Universidad del País Vasco Euskal Herriko Unibertsitatea

BILBOKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE BILBAO

MÁSTER UNIVERSITARIO EN

INGIENERÍA INDUSTRIAL

TRABAJO FIN DE MÁSTER

DISEÑO DEL GEMELO DIGITAL DE CÉLULA ROBOTIZADA PARA SISTEMA DE CONTROL MULTIAGENTE Y PUESTA EN MARCHA VIRTUAL Y REAL

Estudiante	*Artetxe, Lázaro, Eneko*
Director/Directora	*Orive, Revilla, Darío*
Departamento	*Ingeniería de sistemas y automática*
Curso académico	*2020/2021*

Bilbao, 21, septiembre, 2021

Resumen

Los gemelos digitales son uno de los componentes de la llamada Industria 4.0, modelos que no son más que réplicas virtuales de procesos que simulan el comportamiento del sistema original. Este proyecto tiene como objetivo modelar un gemelo digital de una célula robotizada, así como el análisis y simulación. Validar los proyectos de automatización y programa robot para la implementación real. Asimismo, se detalla la realización del gemelo digital, describiendo paso a paso el modelado y la lógica introducida. Por último, se demuestra que el proyecto cumple todos los objetivos establecidos, siendo posible su aplicación en la industria. La posible implementación de este tipo de tecnologías permitirá acortar los plazos de puesta en marcha, resultando así en una reducción de los costes asociados a esta.

Palabras clave: Gemelo Digital, puesta en marcha virtual, simulación, automatización, modelado, Tecnomatix PS, puesta en marcha.

Laburpena

Biki digitalak 4.0 Industria delakoaren osagarrietako bat dira, jatorrizko sistema baten portaera simulatzen duten prozesuen erreplika birtualak. Proiektu honen helburu nagusiak honakoak ziren: zelula robotikoaren biki digital baten eredua burutzea, bai eta modelo horretan analisia eta simulazioa egitea. Automatizazio eta robot programaren balioztapena lortu makina errealan inplementatzeko. Halaber, biki digitalaren gauzapena zehaztatzen da, pausoz pauso deskribatuz bertan barruratutako eredu-egitea eta logika. Azkenik, baieztatzen da proiektuak hasieran ezarritako helburuak betetzen dituela, bere industriarako aplikazioa ahalbidetu lezakeelarik. Gainera, teknologia mota hauen balizko inplementatzeak abiarazpen-prozesuaren epeak laburtuko ditu; horrenbestez, berari lotutako kostuak murriztuz.

Hitz-gakoak: Biki Digitala, abiarazpen birtuala, simulazioa, automatizazio eredu-egitea, Tecnomatix PS, abiaraztea.

Abstract

Digital twins are one of the components belonging to the so-called Industry 4.0, which are nothing but virtual replicas of processes that simulate the behavior of the original system. The main objectives of this project were not only to model the digital twin of a robotic cell, but also to analyze and simulate it. The simulation is going to validate de automatization project and the robot program to then implement it in a real machine. Moreover, the execution of the digital twin itself is detailed, describing step by step the modelling and the logic introduced. Lastly, it is shown that the project achieves the objectives stated above, thus being allowed its feasible implementation in the industry. Therefore, the use of this technologies will permit to shorten the deadline of the launching process, turning out into a reduction of the associated costs.

Keywords: Digital Twin, virtual commissioning, simulation, automatization, modelling, Tecnomatix PS, start up.

Índice de contenidos

Índice de figuras

Índice de tablas

Lista de acrónimos

CAD: Computer Aided Design (Diseño asistido por ordenador)

CPU: Central Processing Unit (Unidad central de procesado)

FB: Function Block (Bloque de función)

FC: Function (Función)

HIL: Hardware in the Loop

HMI: Human Machine Interface (Interfaz humano-máquina)

IO: In & Out (Entradas y salidas)

IP: Internet Protocol (Protocolo de internet)

PS: Process Simulate

PLC: Programable Logic Controller (Controlador lógico programable)

SIL: Software in the Loop

ODK: Open Development Kit

AGV: Automated Guided Vehicle

1 Introducción

En este documento se presenta el desarrollo del diseño, validación e implementación de una célula compuesta por dos estaciones de montaje robotizadas. Cada estación debe completar diferentes solicitudes provenientes de un agente externo. Las piezas constan de cuatro elementos diferentes, y las estaciones deben permitir tanto el ensamblaje de cualquier elemento o elementos en los servicios, como que cada servicio contenga diferente número de ítems. De esta manera se busca dotar a las estaciones de modularidad y flexibilidad.

Para llevar a cabo esta implementación real, previamente se llevará a cabo un estudio en simulación de las estaciones para validar las soluciones diseñadas mediante *virtual commissioning.* Por tanto, es necesario desarrollar un gemelo digital de las estaciones para este fin. De esta forma se puede diseñar, estudiar y corregir diferentes aspectos del proceso. Algunos ejemplos pueden ser estudios de alcance, código robot, posiciones de robot, trayectorias o colisiones entre otras.

Durante el proyecto se lleva a cabo el desarrollo tanto del proyecto de automatización, como el programa robot. En el proyecto se cuenta con varios PLC, por tanto, es necesario realizar una comunicación entre estos PLC para sincronizar la ejecución. Estas comunicaciones están basadas en TCP/IP. El proyecto es un demostrador para probar en simulación y con estaciones reales la tecnología multiagentes que se desarrolla dentro del grupo de investigación.

Los proyectos de automatización y programa robot se validan mediante el comentado anteriormente *virtual commissioning.* Esta validación se puede realizar de diferentes formas según el software y hardware utilizados. En este proyecto se ha realiza mediante *Software in the loop (SiL).* Es decir, todas las simulaciones se realizan mediante software, con un controlador simulado, sin usar ningún hardware específico. Por tanto, no es necesario el uso de hardware extra. Una vez validados, se han volcado los programas creados a sus respectivos hardware para la puesta en marcha real. El desarrollo de este proyecto se ha llevado a cabo en el laboratorio del Departamento Ingeniería de Sistemas y Automática de la Escuela Técnica Superior de Ingeniería de Bilbao. Donde se encuentran los equipos, estaciones y software necesarios para la realización del proyecto.

2 Contexto

Debido a la digitalización dentro de las fábricas, la combinación de tecnologías basadas en Internet y tecnologías prospectivas en el campo de los objetos "inteligentes" (máquinas y productos), se está llevando a cabo un cambio de paradigma fundamental de la producción industrial. La nueva visión en torno al futuro de la producción engloba sistemas de manufactura modulares y eficientes, y describe escenarios en los que los procesos de producción reciben una mayor automatización. Así, se pretende realizar la manufactura de productos individuales conocidos como *batch size of one* (lote de una unidad), manteniendo las ventajas económicas de la producción en masa. Debido a esta expectativa, el término "Industria 4.0" fue acuñado con la llegada de la "cuarta revolución industrial", siendo este término una reminiscencia del versionado de software.

La primera revolución Industrial (figura 1) fue un proceso de transformación económica, social y tecnológica que se inició en la segunda mitad del siglo XVIII en Gran Bretaña. Extendiéndose después a parte de Europa occidental y América anglosajona hasta la primera mitad del siglo XIX. La invención de la máquina de vapor sustituyo la mano de obra basada en el trabajo manual y tracción animal por maquinaria para la fabricación industrial. También trajo consigo un cambio en el transporte con la creación del ferrocarril y los barcos. Esto transformo la sociedad a la creación de grandes núcleos urbanos basados en la industria y la mecanización. Surge así la producción en masa y el consiguiente modo de consumo asociado.

La segunda revolución, en el siglo XIX, surge de la utilización de nuevas fuentes de energía como el petróleo o gas, y las primeras centrales eléctricas. La creación de la industria automovilística introdujo un nuevo método de producción conocido como producción en línea de ensamblaje. Durante esta época surgieron nuevos métodos de comunicación como la radio o el cine supusieron un cambio de mentalidad respecto a la sociedad y el consumo.

A finales del siglo XX surge la tercera revolución industrial basada en el desarrollo de la automatización y las tecnologías de la información. Los procesos de miniaturización, la aparición de los micros, tecnologías de la información, el software y la introducción de la robótica y máquinas de montaje transformaron la industria. Además, la invención de internet como nuevo canal de comunicación impactó de forma importante a la sociedad.

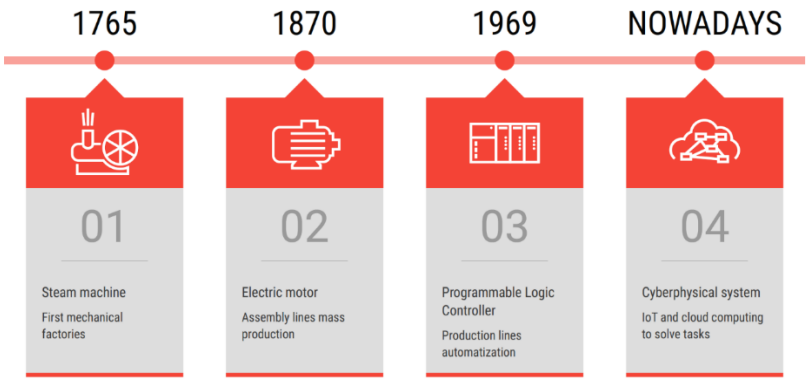


Figura 1. Las 4 revoluciones industriales.

La idea principal de esta nueva industria 4.0 es explotar el potencial de las nuevas tecnologías (figura 2) y conceptos, tales como:

- disponibilidad y uso del internet de las cosas (*Internet of Things - IoT*),
- mapeado digital y virtualización del mundo real, es decir, sistemas ciber-físicos (*cyber-physical systems - CPS*),
- empresa "inteligente", incluyendo recursos inteligentes de la producción industrial y productos "inteligentes".

La cada vez mayor competitividad existente en el mercado global actual exige minimizar los costes a toda empresa que pretenda mantener su posición. Precisamente, la implementación de estas tecnologías puede ayudar a reducir significativamente los costes en la producción industrial. De hecho, la "industria 4.0" parece ser una solución, ya que, según varias fuentes, puede provocar la reducción de los costes de producción y logísticos en un 10-30 %, y los costes de gestión de calidad en torno a un 10-20 % [1].

Figura 2. Tecnologías industria 4.0.

Tal y como se ha mencionado anteriormente, uno de los componentes de la nueva industria son los sistemas de simulación, entre los que se incluye el gemelo digital. Según la definición propuesta por Söderberg et al. (2017) [2], un gemelo digital consiste en una copia digital de un sistema físico para la realización de una optimización en tiempo real. En otras palabras, esta tecnología no es más que la generación de una réplica virtual de un producto, servicio o proceso que simula el comportamiento de su homólogo físico. Esto permite analizar su reacción ante determinados escenarios y mejorar su rendimiento y eficacia. Además, es posible experimentar sin correr riesgos, característica muy beneficiosa para el diseño de los procesos de fabricación.

Hoy en día, debido a la cuarta revolución industrial y sus avances en computación masiva de datos (big data), computación en la nube (cloud computing) o el internet de las cosas (internet of things, IoT), se han logrado nuevas posibilidades en este campo.

Una de esas oportunidades es la puesta en marcha virtual en el Gemelo Digital (*virtual commissioning*), la cual es una estrategia que permite modelar un proceso de manufactura en un entorno virtual, y, por tanto, sin necesidad de tener sistema físico durante la fase de desarrollo. Otra de las ventajas es el hecho de que ingenieros de diferentes campos tengan un modelo común con el que todos sean capaces de trabajar. Además, la puesta en marcha virtual posibilita la fácil reconfiguración de un

sistema ya existente, ya que pueden realizarse cambios en el modelo digital del sistema a nivel de proceso, software o hardware [3]. Esto permite, a su vez, programar el sistema antes de que el hardware físico esté instalado, siendo el objetivo la detección temprana y corrección de los errores generados durante la planificación, el diseño y la programación. De hecho, a lo largo del desarrollo del proyecto, es posible analizar cualquier fallo para así evitarlo en la puesta en marcha del sistema físico. Por tanto, el *virtual commissioning* puede ser una herramienta muy útil a la hora de validar esta fase del proyecto. En efecto, estudios experimentales muestran tanto el efecto positivo de la puesta en marcha virtual en la tasa de error de la puesta en marcha real [4], como la reducción de hasta un 75 % en el tiempo necesario para la puesta en marcha de una planta [5]. (figura 3)

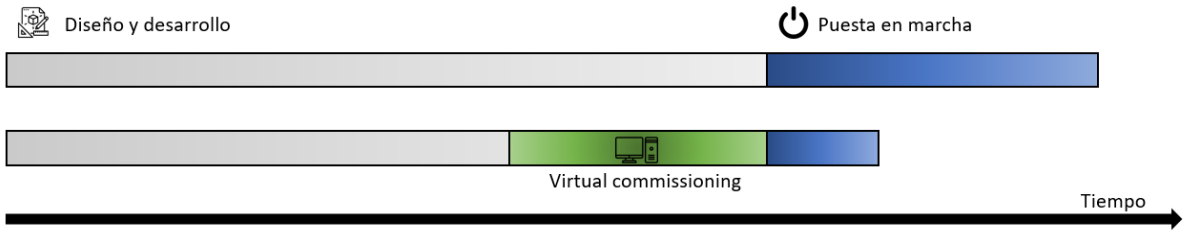


Figura 3. Reducción de puesta en marcha real mediante *virtual commissioning*.

Para poder integrar todas estas herramientas es necesario recurrir a las comunicaciones entre tecnologías. Esto ha llevado a la necesidad de romper las grandes islas de las compañías más importantes tenían en el pasado. También ha traído el desarrollo de diferentes tecnologías de procesamiento como la *cloud* o *fog* (figura 4).

La computación en la nube o *Cloud Computing* hace referencia a la tecnología que posibilita la capacidad computacional y el uso de una gran cantidad de servicios, archivos e información desde Internet. Se trata de una tecnología flexible y escalable, que libera del almacenamiento de toda esta información en dispositivos locales. Estos servicios se ubican en plataformas de los proveedores de servicio. No son aptas para necesidades de tiempo real por su dependencia de internet. Sin embargo, son aptos para el tratamiento de grandes cantidades de datos sin necesidad de tener infraestructura para ello.

El *Fog Computing* o computación en la niebla hace así referencia a la infraestructura de computación cercana a las fuentes de datos. El *Fog Computing* permite que un único dispositivo, denominado comúnmente pasarela, procese los datos que provienen de múltiples fuentes. Con esta computación se reduce la latencia, ya que no es necesaria la conexión a la nube. Con esto se consigue, además,

reducir la cantidad de datos que se envían a la nube, un aspecto muy relevante si tenemos en cuenta que las cantidades de datos a procesar no dejarán de incrementarse en los próximos años.

La cercanía con los dispositivos permite solventar defectos del *Cloud Computing* para el control en tiempo real. Además, no hay que acudir a proveedores externos ya que está dentro de la red local del usuario, por lo que aumenta la seguridad de los datos.

El *Edge Computing* permite la recopilación, procesamiento y tratamiento de datos en un dispositivo inteligente, cerca del lugar donde se generan los datos, es decir, sin necesidad de llevarlos a la nube o *fog*. En este caso, se conectan los sensores a los controladores de automatización programables, lo que reduce los niveles de la comunicación, ya que analizan y procesan la información recibida desde el dispositivo. Esto se lleva a cabo en el mismo punto de generación de los datos y sólo envía a la nube los datos que deben ser almacenados consiguiendo así ser más eficiente. Esta computación es la que se vincula con el *IoT.*

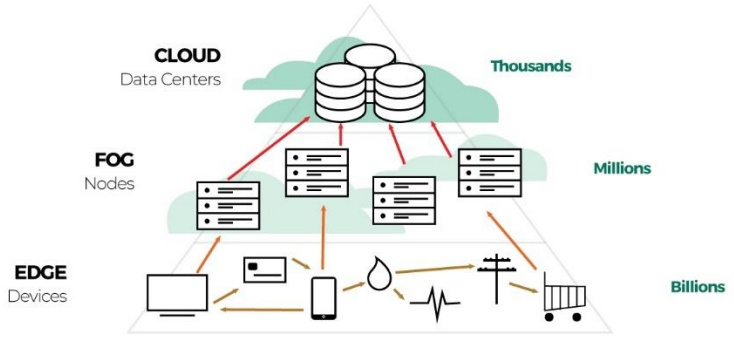


Figura 4. Piramide de automatización *Edge*, *Fog* y *Cloud*.

Estas comunicaciones posibilitan la integración de sistemas multiagente. Estos sistemas están compuestos por diferentes agentes inteligentes que interactúan entre ellos. De esta manera es posible resolver problemas que son difíciles o imposibles para un único agente [6]. Junto a los dispositivos de campo, a nivel *Edge*, se colocan agentes máquina que comunican sus estados a agentes en el *fog* los cuales realizan la planificación de la producción. Este agente maquina comunican la automatización clásica de un PLC tradicional en el dispositivo, con la conexión a la red de sistemas multiagente. Esto crea un entramado de control distribuido descentralizado inteligente. Además, estos sistemas son fácilmente escalables y flexibles.

3 Objetivos y alcance

Este proyecto tiene dos principales objetivos. El primero de ellos, se trata de modelar un Gemelo Digital de una célula robotizada flexible por dos estaciones robotizadas que realizan el montaje de un conjunto total o parcialmente. El montaje consta de varios servicios con todas las posibilidades de entrada y salida del conjunto. También se deberá modelar un AGV (*Auto Guided Vehicle*) o AMR (*Autonomous Mobile Robots*) que realice el transporte entre las estaciones y la entrada y salida del conjunto. Posteriormente se realizará una validación en simulación del control mediante simulación sobre este Gemelo Digital realizando una pueta en marcha virtual distribuida.

El segundo objetivo principal, es realizar la puesta en marcha de una estación robotizada en la estación real del laboratorio. Dado que solo se cuenta con una estación robotizada únicamente se realizará la puesta en marcha de una y la validación tanto del código robot como de control correspondiente a esta estación. En la puesta en marcha la estación debe completar los mismos servicios que en simulación

3.1 Gemelo Digital

El modelo del Gemelo Digital se basa en la estación ubicada en el laboratorio con el *layout* ya diseñado para proyectos anteriores. Sobre ese modelo se deben realizar las trayectorias necesarias para realizar el montaje de los servicios del conjunto. Estos diferentes servicios son los que otorgan la flexibilidad a la célula. Los conjuntos constan de diferentes piezas: base, rodamiento, bulón, tapa interior y tapa exterior y están basados en la célula modular flexible FMS 200. El orden de montaje de los conjuntos es el mismo en el que se han mencionado las piezas. Un servicio puede contener de uno a seis ítems y todos llegan con las mismas piezas y se realiza el mismo montaje en cada una de ellas. El servicio siempre consta de al menos la base a la entrada y siempre ha de montarse al menos una pieza más en cada posición.

Para realizar el transporte de conjuntos fuera de la estación, se modela también el mencionado AGV que consta de una plataforma que se puede elevar y bajar para el acceso del robot y el transporte respectivamente. El AGV tiene cuatro posiciones de parada, una en cada estación y dos que simulan almacenes, uno de entrada y otro de salida.

En el futuro la célula podría ampliar sus servicios a nuevos conjuntos. Por ejemplo, se podrían añadir conjuntos más pequeños que forman parte de la célula FMS 200. Otro de futuros objetivos puede ser añadir más flexibilidad al montaje pudiendo realizar diferentes montajes en cada ítem de un lote.

3.2 Proyecto de control y puesta en marcha virtual distribuida

Cada estación cuenta con una unidad de control y un PLC para realizar el control. Por tanto, se debe desarrollar un código con las instrucciones robot en el lenguaje de programación del robot. Este programa contiene las trayectorias, apertura y cierre de pinzas, y control de señales y de flujo.

En el Programa de control PLC se debe realizar la conexión con el sistema multiagente que controla la planificación de los conjuntos. Además de comunicarse con el resto de PLC de la célula para realizar la lógica según el estado de las estaciones o el AGV. Para realizar la conexión de las estaciones con el sistema multiagente se debe incorporar al programa del PLC el código ODK (*Open Development Kit*) desarrollado por compañeros del grupo de investigación. Las comunicaciones entre los tres PLC se van a realizar mediante TCP/IP.

Además, durante la simulación el programa PLC debe controlar el flujo de material de la simulación con las señales específicas para ello. Más tarde durante la implantación en la estación real este código será desechado puesto que solo es necesario en simulación.

Una vez realizado los proyectos de control y los programas robot, se va a realizar la validación mediante la puesta en marcha virtual distribuida. Mediante esta puesta en marcha se simula el comportamiento del modelo para comprobar el correcto funcionamiento del código desarrollado. La puesta en marcha se va a realizar de manera distribuida, es decir, los programas necesarios para la simulación van a ejecutarse en diferentes PC para dividir el esfuerzo de computación necesario para llevar a cabo la puesta en marcha virtual.

3.3 Puesta en marcha real

Tras validar el código de la puesta en marcha virtual, se va a realizar el volcado de ese código al *hardware* real de la estación del laboratorio. En esta puesta en marcha por las limitaciones del material disponible en el laboratorio, solo se va a realizar la puesta en marcha de una estación sin AGV. Por lo tanto, tampoco se va a realizar comunicaciones TCP/IP. Sin embargo, la conexión con el sistema multiagente si será implantado en la estación real. Tras el volcado, primero se realiza una comprobación del código robot para encontrar y solucionar los posibles fallos que puedan surgir en los programas robot en modo manual. Una vez comprobado que no hay errores en el código robot se lleva a cabo la ejecución de varios servicios integrado dentro del sistema multiagente.

4 Beneficios del proyecto

4.1 Beneficios económicos

En la puesta en marcha tradicional la detección y corrección de errores se realiza durante esta cuando se cuenta con los equipos en destino. Sin embargo, al poder realizar esta detección y error durante el desarrollo y sin necesidad de contar con los equipos, permite reducir los tiempos de puesta en marcha. Desarrollar el código mediante validación por *virtual commissioning* permite llegar a la puesta en marcha con una versión mucho más depurada del *software* desarrollado. En definitiva, se reduce los tiempos de inactividad de la producción con la reducción de los costes que ello implica.

Pese a lo costoso de la tecnología, la utilización de este tipo de herramientas puede tener beneficios económicos para la empresa. Anteriormente se ha mencionado el posible recorte en los plazos hasta el inicio de la producción, el cual libera de una gran cantidad de costes económicos a esta fase de la producción. Esta reducción de costes hace que la inversión inicial se realice de manera más segura. Además, la producción puede comenzar mucho antes, reduciendo así el tiempo para empezar a recuperar la inversión, y adelantando el tiempo de payback.

4.2 Beneficios sociales

La automatización de procesos libera al trabajador de una gran cantidad de carga de trabajo manual pesado, aumentando así su calidad de vida. Otro gran beneficio es el aumento de seguridad de los equipos. Al desarrollar estrategias ante fallos potencialmente peligrosos para la seguridad de los trabajadores, es posible, si bien no erradicar este peligro, reducirlo en lo máximo posible. Ya que con la ayuda de modelos digitales puede comprobarse la eficacia de las medidas adoptadas antes de la puesta en marcha de los equipos.

El aumento de la escasez de recursos, el relacionado aumento de los precios de dichos recursos, y el cambio social en el contexto de los aspectos ecológicos, hacen necesario un enfoque más intensivo en la sostenibilidad de la industria. Siendo el objetivo una mayor eficiencia tanto económica como ecológica, el gemelo digital aporta un claro beneficio social, dado que reduce la cantidad de recursos a utilizar en el ciclo de vida del producto.

Tal y como se ha mencionado en el apartado de beneficios técnicos, la función pedagógica es un punto fuerte de este tipo de tecnologías, no sólo porque mejora el sistema de educación de los alumnos, sino porque, además, aumenta la cualificación de los futuros trabajadores. Todo ello, favorece el mantenimiento de un ecosistema laboral estable y productivo.

4.3 Beneficios técnicos

Las técnicas para la puesta en marcha de equipos pueden beneficiarse enormemente de tecnologías como los gemelos digitales. Estas técnicas aún están por desarrollar en gran medida, debido a su reciente expansión. Asimismo, la consecución de este proyecto puede aportar una visión más amplia tanto de las necesidades por desarrollar en este campo, como de los beneficios que, hoy en día, puede aportar esta tecnología.

Las empresas, antes reticentes a estas tecnologías debido a su alto coste, empiezan a interesarse en las posibilidades que ofrecen. Así, grandes empresas como la planta de Mercedes (Vitoria-Gasteiz) utilizan este tipo de tecnología para la formación de sus empleados. Más allá de su uso como herramienta pedagógica, su mayor aportación a la industria actual es la posibilidad de realizar puestas a punto del control y automatización de máquinas sin la necesidad de tenerlas presentes, permitiendo acortar los plazos de puesta en marcha. Este virtual commissioning o puesta en marcha virtual, combinado con otras herramientas (como la simulación de procesos y de plantas industriales), permite llevar a cabo la planificación de una planta con cierta antelación.

Por otro lado, los gemelos digitales no solo pueden resultar útiles a la hora de ayudar en la puesta en marcha de la maquinaria, si no que pueden ser beneficiosas también para modificar máquinas ya existentes. Por ejemplo, es posible estudiar modificaciones en la máquina (como insertar y retirar elementos) para optimizar la producción de esta. Además, todo ello puede realizarse sin necesidad de parar la máquina.

Otro de los beneficios técnicos que aporta el gemelo digital es la flexibilidad que se consigue. Esta última es la que hace posible diseñar diferentes configuraciones para el desarrollo del producto. La integración con el sistema de agentes permite una mejor toma de decisión para ordenar la producción. Además, este sistema de agentes puede mejorar la fiabilidad de la producción en caso de pérdida de una estación redirigiendo los servicios que se estaban ejecutando en la estación caída en otras estaciones. Añadiendo también el beneficio económico consecuente.

5 Análisis de alternativas

Para desarrollar el proyecto es necesario seleccionar las herramientas que se van a usar durante el mismo. Para ello se va a analizar diferentes aspectos de las diferentes herramientas disponibles y mediante una ponderación de estos aspectos seleccionar las herramientas a utilizar. Han de elegirse el programa en el que llevar a cabo el Gemelo Digital y el tipo de puesta en marcha virtual a realizar.

5.1 Desarrollo del Gemelo Digital

5.1.1 KUKA.Sim 4.0

Este *software* de la empresa fabricante de robots permite la programación *offline* y puesta en marcha virtual de robots KUKA. El desarrollo se hace mediante librerías incluidas en el programa y permite añadir archivos externos para añadir elementos. Este software es adecuado para robots KUKA ya que asegura que los datos son consistentes entre la unidad de control real y virtual. Sin embargo, no permite simular robots de otros fabricantes (figura 5).

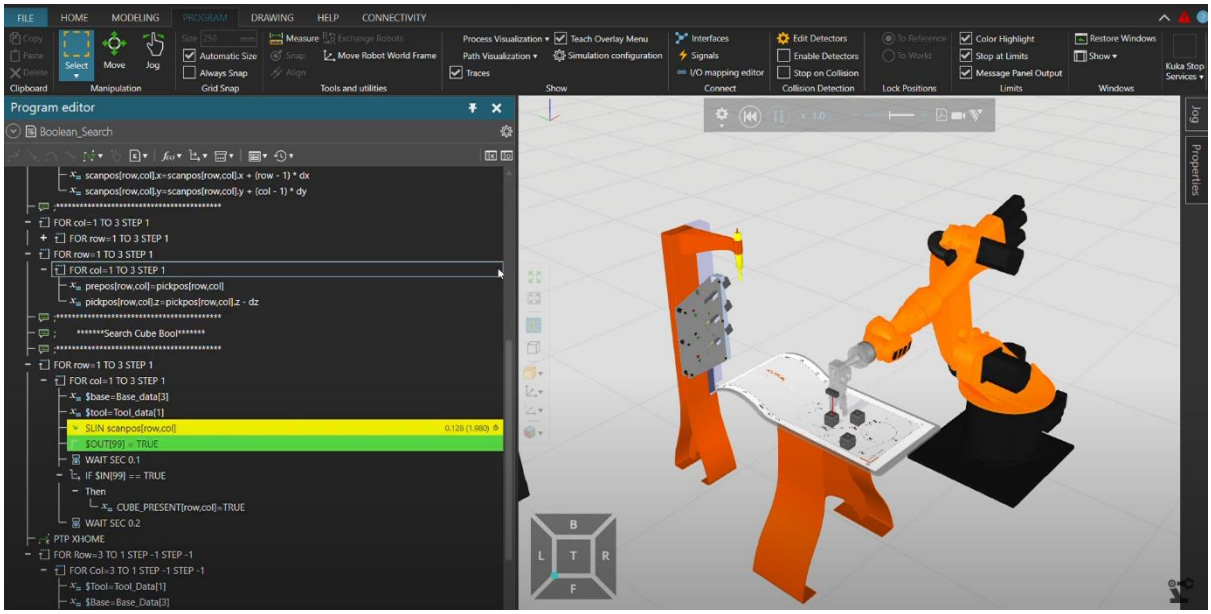


Figura 5. Ventana de trabajo KUKA.Sim 4.0.

5.1.2 RoboDK

RoboDK es un *software* para la programación *offline* de robots y exportación de programas robots. Este software puede trabajar con robots de varios fabricantes como ABB, Fanuc, KUKA o Universal Robots entre otros. Sus principales ventajas es la facilidad de uso para desarrollar programas. Sin

embargo, las posibilidades que ofrece para puesta en marcha virtual son reducidas (figura 6).

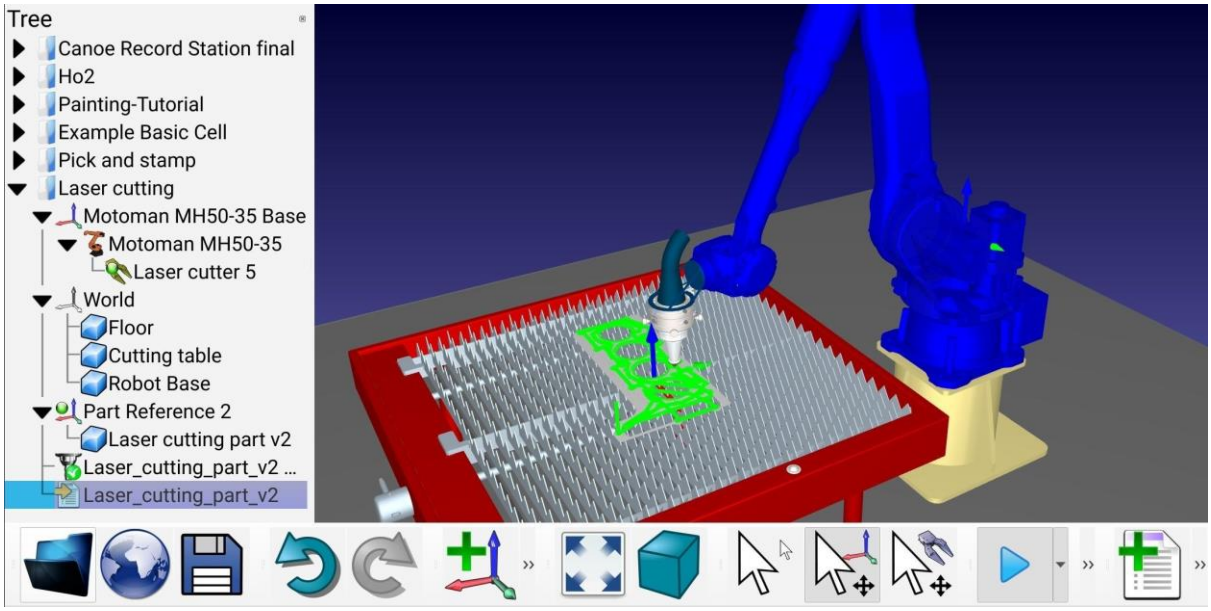


Figura 6. Ventana de trabajo RoboDK.

5.1.3 Siemens Tecnomatix Process Simulate

Este software se engloba dentro de la plataforma PLM de Siemens. Esta herramienta permite el desarrollo *offline* de código robot. Además de permitir validación de lógica de control y elementos auxiliares de la producción. Permite la integración con diferentes métodos de comunicación como OPC UA o otros de la propia Siemens como SIMIT o PLCSIM Advanced (figura 7).

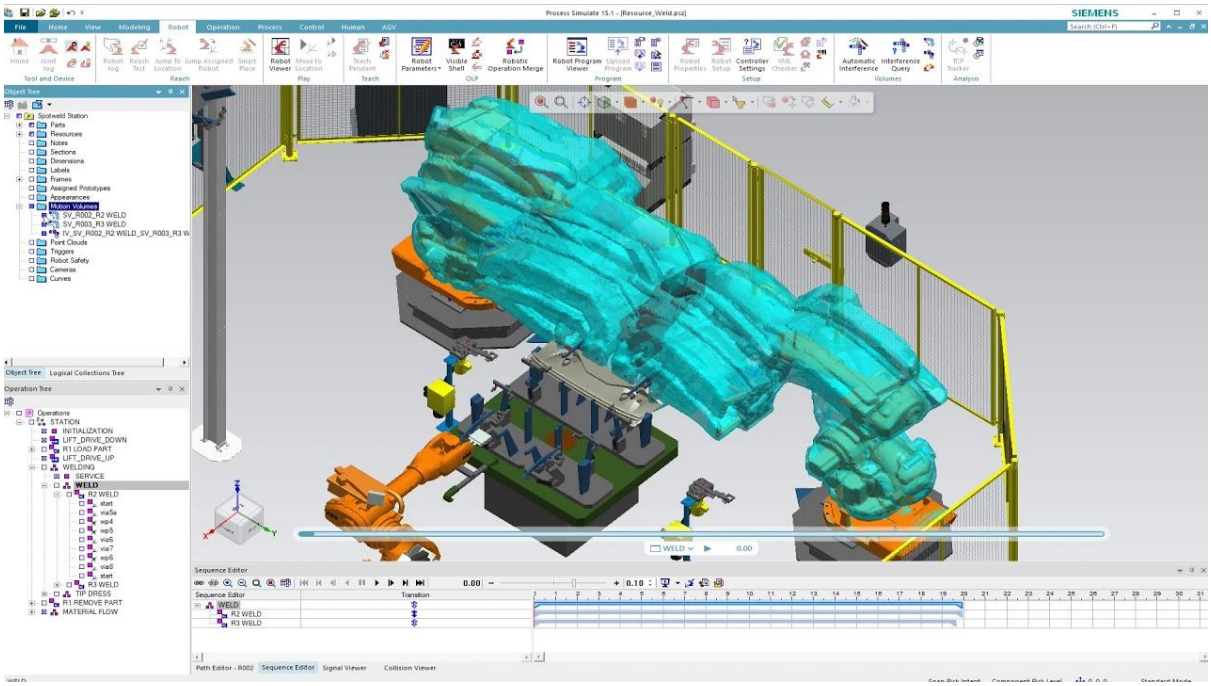


Figura 7. Ventana de trabajo Tecnomatix PS.

5.1.4 Delmia V5

Delmia es la solución de la empresa Dassault Systemes para definir, planificar, crear, monitorear y controlar todos los procesos de producción, desde la planificación del proceso y la simulación de ensamblaje hasta una definición completa de la instalación y el equipo de producción de manera virtual. Permite la validación de líneas de producción especialmente orientadas a células robotizadas (figura 8).

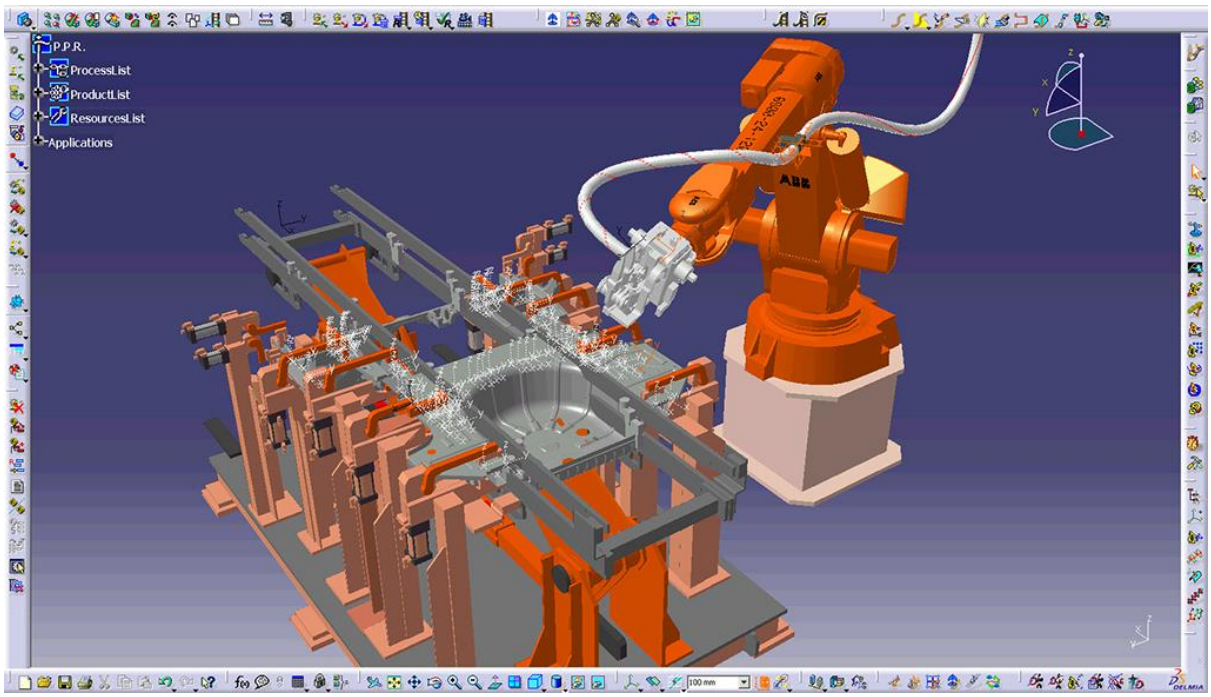


Figura 8. Ventana de trabajo Delmia.

La selección del *software* para modelar el Gemelo Digital se realiza mediante Unificación Ponderada. Los diferentes aspectos analizados se les otorga una ponderación en función de su importancia sumando todas el 100%. Cada programa se puntúa del 1 al 10 en cada aspecto y tras multiplicar y sumar cada factor la mayor puntuación representa el *software* más adecuado.

Para la selección del software se va a realizar el análisis de estos aspectos:

- Modelado (35%): opciones que ofrece para modelar el Gemelo Digital de los elementos de la célula robotizada.
- Flexibilidad (15%): Opciones de añadir robots de diferentes fabricantes, así como introducir recursos externos a las librerías de cada programa.

- Integración con otros programas (30%): integración con otros programas ara la puesta en marcha virtual y la validación de código de control.
- Facilidad de uso (5%)
- Coste (15%)

La tabla 1 muestra los resultados de aplicar la Unificación Ponderada a los diferentes programas analizados:

Tabla 1. Tabla ponderada para la selección de *software* de modelado de Gemelo Digital.

Factores	Peso	Software			
		KUKA.Sim	RoboDK	Tecnomatix	Delmia
Modelado	0.35	7	4	10	8
Flexibilidad	0.15	5	2	9	9
Integración	0.3	6	2	9	8
Facilidad	0.05	9	10	2	4
Coste	0.15	7	9	3	5
	1	6.5	4.15	8.1	7.5

Por tanto, Tecnomatix PS ha sido el Software elegido para llevar a cabo el modelado del Gemelo Digital.

5.2 Puesta en marcha virtual

También es necesario elegir a técnica la usar para la puesta en marcha virtual. Para ello se va a analizar dos posibilidades a implementar.

5.2.1 *Hardware in the loop (HiL)*

El *Hardware in the Loop* (figura 8) es una técnica que permite reducir los tiempos de desarrollo y aumentar la eficacia de los ensayos. El *HiL* proporciona una manera de simular sensores, actuadores y componentes mecánicos de manera directa contra el controlador real. De esta manera se consiguen tiempos de respuesta y ciclo, estímulos eléctricos y casos de uso funcionales representativos. La parte que representa el entorno, sensores y actuadores se conoce como modelo.

Por lo tanto, el *HiL* necesita de hardware real para realizarse, en este caso, el PLC. Además, según el fabricante puede ser necesario elementos adicionales para poder llevar a cabo esta técnica. Sin embargo, algunos dispositivos, como la gama S7-400 de Siemens obliga a utilizar esta técnica ya que no se puede simular. Por tanto, sobre todo con dispositivos más antiguos, es la única técnica posible.

5.2.2 Software in the Loop (SiL)

El *Software in the Loop* (figura 8) permite realizar pruebas similares al *HiL* en las primeras etapas de desarrollo. Para ello se realiza una simulación del controlador frente al modelo mediante una CPU virtual. Por tanto, no es necesario disponer del *hardware* para realizar estas simulaciones, ya que ambos modelos, controlador y célula, son virtuales. Estas pruebas permiten mejorar iterativamente los desarrollos creados alcanzando una mayor madurez del código desde etapas más tempranas, permitiendo descubrir y subsanar errores y optimizar el código.

En este proyecto la simulación del código de automatización se puede realizar con esta técnica mediante instancias virtuales de PLC sin necesidad de contar con el propio PLC y las tarjetas correspondiente necesarias. Por tanto, suponen un coste más reducido.

5.2.3 Elección

Dadas las características de ambas técnicas, cada una de ellas se ajusta más a diferentes estados del desarrollo. El *SiL* permite realizar simulaciones desde etapas más tempranas acelerando los tiempos de desarrollo. Además, evita la necesidad de usar costoso hardware en estas etapas tempranas y simplifica la conexión entre controlador y modelo. Por otra parte, el *HiL* permite analizar los tiempos de ciclo y respuesta reales del sistema de control embebido, por tanto, muestra un comportamiento más representativo de la aplicación real.

Conociendo estas ventajas y desventajas de las técnicas, se ha decidido usar el *Software in the Loop* en este proyecto. La principal ventaja del *HiL* es validar los tiempos de ciclo y respuesta representativos. Dado que en este proyecto estos tiempos no tienen una importancia vital. Se ha decidido priorizar la eficiencia en el desarrollo y la flexibilidad que ofrece el *SiL*. Además, la plataforma PLM de Siemens ofrece productos especificos para este uso que se integran perfectamente con el software Tecnomatix PS elegido en el apartado anterior.

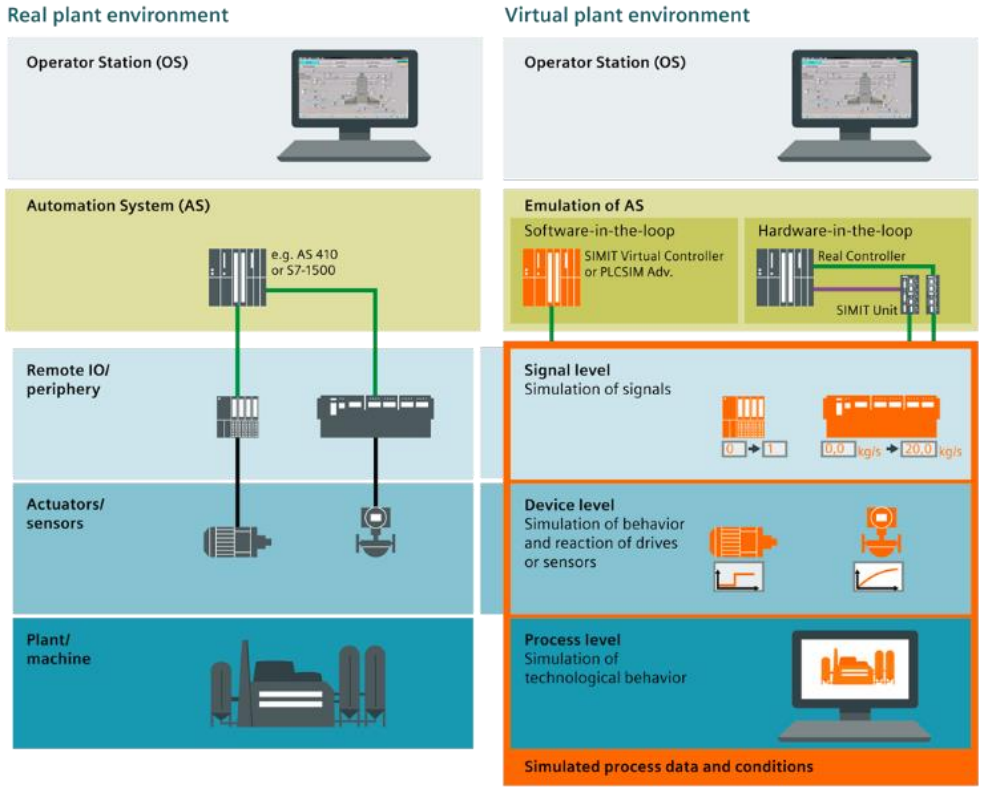


Figura 8. SiL y HiL comparado con el control en entorno real.

6 Análisis de riesgos

1.Riesgo. Alta cualificación necesaria para el uso del software.

Probabilidad: Muy alta. Para la utilización de estos softwares es necesaria un nivel alto de formación, dado que el conocimiento actual no es muy amplio.

Impacto: Muy alto. El proyecto no puede completarse sin el conocimiento necesario para el uso de las herramientas.

Puntuación: Alto (0,72)

Plan de contingencia: Se pondrá a cargo del proyecto a una persona con una cualificación adecuada.

2.Riesgo. Diseño inexacto del gemelo digital.

Probabilidad: Alta. Debido a la cantidad de variables a considerar, resulta factible realizar mal algún ajuste tanto en los sensores como en los actuadores.

Impacto: Alto. El diseño es la base del proyecto y, por ende, un error en este paso producirá un funcionamiento incorrecto.

Puntuación: Alto (0,28)

Plan de contingencia: Los errores deben solucionarse nada más sean detectados.

3.Riesgo. Realizar un mapeo incorrecto.

Probabilidad: Baja. Conectar las variables del gemelo digital a las del PLC no resulta complicado ya que la similitud en los nombres es alta.

Impacto: Medio. Pese a que un fallo en el mapeo tendrá como consecuencia el mal funcionamiento del modelo, su solución es más sencilla, lo cual provoca que el impacto sea menor.

Puntuación: Medio (0,06)

Plan de contingencia: Se usarán identificadores similares en ambas plataformas para evitar este riesgo, y en caso de detectar algún error, se proporcionará una solución lo antes posible.

4.Riesgo: Incompatibilidad en el software.

Probabilidad: Muy baja. Como los softwares que se utilizan pertenecen a Siemens, es poco probable que surjan incompatibilidades entre ellos.

Impacto: Medio. La incompatibilidad de software imposibilita la simulación; aun así, todavía permite realizar el diseño del modelo digital y del proyecto de control por separado.

Puntuación: Bajo (0,02)

Plan de contingencia: Debe comprobarse que las licencias siguen vigentes, que los permisos de usuario han sido concedidos y, como último recurso, contactar con la asistencia de la empresa. **5.Riesgo**. Error en opuesta en marcha real.

Probabilidad: Media. La cantidad de código a desarrollar y el control del flujo para sistemas flexibles implican una mayor probabilidad de realizar errores en la programación

Impacto: Muy alto. Los errores en trayectorias o flujos pueden causar daños físicos en los elementos e incluso amenazar la seguridad de las personas.

Puntuación: Alto (0.4)

Plan de contingencia: Antes de realizar la puesta en marcha definitiva se comprobará manualmente todos los programas a velocidad reducida e intervención del operario.

La Tabla 2 representa la Matriz de Probabilidad-Impacto de los riesgos analizados, la cual ayuda en la determinación de la gravedad de estos. Los valores asignados a cada riesgo se han calculado siguiendo la Matriz.

Tabla 2. Matriz de Probabilidad-Impacto.

PROBABILIDAD	IMPACTO					
		Muy bajo (0,05)	Bajo (0,1)	Medio (0,2)	Alto (0,4)	Muy alto (0,8)
	Muy baja (0,1)	Bajo 0,005	Bajo 0,01	Bajo 0,02	Medio 0,04	Medio 0,08
	Baja (0,3)	Bajo 0,015	Bajo 0,03	Medio 0,06	Medio 0,12	Alto 0,24
	Media (0,5)	Bajo 0,025	Medio 0,05	Medio 0,1	Alto 0,2	Alto 0,4
	Alta (0,7)	Bajo 0,035	Medio 0,07	Medio 0,14	Alto 0,28	Alto 0,56
	Muy alta (0,9)	Medio 0,045	Medio 0,09	Alto 0,18	Alto 0,36	Alto 0,72

7 Metodología

7.1 Descripción de la solución

En este apartado se van a mostrar los elementos que componen la célula de montaje. Las características de cada parte del proyecto son esenciales tanto para la puesta en marcha real, como para realizar el modelo del gemelo digital en Tecnomatix Process Simulate. En el laboratorio se cuenta con una estación como la que se va a describir. Sin embargo, en la validación mediante *software in the loop* se va a duplicar la estación para poder simular diferentes combinaciones de operación. Una vez validado en simulación se llevará a cabo la puesta en marcha de la estación real.

7.1.1 Piezas

El objetivo del proyecto es llevar a cabo el montaje de un conjunto de 5 piezas. Las cinco piezas son la base, el rodamiento, el bulón, la tapa interior y la tapa exterior que cierra el conjunto (figura 9). Las piezas deben montarse en ese orden obligatoriamente. Con el objetivo de probar un diseño más modular y flexible, los servicios pueden ser de uno a seis ítems y con cualquier número de piezas en todos los conjuntos. El servicio debe montar al menos una pieza más en cada elemento del conjunto, pero no tiene por qué finalizar el conjunto. Por ejemplo, puede llegar un pallet con tres conjuntos de base y rodamiento, y llevar a cabo un montaje que añada el bulón y la tapa interior, por falta de tapas exteriores en la estación. Con estas condiciones es posible realizar 10 servicios diferentes de entre 1 y 6 ítems, por lo tanto, hay 60 combinaciones diferentes. En la tabla 2 se muestra en cada servicio que piezas llegan en los conjuntos (azul) y cuales se montan (naranja):

Tabla 3. Piezas que se montan en cada servicio.

N.º. DE SERVICIO	BASE	RODAMIENTO	BULÓN	TAPA INTERIOR	TAPA EXTERIOR
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					

Las piezas no tienen requisitos de esfuerzo y por tanto para su fabricación se ha elegido el plástico debido a su bajo coste. Además, con este material, se puede usar la fabricación aditiva con una impresora 3D. Por tanto, la fabricación y sustitución de piezas por rotura, o mejoras de diseño es rápida y con bajo coste. El diseño del conjunto está basado en otro proyecto del departamento. La fabricación de las piezas se ha llevado a cabo en la impresora 3D del laboratorio.

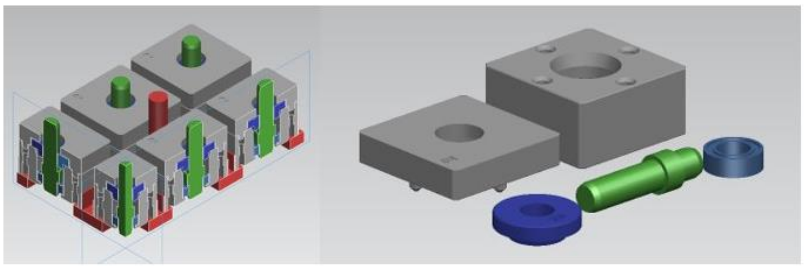


Figura 9 Piezas del conjunto (derecha) y corte del conjunto montado (izquierda).

7.1.2 Garra

A fin de manipular las piezas, se ha diseñado una garra con dos pinzas. Ambas se ubican en el lateral de una plancha en forma de U para separarlas. El centro de la placa se acopla a la brida del robot. Por la naturaleza de la aplicación, prima la velocidad a la precisión y fuerza. Por tanto, se ha elegido usar pinzas neumáticas. Las pinzas son de la familia MHK2 [7] de la compañía SMC. Debido a las características de los elementos de la pieza, podemos categorizar los elementos en dos tipos:

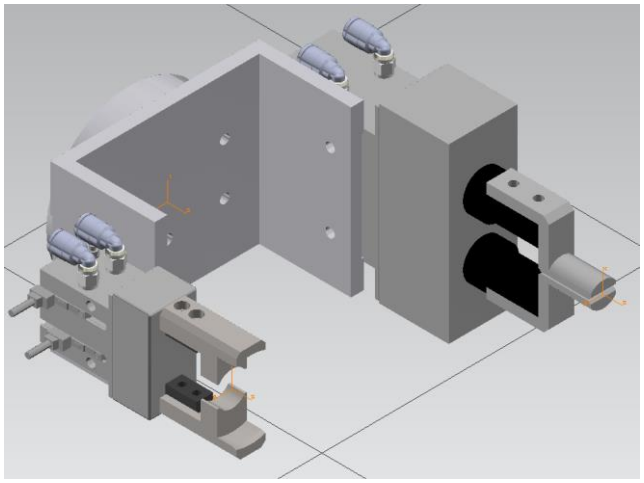


Figura 10. Garra del robot. Pinza de bulones y pallets (izquierda) y pinza de rodamiento y tapas (derecha).

- El primer grupo son las piezas que se manipulan por una parte cilíndrica (bulón y pallet).
- El segundo grupo son las piezas que se manipulan por un orificio cilíndrico (rodamiento, tapa interior y tapa exterior).

En la tabla 3 se muestran las características técnicas de las pinzas.

Tabla 3. Especificaciones de las pinzas.

	DIÁMETRO MM	***FREQ. MÁX. DE OPERACIÓN C.P.M.***	***CARRERA MM***	***MASA G***	***PRESIÓN DE OPERACIÓN MPA***
MHK2-16D	16	90	6	113	0.1-0.6
MHKL2-25D	25	60	22	562	0.1-0.6

Para el primer grupo la solución es usar una pinza normalmente abierta NA. De esta manera, se aproxima la pinza abierta al cilindro. Al actuar sobre ella se cierra agarrando así el pallet o bulón. La garra elegida para este caso es la MHK2-16D (figura 11). Es un cilindro de doble efecto de 16 mm de diámetro con una carrera de 6 mm.

En el caso del segundo grupo, al contrario, se una un agarra normalmente cerrada NC. En este caso, la aproximación se hace con la pinza cerrada, y al actuar se abre agarrando el rodamiento o tapa. La garra elegida ha sido la MHKL2-25D (figura 11). Igual que en el caso anterior es un cilindro de doble efecto. La pinza es de 25 mm de diámetro y carrera larga de 22 mm para adecuarse a todas las piezas. En los dedos de ambas pinzas se han colocado los accesorios necesarios para que el agarre entre pinza y elemento sea correcto.

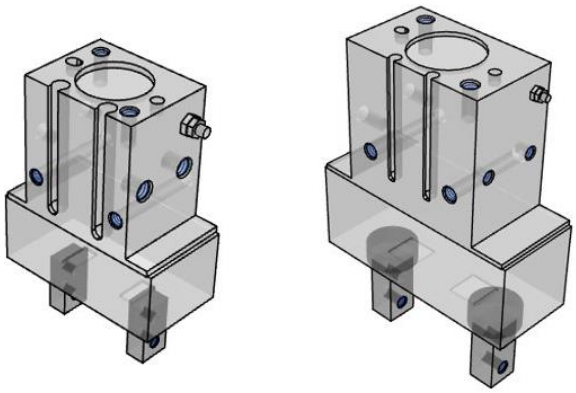


Figura 11. De izquierda a derecha, pinza MHK2-16D y MHKL2-25D.

Las electroválvulas que controlan los cilindros de doble efecto son también de la compañía SMC. Ambas son el modelo SY3100-5UD (figura 12). Estas, son electroválvulas 5/2 servopilotadas y con retorno por muelle neumático [8]. Esto permite que los solenoides tengan un tamaño más reducido. Cuando se actúa sobre el solenoide abre un canal por el que entra aire comprimido cambiando así la posición de la válvula. Este modelo junto con un cilindro de doble efecto puede actuar como NA o NC en función de cómo se coloquen las mangueras. Tal y como se ha comentado anteriormente, es necesario colocar una como NA y la otra como NC.

El aire comprimido proviene de una toma neumática del circuito del laboratorio. En la toma de aire es necesario un pretratamiento para ello se ha introducido un filtro/regulador AW20-F02-A-X64 para mantener la presión dentro del rango de funcionamiento (0.05-0.7 MPa). El accionamiento del solenoide se ha realizado con un bus de campo EtherCAT que actúa directamente sobre la bobina con una tensión de 24 VDC. El esclavo EtherCAT está, por un lado, conectado por cable ethernet a la controladora que actúa como maestro, por otro lado, mediante un conector DB-25 al módulo de las electroválvulas. Desde el programa robot se actúa sobre las variables del bus de campo, y el esclavo transforma esa información en la señal eléctrica que se envía por el conector a los solenoides.

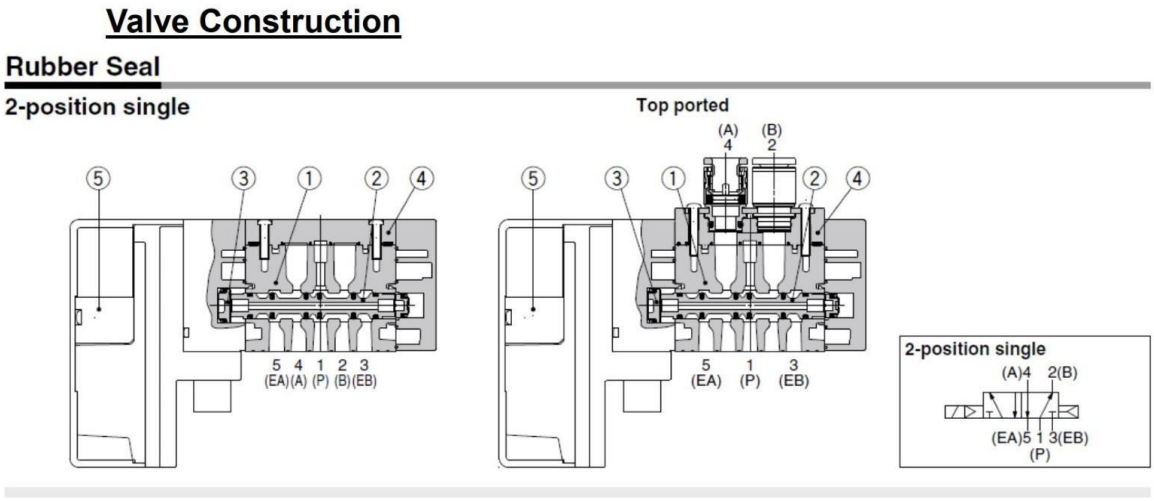


Figura 12. Esquema neumático de la electroválvula.

7.1.3 Escenario

Las células robotizadas son parte de un sistema de fabricación que consta de un robot o más en el que se llevan a cabo operaciones de secuencias repetitivas de carga, descarga, montaje, etc [9]. Toda célula lleva a cabo su trabajo en un espacio diseñado para la operación a realizar, también llamado escenario. En este proyecto la célula recoge el pallet de piezas desde un AGV, realiza el servicio ordenado por el sistema de agentes, y vuelve a colocar el pallet en el AGV para que continúe se producción. En este apartado se muestra el diseño del escenario de la estación y sus componentes.

La estación robotizada cuenta con un robot encargado de realizar las operaciones de manipulación y montaje. El robot está situado en el centro de la mesa para que todas las partes de la mesa se encuentren dentro de su alcance. La mesa tiene una superficie con raíles para facilitar el montaje de los diferentes elementos que conforman el escenario. Además, en la parte frontal de la mesa se encuentran diferentes pulsadores, entre ellos la seta de emergencia que corta la alimentación de la célula en caso de emergencia. La figura 13 muestra el escenario del Gemelo Digital en Tecnomatix PS.

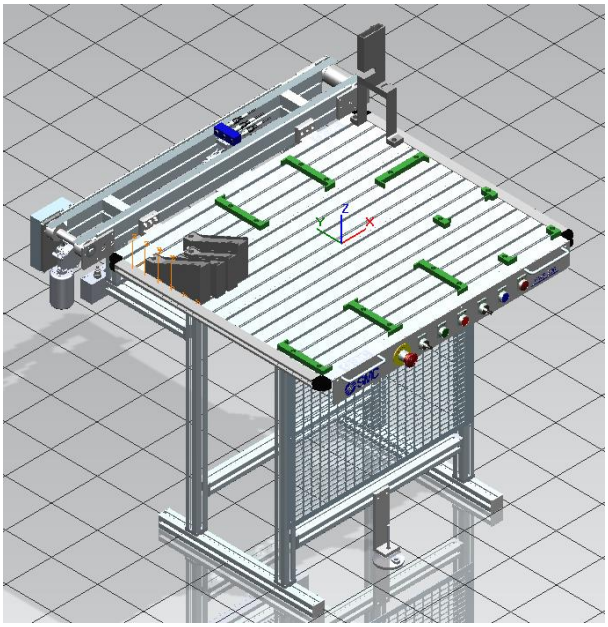


Figura 13. Captura del escenario.

La estación se divide en tres zonas: entrada, montaje y salida. Cada una de las zonas está delimitada por unas barreras del tamaño del pallet. De esta manera, se delimita el espacio que debe ocupar el pallet y otorga estabilidad durante la manipulación y montaje. La zona de entrada consta de dos espacios para colocar a la izquierda el pallet con bases, y a la derecha el pallet con las tapas exteriores. Cuando llega un nuevo pallet este se coloca en esta zona de entrada. A continuación, el pallet de las bases se traslada a la zona de montaje donde se lleva a cabo el servicio. Una vez finalizado, el pallet con los conjuntos se lleva a la zona de salida para ser retirado al AGV (figura 14).

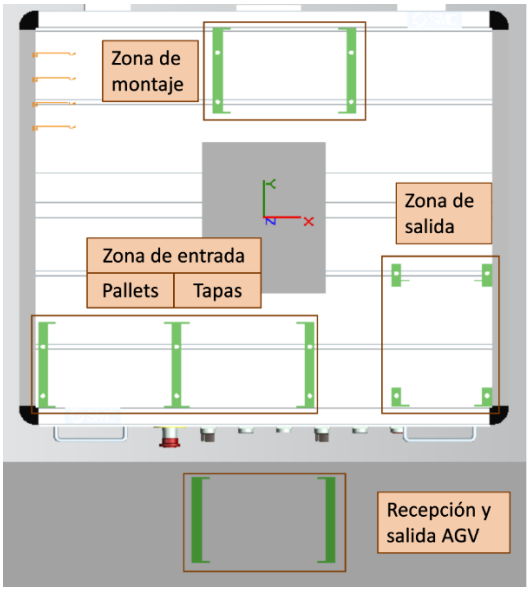


Figura 14. Vista de planta de las diferentes zonas de la estación.

El resto de las partes del conjunto se suministran mediante alimentadores. Los alimentadores deben garantizar que las partes siempre quedan colocadas en la misma posición para evitar problemas con el robot. También, deben permitir un flujo sin interrupciones. Asimismo, es conveniente no introducir mecanismos o elementos móviles innecesarios que puedan ser causa de fallos y costes. Por tanto, los alimentadores hacen uso de la gravedad para cumplir su función. Tanto los alimentadores de los rodamientos como de las tapas interiores se alimentan mediante rampas adecuadas a la geometría y dimensiones de las piezas. El alimentador de bulones, sin embargo, es vertical y diseñado para mantener el bulón en la posición correcta (figura 15). Debido a la forma de la garra este alimentador está elevado para evitar colisiones al manipular el bulón. De lo contrario, la parte de la garra para la manipulación de rodamientos y tapas colisionaría con el propio alimentador.

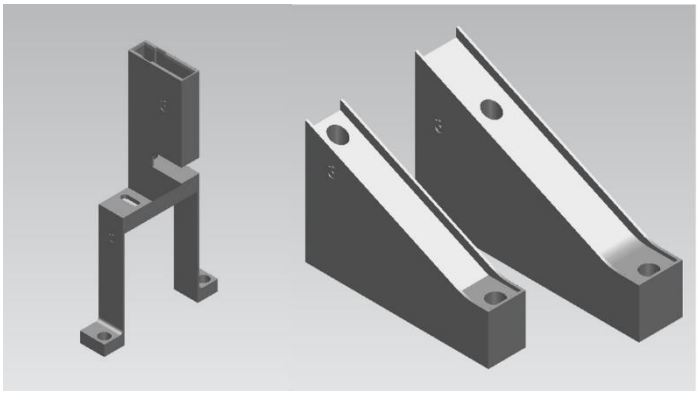


Figura 15. Alimentador de bulones (izquierda) y rampas alimentadoras de tapas interiores y rodamientos (derecha).

Debido a que el plástico garantiza el deslizamiento de las piezas lo hace un material idóneo para este proyecto. En caso de algún fallo en el diseño de trayectorias o ajustes de posición, el plástico simplemente se ocasionaría una rotura del elemento en cuestión y no comprometería el resto de la estructura o el propio robot, pudiendo ser fácilmente reemplazado. Las barreras y alimentadores se van a fabricar junto con las piezas. Por último, se han colocado dos alimentadores adicionales para en un futuro permitir la ampliación de capacidad y servicios.

7.1.4 Robot y controladora

Las principales restricciones para la elección de un robot son el alcance y la capacidad de carga. Para este proyecto, las necesidades de carga son mínimas. De hecho, la gama de capacidad de carga más baja de cualquier compañía oferta robots que superan estas necesidades. Por tanto, los requisitos que debe cumplir el robot es que el alcance sea suficiente para permitir el acceso a todos los elementos del escenario.

El robot elegido ha sido el KR3 R540 de la gama AGILUS de KUKA. El robot de 6 ejes tiene un alcance máximo de 541 mm, cumpliendo así el requisito impuesto. Además, se ha seleccionado este robot debido a que su reducido tamaño permite un mejor uso del espacio del escenario. Entre sus principales características, destacan la carga nominal de 3 Kg y la repetibilidad de ±0.02 mm. En la figura 16 se muestran los datos técnicos y alcance del robot:

Datos técnicos

Alcance máximo	541 mm
Carga máxima	3 kg
Repetibilidad de posición (ISO 9283)	± 0,02 mm
Número ejes	6
Posición de montaje	Suelo; Techo; Pared
Superficie de colocación	179 mm x 179 mm
Peso	aprox. 26,5 kg

Datos de los ejes

Rango de desplazamiento	
A1	±170 °
A2	-170 ° / 50 °
A3	-110 ° / 155 °
A4	±175 °
A5	±120 °
A6	±350 °

Condiciones de servicio

Temperatura ambiente durante el servicio	5 °C hasta 45 °C (278 K hasta 318 K)

Gráfica del campo de trabajo

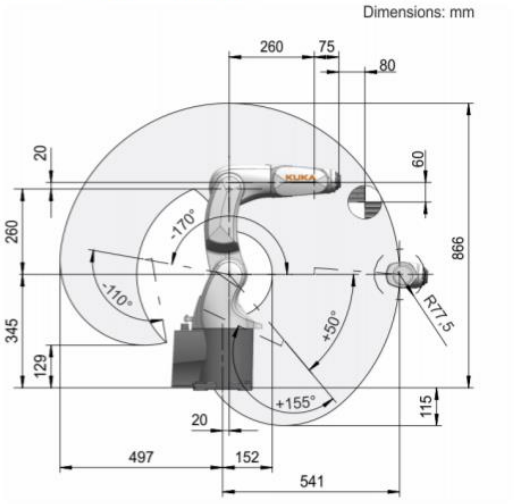


Diagrama de cargas

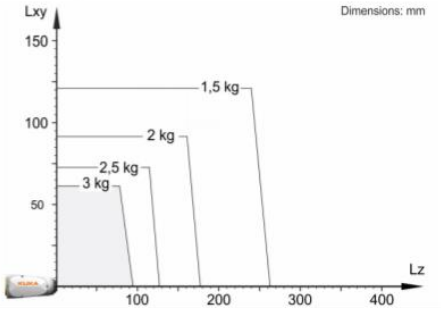


Figura 16. Datos técnicos del robot KUKA KR3 R540.

El robot KR3 R540 permite usar dos controladoras diferentes, como es lógico ambas del mismo fabricante. Por un lado, la KUKA KR C4 Compact, y por otro, la KUKA KR C5 Micro. Ambas controladoras permiten la comunicación por bus de campo mediante al menos PROFINET y EtherCAT necesarios en este proyecto. Por lo tanto, la elección ha sido la KR C4 Compact por el menor coste.

La controladora permite el control de 6 ejes, hasta 8 añadiendo una caja de ejes. Además, cuenta con una *teach pendant* para interactuar con el robot y realizar configuración y diagnóstico. También cuenta con dos puertos ethernet. Uno de ellos únicamente permite añadir esclavos EtherCAT para entradas y salidas que se une con el bus interno de la controladora. El otro permite utilizar diferentes

buses de campo, en el que se va a usar PROFINET para la comunicación entre el PLC y la controladora. En la tabla 4 se muestran las especificaciones de la unidad de control:

Tabla 4. Especificaciones de la unidad de control KR C4 Compact.

KR C4 Compact especificaciones	
Dimensiones	271x483x460mm
Disco duro	SSD
Número de ejes	6+2(con caja de ejes adicional)
Voltaje de alimentación	AC 200V-230V
Frecuencia	50/60Hz ±1Hz
Protección	IP20
Comunicaciones	PROFINET, PROFIBUS, INTERBUS, EtherCAT, Ethernet/IP, DeviceNet y VARANBUS
Peso	33Kg

7.1.4.1 SmartPAD

Junto con el robot y controladora se dispone de la *teach pendant* para llevar a cabo la programación, diagnóstico y mantenimiento llamada *smartPAD*. De esta manera se lleva a cabo la comprobación de los programas desarrollados y realizar los ajustes necesarios. También es posible mover el robot de forma manual o controlar la velocidad de movimiento. En la figura 17 se muestra es aspecto y los elementos de los que dispone:

Nº de elemento	Descripción
1	2 puertos USB 2.0
2	Botón para desenchufar el smartPAD
3	Selector de modos de servicio. Cuando está en posición horizontal, en la pantalla se muestran los 4 modos de servicio disponibles, se selecciona uno de ellos y se vuelve a colocar en posición vertical.
4	Pulsador de emergencia
5	Ratón 6D para el movimiento manual del robot
6	Teclas de desplazamiento manual. Cada una de ellas actúa sobre uno de los accionamientos del robot.
7	Correa de mano con cierre de velcro.
8	Tecla para ajustar el Override del programa. Override es el porcentaje de la velocidad a la que se mueve el robot respecto de la configurada en el programa o en el robot.
9	Tecla para ajustar el Override manual
10	Cable de conexión
11	Teclas de usuario. Su función se puede programar libremente.
12	Tecla de arranque. Inicia la ejecución del programaseleccionado
13	Tecla de arranque hacia atrás. Se emplea para retroceder en las instrucciones de un programa en ejecución.
14	Tecla de parada. Se emplea para detener la ejecución de un programa.
15	Tecla del teclado. Muestra el teclado.
16	Tecla de menú principal. En el menú principal se puede realizar la configuración de diferentes propiedades de la unidad de control del robot, así como la visualización del estado de las señales del robot, por ejemplo.

Figura 17. *SmartPAD* y funciones de los botones.

7.2 Modelado

7.2.1 Definición e inserción de elementos

Tecnomatix Process Simulate es una herramienta de la plataforma PLM de Siemens para simular procesos roboticos y la validación de la solución. Además, permite la interacción con otras herramientas de la plataforma de Siemens. Por lo tanto, entre otras funcionalidades es posible validar proyectos de automatización PLC. Mediante esta herramienta también se va a desarrollar el programa robot directamente en lenguaje KRC, propio de KUKA. También se van a obtener los puntos necesarios para las trayectorias del robot.

Tecnomatix Process Simulate tiene dos modos de funcionamiento, *standar* y *line simulation*. Por norma general en *standar* se lleva a cabo la inserción de componentes, creación del programa robot y más tareas relacionadas con el desarrollo del escenario y los movimientos del robot. En *line simulation mode* se lleva a cabo todo lo relacionado con las señales y simulación. El primer paso para realizar en el desarrollo del Gemelo Digital es introducir los elementos que conforman la célula en el modo de funcionamiento *standar*. Los modelos se han obtenido de los recursos que ofrecen los fabricantes en diferentes formatos. Para poder usar estos recursos, es necesario convertir los archivos al formato de Siemens PLM Software *.jt. Una vez convertidos introducirse en una carpeta con el mismo nombre acabado en *.cojt. Todas las carpetas de este formato deben introducirse en una carpeta que hará la función de librería, y que se debe referenciar la ruta donde se encuentra la carpeta librería en los ajustes de Process Simulate. Una vez todos los archivos están listos y ubicados, hay que indicar para cada *.cojt que tipo de elemento es, y en función de la elección realizada en Tecnomatix, se crean archivos complementarios con sus propiedades.

A la hora de definir el *component* Tecnomatix PS ofrece varias posibilidades. Podemos realizar dos grandes grupos de elementos: *part* y *resource*. Las *parts* van a ser los consumibles que se utilizan en el ensamblaje de la pieza y que serán manipuladas (base, rodamiento, pallet, tapa…). Los recursos, en cambio, serán los elementos persistentes que forman el escenario y los que van a manipular las *parts*. Dentro del grupo *parts*, podemos encontrar los ensamblajes, que forman elementos monolíticos nuevos juntando varias partes diferentes. Sin embargo, dentro de los recursos hay una gran variedad de subgrupos en función de sus características y propiedades. El subgrupo asignado restringe las operaciones que puede realizar el componente, por lo tanto, es muy importante asignar correctamente los mismos. En la tabla 5 se recogen los subgrupos de recursos utilizados en el proyecto:

Tabla 5. Subgrupos de *resources* en Tecnomatix PS.

NOMBRE	DESCRIPCIÓN
ROBOT "ROBOT"	Permite la creación de operaciones robóticas y desarrollar el programa robot. Contiene cinemática, y puede manipular señales.
GARRA "GRIPPER"	Permite la manipulación de partes y tareas de soldadura. Contiene la cinemática del elemento y las posiciones de la herramienta
DISPOSITIVO "DEVICE"	Permite crear tareas para elementos del escenario que tengan movimientos, pero no pertenezcan a los dos subgrupos anteriores. Por ejemplo, puertas, elevadores, útiles…
PROTOTIPO "TOOLPROTOTYPE"	Es el componente por defecto. Generalmente se emplea para definir elementos estáticos de la estación como alimentadores, por ejemplo. No obstante, también se les puede dotar de cinemática.

En el proyecto los robots KUKA kr3 r540 se definen como recurso *robot*. Las herramientas de manipulación de las partes acopladas a los robots se definen como recurso *gripper*. El AGV que traslada las piezas entre los robots se define como recurso *device*, puesto que contiene una parte móvil que sube y baja para facilitar el alcance de los robots. Los elementos estáticos del escenario, es decir, mesas, alimentadores y posicionadores se definen como recurso toolprototype. Las piezas que se van a manipular (pallets, bases, tapas…) se definen como *parts*. Por último, se crean los *assemblies* que forman los diferentes casos de entrada según los servicios. Una vez definidos todos los elementos se insertan todos los elementos de la célula y se colocan cada uno en su lugar, replicando la estación real. Todos los elementos introducidos se pueden organizar en el visor llamado *object tree*. En la figura 18 se pueden observar el *object tree* del proyecto donde se muestran los recursos del proyecto como el robot y sus componentes, garra o alimentadores.

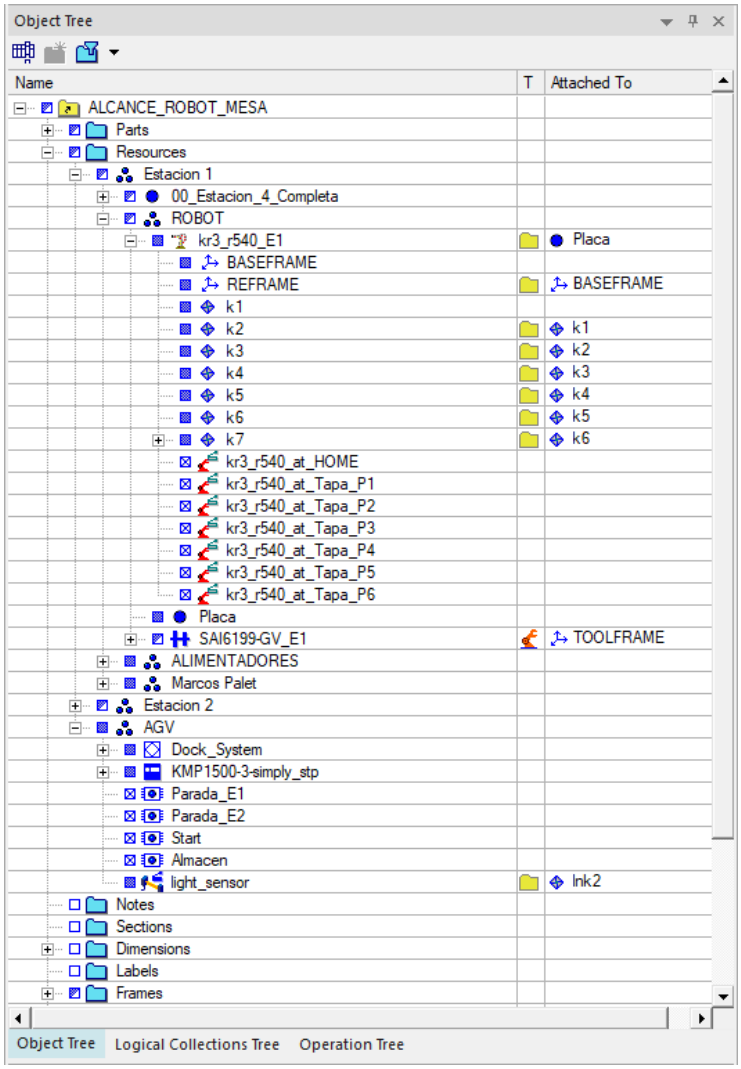


Figura 18. *Object tree* del Proyecto.

7.2.2 Cinemática

Una vez definidos los componentes del Gemelo Digital, el siguiente paso es definir la cinemática de los componentes móviles. Existen varios métodos para analizar el problema cinemático como las matrices jacobianas. Sin embargo, con programas informáticos es más usual utilizar métodos numéricos ya que se ajustan mejor a la computación. Por tanto, el método más usado para este propósito es el método de propagación de velocidades.

El método de propagación de velocidades crea una cadena cinemática siendo sus elementos eslabones y articulaciones. Los eslabones son los elementos rígidos que componen los elementos. Cada eslabón contiene dos articulaciones que unen al eslabón con el anterior y el siguiente. Por tanto, para un número de eslabones “i” tendremos “i+1” articulaciones. Las articulaciones pueden ser de dos tipos: de rotación o de translación. Teniendo en cuenta la hipótesis de solido rígido en los eslabones, la

velocidad de un elemento “i” será la velocidad del elemento “i-1” más la relativa entre los dos elementos. Haciendo este cálculo para todos los elementos desde la base hasta el extremo en orden se determina las velocidades angulares y lineales del extremo.

Por tanto, debemos especificar en el modelo para cada elemento móvil cuales son los eslabones y que tipo de articulaciones unen cada eslabón con el siguiente para poder aplicar el método. En este Gemelo Digital existen cinco componentes que tienen elementos móviles: dos robots, dos pinzas y el AGV. Dado que los robots y las pinzas son iguales se van a modelar uno de cada y posteriormente duplicar el elemento. En el caso del robot, el recurso facilitado en la herramienta tiene definida la cinemática del robot. Por tanto, no es necesario realizar ninguna configuración adicional. Como se puede observar en la figura consta de una base fija y cada eslabón, representado por un color diferente, lo unen articulaciones de rotación donde se determinan sus características. En la figura 19 se muestra en el cuadro de dialogo los cuadrados de lores que representan cada elemento, cada uno con su color correspondiente que coincide con el elemento del robot, y las flechas que forman las articulaciones formando la cadena cinemática uniendo los elementos.

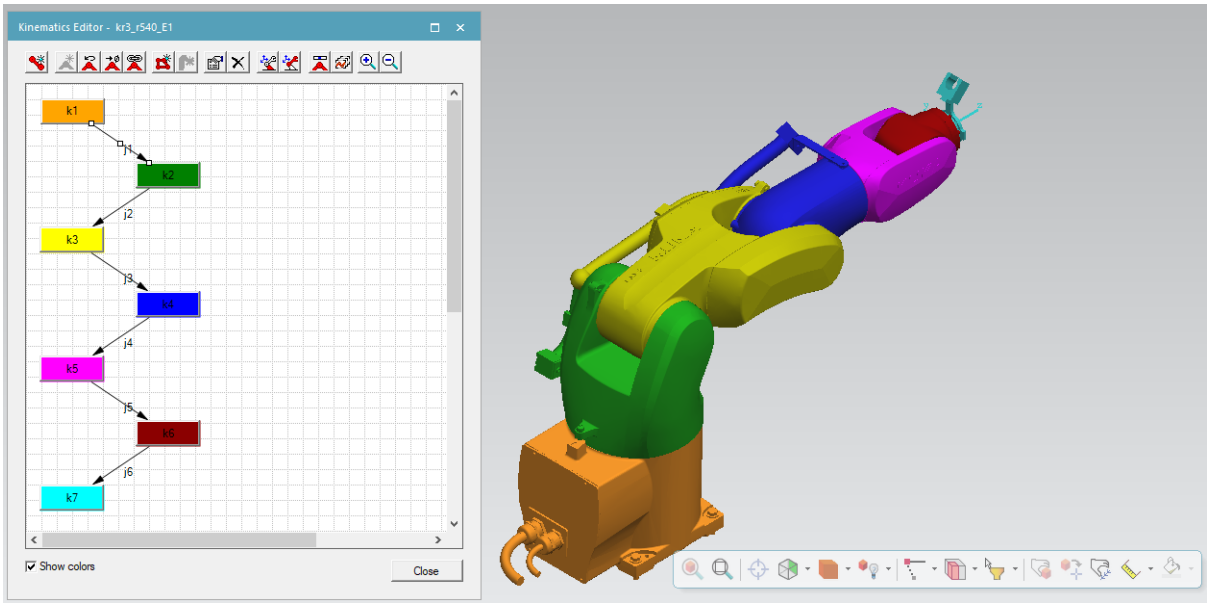


Figura 19. Cadena cinemática del robot KUKA.

En cambio, tanto la garra como el AGV no contiene la cinemática definida, por lo que es necesario configurarla. Se va a detallar la realización de configuración para la garra, ya que los procedimientos son idénticos. La configuración del AGV se resumirá explicando sus particularidades. Para ello, En Tecnomatix Process Simulate se accede al menú *Kinematics Editor*. La configuración se realiza mediante un diagrama en el que cada caja representa un eslabón y las flechas cada articulación.

En el caso de la garra, se ha introducido únicamente con el CAD, por lo que se ha tenido que definir la cinemática. La garra contiene dos pinzas de dos dedos paralelos cada una. Por tanto, cada pinza consiste en un eslabón base y dos eslabones unidos a la base con articulación de traslación. Los dos eslabones que componen los dedos tienen como base el mismo elemento fijo. La otra pinza tiene una configuración análoga, pero con diferentes recorridos. En resumen, la garra consiste en un eslabón fijo al que se le unen cuatro eslabones con articulaciones móviles. Al crear las articulaciones prismáticas es necesario especificar en qué eje se permite el movimiento, junto a los limites superior e inferior de movimiento con los valores especificados en la figura x. En la siguiente figura se muestra el diagrama de la cinemática de la pinza y sus eslabones. En la figura 20, al igual que en la anterior se muestra el cuadro de dialogo que relaciona los elementos y articulaciones y vemos que su composición sigue lo planteado.

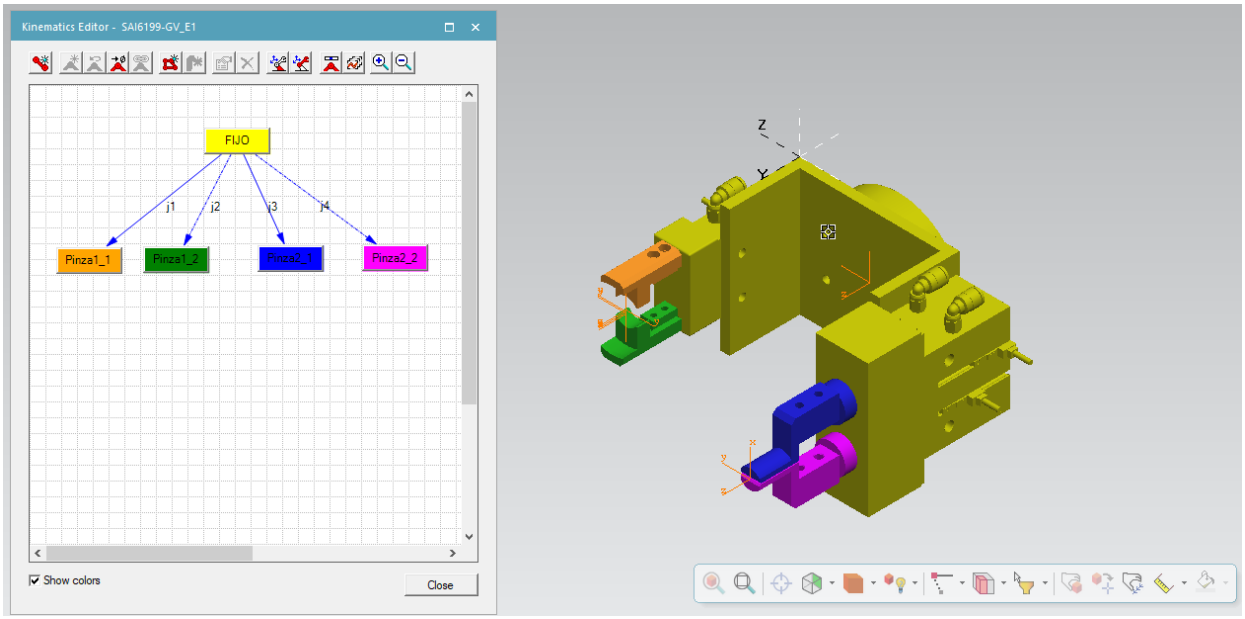


Figura 20. Cadena cinemática de la garra.

Aun así, es necesario añadir otra restricción a los elementos de las pinzas. Esto se debe a que ambos dedos de la pinza se mueven simétricamente respecto al centro. Definido matemáticamente el elemento 2 de cada pinza se mueve un número igual de unidades al elemento 1 pero con el signo contrario. Una vez añadida esta relación a la configuración cinemática de la garra el modelo cinemático de esta queda concluido. En la figura 21 se muestra la implementación de esta restricción.

Figura 21. Cuadro de dialogo para introducir restricciones a las articulaciones.

Una vez finalizado el modelo cinemático de la garra, antes de concluir su modelado definitivo, se van a añadir algunos elementos que faciliten posteriormente trabajar en PS. El primero de ellos añadir los TCP (*Tool Center Points*) de la garra. Se añaden 3 TCP (figura 22) que indican la base y las dos pinzas. Los TCP son ejes de referencia utilizados para el posicionamiento de los robots en el espacio cartesiano. Estos se usarán más tarde para referenciar posiciones respecto a otros ejes de referencia facilitando el posicionamiento del robot. Por ejemplo, en el *pick* de la tapa exterior se posiciona el TCP de la pinza correspondiente con el eje de referencia de las tapas, quedando así perfectamente alineados. El TCP colocado en la base de la garra se utiliza en los movimientos por el escenario que no impliquen el *pick* o *place* de piezas para facilitar eludir obstáculos.

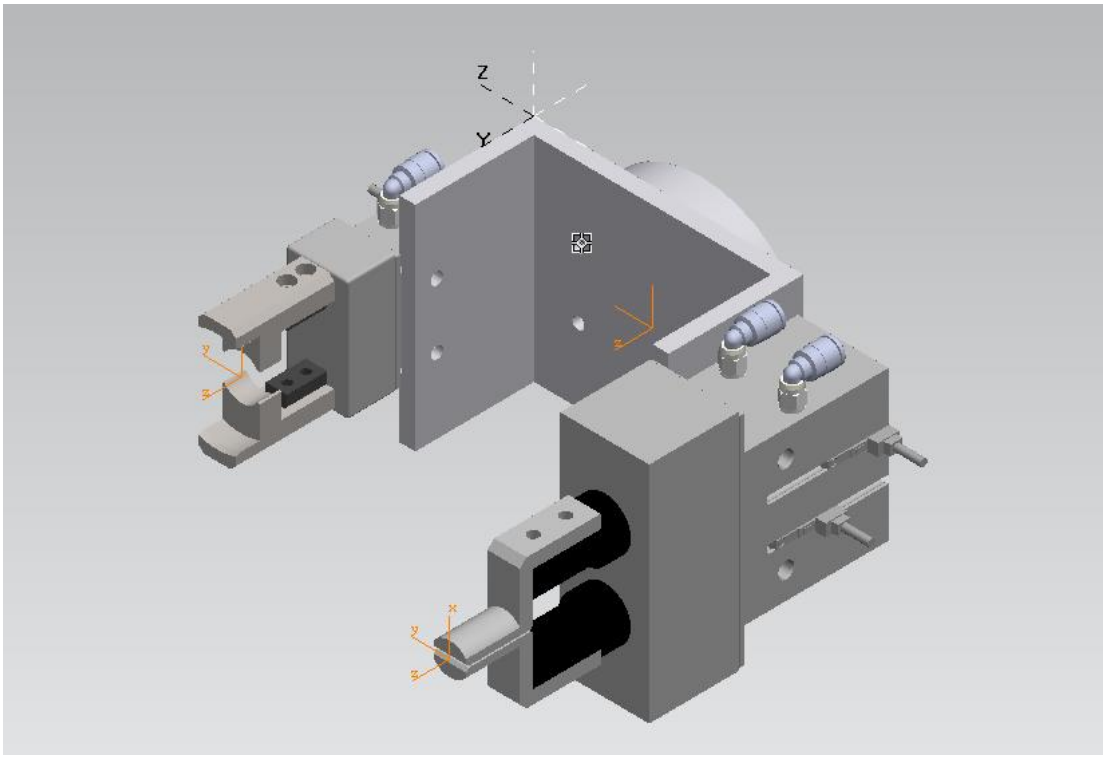


Figura 22. TCPs de la garra. Ejes de coordenadas naranjas, pinza izquierda TCP bulón, pinza derecha TCP tapas y centro TCP de la base de la garra.

Lo segundo que se realiza es añadir las posiciones de abierto y cerrado de cada pinza para simular la apertura y cierre de las pinzas. Para ello en la herramienta *Pose Editor* de Tecnomatix PS se añaden diferentes *poses* de las pinzas con los valores de las articulaciones correspondiente a cada pieza. Estas posiciones se usan después en los comandos OLP del programa robot (comandos para la simulación que no se exportan después al robot) para ordenar la apertura y cierre de las pinzas. En total se añaden 4 posiciones: HOME, Bulon_CLOSED, Rodamiento_CLOSED y TapaExt_CLOSED. La primera es la garra en su posición normal con los dos dedos abiertos. Rodamiento_CLOSED tanto para los rodamientos, como para las tapas interiores. TapaExt_CLOSED para las tapas exteriores, ya que el diámetro es un poco mayor a las tapas interiores. Por último, Bulon_CLOSED se usa tanto para los bulones, como para los pallets. De esta forma todos los elementos de la garra quedan modelados. Aunque no corresponda estrictamente a la cinemática, para los elementos gripper es necesario especificar una lista de cuáles son los elementos con los que puede interactuar la garra. En la figura 23 se muestra el cuadro de dialogo para generar una *pose*, en este caso, para coger rodamientos y tapas.

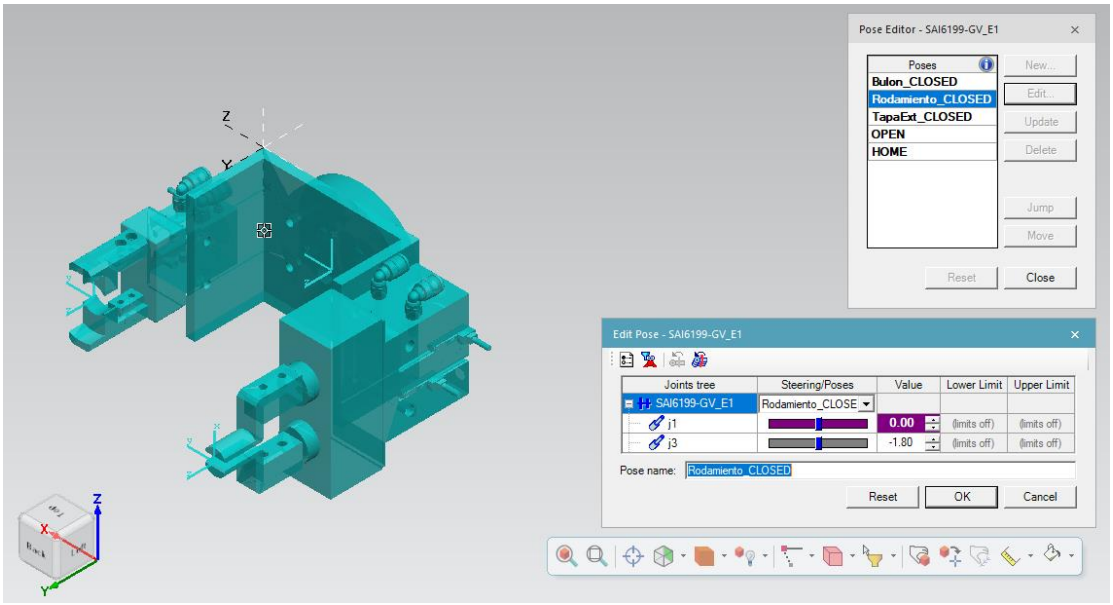


Figura 23. Creación de las *poses* de la garra.

Para el AGV se va a realizar una configuración análoga. El AGV está compuesto por una base y una mesa que sube y baja linealmente para facilitar el acceso del robot al pallet. Por tanto, para definir primero la cinemática de la mesa se elige como elemento fijo la base y se enlaza con una articulación prismática en el eje z global la mesa. A continuación, se definen los límites superior e inferior del desplazamiento. Se añade también un TCP en el lugar de colocación de los pallets para facilitar que se creen en el lugar correcto, y facilitar posicionar el robot para la carga y descarga. Por último, se marcan las posiciones de la mesa tanto en su posición superior como inferior. De esta manera los elementos móviles de la estación quedan definidos.

7.2.3 AGV

Realizar únicamente la configuración cinemática del AGV no nos permite manipularlo. Para poder manejar el AGV es necesario declararlo como tal. Desde la versión 15.1 de Tecnomatix PS es posible definir un elemento como AGV. Declararlo crea tanto los recursos lógicos como las señales necesarias. Sin embrago, el modelado de AGV que nos permite Tecnomatix PS es limitado. Las opciones disponibles son cerradas, por tanto, no es posible realizar un modelado caracterizado al elemento real, por lo que no se puede volcar el desarrollo como con los robots. Aun así, las herramientas disponibles son suficientes para realizar una simulación precisa para este proyecto.

En PS hay tres tipos de elementos relacionados con los AGV. Por una parte, se dispone de los propios AGV. Al definir un elemento como AGV se crean los bloques lógicos y señales para el control de este. El control se puede realizar desde el propio programa de Tecnomatix PS o con un programa de

automatización desde un PLC. En segundo lugar, tenemos los *targets*. Estos son TCP identificados especialmente para interactuar con el AGV, contienen un identificador ID único con el que se asigna el *target* al que se tiene que desplazar al PLC. Por último, la *carpet* crea un trazado bidimensional por el que se desplaza el AGV.

En el modelo de la célula se va a crear un trazado recto que una las dos estaciones con cuatro targets en el trazado, con una *carpet* que una los *targets*. Dos de ellos estarán ubicados enfrente de las estaciones, y los otros dos objetivos en dos puntos que simulen la entrada y salida de material. El modelo del AGV es el KMP 1500 de KUKA (figura 24), recurso facilitado por el fabricante.

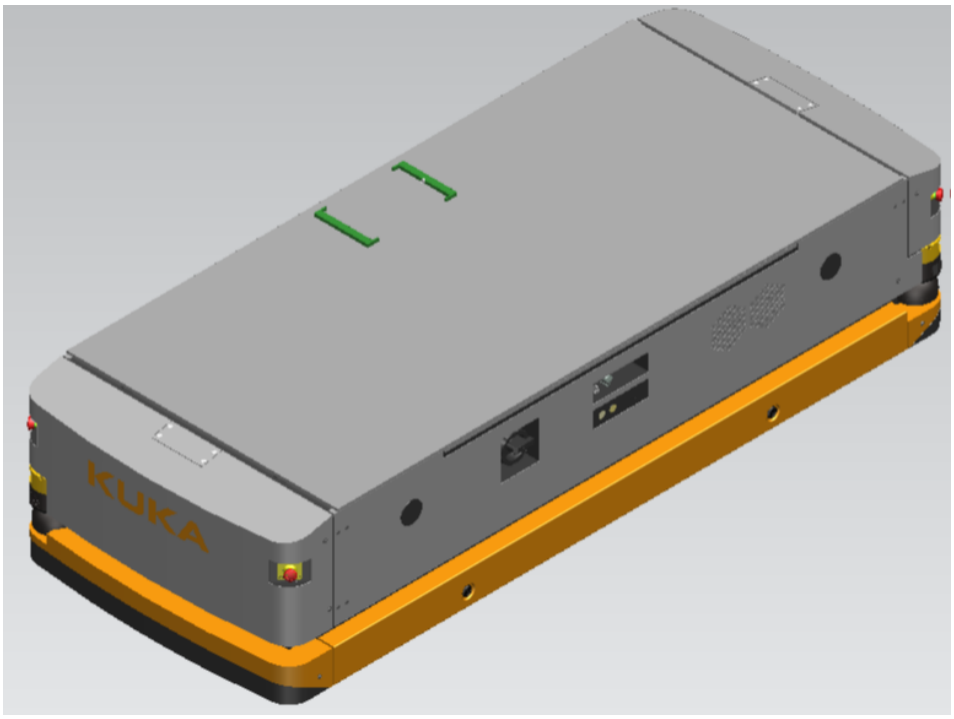


Figura 24. Modelo del AGV KMP 1500 de KUKA.

Se ha comentado previamente que las opciones que ofrece Tecnomatix PS para la simulación del AGV son limitadas y cerradas. A continuación, se van a detallar las principales funciones usadas en el proyecto para poder realizar posteriormente el control mediante PLC. En la tabla 6 se muestran estas funciones:

Tabla 6. Variables del AGV en Tecnomatix PS.

Nombre	*Tipo*	*I/Q*	*Uso*
MotionPlannerType	INT	Q	0-Sigue la carpet 1-Ignora la carpet
PathPlannerType	INT	Q	0-Almacena los *targets* uno tras otro en un *buffer* 1-Suprime el objetivo actual
RotationMethod	INT	Q	0-Moverse hacia adelante 1-Moverse hacia atrás 2-Mantener orientación
Speed	REAL	Q	Velocidad en mm/s
TargetIndex	INT	Q	ID del objetivo
CurrentTarget	INT	I	ID del objetivo actual
OnTarget	BOOL	I	True mientras el AGV este en el objetivo

Tanto el *MotionPlannerType* como el *PathPlannerType* se les asigna por defecto 0, es decir, seguir la *carpet* y almacenar los objetivos. Al *RotationMethod* se le asigna el valor correspondiente a mantener la orientación durante toda la trayectoria. A la velocidad se va a asignar un valor de 500 mm/s. Modificando desde el PLC el valor del *TargetIndex* se va a controlar el objetivo del AGV. Por último, *CurrentTarget* y *OnTarget* se van a usar como entradas al PLC para conocer la situación del AGV.

7.2.4 Operaciones

Tras modelar el escenario y la cinemática de los componentes, el siguiente paso es definir las Operaciones que se llevan a cabo. Tecnomatix PS cuenta con diferentes tipos de operación que se pueden realizar. Se tiene, por un lado, Operaciones relativas a la creación y consumo de las partes, así como generación de trayectorias para estas partes. Con estas Operaciones se va a realizar la creación de *parts* consumibles, y simulaciones de trayectorias de estas como en la aparición de *parts* en las rampas de los alimentadores o la caída de las tapas al soltarlas del robot. También existen las llamadas *Device Operation* que se van a usar en este proyecto para subir y bajar la mesa del AGV. Por último, se encuentran las Operaciones robot. Dentro de esta categoría tenemos a su vez varios tipos de Operaciones como soldadura, pick & place, etc. Aunque estas operaciones facilitan la implementación, para tener un control total de las operaciones, se van a implementar mediante Operaciones genéricas de robot que permite crear las Operaciones manualmente.

Los pallets con los casos de entrada se implementan mediante *Non-Sim Operation*. Para implementarlo se selecciona qué partes están enlazadas con la Operación y se especifica cual debe ser el lugar de aparición. En este caso, los casos de entrada se generarán encima de AGV, mientras que los pallets de tapas en el sitio correspondiente en cada estación. Con ello se crea una señal que permite activar la operación generando el caso seleccionado. Más tarde estas señales se mapearán con el PLC y el control de la generación de partes se realizará con el controlador. Por tanto, se cuenta con 60 operaciones de este tipo para cada caso con cada combinación de numero de ítems, más 6 operaciones en cada estación para la creación de las combinaciones de pallets con tapas exteriores. Además de una que se encargara de borrar todos los elementos creados. En total 73 operaciones. En la figura 25 se muestra el cuadro de dialogo para asignar una *part* como producto de una operación, de esta manera se generan las *parts.*

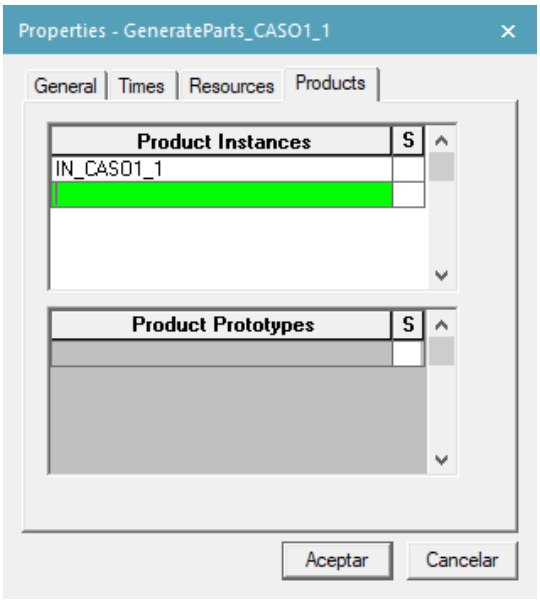


Figura 25. Cuadro de dialogo para introducir *parts* como productos de Operaciones.

Para los bulones, rodamientos, y tapas interiores, se les ha añadido el movimiento para simular los movimientos en los alimentadores. Para ello se ha seleccionado *Object Flow Operation* que permite que la parte siga una trayectoria. Igual que en el caso anterior, se debe especificar qué partes toman parte en la operación y donde han de crearse las partes creándose una señal para activar la operación. Además, hay que configurar cuales son los puntos por los que pasa la trayectoria del objeto y cuál es el tiempo en el que se tiene que recorrer dicha trayectoria. En este caso tendremos 6 operaciones por cada tipo de parte y estación, por lo tanto, 36 operaciones de flujo de objetos en total. En la figura 26 se muestra la realización del recorrido para la creación y la animación de descenso por la rampa par las tapas interiores.

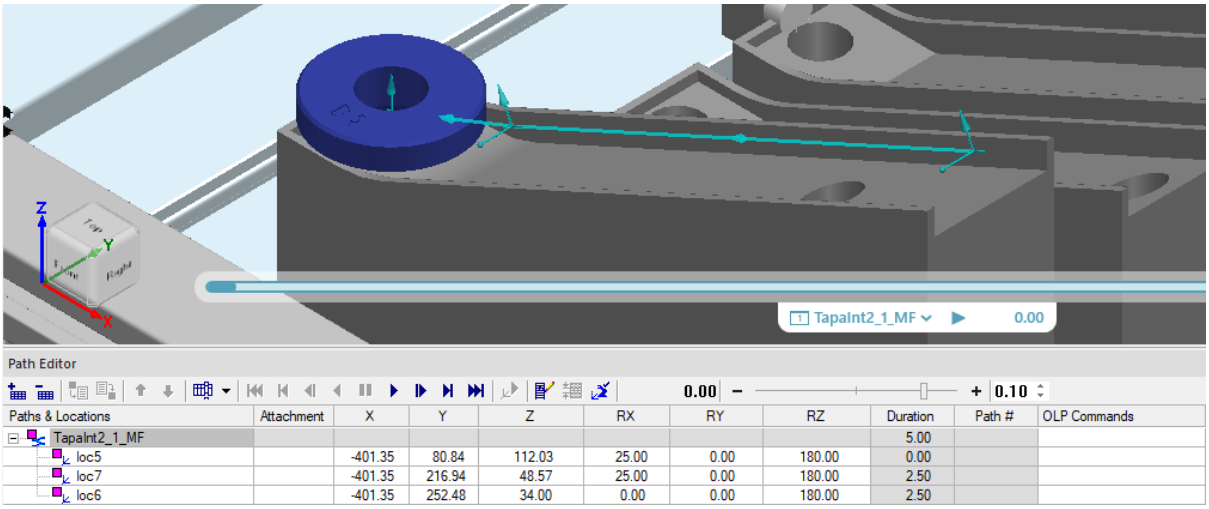


Figura 26. Generación y recorrido de tapas interiores.

Para crear las Operaciones del robot es esencial diseñar la estructura de programa robot. Las Operaciones de robot creadas en Tecnomatix PS se van a traducir posteriormente al lenguaje de programación del robot, por lo tanto, es conveniente pensar estas operaciones como parte del programa robot. Para este proyecto, dadas sus características, se ha decidido fragmentar las operaciones en operaciones sencillas análogas a los subprogramas en el código tradicional. A continuación, se describen las operaciones desarrolladas:

- OP1000 y OP100: Recogida del pallet desde el AGV a la zona de entrada y transportar el pallet desde la zona de entrada a la de montaje respectivamente.
- OP2000 y OP200: Depositar el pallet en el AGV desde la zona de salida y trasporte del pallet desde la zona de montaje a la zona de salida respectivamente.
- PICK_(RODAMIENTO/BULON/TAPA_INT): Recoger las partes del correspondiente alimentador.
- PLACE_(RODAMIENTO/BULON/TAPA_INT)_POS(1..6): Colocar la parte en la posición correspondiente.
- PICK_TAPA_EXT_POS(1..6) y PLACE_TAPA_EXT_POS(1..6): Las tapas exteriores no tienen un punto de alimentación común, por lo que el pick y el place tienen 6 operaciones.

Al contrario que muchos programas de programación, Tecnomatix PS no permite utilizar comandos como SHIFT en VAL II que permite desplazar puntos. Por lo tanto, es necesario crear operaciones diferentes y no ir cambiando el punto de pick o place con esta operación. Esta limitación de Tecnomatix PS hace que el código sea más largo.

La estructura del programa funciona con llamadas a estas operaciones. El programa principal recibe desde el PLC vía Profinet el número de ítems, el tipo de servicio y la señal de comienzo. Si el robot está habilitado ejecuta las operaciones de entrada, si no ejecuta un bucle vacío hasta que esté listo el robot. En función del servicio requerido se llama a un subprograma que ejecuta ese servicio. Este en función del número de ítems llama a los subprogramas requeridos para realizar el servicio correctamente por último se ejecutan las operaciones de salida. En la figura 27 se muestra un ejemplo del pseudocódigo para realizar el servicio 1. En se ve como en función del número de ítems se llama a diferentes programas y se manipulan distintas señales.

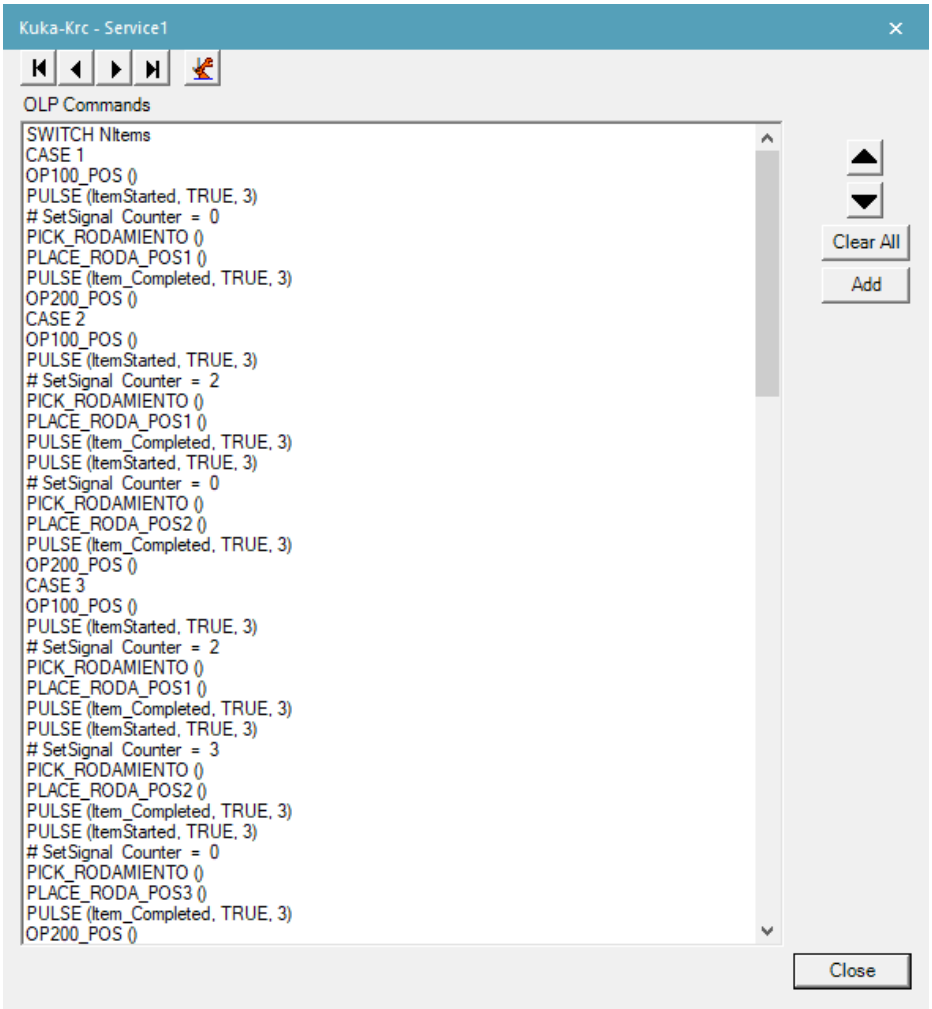
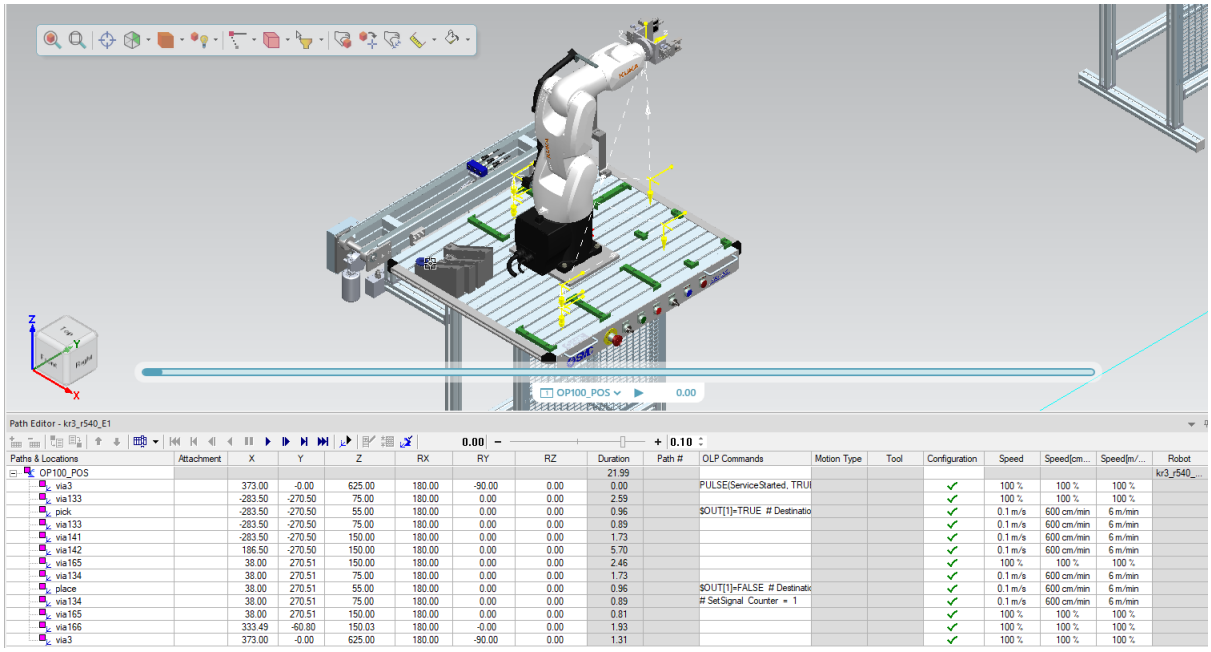


Figura 27. Ejemplo de pseudocódigo para el servicio 1.

Estas operaciones contienen los puntos de la trayectoria, comandos de abrir y cerrar pinzas y manejo de señales. A continuación, se recogen los campos a definir para las operaciones. Toda la configuración se realiza en el *Path Editor*, que además de servir para añadir los comandos, también permite hacer una simulación de los movimientos de la operación. Dentro de la operación se añaden los puntos

necesarios para generar las trayectorias. Al resolver el problema de posición inverso, en los robots de 6 ejes existen más de una solución para una posición especifica. Por lo tanto, hay que indicar en que configuración queremos acceder al punto. Durante toda la trayectoria hay que asegurarse que las configuraciones seleccionadas no crean movimientos innecesarios por la configuración con la que llegan al punto. PS contiene una herramienta *teach configuration* que ayuda a realizar la elección de la configuración. También se tiene que elegir si el movimiento es lineal o *point to point*. Con el segundo tenemos movimientos más rápidos, pero menos precisos. Las operaciones de aproximación siempre serás movimientos lineales. También es posible que durante la trayectoria haya que utilizar movimientos lineales para evitar colisiones. Otro de los aspectos a definir es la velocidad máxima y aceleración. Si no se especifican por defecto toman el valor máximo, pero para movimientos delicados como aproximaciones se reduce la velocidad para ganar precisión. Por último, en la columna de comandos OLP se puede añadir comandos como abrir y cerrar pinzas, y activar y desactivar señales. Por ejemplo, al comenzar un servicio o ítem o finalizar un servicio o ítem, se envía un pulso de 3 segundos para que el PLC sepa el estado de la operación y comunicarlo con el sistema de agentes. También hay que añadir en esta pestaña comandos para la simulación como *attach* para hacer que las piezas se muevan junto a la garra y simular así la manipulación de las piezas. Estos comandos posteriormente no serán volcados al robot real, puesto que solo se implantan para realizar una mejor simulación y añadirlo causaría errores de compilación. En la figura 28 se muestra la OP100 que se encarga de trasladar el pallet desde la zona de entrada a la de montaje. En la lista se observa los puntos de la trayectoria con sus valores, los comandos OLP que se ejecutan durante la trayectoria y la configuración con la que se acerca al punto y sus velocidades. En la imagen 3D se puede observar los puntos y la trayectoria de la Operación.



Paths & Locations	Attachment	X	Y	Z	RX	RY	RZ	Duration	Path #	OLP Commands	Motion Type	Tool	Configuration	Speed	Speed[cm...	Speed[m/...	Robot
OP100_POS								21.99									kr3_r540_...
via3		373.00	-0.00	625.00	180.00	-90.00	0.00	0.00		PULSE(ServiceStarted, TRUI			✓	100 %	100 %	100 %	
via133		-283.50	-270.50	75.00	180.00	0.00	0.00	2.59					✓	100 %	100 %	100 %	
pick		-283.50	-270.50	55.00	180.00	0.00	0.00	0.96		$OUT[1]=TRUE # Destinatio			✓	0.1 m/s	600 cm/min	6 m/min	
via133		-283.50	-270.50	75.00	180.00	0.00	0.00	0.89					✓	0.1 m/s	600 cm/min	6 m/min	
via141		-283.50	-270.50	150.00	180.00	0.00	0.00	1.73					✓	0.1 m/s	600 cm/min	6 m/min	
via142		186.50	-270.50	150.00	180.00	0.00	0.00	5.70					✓	0.1 m/s	600 cm/min	6 m/min	
via165		38.00	270.51	150.00	180.00	0.00	0.00	2.46					✓	100 %	100 %	100 %	
via134		38.00	270.51	75.00	180.00	0.00	0.00	1.73					✓	0.1 m/s	600 cm/min	6 m/min	
place		38.00	270.51	55.00	180.00	0.00	0.00	0.96		$OUT[1]=FALSE # Destinatic			✓	0.1 m/s	600 cm/min	6 m/min	
via134		38.00	270.51	75.00	180.00	0.00	0.00	0.89		# SetSignal Counter = 1			✓	0.1 m/s	600 cm/min	6 m/min	
via165		38.00	270.51	150.00	180.00	0.00	0.00	0.81					✓	100 %	100 %	100 %	
via166		333.49	-60.80	150.03	180.00	-0.00	0.00	1.93					✓	100 %	100 %	100 %	
via3		373.00	-0.00	625.00	180.00	-90.00	0.00	1.31					✓	100 %	100 %	100 %	

Figura 28. Puntos de la trayectoria de la OP_100. Comandos OLP y confirucaiones.

7.2.5 Programa robot

Una vez realizadas las Operaciones, el siguiente paso es la configuración del tipo de unidad de control del robot. Por defecto, al introducir un robot Tecnomatix PS le asigna un tipo de “controladora estándar” desarrollada por Siemens que solo se usa para este programa. Esta controladora es suficiente para realizar las primeras simulaciones y pruebas. Para poder usar la capacidad de volcar los programas robots en los lenguajes de las controladoras reales es necesario seleccionar la unidad de control como la del robot real. El programa sigue ejecutando el pseudocódigo igual que antes, pero al exportar el programa robot los archivos generados son traducidos al lenguaje del robot. Tecnomatix PS ofrece un catálogo amplio de unidades de control entre las que elegir. Cada una de ellas cuanta con una serie de parámetros propios en función del fabricante. Cada fabricante también tiene un lenguaje de programación diferente y una serie de señales necesarias. Entre las opciones disponibles se encuentra KUKA-KRC que es compatible con la controladora KRC C4 elegida para el proyecto. En la figura 29 se muestra el cuadro de dialogo donde se selecciona la unidad de control.

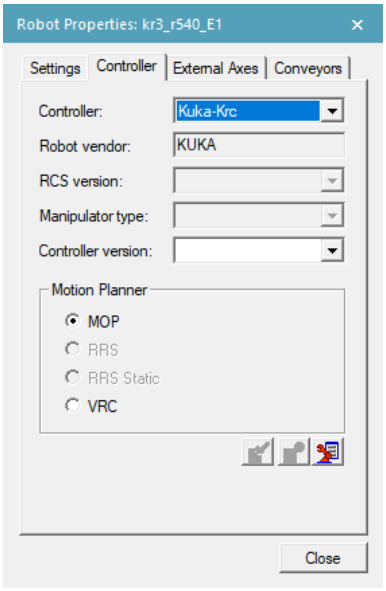


Figura 29. Cuadro de dialogo de propiedades robot.

A continuación, hay que configurar los parámetros específicos de la controladora. Primero se deben asignar los TCP con los que se debe trabajar. Para ello, se hace uso de los TCP creados anteriormente durante el modelado cinemático. Por tanto, se usan por un lado el TCP en la base de la garra, por otro, dos más, cada uno en su correspondiente pinza. En este momento se debe asignar también el TCP localizado en la base del robot para actuar como origen del robot.

El siguiente paso es crear las señales del robot para el control de las operaciones. Al elegir la controladora, se crean las señales del robot por defecto. Aun así, suele ser habitual tener que crear más señales para poder controlar de manera correcta el flujo de las operaciones. Desde el programa robot solo se pueden manejar señales declaradas en el robot. Sin embargo, en el mapeado con el PLC las señales pueden declararse fuera. En la tabla 7 se muestran las señales del robot:

Tabla 7. Señales del robot, tipo, I/O y breve descripción.

SEÑAL	*TIPO*	*I/O*	*DESCRIPCIÓN*
$EXT_START	BOOL	Q	INICIO DE LA EJECUCIÓN DEL PROGRAMA SELECCIONADO EN LA UNIDAD DE CONTROL DEL ROBOT.
$MOVE_ENABLE	BOOL	Q	HABILITACIÓN DE MOVIMIENTO DEL ROBOT
$DRIVES_OFF	BOOL	Q	HABILITACIÓN DE LOS ACCIONAMIENTOS DEL ROBOT
$DRIVES_ON	BOOL	Q	CONEXIÓN DE LOS ACCIONAMIENTOS DEL ROBOT
$IN_HOME1	BOOL	Q	ROBOT EN POSICIÓN HOME
KR3R540_PROGRAMNUMBER	BYTE	Q	NÚMERO DE PROGRAMA A EJECUTAR POR EL ROBOT
NUMBEROFITEMS	BYTE	Q	NÚMERO DE ÍTEMS QUE DEBE ENSAMBLAR EL ROBOT
SERVICETYPE	BYTE	Q	TIPO DE SERVICIO QUE DEBE REALIZAR EL ROBOT
NEW_SERVICE	BOOL	Q	ORDEN DE EJECUCIÓN DEL PEDIDO
ITEMSTARTED	BOOL	I	INDICA EL INICIA DE MONTAJE DE UN ÍTEM
ITEM_COMPLETED	BOOL	I	MONTAJE DE UN ÍTEM COMPLETADO
ITEM_COMPLETED_NUMBER	BYTE	I	NÚMERO DEL ÍTEM COMPLETADO EN LA ÚLTIMA OPERACIÓN DE ENSAMBLADO
SERVICESTARTED	BOOL	I	INICIO DE SERVICIO
SERVICE_COMPLETED	BOOL	I	FINAL DE SERVICIO
CLOSE_BULON	BOOL	I	CIERRE DE LA PINZA PARA MANIPULAR BULONES Y PALLETS
CLOSE_RODAMIENTO	BOOL	I	CIERRE DE LA PINZA PARA MANIPULAR RODAMIENTOS Y TAPAS INTERIORES
CLOSE_TAPAEXT	BOOL	I	CIERRE DE LA PINZA PARA MANIPULAR TAPAS EXTERIORES.

Las limitaciones de Tecnomatix PS plantean un problema a resolver. Las señales de apertura y cierre de pinzas deben estar declaradas junto con la garra. Sin embargo, debemos actuar sobre estas señales desde el robot, pero desde el robot solo podemos actuar sobre señales declaradas en el robot. Por lo tanto, debemos relacionar señales dentro y fuera del robot. Para solucionarlo se va a implementar un "bloque lógico" que una estas señales. En las entradas del bloque lógico van a estar las señales que vienen del robot, y en la salida las señales que manejan las pinzas. Dentro del bloque lógico simplemente se asigna a la señal correspondiente de la pinza la señal correspondiente que proviene del robot, creando así la lógica que controla el bloque. En la figura 30 se muestra esta unión entre

señales, en la parte izquierda se cuenta con las señales que provienen del robot, y el "bloque lógico" las une con las de la parte derecha que son señales de apertura y cierre de la pinza.

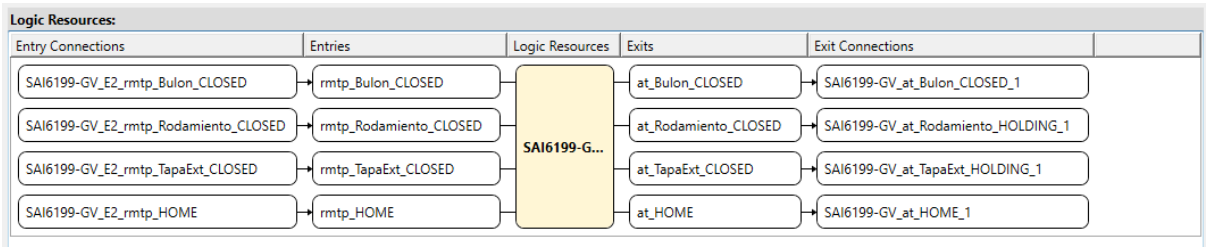


Figura 30. "Bloque lógico" de la pinza.

7.2.6 Flujo de material

Para realizar la simulación y puesta en marcha virtual, debemos trabajar en *Line Simulation Mode*. Para poder realizar la simulación en este modo es necesario determinar cómo va a ser el flujo de materiales del proceso. En este caso, el flujo de las *parts* durante las diferentes etapas de la fabricación. Cuando trabajamos en este modo de funcionamiento las *parts* desaparecen y en su lugar se generan appearances. Las appearances son copias de las partes que representan y se generan mediante las *Non-Sim Operations* realizadas previamente.

Para poder realizar una simulación realista también necesitaremos introducir unos sensores de presencia en los alimentadores. De lo contrario, todas las piezas del mismo tipo se crearían sin control. Para generar estos sensores debemos definir la lista de partes que debe detectar, y como de largo va a ser el rango en el que funcione el sensor. El sensor tiene asignada una señal booleana que se utiliza para saber el estado del sensor. Cuando hay un flanco negativo en el sensor se genera una nueva parte en el alimentador.

Una vez todas las operaciones están listas se modeliza el control de flujo de materiales. Para ello se debe abre el visor *Material Flow*. A continuación, han de añadirse las operaciones que forman parte del flujo de materiales. No es necesario introducir todas las señales, solo aquellas que generan o consumen *appearances* o modifican las partes, por ejemplo, los *attach*.

En la figura se muestra el *Material Flow* para este proyecto. La secuencia comienza con la creación de los casos en el AGV. El AGV comienza con la mesa en la posición superior puesto que el *attach* entre el caso y el AGV se realiza al comienzo de la operación de bajada de la mesa. Es por ello, que todos los casos convergen en esa operación de bajada. Al final de la operación de subida de la mesa se encuentra el comando *detach* que rompe el enlace anterior. Estas operaciones no son necesarias incluirlas en el *Material Flow,* aunque formen parte del flujo del material. después subir la mesa el

robot de la primera estación lleva el pallet a la zona de montaje y allí se crean las piezas del conjunto. Las operaciones de generación de piezas también ha de incluirse en el "Material Flow". Las operaciones de las tapas exteriores e interiores fluyen hacia las operaciones que simulan el caer de la tapa por la gravedad. Posteriormente, todas las operaciones confluyen en la operación de salida, donde se encuentra el *attach* que enlaza todas las piezas del conjunto con el caso de entrada. Tras finalizar depositar el pallet en el AGV de nuevo, el flujo avanza hacia la operación de bajada del AGV para realizar de nuevo el enlace entre el pallet con las piezas y el AGV. Por limitaciones de Tecnomatix PS no se puede utilizar de nuevo la misma operación de bajada, por tanto, se ha creado una nueva con la misma funcionalidad. También hay una unión entre las piezas que han quedado en los alimentadores y la operación *kill* para eliminar estas piezas al final de la secuencia. En la figura 32 se muestra el *Material Flow* de la primera estación.

Una vez finalizada la secuencia de la primera estación el AGV avanza hacia el siguiente y se lleva a cabo un flujo similar. En este caso no es necesario crear nuevos casos puesto que llega el pallet de la primera estación. Las operaciones de generación de piezas confluyen de nuevo hacia la operación de salida de la segunda estación. Y de nuevo el flujo avanza o bien hacia la operación de la bajada de la mesa, o bien hacia la operación de eliminación de partes. Una vez enlazados el pallet y las partes nuevas el AGV avanza hasta la posición final y se eliminan todos los componentes generados. Debido a limitaciones de Tecnomatix PS, el *Material Flow* no permite crear bucles cerrados. Por tanto, no es posible realizar en una misma ejecución, hacer que el montaje comience en la estación uno y finalice en la segunda, y a continuación en el orden inverso. En caso de querer cambiar el orden de montaje, se ha de modificar el orden de las estaciones en el *Material Flow*. Aunque este inconveniente reduce las posibilidades de la simulación, modificando el flujo del material es posible comprobar la validez de las dos posibilidades. En la figura 33 se muestra la segunda mitad del *Material Flow*, el correspondiente a la estación 2.

Para realizar la validación del modelo es posible utilizar la simulación en CEE "Ciclic Event Evaluation" (figura 31). Utilizando esta herramienta Tecnomatix PS controla todas las partes del modelo incluyendo entradas y salidas, las cuales forzamos desde el panel de simulación. El objetivo final es que las entradas y salidas se controlen con el proyecto de control PLC. Sin embargo, en este estado de desarrollo donde aún no es posible permite comprobar el modelo desarrollado. Una vez comprobado el modelado de la célula queda completado a la espera de desarrollar el proyecto de automatización para realizar la validación.

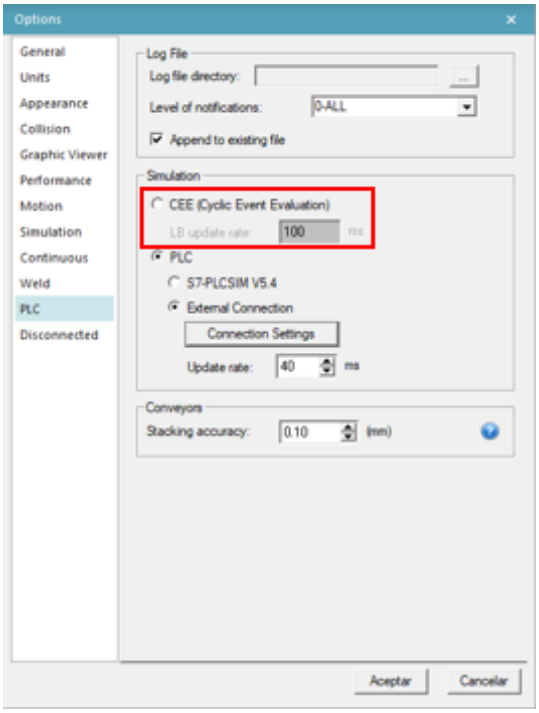


Figura 31. Cuadro de dialogo para seleccionar la simulación CEE.

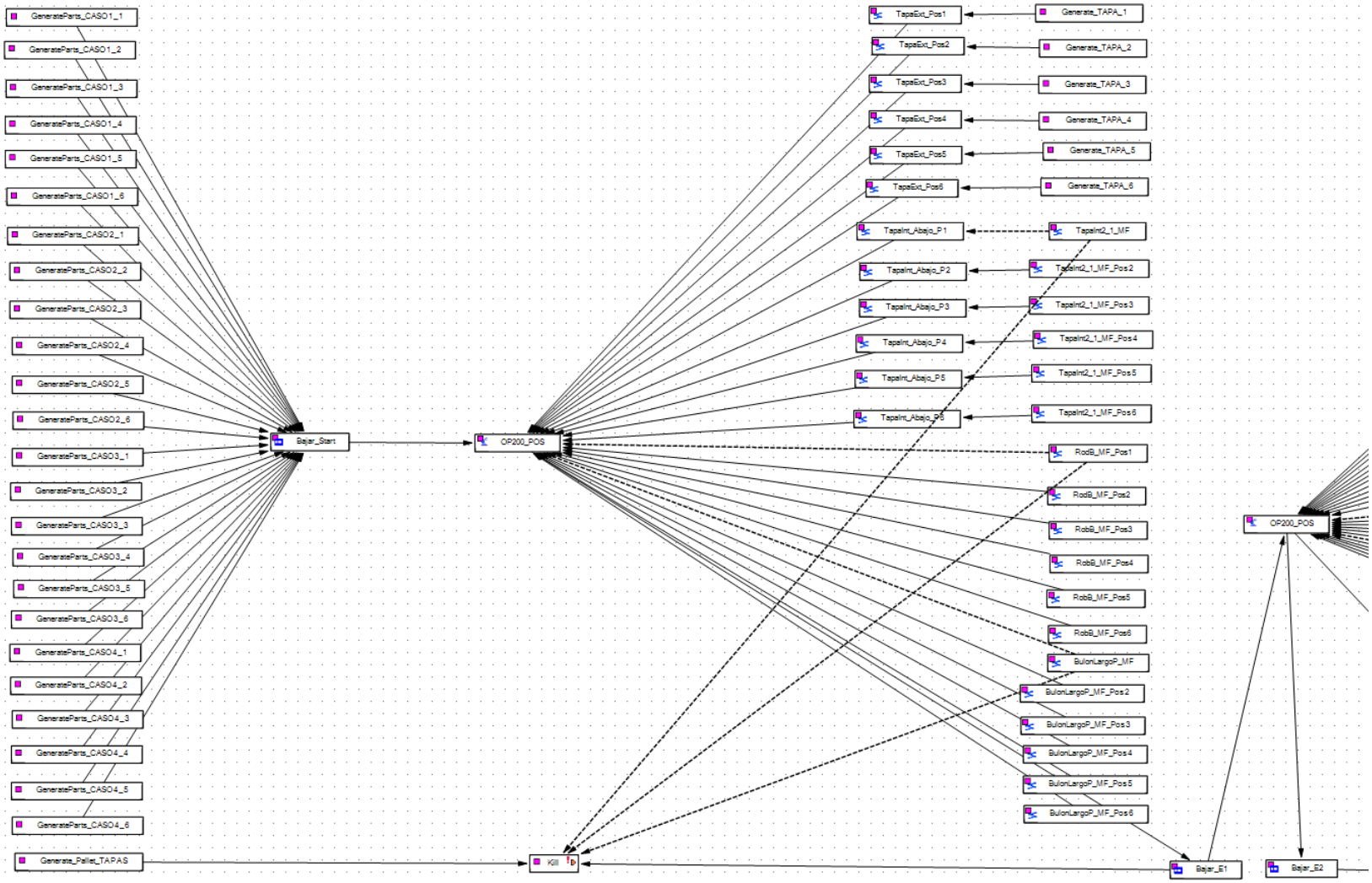


Figura 32. *Material Flow* estación 1.

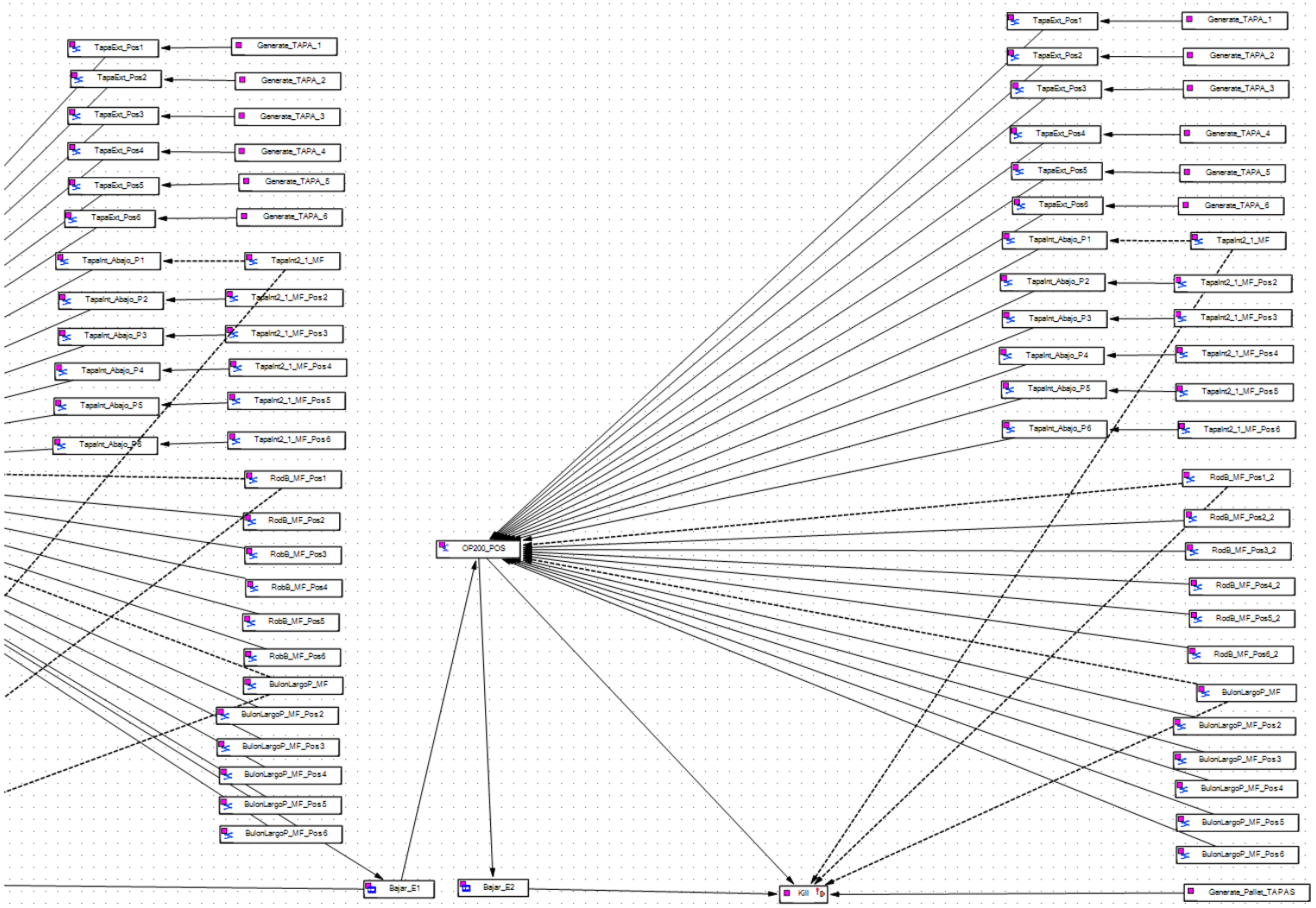


Figura 33. *Material Flow* de la estación 2.

7.3 Proyecto de automatización

En este apartado se expone el diseño y realización del software de automatización. En este proyecto se encuentra la peculiaridad de que para integrar los sistemas multiagente es necesario recurrir a software ejecutándose en Windows. Por tanto, es necesario incluir un PLC que tenga esa característica. Por integración con las otras herramientas de la plataforma PLM de Siemens se va a recurrir a este fabricante para la solución de automatización. Por lo general, los PLC del catálogo de Siemens no cuenta con una parte Windows un la que ejecutar programación en lenguajes de alto nivel. Otros fabricantes como Beckhoff cuentan con esta funcionalidad en todos sus PLC puesto que nativamente se ejecutan sobre Windows. Siemens cuenta con una gama de productos de su catálogo enfocada a estas necesidades. Para este proyecto se ha elegido para la simulación la CPU 1518-4 PN/DP ODK que como su nombre indica integra la implementación de ODK (*Open Development Kit*) en simulación (figura 34). El controlador para el PLC real es el ET200 SP. En la practica el hecho de que los PLC sean diferentes no tiene ningún efecto a la hora de desarrollar el código de automatización puesto que tienen las mismas funcionalidades. PLCSIM Advance limita las opciones de la simulación a la gama Simatic S7-1500 de Siemens, por ello se presenta esta diferencia en los controladores.

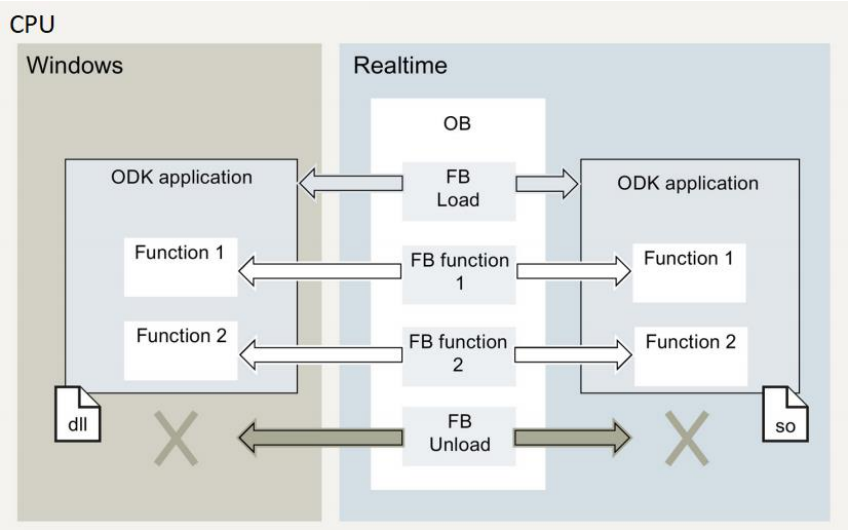


Figura 34. Esquema de la ejecución de aplicaciones ODK.

Los PLC con Windows integrado tienen más ventajas respecto a los PLC tradicionales. La principal, como ya se ha comentado, es poder incluir código desarrollado en lenguajes de alto nivel, generalmente C/C++ en la parte Windows. La otra, es poder integrar el *runtime* del HMI (*Human Machine Interface*) directamente en el PLC evitando así incluir otro dispositivo para este propósito. EL PLC, además, permite las funcionalidades básicas de los PLC como el manejo de entradas y salidas, capacidad de tiempo real, etc. El alcance de este proyecto no trata de diseñar y escribir el código ODK, si no de implementar la solución creada por los compañeros del grupo de investigación.

Los proyectos de automatización para la implementación real y simulación necesitan pequeñas modificaciones para ajustarse a las características de cada implementación. Para la simulación, es necesario incluir un modo de interactuar con el PLC puesto que no se cuenta con la posibilidad de usar los interruptores físicos. La solución implementada para la simulación consta de un HMI desde el cual validar esas acciones (figura 35). En la implementación real se encuentra la limitación de contar con una única estación robotizada y no contar con ningún tipo de AGV o plataforma móvil. Para este caso se va a implementar uno de los proyectos de las estaciones desarrollados para la simulación con las modificaciones pertinentes para funcionar en su entorno. Durante el resto del apartado se mencionarán las necesidades y modificaciones de ambas soluciones.

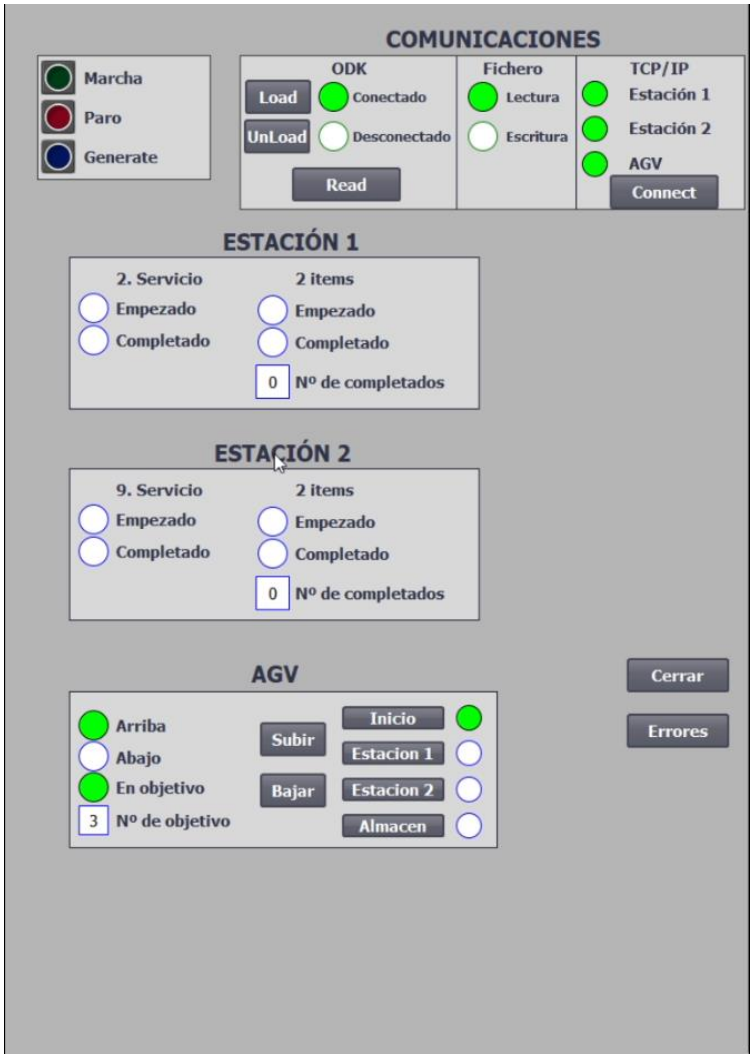


Figura 35. Captura del HMI del proyecto de control.

En la implementación de la parte de simulación contamos con 3 PLC diferentes. Dos de ellos para las estaciones robotizadas y uno para el control del AGV. Es posible hacer el control con un único PLC

siempre y cuando cumpla todos los requisitos. Sin embargo, distribuir los controladores presenta múltiples ventajas. La principal es que facilita la implementación de los controladores. Con un control distribuido es posible llevar el controlador directamente a cada estación, reduciendo el trabajo de cableado. Además, es más sencillo desarrollar el software en bibliotecas e insertarlo en cada controlador. Por lo tanto, es más sencillo implementar nuevas estaciones en la célula, puesto que añadir estaciones no implica modificar las estaciones previamente implementadas. En la estación real se va a llevar a cabo con este modelo distribuido la implementación de la biblioteca creada.

7.3.1 Comunicaciones

Como se ha comentado anteriormente, debido a la distribución de los controladores, en el proyecto hay 3 PLC. Los PLC necesitan conocer el estado de variables de los otros PLC para funcionar de forma coordinada. Por lo tanto, es necesario llevar a cabo una comunicación entre ellos. Ambos controladores de las estaciones deben conocer el estado del AGV para comenzar la secuencia, y deben comunicar cuando finaliza para que el AGV continúe la suya.

Para la comunicación existen varias posibilidades. Entre ellas se encuentras PROFIBUS o Ethernet Industrial, siendo esta última es la opción mayoritaria para nuevas instalaciones. Dentro de Ethernet Industrial encontramos opciones como TCP/IP o funciones S7 (figura 202). Las funciones S7 solo permiten estructuras homogéneas dentro de Simatic NET. Para permitir una mayor flexibilidad en el futuro la opción elegida es usar TCP/IP. TCP/IP se encuentra dentro del estándar IEC 61131-3 por lo que permite la conexión con soluciones no homogéneas con controladores de otros fabricantes. En este proyecto se realiza la comunicación mediante TCP/IP directamente, es decir, la opción C de la figura 36.

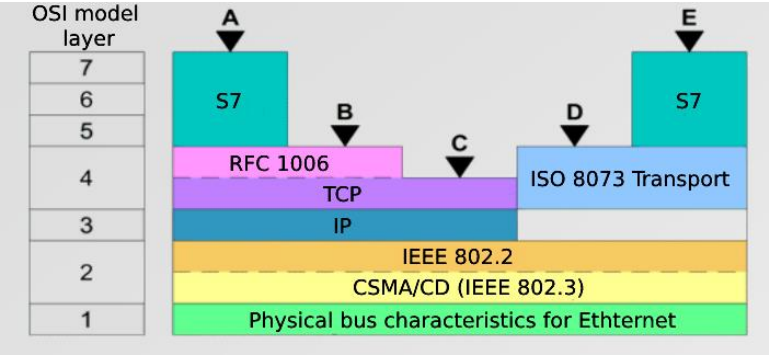


Figura 36. Capas OSI (*Open Systems Interconnection*) Comunicación S7 (A,E) y TCP/IP (C).

Para la simulación es necesario introducir un HMI para actuar sobre las entradas de los interruptores. Para ello es necesario crear enlaces de comunicación S7 entre el HMI y los 3 PLC. El *runtime* del HMI

puede ejecutarse en un dispositivo específico para ello. Sin embargo, en este proyecto se va a aprovechar la posibilidad de ejecutarse en el mismo PC. El HMI en este proyecto tiene como función poder interactuar con las variables de entrada. Por tanto, el diseño del mismo va a ser sencillo, pensando en la funcionalidad.

En la implementación real, solo se cuenta con un PLC, por lo que no es necesario comunicar 3 PLC. Tampoco es necesario incluir el HMI, puesto que contamos con los interruptores físicos y una tarjeta de entradas digitales. Sin embargo, es necesario añadir la comunicación PROFINET entre el PLC y el robot. En esta comunicación se reservan 254 bytes que se mapean a partir de la marca 2000 de la memoria. Algunas de estas direcciones están reservadas para variables del robot como habilitar los accionamientos o la liberación del movimiento. Las direcciones no reservadas pueden ser usadas con variables del usuario.

La comunicación de ODK con el sistema multiagente se maneja es los bloques ODK implementados. Aunque para este proyecto no es importante conocer cómo se realiza la comunicación con los demás agentes, cabe comentar que el protocolo utilizado es MQTT (*Message Queue Telemetry Transport*) que funciona como un protocolo publicista suscriptor. No obstante, si resulta importante conocer como es la estructura de la información para el intercambio que se utiliza en la comunicación para poder manejar sus variables (tabla 8).

Tabla 8. Variables de la estructura de información de comunicación ODK.

Nombre	Tipo
str2info	ODKProject7control_flags
Control_Flag_New_Service	Bool
Control_Flag_Item_Completed	Bool
Control_Flag_Service_Completed	Bool
str2PLC	ODKProject7agent2plc
Id_Machine_Reference	UDInt
Id_Order_Reference	UDInt
Id_Batch_Reference	UDInt
Id_Ref_Subproduct_Type	UDInt
Operation_Ref_Service_Type	UDInt
Operation_No_of_Items	UDInt
str2Agent	ODKPoject7plc2agent
Id_Machine_Reference	UDInt
Id_Order_Reference	UDInt
Id_Batch_Reference	UDInt
Id_Ref_Subproduct_Type	UDInt
Id_Ref_Service_Type	UDInt
Id_Item_Number	UDInt
Data_Initial_time_Stamp	LDT
Data_Final_time_Stamp	LDT
Data_Service_time_Stamp	LDT

Como se observa en la tabla 200, las variables en str2info controlan el flujo de la secuencia. Las variables dentro de str2PLC es la información que llega al PLC desde el sistema de agentes. Ciertas variables se usan para la trazabilidad, las dos últimas son importantes para el control puesto que indican el número de servicio y numero de ítems. Por último, str2Agent es la información que recibe el sistema de agentes desde el PLC. Contiene las mismas variables de trazabilidad, y marcas de tiempo de inicio y final de ítems y servicios.

7.3.2 Configuración del hardware del proyecto de automatización

Para realizar la configuración hardware para la implementación en simulación se introduce tres PLC 1518-4 PN/DP ODK, la CPU de la gama 1500S que permite usar ODK. A continuación, se introduce el

HMI que como se ha comentado antes no se ejecutara en otro dispositivo, si no en un PC. En este punto, una vez introducidos todos los dispositivos del proyecto, es importante comprobar que no hay IP repetidas entre los dispositivos y tarjetas de red que se van a usar posteriormente en el *virtual commissioning*.

Posteriormente, se indican las comunicaciones que se llevan a cabo. Primero, se realiza la conexión de todos los puertos Ethernet de los controladores y HMI indicando así que los dispositivos se encuentran en la misma subred. Para la conexión HMI además de esto hay que indicar el enlace HMI entre el HMI y cada PLC. Por tanto, se realizarán tres enlaces HMI de comunicaciones S7 (figura 37). Para realizar las conexiones TCP/IP se aprovecha la subred creada anteriormente y se indica las IP de las CPU a conectar y el puerto por el que realizarlo. En el caso de las estaciones del robot el puerto da igual puesto que solo tenemos una conexión de este tipo. Sin embargo, para el controlador del AGV debemos indicar dos puertos diferentes para la conexión de cada estación. Se indica también quien establece la conexión activa de la comunicación y se identifica con un ID que se utiliza para trabajar con esta conexión más adelante.

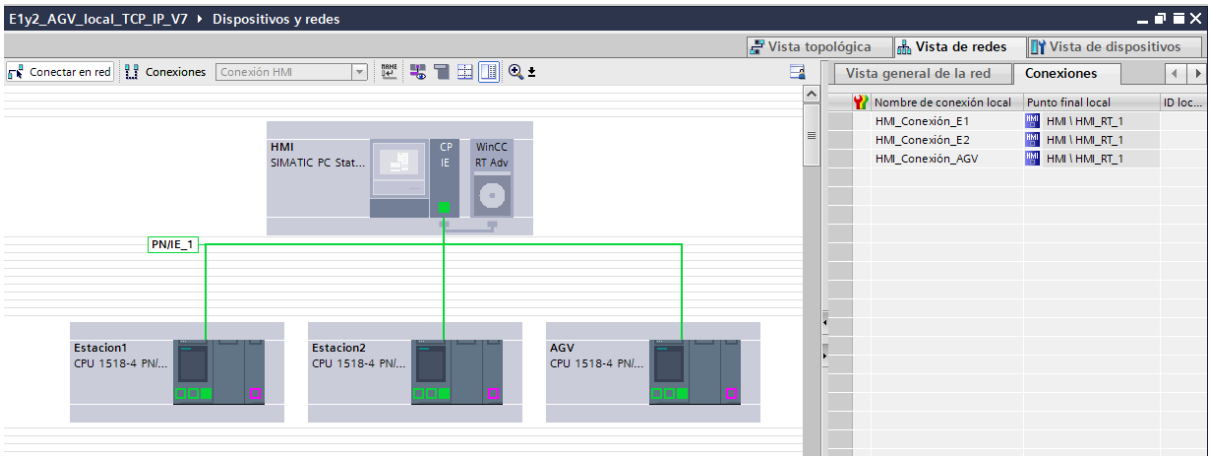


Figura 37. Vista de redes de la solución para la implementación en simulación.

Para la implementación real se introduce, como se ha comentado anteriormente, la CPU ET200 SP. Esta CPU contiene tanto el controlador en sí, como un HMI integrado, aunque este no se va a utilizar. Para poder observar el HMI la CPU contiene tanto salida de video para conectar un monitor, como puertos USB para insertar teclado y ratón. En este caso no hay que realizar los enlaces de comunicación TCP/IP puesto que solo tenemos un controlador. Sin embargo, en este caso es necesario introducir la controladora del robot para poder especificar la comunicación PROFINET (figura 38). La comunicación es PROFINET IO, donde la CPU actúa como Controlador IO y la unidad de control como Dispositivo IO. La controladora es la KR C4 Compact mencionada en apartados anteriores. La unidad

al no ser de Siemens en un principio no se encuentra en la librería de dispositivos incluida. Para poder insertar la controladora es preciso instalar el archivo GSDML correspondiente. Este archivo es un archivo GDS en formato XML que contiene la información de las capacidades básicas del dispositivo facilitado por KUKA. La manera más usual de realizar las comunicaciones es mediante dirección IP y puerto. Sin embargo, en PROFINET IO se realiza mediante nombre. Por tanto, es importante editar el nombre del elemento introducido y el de la configuración de la controladora para que ambos coincidan. A continuación, se indica la dirección y tamaño de los datos. Como se ha comentado en este apartado, se direccionan 254 bytes, es decir, 2032 bits de entradas y salidas digitales a partir de la dirección 2000 del PLC. Es importante recordar las direcciones asignadas a cada variable para realizar posteriormente la configuración robot y mapear correctamente las señales en entre PLC y controladora del robot

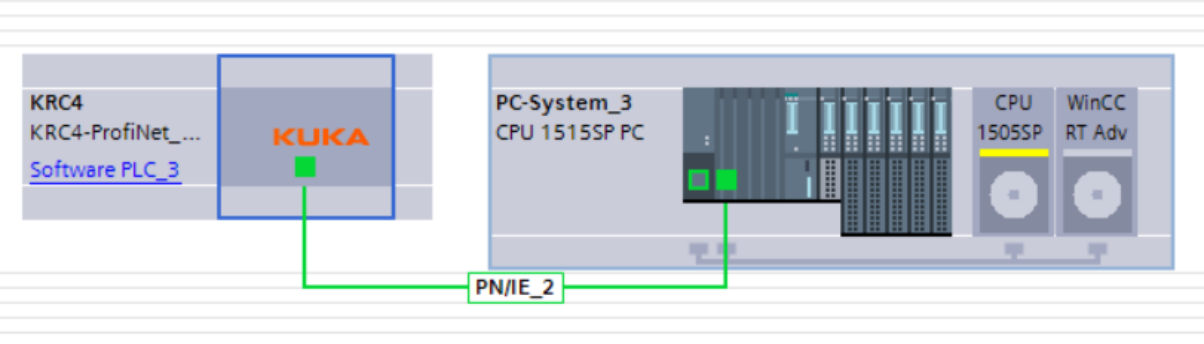


Figura 38. Vista de redes de la solución para la implementación real.

7.3.3 Software

El siguiente paso es desarrollar el código de control en TIA Portal. En el proyecto tenemos dos tipos diferentes de control necesarios. Por un lado, se encuentra el control del AGV de Tecnomatix. Por otro lado, tenemos las estaciones. Las dos estaciones que se van a implementar son idénticas. Por lo tanto, para ahorrar esfuerzo el código desarrollado va a ser igual en las dos estaciones. Las estaciones, además, están integradas en el sistema multiagente con ODK. Estos agentes interactúan entre ellos para lograr una flexibilidad y grado de automatización mayor en los procesos de fabricación. Los agentes negocian según su estado la fabricación de los planes generados (figura 205). Al agente encargado del control de los distintos servicios que puede llevar a cabo la célula se le denomina “Agente Máquina”. Este agente se comunica con el PLC en el que se ha diseñado un software para el control de la estación dividido en dos capas. En la capa superior, se ubican todas funciones encargadas de la gestión de los datos del pedido de servicio. La capa inferior se encarga de actuar sobre la unidad de control del robot y se denomina “capa operativa”. En esta capa se sitúan las funciones para el

control de la ejecución de las operaciones del robot. El código desarrollado se encuentra en el Anexo II de este documento.

7.3.3.1 Bloques de automatización de la estación

El control de las estaciones se ha realizado para poder ser guardado en una librería y así facilitar la implementación de nuevas estaciones en el futuro. El código desarrollado se ha realizado de manera que sea común en todas las estaciones. En el caso de ampliar la célula simplemente deberá incorporarse los bloques de dicha librería en el proyecto de control del nuevo PLC para hacerlo funcionar. La mayoría del código es válido tanto para la puesta marcha real como virtual. La descripción de los bloques se va a realizar para la puesta en marcha virtual, puesto que tiene bloques adicionales en su código. Al final de este segmento se comentan los cambios a introducir para la puesta en marcha real.

La automatización comienza con el OB100, este bloque se ejecuta una vez cada vez que el PLC pasa de estado *stop* a *run*. Por tanto, en este bloque se realiza la inicialización. En este caso se carga un 1 en el número del programa robot. De esta manera, al iniciar la secuencia de **ControlRobot** el robot puede iniciarse en el *main* del programa robot. Los demás bloques se insertan dentro del OB1 que realiza una ejecución cíclica una vez ejecutado el OB100.

ControlRobot [FC5]

En este bloque se lleva a cabo la activación de los actuadores y liberación del movimiento. Para ello hay que llevar a cabo una secuencia temporizada manipulándolas señales del robot. La secuencia y los tiempos se pueden consultar en el manual del robot. Con el pulsador de marcha se habilita el movimiento y 5 segundos más tarde se da la orden de ejecución del programa seleccionado. Cuando se habilita el movimiento, tras un segundo se envía un pulso de un segundo que activan los accionamientos del robot. Junto con esa señal, también se envía un pulso de 300 ms para confirmar los mensajes que aparecen durante la secuencia. El pulsador de paro es un pulsador normalmente cerrado. Con un 0 lógico en el pulsador de paro se deshabilita el movimiento el movimiento del robot.

ComunicacionODK [FC2]

Como se ha comentado anteriormente este proyecto se ubica dentro de un grupo de investigación en el que desarrolla el control de procesos de fabricación mediante sistemas multiagente. Las aplicaciones están desarrolladas con la herramienta Microsoft Visual Studio.

También se ha comentado en este apartado la necesidad de elegir una CPU compatible con ODK 1500S puesto que es la aplicación de Siemens para desarrollar aplicaciones en lenguajes de alto nivel para la ejecución en el PLC. ODK es el paquete de desarrollo que sirve de interfaz para la ejecución de programas de alto nivel en Windows dentro del programa de control PLC.

Para implementar el código desarrollado en el control se obtienen dos tipos de archivos con los que trabajar. Por una parte, se obtiene un archivo de extensión *.scl que se agrega a la herramienta TIA Portal como librería externa. En esta librería se generan los FB necesarios para llevar a cabo las funciones desarrolladas. Por otra parte, obtenemos un archivo *.dll que debemos colocar en el directorio especificado en ODK. En este archivo se encuentra el código a ejecutar por Windows.

Con estos pasos realizados, se introducen los FB generados. El primero de los bloques gestiona la creación del agente, por ello siempre esta activado. A continuación, se encuentran dos bloques que gestionan la conexión entre el agente y el PLC, uno de ellos activa la conexión y el otro la deshace. Al comienzo de la ejecución se activa la señal para realizar esta conexión. Por último, se encuentran dos bloques que realizan la función de leer y escribir el datagrama. Cuando el programa no encuentre ningún pedido activo leerá del ODK el datagrama hasta que encuentre un nuevo pedido. También se puede forzar una nueva lectura en cualquier momento desde el HMI. El FB para escribir se activa cada vez que se termina un ítem o servicio con las señales que llegan del robot.

EscribirDatosAgente [FC3]

En este bloque se rellena la información del datagrama que se envía al sistema de agentes. Durante la realización de un servicio, en varios momentos programados en el código del robot se envía información de la situación del servicio al sistema multiagente para que el sistema lo procese. De esta manera el sistema puede llevar una trazabilidad del servicio, tiempos de ejecución o número de ítems completados. El bloque escribe en el DB del datagrama esta información para a través del **FC2** enviarlo posteriormente.

El bloque comienza leyendo del reloj interno del PLC el *timestamp* del momento del inicio del servicio y ítem. Cuando finaliza un ítem añade el instante en el que ocurre y la información de referencia de máquina, orden, tipo de servicio, etc. En ese instante también activa el *flag* de ítem completado para enviar el datagrama. Dentro del bloque se cuenta con un contador para saber el número de ítems completados en el servicio, que se receta al finalizar el servicio. De forma similar se realiza una operación similar cuando finaliza el servicio.

GestionarPedido [FC6]

En este bloque se realiza la gestión y comprobación de los datos leídos de ODK y se enlazan con las variables que se comunican con el robot para llevar a cabo el servicio. Este bloque sirve de enlace entre la información obtenida mediante el sistema de agentes por ODK y el robot. Recordemos que el robot recibe y envía señales mediante la comunicación PROFINET IO localizada a partir de la dirección de memoria 2000. A su vez, la estructura de información recibida se guarda en un DB. Por tanto, es necesario mapear estas variables para poder comunicar las señales al robot.

En primer lugar, se lee el *flag* de nuevo servicio hasta recibir un 1 lógico. A continuación, se lee tanto el número de servicio como el número de ítems y se comprueba que están en los rangos aceptables. En el caso del servicio se comprueba que está entre 1 y 10, y para el caso de los ítems, entre 1 y 6. En caso de no cumplirse esa condición se activa una señal de servicio o ítems no valido y se guarda en un registro de alarmas para mostrar la información en el HMI. Si en la comprobación no ha habido errores, se envía al robot el servicio y el número de ítems a realizar moviendo del datagrama a las variables que se comunican mediante PROFINET IO los valores. Si hay algún error, o se finaliza el servicio se resetea el valor de estas variables para evitar fallos. Si hay un pedido activo y ningún error, se activa mediante un biestable la marca mapeada al robot de inicio de la ejecución que el robot está esperando en su código. Esta variable se resetea cuando se ha completado el último ítem del servicio. En principio puede parecer más sencillo emplear el fin de servicio para esta función. Sin embargo, produce un error, ya que durante un ciclo de ejecución el robot lee la señal de ejecutar pedido en estado activa antes de ser reiniciada. Esto es suficiente para que el robot vuelva a iniciar la secuencia quedando atrapado en un bucle infinito.

TCP/IP[FC4]

En este bloque se realiza el envío y recepción mediante TCP/IP. Las funciones empleadas realizan la conexión y la comunicación. La función de recibir escucha continuamente el canal y espera recibir datos con tamaño de 3 bytes. La información recibida es el estado del AGV para saber cuándo comenzar el servicio. Cuando los recibe, los guarda en un DB creado para este propósito mediante un puntero. La función de enviar envía 1 byte de datos del DB creado para ello solamente cuando hay un cambio en las señales. Es común realizar una implementación más sencilla utilizando pulsos periódicos, pero para ahorrar envíos innecesarios se ha implementado una solución para enviarlos cuando hay cambios en la señal. Los bytes que se envían no tienen por qué estar completamente

rellenos. De hecho, el byte de envío solo contiene un bit de información, el final del servicio. Sin embargo, la unidad mínima de datos transmitidos debe ser un byte, y un número natural de ellos.

MaterialFlow [FC1]

Durante el modelado en Tecnomatix PS se ha visto las operaciones de generación de *appearances*. Estas operaciones tienen asociadas unas señales que inician las operaciones. Para la simulación es necesario activar las señales correspondientes en cada caso en función del servicio y número de ítems. En este bloque se realiza la gestión y activación en instantes precisos de las señales de generación necesaria.

Para la generación de los casos espera a la señal de generar que proviene del HMI y se comprueba que no hay ningún caso creado mediante un sensor en el AGV. En función del número de servicio e ítems se activa la señal correspondiente degeneración de pallets de entrada. La generación de tapas se realiza de manera análoga teniendo en cuenta que no todos los servicios necesitan generar tapas exteriores. Para las tapas interiores bulones y rodamientos el funcionamiento es similar en los tres casos. Cuando el contador del robot incrementa, se crea una nueva pieza siempre y cuando el sensor correspondiente no detecte una pieza en el modelo. De esa manera, las piezas se van alimentando de una en una según se van consumiendo. Por último, también se activa la señal *kill* para hacer desaparecer todas las *appearances* cuando la célula finaliza. Esta señal llega mediante TCP/IP del PLC del AGV cuando este llega al almacén de salida.

EL caso de la segunda estación este bloque varía ligeramente. En la segunda estación no es necesario añadir de nuevo la generación de casos puesto que sería redundante. Por ello solo se introduce la generación de Tapas exteriores e interiores, rodamientos y bulones. La elección de dónde colocar la generación de casos de entrada es arbitraria y no tiene ningún efecto en la simulación, en este caso se ha decidido que esté en el control de la primera estación.

Para la implementación en la estación real han de hacerse ligeras modificaciones. En primer lugar, la estación real no necesita el bloque de **MaterialFlow** puesto que este solo se ocupa de la generación de piezas en Tecnomatix. Por lo cual, el bloque se elimina para la implementación real. En segundo lugar, la implementación real solo cuenta con un PLC. Por lo que la introducción de la comunicación TCP/IP carece de sentido. Es posible dejar el bloque, aunque no realice ninguna tarea puesto que no va a afectar al control. De esta manera, en un futuro, si se añaden más estaciones la comunicación estaría lista para ser usada. Sin embargo, en esta implementación se ha retirado para no tener código

inservible ocupando espacio. De todas maneras, el bloque de comunicación TCP/IP es fácilmente recuperable en un futuro desde la biblioteca creada para las estaciones.

7.3.3.2 Bloques de control del AGV

El control de la plataforma móvil se va a realizar para el modo de funcionamiento de Tecnomatix PS. Dentro del grupo de investigación se están desarrollando agentes para cumplir con esta función. Sin embargo, las necesidades de Tecnomatix y la implementación llevada a cabo en el laboratorio son muy diferentes, además de no estar finalizadas cuando se realiza este proyecto. Es por ello que es imposible utilizar el desarrollo para introducirlo en el control. Por tanto, para esta implementación se ha decidido hacer una secuencia automática teniendo en cuenta el funcionamiento en Tecnomatix PS con un control fuera del sistema multiagentes. Resulta interesante para futuros proyectos expandir estas capacidades al control mediante agentes también de los AGV supeditados por nuevas características en actualizaciones de PS.

De manera similar a las estaciones, en este caso también hay que inicializar el AGV. En el OB100 se ordena que el AGV se coloque en el almacén de entrada con la mesa en la posición superior. Se inicializa una velocidad y los diferentes valores del AGV mostrados en el modelado del AGV:

- *Velocidad*: 500mm/s
- *Motion planner type*: 0. Hacer uso de la *carpet*.
- *Path planner type*: 0. Almacenar en pila los *targets*.
- *Rotation Method*: 2. Mantener la orientación.

TCP/IP [FC1]

El funcionamiento de este bloque es similar al de las estaciones. Contiene cuatro bloques de para enviar y recibir de la primera estación y otros dos para la segunda estación. Sin embargo, estos bloques tienen una ligera modificación. Al realizar la configuración del enlace se ha determinado este PLC como el que realiza el establecimiento activo. Por tanto, a la conexión de estos bloques se le han añadido un retraso de 0.1 s ya que cuando quieran realizar la conexión los bloques de los otros PLC ya deben de estar activos. Con este retraso se asegura esta condición. El motivo establecer este PLC como el que realiza esta función permite que la implementación de las estaciones sea más sencilla. Por lo demás el funcionamiento de los bloques es igual teniendo diferentes DBs para almacenar la

información enviada y recibida. Los datos que se envían desde el AGV a las estaciones son idénticos para ambas estaciones.

Automatico [FC2]

Este último bloque controla secuencia que debe seguir el AGV para completar los traslados entre diferentes almacenes y estaciones de la célula. La inicialización deja el AGV en el almacén de entrada con la mesa subida. Una vez se generé el caso y se detecté el nuevo pallet, la mesa baja. Cuando la mesa se encuentra en la posición inferior, se avanza hasta la estación 1 y se levanta la mesa. Con la mesa ya subida se envía mediante la comunicación TCP/IP la información para que comience el robot el servicio asignado.

Cuando el robot finaliza el servicio y deposita el pallet se recibe la información y se procede a realizar la misma secuencia, esta vez con el objetivo de la estación 2. De nuevo, una vez colocado el AGV en posición con la mesa subida se envía la información para el comienzo del robot. Al finalizar la segunda estación se repite la secuencia avanzando al almacén de salida. Cuando El AGV alcanza el almacén de salida se envía la señal *kill* para hacer desaparecer los *appearances* creados y tras dos segundos para asegurar que se realiza correctamente se vuelve al almacén de entrada y se sube la mesa esperando recibir una nueva orden.

8 Puesta en marcha virtual. *Virtual Commissioning*

Con el objetivo de validar el programa de control desarrollado se lleva a cabo la puesta en marcha virtual. De esta manera se puede comprobar el correcto funcionamiento del proyecto de control diseñado sobre el modelo desarrollado. Con esta puerta en marcha virtual es posible detectar y depurar errores para corregirlos antes de la implementación en la célula real. Las mayores ventajas que ofrece el *virtual commissioning* es la reducción de los tiempos de puesta en marcha real, especialmente importante para implantaciones en la industria.

La simulación se va a llevar a cabo mediante *software in the loop* (*SiL*). Este método se basa en simulación sin hacer uso de hardware real. Al contrario de su opuesto *Hardware in the loop* (*HiL*) no es necesario contar con hardware especifico como PLCs, por lo que hace más económico realizar la simulación. Además, facilita la realización de dicha simulación puesto que para realizar *HiL* es necesario no solo contar con el PLC real, sino que también es necesario contar con tarjetas externas expresamente para ello (figura 39).

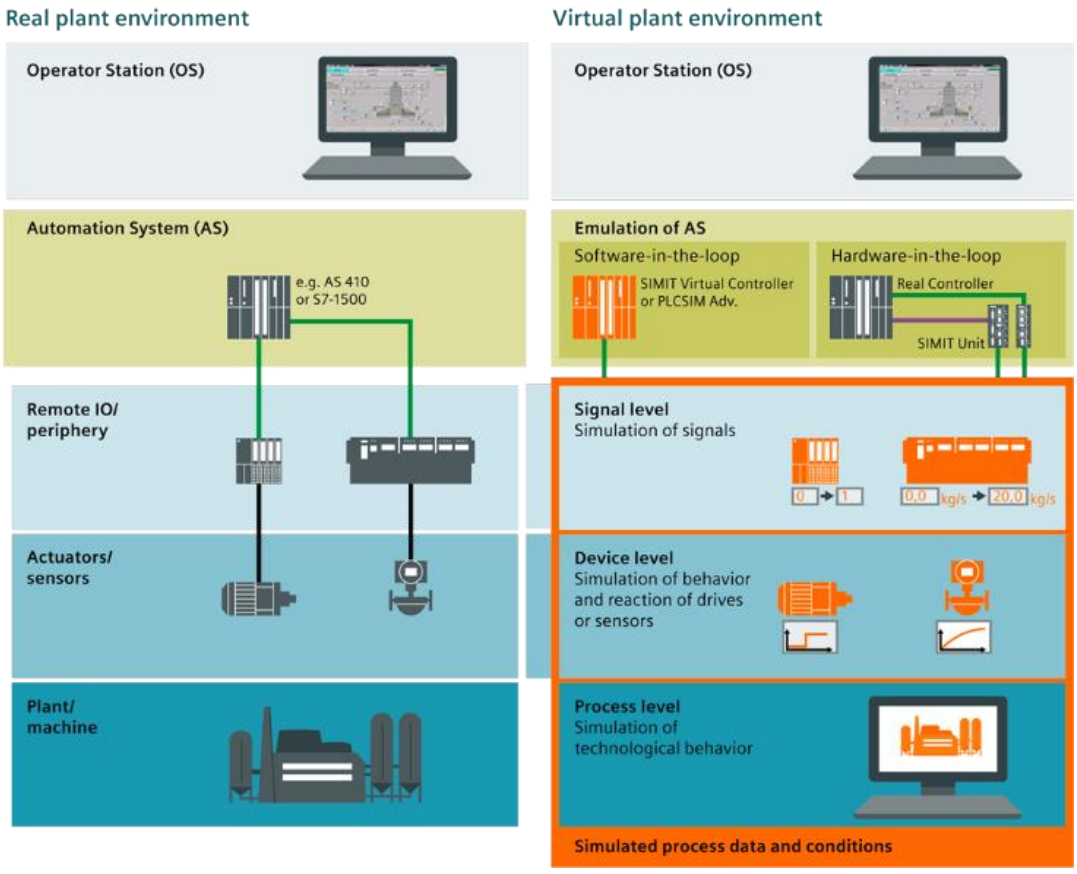


Figura 39. Representación de control en planta real y simulación con SiL y HiL.

En Tecnomatix PS es posible llevara a cabo la simulación bien dentro del mismo programa, bien realizando una conexión con uno o varios programas externos. El primer método se realiza mediante el ya mencionado CEE. Este es un simulador de tecnomatix basado en eventos. El CEE recopila y evalúa las señales cada ciclo para determinar el siguiente paso de la simulación. Con este método es necesario que el usuario fuerce las señales manualmente. Por lo tanto, este método es beneficioso para realizar las primeras pruebas del modelo para comprobar el funcionamiento. Sin embargo, no permite validar el proyecto de control. Una vez comprobado el correcto funcionamiento del modelo se pasa a realizar simulación con el programa externo, pudiendo así validar la automatización.

Tecnomatix PS ofrece diferentes métodos para poder realizar esta función. La forma más sencilla de realizarlo es con las aplicaciones de la misma plataforma PLM de Siemens, utilizando programas como SIMIT o PLC Advanced. Sin embargo, también es posible realizar con herramientas fuera de la plataforma mediante la tecnología de comunicación multiplataforma OPC UA. No obstante, en este proyecto se va a aprovechar la circunstancia de trabajar dentro de la plataforma PLM de Siemens con las facilidades que ello implica. Por tanto, la puesta en marcha virtual se va a llevar a cabo mediante PLCSIM Advanced.

SIMATIC S7-PLCSIM Advanced crea CPUs virtuales que permiten simular en el PC controladores de la gama S7-1500. Generalmente se conocen como instancias a estos PLC simulados. Además, ofrece una API que permite validar el código en conexión con simulaciones en el contexto máquina, planta. Esta característica es la que se va a aprovechar para realizar el *virtual commissioning*. En estas instancias se carga el proyecto de control desarrollado. Posteriormente se realiza la conexión de estas instancias con PS para realizar las simulaciones. PLCSIM Advance permite simular varias instancias dentro del programa, y en ellas es posible simular todas las funcionalidades, incluidas las comunicaciones (figura 40).

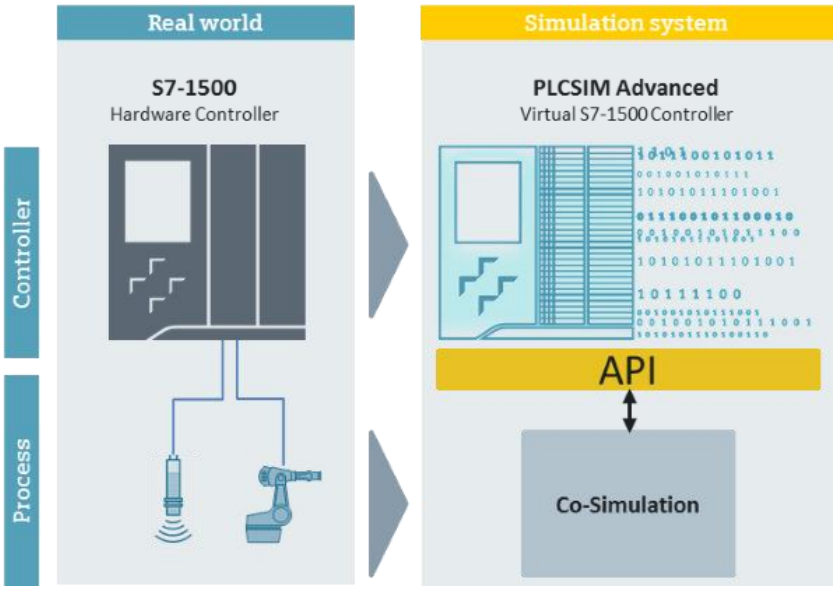


Figura 40. Comparación entre control y proceso real y simulado.

8.1 Simulación distribuida

Al realizar las simulaciones se presenta un problema. Llevar a cabo las simulaciones en un solo PC conlleva ralentizaciones debido al alto coste computacional necesario. Los equipos utilizados en el laboratorio tienen unas características avanzadas, aun así, no son capaces de llevar a cabo una simulación sin ralentizarse. Ante este problema las soluciones pasan bien por añadir potencia de computación al PC, o bien por fraccionar las necesidades ejecutando programas en diferentes PCs. Ampliar el equipo requiere realizar una nueva inversión, por el contrario, Hacer uso de PCs disponibles en el laboratorio no requiere ninguna inversión adicional. Por lo tanto, esta última es la opción elegida. No todos los programas contemplan esta opción, no obstante, tanto PLCSIM Advanced como Tecnomatic PS permiten realizarlo.

Cuando trabaja en local, la comunicación de las instancias de PLCSIM Advance se comunican vía Softbus (figura 41). Softbus es un bus de comunicación de *end to end* que esconde las comunicaciones de la red exterior. También es posible usar una comunicación TCP/IP, sin embargo, por defecto, esta es la opción usada. En el caso de utilizar TCP/IP PLCSIM Advance utiliza un adaptador virtual de ethernet que actúa como una interfaz de tarjeta de red real. De hecho, en los ajustes de dispositivos de redes de Windows aparece el *PLCSIM Virtual Ethernet Adapter* como otra tarjeta de red del dispositivo.

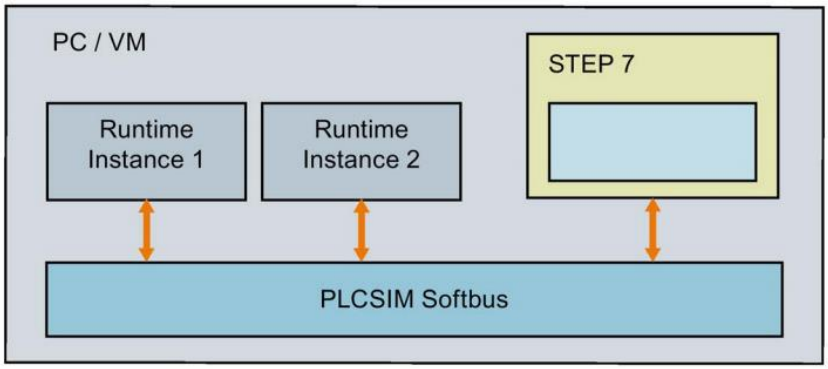


Figura 41. Diagrama bus Softbus local.

Cuando se realiza la simulación con los controladores distribuidos en dos o más PCs las comunicaciones se parecen a las realizadas en el caso local utilizando TCP/IP. En este caso, la comunicación TCP/IP se comunica por la tarjeta de red real en vez de por el *Virtual Adapter*. Estas comunicaciones utilizan el *PLCSIM Virtual Switch*, este hace la labor de un *switch* de direccionar los mensajes a través del adaptador de red real o simulado en función de las necesidades. Las comunicaciones pueden ser con CPUs reales o simuladas o HMIs reales o simulados (figura 42).

En el caso de utilizar el modo local no hace falta preocuparse por las direcciones IP de los diferentes elementos. Sin embargo, al utilizar la estructura distribuida es muy importante no repetir direcciones IP en la red. Es decir, las tarjetas de red e instancias de PLC no deben tener IP repetidas.

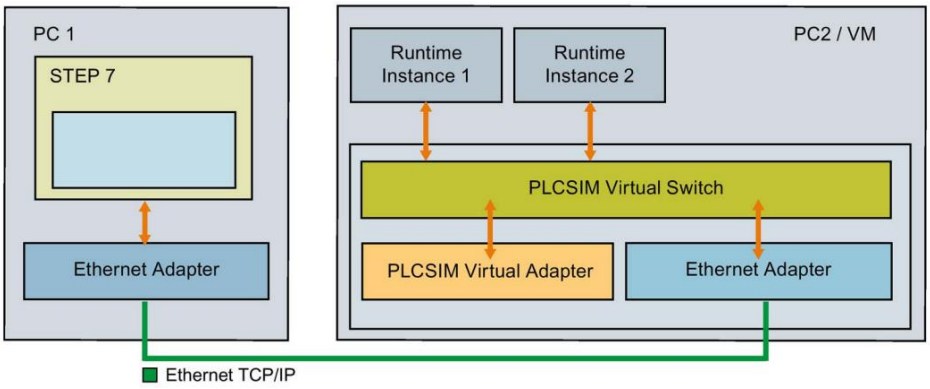


Figura 42. Diagrama comunicaciones TCP/IP distribuido.

Para poder realizar la comunicación con ODK en las instancias de PLC se debe instalar el mismo *.dll que se instala en el controlador real para esa función. PLCSIM Advance crea una carpeta al crear una instancia donde se encuentran los archivos que nos encontraríamos en el PLC real. Por tanto, dentro de esa carpeta se debe colocar el archivo en la misma ruta que en el PLC real. Una vez añadido el archivo es irrelevante para el funcionamiento de ODK si realizamos la simulación en local o distribuido.

La carga de archivos en las instancias PLC se realiza de manera similar a la carga en un PLC real cuando se realiza desde otro PC diferente al que contiene las instancias. Cuando se carga un proyecto, en el cuadro de dialogo de carga se debe seleccionar la tarjeta de red en la cual está la red para el control distribuido. Una vez finalizada la búsqueda el PLC muestra las instancias creadas en el programa con las IP asignadas en la creación de estas. Realizar la carga solo requiere seleccionar la instancia de PLC en la que se quiere realizar la carga.

En este proyecto se va a realizar una red de control distribuida. Todos los PCs contienen una tarjeta de red conectada a internet y una tarjeta de red adicional de la cual se va a hacer uso para crear esta red de control y simulación. Para la simulación se van a usar 3 PCs:

1. En este PC se lleva a cabo la simulación y se ejecuta el runtime del HMI. Además, también se realiza la carga de los proyectos de control
2. En el PC se ejecutan las instancias de PLC de las dos estaciones.
3. En el último PC se ejecuta la instancia PLC del control del AGV.

Para llevar a cabo las conexiones, se utiliza un *switch* al que se conectan todas las tarjetas de red, creando así una topología en estrella (figura 43). Se comprueba que las IP de las tarjetas de red. Todas deben estar en la misma subred, en este caso 192.168.1 y tener una dirección host diferente para cada una, en este caso .46, .20 y .16 respectivamente. Para 3 instancias de PLC se ha decidido utilizar las direcciones .120, .121 y .122. Para llevar a cabo las comunicaciones con el HMI, hay que realizar la parametrización de la interfaz PG/PC en Windows. En ella se indica que las comunicaciones S7 deben realizarse por la tarjeta de red correspondiente para poder enviar y recibir los mensajes.

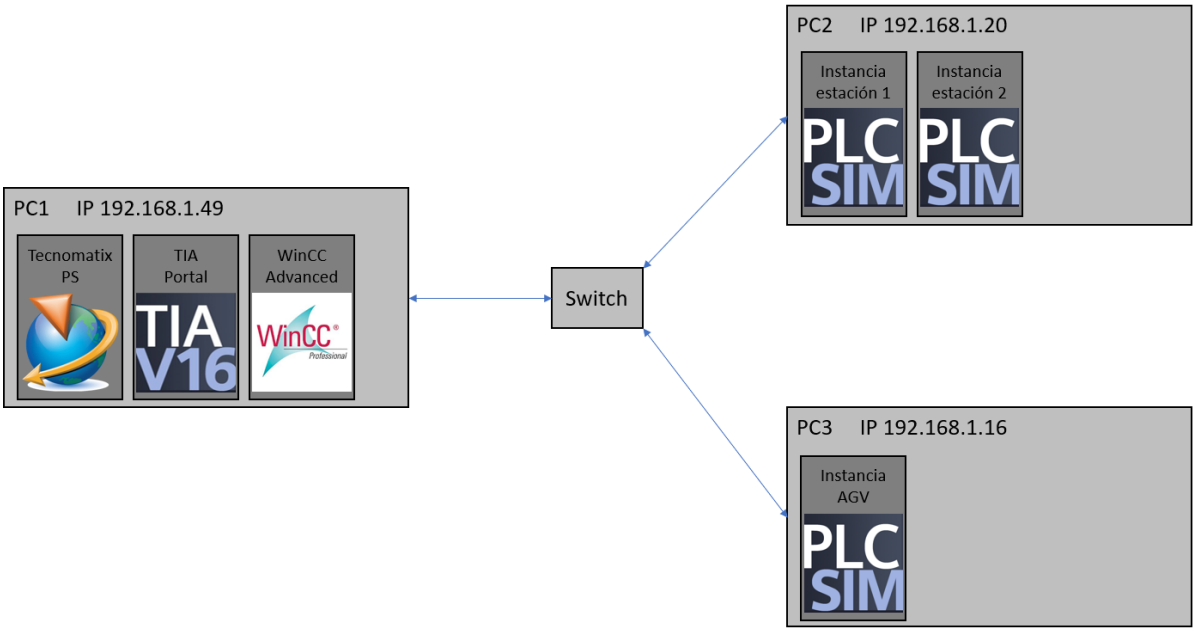


Figura 43. Diagrama de la topología de red.

8.2 Conexión PLCSIM Advanced y Tecnomatix Process Simulate

El primer paso es añadir las instancias en las que se han cargado los proyectos de control para que de esta manera se pueda realizar la comunicación de las señales más tarde. Para añadir estas instancias se debe abrir el cuadro de dialogo de opciones y seleccionar PLC en él. Dentro de esta opción es posible seleccionar el modo de simulación, si se quiere emplear el CEE o un PLC externo (figura 44). Dentro de los PLC externos hay que seleccionar el programa con el que se va a añadir la instancia. Entre las opciones encontramos las ya mencionadas SIMIT, OPC UA o PLCSIM Advanced. Es posible realizar la simulación con una combinación de estos programas. Por ejemplo, se puede usar PLCSIM Advanced para simular un PLC, SIMIT para las señales de elementos modelados como servomotores, o cualquier combinación con los programas que ofrece PS. En este proyecto las tres instancias se ejecutan en dos PCs diferentes al de Tecnomatix, por tanto, solo se va a utilizar la conexión con PLCSIM.

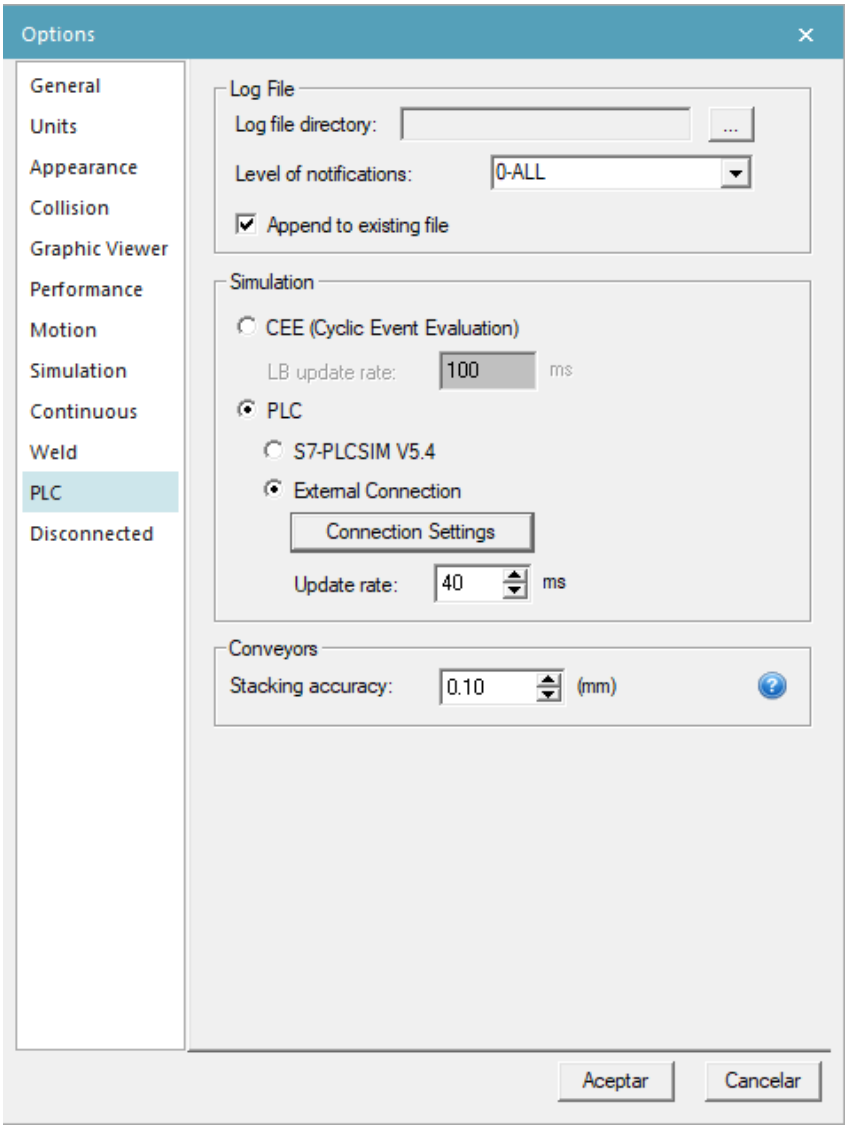


Figura 44. Cuadro de dialogo PLC.

Dentro de PLCSIM Advanced es posible elegir si la instancia está en local o remoto. De nuevo es posible combinar ambos tipos de instancias. Para poder añadir las instancias remotas primero ha de comprobarse que los ordenadores en la red son visibles en ambas direcciones. Para ello es necesario cambiar los ajustes de Windows para hacer los equipos públicos en la red. Una vez ambos son visibles, se debe indicar en el cuadro de dialogo para añadir una instancia remota el nombre del equipo donde se encuentra la instancia y el puerto asignado a PLCSIM Advanced (figura 45). Después, aparecen las instancias ejecutándose en el equipo remoto para seleccionar y nombrar la instancia deseada. Una vez añadida todas las instancias remotas se puede realizar el mapeo de señales.

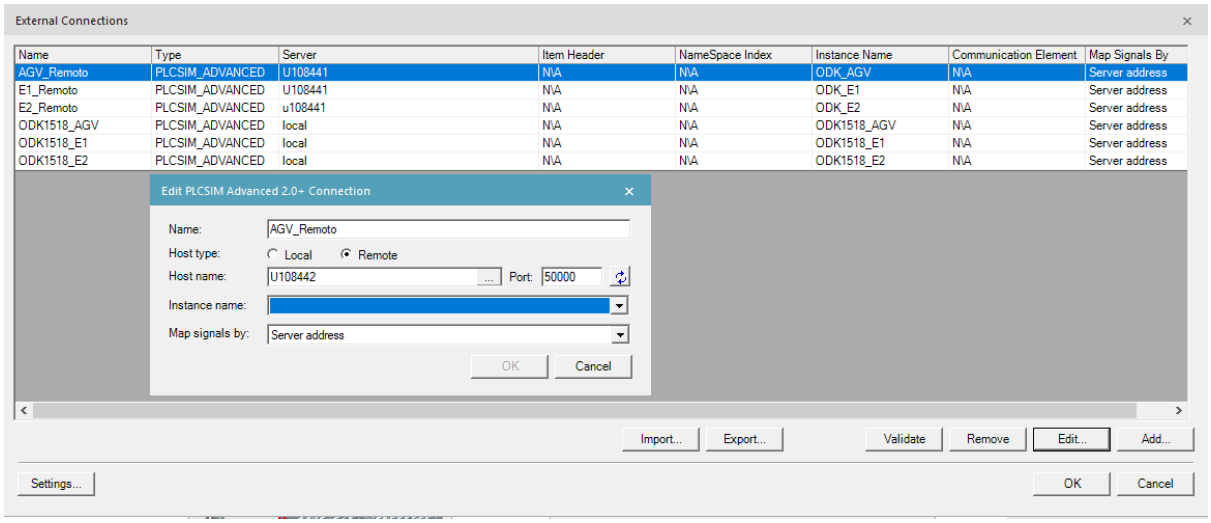
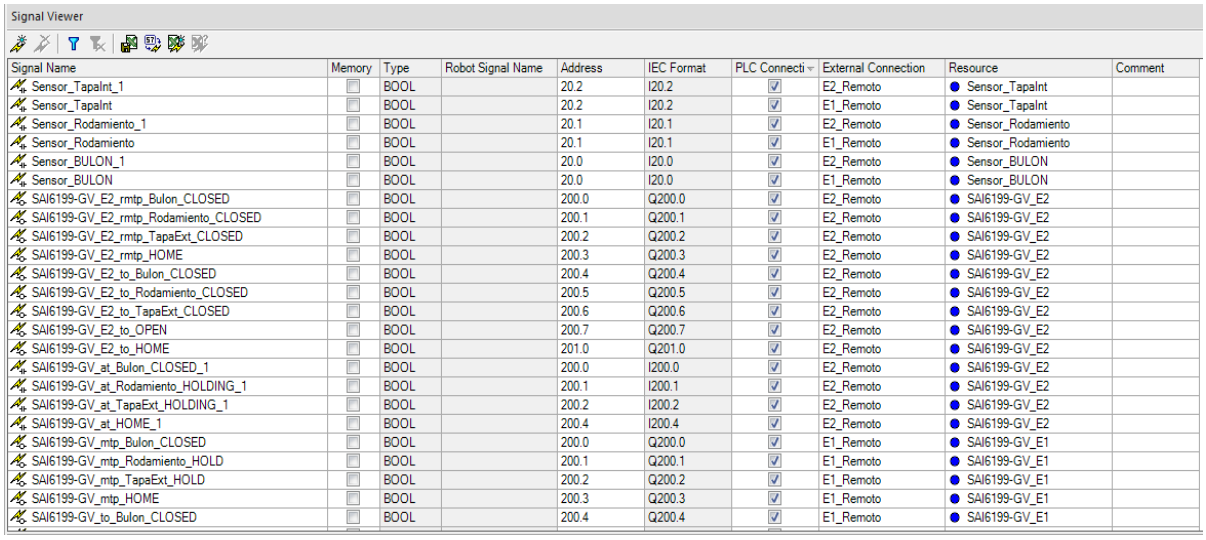


Figura 45. Cuadro de dialogo para añadir instancias remotas de PLCSIM Advanced.

El siguiente paso es mapear las señales entre Tecnomatix PS y PLCSIM Advanced. Existen dos formas de realizar este mapeo de señales: por nombre o por dirección. El mapeo de señales por nombre requiere que las variables de Tecnomatix y los proyectos de control coincidan. Por norma general, hace más rápido el proceso de mapeado. Sin embargo, en proyectos con gran cantidad de variables y nombres muy parecidos como este, es difícil mantener el mapeado sin errores. Además, de esta manera no es posible reciclar el código de TIA Portal, ya que habría que cambiar en cada estación los nombres para que no coincidan. Por ello, se ha elegido realizar el mapeo por dirección, que hace el proceso. más largo, pero ofrece mayor control y facilidad para detectar y solucionar fallos. En el visor de señales (figura 46) hay que indicar la dirección de la variable en el PLC. También hay que indicar con que instancia de las añadidas se quiere realizar el mapeo. Esta es la mayor ventaja de realizar el mapeo por dirección, puesto que las direcciones de las variables de las estaciones son las mismas, por lo tanto, solo es necesario copiar y pegar las direcciones de la primera estación en la segunda, y seleccionar para cada variable la instancia adecuada.


Signal Viewer

Signal Name	Memory	Type	Robot Signal Name	Address	IEC Format	PLC Connecti	External Connection	Resource	Comment
Sensor_TapaInt_1		BOOL		20.2	I20.2		E2_Remoto	Sensor_TapaInt	
Sensor_TapaInt		BOOL		20.2	I20.2		E1_Remoto	Sensor_TapaInt	
Sensor_Rodamiento_1		BOOL		20.1	I20.1		E2_Remoto	Sensor_Rodamiento	
Sensor_Rodamiento		BOOL		20.1	I20.1		E1_Remoto	Sensor_Rodamiento	
Sensor_BULON_1		BOOL		20.0	I20.0		E2_Remoto	Sensor_BULON	
Sensor_BULON		BOOL		20.0	I20.0		E1_Remoto	Sensor_BULON	
SAI6199-GV_E2_rmtp_Bulon_CLOSED		BOOL		200.0	Q200.0		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_E2_rmtp_Rodamiento_CLOSED		BOOL		200.1	Q200.1		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_E2_rmtp_TapaExt_CLOSED		BOOL		200.2	Q200.2		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_E2_rmtp_HOME		BOOL		200.3	Q200.3		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_E2_to_Bulon_CLOSED		BOOL		200.4	Q200.4		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_E2_to_Rodamiento_CLOSED		BOOL		200.5	Q200.5		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_E2_to_TapaExt_CLOSED		BOOL		200.6	Q200.6		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_E2_to_OPEN		BOOL		200.7	Q200.7		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_E2_to_HOME		BOOL		201.0	Q201.0		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_at_Bulon_CLOSED_1		BOOL		200.0	I200.0		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_at_Rodamiento_HOLDING_1		BOOL		200.1	I200.1		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_at_TapaExt_HOLDING_1		BOOL		200.2	I200.2		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_at_HOME_1		BOOL		200.4	I200.4		E2_Remoto	SAI6199-GV_E2	
SAI6199-GV_mtp_Bulon_CLOSED		BOOL		200.0	Q200.0		E1_Remoto	SAI6199-GV_E1	
SAI6199-GV_mtp_Rodamiento_HOLD		BOOL		200.1	Q200.1		E1_Remoto	SAI6199-GV_E1	
SAI6199-GV_mtp_TapaExt_HOLD		BOOL		200.2	Q200.2		E1_Remoto	SAI6199-GV_E1	
SAI6199-GV_mtp_HOME		BOOL		200.3	Q200.3		E1_Remoto	SAI6199-GV_E1	
SAI6199-GV_to_Bulon_CLOSED		BOOL		200.4	Q200.4		E1_Remoto	SAI6199-GV_E1	

Figura 46. Visor de señales. Mapeado con las instancias remotas.

Para llevar a cabo una monitorización durante las señales durante la simulación Tecnomatix PS dispone de un panel de simulación (figura 47). Además de poder visualizar el estado de las señales durante la simulación, también es posible forzar si es necesario variables no mapeadas con el PLC.

Una vez realizado el mapeo de las señales, se lanzan diferentes simulaciones de servicios y números de ítem. Durante las simulaciones se observa que la activación y desactivación de las señales son correctas. También se comprueba que la simulación de los movimientos y flujo de material se realiza correctamente logrando una simulación acorde a lo esperado en la realidad. Una vez finalizadas las simulaciones si todo ha funcionado correctamente el código de control y el código robot desarrollado queda validado y puede ser volcado al *hardware* real para validar su funcionamiento en la estación real.

El *virtual commissioning* en Tecnomatix PS permite la validación de los procesos desarrollados en un entorno virtual 3D mediante el modelado de Gemelos Digitales. Esto permite evaluar diferentes aspectos del proceso como el análisis de colisiones, distribución de los elementos, layout, validación de código robot y control o estudios de tiempos de fabricación. Permitiendo un análisis exhaustivo de estos campos para optimizar los procesos de fabricación que se van a trasladar en la realidad. Por lo tanto, resulta en un campo de pruebas flexible y sin necesidad de contar con el hardware real o espacio físico. Adelantando así los procesos de puesta en marcha o cambios en la producción. Por tanto, se demuestra como una herramienta muy potente cada vez más asimilada en la industria.

Simulation Panel

Simulation	Inputs	Outputs	....	Forced	Force...	Address	Robot Signal ...
ALCANCE_ROBOT_MESA							
AGV							
KMP1500-3-simply_stp_ID1	0				0	I	
KMP1500-3-simply_stp_O...						I0.2	
KMP1500-3-simply_stp_C...	0				0	I1	
KMP1500-3-simply_stp_A...						Q	
KMP1500-3-simply_stp_T...		0			1	Q12	
KMP1500-3-simply_stp_S...		0.00			500.00	Q7	
Subir_E1_start						Q6.1	
Bajar_E1_start						Q6.2	
ESTACION 1							
ROBOT 1							
$EXT_START						Q200...	$EXT_START
kr3_r540_programNu...		0			1	Q110	programNumber
kr3_r540_at_HOME						I100.0	kr3_r540_at_...
kr3_r540_ServiceType		0			0	Q2246	ServiceType
kr3_r540_NItems		0			0	Q2247	NItems
kr3_r540_Counter	0				0	I102	Counter
kr3_r540_NewService						Q100.1	NewService
$DRIVES_ON						Q200...	$DRIVES_ON
$DRIVES_OFF						Q200...	$DRIVES_OFF
$CONF_MESS						Q200...	$CONF_MESS
$MOVE_ENABLE						Q200...	$MOVE_ENA...
Service_Completed						I2247.1	Service_Com...
kr3_r540_NewService_1						Q100.1	NewService
kr3_r540_ServiceType_1		0			0	Q2246	ServiceType
kr3_r540_NItems_1		0			0	Q2247	NItems
ESTACION 2							
SENSORES							
Sensor_TapaInt						I20.2	
Sensor_Rodamiento						I20.1	
Sensor_BULON						I20.0	
GENERATE							
GenerateParts_CASO1_1...						Q0.3	
GenerateParts_CASO1_2...						Q0.4	
GenerateParts_CASO1_3...						Q0.5	
GenerateParts_CASO1_4...						Q0.6	
GenerateParts_CASO1_5...						Q0.7	
GenerateParts_CASO1_6...						Q1.0	

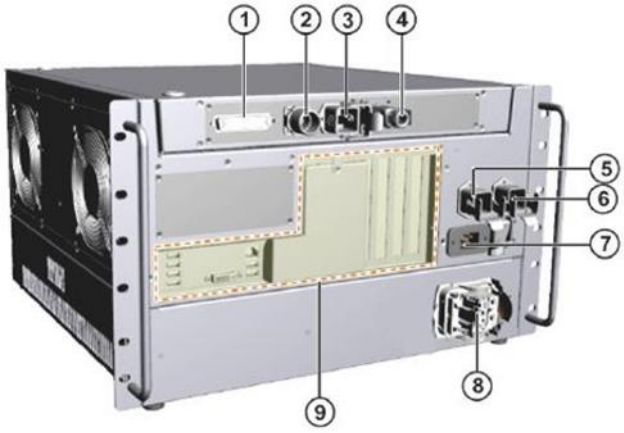
Figura 47. Panel de simulación de la célula en un instante cualquiera de la simulación.

9 Puesta en marcha real

Una vez validado, antes de llevar a cabo el volcado del código desarrollado hay que realizar la configuración y conexión de la estación robotizada. La configuración se lleva a cabo con el programa Workvisual. Workvisual es el programa de KUKA para realizar programación *offline*, diagnostico *online* y mantenimiento.

9.1 Conexiones

La unidad de control contiene diferentes interfaces para realizar las conexiones de los diferentes elementos. En la figura 48 se muestran todas las interfaces de la controladora:



1. X11 Interfaz de seguridad (opcional)
2. X19 Conexión a la smartPAD
3. X65 Interfaz de extensión
4. X69 Interfaz de servicio
5. X21 Interfaz de manipulador
6. X66 Interfaz de seguridad Ethernet
7. X1 Conexión de fuente de alimentación
8. X20 Conector de motor
9. Interfaces de PC de control

Figura 48. Conexiones de la unidad de control KR C4 Compact.

El robot también cuanta con diferentes conexiones para la alimentación y control de los motores, transferencia de señales y conexiones para líneas de aire del sistema neumático. En la figura 49 se muestran estas conexiones:

1. Cable de motor (XM1/XM2-6)
2. Cable de datos (X15/X18)

1. Conexión para línea de aire AIR1 (Diámetro exterior 4mm)
2. Conexión para línea de aire AIR2 (Diámetro exterior 4mm)
3. Conexión para línea de aire AIR3 (Diámetro exterior 4mm)
4. Conexión para línea de aire AIR4 (Diámetro exterior 4mm)
5. Conexión MEMD X32
6. Conexión de sistema de suministro de energía X76

Figura 49. Conexiones de la base del robot KUKA KR3 R540.

EL primer paso para realizar las conexiones es alimentar la unidad de control. Para ello, se conecta el conector de la fuente de alimentación de la controladora a la red eléctrica. A continuación, se realiza la conexión entre el robot y la unidad de control. Se deben conectar los cables de motor y datos al conector del motor e interfaz del manipulador de la controladora respectivamente. De esta manera se alimentas los accionamientos del robot y se envían y reciben las señales de control de estos. Además, se conecta la SmartPAD a su conector para confirmar que las conexiones se han realizado correctamente.

A continuación, se conectan los elementos de seguridad de la estación como la seta de emergencia a la interfaz de seguridad X11. Otros elementos de seguridad del operario se conectan a los correspondientes pines de este conector.

Para realizar la comunicación entre la controladora y los actuadores externos del robot, es decir las pinzas de la garra, se va a hacer uso de la interfaz de extensión. Esta interfaz ofrece la posibilidad de añadir esclavos EtherCAT a la red de la controladora. Las comunicaciones internas se realizan mediante esta comunicación y ofrece una conexión al exterior para añadir dispositivos. En este conector se conecta mediante ethernet el dispositivo EtherCAT EK1100 EtherCAT Coupler (2A E-bus). A este acoplador se añaden dos módulos de entradas y salidas EL1809 (16 entradas digitales) y EL2809 (16 salidas digitales). Cuatro de las entradas se usan para los sensores de posición de las dos pinzas. Dos salidas digitales controlan las electroválvulas del sistema neumático de las pinzas para abrirlas y

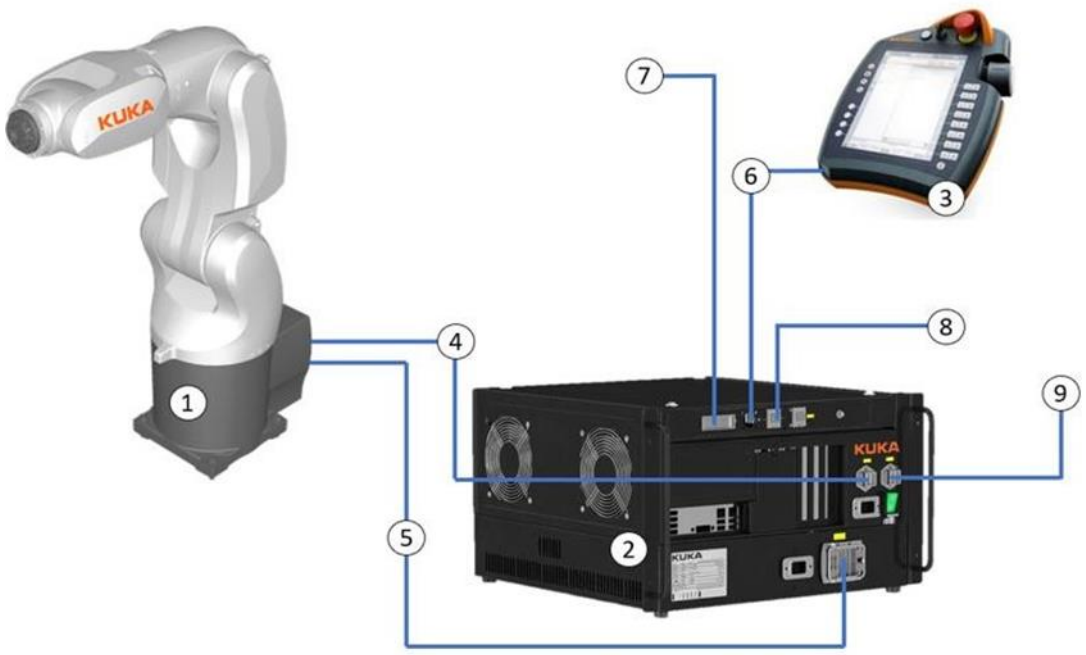
cerrarlas. También se añade una fuente de alimentación de 24V para alimentar este dispositivo. En la tabla 8 se muestran las señales y direcciones del bus EtherCAT.

Tabla 8. Señales y direcciones de las entradas y salidas EtherCAT.

DIRECCIÓN	SEÑAL
ENTRADAS	
$IN[1]	Pinza rodamiento abierta
$IN[2]	Pinza rodamiento cerrada
$IN[3]	Pinza bulón abierta
$IN[4]	Pinza bulón cerrada
SALIDAS	
$OUT[1]	Cerrar pinza rodamiento
$OUT[2]	Cerrar pinza bulón

La comunicación con el PLC se va a realizar mediante PROFINET a través de la interfaz de seguridad ethernet. El cable ethernet se conecta al puerto X1 del PLC tal y como se configuró en la configuración *hardware* de TIA Portal. Desde este puerto ethernet también se va a realizar la configuración online desde Workvisual. Una vez realizada la carga de configuración y programas robots, se conecta el puerto a la red del PLC para realizar la comunicación.

Por último, se añaden las cuatro vías de aire de las electroválvulas a las conexiones de la base del robot. Estas vías siguen por dentro de la cubierta del robot y terminan cerca de la garra del robot para conectarse con las pinzas. De esta manera, las mangueras de aire quedan protegidas de elementos externos. En la figura 50 se muestra de manera esquemática estas conexiones:



1. Robot
2. Unidad de control del robot
3. Panel de control smartPAD
4. Conexión de datos
5. Conexión de motor
6. Cable de conexión con smartPAD
7. Conexión de seguridad
8. Conexión con dispositivo EtherCAT
9. Conexión con PLC y PC

Figura 50. Esquema de las conexiones del robot.

9.2 Configuración hardware del robot

Una vez realizadas las conexiones, hay que llevar a cabo la configuración hardware del proyecto robot en WorkVisual. Durante la configuración online la interfaz de seguridad ethernet está conectada a la tarjeta de red del PC con Workvisual para poder realizar la carga *online*. De esta manera el programa reconoce la controladora conectada y visualizar todos los archivos en ella.

En el proyecto de control solo se encuentra la controladora KUKA KR C4 Compact con los archivos por defecto. En el mismo proyecto de control se debe añadir el robot con el que está conectado, en este caso, el KUKA KR3 R540. Una vez añadido el robot, se marca la controladora como la unidad de control activa. Tras esto, se lleva a cabo la modificación de la estructura del bus de control (figura 51) para añadir las comunicaciones mencionadas.

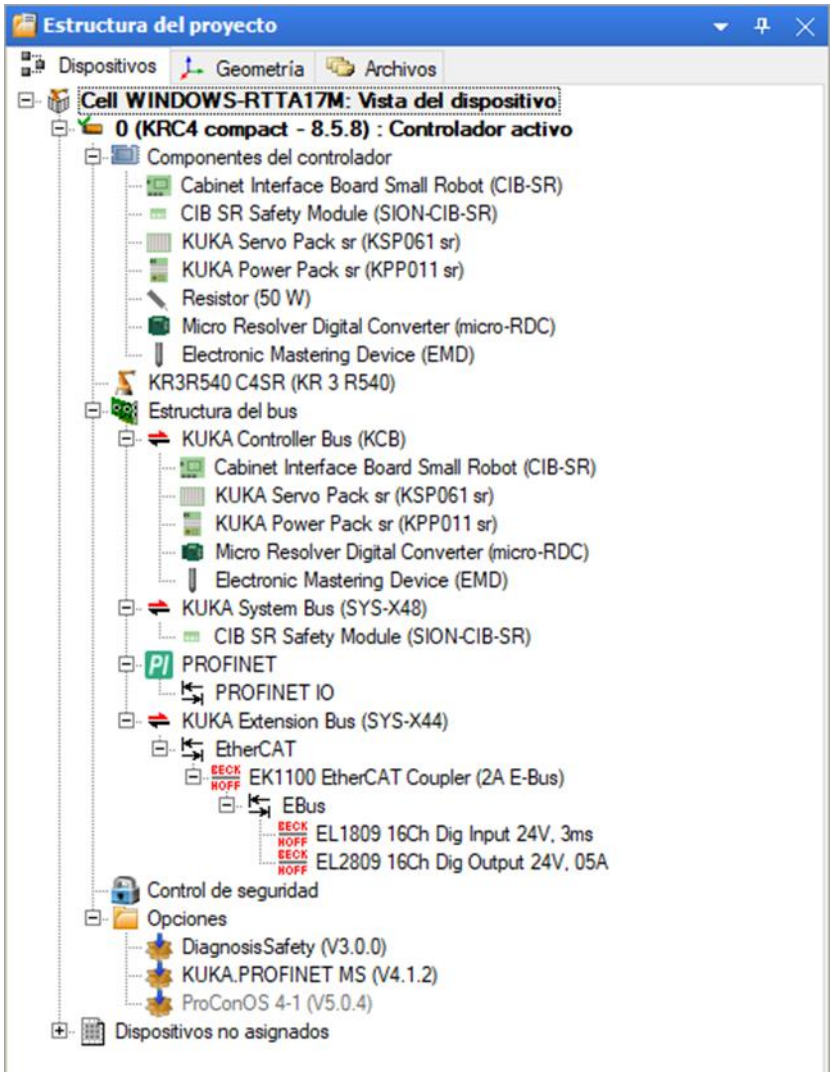


Figura 51. Configuración hardware del proyecto robot.

En primer lugar, se va a añadir la comunicación mediante EtherCAT. Para ello, se abre la pestaña de KUKA Extension Bus y se añade el acoplador EK1100. A continuación, se entra en la pestaña de este dispositivo para añadir las tarjetas que van conectados al mismo. En este caso, se añaden las dos tarjetas de entradas y salidas digitales EL1809 y EL 2809. A estas tarjetas hay que indicar las direcciones que van a ocupar, en este caso se han asignado las direcciones de la 1 a la 16 para las entradas y salidas.

Una vez introducidos los dispositivos EtherCAT se deben mapear las entradas y salidas con el bus de campo. Para ello hay que dirigirse a la pestaña Circuito EA dentro de WorkVisual. Como se observa en la figura 52, la pestaña cuenta con 5 ventanas diferentes. En las dos ventanas superiores se determinan los dispositivos que se quieren conectar. La ventana central muestra los mapeos ya realizados, mientras que las dos inferiores muestran las señales de cada dispositivo. Para realizar este primer

mapeado, hay que seleccionar en la ventana superior izquierda las entradas digitales, y en la derecha, dentro de buses de campo, EtherCAT entradas digitales. A continuación, se seleccionan los 16 canales EtherCAT y la señales $IN[1] .. $IN[16] y se conectan. Con las salidas digitales se lleva a cabo el mismo proceso, finalizando el mapeo de las señales EtherCAT.

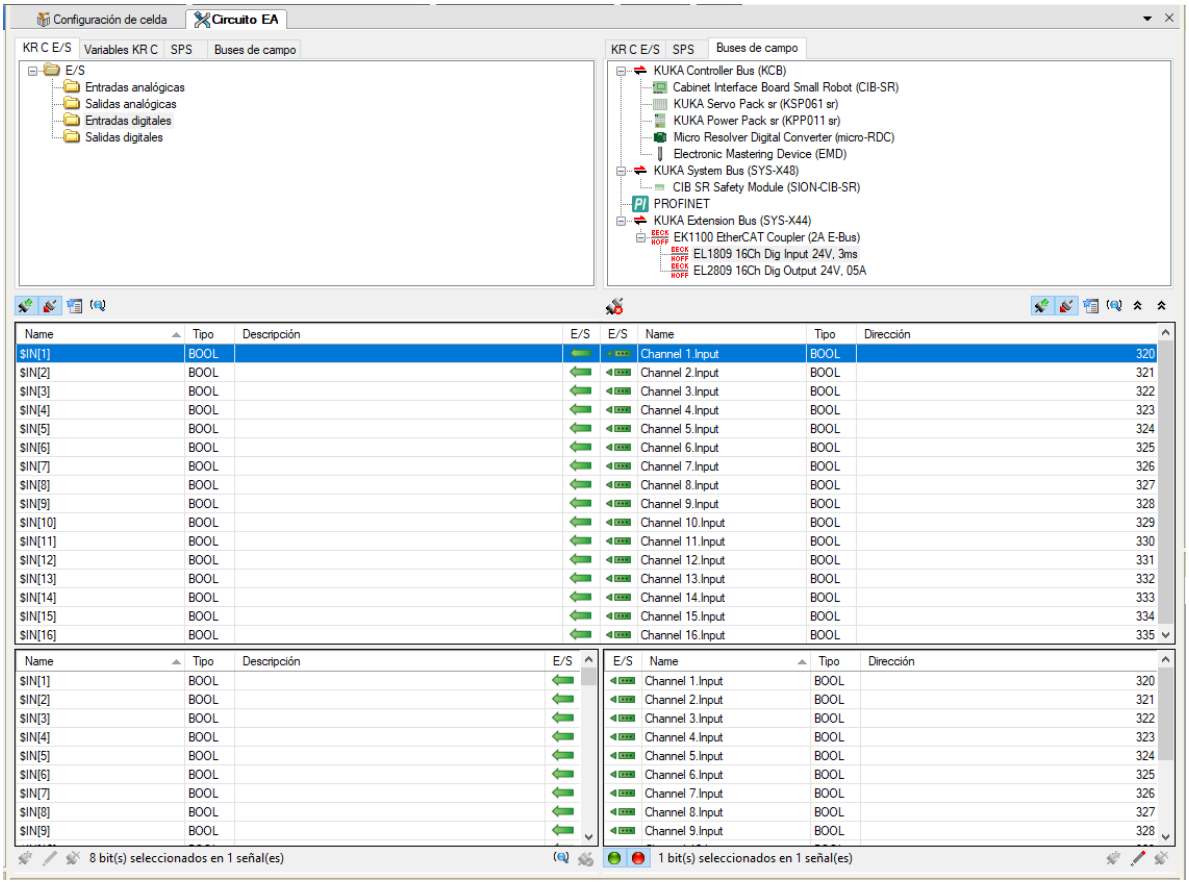


Figura 52. Mapeado de señales entre EtherCAT y las señales del robot

En segundo y último lugar hay que realizar un proceso similar para la comunicación PROFINET. De nuevo, en la estructura de bus hay que añadir un bus PROFINET IO (figura 53). Dentro del módulo se deben configurar las características de este. Como ya se ha comentado anteriormente la identificación en PROFINET se realiza mediante el nombre del dispositivo. Por ello, en primer lugar, se debe cambiar el nombre para que coincida con el especificado en la configuración *hardware* del PLC. LA configuración PROFINET soporta el intercambio de hasta 254 bytes de entradas y salidas digitales. En este proyecto se opta por permitir el máximo de entradas y salidas permitidas para la comunicación, por si son necesarios en futuros proyectos, como el que se lleva a cabo paralelamente con mxAutomation. Por tanto, se debe indicar que este es el número de bytes de I/O que se van a intercambiar.

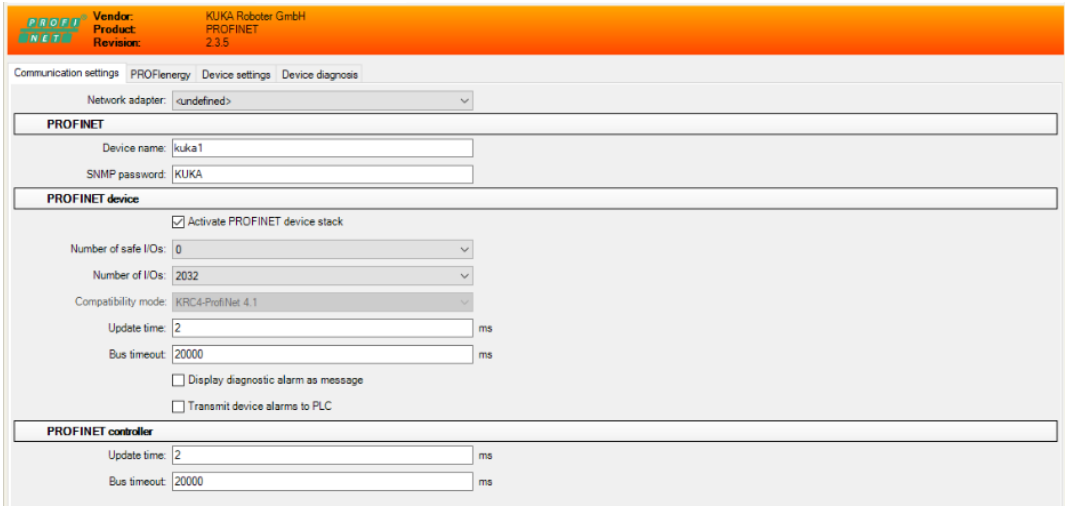


Figura 53. Configuración de la comunicación PROFINET IO.

Por último, se lleva a cabo un mapeado de señales similar al de EtherCAT. En la pestaña superior derecha se selecciona el bus PROFINET y en el izquierdo las entradas digitales. El robot cuenta con 4096 entradas y salidas digitales, en este caso se ha decidido mapear los 254 bytes a partir de la dirección 2049 para ocupar la parte final de la memoria. Algunas de las señales vienen predefinidas y son necesarias para habilitar el movimiento. Estas señales deben conectarse con la comunicación PROFINET para poder actuar sobre ellas desde el PLC. En la tabla 9 se muestran estas señales y las direcciones:

Tabla 9. Señales para control de robot desde PLC.

Señal	Dirección Robot	Dirección Profinet	Función
$EXT_START	$IN[2059]	I19.2	Puesta en marcha del programa de robot en modo EXTERNO.
$MOVE_ENABLE	$IN[2060]	I19.3	Habilita el movimiento del Robot. Cuando se activa, el robot puede moverse de forma manual desde la smartPAD o ejecutar programas.
$CONF_MESS	$IN[2061]	I19.4	Confirma los mensajes de error del robot para borrarlos.
$DRIVES_OFF	$IN[2062]	I19.5	Habilita los accionamientos del robot
$DRIVES_ON	$IN[2063]	I19.6	Conecta los accionamientos del robot

Una vez realizado el mapeo también con las salidas digitales, se carga y activa el proyecto en el robot para cargar la configuración. Una vez cargado se observa que no surge ningún error por la configuración.

9.3 Configuración software y programa robot

9.3.1 Señales robot

Con la configuración *hardware* realizada, antes de volcar los programas robot hay que realizar la configuración software. En este paso se tienen que declarar las variables usadas en el programa robot y declarar el tipo de dato. Para ello, es necesario modificar el archivo $config.data mientras el robot está en conexión *online*. En este archivo se declaran las señales y se las direcciones de memoria donde se encuentran (figura 54). Varias variables como en número de ítems o servicio tienen longitud de un byte, por lo tanto, es necesario indicar desde que dirección hasta que dirección ocupa.

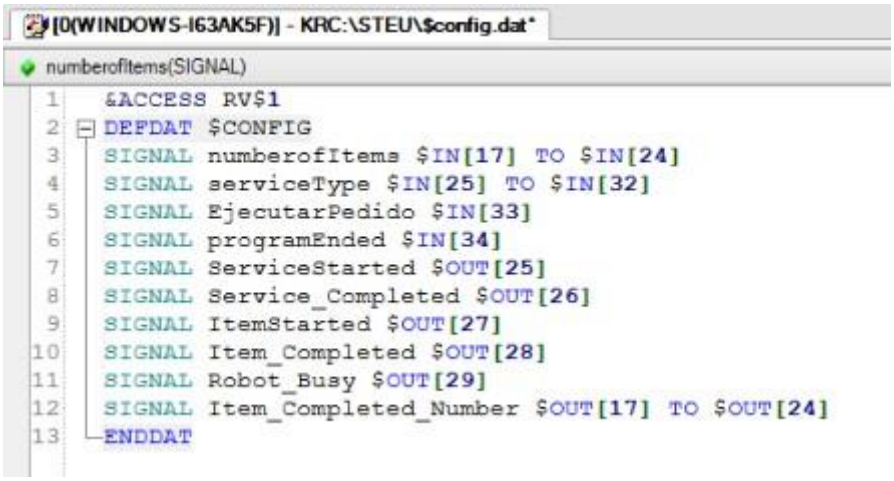


Figura 54. Señales y direcciones robot.

Antes de trasladar los cambios a la controladora hay que indicar el tipo de variable en PROFINET. Es decir, las señales que no son booleanas se les ha asignado un rango de direcciones que encajan con el tipo de dato, para poder llevar a cabo la comunicación con PROFINET hay que indicar estos rangos también en el mapeo de señales. Las señales están definidas en las variables robot, pero no en la comunicación PROFINET.

De nuevo hay que dirigirse a la pestaña de circuito EA y abrir el mapeo entre las entradas digitales y PROFINET. A continuación, hay que abrir el mapa de bits en el editor de señales. Por defecto todos los bits están mapeados individualmente, es decir, como booleanas. Teniendo en cuenta las direcciones asignadas en la declaración hay que agrupar las direcciones correspondientes en grupos de 8 bits para

formar los bytes de dichas variables. Una vez agrupados los bits el tipo de dato cambia de BOOL a BYTE indicando que la agrupación se ha realizado de manera correcta (figura 55).

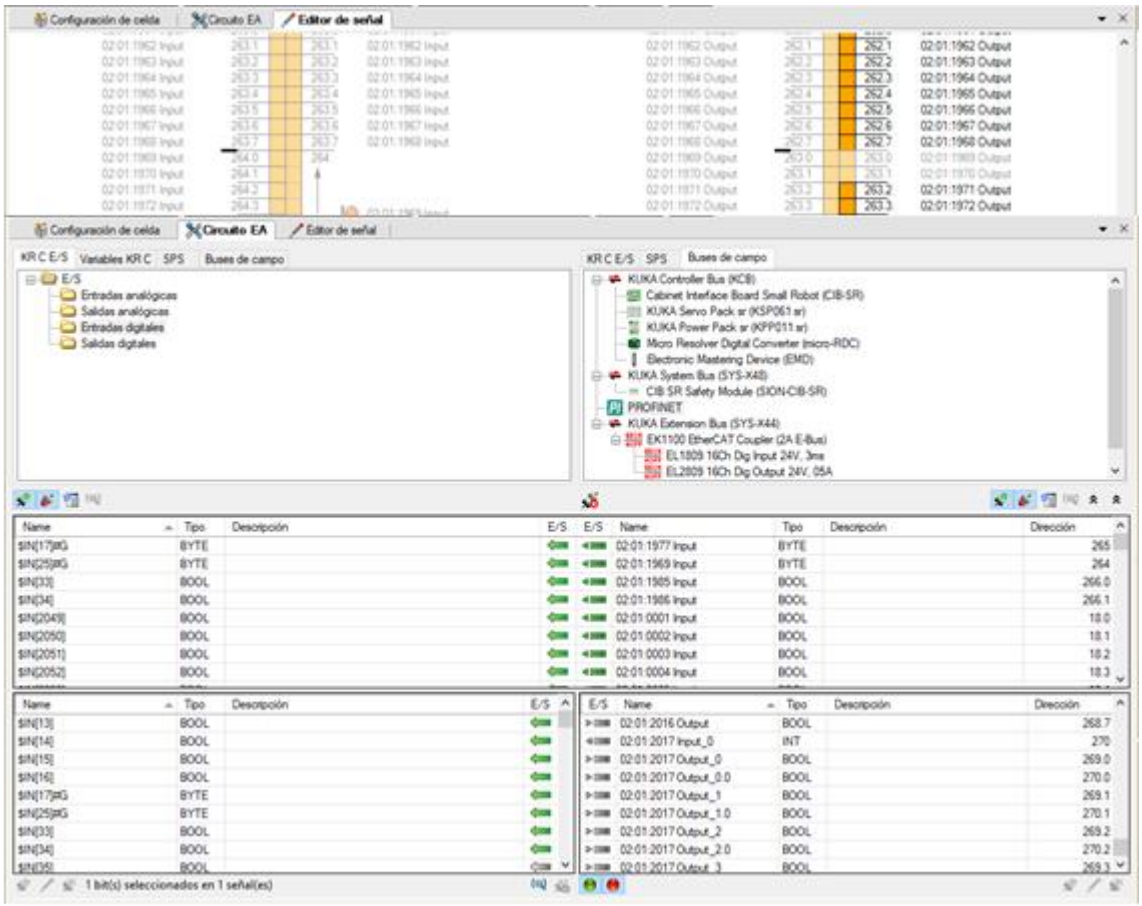


Figura 55. Editor de señales y mapeo PROFINET.

Una vez realizados los grupos de bytes y realizar el mapeo con las señales correspondientes se transfieren los cambios realizados a la controladora quedando cargada tanto la configuración Hardware como software.

Por último, desde la *teach pendant* hay que indicar la posición de los TCP con los que hay que acceder a los puntos para que la controladora pueda realizar los cálculos para el control de movimientos. Los nombres de los TCP que se van a añadir tienen que ser los mismos que los que se han puesto en Tecnomatix en el modelado para que no surjan fallos. En la pantalla se debe indicar la posición del TCP respecto al último TCP del robot y la masa de la garra (figura 56). De esta manera el robot puede calcular el problema de posición, velocidad y dinámica inverso.

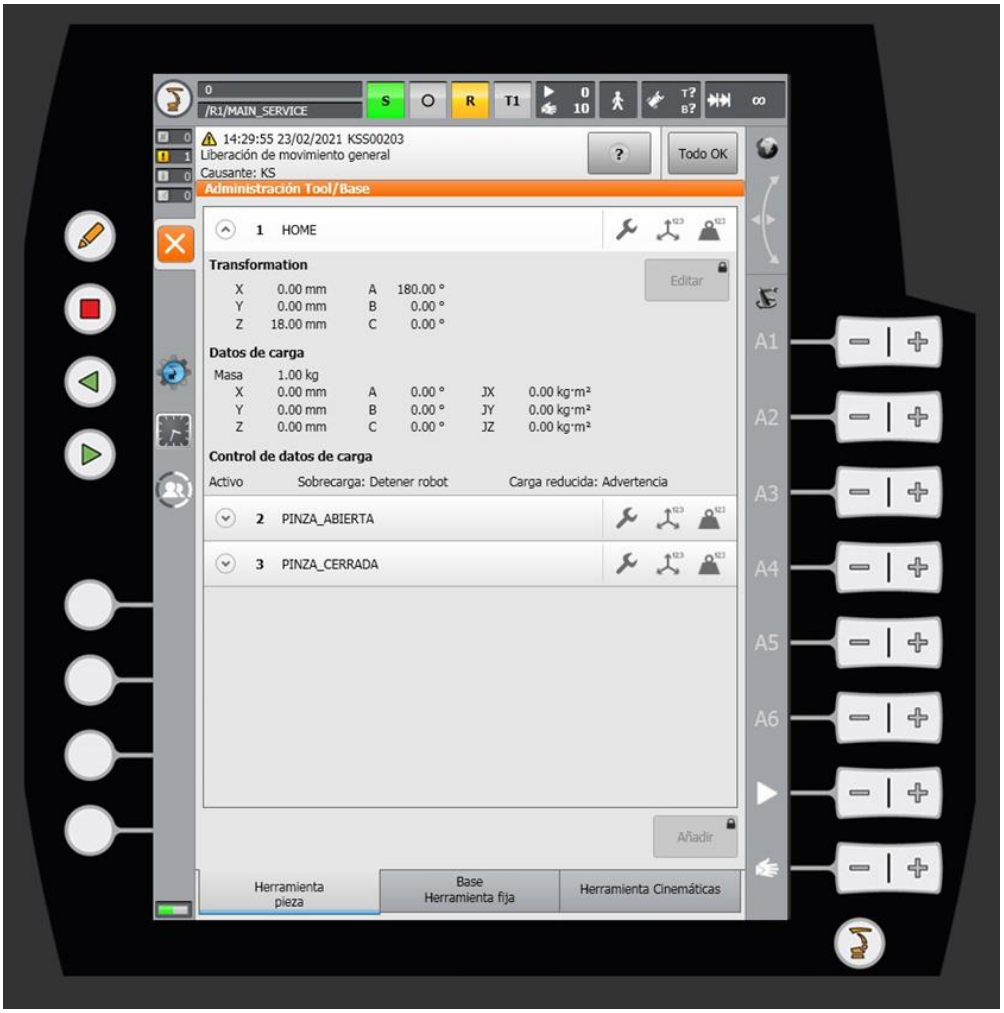


Figura 56. Configuración de TCP.

9.3.2 Carga de programas robot

El siguiente paso es realizar el volcado de los programas robot desarrollados en Tecnomatix PS. Los programas deben tener dos archivos diferentes *.src y *.dat. El primero contiene el código del programa, el segundo, sin embargo, los datos del programa, como los puntos del programa.

El archivo *.src contiene una cabecera con las declaraciones que se evalúan durante la compilación. Es esta cabecera no debe haber ninguna instrucción de programa. Después de la cabecera se encuentran las instrucciones En el código se encuentra las instrucciones de movimiento y otras funciones como la apertura y cierre de pinzas, activación, desactivación y evaluación de señales, sentencias de control de flujo etc.

El lenguaje de programación de KUKA, KRL agrupa el código de las instrucciones en FOLDs (figura 57), suprimiendo la visualización de algunos parámetros de estos. De esta manera el código queda más comprimido para la revisión. Estos FOLDs se pueden ampliar para ver todos los parámetros.

```
;FOLD PTP via133 Vel=10 % Pvia133 Tool[2]:PINZA_ABIERTA Base[0]
;FOLD LIN pick Vel=0.01 m/s Lpick Tool[2]:PINZA_ABIERTA Base[0] ;%{PE}
;FOLD Parameters
$BWDSTART = FALSE
LDAT_ACT= LLpick
FDAT_ACT= Fpick
BAS(#CP_PARAMS,0.01)
SET_CD_PARAMS(0)
LIN Xpick
;ENDFOLD
```

Figura 57. Visualización de una instrucción FOLD.

Desde Tecnomatix PS se selecciona el robot y se exporta el código creando los archivos *.src y *.dat de cada operación del robot (figura 58). Una vez se obtienes todos los archivos, existen dos maneras de añadir los programas a la unidad de control. Por un lado, se puede añadir los archivos desde WorkVisual mediante una carga *on line*. Por otro lado, es posible conectar un USB con los archivos al *teach pendant* y añadirlos desde allí. Aprovechando que el robot está conectado al WorkVisual de la carga de la configuración se van a cargar los programas *on line*.

Figura 58. Programas robot en la unidad de control visualizados desde la *smartPAD*.

9.4 Puesta en marcha

El robot cuenta con cuatro modos de funcionamiento:

- **T1**: Ejecución paso a paso del programa en modo manual con limitación de la velocidad. Es necesario mantener el pulsador de hombre muerto y la tecla de *run*.
- **T2**: Similar a T1, pero sin limitador de velocidad. La velocidad de ejecución es la indicada en el programa
- **Automático**: No es necesario mantener el pulsador de hombre muerto. La ejecución del programa comienza al pulsar la tecla *run* a la velocidad programada.
- **Automático externo**: El programa se ejecuta mediante una unidad de control superior, en este caso, el PLC.

Desde la smartPAD se selecciona el programa que se quiera ejecutar. En el modo T1 se comprueba de uno en uno los programas exportados realizando los ajustes pertinentes en caso de ser necesarios. Es común tener que ajustar algunas posiciones de *pick* y *place* más delicadas como las del bulón para que el ajuste se bueno. Durante la ejecución de los programas se comprueba que no hay colisiones y que todas las operaciones siguen la trayectoria correctamente.

Tras comprobar el funcionamiento de los programas sin errores ni colisiones se procede a preparar la estación para el funcionamiento automático. En primer lugar, se debe conectar la interfaz de seguridad ethernet al puerto X1 del PLC. A continuación, se selecciona el programa *main* como activo para que la ejecución comience como ha sido diseñada. Se carga en ODK los servicios y ítems a completar. Después, se pulsa el interruptor de marcha para que comience la secuencia de arranque. Una vez finalizada, se carga el contenido de ODK al PLC y comienza la ejecución llevándose a cabo el montaje tal y como se ha programado (figura 59).

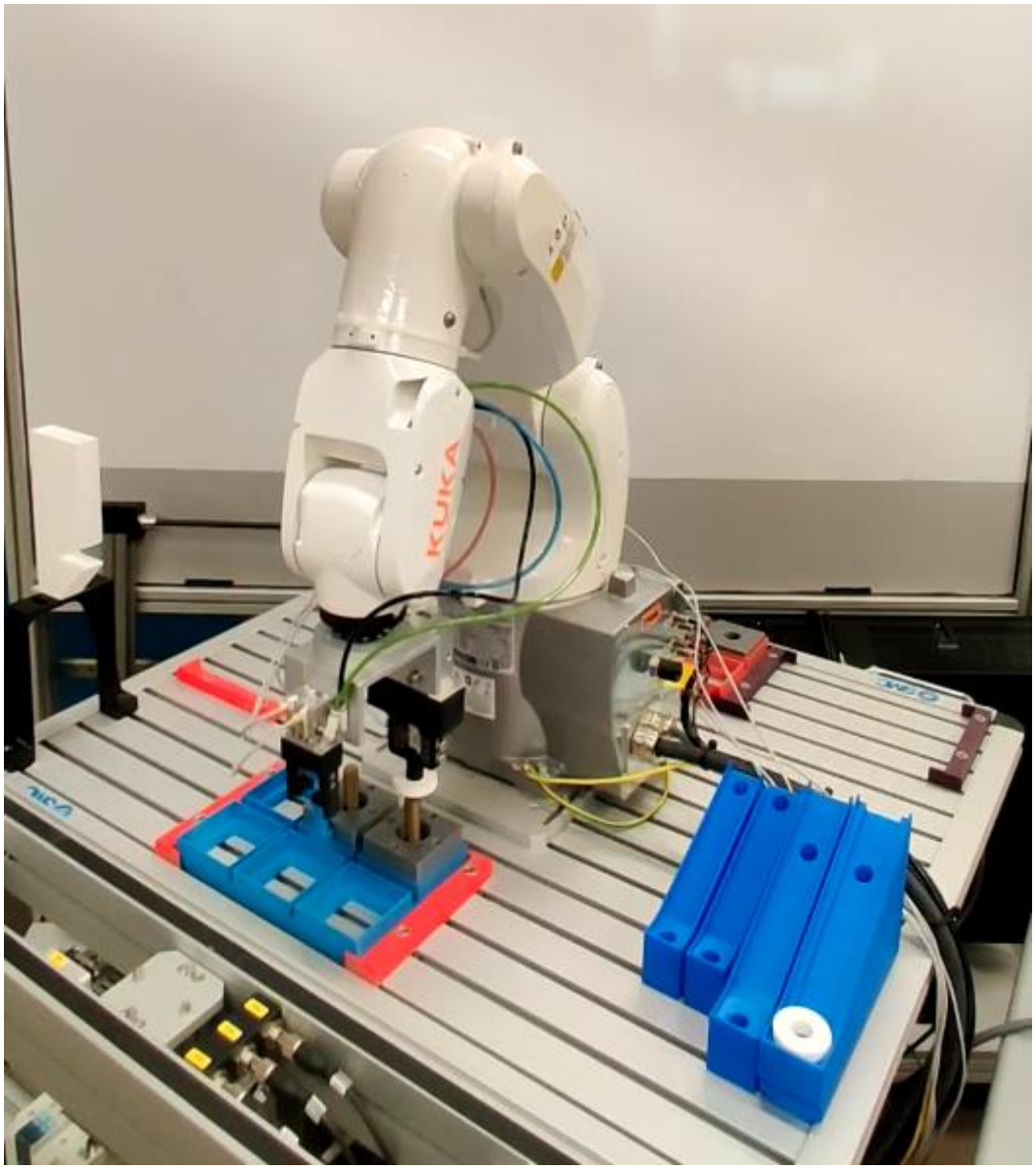


Figura 59. Robot durante un instante de la ejecución.

10 Descripción de tareas y diagrama de Gantt

Tareas previas

El comienzo del proyecto tiene como fecha el 29/10/2020

Propuesta y objetivos del TFM. Elección del tema del TFG y fijación de los objetivos a completar durante el desarrollo de este. Reunión en la fecha de comienzo del proyecto.

Estudio del contexto. Fase de búsqueda de información sobre las herramientas y métodos de trabajo.

Análisis de alternativas. Estudio y elección del software con el que se desarrollará el proyecto.

Formación con las herramientas

Debido a que las herramientas a utilizar no se estudian durante el master, se creó el gemelo digital de una máquina conocida con el fin de conocer el uso del software.

Compra e instalación del software.

Formación Tecnomatix PS. Familiarización mediante el modelado de una célula Fanuc realizada en proyectos anteriores. Esta tarea ocupa la mayor parte de este segmento con una duración de 60 días.

Modelado y puesta en marcha virtual

Fase más larga del proyecto, en él se recogen todas las tareas para el modelado del Gemelo Digital y la puesta en marcha virtual mediante *SiL*.

Modelado de estación robotizada. Creación del gemelo digital, tanto de sus movimientos como sus restricciones, y creación de señales necesarias para el control mediante PLC. Desarrollo del programa robot.

Duplicado de estación robotizada. Duplicado del gemelo digital para obtener las dos estaciones de la célula. Adaptación de las señales y programa robot para la segunda estación

Modelado de AGV. Inserción del AGV y creación de los bloques lógicos que lo gobierna. Creación de las señales para el control desde PLC

Desarrollo del código de control. Edición del código de automatización de PLC para cada estación y AGV. Comunicación entre PLCs mediante TCP/IP.

Puesta en marcha virtual. Puesta en marcha mediante *SiL* y validación tanto del modelo como de los programas robot y PLC.

Puesta en marcha real

Fase final del desarrollo, se han de adaptar los códigos desarrollados para la puesta en marcha real de la estación robotizada. Posteriormente se lleva a cabo la validación en la estación real.

Adaptación del control PLC. Adaptación del proyecto de automatización a una estación robotizada. Carga del proyecto al PLC

Volcado de programa robot y configuración. Configuración del robot y controladora. Carga de los programas extraídos de Tecnomatix PS.

Puesta en marcha real. Puesta en marcha y validación de la solución desarrollada.

Documentación

Redacción del documento en el que se recoge toda la información derivada del proyecto y preparación del documento para su entrega en la fecha 21/09/2021.

En las siguientes ilustraciones se recoge la información en un diagrama de Gantt (tabla 10 y figura 60):

Tabla 10. Tareas, fechas y duración del proyecto.

WBS	Task Name	Duration	Start	Finish	WBS Predecessors
1	**Tareas previas**	**12 days**	**Thu 10/29/20**	**Fri 11/13/20**	
1.1	Propuestas y objetivos	2 days	Thu 10/29/20	Fri 10/30/20	
1.2	Estudio del contexto	10 days	Mon 11/2/20	Fri 11/13/20	1.1
1.3	Análisis de alternativas	5 days	Mon 11/9/20	Fri 11/13/20	1.1
2	**Formación con las herramientas**	**32 days**	**Mon 11/16/20**	**Tue 12/29/20**	**1**
2.1	compra e instalacion del software	2 days	Mon 11/16/20	Tue 11/17/20	1.2,1.3
2.2	Formación Tecnomatix PS	30 days	Wed 11/18/20	Tue 12/29/20	2.1
3	**Modelado y puesta en marcha virtual**	**112 days**	**Wed 12/30/20**	**Fri 6/4/21**	**2**
3.1	Modelado de la estación	60 days	Wed 12/30/20	Tue 3/23/21	2.2
3.2	Duplicado de la estación	10 days	Wed 3/24/21	Tue 4/6/21	3.1
3.3	Modelado del AGV	12 days	Wed 4/7/21	Thu 4/22/21	3.2
3.4	Desarrollo del código de control	25 days	Fri 4/23/21	Thu 5/27/21	3.3
3.5	Puesta en marcha virtual	5 days	Fri 5/28/21	Thu 6/3/21	3.4
3.6	Validación	0 days	Fri 6/4/21	Fri 6/4/21	3.5
4	**Puesta en marcha real**	**40 days**	**Fri 6/4/21**	**Fri 7/30/21**	**3**
4.1	Adaptación del control PLC	10 days	Fri 6/4/21	Thu 6/17/21	3.6
4.2	Volcado del programa robot y configuración	25 days	Fri 6/18/21	Thu 7/22/21	4.1
4.3	Puesta en marcha real	5 days	Fri 7/23/21	Thu 7/29/21	4.2
4.4	Validación	0 days	Fri 7/30/21	Fri 7/30/21	4.3
5	**Documentación**	**26 days**	**Tue 8/17/21**	**Tue 9/21/21**	**4**
5.1	Escitura del TFM	26 days	Tue 8/17/21	Tue 9/21/21	4.4[FS+1 day]
5.2	Entrega	0 days	Tue 9/21/21	Tue 9/21/21	5.1

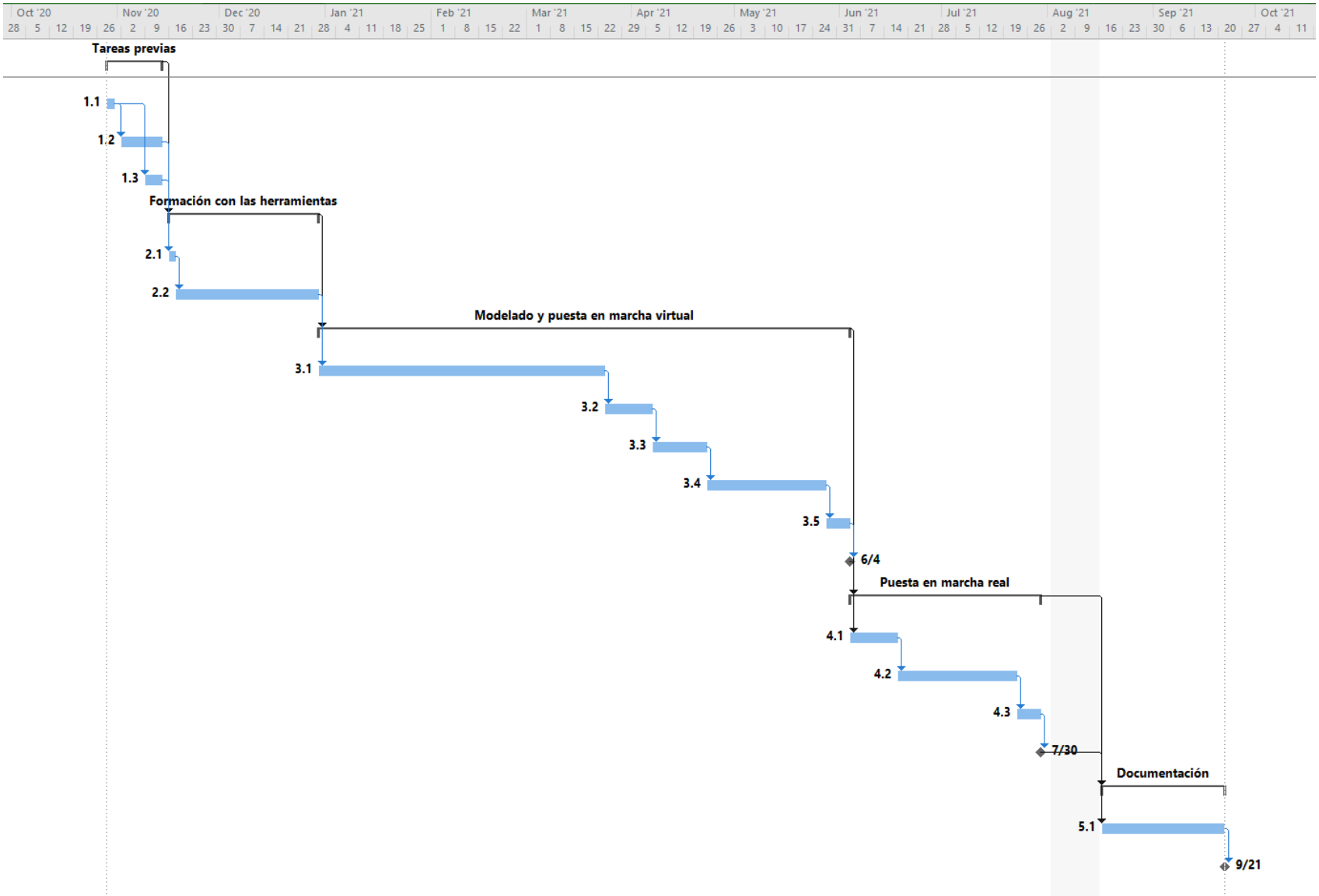


Figura 60. Diagrama de Gantt.

11 Presupuesto

En este apartado se muestra el presupuesto del proyecto. La mayor parte de este se destina a las horas internas tanto del alumno como del tutor. Para realizar una aproximación tanto el precio unitario del alumno, como los diferentes parámetros para el cálculo de las amortizaciones se ha tenido en cuenta el convenio del metal de Bizkaia. El convenio estipula 1700 horas de trabajo al año, que se han tenido en cuenta para el cálculo de la vida útil. Para el coste del alumno se ha usado el salario mínimo del convenio para ingenieros titulados, es decir, 20€/h. El coste del tutor se ha aproximado con un salario de 50€/h. Respecto a las amortizaciones se han incluido los equipos y licencias que pueden ser usados en otros proyectos. Por una parte, los PC utilizados para la simulación, modelado y edición del código, a los cuales se les ha estimado una vida útil de 5 años. Por otro lado, el robot y la garra, que se ha estimado en 10 años su vida útil. Por último, las licencias necesitan ser renovadas anualmente. En cuanto a los gastos se refiere, se han añadido tanto los costes de las piezas impresas, como el material de oficina y los documentos impresos. Estos gastos son reducidos en el total del proyecto.

La suma de lo anterior forma los costes directos, a los cuales hay que añadir los costes indirectos (electricidad, servicio de limpieza...) estimados en un 8% de los costes directos. Además, se creó una partida del 10% del subtotal para gastos imprevistos, por ejemplo, la compra de periféricos del ordenador en caso de rotura. El total del proyecto se estimó así en 27,941.05€. A continuación, en la tabla 11 se muestra esta información sintetizada.

En la figura 61 se puede observar el presupuesto desglosado. En ella se muestra que las horas internas representan mayor parte del presupuesto, un 68%. Las amortizaciones y licencias suman un 15.9% del total, siendo la segunda mayor categoría. A su vez, la mayoría de esta se debe a las licencias, 14.3%, debido a ser anuales. Aunque los costes de los PC y el robot superan las licencias, su mayor vida útil reduce el coste en el proyecto. Los costes indirectos e imprevistos suponen juntos un 15.8%. Por ultimo los gastos son prácticamente despreciables ya que solo representan un 0.3% del total.

Tabla 11. Resumen del presupuesto.

Horas internas	Nº de horas	Precio unitario	Total
Alumno	650	20.00 €	13,000.00 €
Director	120	50.00 €	6,000.00 €
			19,000.00 €

Amortizaciones	Coste	Cantidad	Nº de horas	vida útil (h)	Precio untario (€/h)	Amortización
Robot	18,000.00 €	1	150	16700	1.077844311	161.68 €
PC Modelado	1,500.00 €	1	780	8350	0.179640719	140.12 €
PC simulación	1,000.00 €	2	450	8350	0.119760479	107.78 €
Nodo EtherCAT	600.00 €	1	150	16700	0.035928144	5.39 €
Garra	1,200.00 €	1	150	16700	0.071856287	10.78 €
Mesa de trabajo SMC	1,500.00 €	1	150	16700	0.089820359	13.47 €
PLC ET200SP	1,800.00 €	1	150	16700	0.107784431	16.17 €
						455.39 €

Licencias	Coste	Cantidad	Nº de horas	vida útil (h)	Precio untario (€/h)	Amortización
TIA Portal + PLCSIM ADVANCED	1,200.00 €	1	550	1670	0.718562874	395.21 €
Tecnomatix	8,000.00 €	1	720	1670	4.790419162	3,449.10 €
WINCC Advanced	500.00 €	1	500	1670	0.299401198	149.70 €
						3,994.01 €

Gastos	Coste	Unidades	Total
Impresión	20.00 €	-	20.00 €
Material de oficina	10.00 €	-	10.00 €
Piezas impresas	40.00 €	-	40.00 €
			70.00 €

Costes directos	23,519.40 €
Costes indirectos (8%)	1,881.55 €
Subtotal	25,400.95 €
Imprevistos (10%)	2,540.10 €
TOTAL	27,941.05 €

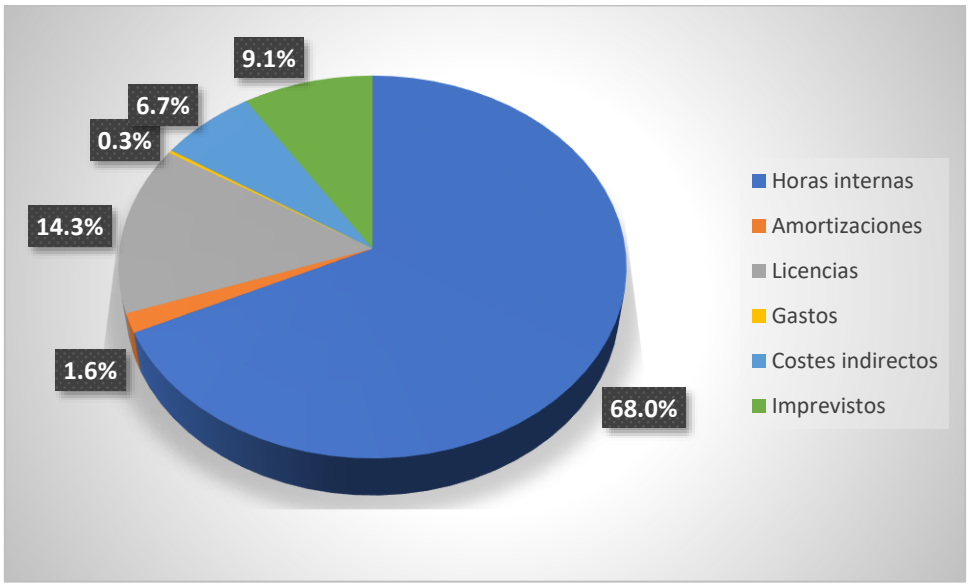


Figura 61. Desglose del presupuesto total indicado en porcentajes.

12 Conclusiones

El modelado de un Gemelo Digital de la célula de montaje ha sido satisfactorio. El Gemelo Digital es plenamente funcional. Reproduce los movimientos fielmente y es capaz de simular correctamente la activación y desactivación de las señales necesarias. Además, el desarrollo de este modelo prueba una de las fortalezas del *virtual commissioning*, ya que se ha creado una célula pese a solo contar con una estación en el laboratorio. De esta manera, se ha podido desarrollar el control de la célula antes de contar con una segunda estación real para completar la célula. Validando de esta manera el control y la automatización en simulación.

Tecnomatix PS se ha demostrado como una herramienta adecuada para este modelado. Sin embargo, su complejidad añade una barrera de entrada muy alta. Por esa razón, se echa en falta funciones que simplifiquen el modelado de Gemelos Digitales. Por suerte, Siemens mantiene constantemente actualizada la herramienta por lo que es posible añadir estas funciones en futuras versiones. El modelado del AGV está muy limitado y no es capaz de trasladarse al equipo real como en el caso del robot. Esta funcionalidad es añadida en la última versión del *software* por lo que se esperan mejoras en próximas. La programación robot ha sido sencilla y su volcado directo al robot real permite optimizar mucho el proceso. Sin embargo, faltan comandos del lenguaje KRL como SHIFT que optimizarían el programa. Siemens cuenta con una manera de superar este problema mediante lo que denominan *VRC (Virtual Robot Controller) Server* pero se ha descartado por complejidad y costes.

La exportación de los programas robots ha simplificado y acelerado la puesta en marcha real de la estación. Solo han sido necesarios ajustes en una posición de agarre del bulón por lo que el *virtual commissioning* se ha mostrado como una tecnología eficaz. Además, se ha logrado integrar el control dentro de la red de agentes gracias a los programas desarrollados por otros miembros del grupo. Por lo tanto, se ha cumplido el principal objetivo de desarrollar un demostrador de un proceso de fabricación de esta tecnología de manera tanto virtual como real.

Por último, el código desarrollado tanto para el control PLC como para el programa robot ha sido validado como una solución satisfactoria. Tanto la solución PLC para la puesta en marcha virtual con la comunicación TCP/IP entre PLCs, implementada de forma distribuida para separar los recursos de computación, como la implementación de la automatización de estación en el PLC real ha demostrado cumplir los objetivos. La programación robot también ha demostrado ser válida llevando a cabo sin errores ejecuciones de diferentes servicios y numero de ítems.

12.1 Líneas futuras

El siguiente paso natural sería ampliar la flexibilidad de la célula. Para ello se pueden añadir más estaciones a ella. Otra manera de ampliar esta flexibilidad sería eliminar algunas de las restricciones impuestas. La estación en la que se basa, la estación modular FMS-201, permite el montaje de un conjunto de piezas pequeño similar. Añadir la posibilidad de montar estas piezas, también añadiría flexibilidad a la célula, creando nuevos servicios para este conjunto.

Otro de los aspectos a investigar en el futuro es añadir un sistema de trazabilidad de la célula. En la industria 4.0 la trazabilidad adquiere una gran importancia. Es por ello que mejorar está podrían ser trabajos a futuro. Dotar a la célula de sistemas de visión para reconocer los pallets, o llevar a cabo un seguimiento de los rodamientos, bulones, etc que se encuentran a la estación añaden variables que se pueden introducir en el sistema de negociación de los agentes para tomar mejores decisiones.

En cuanto al *virtual commissioning* ha demostrado ser una herramienta muy potente para el análisis de la automatización y optimizar la puesta en marcha. Siemens cuanto con otras herramientas que elevan este concepto a nivel de planta llamado Tecnomatix Plant Simulation. La introducción de esta herramienta al análisis de la producción podría resultar satisfactorio. De esta manera se podría ampliar la capacidad de análisis del *virtual commissioning* aún más para optimizar la producción.

También sería interesante introducir el mencionado *VRC server* descartado en este proyecto. Una vez se cuenta con la célula desarrollada, se puede centrar los esfuerzos en realizar esta conexión con el software de KUKA. De esta manera el programa robot se ejecutaría en un simulador de la controladora desarrollada por KUKA que cuenta con todos los comandos del lenguaje KRL. Su conexión con Tecnomatix PS permite desarrollar el programa robot con todas las posibilidades del lenguaje. De esta manera se puede optimizar el código del programa robot logrando una solución mejor.

13 Bibliografía

[1] Rojko, A. (2017). Industry 4.0 Concept: Background and Overview. *International Journal of Interactive Mobile Technologies*, 11, 77-90. doi: g/10.3991/ijim.v11i5.

[2] Söderberg, R., Wärmefjord, K., Carlson, J. S., Lindkvist, L. (2017). Toward a Digital Twin for real- time geometry assurance in individualized production. *CIRP Annals*, 66, 137-140. doi: 10.1016/j.cirp.2017.04.038.

[3] Krause, H. (2007). Virtual commissioning of a large LNG plant with the DCS 800XA by ABB. 6th EUROSIM Congress on Modelling and Simulation. Ljubljana, Slovénie.

[4] Bargiela, A., Ali, S. A., Crowley, D., Kerckhoffs, E. J. H. (2010). *Simulation meets global challenges*. Dudweiler, Germany: Digitaldruck Pirrot GmbH.

[5] Reinhart, G. & Wünsch, G. (2007). Economic application of virtual commissioning to mechatronic production systems. *Computer Aided Engineering*, 1, 371-379. doi: 10.1007/s11740-007-0066-0.

[6] Delgado Román, M. C. (2009). Una revisión sobre los Sistemas Multiagente en la ingeniería de organización. *Dirección y organización: Revista de dirección, organización y administración de empresas*, *38*, 58–65. https://dialnet.unirioja.es/servlet/articulo?codigo=3055655

[7] SMC Corporation. (s. f.-a). *Operation Manual MHK2 Gripper*. SMC. Recuperado 23 de febrero de 2021, de https://static.smc.eu/binaries/content/assets/smc_global/product-documentation/operation-manuals/en/om_mhk_omf0058en.pdf

[8] SMC Corporation. (s. f.-b). *SY3000/5000/7000 Solenoid Valve Operation Manual*. SMC. Recuperado 23 de febrero de 2021, de https://static.smc.eu/binaries/content/assets/smc_global/product-documentation/operation-manuals/en/om_sy3000v_omm0002en-b.pdf

[9] Chen, H., Chu, C., & Proth, J. M. (1997). Sequencing of Parts in Robotic Cells. *International Journal of Flexible Manufacturing Systems*, *9*(1), 81–104. https://doi.org/10.1023/a:1007930010707