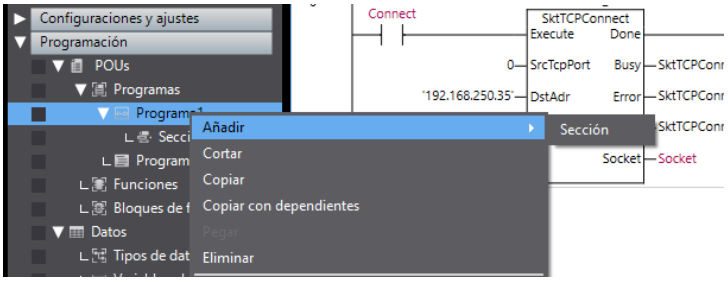


Ilustración 81. Insertar una sección en Sysmac Studio.

Una vez hayamos insertado la nueva sección, podemos acceder a ella, obteniendo la siguiente ventana.

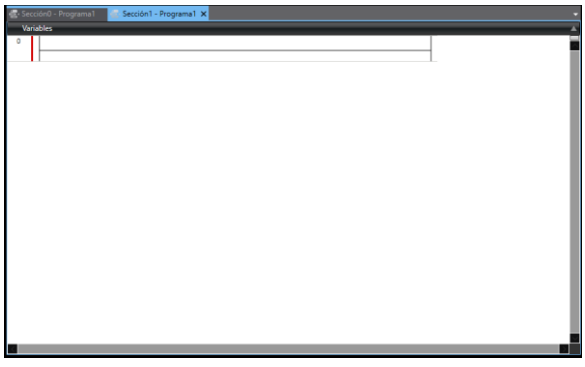


Ilustración 82. Ventana de sección en Sysmac Studio.

Posteriormente a crear nuestra sección o nuevo programa podemos hacer uso de la caja de herramientas que tenemos en la sección derecha de la ventana de aplicación, en dicho apartado encontraremos numerosas funciones que podrán servirnos a la hora de programar.

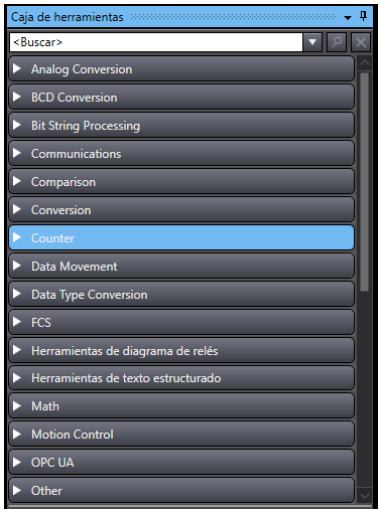


Ilustración 83. Caja de herramientas en Sysmac Studio.

En segundo lugar explicamos la creación de un programa en ST, es decir en texto estructurado. De la misma manera que para el anterior, realizamos clic izquierdo sobre "Programas" y en el desplegable seleccionamos "ST".

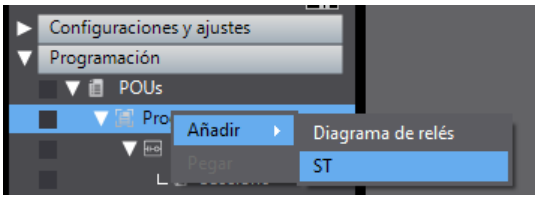


Ilustración 84. Insertar un programa en Structured Test en Sysmac Studio.

Una vez insertado la ventana que obtenemos es la siguiente, y sobre ella debemos de programar en texto estructurado, el cual es un lenguaje de marcas ligero creado para escribir textos de manera cómoda y rápida. Tiene la principal ventaja de que ese texto puede usarse para generar documentos equivalentes en HTML, TeX, docbook u otros lenguajes.

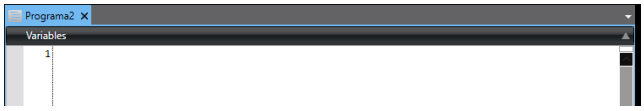


Ilustración 85. Ventana de ST en Sysmac Studio.

Por último, algo común para los dos tipos de programas es la asignación de variables internas, las cuales solo sirven para el programa en las que están definidas. Para la creación de estas variables se debe de abrir el desplegable situado en la parte superior de la ventana del programa, tal y como se puede ver en la siguiente ilustración.

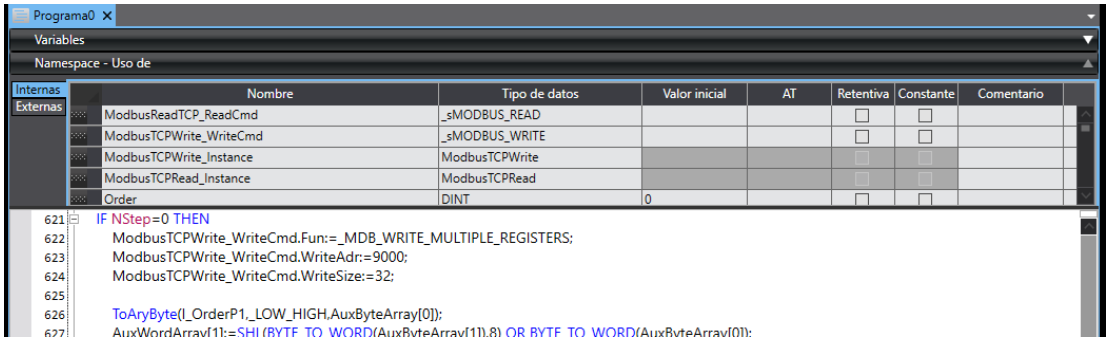


Ilustración 86. Ventana de variables internas de programa en Sysmac Studio.

Para insertar se debe de clicar con el botón izquierdo dentro de la pestaña, y en el desplegable obtenido se selecciona la opción insertar.

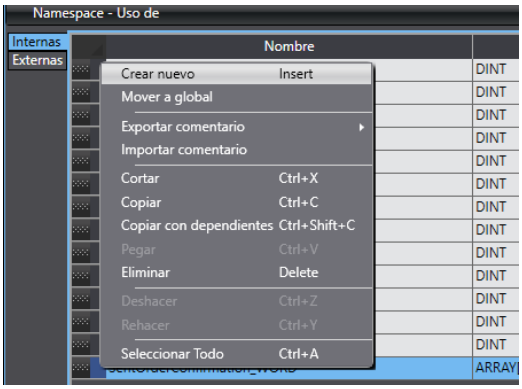


Ilustración 87. Insertar variable interna en Sysmac Studio.

Para crear una variable debes de asignarle un nombre, el cual no este repetido, el tipo de dato que es, y si es necesario el valor inicial.

5.3.2.3. Configuración de tareas

Las tareas nos sirven para designar una prioridad de eventos en el programa, esto es útil si tenemos varios programas en un mismo proyecto, podemos determinar que jerarquías o prioridades debe tener Sysmac Studio a la hora de procesar datos. Para acceder a la configuración de tareas nos dirigimos a la zona izquierda de la ventana de aplicación, concretamente en la parte "Configuración de tareas", en esta parte podremos definir la jerarquía y la prioridad comentadas anteriormente.

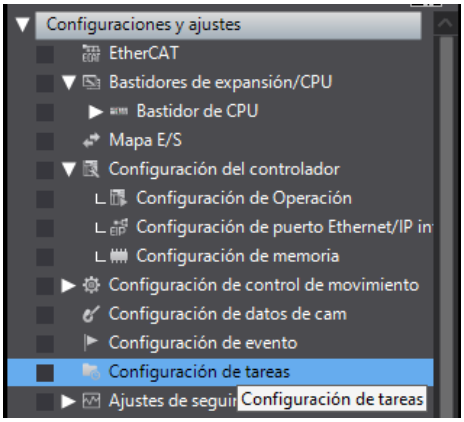


Ilustración 88. Configuración de tareas en Sysmac Studio.

Si tenemos un único programa en el proyecto, automáticamente se asignará como tarea principal, en este proyecto solo tenemos dos programas y los dos deben de estar al mismo nivel, por lo que esta parte no será modificada.

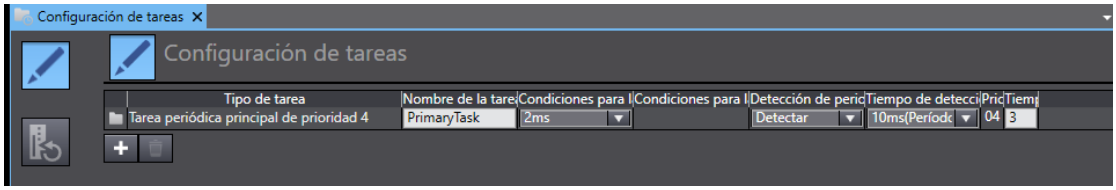


Ilustración 89. Asignación de la tarea principal en Sysmac Studio.

5.3.2.4. Realización del proyecto

Para la realización de este proyecto se utilizan dos programas, para la simplificación de la programación, ya que el primero de ellos está realizado en ST y el segundo en diagrama de relés.

El diagrama UML para explicar la programación de este proyecto es el siguiente.

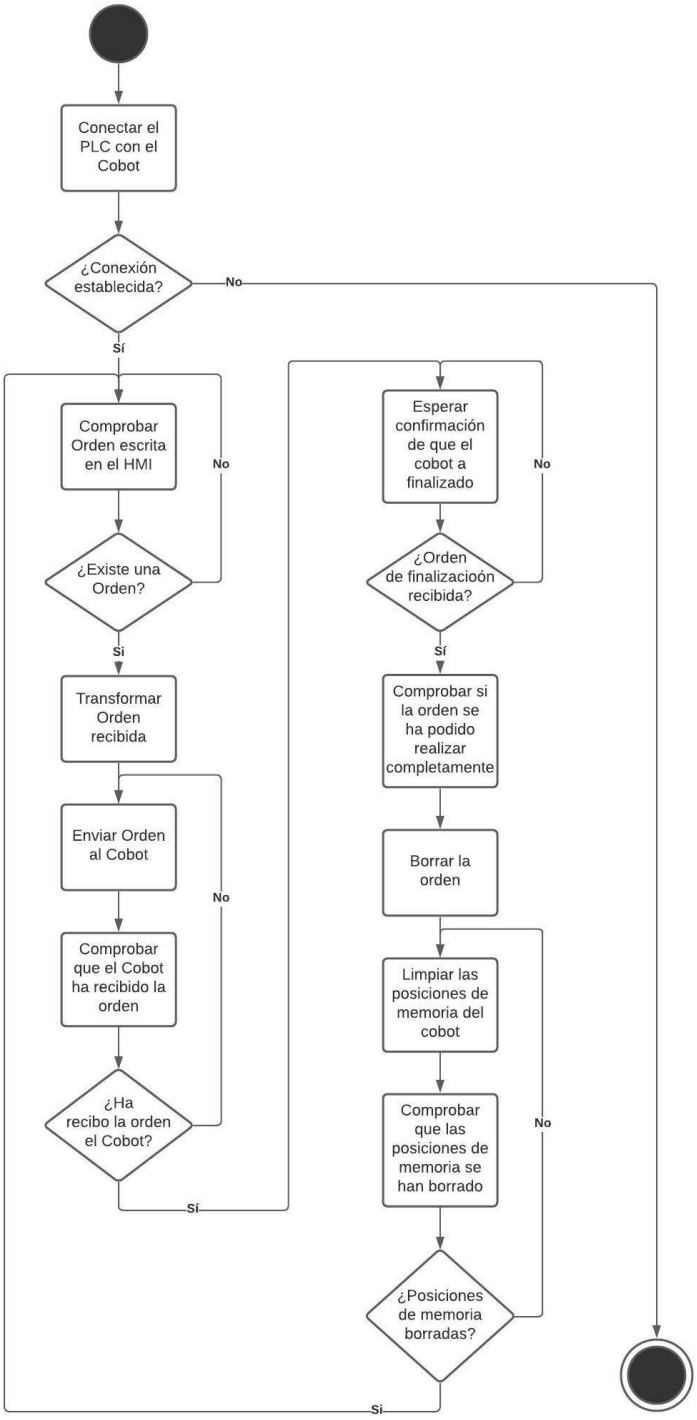


Ilustración 90. Diagrama UML de Sysmac Studio.

Las variables globales compartidas por todo el proyecto son muchas debido a que la mayoría son usadas por el HMI, y son las mostradas en las siguientes ilustraciones.

Variables globales

Nombre	Tipo de datos	Valor inicial	AT	Retentiva	Constante	Publicación en red
Socket	_sSOCKET			☐	☐	No publica
Connected	BOOL			☐	☐	No publica
Order_HMI	DINT			☐	☐	No publica
Pos_HMI	DINT			☐	☐	No publica
Connect	BOOL			☐	☐	No publica
Enable	BOOL			☐	☐	No publica
Color_Pieza_HMI	DINT			☐	☐	No publica
Forma_Pieza_HMI_DINT	DINT			☐	☐	No publica
Posicion_Pieza_HMI	DINT			☐	☐	No publica
OrderConfirmation	DINT	0		☐	☐	No publica
Enviar_Orden_Receta_Paletizado	BOOL	False		☐	☐	No publica
Reset_Pos	BOOL			☐	☐	No publica
NStep	DINT	0		☐	☐	No publica
Modificando_Orden	DINT	0		☐	☐	No publica
OrderP1	DINT			☐	☐	No publica
OrderP2	DINT			☐	☐	No publica
OrderP3	DINT			☐	☐	No publica
OrderP4	DINT			☐	☐	No publica
OrderP5	DINT			☐	☐	No publica
OrderP6	DINT			☐	☐	No publica
OrderP7	DINT			☐	☐	No publica
OrderP8	DINT			☐	☐	No publica
OrderP9	DINT			☐	☐	No publica
OrderP10	DINT			☐	☐	No publica
OrderP11	DINT			☐	☐	No publica
OrderP12	DINT			☐	☐	No publica
OrderP13	DINT			☐	☐	No publica
Error_Com	BOOL			☐	☐	No publica
Fin_Paletizado	BOOL			☐	☐	No publica
TipoPallet	DINT			☐	☐	No publica
Finalizacion_Palletizado	BOOL			☐	☐	No publica
OrderConfirmationFinish	DINT	0		☐	☐	No publica
BotonDespaletizado	BOOL			☐	☐	No publica
TM_Pos1	DINT			☐	☐	No publica
TM_Pos2	DINT			☐	☐	No publica
TM_Pos3	DINT			☐	☐	No publica
TM_Pos4	DINT			☐	☐	No publica
TM_Pos5	DINT			☐	☐	No publica
TM_Pos6	DINT			☐	☐	No publica
TM_Pos7	DINT			☐	☐	No publica
TM_Pos8	DINT			☐	☐	No publica
TM_Pos9	DINT			☐	☐	No publica
TM_Pos10	DINT			☐	☐	No publica
TM_Pos11	DINT			☐	☐	No publica
TM_Pos12	DINT			☐	☐	No publica
TM_Pos13	DINT			☐	☐	No publica

Ilustración 91. Variables globales del PLC en Sysmac Studio.

5.3.2.4.1. Programa 0

El programa 0 es la parte principal de programación del PLC, y está realizado en texto estructurado (ST). El programa realizado se encuentra en el anexo1.

Las variables internas que posee este programa son las siguientes.

Variables

Namespace - Uso de

Internas | Externas

Nombre	Tipo de datos	Valor inicial
ModbusReadTCP_ReadCmd	_sMODBUS_READ	
ModbusTCPWrite_WriteCmd	_sMODBUS_WRITE	
ModbusTCPWrite_Instance	ModbusTCPWrite	
ModbusTCPRead_Instance	ModbusTCPRead	
Order	DINT	0
SentOrderConfirmation_DINT	DINT	
SentOrderConfirmation_BYTE	ARRAY[0..3] OF BYTE	
TM_Posiciones_BYTE	ARRAY[0..55] OF BYTE	
AuxWordArray	ARRAY[0..31] OF WORD	
AuxByteArray	ARRAY[0..3] OF BYTE	
Invertida_Enviar_Orden_Receta_Paletizado	BOOL	False
I_OrderP1	DINT	
I_OrderP2	DINT	
I_OrderP3	DINT	
I_OrderP4	DINT	
I_OrderP5	DINT	
I_OrderP6	DINT	
I_OrderP7	DINT	
I_OrderP8	DINT	
I_OrderP9	DINT	
I_OrderP10	DINT	
I_OrderP11	DINT	
I_OrderP12	DINT	
I_OrderP13	DINT	
I_TipoPallet	DINT	
I_OrderConfirmation	DINT	
SentOrderConfirmation_WORD	ARRAY[0..28] OF WORD	

Ilustración 92. Variables internas del Programa 0 en Sysmac Studio.

Las variables externas que utiliza este programa son las siguientes.

Internas
Externas

Nombre	Tipo de datos
Connected	BOOL
Socket	_sSOCKET
Order_HMI	DINT
Enable	BOOL
OrderConfirmation	DINT
Color_Pieza_HMI	DINT
Forma_Pieza_HMI_DINT	DINT
Pos_HMI	DINT
Enviar_Orden_Receta_Paletizado	BOOL
Reset_Pos	BOOL
NStep	DINT
Modificando_Orden	DINT
OrderP1	DINT
OrderP2	DINT
OrderP3	DINT
OrderP4	DINT
OrderP5	DINT
OrderP6	DINT
OrderP7	DINT
OrderP8	DINT
OrderP9	DINT
OrderP10	DINT
OrderP11	DINT
OrderP12	DINT
OrderP13	DINT
Error_Com	BOOL
Fin_Paletizado	BOOL
TipoPallet	DINT
Finalizacion_Palletizado	BOOL
OrderConfirmationFinish	DINT
BotonDespaletizado	BOOL
TM_Pos1	DINT
TM_Pos2	DINT
TM_Pos3	DINT
TM_Pos4	DINT
TM_Pos5	DINT
TM_Pos6	DINT
TM_Pos7	DINT
TM_Pos8	DINT
TM_Pos9	DINT
TM_Pos10	DINT
TM_Pos11	DINT
TM_Pos12	DINT
TM_Pos13	DINT

Ilustración 93. Variables externas que usa el Programa 0 en Sysmac Studio.

Este código posee tres IF generales, los dos últimos solo comprueban que la variable global "NStep" no esté en 999, ya que si esta se encuentra en ese valor, quiere decir que se ha producido algún error en el proceso.

El primer IF es el código general, y a este se entra cuando las variables "Connected" y "Enable" están en true. La variable "Connected" se modifica desde el Programa1 y se explica en el siguiente apartado. La variable "Enable" se modifica desde el propio HMI, y sirve únicamente para comenzar el ciclo en el momento deseado, ya que si no existiera esta variable, en el momento en el que el Cobot estuviera conectado, el PLC comenzaría el "Switch Case". Este "Switch Case" es creado mediante la variable global "NStep", la cual es de tipo DINT, ya que es la manera más óptima de realizarlo.

Antes de entrar al "Switch Case", se realiza la modificación de cada posición del pallet, seleccionada en el HMI. Es decir, cuando enviamos una orden desde el HMI modificamos el valor de la variable "Enviar_Orden_Receta_Paletizado", la cual se pone a true, dejando escribir la orden seleccionada.

Otra parte del código, es en la que se realiza el reseteo de las variables de las posiciones, ya que en el HMI hay una opción de crear una receta desde cero. Esto se consigue poniendo a true la variable "Reset_Pos".

Una vez que vamos a visualizar la selección desde la pantalla, inscribimos las variables globales sobre unas variables internas, para evitar las posibles modificaciones mientras se realiza el proceso, a la vez que modificamos el valor de las órdenes de confirmación que debe de realizar el Cobot y el PLC, para saber en qué estado se encuentra el proceso. Estas variables son "OrderConfirmation", que sirve para comprobar que se han recibido los datos del pallet y que se ha comenzado a realizar el paletizado o despaletizado sin fallos, y "OrderConfirmationFinish" que sirve para confirmar la finalización del paletizado o despaletizado, y para comenzar a comprobar que piezas han podido ser paletizadas y cuales no han sido encontradas.

Por último se encuentra el IF del despaletizado, el cual se activa desde el HMI, usando la variable global "BotonDespaletizado", la cual se pone a true para entrar, y el tipo de pallet se transforma a 1 para mandar la orden al Cobot, ya que para el despaletizado el tipo de pallet es irrelevante.

Todas las condiciones anteriores van controladas por la variable global "Modificando_Orden", la cual sirve para que en cada página del HMI puedas modificar solo unos parámetros concretos. Cuando su valor es uno, quiere decir que modificas los parámetros en las variables globales, cuando su valor es 2 quiere decir que modificas las variables internas, y si su valor es 3, modificas solo el tipo del pallet.

Ahora vamos explicar el proceso que sigue el "Switch Case", dependiendo siempre de la variable "NStep". Comenzamos cuando la variable "NStep" tiene valor 0, en este momento se escribe en las posiciones de memoria del Cobot usando la variable "ModbusTCPWrite_WriteCmd" que es de tipo "_sMODBUS_WRITE", y usando el function block "ModbusTCPWrite", el cual sirve para escribir en la memoria de la serie TM. Este Function Block posee las siguientes características generales.

The ModbusTCPWrite instruction sends write commands using Modbus-TCP protocol.

Instruction	Name	FB/FUN	Graphic expression	ST expression
ModbusTCPWrite	Send Modbus TCP Write Command	FB	ModbusTCPWrite_instance ModbusTCPWrite Execute, Socket, UnitIdentifier, WriteCmd, WriteDat, TimeOut Done, Busy, Error, ErrorID, ErrorIDEx	ModbusTCPWrite_instance(Execute, Socket, UnitIdentifier, WriteCmd, WriteDat, TimeOut, Done, Busy, Error, ErrorID, ErrorIDEx);

An NX102 CPU Unit is required to use this instruction.

Variables

	Meaning	I/O	Description	Valid range	Unit	Default
Socket	Socket	Input	Socket	---	---	---
UnitIdentifier	Unit ID		Unit ID *1	Depends on data type.	---	255
WriteCmd	Write command		Command data	Depends on data type.	---	---
WriteDat	Write data		Write data	Depends on data type.	---	---
TimeOut	Timeout time		Specify the timeout time in 0.1 seconds. If 0 is specified, it will be treated as 2 seconds.	Depends on data type.	0.1 seconds	20

*1 When you send commands to Modbus-TCP slaves, the default value is used for operation.

Ilustración 94. Características de "ModbusTCPWrite" en Sysmac Studio. [41]

En este caso usamos "ModbusTCPWrite_WriteCmd.Fun" para asignar el tipo de datos a enviar, y estos datos son de tipo "_MDB_WRITE_MULTIPLE_REGISTERS", ya que enviamos un Array de Words, porque nuestro Cobot lee sus posiciones de memoria de esta forma.

Seguimos definiendo la posición de memoria en la que empezaremos a grabar los datos enviados mediante la variable "ModbusTCPWrite_WriteCmd.WriteAdr", que en este caso tiene el valor de 9000, ya que nuestro Cobot posee de la posición 9000 a la 9999 de memoria libre. Y por último, seleccionamos el tamaño del Array de Words que vamos a enviar, usando la variable "ModbusTCPWrite_WriteCmd.WriteSize", que tiene un valor de 32. Este número viene dado ya que enviamos 13 dobles enteros correspondientes a las trece posiciones de nuestro pallet, 2 dobles enteros con las órdenes de confirmación y 1 doble entero con el tipo de pallet. Y cada doble entero o DINT, ocupa 2 Words.

El proceso continúa realizando la transformación de los dobles enteros a Bytes, esto se realiza usando la función "ToAryByte", mediante la cual obtenemos un Array de cuatro Bytes. Con este Array de Bytes contenido en la variable "AuxByteArray", conseguimos un Array de dos Words, contenido en la variable "AuxWordArray", usando la función "BYTE_TO_WORD", y organizando la posición de cada Word obtenido, mediante la función "SHL". Con todo esto obtenemos un Array de dos Words, teniendo previamete un DINT. Y este proceso lo repetimos para cada doble entero que queremos enviar, es decir para los 16 que tenemos, grabando cada valor en una posición superior de la variable "AuxWordArray".

Una vez realizado todo este proceso pasamos a realizar un pulso en el function block "ModbusTCPWrite_Instance", poniendo la entrada "Execute" a false y después a true, provocando la escriturada de los datos en las posiciones de memoria seleccionadas. Y modificando al final de este proceso a NStep igual a 10.

Una vez que NStep es igual a 10 se comprueba si la variable "ModbusTCPWrite_Instance.Done" es igual a true, para confirmar que el proceso se ha realizado correctamente, y el "NStep" pasa a valer 20. Si esta variable no es true se comprueba que la variable "ModbusTCPWrite_Instance.Error" sea igual a true, si es así el PLC pasa a modo error y "NStep" pasa a valer 999. Si ninguna de las dos condiciones anteriores se cumplen se queda dentro de este bucle.

Seguimos el "Switch Case" cuando el valor de "NStep" es igual a 20. En este momento se realiza la lectura de la posición 9600 de la memoria del Cobot usando la function block llamado "ModbusTCPRead", el cual posee las siguientes características.

The ModbusTCPRead instruction reads data that is requested by sending read commands using Modbus-TCP protocol.

Instruction	Name	FB/FUN	Graphic expression	ST expression
ModbusTCPRead	Send Modbus TCP Read Command	FB	ModbusTCPRead_instance ModbusTCPRead Execute, Socket, UnitIdentifier, ReadCmd, ReadDat, TimeOut Done, Busy, Error, ErrorID, ErrorIDEx, ReadSize	ModbusTCPRead_instance(Execute, Socket, UnitIdentifier, ReadCmd, ReadDat, TimeOut, Done, Busy, Error, ErrorID, ErrorIDEx, ReadSize);

An NX102 CPU Unit is required to use this instruction.

Variables

	Meaning	I/O	Description	Valid range	Unit	Default
Socket	Socket	Input	Socket	---	---	---
UnitIdentifier	Unit ID		Unit ID *1	Depends on data type.	---	255
ReadCmd	Read command		Command data	Depends on data type.	---	---
TimeOut	Timeout time		Specify the timeout time in 0.1 seconds. If 0 is specified, it will be treated as 2 seconds.	Depends on data type.	0.1 seconds	20
ReadDat	Read data	In-out	Read data	Depends on data type.	---	---
ReadSize	Read size	Output	Amount of read data	1 to 2000 *2 1 to 125 *3	Bits *2 Words *3	---

*1 When you send commands to Modbus-TCP slaves, the default value is used for operation.

Ilustración 95. Características de " ModbusTCPRead" en Sysmac Studio. [41]

Para su uso creamos una variable llamada "ModbusReadTCP_ReadCmd" la cual es del tipo "_sMODBUS_READ". Y como en el apartado anterior vamos definiendo las características que debe de poseer esta lectura de memoria.

Comenzamos usando la variable "ModbusReadTCP_ReadCmd.Fun", para definir el tipo de datos que esperamos recibir, que en este caso es del tipo "_MDB_READ_HOLDING_REGISTERS", ya que queremos recibir un Array de dos Words, para obtener de este un doble entero. Seguimos seleccionando la posición de memoria que queremos leer, en este caso la posición 9600, mediante la variable "ModbusReadTCP_ReadCmd.ReadAdr", y también el tamaño del Array que esperamos, en este caso de 2, para ello usamos la variable "ModbusReadTCP_ReadCmd.ReadSize". Y por último, realizamos un pulso, poniendo de false a true, la entrada "Execute" e indicamos la variable en la que queremos guardar el Array de dos Words recibido, que en este caso es la variable "SentOrderConfirmation_WORD". Terminamos dando el valor de 30 a la variable "NStep".

Para "NStep" igual a 30 se sigue el proceso de lectura de la posición 9600, ya que esta condición posee otras dos condiciones dentro de ella, la primera es que si la variable de que la lectura se ha producido, es decir si la variable "ModbusTCPRead_Instance.Done" es igual a true, se produce la transformación del Array de dos Words, de la variable "SentOrderConfirmation_WORD", mediante la función "WORD_TO_BYTE" a un Array de cuatro Bytes, guardados en la variable "SentOrderConfirmation_BYTE", y este se transforma a un DINT mediante la función "AryByteTo" y se guarda en la variable "SentOrderConfirmation_DINT". Después de esto se realiza una comprobación de que el valor de "SentOrderConfirmation_DINT" es dos. Si es así "NStep" pasa a ser 40, y si no es así "NStep" pasa a valer 0, y comienza de nuevo el ciclo. Por otro lado si "ModbusTCPRead_Instance.Done" no es igual a true, se comprueba el valor de la variable "ModbusTCPRead_Instance.Error", ya que si es true quiere decir que se ha producido un error en la comunicación provocando que el "NStep" pasase a valer 999.

Seguimos explicando el proceso para cuando "NStep" vale 40, en este momento el proceso seleccionado, ya sea paletizado o despaletizado, se está produciendo. Esto significa que debemos de esperar hasta que este proceso termine para poder realizar otro, y lo conseguimos mandando una orden desde el Cobot de que el proceso ha finalizado, y de si se ha podido completar sin fallos, o en su defecto de que posiciones no han podido ser paletizadas correctamente.

Para ello realizamos una lectura desde la posición 9602 hasta la 9629, ya que esperamos recibir un Array de 28 Words, para ello realizamos el proceso de lectura explicado previamente. Este comienza mediante un pulso en la entrada "Execute", e indicamos la variable en la que queremos guardar los datos obtenidos, que en este caso es "SentOrderConfirmation_WORD". Una vez terminado este proceso cambiamos el valor de "NStep" a 70.

Cuando "NStep" vale 70, comenzamos la comprobación de que se ha leído correctamente las posiciones de memoria deseadas. Hay dos condiciones generales, si la variable "ModbusTCPRead_Instance.Done" es true, se transforma el primer Word que hay en el Array de Words recibido en la variable "SentOrderConfirmation_WORD", primero se transforma a Bytes y después a DINT, si este DINT vale 5, se realiza la transformación a DINT del resto de posiciones del Array del Words, para saber las partes que han sido paletizadas correctamente y las que no. Estos valores se guardan en las variables globales "TM_PosXX", si el valor almacenado en esta variables es diferente a cero quiere decir que no se ha realizado el paletizado correctamente, por lo que no se mostrara por pantalla en el pallet final, intentando aproximar lo visualizado en el HMI lo máximo posible a la realidad, y la variable "NStep" pasara a valer 80.

Y si el valor del primer Word de la variable "SentOrderConfirmation_WORD", al ser transformado a DINT, no vale 5, la variable "NStep" pasa a valer 40 comenzando así el ciclo de lectura de la orden de finalización.

Si el valor de "ModbusTCPRead_Instance.Done" es igual a false, se comprueba el valor de "ModbusTCPRead_Instance.Error", ya que si este es igual a true el PLC entra en modo fallo, cambiando el valor de "NStep" a 999.

Seguimos la explicación del "Switch Case" para cuando "NStep" tiene el valor 80. En este momento comenzamos la limpieza de los datos de confirmación recibidos, ya que realizamos la limpieza desde la posición 9600 a la 9603, es decir ponemos un cero en estas posiciones. A la vez realizamos la limpieza mediante la variable "Clear" de las órdenes enviadas y de las posiciones completadas por el Cobot, es decir, ponemos a cero las variables "AuxWordArray" y "TM_Posiciones_BYTE".

Para la limpieza de las posiciones de memoria del Cobot debemos de escribir en ellas mediante el function block "ModbusTCPWrite", de la misma manera que hemos realizado anteriormente. Una vez completados estos procesos damos el valor de 90 a la variable "NStep".

Y el último caso del "Switch Case" que se da cuando "NStep" vale 90, terminamos de realizar la escritura de las posiciones mencionadas anteriormente y comprobamos mediante la variable "ModbusTCPWrite_Instance.Done" que se haya realizado. Cuando esta variable se pone a true se ponen a cero las variables "OrderConfirmationFinish" y "OrderConfirmation", y se completa el ciclo poniendo también a cero la variable "NStep".

Si "ModbusTCPWrite_Instance.Done" no es igual a true, se comprueba el valor de la variable "ModbusTCPWrite_Instance.Error", ya que si esta es true quiere decir que se ha producido un fallo en la comunicación y el PLC entra en modo error poniendo el NStep a 999.

5.3.2.4.2. Programa 1

Este programa usa como lenguaje el diagrama de relés, para simplificar la conexión con el Cobot de la serie TM, ya que esta serie posee un function block para ello. Este programa solo tiene una sección, "Sección0", y está compuesta únicamente por un botón, el cual es una variable global llamada "Connect", utilizada únicamente para controlar cuando deseamos establecer comunicación con el Cobot.

Por otro lado, tenemos el function block llamado "SktTCPConnect", el cual sirve para realizar una conexión TCP mediante el puerto Ethernet/IP con el Cobot, y pose las siguientes características.

The SktTCPConnect instruction connects to a remote TCP port from the built-in EtherNet/IP.

Instruction	Name	FB/ FUN	Graphic expression	ST expression
SktTCP Connect	Connect TCP Socket	FB	SktTCPConnect_instance SktTCPConnect Execute — Done SrcTcpPort — Busy DstAdr — Error DstTcpPort — ErrorID Socket	SktTCPConnect_instance(Execute, SrcTcpPort, DstAdr, DstTcpPort, Done, Busy, Error, ErrorID, Socket);

Variables

Name	Meaning	I/O	Description	Valid range	Unit	Default
SrcTcpPort	Local TCP port number	Input	Local TCP port number. If 0 is specified, an available TCP port that is 1024 or higher is automatically assigned. Well-known port numbers are not assigned.	Depends on data type.	---	0
DstAdr	Destination address		Destination IP address or host name	200 bytes max.		---
DstTcpPort	Destination TCP port number		Destination TCP port number	1 to 65,535		1
Socket	Socket	Output	Socket	---	---	---

	Boolean	Bit strings				Integers								Real numbers		Times, durations, dates, and text strings				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
SrcTcpPort							OK													
DstAdr																				OK
DstTcpPort							OK													
Socket	Refer to *Function* for details on the structure _sSOCKET.																			

Ilustración 96. Características de SktTCPConnect en Sysmac Studio.[41]

Para este function block debemos darle valor a todas sus variables. Comenzamos dando un pulso en la entrada "Execute", controlado por la variable "Connect". Seguimos dándole un valor al puerto de entrada TCP, en la entrada "SrcTcpPort", el cual posee el valor 0. Para la variable "DstAdr", le damos el valor de la dirección IP que pose nuestro Cobot, '192.168.250.35'. Y para la última variable de entrada que sirve para dar el puerto de destino TCP, "DstTcpPort", le damos el valor de 502.

Por último asignamos variables a las salidas del function block. Comenzamos por la salida "Done", a la cual le hemos asignado la variable global "Connected", que sirve para saber si el Cobot ha sido conectado correctamente, esta variable la usaremos en el programa 1 para iniciar el ciclo. Seguimos asignando la variable "Busy", a la cual le asignamos una variable interna "SktTCPConnect_Instrance_Busy", que sirve para saber si la comunicación no se puede realizar por que la dirección de destino esta ocupada. Continuamos asiganando valores a las salidas de error, para la salida "Error" se le asigna la variable "SktTCPConnect_Instrance_Error", y para la salida "ErrorID", se le asigna la variable "SktTCPConnect_Instrance_ErrorID", la primera es una boolean que solo da si el error es true o false, y la segunda da un número que es asignado al error producido, este valor lo puedes buscar en la documentación del function block, para saber exactamente el error que se ha producido. Y por último, tenemos la variable global "Socket", la cual es asignada a la salida "Socket", y es una variable compleja de tipo "_sSOCKET" y posee las siguientes características.

Name	Meaning	Description	Data type	Valid range	Unit	Default
Socket	Socket	Socket	_sSOCKET	---	---	---
Handle	Handle	Handle for data communications	UDINT	Depends on data type.	---	0
SrcAdr	Local address	Local IP address and port number	_sSOCKET_ADDRESS	---	---	---
PortNo	Port number	Port number	UINT	1 to 65535	---	0
IpAdr*1	IP address	IP address or host name. A DNS or Hosts setting is required to use a host name.	STRING	Depends on data type.		''
DstAdr	Destination address	Destination IP address and port number	_sSOCKET_ADDRESS	---	---	---
PortNo	Port number	Port number	UINT	1 to 65535	---	0
IpAdr	IP address	IP address or host name. A DNS or Hosts setting is required to use a host name.	STRING	Depends on data type.		''

*1 NULL is output for this member.

Ilustración 97. Características de la variable "Socket" en Sysmac Studio. [41]

Con todo esto el programa realizado queda tal y como se muestra en la siguiente ilustración.

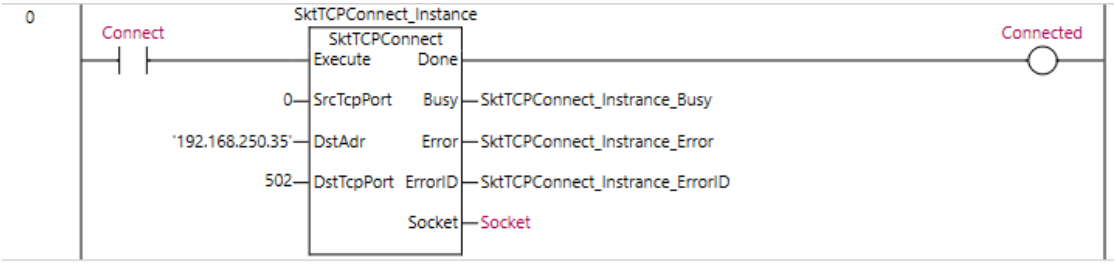


Ilustración 98. Programa 1 en Sysmac Studio.

Y las variables internas y externas en este programa usadas son las siguientes.

Variables

Namespace - Uso de

Internas | Externas

Nombre	Tipo de datos	Valor inicial
SktTCPConnect_Instance	SktTCPConnect	
SktTCPConnect_Instrance_Error	BOOL	
SktTCPConnect_Instrance_Busy	BOOL	
SktTCPConnect_Instrance_ErrorID	WORD	

Ilustración 99. Variables internas del programa 1 en Sysmac Studio.

Variables

Namespace - Uso de

Internas | Externas

Nombre	Tipo de datos	Constante
Socket	_sSOCKET	☐
Connected	BOOL	☐
Connect	BOOL	☐

Ilustración 100. Variables externas del programa 1 en Sysmac Studio.

5.3.3. Programación del HMI en Sysmac Studio

Comenzamos la programación del HMI añadiéndola al mismo proyecto en el que hemos realizado la programación del PLC, llamado NX102MQTT. Para ello realizamos clic izquierdo sobre el símbolo del PLC y en el desplegable seleccionamos "Añadir dispositivo", tal y como se ve en la siguiente ilustración.

Ilustración 101. Añadir dispositivo en Sysmac Studio.

Una vez seleccionado esa opción nos sale la siguiente ventana emergente, en la cual debemos de seleccionar las características del HMI que vamos a usar, en mi caso son las siguientes.

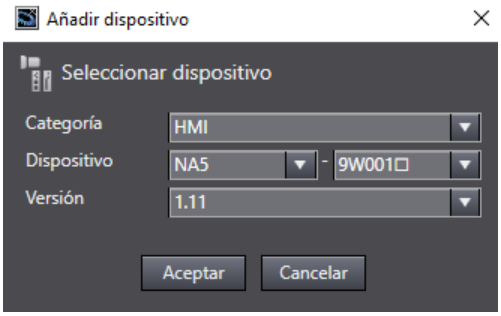


Ilustración 102. Características del HMI añadido a Sysmac Studio.

Seguimos modificando la dirección IP del HMI desde "Ajustes de HMI", en "Ajustes de TCP/IP", y seleccionamos el puerto Ethernet, que en nuestro caso es el 1, y le damos la dirección IP 192.168.250.010, y la máscara subred dejamos que se autocomplete, tal y como podemos observar en la siguiente imagen.

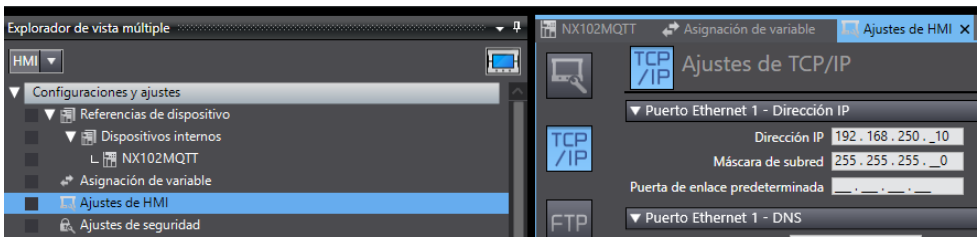


Ilustración 103. Asignación de la dirección IP para el HMI en Sysmac Studio.

5.3.3.1. Creación de variables globales

Las variables globales en el HMI pueden ser de dos tipos, ya que pueden ser propias del HMI o pueden estar asignadas a un elemento externo. Para crear una variable se debe de hacer clic izquierdo sobre cualquier lugar en la pestaña de "Variables globales". Se le debe de asignar un nombre y el tipo de dato que es, y optativamente se puede añadir un valor inicial y en la columna "AT" se le puede asignar a una variable global de un controlador dentro del mismo proyecto, para ello debes de poner el nombre del controlador, en este caso "NX102MQTT", poner un punto detrás suyo y añadirle el nombre de la variable global de ese controlador que quieras asignar a la variable global del HMI. Las variables globales del HMI que tenemos son las siguientes.

Nombre	Tipo de datos	Valo	AT	Rete	Cons	Actualizar velocidad
Order_HMI	Integer			☐	☐	Ninguno
NX102MQTT_Connected	Boolean		NX102MQTT.Connected	☐	☐	500 milisegundos
NX102MQTT_Order_HMI	Integer		NX102MQTT.Order_HMI	☐	☐	500 milisegundos
NX102MQTT_Socket	NX102MQTT_sSO...		NX102MQTT.Socket	☐	☐	500 milisegundos
NX102MQTT_Connect	Boolean		NX102MQTT.Connect	☐	☐	500 milisegundos
Color_Pieza_Pick	Integer		NX102MQTT.Color_Pieza_HMI	☐	☐	Ninguno
NX102MQTT_Enable	Boolean		NX102MQTT.Enable	☐	☐	500 milisegundos
Place_Pieza	Integer		NX102MQTT.Pos_HMI	☐	☐	Ninguno
Forma_Pieza	Integer		NX102MQTT.Forma_Pieza_HMI_DINT	☐	☐	Ninguno
Establecer_Receta_Paletizado	Boolean		NX102MQTT.Enviar_Orden_Receta_Paletizado	☐	☐	Ninguno
Reset_Pos	Boolean		NX102MQTT.Reset_Pos	☐	☐	Ninguno
NX102MQTT_Modificar_Orden	Integer		NX102MQTT.Modificando_Orden	☐	☐	Ninguno
NX102MQTT_OrderP1	Integer		NX102MQTT.OrderP1	☐	☐	Ninguno
NX102MQTT_OrderP2	Integer		NX102MQTT.OrderP2	☐	☐	Ninguno
NX102MQTT_OrderP3	Integer		NX102MQTT.OrderP3	☐	☐	Ninguno
NX102MQTT_OrderP4	Integer		NX102MQTT.OrderP4	☐	☐	Ninguno
NX102MQTT_OrderP5	Integer		NX102MQTT.OrderP5	☐	☐	Ninguno
NX102MQTT_OrderP6	Integer		NX102MQTT.OrderP6	☐	☐	Ninguno
NX102MQTT_OrderP7	Integer		NX102MQTT.OrderP7	☐	☐	Ninguno
NX102MQTT_OrderP8	Integer		NX102MQTT.OrderP8	☐	☐	Ninguno
NX102MQTT_OrderP9	Integer		NX102MQTT.OrderP9	☐	☐	Ninguno
NX102MQTT_OrderP10	Integer		NX102MQTT.OrderP10	☐	☐	Ninguno
NX102MQTT_OrderP11	Integer		NX102MQTT.OrderP11	☐	☐	Ninguno
NX102MQTT_OrderP12	Integer		NX102MQTT.OrderP12	☐	☐	Ninguno
NX102MQTT_OrderP13	Integer		NX102MQTT.OrderP13	☐	☐	Ninguno
NX102MQTT_NStep	Integer		NX102MQTT.NStep	☐	☐	Ninguno
NX102MQTT_Error_Com	Boolean		NX102MQTT.Error_Com	☐	☐	Ninguno
NX102MQTT_Fin_Paletizado	Boolean		NX102MQTT.Fin_Paletizado	☐	☐	Ninguno
NX102MQTT_TipoPallet	Integer		NX102MQTT.TipoPallet	☐	☐	Ninguno
NX102MQTT_BotonDespaletizado	Boolean		NX102MQTT.BotonDespaletizado	☐	☐	Ninguno
Temp_100ms	Boolean			☐	☐	Ninguno
VisualizarPalletInt	Integer			☐	☐	Ninguno
NX102MQTT_TM_Pos1	Integer		NX102MQTT.TM_Pos1	☐	☐	Ninguno
NX102MQTT_TM_Pos2	Integer		NX102MQTT.TM_Pos2	☐	☐	Ninguno
NX102MQTT_TM_Pos3	Integer		NX102MQTT.TM_Pos3	☐	☐	Ninguno
NX102MQTT_TM_Pos4	Integer		NX102MQTT.TM_Pos4	☐	☐	Ninguno
NX102MQTT_TM_Pos5	Integer		NX102MQTT.TM_Pos5	☐	☐	Ninguno
NX102MQTT_TM_Pos6	Integer		NX102MQTT.TM_Pos6	☐	☐	Ninguno
NX102MQTT_TM_Pos7	Integer		NX102MQTT.TM_Pos7	☐	☐	Ninguno
NX102MQTT_TM_Pos8	Integer		NX102MQTT.TM_Pos8	☐	☐	Ninguno
NX102MQTT_TM_Pos9	Integer		NX102MQTT.TM_Pos9	☐	☐	Ninguno
NX102MQTT_TM_Pos10	Integer		NX102MQTT.TM_Pos10	☐	☐	Ninguno
NX102MQTT_TM_Pos11	Integer		NX102MQTT.TM_Pos11	☐	☐	Ninguno
NX102MQTT_TM_Pos12	Integer		NX102MQTT.TM_Pos12	☐	☐	Ninguno
NX102MQTT_TM_Pos13	Integer		NX102MQTT.TM_Pos13	☐	☐	Ninguno

Ilustración 104. Variables globales del HMI en Sysmac Studio.

5.3.3.2. Eventos globales

Los eventos globales son eventos que ocurren en todas las páginas del HMI, y se crean dentro de la ventana "Eventos globales", en la cual debes de seleccionar el tipo de evento. En mi caso solo uso un evento global, que sirve para actualizar variables cada 200 milisegundos. Para ello he creado un evento de tipo "Interval", le he dado el valor de 100 ms al Intervalo y he seleccionado que se realice la llamada a una subrutina mediante la acción "CallSubrutine", y la he llamado "SubroutineGroup0.Temporizador". Así que cada vez que quiera usarla solo tendré que poner esta denominación.

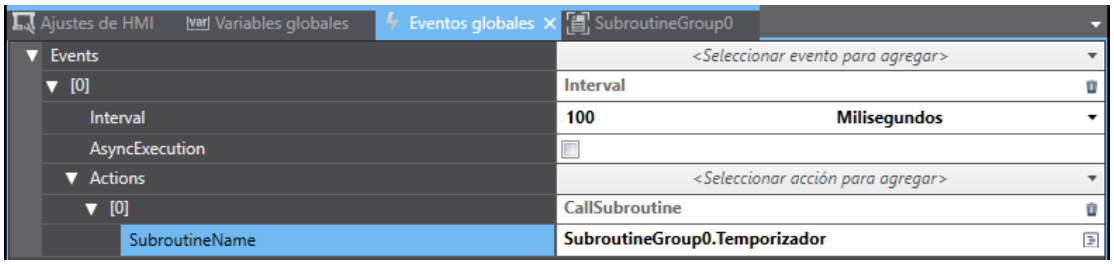


Ilustración 105. Creación de un evento global en Sysmac Studio.

La subrutina está compuesta por un código en ST, al que se accede desde el símbolo situado a la derecha del nombre de la subrutina, es decir, a la derecha del nombre "SubroutineGroup0.Temporizador", y el código creado es el siguiente.

```
Sub Temporizador
    If Temp_100ms =False Then
        Temp_100ms=True

    Else
        Temp_100ms=False
    End if

End Sub
```

Con este código lo que hacemos es que cada 100ms cambie el estado de la variable, pudiendo con esto actualizar las variables obtenidas por el PLC, consiguiendo saber cuándo se ha terminado el paletizado o el despaletizado entre otras cosas.

5.3.3.3. Alarmas de usuario

La creación de alarmas de usuario en nuestro proyecto solo es usada cuando existe un fallo en la comunicación por Ethernet con el Cobot, es decir cuando el valor de la variable global del controlador, "NStep" tiene el valor de 999. Para crear estas alarmas vamos a la pestaña "Alarmas de usuario" y realizamos clic izquierdo sobre ellas, le damos a "Añadir grupo". Tal y como podemos observar en la siguiente imagen.

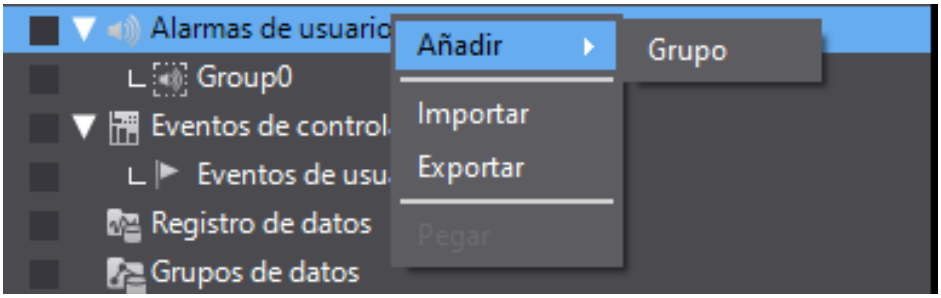
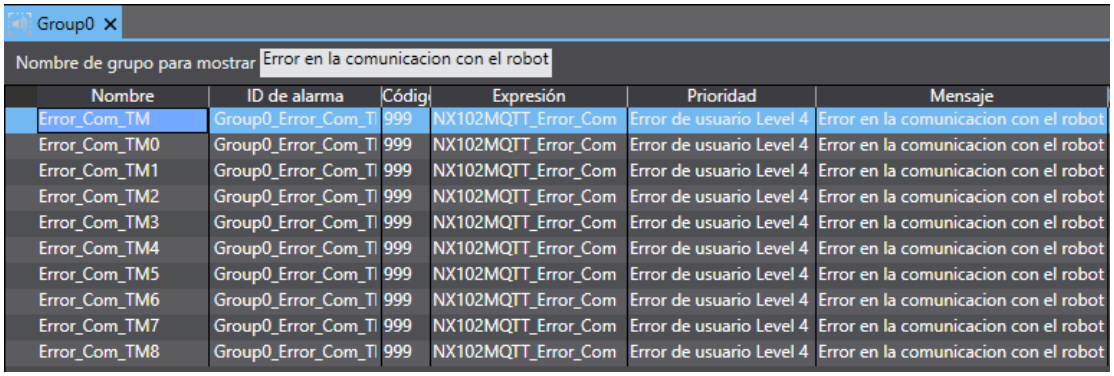


Ilustración 106. Añadir grupo de alarmas de usuario en Sysmac Studio.

Una vez añadido creamos nuestras alarmas de usuario en el "Group0", que en el caso de este proyecto serán las siguientes.


Group0

Nombre de grupo para mostrar: Error en la comunicacion con el robot

Nombre	ID de alarma	Códig	Expresión	Prioridad	Mensaje
Error_Com_TM	Group0_Error_Com_TI	999	NX102MQTT_Error_Com	Error de usuario Level 4	Error en la comunicacion con el robot
Error_Com_TM0	Group0_Error_Com_TI	999	NX102MQTT_Error_Com	Error de usuario Level 4	Error en la comunicacion con el robot
Error_Com_TM1	Group0_Error_Com_TI	999	NX102MQTT_Error_Com	Error de usuario Level 4	Error en la comunicacion con el robot
Error_Com_TM2	Group0_Error_Com_TI	999	NX102MQTT_Error_Com	Error de usuario Level 4	Error en la comunicacion con el robot
Error_Com_TM3	Group0_Error_Com_TI	999	NX102MQTT_Error_Com	Error de usuario Level 4	Error en la comunicacion con el robot
Error_Com_TM4	Group0_Error_Com_TI	999	NX102MQTT_Error_Com	Error de usuario Level 4	Error en la comunicacion con el robot
Error_Com_TM5	Group0_Error_Com_TI	999	NX102MQTT_Error_Com	Error de usuario Level 4	Error en la comunicacion con el robot
Error_Com_TM6	Group0_Error_Com_TI	999	NX102MQTT_Error_Com	Error de usuario Level 4	Error en la comunicacion con el robot
Error_Com_TM7	Group0_Error_Com_TI	999	NX102MQTT_Error_Com	Error de usuario Level 4	Error en la comunicacion con el robot
Error_Com_TM8	Group0_Error_Com_TI	999	NX102MQTT_Error_Com	Error de usuario Level 4	Error en la comunicacion con el robot

Ilustración 107. Group0 de alarmas de usuario en Sysmac Studio.

5.3.3.4. Páginas

En esta parte es donde se encuentra la programación real del HMI, en ella se realizan eventos, animaciones y acciones de todos los componentes necesarios, para mostrar por pantalla de una forma fácil e intuitiva los procesos que se están realizando y los que se pueden realizar.

Para crear una página o grupo de páginas debemos de realizar clic izquierdo sobre sobre el desplegable "Páginas", y seleccionar lo que queremos crear.

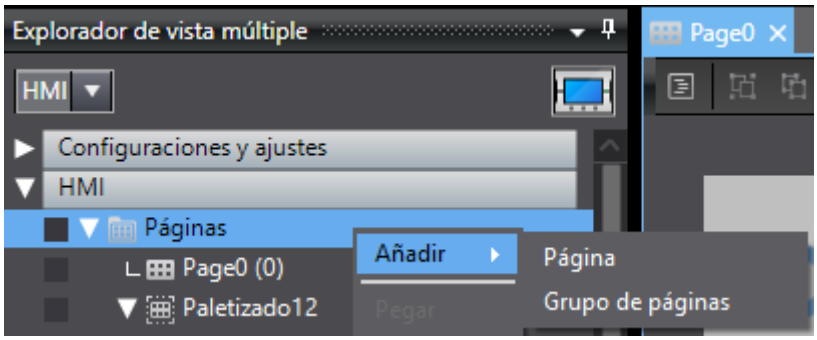


Ilustración 108. Creación de grupo de páginas o páginas en Sysmac Studio.

Si deseamos crear una página dentro de un grupo de páginas debemos de realizar clic izquierdo sobre este grupo y en el desplegable seleccionar "Añadir" y después "Página".

Una vez creada la página debemos de observar las propiedades de esta, realizando doble clic derecho sobre ella, esto genera un desplegable de "propiedades", desde el cual podemos modificar las características generales de la página, como su nombre o su plantilla., tal y como se ve en la siguiente imagen.

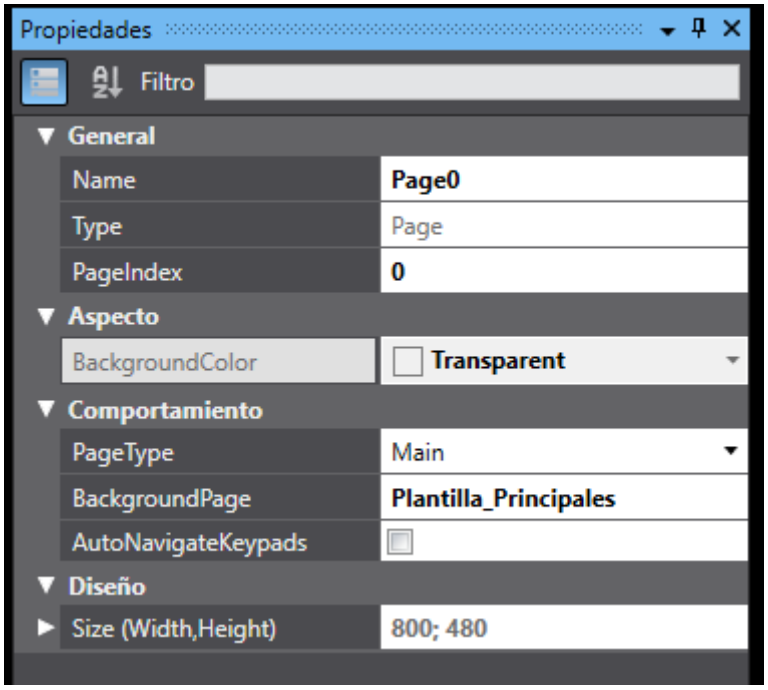


Ilustración 109. Modificar las propiedades de una página en Sysmac Studio.

También podemos hacer que las páginas tengan animaciones, eventos y acciones, para ello debemos de ir a los botones situados en la barra de tareas superior, con los que obtendremos sus correspondientes desplegables, tal y como se ve en las siguientes imágenes.

Ilustración 110. Botones de acceso directo a los eventos y acciones, y a las animaciones en Sysmac Studio.

Para crear un evento, una acción o una animación hace falta un flanco de activación, y esos flancos deben ser los que se observan en las siguientes imágenes.

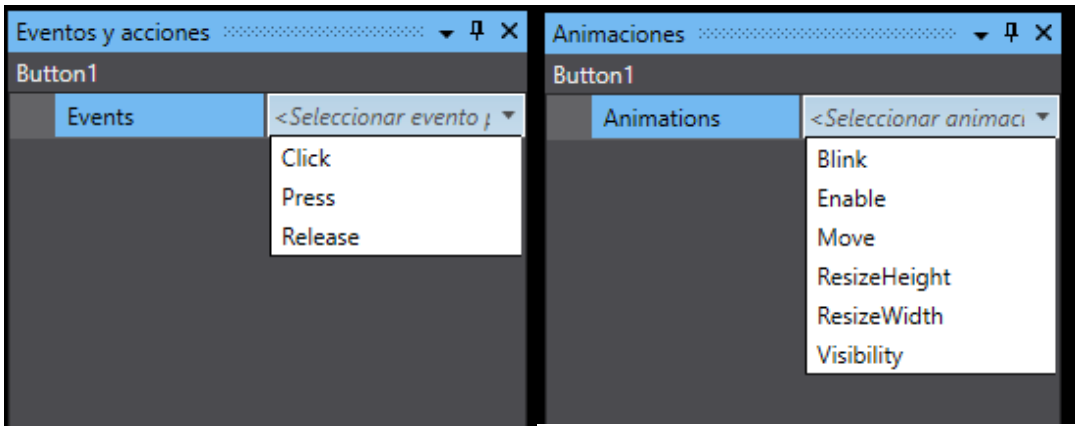


Ilustración 111. Ventanas emergentes para crear eventos, acciones, y animaciones en Sysmac Studio.

Al igual que las subrutinas globales cada página posee subrutinas, las cuales se programan en ST, y se deben de desarrollar en la "Página de código subyacente", y ha este se accede desde el siguiente botón.

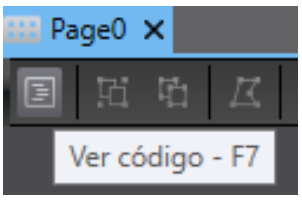


Ilustración 112. Botón de acceso directo a la página de código subyacente en Sysmac Studio.

Una vez completadas las características de la página, podemos añadir todo tipo de complementos desde la "Caja de herramientas". Tal y como se puede observar en la siguiente imagen.

Ilustración 113. Caja de herramientas en Sysmac Studio.

A su vez cada objeto que añadamos posee unas propiedades concretas y puede tener sus propias acciones, eventos y animaciones, haciendo que cada elemento sea independiente, simplificando así la programación.

A continuación vamos a explicar las páginas que hemos usado en el HMI del proyecto y los diferentes objetos que las componen. Las vamos a agrupar por los propios grupos de página a los que pertenecen, ya que cada grupo posee un propósito general.

5.3.3.4.1. Plantillas de paginas

Este grupo de páginas está compuesto por dos plantillas creadas para simplificar la programación del resto de páginas. Para ello he creado una plantilla usada en las páginas principales, con los logotipos de la Universidad Politécnica de la Almunia de Doña Godina y de la empresa Omron en grande, junto con la fecha, la hora, el nombre del creador del proyecto, y una línea negra para separar la parte del título del resto de la página. Esta página no posee eventos, acciones o animaciones, y los objetos añadidos tampoco. El resultado obtenido es la página con el nombre de "Plantilla_Principales"

Ilustración 114. Plantilla principal del HMI en Sysmac Studio.

Para las páginas que necesito mucho espacio he creado una plantilla con solo el logotipo de la universidad, una línea negra para situar el título, y un botón de inicio, el cual lleva a la página inicial del HMI. Esta plantilla tampoco pose eventos, animaciones o acciones. Solo posee el objeto llamado "Button2", el cual tiene el evento y la acción de que cuando se realiza una pulsación nos manda a la "Page0". Para conseguir esto, dentro de la pestaña saliente de los eventos y acciones, realizamos la siguiente selección.

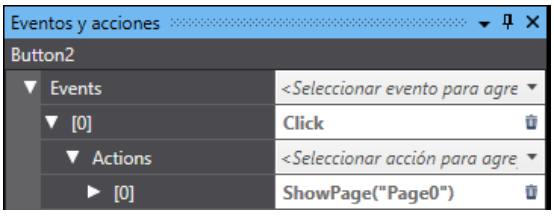


Ilustración 115. Evento y acción de que el Button2 nos traslade a la "Page0" tras una pulsación.

Con todo esto nos queda la siguiente página usada como plantilla y con el nombre de "Plantilla_Secundarias".

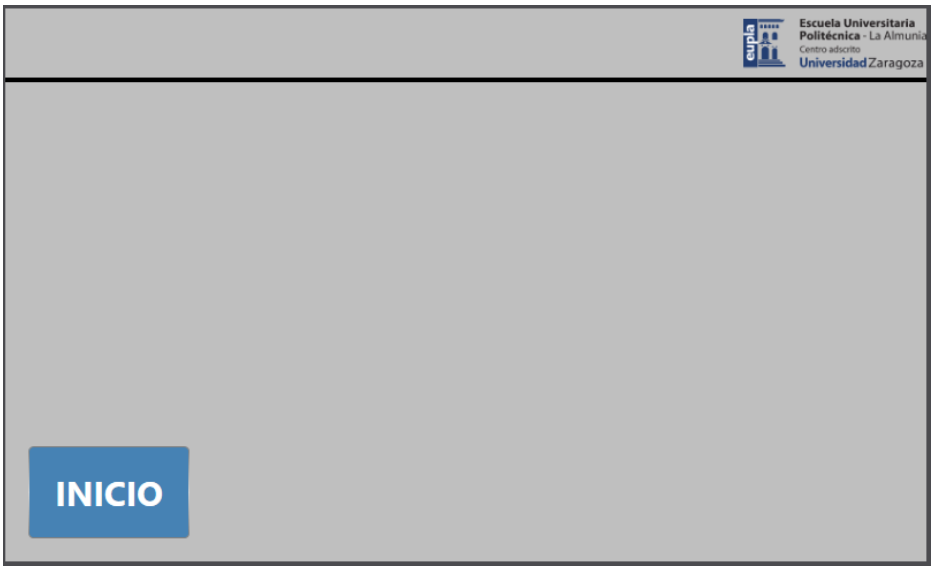


Ilustración 116. Plantilla secundaria del HMI en Sysmac Studio.

5.3.3.4.2. Página de inicio

La página de inicio es la primera que se muestra y se denomina "Page0". Esta usa la plantilla principal, denominada "Plantilla_Principales" y posee tres zonas diferenciadas, tal y como se ve en la siguiente imagen.

Ilustración 117. Página principal del HMI en Sysmac Studio.

.

La primera zona es donde se produce la activación, mediante un botón, de la conexión con el Cobot, y esta conexión es confirmada por el avisador luminoso que posee en su parte inferior.

Para la programación del "Button0", que corresponde al botón "CONECTAR COBOT" se designa su funcionamiento directamente desde las propiedades, siendo que cuando este es presionado pone a true la variable "NX102MQTT_Connect", y cuando se vuelve a presionar la pone false.

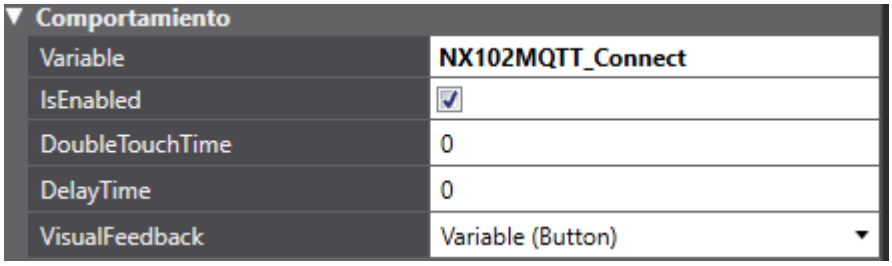


Ilustración 118. Evento provocado por el botón Conectar Cobot en Sysmac Studio.

Por otro lado el objeto "BitLamp0" cambia de color cuando la variable "NX102MQTT_Connected" se encuentra en true, y también se realiza la programación de este evento sobre sus propiedades, tal y como podemos observar en la siguiente imagen.

Ilustración 119. Comportamiento de la "BitLamp0" en Sysmac Studio.

Seguimos la explicación con el segundo bloque de componentes, los cuales realizan una función similar a los explicados anteriormente. Estos inician y pausan el "Switch Case" del PLC, en este caso el "Button1", que corresponde al botón "INICIAR BUCLE PLC" modifica la variable "NX102MQTT_Enable", al ser pulsado. Y el "BitLamp1" cambia de color cuando esta es modificada.

Por ultimo explicamos el funcionamiento de los tres botones de la derecha, los cuales tienen funciones muy similares. Comenzamos con el botón de "Paletizado 12u", el cual es denominado "Button4", y sirve para comenzar a realizar la selección de las posiciones del pallet de doce unidades. Este botón pose tres eventos que se producen cuando realizamos clic sobre él. Los cuales son los siguientes.

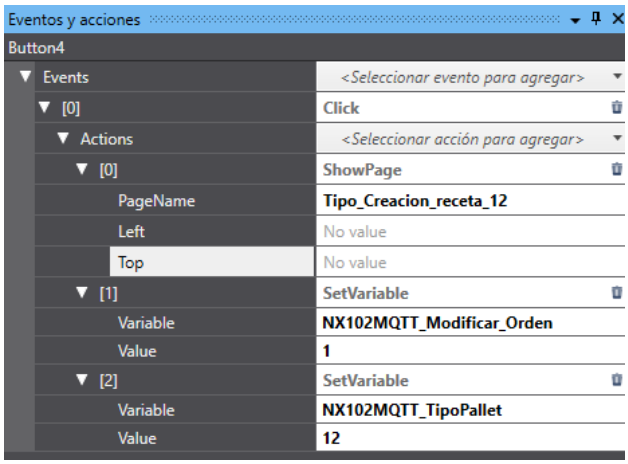


Ilustración 120. Eventos y acciones del botón del paletizado de 12u.

Es decir, cuando realizamos clic nos envía a la página "Tipo_Creacion_receta_12", modifica la variable "NX102MQTT_Modificar_Orden" y la pone a 1, y por ultimo modifica la variable "NX102MQTT_TipoPallet" y la pone a 12.

El segundo botón a explicar realiza eventos y acciones muy parecidas al anterior, pero modificando algunos datos, es decir, cuando realizamos clic sobre el botón "Paletizado 13u" nos envía a la página "Tipo_Creacion_receta_13", modifica la variable "NX102MQTT_Modificar_Orden" y la pone a 1, y por ultimo modifica la variable "NX102MQTT_TipoPallet" y la pone a 13.

Y por último el botón de "Despaletizado", que posee cuatro eventos y acciones cuando realizamos clic sobre él. Este clic provoca que te traslade a la "Page6", que modifique la variable "NX102MQTT_TipoPallet" al valor 1, que modifique la variable "NX102MQTT_Modificar_Orden" al valor 3 y por último que invierta la variable "NX102MQTT_BotonDespaletizado".

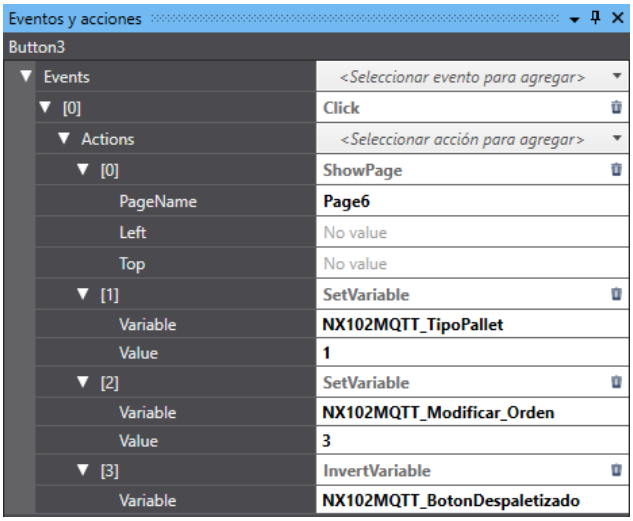


Ilustración 121. Eventos y acciones del botón de despaletizado.

5.3.3.4.3. Paletizado de 12 unidades

La acción del paletizado de doce unidades en el HMI está compuesta por cinco páginas. A continuación realizaremos la explicación de las cinco páginas y sus objetos, y las nombraremos por su denominación en Sysmac Studio.

- **Tipo_Creacion_receta_12**

Esta es la primera página que vemos una vez seleccionado el paletizado de doce unidades, esta página usa de plantilla la denominada "Plantilla_Principales", y pose un botón a la pantalla de inicio, es decir, a la "Page0", de la misma manera que hemos explicado con anterioridad. También posee dos zonas muy diferenciadas, una para trabajar con una receta desde cero y la otra para trabajar con una receta anterior, ya sea enviándola sin modificar o pudiendo realizar cambios sobre ella.

Ilustración 122. Página "Tipo_Creacion_receta_12" en Sysmac Studio.

Comenzamos explicando el botón "CREAR RECETA DESDE CERO", que corresponde con la asignación de "Button3", con este botón reseteamos las variables de las posiciones y comenzamos a crear una receta desde cero. Para ello son necesarias dos acciones o eventos, en la primera mostramos la página para seleccionar la nueva receta, que es la página con el nombre "Receta_Paletizado_12", y en la segunda en la que se invierte la variable global "Reset_Pos", para poner en el PLC todas las posiciones a cero, es decir sin ninguna pieza. Tal y como se puede ver en la siguiente imagen.

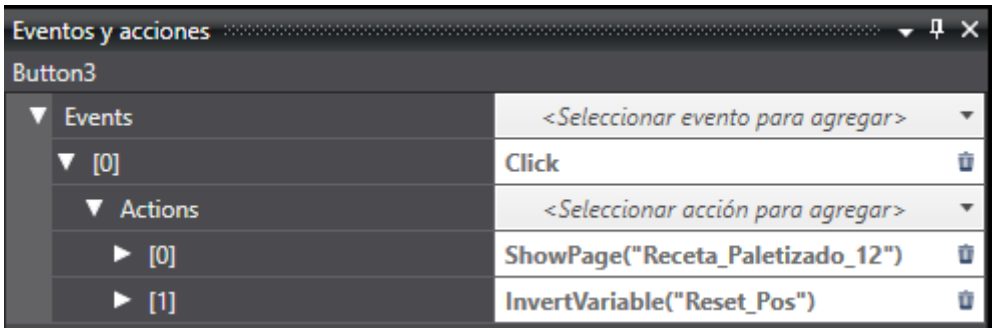


Ilustración 123. Acciones y eventos del botón crear receta desde cero

Por otro lado, tenemos los dos botones que no resetean las posiciones, sino que las guardan y las usan. El botón superior que es el de "MODIFICAR RECETA ANTERIOR", y que corresponde al "Button4", solo posee una acción, y es mostrar la página para poder seleccionar la receta al realizar clic, es decir, te manda a la página "Receta_Paletizado_12". Y sin embargo, el botón "ENVIAR RECETA ANTERIOR" correspondiente al "Button0", se salta la página de la selección de la receta y te manda directamente a visualizarla, es decir, al realizar clic sobre él nos traslada a la página llamada "Visualizar_receta_Pallet12".

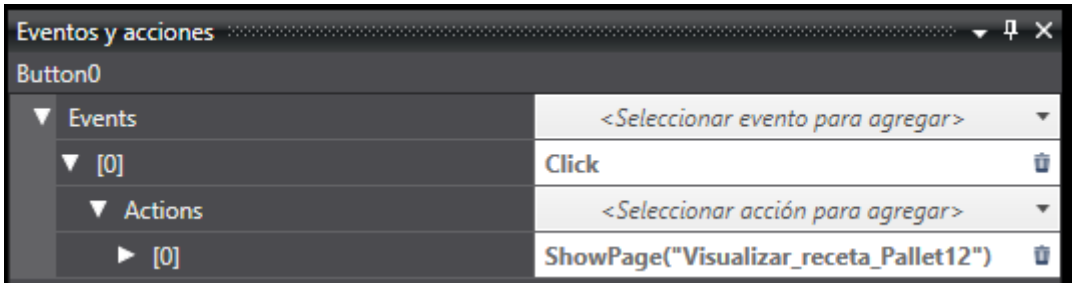


Ilustración 124. Acciones y eventos del botón enviar receta anterior.

- **Receta_Paletizado_12**

En esta página se realiza la selección de las figuras que quieres asignar a cada posición, por forma y color, cada vez que tienes seleccionada un componente de la receta, le debes dar al botón "ENVIAR ORDEN", y una vez terminada tu selección debes de darle al botón "VISUALIZAR SELECCIÓN" para comprobar que la receta es correcta, antes de realizar el paletizado. Si en alguno de los dos desplegables de la derecha, es decir, los de forma o color, seleccionas la opción "VACIO", la posición del pallet quedara vacía. Para esta página hemos usado la plantilla denominada "Plantilla_Secundarias".

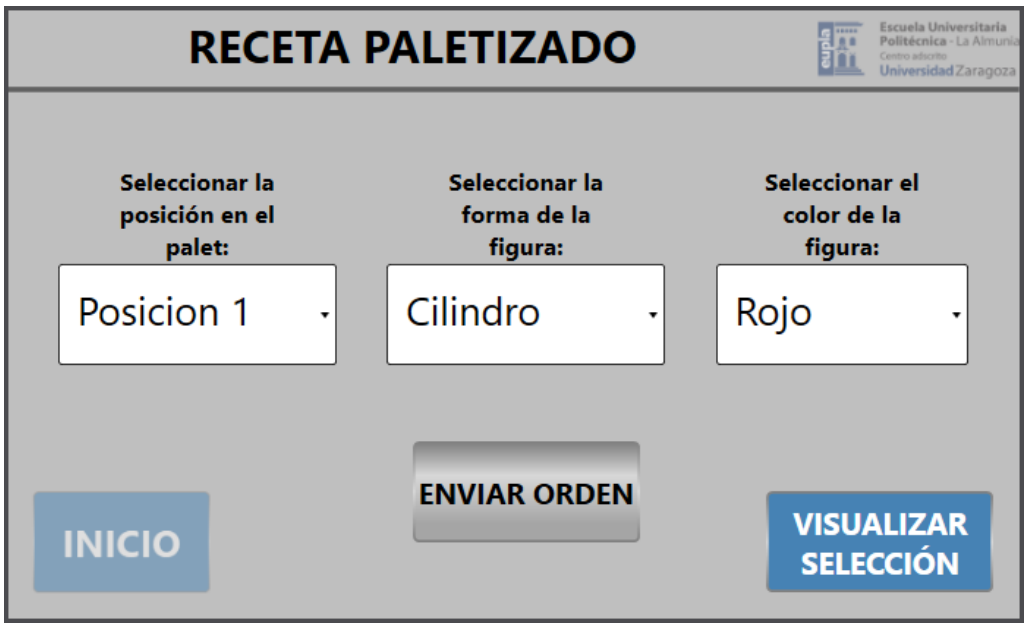


Ilustración 125.Página "Receta_Paletizado_12" en Sysmac Studio.

Comenzamos la explicación de la página con el botón de "ENVIAR ORDEN", el cual corresponde al "Button0", este solo posee una acción, y es la de invertir la variable "Establecer_Receta_Paletizado", al realizar clic. Con esta variable actualizamos el valor de la posición a la que estamos haciendo referencia.

Seguimos con el botón denominado "VISUALIZAR SELECCION", que corresponde al "Button1". Este botón pose solo una acción, y es la de mostrar la página "Visualizar_receta_Pallet12", donde podremos ver la receta realizada.

Y por último explicamos los tres desplegables que son la parte principal de esta página. Estos objetos son del tipo DropDown, es decir, podemos asignar unos Strings a unos valores numéricos del tipo DINT. Por lo tanto obtenemos que de cada desplegable sacamos un número correspondiente al elemento seleccionado, esto en el PLC lo usamos para sacar el tipo de pieza que el usuario ha seleccionado por pantalla.

Estos componentes se programan dentro de sus propiedades de una manera muy sencilla e intuitiva, solo debemos de asignarle una variable de tipo integer.

A Continuación mostraremos los valores de los tres componentes.

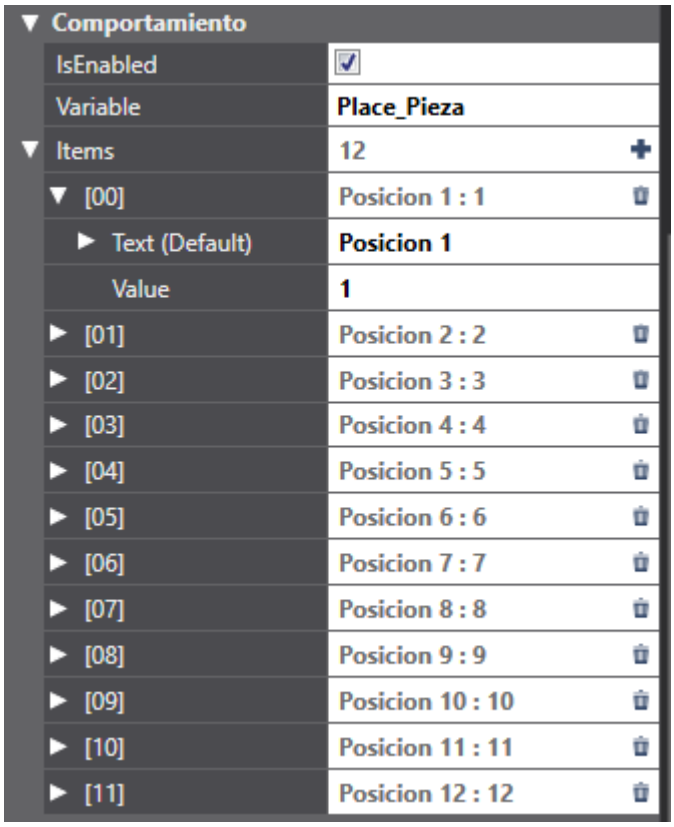


Ilustración 126. Selección de posiciones por el DropDown0.

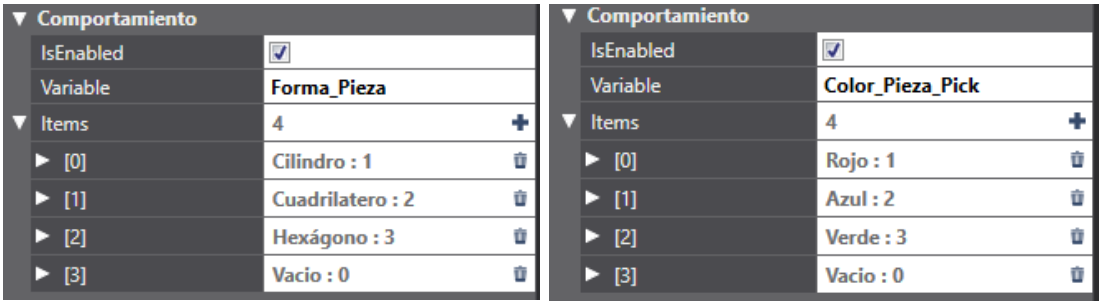


Ilustración 127. Selección de formas y colores por el DropDown1 y el DropDown2.

Por lo tanto, cuando "Place_Pieza" tenga el valor de 1, "Forma_Pieza" tenga el valor 1, y "Color_Pieza_Pick" tenga el valor de uno, esto se traduce en que para la posición uno se desea paletizar un cilindro rojo, y así para todos los componentes.

- **Visualizar_receta_Pallet12**

En esta página se visualiza la receta a enviar, siempre con la posibilidad de volver a modificarla antes de enviarla. La programación de esta página se vuelve muy compleja ya que existen muchos elementos en ella, y es necesario programar en ST mediante subrutinas locales para llevar a cabo esta visualización del pallet. Esta página usa la plantilla denominada "Plantilla_Principales".

Ilustración 128. Página "Visualizar_receta_Pallet12" en Sysmac Studio.

Comenzamos explicando los dos botones que se encuentran en la página. El botón de "MODIFICAR RECETA", que corresponde al "Button3", sirve para regresar a la pantalla anterior, es decir, nos traslada a la página llamada "Receta_Paletizado_12". Y el botón "ENVIAR RECETA", que corresponde al "Button0", sirve para enviar la orden al PLC de que la receta ya ha sido seleccionada, modificando la variable "NX102MQTT_Modificar_Orden" al valor 2, para enviarnos a la página denominada "Proceso_Palletizacion_12", donde tendremos que esperar a que se finalice el paletizado de doce unidades.

Esto se consigue con dos acciones producías al presionar el botón, tal y como se muestran en la siguiente imagen.

Ilustración 129. Eventos y acciones del botón enviar receta

Por otro lado, vamos a explicar cómo se consigue la visualización de las piezas seleccionadas. Primero se añade al fondo la foto del pallet seleccionado, y después se añade en cada posición del pallet todos tipos de figuras que poseemos, en este caso, existen tres colores y tres formas, por la tanto añadimos nueve figuras una sobre otra, y ponemos que no sean visibles. Una vez realizado este proceso sobre todas posiciones del pallet, realizamos una llamada a una subrutina, que se produzca cuando ingresamos a la página, ya que no se van a modificar las piezas seleccionadas una vez estemos en ella. Los datos que debes de añadir en eventos y acciones de la página son los que aparecen en la siguiente imagen.

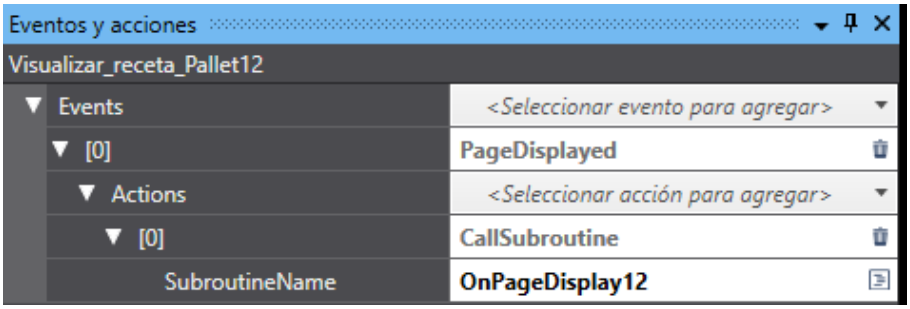


Ilustración 130. Subrutina de la página "Visualizar_receta_Pallet12" en Sysmac Studio.

Para acceder a la subrutina solo debemos de clicar a la derecha del nombre de esta, es decir, a la derecha de "OnPageDisplay12". Una vez dentro debemos de realizar un código en texto estructurado para que cuando se seleccione una pieza en una posición, solo sea visible esa y el resto desaparezcan, y así para todas las posiciones.

Por lo tanto el programa que queda es de una gran cantidad de líneas, ya que debemos de poner a false el resto de objetos que se encuentran en la misma posición que el elemento seleccionado, y el código resultante se encuentra en el anexo. Mostramos aquí una parte del código para que se entienda la estructura.

```
Sub OnPageDisplay12
        If NX102MQTT_OrderP1=0 Then
                Rectangle_Red_P1.IsVisible=False
                Ellipse_Roja_P1.IsVisible=False
                Hexagono_Red_P1.IsVisible=False
                Rectangle_Blanco_P1.IsVisible=False
                Ellipse_Blanca_P1.IsVisible=False
                Hexagono_Blanca_P1.IsVisible=False
                Rectangle_Verde_P1.IsVisible=False
                Ellipse_Verde_P1.IsVisible=False
                Hexagono_Verd_P1.IsVisible=False
        End If

        If NX102MQTT_OrderP1=1 Then
                Rectangle_Red_P1.IsVisible=False
                Ellipse_Roja_P1.IsVisible=True
                Hexagono_Red_P1.IsVisible=False
                Rectangle_Blanco_P1.IsVisible=False
                Ellipse_Blanca_P1.IsVisible=False
                Hexagono_Blanca_P1.IsVisible=False
                Rectangle_Verde_P1.IsVisible=False
                Ellipse_Verde_P1.IsVisible=False
                Hexagono_Verd_P1.IsVisible=False
        End If

                                .
                                .
                                .
End Sub
```

Tal y como se puede ver, para cada variables "NX102MQTT_OrderPXX" debemos de comprobar qué valor tiene del 1 al 9, y mostrar por pantalla solo el valor seleccionado. Con este código conseguimos el resultado deseado, y mostramos por pantalla lo que el usuario ha seleccionado de manera inmediata.

- **Proceso_Palletizacion_12**

En esta página mostramos, igual que en la anterior, por medio de una subrutina, las piezas que ha elegido el usuario en cada posición, y usamos la subrutina global para saber el estado de la paletización. Según este mostraremos un letrero u otro por pantalla, y solo cuando se haya terminado, dejaremos al usuario ver el pallet obtenido por pantalla.

Ilustración 131. Página "Proceso_Palletizacion_12" en Sysmac Studio.

La visualización del pallet no la vamos a explicar ya que es el mismo procedimiento que en la página anterior, sin embargo, en esta se usa la subrutina "Finalizado_Paletizado", para saber el estado del paletizado cada 200 milisegundos.

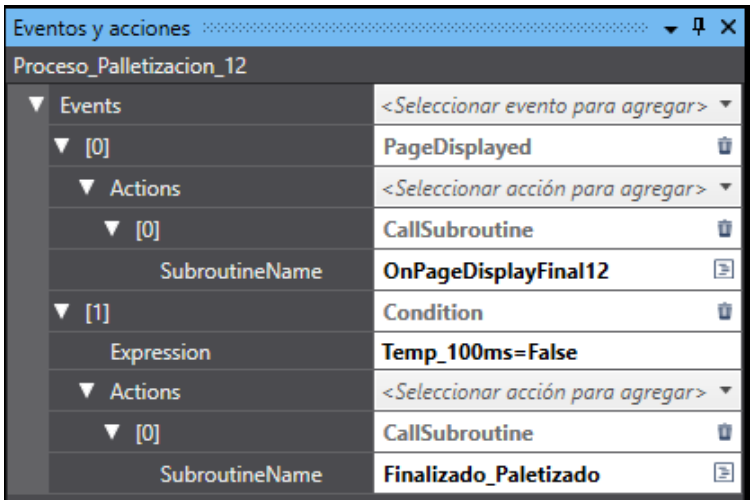


Ilustración 132. Subrutinas de la página "Proceso_Palletizacion_12" en Sysmac Studio.

La subrutina "Finalizacion_Paletizado", está realizado en ST, y se basa en un contador, ya que si es necesario podríamos alargar el tiempo con el que se revisa que la finalización del paletizado no ha ocurrido, y una condición que comprueba en qué estado del bucle esta. Es decir, si la orden de finalizado no ha sido enviada por el Cobot, la pantalla permaneceré haciendo visible solo el letrero de "REALIZANDO PALLETIZADO DE 12U", y en el momento que esta orden sea recibida, la pantalla mostrara el botón de "VER PALLET CREADO" y el letrero de "PALLET DE 12U FINALIZADO". El código realizado para esta subrutina es el siguiente.

```
Dim Contador As Integer

Sub Finalizado_Paletizado

        If Contador < 2 Then
                Contador=Contador+1
        Else
                Contador=0
                If NX102MQTT_NStep= 0 Or NX102MQTT_NStep=10 Or NX102MQTT_NStep=20 Or
NX102MQTT_NStep=30 Then
                        TextBox0.IsVisible=False
                        TextBox1.IsVisible=True
                        Button2.IsVisible=True
                Else
                        TextBox0.IsVisible=True
                        TextBox1.IsVisible=False
                        Button2.IsVisible=False
                End If
        End If
End Sub
```

Junto a todo esto, una vez que el botón "VER PALLET CREADO", que corresponde al "Button2", realiza la acción de enviarnos a otra página al realizar una pulsación sobre él, y nos traslada a la página "Pallet_Creado_12", en la cual podremos ver el pallet que se ha realizado realmente.

- **Pallet_Creado_12**

Esta es la página que se muestra al final del ciclo, en ella se saca por pantalla las piezas que han sido paletizadas, y un pequeño letrero que muestra si la paletización se ha realizado completamente o alguna de las piezas no ha podido ser paletizada.

Ilustración 133. Página "Pallet_Creado_12" en Sysmac Studio.

Uno de los componentes de esta página es el botón de "INICIO" ya explicado con anterioridad. Por otro lado el resto de componentes depende directamente de la subrutina llamada "MostrarPalletFinal", la cual se ejecuta cuando ingresamos en la página. La programación de esta subrutina se basa en el mismo principio que en la subrutina llamada "OnPageDisplay12", pero se le añade la comparación de que las variables "TM_PosXX" sean diferentes de 0, ya que el cobot devuelve 0 si el paletizado de esa posición se ha realizado sin error. También se usa una variable para realizar un pequeño "Switch Case", la variable es un integer y se denomina "VisualizarPalletInt".

El código usado se encuentra en el anexo. Pero aquí mostramos una parte para que se pueda entender la estructura.

```
"_______Mostrar pallet creado_____________

Sub MostrarPalletFinal
        '_______Mostrar las piezas envidas a paletizar_____________________________
        If VisualizarPalletInt=0 Then
                If NX102MQTT_OrderP1=0 Then
                        Rectangle_Red_P1.IsVisible=False
                        Ellipse_Roja_P1.IsVisible=False
                        Hexagono_Red_P1.IsVisible=False
                        Rectangle_Blanco_P1.IsVisible=False
                        Ellipse_Blanca_P1.IsVisible=False
                        Hexagono_Blanca_P1.IsVisible=False
                        Rectangle_Verde_P1.IsVisible=False
                        Ellipse_Verde_P1.IsVisible=False
                        Hexagono_Verd_P1.IsVisible=False
                        End If
                                .
                                . (Mismo código que para la subrutina "OnPageDisplay12")
                                .
                VisualizarPalletInt=2
         End If

        '__________Sacar en un cuadro de texto las posiciones no paletizadas_________
        If VisualizarPalletInt=2 Then
                If NX102MQTT_TM_Pos1=0 And NX102MQTT_TM_Pos2=0 And
NX102MQTT_TM_Pos3=0 And NX102MQTT_TM_Pos4=0 And NX102MQTT_TM_Pos5=0 And
NX102MQTT_TM_Pos6=0 And NX102MQTT_TM_Pos7=0 And NX102MQTT_TM_Pos8=0 And
NX102MQTT_TM_Pos9=0 And NX102MQTT_TM_Pos10=0 And NX102MQTT_TM_Pos11=0 And
NX102MQTT_TM_Pos12=0 Then
                        TextBox0.IsVisible=True
                        TextBox1.IsVisible=False
                Else
                        TextBox0.IsVisible=False
                        TextBox1.IsVisible=True

                '_______Mostrar pallet creado_____Quitando las piezas no puestas___________
                        If NX102MQTT_TM_Pos1 <>0 Then
                                Rectangle_Red_P1.IsVisible=False
                                Ellipse_Roja_P1.IsVisible=False
                                Hexagono_Red_P1.IsVisible=False
                                Rectangle_Blanco_P1.IsVisible=False
                                Ellipse_Blanca_P1.IsVisible=False
                                Hexagono_Blanca_P1.IsVisible=False
                                Rectangle_Verde_P1.IsVisible=False
                                Ellipse_Verde_P1.IsVisible=False
                                Hexagono_Verd_P1.IsVisible=False
                                End If
                                .
                                .
                                .
                        End If

                VisualizarPalletInt=0
                End If
        End Sub
```

Con esto finalizamos el grupo de páginas del HMI para la paletización de doce unidades.

5.3.3.4.4. Paletizado de 13 unidades

La acción del paletizado de trece unidades en el HMI está compuesta por cinco páginas muy similares a las utilizadas en el paletizado de doce unidades, ya que solo se modifican las páginas a las que nos llevan los botones, se le añade una posición más para la selección de la receta, se modifica la imagen del pallet y se le añade las nueve figuras más de esta posición cuando son necesarias. Realizaremos la explicación de las cinco páginas y sus objetos, basándonos en la explicación del apartado anterior, y las nombraremos por su denominación en Sysmac Studio.

- **Tipo_Creacion_receta_13**

Esta es la primera página que vemos una vez seleccionado el paletizado de trece unidades, esta página usa de plantilla la denominada "Plantilla_Principales", y es exactamente igual que la página "Tipo_Creacion_receta_12", ya explicada previamente, solo cambia la acción de los botones, ya que en vez de enviarnos a las páginas del grupo "Paletizado12" nos envían a las páginas del grupo "Paletizado13".

Ilustración 134. Página "Tipo_Creacion_receta_13" en Sysmac Studio.

- **Receta_Paletizado_13**

En esta página se realiza la selección de las figuras que quieres asignar a cada posición, por forma y color, de la misma forma que hemos explicado para la página "Receta_Paletizado_12", la única diferencia es que en el desplegable para seleccionar la posición se puede seleccionar la opción de "Posición 13", y que el botón de "Visualizar Selección", nos envía a la página "Visualizar_receta_Pallet13" en vez de a la página del anterior grupo.

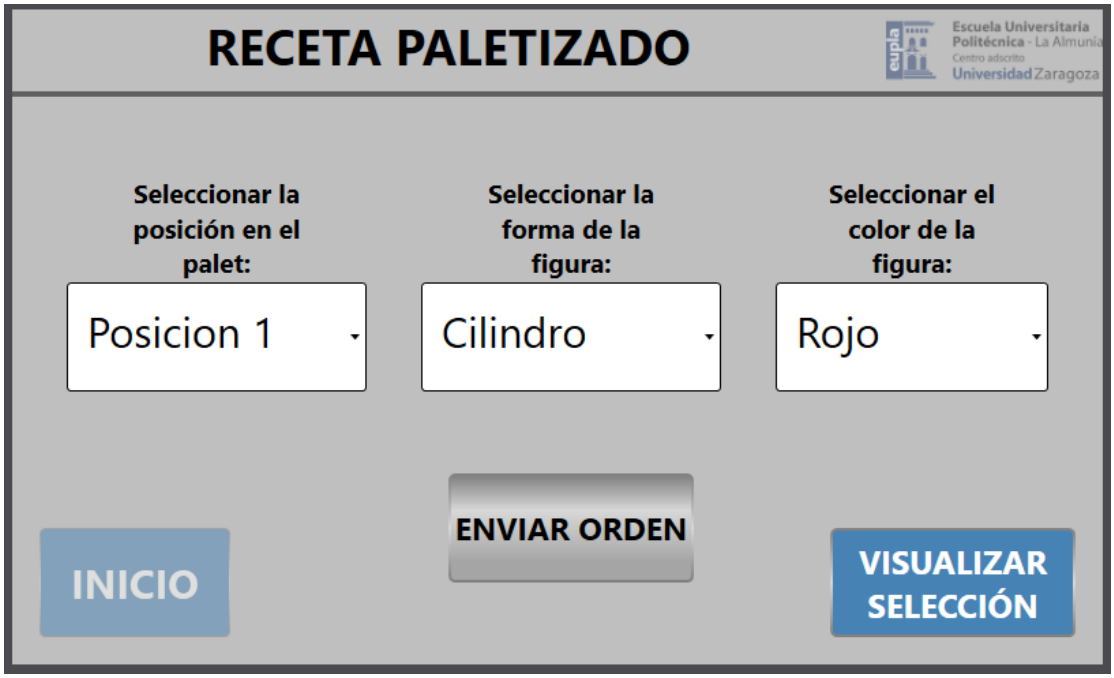


Ilustración 135.Página "Receta_Paletizado_13" en Sysmac Studio.

- **Visualizar_receta_Pallet13**

En esta página se visualiza la receta a enviar, siempre con la posibilidad de volver a modificarla antes de enviarla. A partir de esta página encontramos diferencias gráficas, ya que modificado la fotografía del pallet y hemos añadido nueve figuras más, las correspondientes a la nueva posición. Pero el funcionamiento es exactamente el mismo que para la página "Visualizar_receta_Pallet12", solo que añadiéndole a la programación en ST la nueva posición son sus respectivos objetos. El código se muestra en el anexo.

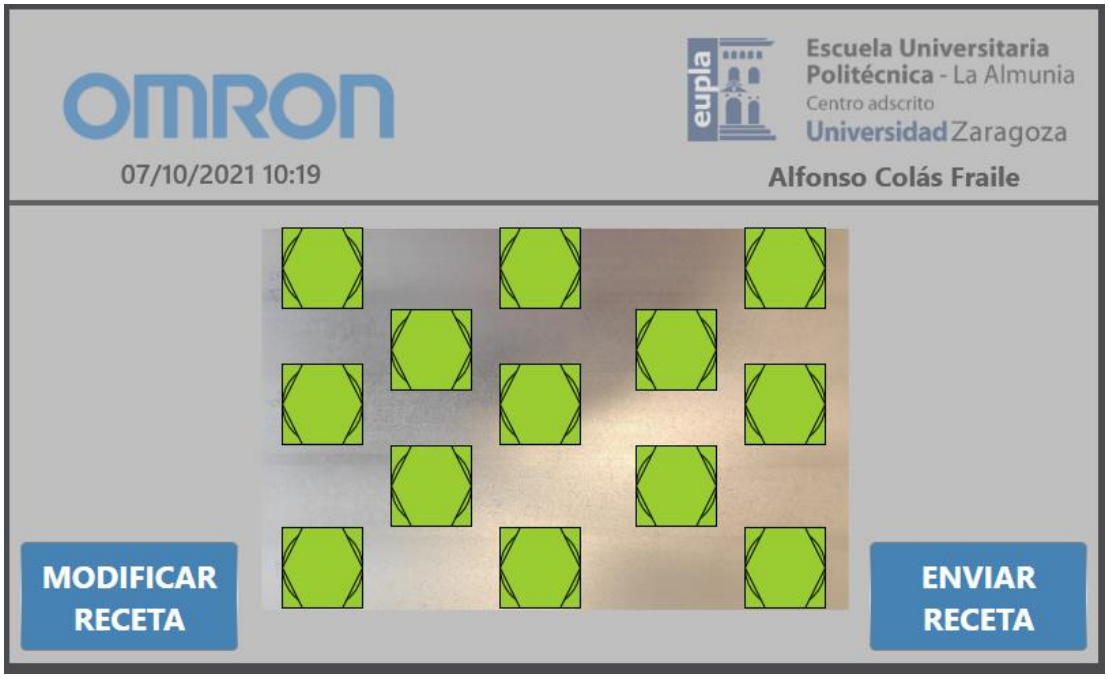


Ilustración 136. Página "Visualizar_receta_Pallet13" en Sysmac Studio.

- **Proceso_Palletizacion_13**

Esta página es igual que la página "Proceso_Palletizacion_12", con la única diferencia de que modificamos la imagen del pallet y añadimos los nueve objetos correspondientes, y a la vez que el botón "VER PALLET CREADO" nos traslada a la página "Pallet_Creado_13" en vez de "Pallet_Creado_12".

En esta página se muestra, por medio de una subrutina, las piezas que ha elegido el usuario en cada posición, y usamos la subrutina global para saber el estado de la paletización. Según este mostraremos un letrero u otro por pantalla, y solo cuando se haya terminado, dejaremos al usuario ver el pallet obtenido por pantalla. El código se muestra en el anexo.

Ilustración 137. Página "Proceso_Palletizacion_13" en Sysmac Studio.

- **Pallet_Creado_13**

Esta página es igual que la página "Pallet_Creado_12", con la única diferencia de que modificamos la imagen del pallet y añadimos los nueve objetos correspondientes.

La página se muestra al final del ciclo, en ella se saca por pantalla las piezas que han sido paletizadas, y un pequeño letrero que muestra si la paletización se ha realizado completamente o alguna de las piezas no ha podido ser paletizada.

Ilustración 138. Página "Pallet_Creado_13" en Sysmac Studio.

```
Dim Contador As Integer

Sub Finalizado_Despaletizado

    If Contador < 2 Then
        Contador=Contador+1
    Else
        Contador=0
        If NX102MQTT_NStep= 0 Or NX102MQTT_NStep=10 Or NX102MQTT_NStep=20 Or
NX102MQTT_NStep=30 Then
            TextBox0.IsVisible=False
            TextBox1.IsVisible=True
            Button0.IsVisible=True
        Else
            TextBox0.IsVisible=True
            TextBox1.IsVisible=False
            Button0.IsVisible=False
        End If
    End If
End Sub
```

5.3.3.4.5. Despaletizado

La acción del despaletizado en el HMI está compuesta por una sola página, llamada "Page6", ya que en este proceso no debemos de realizar ninguna selección, una vez que decidimos despaletizar solo debemos esperar a que el proceso se haya realizado.

- **Page6**

Esta página usa la plantilla denominada "Plantilla_Principales" y está compuesta por tres elementos, los cuales dependen todos de una subrutina.

Ilustración 139. Página "Page6" en Sysmac Studio.

La subrutina utilizada es llamada "Finalizado_Despaletido", y coprueba en que caso del "Switch case" del PLC se encuentra, mediante la variable "NStep". Con esto hace visibles e invisibles a los tres objetos de la página.

Estos objetos son, un botón, "Button0", que nos lleva a la página de inicio, es decir, a la "Page0", un texto que muestra por pantalla "DESPALETIZADO FINALIZADO", y que corresponde al "TextBox1", y por ultimo un texto que muestra por pantalla "REALIZANDO DESPALETIZADO", y que corresponde al "TextBox0".

Mediante el siguiente código se consigue que cuando se está realizando el despaletizado solo aparezca por pantalla el texto que pone "REALIZANDO DESPALETIZADO", sin embargo cuando el despaletizado termina se muestre por pantalla el botón para ir a inicio y el texto de "DESPALETIZADO FINALIZADO".

```
Dim Contador As Integer

Sub Finalizado_Despaletizado

    If Contador < 2 Then
        Contador=Contador+1
    Else
        Contador=0
        If NX102MQTT_NStep= 0 Or NX102MQTT_NStep=10 Or NX102MQTT_NStep=20 Or
NX102MQTT_NStep=30 Then
            TextBox0.IsVisible=False
            TextBox1.IsVisible=True
            Button0.IsVisible=True
        Else
            TextBox0.IsVisible=True
            TextBox1.IsVisible=False
            Button0.IsVisible=False
        End If
    End If
End Sub
```

5.4. PROGRAMACIÓN TMFLOW

5.4.1. Establecer conexión

La programación en TMflow se realiza siempre conectado por medio de Ethernet con un ordenador o mediante una pantalla conectada por HMI, un teclado y un ratón conectados por USB. En este caso la programación se ha realizado por medio de un ordenador, por lo que se debe de iniciar el robot desde el mando. Una vez iniciado se debe de poner en modo manual desde el mando, tal y como se ve en la siguiente ilustración.

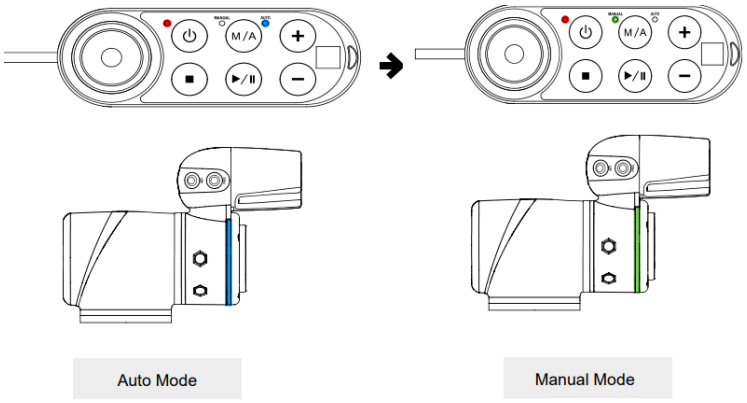


Ilustración 140. Selección del modo manual en el Cobot.[42]

A partir de este momento podremos tomar el control del cobot desde el ordenador. Para ello debes de estar en la misma red IP que el Cobot, en este caso el Cobot tiene la dirección IP 192.168.250.35.

Tal y como se ve en la siguiente imagen, el robot sale por pantalla una vez abierto el programa TMflow, y solo debemos de darle al botón "Get Control", para poder realizar nuestro proyecto.

Ilustración 141. Conectarse al cobot por medio de Ethernet en TMflow.

5.4.2. Creación de un nuevo proyecto

Una vez realizada la conexión debemos de ir a la pestaña "Project", para poder crear nuestro proyecto.

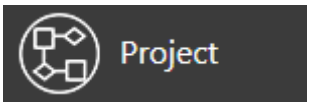


Ilustración 142. Símbolo para entrar a los proyectos en TMflow.

Dentro de proyecto debemos de ir a los símbolos iluminados que se encuentran arriba a la izquierda, y debemos de clicar sobre el que aparece en la siguiente imagen.

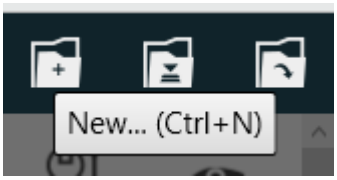


Ilustración 143. Botón para la creación de nuevo proyecto en TMflow.

Debemos de asignarle un nombre al nuevo proyecto, en este caso le asignaremos el nombre de TFG_Alfonso_Colas, y presionaremos el boton "OK", tal y como se muestra en la siguiente imagen.

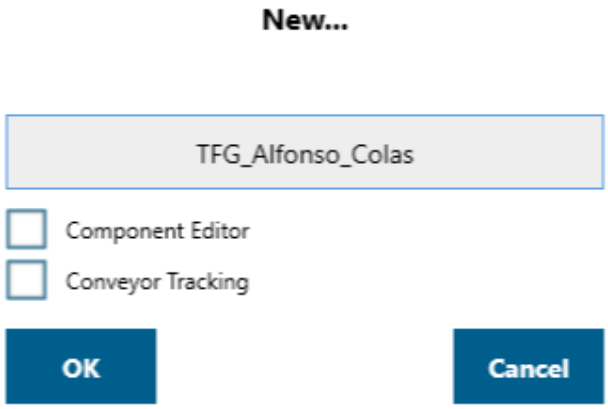


Ilustración 144. Creación de nuevo proyecto en TMflow.

Una vez terminada la creación del nuevo proyecto la ventana que obtenemos, siempre a pantalla completa, es la siguiente.

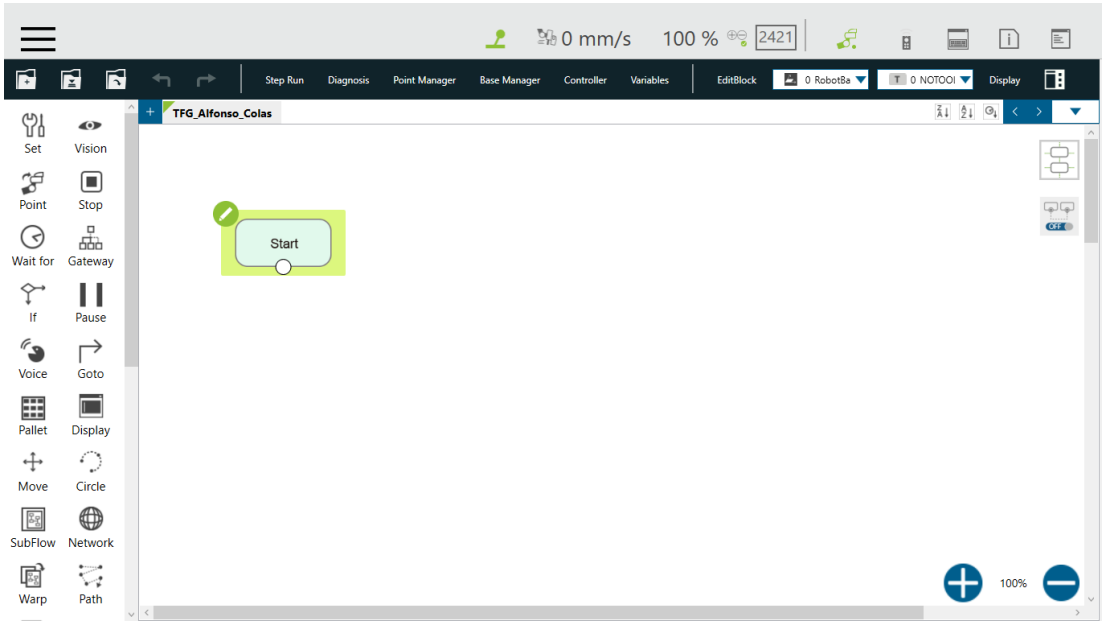


Ilustración 145. Nuevo proyecto en TMflow.

5.4.3. Ejecución de un proyecto

Ahora vamos a aprender a realizar la ejecución del proyecto. La ejecución la realizaremos desde el mando, exactamente desde el botón Play/Pause, tal y como se ve en la siguiente ilustración.

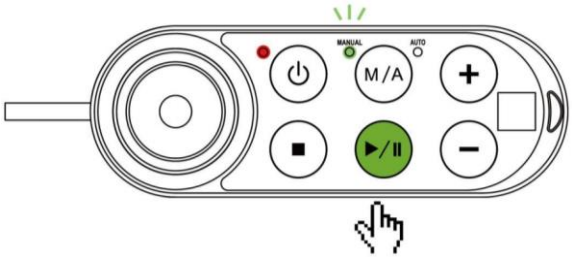


Ilustración 146. Ejecución de un proyecto en TMflow. [42]

Una vez que el proyecto está en ejecución podemos modificar su velocidad desde los botones + y – situados en el mando, solo cuando el Cobot este en modo manual. Para realzar la pausa del proyecto debemos de volver a presionar el botón Play/Pause.

Y por último, para finalizar la ejecución del proyecto debemos de presionar el botón Stop, tal y como se ve en la siguiente imagen.

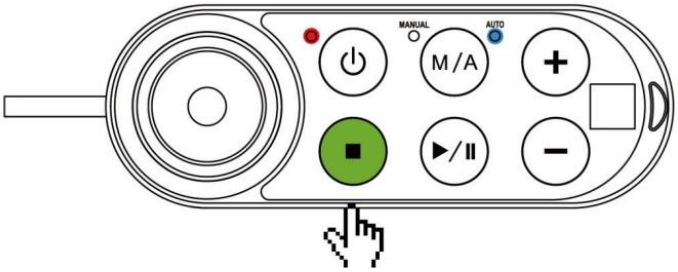


Ilustración 147. Stop de un proyecto en TMflow. [42]

5.4.4. Programación de movimientos

En este apartado vamos a explicar los tres tipos de movimientos que realizara el cobot. El primer movimiento explicado es el PTP, y este va de un punto a otro calculando la trayectoria más rápida, tal y como se puede observar en la imagen.

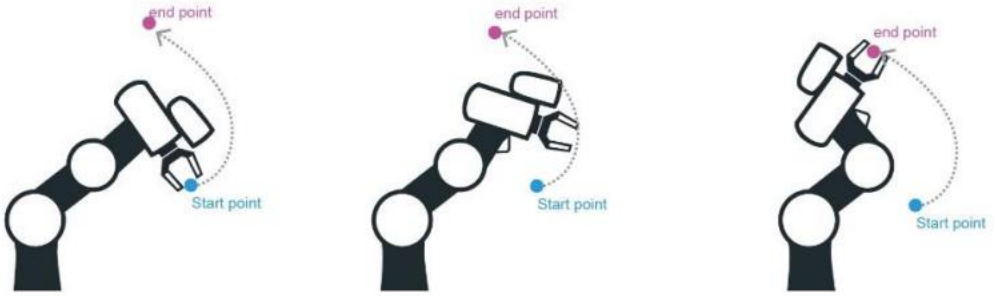


Ilustración 148. Movimiento PTP en TMflow. [42]

El segundo es el LINE, el cual sigue una trayectoria recta de un punto al otro, tal y como se puede ver en la siguiente imagen.

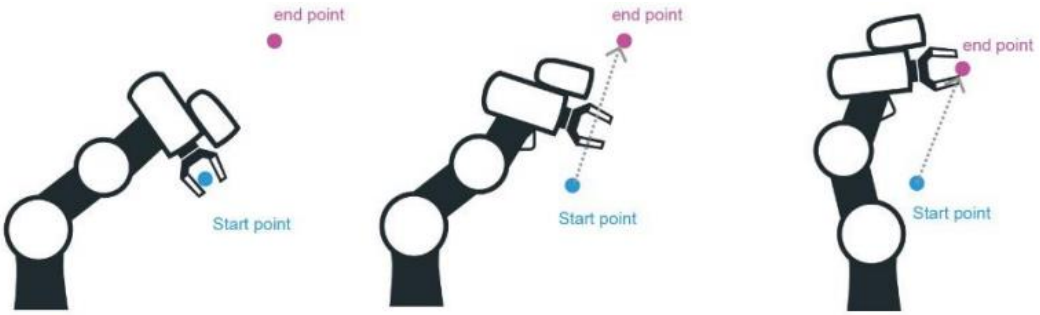


Ilustración 149. Movimiento LINE en TMflow. [42]

Y por último el WayPoint, en este caso se sigue primero una trayectoria PTP para llegar a la vertical del punto y aproximarse desde la parte superior, este punto es usado para realizar tanto el pick como el place de mis piezas.

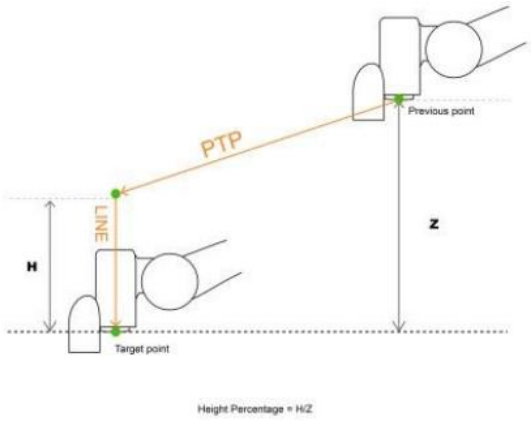


Ilustración 150. Movimiento WayPoint en TMflow. [42]

5.4.5. Lógica de programacion

En este apartado realizaremos una breve explicación de los nodos básicos que se utilizan en TMflow, junto con la explicación de la lógica que se usa para las variables.

5.4.5.1. Variables

Las variables en TMflow pueden ser locales y globales, pero las dos deben de ser de los tipos mostrados en la siguiente tabla.

Type	Type Description	Saved Data
string	String	Structure compostured of characters, such as "TMflow" (double quotes must be added to represent the string)
int	Integer	-2^{31}~$2^{31}-1$
float	Floating point number (decimal)	10^{-37}~10^{38} (Effective digit 6~7 digits)
double	Double-precision floating-point number	10^{-307}~10^{308} (Effective digit 15~16 digits)
bool	Boolean	True, False
byte	Byte	-2^{7}~$2^{7}-1$

Tabla 1.Tipos de variables admitidas en TMflow. [42]

5.4.5.1.1. Variables locales

Las variables locales solo pueden ser usadas en el proyecto que fueron creadas. Para crear estas variables se debe clicar sobre en la pestaña "Variables", y debes de seleccionar si deseas crear una sola variable o un Array.

5.4.5.1.2. Variables globales

Para acceder a las variables globales se debe acceder desde la configuración global del sistema, desde el menú accedemos al apartado "Settings" y clicamos en "Global Variables". Estas variables pueden usarse en todos los proyectos, y el valor que se les asigna es guardado en la memoria, por lo que esos valores se pueden usar en varios proyectos a la vez.

5.4.5.2. Nodos lógicos

5.4.5.2.1. Nodo SET

Este nodo puede establecer los estados de IO y cambiar el tipo y valor de las variables. Al pasar por este nodo, todos los parámetros se cambiarán al valor preestablecido.

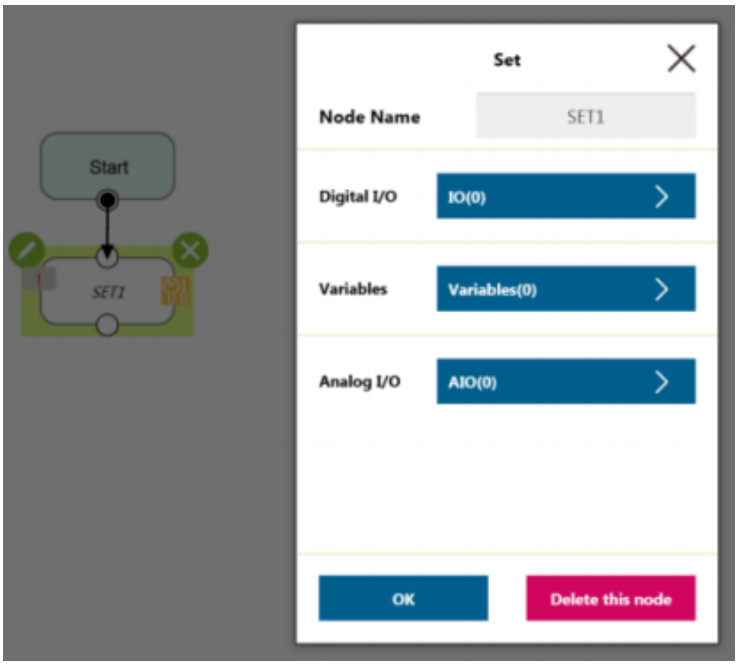


Ilustración 151. Nodo SET en TMflow.

Este nodo solo puede realizar ciertas modificaciones en las variables, y estas son las que aparecen en la siguiente tabla.

Symbol	Symbol description
a+= b	a = a + b
a−= b	a = a − b
a *= b	a = a * b
a/= b	a = a/b
a = b	Specifieda Value is b

Tabla 2. Acciones admitidas a las variables en TMflow. [42]

5.4.5.2.2. Nodo IF

El nodo IF establece condiciones para poder realizar el control de un proceso. El IF puede juzgar o comparar el estado de IO, el estado de una variable y juzgar el cumplimiento de una o varias condiciones. Y tomar el camino del Sí o del No según se alcance la condición.

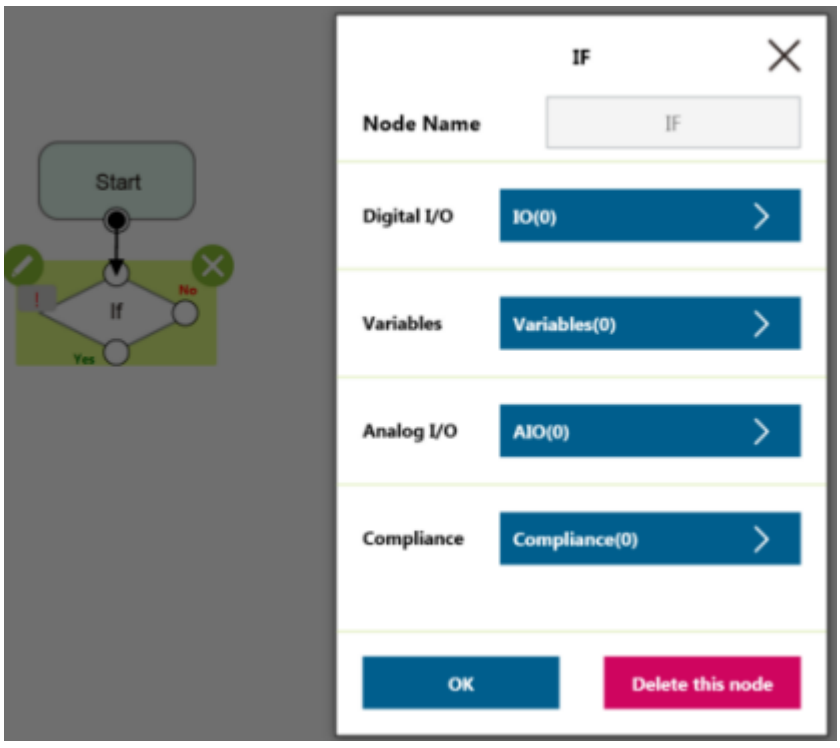


Ilustración 152. Nodo IF en TMflow

Los símbolos que se pueden usar en las comparaciones de variables son los que aparecen en la siguiente tabla.

Symbol	Symbol Description
<	Less than
>	More than
==	Equal to
<=	Less than or Equal to
>=	More than or Equal to
!=	Not Equal to

Tabla 3. Comparaciones admitidas a las variables en TMflow. [42]

5.4.5.2.3. Nodo WaitFor

La función principal del nodo WaitFor es pausar el proyecto y continuar ejecutándolo después de que se cumplan las condiciones establecidas.

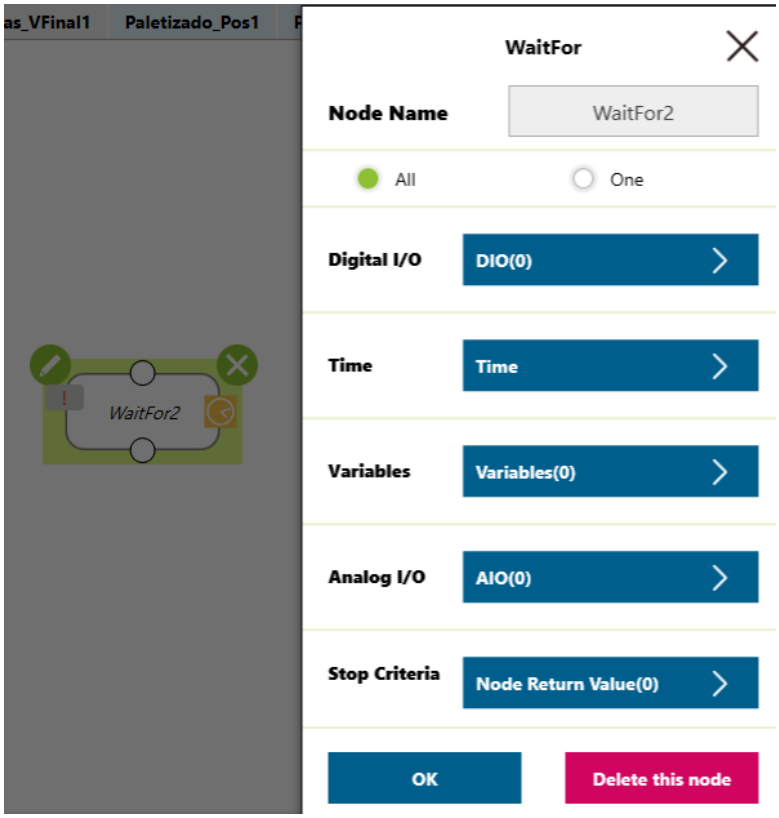


Ilustración 153. Nodo WaitFor en TMflow.

5.4.5.2.4. Nodo Gateway

El nodo Gateway es un nodo similar al nodo IF, ya que se realiza una comparación, pero en este caso la condición se realiza de izquierda a derecha. Cada Subnodo puede usar una variable diferente para la comparación.

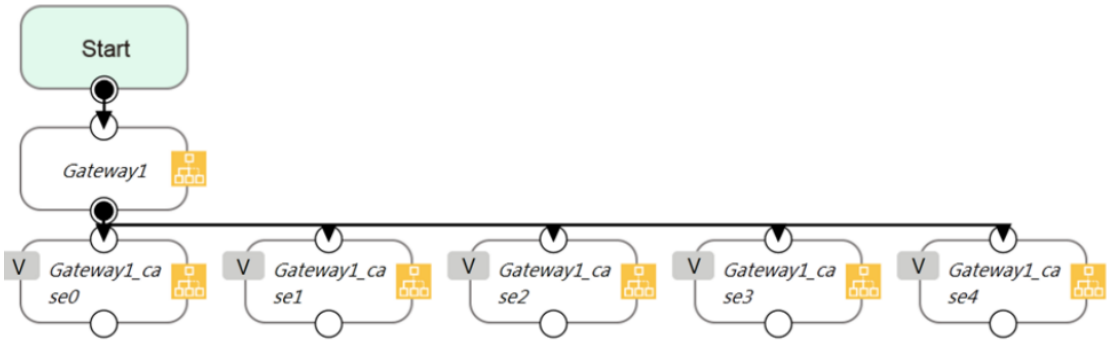


Ilustración 154. Nodo Gateway en TMflow.

5.4.5.2.5. Nodo Goto

El nodo Goto es usado para eliminar líneas de flujo, con el propósito de realizar un programa más comprensible. Este nodo se relaciona a cualquier otro dentro de la misma pestaña, cuando lo clicamos podemos ver con una línea roja al nodo al que está relacionado, tal y como podemos ver en la siguiente imagen.

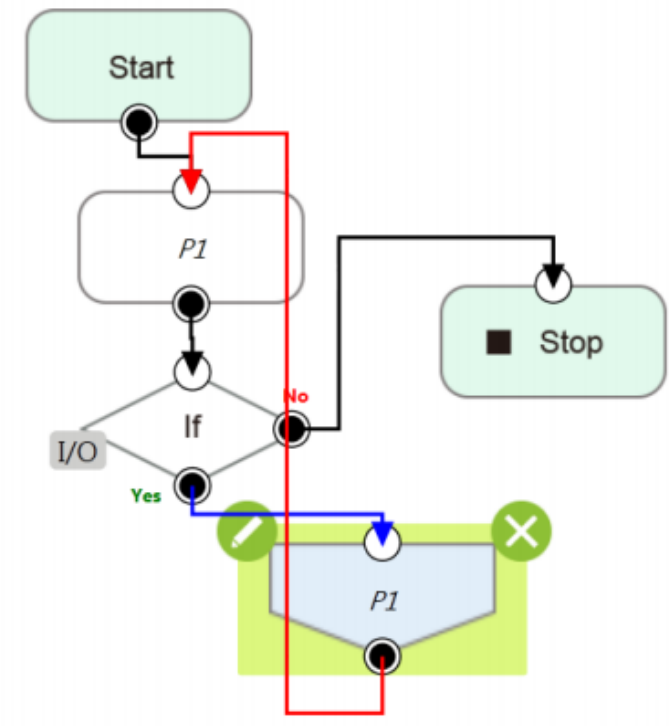


Ilustración 155. Funcionamiento del nodo Goto en TMflow.

5.4.5.2.6. Nodo Subflow

EL nodo Subflow es usado para crear nuevas páginas de nodos, haciendo el programa más limpio y comprensible. Este nodo también es usado como una función, ya que se puede hacer referencia a él todas las veces que sean necesarias, incluso desde otros subflows.

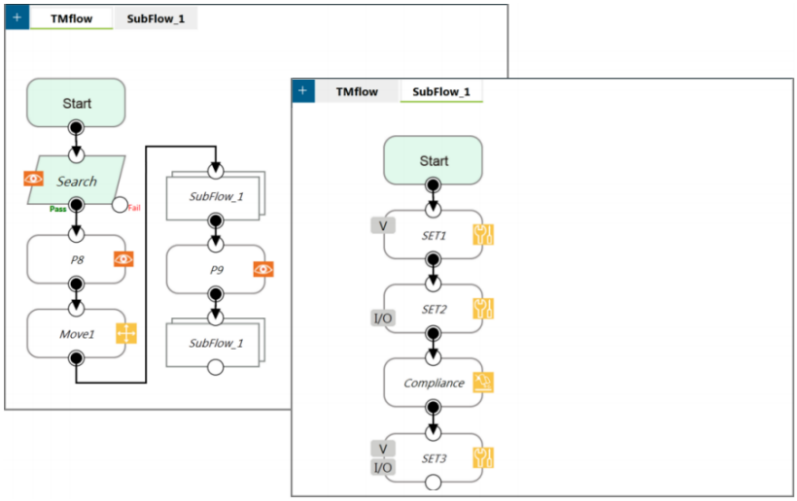


Ilustración 156. Nodo Subflow en TMflow.

5.4.5.2.7. Nodos del gripper

Los nodos para el uso de la pinza te los debes descargar de la página oficial de ROBOTIQ, y posteriormente los debes de instalar en el controlador del robot. En este caso poseemos tres nodos, el de apertura de las pinzas, el de cierre de las pinzas, y por ultimo el de setting, con el cual modificamos los parámetros de apertura, cierre o presión, este se debe de ejecutar mínimo una vez al realizar un proyecto.

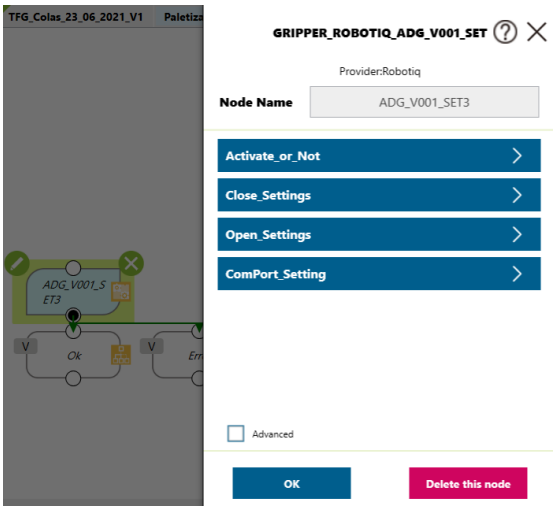


Ilustración 157. Nodo Setting gripper en TMflow.

5.4.6. Nodo de visión

El Vision Node crea un plano usando un punto fijo o un objeto, y a su vez puedes usar multitud de funciones de identificación de AOI. La visualización del nodo de visión en el flujo se realiza como en la siguiente imagen, en la que aparece el icono de base en el lado derecho, que corresponde con el lugar de referencia desde el que se realiza la tarea de visión, y el ícono del lado izquierdo es para saber el número de la base de visión que sea generado con este nodo.

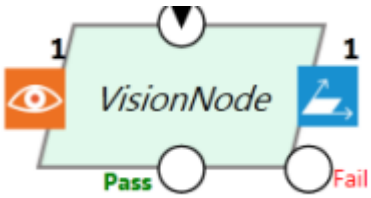


Ilustración 158. Nodo de visión en TMflow.

Para según que tareas de visión es necesaria realizar una calibración específica de la zona en la que se va a realzar el proceso, para ello realizaremos una calibración automática usando la tabla de calibración. Debemos de colocar el robot a la altura que queramos que realice la foto para la tarea de visión, y la tabla de calibración a la altura en la que se observaran los objetos.

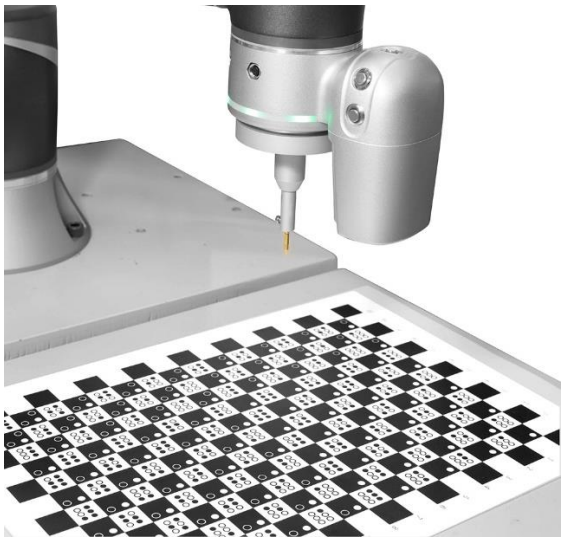


Ilustración 159. Calibración de visión en TMflow. [42]

Debido a la gran cantidad de aplicaciones y tareas de visión que posee la serie TM, solo explicaremos con detalle las que han sido usadas para este proyecto.

5.4.6.1. Aplicaciones

Las aplicaciones que existen en TMflow son las representadas en la siguiente tabla.

Aplicaciones	Tipo de cámara aceptada	Workspace	Creación de una basa
Fixed	Eye-in-Hand / Eye-to-Hand	✓	Crear una base usando una referencia
Servoing	Eye-in-Hand	×	Cree una base. usando la posición del robot
AOI-only	Eye-in-Hand / Eye-to-Hand	×	×
Vision IO	Eye-in-Hand / Eye-to-Hand	×	×
Landmark Alignment	Eye-in-Hand	×	Crear una base usando una Landmark
Object-based Calibration	Eye-in-Hand	×	Crear una base usando la posición del objeto
Smart-Pick	Eye-in-Hand		

Tabla 4. Aplicaciones de las tareas de visión en TMflow. [42]

Los usuarios pueden guardar imágenes de visión estableciendo criterios basados en los resultados de detecciones, reconocimientos y mediciones de objetos. Las imágenes disponibles para guardar incluyen la imagen original y la última imagen tomada.

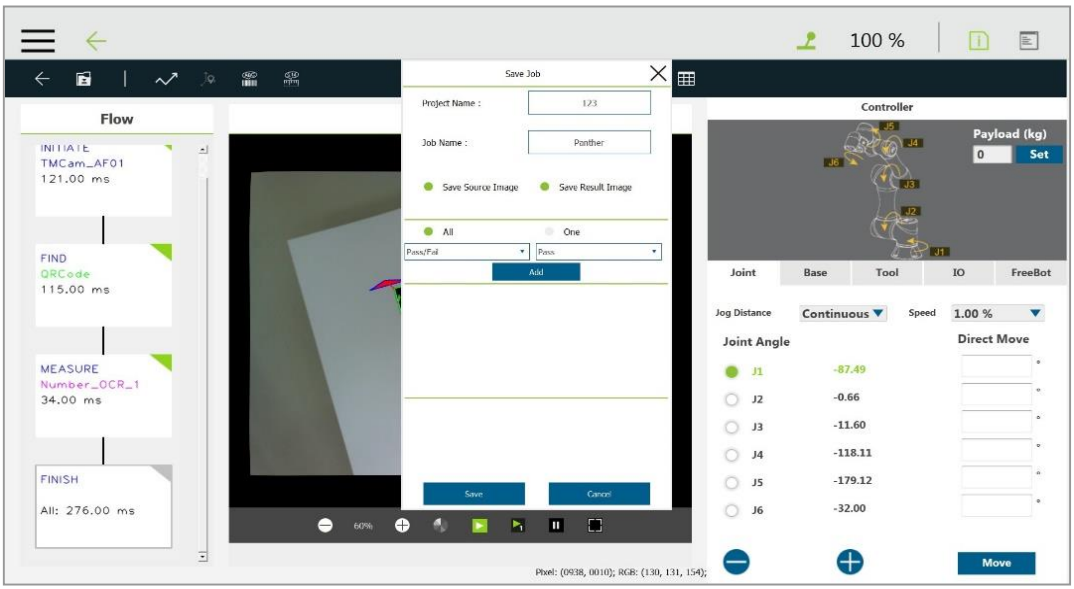


Ilustración 160. Guardado de imágenes según resultados. [42]

5.4.6.1.1. Fixed Point

Para realizar una tarea del tipo Fixed Point debe ingresar en la ventana "TMvision Task Designer" y seleccionar "Fixed Point". Esta función está diseñada para EIH y ETH para que el robot calcule y coloque objetos con coordenadas absolutas mediante la creación de espacios de trabajo. La precisión varía con la de la calibración del espacio de trabajo.

El primer paso que debes de realizar en esta tarea es ajustar los parámetros que aparecen en la siguiente tabla.

Nombre	Función
Adjust camera parameters	Ajustar los parámetros de la cámara como el enfoque de la cámara, los contrastes y el balance de colores.
Switch to record image	Utilice las imágenes internas de TM SSD para la identificación.
Start at initial position	Marque esto para devolver el robot a su posición inicial antes de la identificación visual. Desmarque esto y el robot ejecutará la identificación visual en la posición actual.
Move to the initial position	Mueva el robot a la posición inicial.
Reset workspace	Restablece el espacio de trabajo del robot.
Lighting	Enciende o apaga la luz de la cámara.
Light Intensity*	Utilice el control deslizante para establecer el nivel de brillo.
Idle for Robot Stablilization	Configure el tiempo de forma manual o automática para que el robot se autoajuste antes de tomar fotografías.
Snap-n-go	Mejore la eficiencia tomando instantáneas al mismo tiempo y manteniendo el flujo para ahorrar tiempo para las tareas posteriores que no son de visión.

Tabla 5. Ajustes para las tareas de visión Fixed Point y AOI-only.

Después de configurar los parámetros básicos de la cámara, seleccione la función. Buscar en la parte superior y seleccione la función de coincidencia de patrones como se muestra a continuación. TMvision utilizará la función de "forma enmarcada" para encontrar su alineación en la imagen y construir la base visual en el objeto.

Tal y como se puede ver en la siguiente imagen, en la que se realiza la tarea buscando las letras del logo TM.

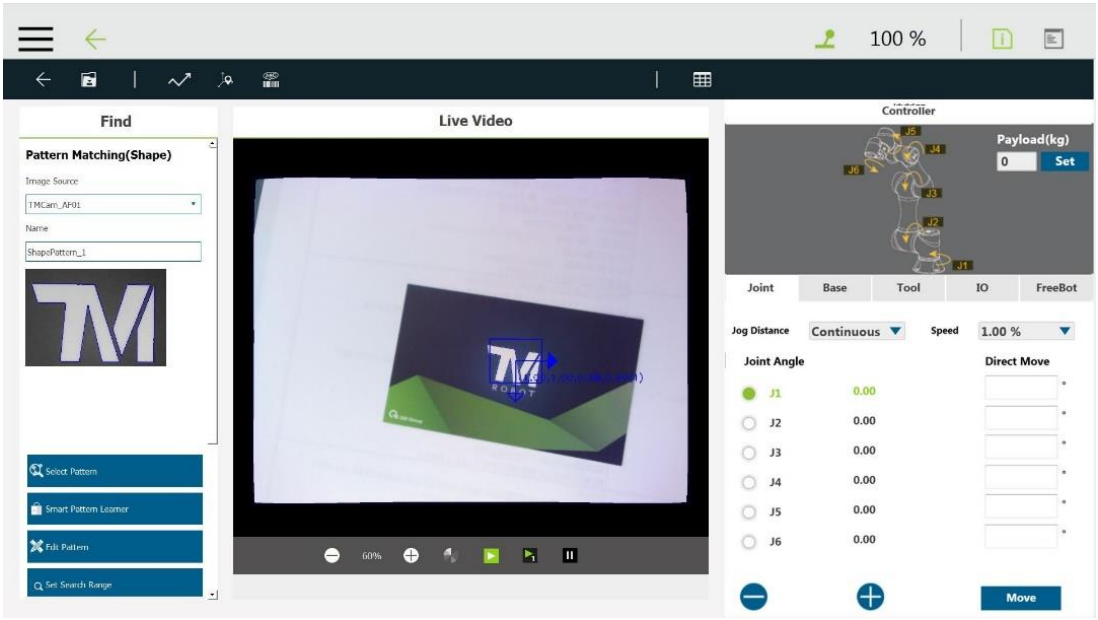


Ilustración 161. Tarea de visión Fixed Point. [42]

5.4.6.1.2. AOI-only

Para realizar una tarea del tipo AOI debe ingresar en la ventana "TMvision Task Designer" y seleccionar "AOI-only" para usar esta función. La identificación de AOI-only es aplicable a EIH o ETH para leer códigos de barras y códigos QR, clasificador de colores y coincidencia de cadenas sin espacio de trabajo y salida del sistema base.

Los ajustes que se deben de realizar son los mismos que aparecen en la tabla 5 del apartado anterior.

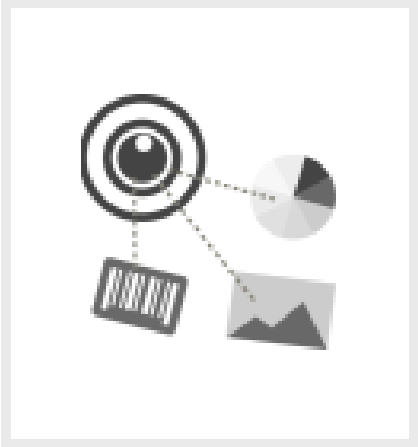


Ilustración 162. Tarea de visión AOI-only. [42]

5.4.6.2. Tareas de visión

Una vez seleccionada la aplicación debes de seleccionar las tareas de visión que se deben de realizar para llevarla a cabo. Estas tareas son las que aparecen en la siguiente tabla.

Tarea de visión	Descripción	Output(Punto flotante)
Pattern Matching (Shape) 	Ubique un objeto en la imagen según sus características geométricas.	En relación con las coordenadas X, Y y el ángulo de rotación R del inicio de la imagen (arriba a la izquierda).
Pattern Matching (Image) 	Localice un objeto en la imagen según sus características de distribución de valor de píxeles.	En relación con las coordenadas X, Y y el ángulo de rotación R del inicio de la imagen (arriba a la izquierda).
Blob Finder 	Localice un objeto por la diferencia de color entre el objeto y el fondo.	En relación con las coordenadas X, Y y el ángulo de rotación R del inicio de la imagen (arriba a la izquierda).
Anchor 	Cambie las coordenadas de inicio de la detección de objetos ajustando manualmente el punto de anclaje.	En relación con las coordenadas X, Y y el ángulo de rotación R del inicio de la imagen (arriba a la izquierda).
Fiducial Mark Matching 	Utilice las dos características obvias del objeto para hacer coincidir.	En relación con las coordenadas X, Y, el ángulo de rotación R del inicio de la imagen (arriba a la izquierda) y la etiqueta del objeto.
External Detection 	Utilice una plataforma informática remota con el protocolo HTTP para la detección y el posicionamiento de objetos.	En relación con las coordenadas X, Y, el ángulo de rotación R del inicio de la imagen (arriba a la izquierda) y la etiqueta del objeto.

Tabla 6. Tareas de visión en TMflow.

5.4.6.2.1. Pattern Matching (Shape)

La función usa la forma geométrica del objeto como su modelo de patrón y la compara con la imagen de entrada para encontrar el objeto en la imagen. Admite variaciones debidas a la rotación y dimensión de los objetos. Debes de completar las funciones que aparecen en la siguiente tabla para realizar la tarea de visión.

Nombre	Función
Pattern Selection	Después de la selección, aparecerá la imagen, y el usuario puede seleccionar el objeto en ella.
Smart Pattern Learner	Para crear tareas de extracción visual rápida con proceso de aprendizaje del modelo de patrón.
Pattern editor	Haga clic y aparecerá la ventana de edición para editar la característica de forma del objeto.
Set search range	Establezca la ubicación, el tamaño y la rotación del rango para buscar el objeto.
Number of PyramidLayers	El número de iteraciones de procesamiento que se realizarán en la imagen. Más capas reducen el tiempo de procesamiento, pero para imágenes con mucho detalle, el detalle puede perderse, lo que resulta en errores de detección.
Mininum Score	El objeto solo se puede identificar cuando la puntuación de la detección es mayor que el ajuste mínimo.
Max.Num. ofObjects	El número máximo de objetos que se pueden detectar en la imagen.
Sorted by	Cuando el número máximo de objetos es mayor que 1, las salidas se ordenarán según la configuración de este campo.
Directional Edge	Seleccione si el borde de la forma es direccional.

Tabla 7. Funciones de la tarea Patten Matching (Shape).

5.4.6.2.2. Anchor

La función de Anchor establece la posición inicial y la orientación de la base del objeto. Pudiéndolo situar en la posición y con la orientación que decidamos.

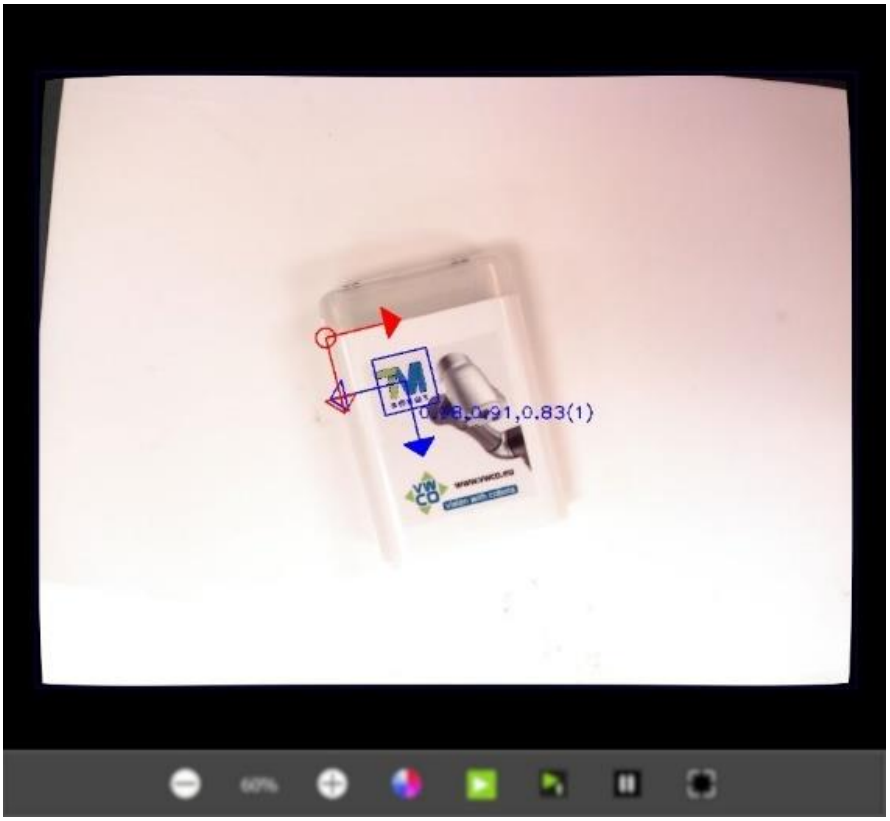


Ilustración 163. Tarea de visión Anchor en TMflow.

Nombre	Función
Manual adjustment	Arrastre manualmente el punto de anclaje a la posición de destino.
X direction shift (pixels)	Mueva el Anchor en la dirección X.
Y direction shift (pixels)	Mueva el Anchor en la dirección Y.
Rotation	Gire el ancla sobre su posición inicial.

Tabla 8. Funciones de la tarea Anchor.

5.4.6.2.3. Blob Finder

Con esta función se pueden detectar los colores que componen a los diferentes objetos. Y las funciones que posee son las siguientes.

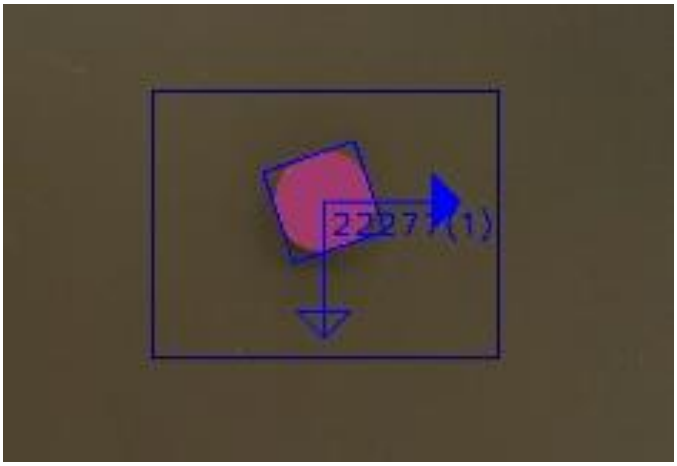


Ilustración 164. Tarea de visión Blob Finder en TMflow.

Nombre	Función
Set search range	Establecer rango de detección efectivo
Color plane	Elija el rango en el que se detecta el color
Extract color	Haga clic y adjunte el color del ROI en la imagen.
Red, green, bluePlane	Rango de distribución de color ROI
Area size	Configurar los pixeles que componen en el objeto
Max. Num. of Objects	El número máximo de objetos que se pueden detectar en la imagen.
Sorted by	Cuando el número máximo de objetos es mayor que 1, las salidas se ordenarán según la configuración de este campo.

Tabla 9. Funciones de la tarea Blob Finder.

5.4.7. Comunicaciones

Las comunicaciones con el cobot se usa el Modbus, el cual es un protocolo maestro/esclavo, el usuario puede usar Modbus maestro para leer o escribir los parámetros y guardarlos en el registro del robot, como posición, postura y estado de E/S. M Robot proporciona dos protocolos de comunicación: Modbus TCP y Modbus RTU. Para este proyecto explicaremos solo el que ha sido utilizado, es decir el Modbus TCP.

Para establecer la comunicación lo haremos mediante nodos del tipo SET, en el cual usaremos dos funciones principales "modbus_read_int32(…)" y "modbus_write(…)", con ellas se enviaran y recibirán datos del PLC. Hay que tener en cuenta que la serie TM solo posee de la posición 9000 a la 9999 con memoria libre para realizar las comunicaciones, y en ellas el robot debe de recibir los datos como tipo byte.

La programación se realiza tal y como se puede ver en las siguientes ilustraciones.

```
int[]  var_Received Order  =  modbus_read_int32 ("localhost",0,"RO",9026,2)

bool   var_Done            =  modbus_write ("localhost",0,"RO",9600,var_ OrderConfirmationEntrada)
```

Ilustración 165. Función modbus_read_int32 y modbus_write en TMflow.

5.4.8. Realización del proyecto

Para comenzar a realizar un proyecto en el que vamos a usar tareas de visión, debemos de realizar la calibración de la base sobre la que vamos a trabajar. Existen dos calibraciones, una denominada "Surface", realizada para realizar el pick de las piezas, y la otra denominada "Pallet", realizada para el despaletizado y el place de las piezas.

El proyecto está compuesto de una página principal, en la cual se realizan las comunicaciones con el PLC y las diferentes condiciones que se han de cumplir para entrar en cada Subflow. Existen catorce Subflows, trece de ellos pertenecientes a las trece posiciones del pallet y el restante al despaletizado.

El diagrama UML que corresponde a este proyecto es el que se muestra en la siguiente ilustración.

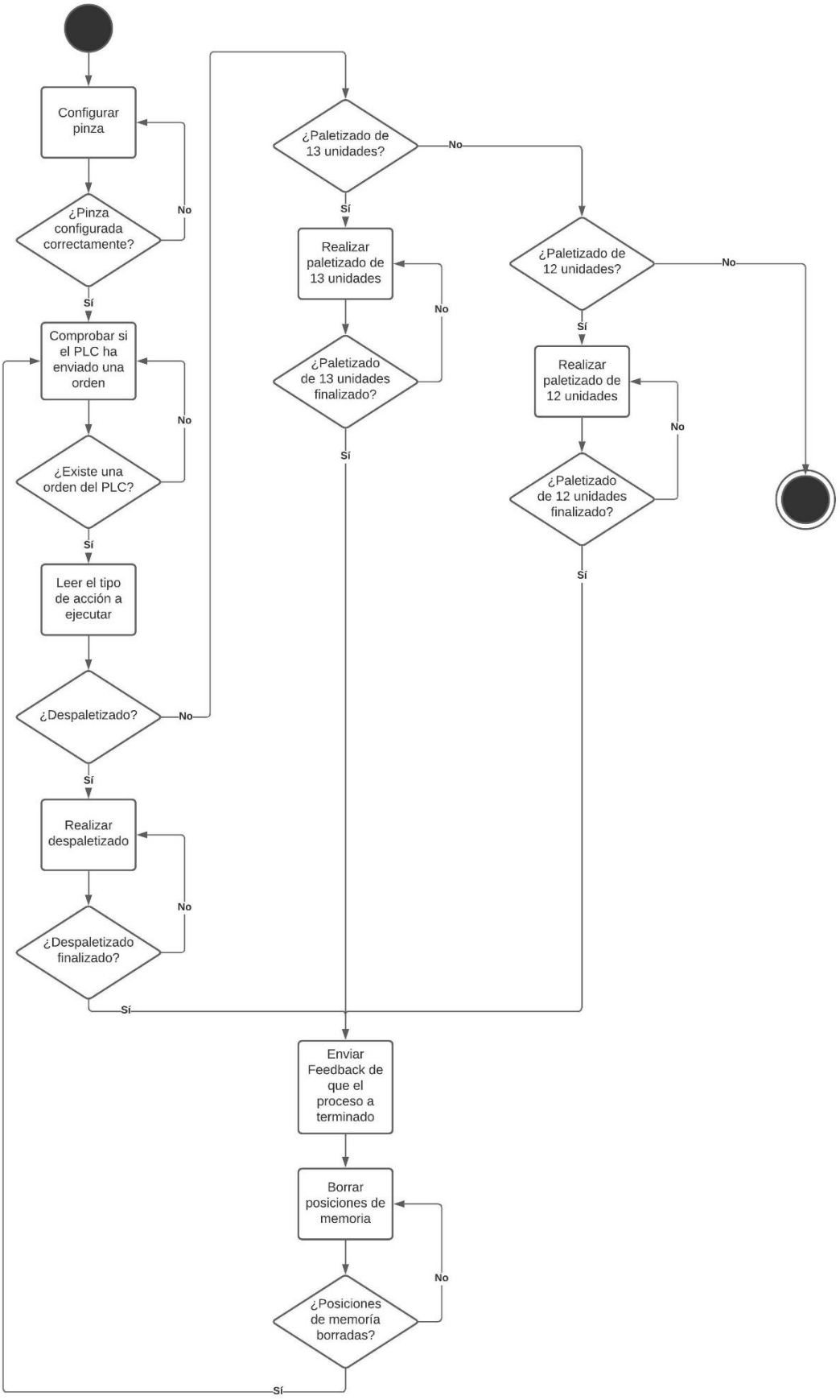


Ilustración 166. Diagrama UML del programa en TMflow.

El proyecto comienza en la página principal, comprobando el funcionamiento de la pinza y modificando las variables para abrir y cerrar la pinza de la manera adecuada, cuando esto finaliza se realiza una apertura de la pinza, para iniciar la orden que se le asigne.

Continuamos el proceso leyendo las posiciones de memoria 9026 y 9027, en las cuales se graba el valor de la orden enviada. Después realizamos una retroalimentación al PLC, escribiendo en la posición de memoria 9600 la variable "var_OrderConfirmationEntrada", para poder saber si la orden enviada es la que se ha recibido en el cobot. Y por último usamos un IF con dos condiciones, una es que la variable "var_OrderConfirmationEntrada" sea igual a 2, ya que es el valor que enviamos cuando existe alguna orden desde el PLC, y la otra es para saber que la retroalimentación se ha ejecutado correctamente, comprobando que la variable "var_Done" es igual a true. Si no se cumplen las dos condiciones, volvemos a leer las posiciones de memoria 9026 y 9027, hasta que se cumplan.

En el momento que se cumplen ponemos la variable "var_Done" a false, y leemos en la posición 9028 y 9029 el tipo de acción a ejecutar, y las posiciones que tiene el pallet si se ha seleccionado un paletizado. Sabiendo estos datos entramos en un IF, que usa la variable "var_TipoPallet[0]", para comprobar si se trata de un despaletizado. Si este nodo es true, pasamos al Subflow denominado "Despaletizado", y la salida de este Subflow nos envía al nodo "ClearOrder". Si la comparación es false, realizamos la visualización del pallet, usando la tarea de visión denominada "View_Pallet", que es una tarea del tipo AOI, en la cual buscamos las posiciones vacías que dispone el pallet.

Una vez que tenemos el número de posiciones vacías de las que disponemos y el número posiciones totales de las que dispone el pallet realizamos unas comparaciones para saber el estado del pallet, ya que si el número de posiciones vacías no es igual al número de posiciones totales realizamos un despaletizado desde el Subflow "Despaletizado", pero si la condición anterior si se cumple pasamos a realizar la tarea de visión denominada "PositionPallet", cuando el pallet es de trece posiciones, o "PositionPallet12", si el pallet es de doce posiciones. Y a su vez usamos la variable interna "var_Identificate_Pallet", para asignarle el valor de las posiciones de nuestro pallet actual. Estas dos tareas de visión son del tipo "Anchor". Y en este momento pasamos a leer la receta que ha sido seleccionada en el HMI.

Para la lectura de esta receta se leen desde la posición 9000 hasta la 9024, y se asignan de dos en dos, ya que dos posiciones de memoria son un entero, a las variables denominadas como "var_Order_PosicionXX", siendo las X los diferentes números de las posiciones.

Una vez completado este proceso se entra a trece IF en serie, en el que se realiza la comparación de que el valor de que cada posición sea diferente a cero, si esto se cumple, la variable "var_PosicionesNoRellenadas[X]" vale 0 y pasa a la siguiente posición. Sin embargo si se cumple nos envía al Subflow del paletizado correspondiente, siempre denominados como "Paletizado_PosX", y la salida de este Subflow nos envía al siguiente IF.

Una vez realizadas todas las comparaciones pasamos a la limpieza de la memoria y al envío de la confirmación de que el proceso ha finalizado, para ello mediante un goto vamos al nodo "ClearOrder", en el cual ponemos a cero desde la posición de memoria 9000 a la 9030, y modificamos el valor de la variable "var_OrderConfirmationEntrada" asignándole un cero. Confirmamos mediante un IF de que el proceso se ha realizado y le asignamos a la variable "var_RecivedOrderFinish" el valor cero.

Por ultimo solo nos queda realizar la retroalimentación de que el cobot ha finalizado el proceso, para ello leemos la posición de memoria 9030 y 9031, para saber la acción que hemos ordenado, y después escribimos en la posición 9602 el valor que acabamos de leer y de la posición 9604 hasta la 9630 las piezas que no se han podido paletizar correctamente. Realizamos la comprobación de que este proceso se ha realizado mediante un bucle de IF, y por ultimo usamos un goto para ir a al nodo "ReadOrder", para esperar la siguiente orden enviada desde el HMI.

Ahora vamos a explicar los diferentes Subflows que utilizamos. Comenzamos explicando los utilizados para el paletizado, los cuales son todos idénticos. Estos comienzan con un nodo Gateway, que posee 9 casos. En cada caso se compara el valor de la variable "var_Order_PosicionXX" con número del 1 al 9, ya que cada uno de ellos está asignado a un tipo de pieza en concreto. En el caso en el que la igualdad se cumpla se realiza la tarea de visión para localizar la pieza, esta tarea es del tipo "fixed point", creando así un punto de referencia para realzar el pick de la pieza. Si la tarea de visión no se realiza con éxito se le asigna el valor de la variable "var_Order_PosicionXX" a la variable "var_PosicionesNoRellenadas[X]", para saber la pieza que no ha sido paletizada. Si la tarea de visión se completa con éxito, se asigna un punto de place y se realiza el cerrado de la pinza.

Una vez que la pieza ya es sujeta por el cobot realizamos un movimiento vertical de 100mm para evitar chocar con las otras piezas al desplazarnos al pallet, y actualizamos la posición que tenemos que rellenar según el tipo de pallet. Realizamos el place en esta posicion mediante un punto del tipo "WayPoint", sacado con la tarea de visión "PositionPallet". Terminamos el movimiento abriendo la pinza y realizamos un movimiento vertical de 100 mm. Y por último ponemos la variable

"var_PosicionesNoRellenadas[X]" a cero, para ratificar que la posición del pallet ha sido rellenado correctamente.

Continuamos explicando el Subflow denominado "Despaletizado", el cual es el programa más complejo de todo el proyecto, ya que existen multitud de opciones que ocurren durante el despaletizado, junto con los problemas generados en las tareas de visión.

Este Subflow comienza con una tarea de visión en la cual se busca la posición del pallet, la cual se denomina "DespaletizadoPosicionPallet", y es del tipo "Anchor". Cuando el pallet es encontrado, se realiza una tarea de visión sobre su primera fila, denominada "DespaletizadoFila1" si en esta tarea de tipo "fixed point", si se encuentra algún cilindro, se procede a despaletizarlo. Para ello se va a la ubicación de él mediante un punto referenciado con la tarea de visión anterior, y se procedo a cerrar la pinza, realizar un desplazamiento vertical, después un desplazamiento al punto del place y por último una apertura de la pinza. Si en la tarea de visión se ha encontrado más de un cilindro se realiza el mismo proceso. Si no es así, se pasa al pick de los cuadrados y al de los hexágonos con el mismo procedimiento.

Una vez despaletizada la fila 1, se realiza la tarea de visión sobre la fila dos y se repite el mismo proceso. Y así sucesivamente para todas las filas.

5.5. IMPLEMENTACIÓN DEL SISTEMA FINAL

Para la implantación del sistema final se ha decidido realizar toda la demo sobre la propia base del robot, de esta manera puede ser transportado fácilmente. Para esta demo se ha usado una tabla negra sobre las que se sitúan las piezas, una caja para dejar las piezas después de despaletizarlas, y dos pizas de aluminio de doce y trece agujeros usadas como pallets.

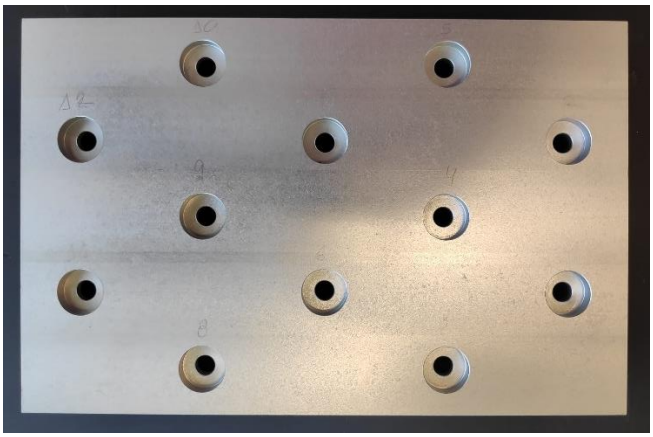


Ilustración 167. Pallet de 12 unidades.

También hemos creado tres piezas usando un programa de Autodesk, denominado tinkercad. Estas piezas se imprimieron usando una impresora 3D, y debido a que solo poseía un color, el negro, tuve que usar cartulinas recortadas para diferenciar los tres colores que se necesitaban en el proyecto.

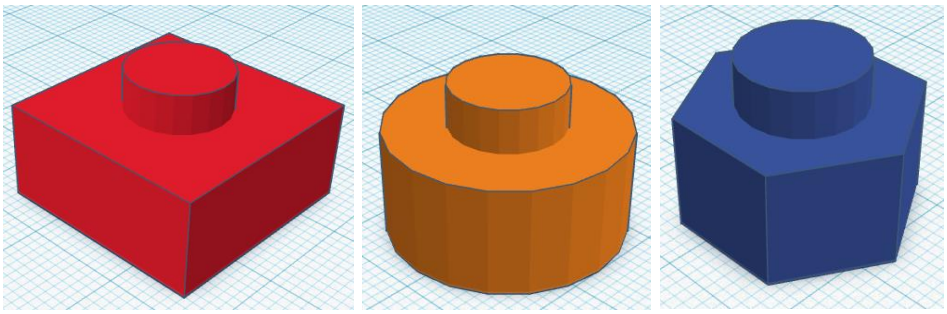


Ilustración 168. Piezas 3D en tinkercad.

Las tres piezas posen un cilindro con un diámetro de mm, para poder dejarlas sobre el pallet, y también tiene una anchura de mm, para poder usar la misma configuración de la pinza.

6. RESULTADO

En este apartado se exponen el resultado obtenido de una demo realizada aleatoriamente, en la que se realiza un paletizado de trece unidades y posterior mente el despaletizado. El pallet no se encuentra vacío en el inicio, provocando que este tenga que ser despaletizado por el cobot antes de iniciar la primera orden. Una vez vaciado el pallet se comienza el paletizado y cuando este termina enviamos la orden para realizar un despaletizado.

Comenzamos la demo, conectado el PLC con el cobot, y posteriormente iniciamos el ciclo del PLC desde la pantalla de inicio. Tal y como se ve en la siguiente ilustraciones.

Ilustración 169. Conexión DEMO.

Una vez realizados los pasos previos, seleccionamos, en la página principal, el paletizado de trece unidades.

Ilustración 170. Selección paletizado de trece unidades DEMO.

Este nos traslada a la página para seleccionar que tipo de receta queremos realizar. En este caso seleccionaremos la creación de una receta desde cero.

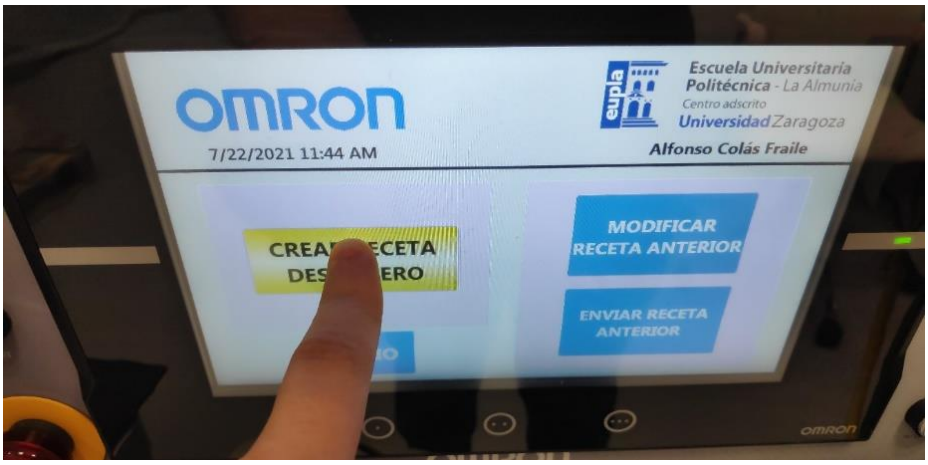


Ilustración 171. Selección tipo de receta DEMO.

En esta receta rellenaremos ocho posiciones del pallet. Para ello seleccionaremos la pieza que debe ir en cada posición.

Las piezas seleccionadas son:

- Posición 1 → Hexágono azul
- Posición 2 → Cuadrado rojo
- Posición 5 → Cilindro azul
- Posición 7 → Cuadrado verde
- Posición 8 → Cilindro verde
- Posición 10 → Cilindro rojo
- Posición 12 → Cilindro azul
- Posición 13 → Hexágono rojo

Esto se realiza usando los desplegables, y asignando en cada uno de ellos la opción que queremos. Una vez seleccionada presionamos el botón para enviar orden. Este proceso lo debemos de realizar con cada una de las posiciones que queremos paletizar.

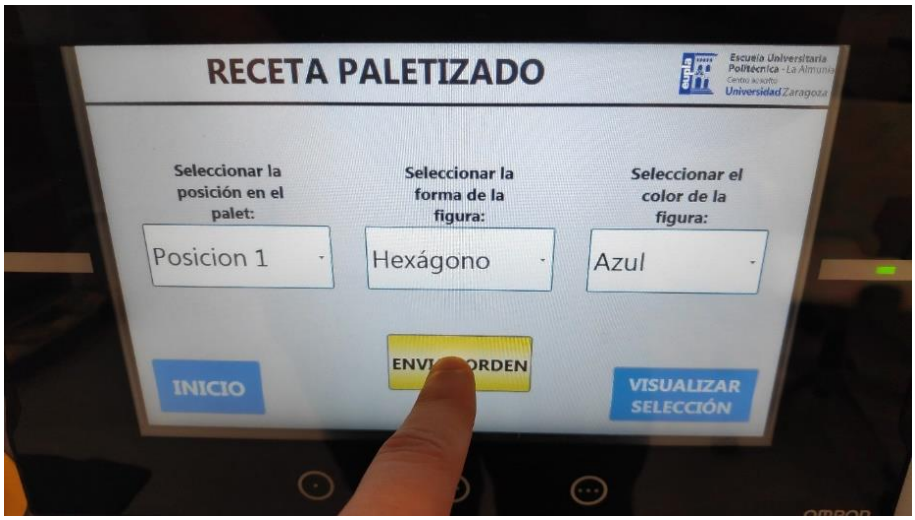


Ilustración 172. Selección Orden DEMO.

Una vez seleccionadas todas las posiciones presionamos el botón para visualizar la receta.

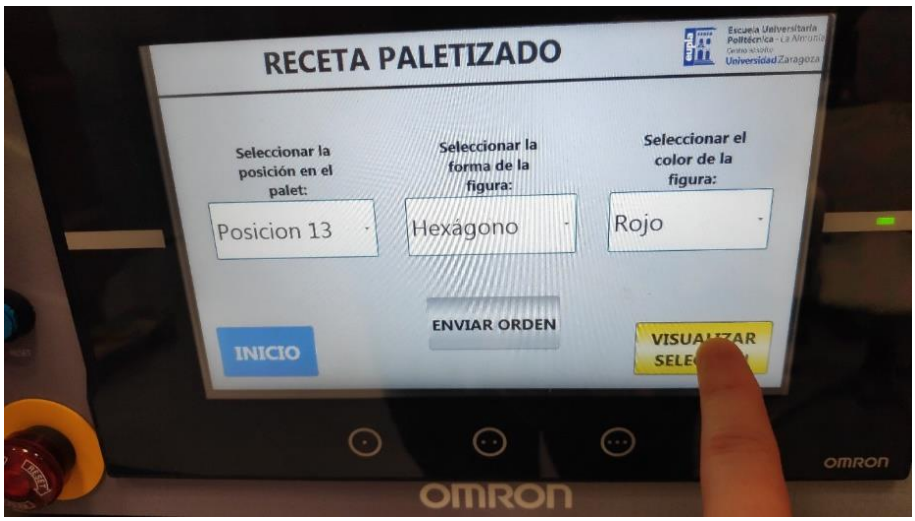


Ilustración 173. Visualizar receta DEMO.

Y comprobamos que la receta que hemos seleccionado sea la correcta. Si es así, enviamos la receta y esperamos a que el cobot realice la operación.

Ilustración 174. Enviar receta DEMO.

Primero comienza vaciando el pallet, que en este caso tenía dos piezas, un cilindro rojo en la posición 5 y un hexágono verde en la posición 8.

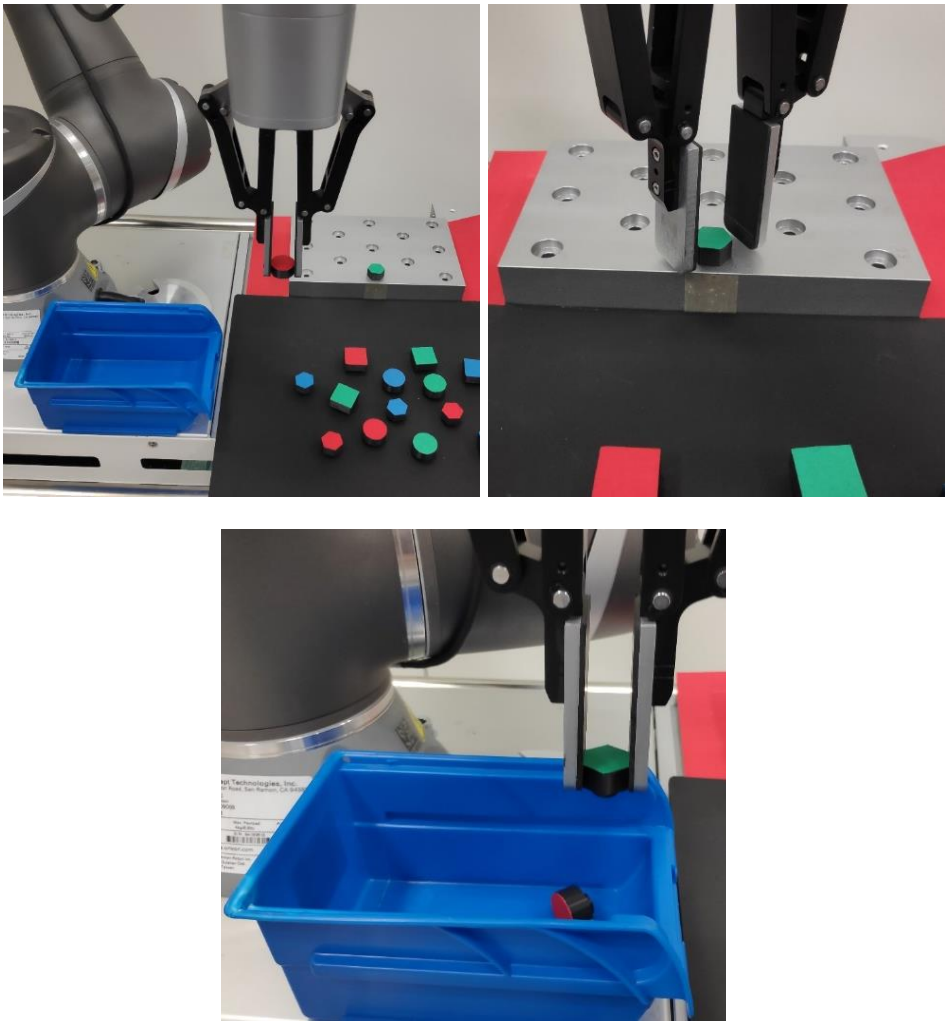


Ilustración 175. Vaciado del pallet DEMO.

Una vez que el pallet se encuentra vacío, se comienza el paletizado de la receta enviada.

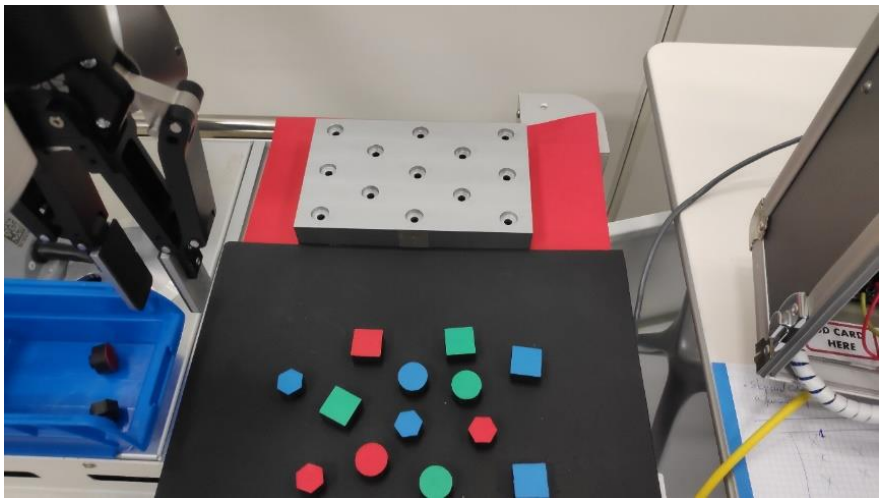


Ilustración 176. Visualización del pallet vacío DEMO.

Se comienza a paletizar por orden de posición, es decir comienza paletizando el hexágono azul, para ello realiza una foto para localizar la posición de este. A continuación se aproxima a él y se posiciona para poder realizar el pick mediante un movimiento vertical. Y realiza el mismo proceso para dejar la pieza en la posición ordenada. Tal y como se pude ver en las siguientes ilustraciones.

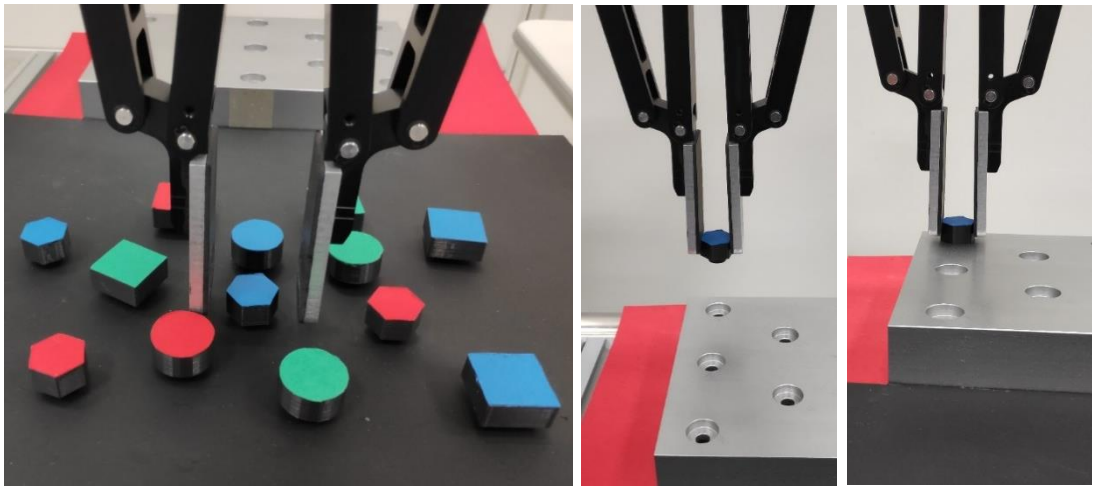


Ilustración 177. Pick and place hexágono verde DEMO.

Este proceso lo repite con todas las pizas hasta terminar la orden. Una vez finalizada debemos seleccionar el botón para visualizar la receta creada.

Ilustración 178. Paletizado terminado DEMO.

En la siguiente ilustración podemos observar que el cilindro azul no ha sido paletizado ya que no se ha encontrado. En este ejemplo solo hemos colocado un cilindro azul, para poder observar lo que ocurre cuando una pieza no se puede paletizar.

Ilustración 179. Visualización del pallet creado DEMO.

Por ultimo volvemos a la página de inicio y asignamos la opción despaletizar, y esperamos a que se realice el despaletizado.

Ilustración 180. Selección del despaletizado DEMO.

Para el despaletizado el cobot realiza una foto para comprobar el tipo de pallet y si este tiene alguna pieza. Si existe alguna pieza, para poder coger las piezas con la máxima precisión, realiza una foto a cada fila del pallet y las despaletiza por orden. Tal y como podemos ver en las siguientes imágenes.

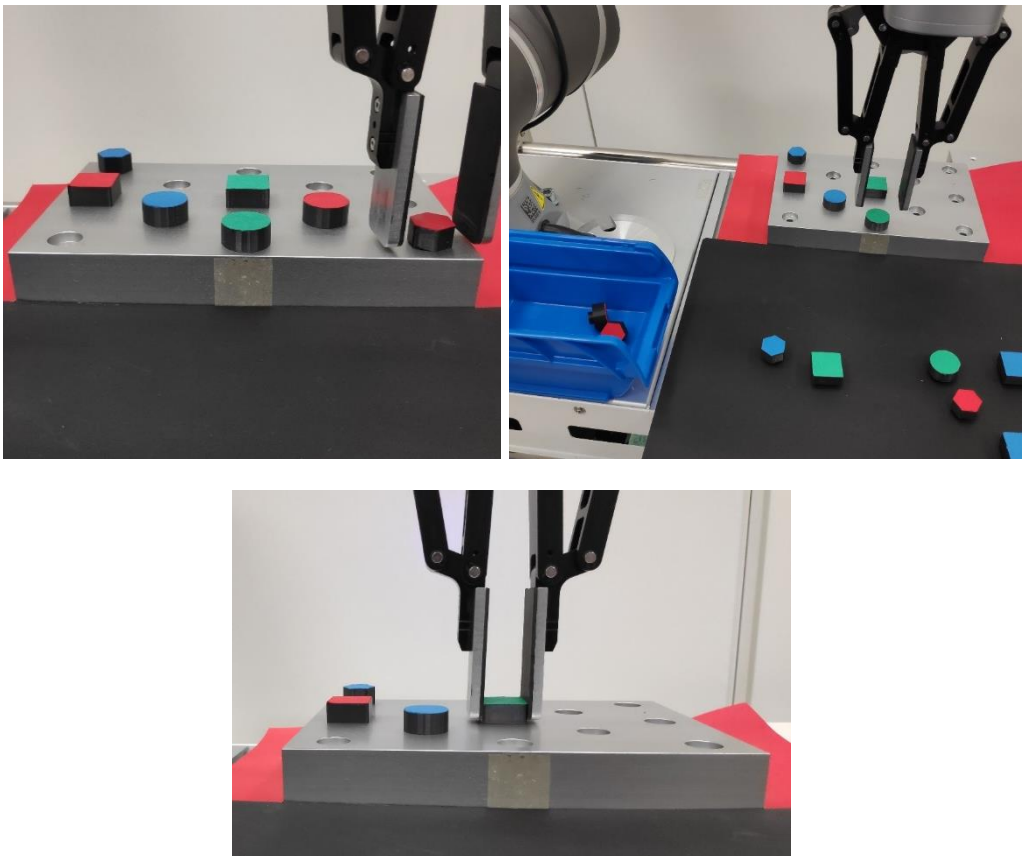


Ilustración 181. Despaletizado DEMO.

Una vez finalizado el paletizado, se nos avisa por pantalla dejando la opción de volver a la pantalla de inicio para poder seleccionar el siguiente proceso.

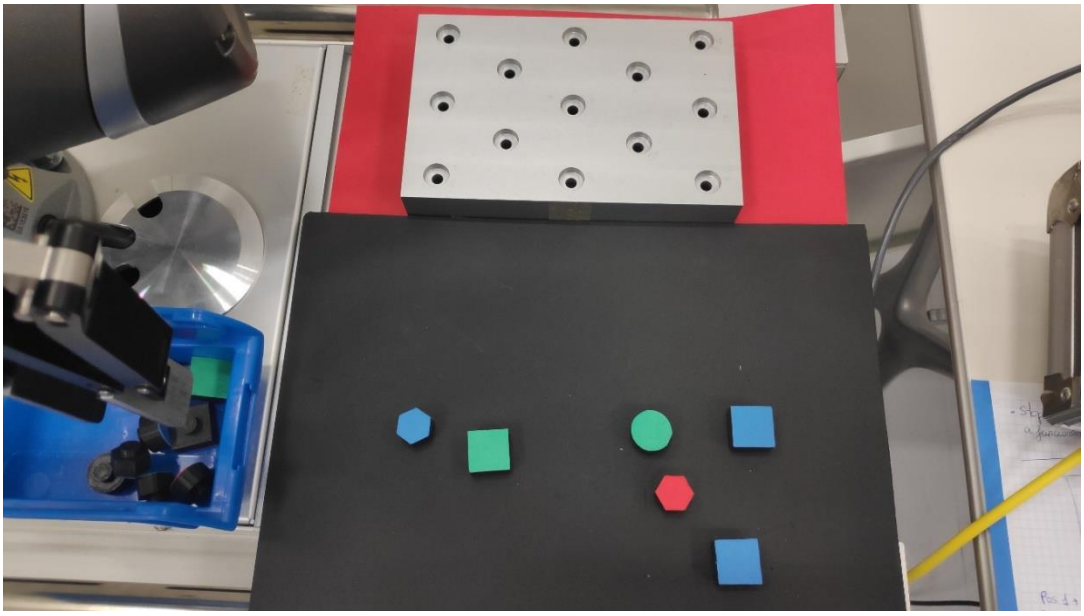


Ilustración 182. Despaletizado finalizado DEMO.

7. CONCLUSIONES

Este trabajo tenía como principal objetivo la puesta en marcha de un proceso productivo del tipo pick and place, usando un robot colaborativo.

El reto al que nos enfrentamos en este proyecto es el aprendizaje de dos software de programación, y sus respectivos lenguajes, nunca vistos antes. Junto con la dificultad de realizar el montaje físico del proyecto, ya que utilizamos dos tipos de comunicación nuevas, denominadas Ethernet/IP, para la comunicación entre el PLC y el HMI, y TCP, para la comunicación entre el PLC y el cobot.

Nos hemos encontrado numerosos problemas en la programación de todos los elementos. Por ejemplo, la transformación de datos para poder enviar las ordenes mediante enteros al cobot, la programación del HMI usando subrutinas con las que actualizar las variables continuamente, o la visión del robot, que con los cambios de luz genera una gran cantidad de problemas, tanto en la detección de formas como en la detección de colores.

Pero como hemos podido ver en el apartado anterior, el montaje de la célula de producción se ha realizado correctamente, con una gran similitud a lo que se realizaría en un proceso productivo real.

Por último, decir que esta pequeña demo puede ser la base para poner en funcionamiento un proceso industrial real, ya que este tipo de aplicaciones son cada vez más comunes. Y el uso de la visión y de la robótica colaborativa son el futuro de la industria.

8. BIBLIOGRAFÍA

[1] «Qué es la cuarta revolución industrial (y por qué debería preocuparnos)», *BBC News Mundo*. Accedido: may 20, 2021. [En línea]. Disponible en: https://www.bbc.com/mundo/noticias-37631834

[2] «Sistema de Fabricación Flexible (FMS) Término Definición». https://www.manufacturingterms.com/Spanish/Flexible-Manufacturing-System-(FMS).html (accedido may 20, 2021).

[3] «Informe-CODDII-Industria-4.0.pdf». Accedido: may 20, 2021. [En línea]. Disponible en: http://coddii.org/wp-content/uploads/2016/10/Informe-CODDII-Industria-4.0.pdf

[4] «Industria 4 0, Automatización E Innovación Infographic Ilustración del Vector - Ilustración de teamwork, ingenieros: 123223413», *Dreamstime*. https://es.dreamstime.com/industria-automatización-e-innovación-infographic-image123223413 (accedido may 26, 2021).

[5] «URL_03_MEC02.pdf». Accedido: may 20, 2021. [En línea]. Disponible en: http://www.fgsalazar.net/LANDIVAR/ING-PRIMERO/boletin03/URL_03_MEC02.pdf

[6] C. Pérez, «Controladores Lógicos Programables (PLCs)», p. 21.

[7] «Título: NX1 | OMRON, España». https://www.google.com/imgres (accedido may 26, 2021).

[8] «Qué es un HMI: para qué sirve la Interfaz Hombre-Máquina | Aula21», *aula21 | Formación para la Industria*, jun. 10, 2019. http://www.cursosaula21.com/que-es-un-hmi/ (accedido may 25, 2021).

[9] «Serie NA». https://industrial.omron.es/es/products/na (accedido may 26, 2021).

[10] «Implantación de robot colaborativo en linea», p. 126.

[11] «Robots colaborativos». https://industrial.omron.es/es/products/collaborative-robots (accedido may 26, 2021).

[12] «cobot_safety_expert_article_en.pdf». Accedido: may 25, 2021. [En línea]. Disponible en: https://assets.omron.eu/downloads/publication/en/v2/cobot_safety_expert_article_en.pdf

[13] «i836_tm_collaborative_robot_brochure_es.pdf». Accedido: may 26, 2021. [En línea]. Disponible en: https://www.tecnical.cat/PDF/OMRON/Robotics/TM/DataSheet/i836_tm_collaborative_robot_brochure_es.pdf

[14] C. A. S. Serna y L. C. C. Ortiz, «Buses de campo y protocolos en redes industriales», *Ventana Informática*, n.º 25, Art. n.º 25, 2011, doi: 10.30554/ventanainform.25.126.2011.

[15] V. G. Jimenez, R. Y. Yuste, y L. Martínez, *Comunicaciones Industriales Siemens*. Marcombo, 2012.

[16] J. G. Mejía Arango, *Introducción a las comunicaciones industriales con Profibus: aplicaciones con controladores lógicos programables y variadores de velocidad*. Medellín, Colombia: Instituto Tecnológico Metropolitano, 2009.

[17] «Profinet, Industrial Ethernet for advanced manufacturing», *PI Norte América*. https://us.profinet.com/tecnologia/profinet-es/ (accedido may 25, 2021).

[18] D. García, «Profibus cumple 30 años mirando al futuro - infoPLC». https://www.infoplc.net/noticias/item/107411-profibus-cumple-30-anos (accedido may 26, 2021).

[19] «Ethernet/IP - Protocolo de red en niveles para aplicaciones de automatización industrial», *Default*. https://www.siemon.com/es/home/support/education/white-papers/03-10-13-ethernet-ip (accedido may 25, 2021).

[20] Walter, «Funcionalidad IIoT: EtherNet/IP y PROFINET». http://www.edcontrol.com/index.php/instrumentacion/instrumentacion-194/item/280-funcionalidad-iiot-ethernet-ip-y-profinet (accedido may 25, 2021).

[21] «Sistemas de visión artificial: tipos y aplicaciones», *[R]evolución artificial*, oct. 28, 2019. https://blog.infaimon.com/sistemas-de-vision-artificial-tipos-aplicaciones/ (accedido may 25, 2021).

[22] «Sistemas de inspección | OMRON, España». https://industrial.omron.es/es/products/inspection-systems (accedido may 26, 2021).

[23] «omron tm - Búsqueda de Google». https://www.google.com/search?q=omron+tm&sxsrf=ALeKk02j8uWgJP0FAeeca3AumXmqSi7_iA:1622064894195&source=lnms&tbm=isch&sa=X&ved=2ahUKEwirssjOpujwAhWE2aQKHeV-DF8Q_AUoAXoECAEQAw#imgrc=rcOp1F2ye14j-M (accedido may 26, 2021).

[24] «NX102-[][][][] NX-Series NX1 CPU Units/Features | OMRON Industrial Automation». http://www.ia.omron.com/products/family/3705/ (accedido jun. 18, 2021).

[25] «NX102-[][][][] NX-Series NX1 CPU Units/Features | OMRON Industrial Automation». http://www.ia.omron.com/products/family/3705/ (accedido may 25, 2021).

[26] «Sysmac Studio». https://industrial.omron.es/es/products/sysmac-studio (accedido jun. 03, 2021).

[27] «Sysmac_Studio_Ver1_18_OperationManual_en_201704_W504-E1-20.pdf». Accedido: jun. 03, 2021. [En línea]. Disponible en: http://products.omron.us/Asset/Sysmac_Studio_Ver1_18_OperationManual_en_201704_W504-E1-20.pdf

[28] «NA Series Programmable Terminal/Features | OMRON Industrial Automation». http://www.ia.omron.com/products/family/3405/ (accedido may 25, 2021).

[29] «NA-series Programmable Terminal Software User's Manual», p. 260.

[30] «Sysmac studio HMI - Búsqueda de Google». https://www.google.com/search?q=Sysmac+studio+HMI&hl=es&sxsrf=ALeKk00WCS7Lmzubn5nFfD3jW5xRfXGgSg:1624012927952&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjehbnN_6DxAhWaFMAKHS__BGMQ_AUoAXoECAEQAw&biw=1920&bih=969#imgrc=DADICPRCEJgh0M (accedido jun. 18, 2021).

[31] «i623_collaborative_robots_hardware_installation_manual_en.pdf». Accedido: jun. 11, 2021. [En línea]. Disponible en: https://assets.omron.eu/downloads/manual/en/v2/i623_collaborative_robots_hardware_installation_manual_en.pdf

[32] «TM Series Collaborative Robots/Features | OMRON Industrial Automation». http://www.ia.omron.com/products/family/3739/ (accedido may 25, 2021).

[33] «El robot colaborativo de OMRON - TM series», *Tecnología en Electrónica y Control S.R.L.*, oct. 22, 2019. https://www.tecsc.com.ar/empresa/noticias/el-robot-colaborativo-de-omron-tm-series/ (accedido may 26, 2021).

[34] «Omron Mobile Manipulator Solution». https://industrial.omron.eu/en/solutions/product-solutions/omron-mobile-manipulator-solution (accedido may 26, 2021).

[35] «Slightly whimsical but fully capable Hybrid Cobot», *MRK-Blog.de*, dic. 09, 2019. https://mrk-blog.de/en/leicht-skurriler-aber-voll-tauglicher-hybrid-cobot/ (accedido may 26, 2021).

[36] «Robots colaborativos | OMRON, España». https://industrial.omron.es/es/products/collaborative-robots (accedido may 25, 2021).

[37] «Software Manual TMflow», p. 358.

[38] «TMflow», *Techman Robot*. https://cobot.mx/software/tmflow/ (accedido jun. 11, 2021).

[39] «Robotiq 2F-140 | Collaborative Robot Gripper | Cobot Webshop», *WiredWorkers*. https://wiredworkers.io/product/robotiq-2f-140/ (accedido jun. 18, 2021).

[40] «Robotiq 2F-140 | Collaborative Robot Gripper | Cobot Webshop», *WiredWorkers*. https://wiredworkers.io/product/robotiq-2f-140/ (accedido may 26, 2021).

[41] «NJ/NX-series Instructions Reference Manual», p. 1626.

[42] «tm_flow_software_manual_installation_manual_en.pdf». Accedido: jun. 29, 2021. [En línea]. Disponible en: https://assets.omron.eu/downloads/manual/en/v1/tm_flow_software_manual_installation_manual_en.pdf

Relación de documentos

(X) Memoria	155	páginas
(_) Anexos	200	páginas

La Almunia, a 18 de Septiembre de 2021

Firmado: Alfonso Colás Fraile