UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial,
Aeroespacial i Audiovisual de Terrassa

ESTUDIO DE LAS ETAPAS DE IMPLEMENTACIÓN DE UNA ARQUITECTURA CIBERFÍSICA PARA LA SUPERVISIÓN DE UNA CÉLULA INDUSTRIAL

Trabajo final de **Grado en Ingeniería Electrónica Industrial y Automática**

GERARD LLOBET ESPINOSA

Director:
MIGUEL DELGADO PRIETO
Codirector:
ÁNGEL FERNÁNDEZ SOBRINO

Departamento de Ingeniería Electrónica

26 de abril de 2021

I declare that,

the work in this Master Thesis / **Degree Thesis** (*choose one*) is completely my own work,

no part of this Master Thesis / **Degree Thesis** (*choose one*) is taken from other people's work without giving them credit,

all references have been clearly cited,

I'm authorised to make use of the company's / research group (*choose one*) related information I'm providing in this document (*select when it applies*).

I understand that an infringement of this declaration leaves me subject to the foreseen disciplinary actions by *The Universitat Politècnica de Catalunya - BarcelonaTECH*.

Gerard Llobet Espinosa		13/01/2021
Student Name	Signature	Date

Title of the Thesis:

ESTUDIO DE LAS ETAPAS DE IMPLEMENTACIÓN DE UNA ARQUITECTURA CIBERFÍSICA PARA LA SUPERVISIÓN DE UNA CÉLULA INDUSTRIAL

Índice de contenidos

Índice de figuras

Índice de tablas

Glosario

TÉRMINO	DEFINICIÓN
Algoritmo	Conjunto ordenado de operaciones sistemáticas que permite hacer un cálculo y hallar la solución de un tipo de problemas
Array	En programación se conoce como array a una zona de almacenamiento contiguo que contiene una serie de elementos del mismo tipo
Automatización	Aplicación de máquinas o de procedimientos automáticos en la realización de un proceso o en una industria
Base de datos	Conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso
Bus	Línea de transmisión enseriada de datos entre dispositivos
Campo	Nivel dentro de la pirámide CIM donde se gestionan los sensores del nivel inferior. En él aparecen los PLC's, robots y tarjetas de control
Célula industrial	Lugar físico, normalmente dentro de una planta industrial, donde se implican varia maquinaria para realizar una determinada tarea
CPU	*(Central Processing Unit).* Es el hardware dentro de un ordenador u otros dispositivos programables, que interpreta las instrucciones de un programa informático
Datos de series temporales	Colección ordenada de mediciones y datos tomados en un determinado momento, en intervalos regulares de tiempo y ordenados cronológicamente
Estructura ciberfísica	Resultado de dotar a los componentes u objetos físicos de capacidades de computación y de comunicación para convertirlos en objetos inteligentes que pueden cooperar entre ellos formando ecosistemas distribuidos y autónomos
Hardware	Conjunto de elementos físicos o materiales que constituyen una computadora o un sistema informático
IIoT	*(Industrial Internet of Things).* Uso del IoT con el objetivo de mejorar la eficiencia de los procesos industriales y de fabricación
IoT	*(Internet of Things).* Concepto que se refiere a una interconexión digital de objetos cotidianos con internet. Es la conexión de internet más con objetos que con personas
Nube	Paradigma que permite ofrecer servicios de computación a través de una red, que usualmente es internet

Objeto	En programación se conoce como objeto a un ente abstracto que permite separar diferentes componentes de un programa, simplificando así su elaboración, depuración y posibles mejoras
Pasarela	Equipo capaz de comunicar dos elementos de una red de comunicaciones con protocolos distintos
Pirámide CIM	Concepto que recoge los cinco niveles tecnológicos que se pueden encontrar en un entorno industrial. Las tecnologías se relacionan entre sí, tanto dentro de cada nivel como entre los distintos niveles a través de los diferentes estándares de comunicaciones industriales
Planta industrial	Lugar físico abastecido de máquinas, herramientas, y espacio, necesarios para la elaboración o producción de algún objeto material o servicio
PLC	*(Programmable Logic Controller).* Computadora utilizada en la ingeniería automática o automatización industrial para automatizar procesos
Plugin	Programa complementario a otro que amplía sus funcionalidades
Protocolo	Sistema de reglas entre dos o más entidades que permiten su comunicación para la transmisión de información
Rutina	Parte de un programa que se encarga de realizar tareas repetitivas
Sistema	Conjunto organizado de procesos donde la tecnología, la información, los equipos y las materias primas configuran productos
Sistema ciberfísico	Todo aquel dispositivo que integra capacidades de computación, almacenamiento y comunicación para controlar e interactuar con un proceso físico
Sistema embebido	Sistema aislado de computación diseñado para realizar una o algunas pocas funciones. No puede ser modificado para que realice otro tipo de tareas que no sean para los que esté diseñado
Software	Conjunto de programas y rutinas que permiten a la computadora realizar determinadas tareas
String	En programación se conoce como string un tipo de dato que contiene valores de caracteres concatenados
Variable	En programación, una variable está formada por un espacio que contiene una cierta información en el sistema de almacenaje y un nombre simbólico que está asociado a dicho espacio

Agradecimientos

El primer lugar, agradecer a Miguel y Ángel, director y codirector del presente Trabajo Final de Grado, por la dedicación mostrada y el tiempo invertido en la guía para la elaboración y redacción de todo el trabajo, tanto en su parte teórica como en su parte experimental.

Por otro lado, quiero agradecer a todos los compañeros de universidad que he ido teniendo a lo largo de la carrera, y sobre todo a Varas, Pere, Víctor, Sergio, Rafa, Laura y Joan, por soportar juntos esas duras épocas de exámenes y entregas finales, y quedarnos hasta altas horas de la noche repasando para poder sacar nuestra carrera universitaria adelante.

A todos mis amigos, y en especial a Miki, Alex, Bea, Laia y David. Por ser capaces de animarme en las épocas más duras de la carrera, y ser capaces de despejarme la mente en los momentos que más lo necesitaba. Gracias a todos por eso y por hacer de esta vida algo tan especial.

Y para finalizar, gracias a toda mi familia. Gracias a mi madre Montse, a mi padre Ernest y a mi hermano Roger. Por estar siempre a mi lado, soportarme y apoyarme en mis momentos más complicados, y animarme a alcanzar todas mis metas.

Sin todos vosotros esto no habría sido posible.

Resumen (ESP)

Hoy en día cada vez se está haciendo más presente el concepto de Industria 4.0 dentro de la sociedad productiva. Cada vez son más empresas las interesadas en ir implementando sistemas que permitan conexiones entre todos sus dispositivos, para así poder tener un mayor control sobre la productividad y más rapidez y facilidad ante cualquier tipo de incidencia.

El presente proyecto tratará de la implementación de un sistema basado en tecnologías pertenecientes a la Industria 4.0, aprovechando una infraestructura de un laboratorio de la Escuela Superior de Ingeniería Industrial, Aeroespacial y Audiovisual de Terrassa (ESEIAAT).

En este laboratorio se ubica una célula industrial controlada mediante autómatas programables Schneider. La finalidad de esta célula es simular el transporte de elementos mediante bandejas dentro de un entorno industrial.

Para la supervisión del sistema, se implementarán soluciones basadas en softwares de adquisición de datos, almacenamiento en base de datos y visualización de los mismos. Estos programas involucrados son el Node-RED, InfluxDB y Grafana.

Para la recogida de los datos se creará una simulación de los mismos a partir de un archivo de formato CSV, los cuales almacenaran los datos virtuales pero equivalentes a la célula real. Por otra parte, se generará una comunicación real con un sensor de vibración ubicado en un motor de la planta para comprobar el correcto funcionamiento de las comunicaciones.

Por último, a partir de los datos recogidos de la planta se realizarán cálculos que puedan sacar información relevante para la producción de la planta, generando así una supervisión que pueda llevar a la detección de posibles mejores y evitar errores dentro de la misma.

Al final de la memoria, se detallarán unas conclusiones de todo el trabajo realizado y las posibles mejoras que podrían implementarse al mismo, para crear un conjunto de propuestas que pudieran ayudar a la continuación de este trabajo.

Resum (CAT)

Avui en dia cada cop s'està fent més present el concepte d'Industria 4.0 dins de la nostra societat productiva. Cada cop són més empreses les interessades en anar implementant sistemes que permetin connexions entre tots els seus dispositius, per així poder tenir un major control sobre la productivitat i major rapidesa i facilitat davant de qualsevol tipus d'incidència.

El present projecte tractarà de la implementació d'un sistema basat en tecnologies pertanyents a l'Industria 4.0, aprofitant una infraestructura d'un laboratori de l'Escola Superior d'Enginyeria Industrial, Aeroespacial i Audiovisual de Terrassa (ESEIAAT).

En aquest laboratori s'ubica una cèl·lula industrial controlada mitjançant autòmats programables Schneider. La finalitat d'aquesta cèl·lula és simular el transport d'elements mitjançant safates dins d'un entorn industrial.

Per a la supervisió del sistema, s'implementaran soluciones basades en softwars d'adquisició de dades, acumulació en base de dades i visualització dels mateixos. Aquests programes involucrats són el Node-RED, InfluxDB i Grafana.

Per a la recollida de les dades es crearà una simulació dels mateixos a partir d'un arxiu en format CSV, els quals guardaran les dades virtuals però equivalents a la cèl·lula real. Per altra banda, es generarà una comunicació real amb un sensor de vibració ubicat en un motor de la planta per a comprovar el correcte funcionament de les comunicacions.

Per últim, a partir de les dades de recollida de la planta es realitzaran càlculs que puguin treure informació rellevant per a la producció de la planta, generant així una supervisió que pugui portar a la detecció de possibles millores i evitar errors dins de la planta.

Al final de la memòria, es detallaran unes conclusions de tot el treball realitzat i les possibles millores que podrien implementar-se al mateix, per a crear un conjunt de propostes que poguessin ajudar a la continuació del treball.

Abstract (ENG)

Nowadays, the concept of Industry 4.0 is becoming increasingly present in productive society. More and more companies are interested in implementing systems that allow connection among all their devices, so that they can have greater control over productivity and also faster and easier problem-solving capacity.

The present project will be devoted to the implementation of a system based on Industry 4.0 technologies, making use of the infrastructure of a laboratory in the School of Industrial, Aerospace and Audiovisual Engineering of Terrassa (ESEIAAT).

An industrial cell controlled by Schneider PLC is located in the above-mentioned laboratory. The purpose of this cell is to simulate the transport of elements by means of trays within an industrial environment. In order to monitor the system, solutions based on data acquisition, database storage, and visualization software will be implemented. The programs involved are Node-RED, InfluxDB and Grafana.

Data collection will be simulated using a CSV format file, which stores the virtual data that are equivalent to the data of the real cell. Moreover, real communication will be generated with a vibration sensor placed in an engine at the plant, to ensure the proper functioning of communications.

Finally, calculations will be made based on the data collected from the plant. These calculations can extract valuable information to the production of the plant, generating additional supervision and an insight that can lead to the detection of possible improvements and also to avoid errors within the plant.

The conclusions of the project will be detailed at the end of the paper, as well as potential improvements that could be implemented, with the aim of establishing a set of proposals that could encourage the continuation of this work.

1. Introducción

1.1. Justificación y alcance del proyecto

A causa de la constante evolución de la tecnología dentro de la industria y la introducción de IIoT dentro de nuestra sociedad productiva, cada vez más los equipos tienen la necesidad de comunicarse entre ellos desde cualquier parte del mundo. Hoy en día se tienen gran cantidad de sistemas capaces de gestionar enormes cantidades de información dentro de las industrias. Gracias a ello las empresas pueden optimizar sus resultados a base de estudiar los puntos en los que se puede mejorar la productividad.

El desarrollo de este trabajo permitirá ver cómo puede llevarse a cabo todo este proceso, desde la recopilación de los datos de una célula industrial, las comunicaciones con bases de datos en la nube, y la gestión y visualización de variables y resultados que puedan brindar información sobre el funcionamiento y estadísticas de una planta.

A lo largo de este proyecto, se estudiará la actual célula industrial del Aula SCHNEIDER, dentro del edificio TR2 del campus ESEIAAT-UPC de Terrassa. Este estudio permitirá ver qué funcionalidades dispone la célula, y cuáles son los datos que pueden interesar para el nivel más alto de la pirámide CIM.

Para llevar a cabo la comunicación con los múltiples sistemas automatizados que hay en la célula industrial, se dispondrá del equipo Magelis Edge Box de la casa Schneider Electric. Este equipo requiere el uso de varios programas para la comunicación, gestión de base de datos, cálculo y creación de una interfaz de usuario para crear una interacción hombre-máquina.

Los antecedentes de este equipo y los programas se incluirán en futuros puntos de esta memoria, donde también podrá verse la metodología usada con ellos y su uso final.

1.2. Objetivos

Este proyecto tratará el estudio para la implementación de un sistema basado en una estructura ciberfísica para supervisar una célula industrial ubicada en el edificio TR2 de la Universitat Politècnica de Catalunya de la localidad de Terrassa. La célula industrial consta de varios transportadores conectados entre sí a través de motores y rodeados de varios accionamientos neumáticos y sensores para interactuar con el producto. Todo este sistema está controlado por varias CPU’s de Schneider Electric comunicados entre sí.

Comunicando con todas las CPU’s de la célula se encuentra una estructura basada en el equipo Magelis Edge Box de Schneider Electric. Este equipo da múltiples soluciones mediante software y hardware para necesidades en las soluciones del IIoT.

El objetivo principal será el diseño y configuración a través de varios softwares como son los propios de Schneider Electric, Node-RED, InfluxDB y Grafana para monitorizar los datos de la célula industrial.

Por otra parte, y para darle un mayor antecedente y una mayor necesidad a este trabajo, se estudiará brevemente la evolución que ha sufrido la industria en los últimos siglos, junto con la implicación de este hecho para la vida socioeconómica de la población. Con esto, se darán unas conclusiones de las repercusiones que puede traer la gran automatización de la industria que estamos viviendo hoy en día.

1.3. Planificación

La planificación de este trabajo se dividirá en tareas de búsqueda y acopio de información, realizadas paralelamente con tareas más prácticas y técnicas que se realizarán en parte en el laboratorio.

Junto a estas tareas y para el correcto seguimiento del presente trabajo, se realizarán hojas de seguimiento del trabajo, así como la redacción de un Project Charter donde se podrá ver la definición del proyecto y sus alcances y objetivos.

Las fechas de realización de las tareas, seguidamente enumeradas, se pueden ver en el ANEXO I: GANTT PLANIFICACIÓN DE TRABAJOS, donde aparecen:

Tareas de búsqueda y acopio de información, y de seguimiento:

- Redacción Project Charter
- Entrega Project Charter
- Redacción Memoria TFG
- Entrega Memoria TFG
- Seguimiento I
- Seguimiento II
- Seguimiento III
- Preparación de la presentación
- Presentación delante del tribunal

Tareas prácticas y técnicas:

- Acopio de información
- Visitas presenciales al laboratorio
- Familiarización con la célula industrial
- Comunicación Edge Box - Nube
- Realización de software de adquisición de datos con Node-RED
- Guardado de los datos recopilados en InfluxDB
- Diseño de pantallas para mostrar la información en Grafana
- Realización de software de usuario
- Resolución de problemas

2. Estado del arte y antecedentes

2.1. Evolución de la industria

El ser humano ha tenido siempre una gran capacidad para convertir productos dados por la naturaleza en productos útiles para sí mismo. Desde la segunda mitad del siglo XVIII, y teniendo como epicentro inicial el Reino Unido, se está llevando a cabo una revolución constante de la industria, pasando así por varias etapas de mejora hasta llegar a la actualidad. Esta progresión se diferencia en cuatro etapas marcadas por uno o varios descubrimientos tecnológicos que las hizo avanzar a cada una de ellas.

En la Primera Revolución Industrial se experimentó el cambio de una vida basada en la agricultura, elaboración artesanal y tracción animal hacia otra más centrada en el inicio de la fabricación industrial y la mecanización. Esta revolución dejó paso a nuevas formas de energía, gracias principalmente al perfeccionamiento de la máquina de vapor llevado a cabo por James Watt, y las principales industrias que se vieron afectadas fueron la textil, metalúrgica y química.

En la Segunda Revolución Industrial se engloban los cambios producidos entre el periodo de finales del siglo XIX y principios del siglo XX. En esta etapa se produjeron cambios en las técnicas de fabricación, y se dejó paso a las nuevas fuentes de energía como el gas, el petróleo y la electricidad. Se produjeron grandes avances en transportes y comunicaciones, que cambiaron la forma de producir y a la comunidad científica y educativa. Todos estos cambios propiciaron un aumento en la producción, y se introdujo el concepto de producción en cadena.

A partir de este punto las siguientes revoluciones adquieren otros calificativos, incidiendo más en el sector donde se producen los avances de producción y no tanto en el orden de éstas. Esto puede ser debido al crecimiento tecnológico exponencial en el que nos hemos visto envueltos hasta hoy en día, creando tecnología a base de ella misma y haciendo difícil la propia concepción de la división de etapas en el avance industrial.

Hasta día de hoy, se ha estado produciendo la Revolución Científico-Tecnológica, o también llamada la Revolución de la Inteligencia. Este término fue acuñado por el sociólogo Jeremy Rifkin, y avalado por el propio Parlamento Europeo en junio de 2006. Esta revolución se ha visto caracterizada por la introducción de la utilización de las energías renovables, el desarrollo tecnológico en almacenamiento de energías y el impulso en la red eléctrica inteligente. A grandes rasgos, en esta etapa se desarrolló en gran medida la informática y la automatización dentro de las industrias, creando las conocidas tecnologías de la información y la comunicación (TIC).

En la actualidad la industria se compone de gran cantidad de equipos, formados por sensores, sistemas de adquisición de datos, sistemas de visión artificial y robots que ayudan al ser humano a

desarrollar un trabajo. Es innegable decir que a día de hoy la mayoría de países altamente industrializados están apostando por una fabricación automatizada y robotizada, invirtiendo así grandes cantidades de capital en la compra de estos sistemas. A principios de la segunda década del siglo XX la demanda de robots se ha ido incrementando anualmente en un 15%, hasta llegar a finales de década donde la cantidad de robots a nivel mundial ha aumentado hasta los 2,6 millones de unidades. Uno de los países más punteros en la introducción de robótica y sistemas automatizados es Corea del Sur, donde se estima que hay un total de 531 robots por cada 10.000 trabajadores.

A pesar de que estos datos hagan ver que se está acercando una nueva reindustrialización, ¿cuáles son esos descubrimientos y avances tecnológicos que nos pueden hacer ver que estamos viviendo otra revolución industrial? La respuesta está en la Industria 4.0.

2.2. Industria 4.0

La Industria 4.0 se presenta como un cambio en el sector técnico-económico, datado para la tercera década del siglo XXI. Se caracteriza por el gran aumento en la velocidad de los sistemas computacionales y la versatilidad de conectividad entre ellos, causa que genera un aumento masivo en la cantidad de datos a tratar, introduciendo así el término de *Big Data*.

Los principales pilares de este cambio en la industria se indican en la Figura 1:[GLE1]

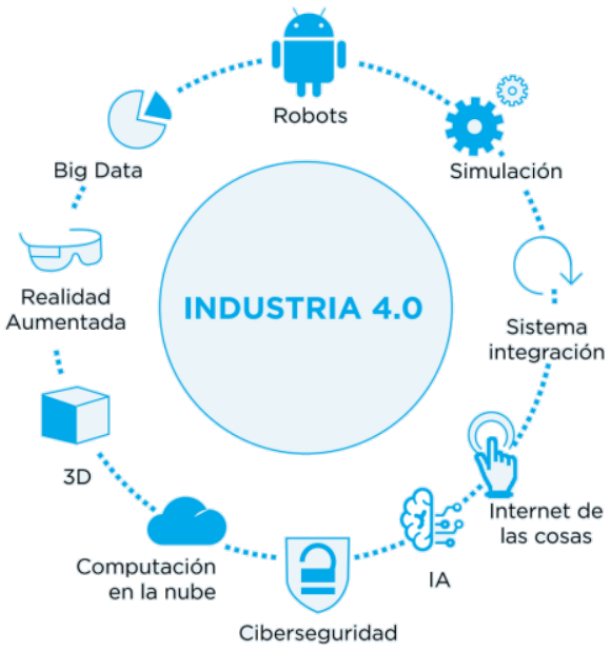

Figura 1: Pilares de la Industria 4.0. Fuente: AMETIC

I | Robots

Desde su invención, los robots han facilitado las tareas repetitivas y con necesidad de un gran esfuerzo físico al ser humano. Estos han generado un gran impacto en la industria, incrementando los beneficios sustancialmente gracias a su rapidez de trabajo, innecesidad de descanso y poco mantenimiento.

II | Simulación

Gracias a los grandes avances en la computarización, hoy en día pueden simularse grandes procesos de conjuntos de sistemas. Esto está ayudando a la reducción de costes, prevención de averías y a evaluar el resultado dentro de un entorno el cual podemos controlar y modificar.

III | Integración

Los sistemas de integración permiten comunicar los elementos y equipos de campo junto con los niveles de gestión y de planta. Esto genera grandes beneficios para la gestión de las plantas de la empresa, permitiendo estudiar gran cantidad de procesos para evaluar sus resultados, y focalizar los puntos donde mejorar.

IV | Internet de las cosas

Se conoce con este concepto a la interconexión de objetos cotidianos mediante internet. Constituye un gran cambio en la vida diaria de las personas, facilitando aún más las tareas diarias en el hogar, así como aumentando la seguridad y pudiendo mejorar la eficiencia energética dentro de las casas.

V | Inteligencia Artificial (IA)

La IA se trata de un gran avance en el mundo de la programación. Se define como el desarrollo de algoritmos capaces de evaluar grandes cantidades de datos, aprendiendo y generando nuevas rutinas en ese proceso. Esto genera programas capaces de aprender mediante la experiencia y ejecución, pudiendo mejorar su rendimiento, resultados y conclusiones con el tiempo.

VI | Ciberseguridad

El almacenamiento de grandes cantidades de datos en espacios como Internet genera muchas incertidumbres e inseguridades para las empresas en cuanto a la protección de estos. Es tarea de las herramientas de ciberseguridad el proteger estos datos detectando posibles amenazas para estos datos, así como anticipar estos ataques y ser capaces de neutralizarlos.

VII | Computación en la nube

Se trata del almacenamiento de datos en la nube, así como el acceso a servicios informáticos en línea. Genera gran flexibilidad en el acceso de recursos informáticos, agilizando los procesos normalmente administrativos.

VIII | 3D

Gracias a la simulación, previamente explicada, se genera un modelo tridimensional dentro de una computadora. Este modelo se puede crear físicamente gracias a la impresión 3D. Esta tecnología se basa en la impresión de varias capas de distintos materiales hasta la creación del objeto. Esto permite no necesitar moldes para la creación de un objeto, abaratando en gran medida procesos como pueden ser el prototipado.

IX | Realidad aumentada

La realidad aumentada es un recurso tecnológico que ofrece experiencias interactivas a un usuario a partir de la creación de una dimensión virtual unida con la dimensión física. Esta técnica presenta gran cantidad de ventajas, donde la mayoría y más conocidas se encuentran en el sector del entretenimiento. A pesar de ello, la realidad aumentada también tiene incidencia en diferentes ramas, como pueden ser la educación, medicina, sector inmobiliario, o desarrollo de estrategias para el sector del marketing.

X | Big Data

El Big Data se trata de la combinación y conjuntos de datos cuyo tamaño, complejidad y velocidad de ampliación, hacen de la tarea de gestión y procesado una meta difícil de alcanzar. El Big Data ofrece a las empresas respuestas y conclusiones a problemas que comúnmente no sabían ni que los tenían. Ofrece un gran control y mejora en la planificación comercial y de fabricación, y se la considera como una herramienta muy útil para ofrecer un producto concreto a un cliente determinado.

2.3. Impacto de la Industria 4.0 a la sociedad

Las revoluciones industriales han mejorado en gran medida la calidad de vida, generando un gran impacto en los sectores económicos y enormes transformaciones sociales.

Desde el inicio de la revolución industrial, la sociedad estamental se vio sustituida por una sociedad dividida en clases, cada una de ellas determinada por una serie de bienes materiales o terrenales. La nobleza, que poseía títulos nobiliarios otorgados por un rey o una herencia, se vio desplazada por la burguesía, que poseía el poder mediante el capital y los bienes. Se generó un éxodo desde los campos hacia las ciudades creando al proletariado, formado en su mayoría por trabajadores del sector industrial que únicamente poseían su salario. A día de hoy se presenta un panorama muy distinto. A pesar de la innegable división de clases sociales, un individuo puede ver mejorada su calidad de vida y calidad económica mediante su esfuerzo y capacitación, entre otros factores. Aun así, en la actualidad se está generando una incertidumbre a causa de la poca necesidad de mano de obra para un proceso de producción.

Acudiendo a algunos datos históricos, se puede observar que las revoluciones industriales han causado un impacto positivo en la sociedad si se examina la riqueza de las regiones. En la Figura 2[GLE2] se muestra el PIB per cápita del Reino Unido, cuna de la revolución industrial, y de España.

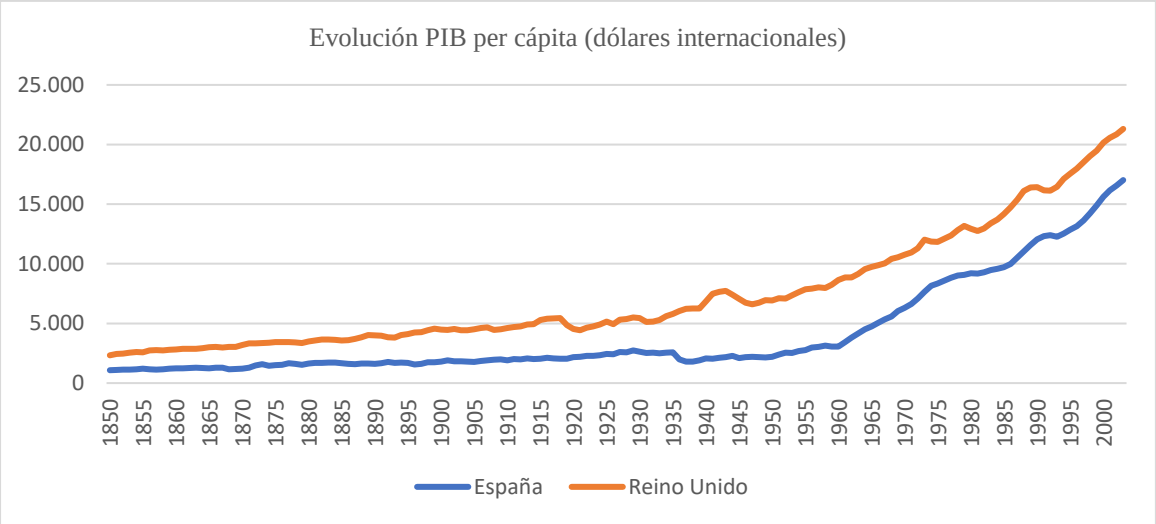

Figura 2: Evolución PIB per cápita. Fuente: Elaboración propia. Datos extraídos de Historical Statistics for the World Economy (Angus Maddison)

El crecimiento del producto interior bruto de un país se ve causado por, entre otros muchos factores, una gran cantidad de gasto destinado a producir más cantidad y mejor calidad de mercancía. Esto está involucrado con una gran inversión a nivel científico y tecnológico, mejorando el desarrollo de maquinaria, equipos e instalaciones, así como las telecomunicaciones y transportes. De esta forma, la consecuencia de un aumento del PIB de una región es un aumento en la producción de bienes y servicios, generando un mercado con mucha más oferta y disminuyendo la escasez, que es uno de los causantes de un bajo bienestar en un país.

El aumento del PIB no es la única causa para que hoy en día la sociedad viva en un mundo más acomodado. Un factor muy importante y determinante para compaginar correctamente la vida profesional con la personal, es la jornada laboral. En el mes de abril de 1919, en Barcelona se celebró una huelga en la central eléctrica La Canadiense durante un mes entero. Esto ocasionó la entrada en vigor de una ley que se aprobaría el 1 de octubre de ese mismo año, que fijaría la jornada laboral en ocho horas diarias. Más de cien años después la sociedad sigue amparada por esa ley pese al aumento masivo de la producción. Este hecho es posible gracias a todos los cambios y evoluciones que ha ido sufriendo la industria, que han facilitado los trabajos de una forma muy notable.

La Industria 4.0 está empezando a impactar sobre la organización de las empresas, y está surgiendo una gran preocupación por la posibilidad de que estos cambios puedan dejar sin trabajo a una gran

cantidad de personas. Un estudio realizado en 2018 por el Foro Económico Mundial, llamado *The Future of Jobs Report 2018*, reporta que en el período de 2018 hasta 2022 desaparecerá un total de 0,98 millones de empleo, pero por contraparte se crearán 1,74 millones de nuevos empleos, generados gracias a esta revolución industrial. Esto deja un balance positivo dentro de esta evolución.

En el año 2018, la proporción de tareas humano-robot se encontraba en torno al 71-29%. Se estima que para el año 2022, esta relación evolucione hasta un 58-42%. Esto causará que hasta el 42% de las habilidades que se requerirán en los trabajos cambiarán, necesitando un personal más cualificado y con una formación constante para la realización de los nuevos trabajos.

En resumen, es un hecho que una gran cantidad de empleos van a desaparecer ya que una gran mayoría de ellos serán sustituidos por sistemas automatizados. A pesar de ello, se crearán una gran cantidad de nuevos trabajos, muchos de ellos aún por descubrir, que necesitarán gente con una mayor formación para el correcto desempeño de esas labores, y que necesitarán una constante formación de cara a las nuevas tecnologías que irán surgiendo con el paso de los años.

2.4. Los sistemas ciberfísicos

Hasta hace unos pocos años era muy habitual que las máquinas de producción dentro de las industrias estuviesen aisladas entre sí. Cada máquina podía generar valores y datos que podían ser almacenados y visualizados, pero no era común, o posible, encontrar un flujo de información con otras máquinas operativas. Hoy en día la tendencia dentro de la industria es algo muy distinto, ya que cada vez se encuentra un mayor número de equipos interconectados entre sí dentro de una planta industrial. Gracias a ello se pueden enviar datos a otro punto de la planta que pueden ser interesantes para algún punto de la producción.

Para poder definir correctamente qué son y lo que implican los sistemas ciberfísicos, se debe conocer dos partes muy diferenciadas dentro de una industria hoy en día.

Primeramente, encontramos las tecnologías de operación (*OT* por sus siglas en inglés) que se basan en el conjunto de equipos y tecnologías utilizados dentro de un proceso destinados a realizar la operación de la fabricación. Dentro de esta definición se pueden encontrar algunos ejemplos dentro del ámbito industrial como los sensores, robots, PLC, SCADA… Este hardware y software detecta o provoca cambios mediante una monitorización o un control directo de los equipos.

Por otra parte, se encuentran las tecnologías de información (*IT* por sus siglas en inglés) que se basan en la computarización de la información para su almacenado, transmisión o manipulación de datos. De igual forma que las OT, esta tecnología también posee hardware y software, pero su

finalidad es muy distinta ya que se las requiere para los niveles más elevados dentro de la pirámide CIM.

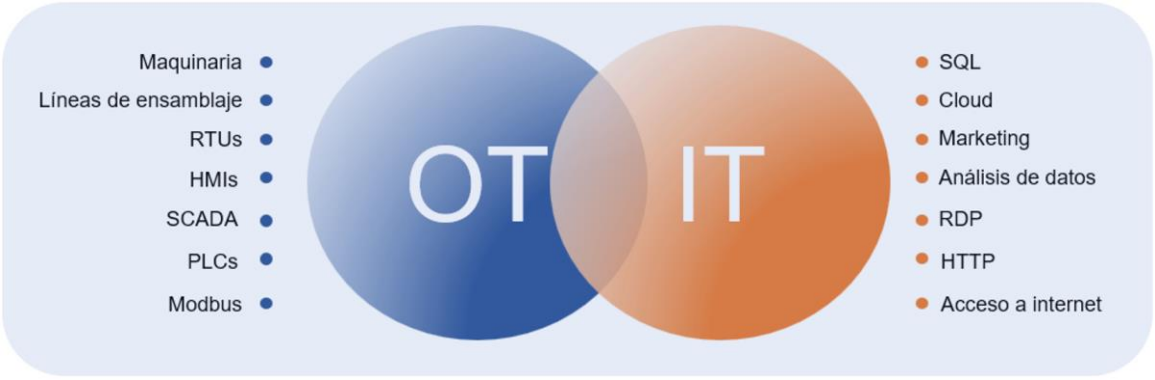

Figura 3: Ejemplos de OT e IT. Fuente: Elaboración propia

Los sistemas ciberfísicos son aquellos que unen las tecnologías de operación con las tecnologías de información. Son todos aquellos dispositivos que pueden integrar capacidades de computación, almacenamiento de datos y comunicación para controlar e interactuar con algún tipo de sistema físico.

La metodología de estos dispositivos conectados a un sistema ciberfísico es la generación de información y el posterior envío de ésta hacia unos servidores externos en tiempo real. Una vez esto ha ocurrido, se ponen en marcha herramientas de análisis de datos que son capaces de trasladar órdenes a la maquinaria o a otros dispositivos.

Cada vez más es común ver este tipo de sistemas en redes eléctricas inteligentes, sistemas de monitorización en medicina, sistema del vehículo autónomo, control de procesos en la industria, en el campo de la robótica y la domótica dentro de los hogares.

Previo a la existencia de los sistemas ciberfísicos, se encontraban los sistemas embebidos. Éstos son capaces de realizar una o unas pocas funciones para las cuales ha sido diseñado, y no es posible realizar otro tipo de tareas. En cambio, los sistemas ciberfísicos tienen una mayor adaptabilidad, capacidad y usabilidad, haciendo posible trabajar en conjunto con múltiples procedimientos generando un ecosistema distribuido y totalmente autónomo.

El gran aumento del uso de este tipo de sistemas que se está llevando a cabo hoy en día es debido en gran parte a las mejoras en la capacidad de procesamiento de los dispositivos, junto a su gran reducción de tamaño y los avances en comunicaciones, que han permitido una gran interconectividad de dispositivos.

2.5. Tecnologías aplicadas en este estudio

Previo a la elaboración de este estudio o a su lectura, es necesario el conocimiento de varias tecnologías que se verán involucradas en el desarrollo de esta implementación. Para ello, en los siguientes puntos se explicarán con detalle varios conceptos que harán más fácil la realización y la comprensión de este proyecto.

Las tecnologías que se explicarán a continuación están detalladas dentro del alcance de este proyecto, obviando así puntos históricos o técnicos que el autor considera poco relevantes para el caso de este estudio.

2.5.1. La pirámide de la manufactura integrada por computador (CIM)

La pirámide CIM es un concepto introducido por primera vez por el Dr. Joseph Harrington en un libro publicado a principios de los años 70 llamado *Computer Integrated Manufacturing* (CIM). En este libro se postulaba la idea de que la industria estuviera controlada por la nueva tecnología en aquellos tiempos, los ordenadores, y se proponía la idea de potenciar estas computadoras para integrarlas en su máximo exponente en el mundo industrial y así poder maximizar los resultados de los procesos de producción.

La pirámide CIM es la representación de las partes de un proceso industrial, desde la parte más cercana al proceso físico de fabricación hasta la parte de gestión de la producción. Así pues, se deben recordar los conceptos de tecnologías de operación (OT), y tecnologías de información (IT), explicadas con más detalle en el apartado 2.4. Los sistemas ciberfísicos[GLE3]. Cada una de estas partes se caracteriza por la tener una función concreta dentro del proceso de producción, y cada capa tiene comunicación directa con los niveles inmediatamente superior e inferior. En la Figura 4[GLE4] se muestran las divisiones que tiene esta estructura.

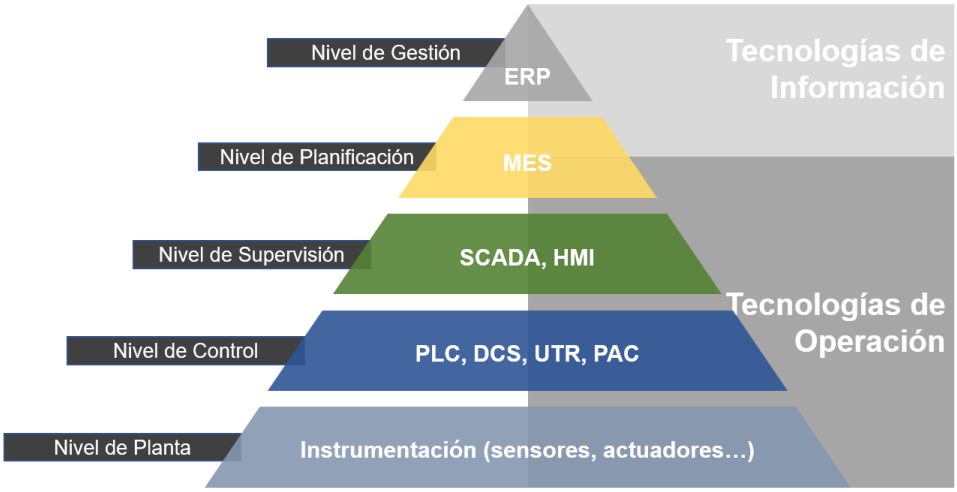

Figura 4: Representación de la pirámide CIM. Fuente: Elaboración propia

I | Nivel de Planta

Este nivel está formado por los elementos encargados de la medición del estado de los equipos y productos y los encargados de generar cambios físicos y actuar en ellos. Los sensores son los elementos que miden el estado de sistema, por ejemplo el nivel de un fluido, su caudal, su presión, o un final de carrera que detecta la posición de un producto o equipo. Por otra parte, los actuadores que son los encargados de generar cambios dentro de la planta pueden ser motores, válvulas o resistencias calentadoras.

II | Nivel de Control

Este nivel lo componen los elementos capaces de gestionar los actuadores y sensores que componen el nivel inferior. Está formado por los PLC, controladores de motores y sistemas robóticos. La función de estos dispositivos puede variar según las necesidades ya que son sistemas programables, y logran conjugar a los sensores y actuadores para que realicen el proceso industrial que se requiera. Mediante buses de campo es posible la interconexión entre estos dispositivos y la conexión directa con su capa superior.

III | Nivel de Supervisión

Este nivel es capaz de realizar la visualización de los estados de los procesos de la planta. A través de entornos de supervisión, control, y adquisición de datos (SCADA) se puede conseguir una imagen virtual de la planta para tener una supervisión de los procesos y posibles errores y fallos que puedan darse en la planta.

IV | Nivel de Supervisión

Este nivel lo componen los conocidos *Manufacturing Execution Systems* (MES), basados en softwares enfocados para el control de la producción para monitorizar y documentar la gestión de la planta. Las tareas que realizan los componentes de esta planta varían entre la planificación de la producción, la focalización en el producto y decisiones a nivel de planta.

V | Nivel de Gestión

Este nivel está formado por sistemas de planificación de recursos empresariales y de gestión de la información, capaces de automatizar muchas prácticas de negocio asociadas con aspectos de operación y producción de una empresa.

2.5.2. Fundamentos básicos de las comunicaciones industriales

En cualquier forma de comunicación deben intervenir un emisor y un receptor. Para que sea posible el entendimiento entre estos dos involucrados, el emisor debe transmitir la información mediante un código común con el receptor. Esta idea puede ejemplificarse mediante el idioma que usamos

para comunicarnos entre nosotros. En cambio, si llevamos esta idea hacia un ámbito más industrial, el código de transmisión es identificado como el protocolo de comunicación.

Para que sea posible esta comunicación debe haber un medio a través del cual se transmita el mensaje. De igual forma que se ejemplifica en el párrafo anterior, el acto de hablar entre personas se transmite a través del aire. Llevándolo de nuevo a un terreno más industrial, el mensaje puede transmitirse a través de un cableado físico o con tecnología inalámbrica.

Las comunicaciones industriales codifican los mensajes siguiendo una serie de normas, creando así el protocolo de comunicación. La información se transmite mediante una trama de señales binarias que se activan y desactivan siguiendo un conjunto de normas basadas en su frecuencia de conmutación, amplitud de la señal, y las propias reglas que rige el protocolo para poder ser interpretadas.

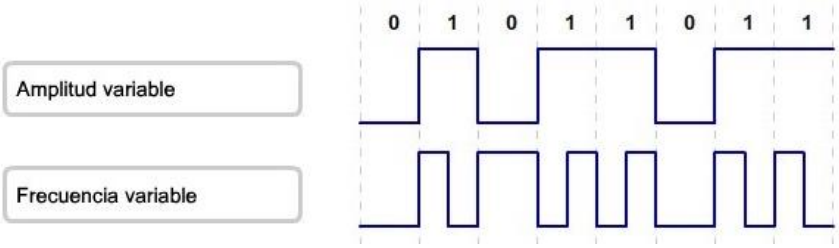

Figura 5: Formas de representar una señal en el medio. Fuente: Curso en línea CCNA1

Una vez introducidas las partes y reglas que forman las comunicaciones industriales, se debe tener en cuenta las características técnicas que deberá tener una instalación para poder implementar una comunicación entre los equipos.

I | Medio o soporte físico

El medio o soporte físico será el elemento por el cual se transmitirá la información. Dentro de este criterio existe el tipo de medio que será utilizado para la transmisión de las señales, la arquitectura o topología que se usará en la comunicación, y el tipo de conexionado que se usará.

Hoy en día existen dos tipos de medio, el cableado y el inalámbrico. El medio cableado se forma por un subconjunto de tipos de cables con unas características propias, como son la velocidad de transmisión, inmunidad a las interferencias causadas por ondas electromagnéticas externas, o la distancia máxima a la que pueden conectarse los equipos. En la Tabla 1[GLE5] se pueden observar algunos ejemplos de medio cableado junto a sus características.

Medio	Velocidad de transmisión	Inmunidad ondas E.M.	Distancia máxima
Par trenzado	100 kbits/s – 500 kbits/s	Muy baja	200 metros
Par trenzado apantallado	100 kbits/s – 500 kbits/s	Media	1 kilómetro
Coaxial (banda ancha)	300Mbits/s	Media	10 – 50 kilómetros
Fibra óptica (monomodo)	100 – 1.000 Gbps	Alta	100 kilómetros

Tabla 1: Ejemplos de medios cableados. Fuente: Elaboración propia

Un concepto necesario para la interpretación de estas características es la diferencia entre dos tipos de dispositivos: los maestros y los esclavos. El maestro es un equipo integrante en la comunicación capaz de iniciar una comunicación, en cambio el esclavo, también dentro de una red, no es capaz de iniciarla, y su función es dar respuesta a las peticiones que llegan desde el maestro.

La arquitectura, o comúnmente también llamada topología de conexión, es la forma en la que se conectan los dispositivos entre ellos. Algunas arquitecturas comúnmente utilizadas en la industria son la conexión en bus, estrella, punto a punto o en malla. Cada una de estas topologías da al sistema unas características independientes en cuanto al acceso a la información, solidez, rendimiento, diagnóstico de problemas o velocidad de transmisión entre otras.

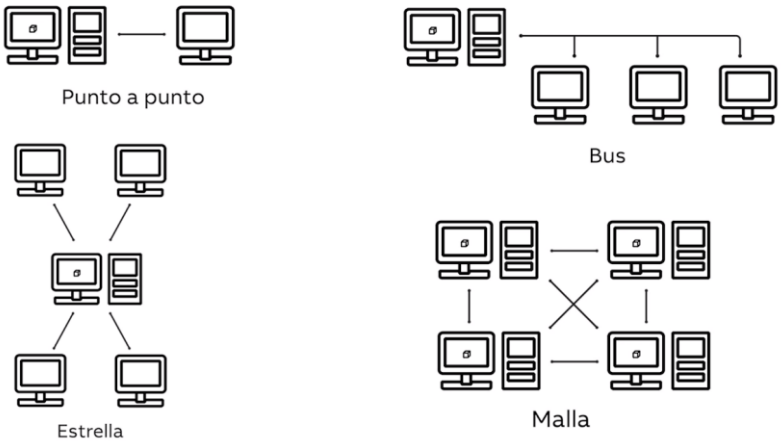

Figura 6: Ejemplos de topologías de conexión entre dispositivos. Fuente: ABB

El tipo de conexionado de una red se diferencia por utilizar una comunicación en serie o en paralelo. Una comunicación en serie es aquella en la que el maestro solo puede iniciar peticiones de información a los esclavos en diferentes tiempos, pudiendo solo iniciar una nueva petición a otro esclavo solo en el momento que haya procesado la respuesta del anterior. En cambio, una comunicación en paralelo permite al maestro iniciar una o varias peticiones al mismo tiempo, siendo también capaz de procesar una o varias respuestas de los esclavos. En este último tipo de conexionado, los equipos pasan de llamarse maestro y esclavo a cliente y servidor respectivamente.

En la Tabla 2[GLE6] se muestran algunos ejemplos de buses y redes de comunicación.

Buses de comunicación	Redes de comunicación
ModBus RTU	ModBus TCP
CANopen	ProfiNet
Profibus DP	Ethernet IP

Tabla 2: Ejemplos de buses y redes de comunicación. Fuente: Elaboración propia

El tipo de conexionado también caracterizará el tipo de conector que será usado en la comunicación. La elección de los conectores es independiente al sistema de comunicaciones, haciendo de esta elección algo dependiente de las necesidades que tengan los equipos que se integraran o las especificaciones.

II | Protocolo de comunicación

Como se ha comentado anteriormente, un protocolo de comunicación es un conjunto de reglas usadas para definir la forma de interpretar la transferencia de datos entre los diferentes dispositivos que forman la red. Hace unos años, era necesario que este protocolo fuera compatible entre los emisores y los receptores, pero hoy en día se pueden poner en uso las pasarelas, capaces de conectar dos equipos con protocolos distintos convirtiendo uno o más protocolos para que los distintos equipos de la red puedan entenderse entre ellos.

Hoy en día existen multitud de protocolos de comunicación. En la Tabla 3[GLE7] se muestran algunos de los más utilizados hoy en día en la industria, junto con algunas características que se han ido mencionando en puntos anteriores.

Protocolo	Topología	Medio	Velocidad de transmisión
Profibus	Bus lineal	Par trenzado apantallado	9,6 – 12.000 kbps
Modbus	Bus lineal	Par trenzado apantallado	300 – 19.200 bps
CANopen	Bus lineal	Par trenzado apantallado	50 – 1.000 kbps
As-Interface	Bus lineal, árbol o estrella	Cable de 2 hilos	167 kbps
Ethernet	Malla	Par trenzado apantallado	10 – 100 Mbps
	Estrella	Fibra óptica	100 – 1.000 Mbps

Tabla 3: Ejemplos de protocolos de comunicación. Fuente: Elaboración propia

III | Compatibilidad entre equipos

La compatibilidad entre equipos se aplica siempre que quiera modificarse o ampliarse un sistema de comunicación ya existente. Es necesario tener en cuenta las características anteriormente explicadas, medio y protocolo, y que los equipos nuevos que vayan a instalarse o modificarse cumplan como mínimo con las especificaciones máximas de los equipos en la red ya instalada.

2.5.3. Los autómatas programables (PLC)

Los autómatas programables (comúnmente también llamados PLC) se definen como equipos electrónicos capaces de controlar en tiempo real los procesos industriales. Uno de los principales motivos por los que fueron desarrollados fueron las dificultades que presenta el control a base de relés con lógica cableada de los sistemas de control. El control llevado por autómatas presenta una mayor versatilidad, eficiencia y adaptabilidad frente a los sistemas de relés.

En los siguientes apartados se explicarán sus varios puntos de sus características que permitirán comprender su necesidad y su alcance para los trabajos de control.

I | Ámbitos de aplicación

Hoy en día pueden encontrarse autómatas programables en gran cantidad de tipologías de factoría, como son la industria del automóvil, petroquímicas y plantas químicas, metalurgia, alimentación, y un extenso listado donde se aplican en trabajos de:

- Control de dosificaciones, mezclas, pesaje de productos
- Cadenas de montaje
- Sistemas de clasificación de materiales o productos
- Envasado, empaquetado, embotellado o almacenaje
- Control de sistemas de horneado
- Control de calderas y sistemas de refrigeración

II | Estructura del autómata programable

Las partes fundamentales que forman a un autómata programable son la Unidad Central de Procesado (CPU por sus siglas en inglés), la Memoria y el Sistema de Entradas y Salidas (E/S).

La CPU es la encargada de procesar toda la información que se interpreta del programa que se ejecuta dentro de la misma. La CPU interpreta las señales que le llegan a través del sistema de entradas para procesar con una lógica, siguiendo el programa que esté ejecutándose, y finalmente ejecutar unas señales a través del sistema de salidas que generarán en última instancia unos cambios físicos en la planta donde se encuentre el autómata.

La memoria del autómata se divide en dos clases generales: la memoria ROM, encargada de almacenar el programas para el correcto funcionamiento del equipo, comúnmente llamados *firmware*; y la memoria RAM, que se divide en una parte donde se almacena la información de los estados de las entradas, de las salidas y de las variables internas, comúnmente llamadas marcas según el fabricante, y en otra parte donde se almacena el propio programa que se está ejecutando y con el que el autómata trabajará.

El sistema de entradas se encarga de recoger datos de la instalación para posteriormente poder ejecutarlos dentro del programa. Se dividen dos clases de tipos de señales, las digitales y las analógicas. Las digitales son esas entradas que se pueden interpretar únicamente como dos valores, como en el caso de pulsadores, interruptores, finales de carrera o fotocélulas. Las señales analógicas son aquellas que disponen de un rango mucho más elevado de valores, como es el caso de sensores de temperatura, sensores de presión o de nivel.

El sistema de salidas se encarga de ejecutar las acciones dentro de la planta de manera física. Así como las entradas, las señales pueden se de tipo digital o de tipo analógico.

Las partes anteriormente explicadas forman un sistema autómata básico dentro de la industria, y hoy en día se pueden encontrar multitud de otros módulos con distintas funcionalidades. Entre estos módulos, se pueden encontrar por ejemplo módulos de comunicaciones que son capaces de transferir datos hacia otros sistemas o servidores, pasarelas de comunicación para realizar la conexión entre dos sistemas que utilicen diferente protocolo, o sistemas externos de entradas y salidas para conseguir una arquitectura más descentralizada y ubicada prácticamente a pie de máquina.

III | Funcionamiento general

El funcionamiento general de los autómatas viene dado por una primera lectura de los datos de entrada, con los cuales se recogen los estados del sistema a controlar en un momento indicado y a tiempo real. Posterior a esta lectura de las entradas, se ejecuta el programa teniendo en cuenta estos valores recogidos, para generar posteriormente a este paso la ejecución en las salidas, donde se generarán cambios del sistema. A este ciclo se le suele denominar ciclo de *scan*, y suele ser de suma importancia que tenga un valor lo más mínimo posible. Este valor mínimo ayudara a generar reacciones casi instantáneas en el sistema a controlar.

Los lenguajes de programación de los PLC varían en función de su fabricante, y algunos ofrecen más versatilidad y otros menos. En general, todas las marcas de autómatas presentan los siguientes lenguajes:

- Lista de instrucciones (IL o STL): Lenguaje de texto y de bajo nivel, el cual se suele utilizar para aplicaciones pequeñas debido a la complejidad de su estructura, y tiene una gran similitud con el lenguaje ensamblador.
- Lenguaje estructurado (ST): Lenguaje de texto y bajo nivel, el cual se utiliza para codificar expresiones aritméticas complejas con valores analógicos y digitales, y se dispone de estructuras de bucle, condicionales y funciones.
- Lenguaje de contactos (LD o KOP): Lenguaje gráfico y de alto nivel con una estructura muy similar a un esquema eléctrico. Es de los lenguajes más utilizados en el campo gracias a su fácil comprensión.
- Diagrama de bloques (SFD): Lenguaje gráfico y de alto nivel que utiliza los símbolos lógicos en forma de bloque donde se albergan las variables que transforman la secuencia.

2.5.4. Variables utilizadas en los autómatas programables

Dentro del ámbito de la programación, se conoce como variable a un espacio de almacenamiento dentro del sistema para guardar algún tipo de valor, y que lleva consigo un nombre identificativo que apunta a ese espacio. Simplificando y viendo un uso más práctico para este concepto, las variables son espacios de memoria para almacenar información que luego se podrán usar en algún punto del programa.

En programación suele ser útil la optimización y el uso del espacio justo y necesario para las variables, es por ello por lo que existen varios tipos de variables las cuales ocuparán una cantidad de memoria específica y podrán representar un número dentro de un rango concreto de posibilidades. Unos puntos más adelante, se explicarán los tipos de variable más comunes dentro del ámbito de los autómatas programables.

Antes de eso, merece la pena realizar una mínima explicación del concepto de bit dentro del ámbito de la informática. El bit es la mínima unidad de información utilizada por las computadoras, y la única unidad que pueden interpretar y realizar variaciones en nuestro entorno con ellas. Únicamente pueden tener dos valores, que suelen tener varias interpretaciones, pero a la práctica su uso es exactamente el mismo: cero o uno, encendido o apagado, o también *true* o *false.*

Siendo así, y sabiendo que las computadoras únicamente pueden interpretar valores de bit, cualquier otro valor que se pueda dar dentro de un programa se consigue gracias a la concatenación y unión de estos bits, formando así números más representativos en código binario y con un mayor número de posibilidades, que ahora sí, se explicaran en los siguientes puntos:

I | Booleano

Los booleanos representan a un valor de bit, que como se ha explicado anteriormente es la cantidad mínima de información que posee una computadora, y únicamente puede devolver o interpretar los valores de 0 o de 1. La representación que suelen tener dentro de los autómatas programables es a partir de la dirección, o nombre de variable, de %M0.0, %M0.1… Como se explicará en el siguiente punto, esta representación viene dada por el apunte a un bit en concreto que hay dentro de un byte.

II | Byte

Un operando de tipo Byte es una secuencia de 8 bits consecutivos, con lo que se puede representar una cantidad total de 256 valores, que su rango dependerá de si los valores son enteros con signo o enteros sin signo. Su representación viene dada a partir de la dirección %MB0, %MB1… En la siguiente figura se puede observar una representación más gráfica de las variables de tipo byte y los booleanos:

0	1	0	1	0	1	1	0	1	1	0	1	1	0	0	1
%M1.7	%M1.6	%M1.5	%M1.4	%M1.3	%M1.2	%M1.1	%M1.0	%M0.7	%M0.6	%M0.5	%M0.4	%M0.3	%M0.2	%M0.1	%M0.0
%MB1								%MB0							

Figura 7: Representación de bits y bytes. Fuente: Elaboración propia

III | Word

Se conoce como Word a una concatenación de 16 bits de memoria. Con esta variable se consigue un total de 65.536 combinaciones posibles, que de igual forma que los valores de las variables de byte irán determinados por si los valores tienen representación de enteros con signo o enteros sin signo. Su representación viene dada a partir de la dirección %MW0 (que ocuparía los bytes %MB0 y %MB1), %MW2 (que ocuparía los bytes %MB2 y %MB3) …

IV | Double Word

Se conoce como Double Word a una concatenación de 32 bits de memoria. Con esta variable se consigue un total de 4.294.967.296 combinaciones posibles, que de igual forma que los valores de las variables de byte irán determinados por si los valores tienen representación de enteros con signo o enteros sin signo. Su representación viene dada a partir de la dirección %MW0 (que ocuparía los bytes %MB0, %MB1, %MB2 y %MB3), %MW4 (que ocuparía los bytes %MB4, %MB5, %MB6 y %MB7) …

Estas variables son las básicas utilizadas por un sistema de PLC. Así pues, existen varias representaciones en función del código de numeración que se utilice. Las explicadas anteriormente

son las variables a partir de números binarios, pero también existen las variables con números enteros y los números con coma flotante. En la siguiente tabla se muestra una comparativa del tamaño que ocupan y su relación entre código de numeración:

Ocupación de memoria	Números binarios	Números enteros	Números de coma flotante
1 bit	Booleano	N/A	N/A
8 bits	Byte	Short Integer (SINT) / Unsigned Short Integer (USINT)	N/A
16 bits	Word	Integer (INT) / Unsigned Integer (UINT)	N/A
32 bits	Double Word	Double Integer (DINT) / Unsigned Double Integer (UDINT)	Real (REAL)
64 bits	N/A	Long Integer (LINT) / Unsigned Long Integer (ULINT)	Long Real (LREAL)

Tabla 4: Ocupación de memoria de variables según código de numeración. Fuente: Elaboración propia

A todas estas variables dentro del sistema de memoria de los PLC se les llama marcas, y vienen indicadas por una %M (por ejemplo, %M40.0 en caso de un bit). Estas marcas, son las variables, el espacio de memoria interna del PLC para almacenar resultados de combinaciones de entradas y salidas.

Para el caso de las entradas y salidas, la nomenclatura utilizada para hacer referencia a ella suele venir determinada por el fabricante del autómata. En este caso en concreto, y como regla casi general para todos, los PLC de Schneider suelen indicar sus relaciones a puntos de entrada y salida como, por ejemplo, %I1.3 en le caso de las entradas y %Q4.0 en el caso de las salidas.

3. Descripción del sistema de estudio

Como se ha comentado en puntos anteriores, el presente proyecto se basa en la implementación de un sistema ciber físico para la célula industrial del aula 004 del edificio TR2 de la Escuela Superior de Ingenierías Industrial, Aeroespacial y Audiovisual de Terrassa (ESEIAAT), de la Universitat Politècnica de Catalunya (UPC). En los siguientes apartados se comentarán los elementos que forman esta célula, así como su arquitectura.

3.1. La célula industrial

La célula industrial está diseñada para la simulación del transporte de materiales y recursos de una planta industrial.

Figura 8: Imagen de la célula industrial a estudio. Fuente: Elaboración propia

Este sistema está formado por un total de 4 PLC de la marca Schneider Electric, que se encargan del control automático de la célula. Estos controladores están conectados a campo mediante diferentes buses de comunicación, siendo estos el Profibus, CANopen, Ethernet y AS-Interface. Cada PLC se encarga del control de una sección de la célula industrial, generando así un sistema distribuido donde los controladores se encuentran unidos entre sí mediante una red de comunicaciones.

En la Figura 9[GLE8] se muestra un esquema de las partes que controla cada uno de los PLC junto al bus de comunicación utilizado. También se muestran otras nomenclaturas para la identificación de varias partes y equipos de la célula industrial que serán explicadas en los siguientes puntos.

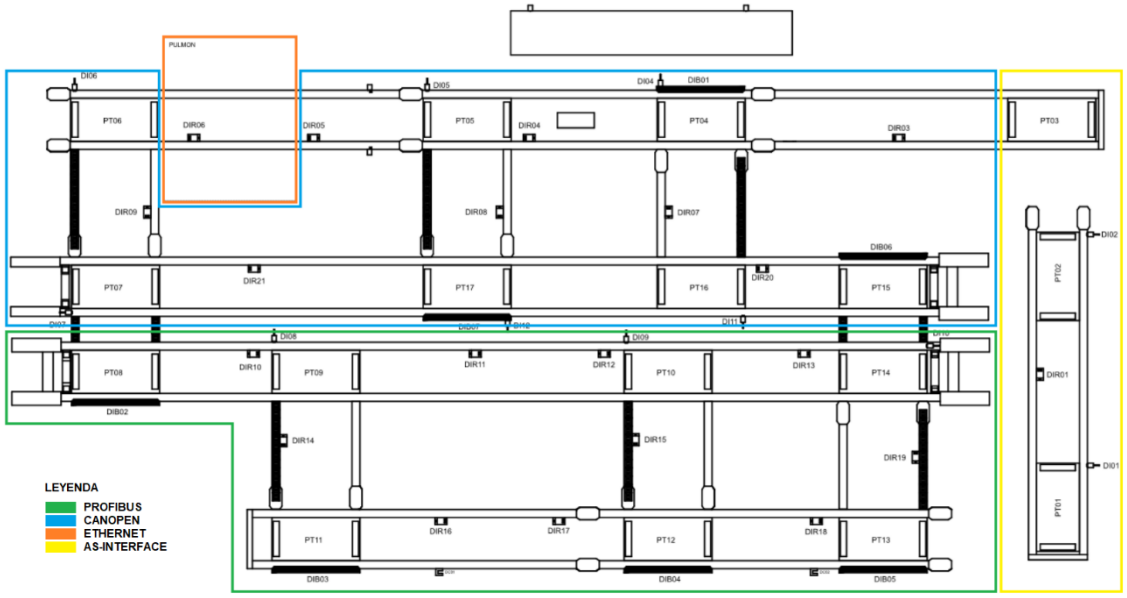

Figura 9: Esquema de la célula industrial a estudio. Fuente: Departamento Ingeniería Electrónica ESEIAAT (UPC)

3.1.1. Funcionamiento de la célula industrial

La celda industrial está diseñada para el transporte de bandejas entre líneas de producción. Este transporte se genera mediante el accionamiento de unos motores eléctricos conectados a unas cintas transportadoras que permiten el movimiento de las bandejas.

Los elementos que conforman la célula, y que se pueden ver indicados en su mayoría en la Figura 9[GLE9], son las cintas transportadoras, las plataformas (PT), los retenedores (DIR), los sensores inductivos (DI) y los basculantes (DIB). Estos elementos serán explicados independientemente en el apartado 3.1.2. Componentes de la célula industrial[GLE10].

Junto a todo este conjunto de equipos, la célula industrial cuenta con un pulmón para el almacenado y abastecimiento de plataformas hacia la célula, una pinza para la extracción de productos desde la célula, y otra pinza para el traspasarse de plataformas desde la línea AS-Interface hacia la línea CANopen.

El recorrido que tienen las bandejas empieza con la introducción manual de éstas desde la plataforma PT08 de la línea Profibus. También puede abastecerse a la célula de plataformas mediante la descarga desde el pulmón. Una vez la bandeja ha sido introducida, ésta sigue el recorrido que marca el programa de los PLC. El estudio y explicación de las trayectorias que pueden seguir las plataformas queda fuera del alcance de este proyecto, por la cual cosa no serán detalladas.

Finalmente, si una plataforma llega a la línea CANopen se seguirán las necesidades de la línea regidas por el programa para poder ser almacenada en el pulmón o continuar el trayecto por la célula.

3.1.2. Componentes de la célula industrial

En el este apartado se explicarán con más detalle los componentes físicos que forman la totalidad de la célula industrial. Estos elementos forman parte del nivel de campo dentro de la pirámide CIM.

I | Motores y cintas transportadoras

Estos elementos se encargan en sintonía del movimiento de las plataformas dentro del circuito de la planta industrial. Los motores están alimentados por una corriente trifásica y son los encargados de generar el movimiento que posteriormente se trasladará a las cintas transportadoras para ejecutar el movimiento lineal a las plataformas.

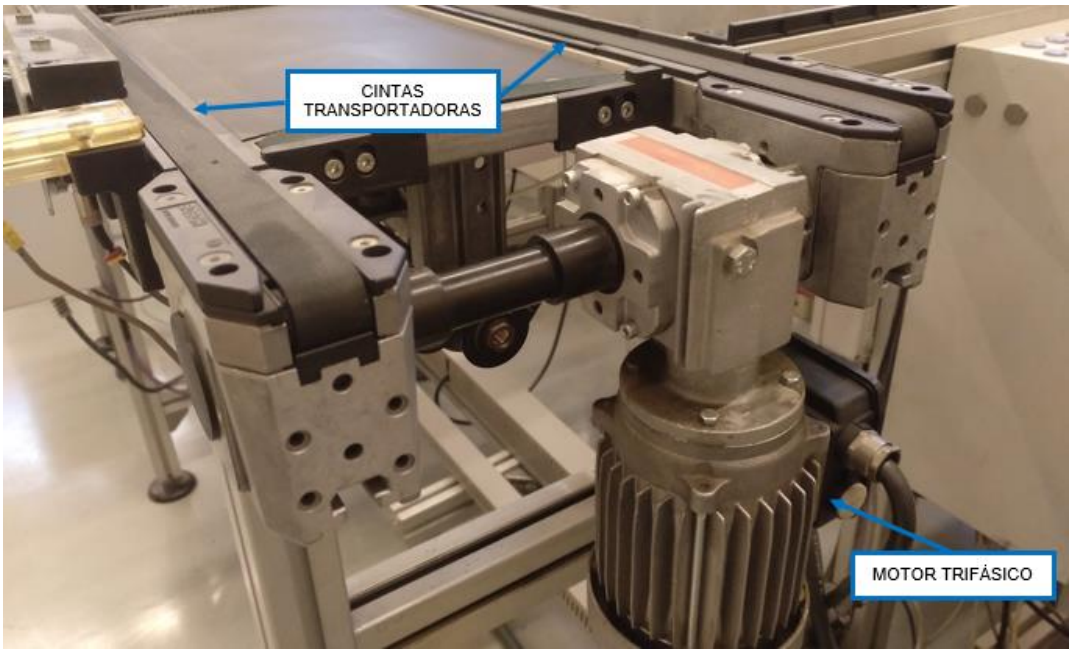

Figura 10: Motor y cintas transportadoras de la célula industrial. Fuente: Elaboración propia

II | Plataformas

Las plataformas son las encargadas de traspasar las bandejas de los materiales entre las diferentes líneas de la célula industrial. En el esquema de la Figura 9[GLE11] se indican estas plataformas con la nomenclatura PT.

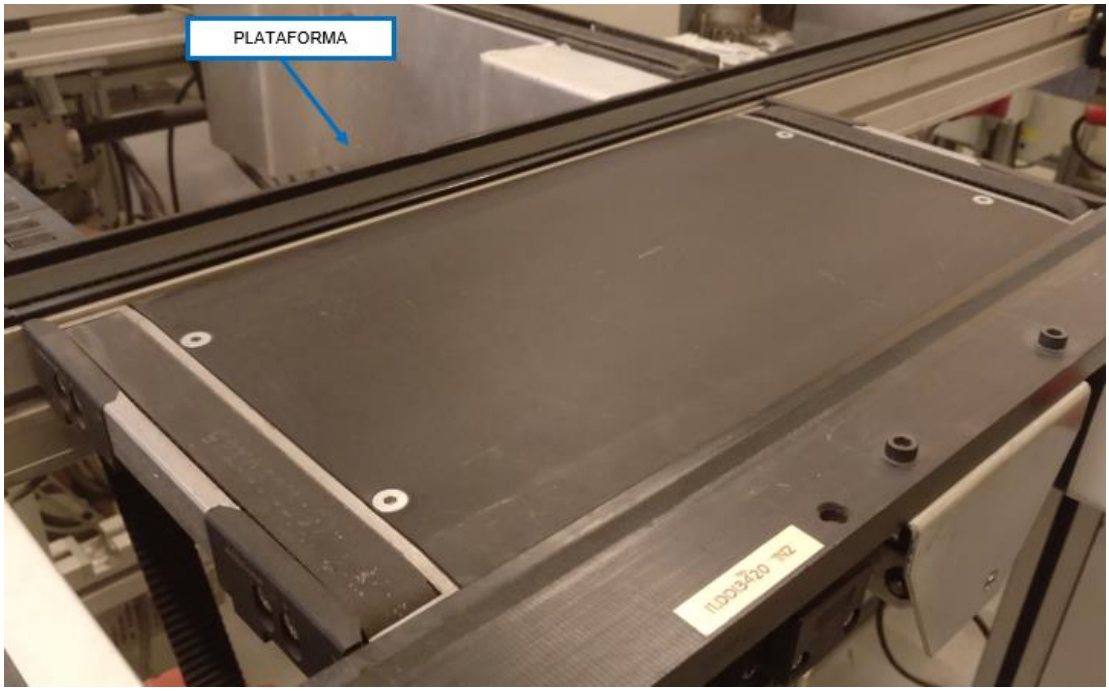

Figura 11: Plataforma de la célula industrial. Fuente: Elaboración propia

III | Sensores inductivos

Los sensores inductivos son capaces de detectar la presencia de elementos metálicos. Éstos están divididos en tres tipos dentro de la célula industrial: los encargados de detectar la presencia de una bandeja en su posición cuando una plataforma está bajada (indicados como DI en el esquema de la célula industrial), los encargados de detectar la presencia de una bandeja encima de una plataforma mediante un elemento basculante (indicados como DIB en el esquema de la célula industrial), y finalmente los encargados de detectar la presencia de una bandeja en su posición y capaces de retenerla entre plataformas según lo que indique el ciclo del programa (indicados como DIR en el esquema de la célula industrial).

A partir de ahora, los DI se nombrarán sensor inductivo, los DIB como basculante, y los DIR como retenedor.

Figura 12: Diferentes sensores capacitivos de la célula industrial. Fuente: Elaboración propia

IV | Setas de emergencia, pulsadores e indicadores

Estos equipos se encargan de dar órdenes al programa de la célula o de informar sobre el estado de ésta. Los pulsadores y setas de emergencia se tratan como entradas a los autómatas programables, mientras que los indicadores, como pueden ser las balizas de señalización o pilotos luminosos, se tratan como salidas de los autómatas para informar del estado de la célula industrial.

Figura 13: Diferentes setas de emergencia, pulsadores e indicadores de la célula industrial. Fuente: Elaboración propia

3.1.3. Arquitectura de la célula industrial

La célula industrial está compuesta por varios autómatas programables que controlan una o varias estaciones con un protocolo de comunicación distinto entre ellas. Cada una de estas secciones está compuesta de islas descentralizadas para cada red de bus de campo. Así pues, se consiguen situar los periféricos de entradas y salidas al lado de cada sistema o estación a controlar, conectados entre sí mediante un mínimo de cables hacia el dispositivo maestro PLC, y utilizando el protocolo de comunicación más adecuado en cada caso.

En la Figura 14[GL12] se puede observar la arquitectura que posee la célula industrial.

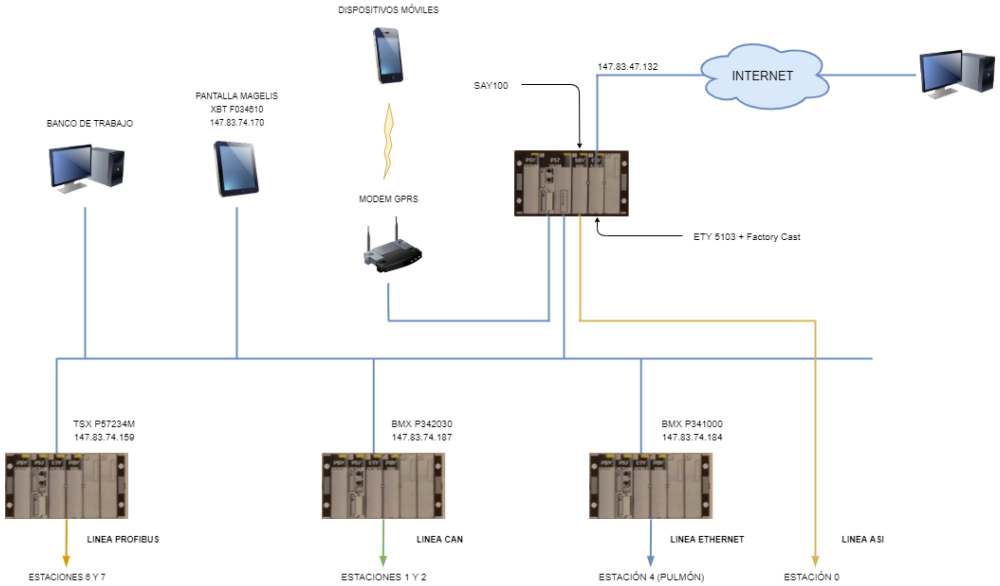

Figura 14: Arquitectura de la célula industrial. Fuente: Elaboración propia

La comunicación de cada estación viene dada por los protocolos que se indican en la siguiente tabla:

Protocolo de comunicación	Estaciones involucradas
Profibus	6 y 7
CanOpen	1 y 2
Ethernet	4 (pulmón)
AS-Interface	0

Tabla 5: Protocolos de comunicación por estación. Fuente: Elaboración propia

Como se observa en la Figura 14[GL13], podemos ver que la comunicación entre los autómatas programables descentralizados hacia el maestro se da a través del protocolo Ethernet.

3.1.4. Estructura de datos de la célula industrial

En este apartado se explicará la estructura de los datos del programa del PLC, usados para la ejecución del proyecto.

Cabe destacar que los únicos datos necesarios y suficientes para este caso de estudio son los datos provenientes de los retenedores (DIR) y de las plataformas (PT). En este caso, ambos equipos poseen la misma estructura a pesar de que no siempre se utilizan todos los datos que contienen. Por otra parte, también se trabajará sobre los datos de un sensor de vibración ubicado en un motor de la célula. Esta estructura de datos será explicada puntos más adelante.

Puede entenderse como estructura de datos a un formato concreto de variable dentro de un ámbito determinado y para un equipo en concreto, que podrá reproducirse de igual manera para todos los equipos del mismo tipo o de la misma función a lo largo del programa. En este caso, la estructura de datos que poseen tanto las plataformas como los retenedores la podemos observar en la siguiente tabla:

Estructura de datos de DIR y PT		**Cantidad de memoria**
Bandeja entrada	ID	2 bytes | 1 Word
	ID Producto	2 bytes | 1 Word
	Tipo de Producto	2 bytes | 1 Word
	Estado del Producto	2 bytes | 1 Word
Bandeja salida 1	ID	2 bytes | 1 Word
	ID Producto	2 bytes | 1 Word
	Tipo de Producto	2 bytes | 1 Word
	Estado del Producto	2 bytes | 1 Word
Bandeja salida 2	ID	2 bytes | 1 Word
	ID Producto	2 bytes | 1 Word
	Tipo de Producto	2 bytes | 1 Word
	Estado del Producto	2 bytes | 1 Word
Estado	-	2 bytes | 1 Word

Tabla 6: Estructura de datos de DIR y PT. Fuente: Elaboración propia

Como podemos ver en la tabla, las estructuras de datos de DIR y PT están formadas a su vez como un conjunto de otras subestructuras (bandeja y estado). El tamaño que tienen los valores de estas

subestructuras es de 16 bits, comúnmente conocido dentro del ámbito de los autómatas programables como Word, o también conocido como *Integer (int).* Este tipo de dato puede representar un número entero y positivo, pudiendo adquirir un total de 2^{16} valores posibles, eso significa un total de 65.536 valores posibles, con un rango de 0 hasta el 65.535.

En los siguientes puntos se explicará brevemente qué funcionalidad y significado tienen todas estas variables dentro del programa del PLC.

I | ID

El ID representa un identificador único para cada bandeja que se encuentra dentro del circuito de la célula industrial. De esta forma se podrá realizar un seguimiento de la posición de la bandeja y ejecutar ordenes sobre ésta misma según las peticiones del programa.

II | ID Producto

El ID Producto representa un identificador único para el producto que se encuentra dentro de una plataforma. De esta forma se podrá saber en cualquier momento qué producto en concreto se encuentra en una bandeja dentro del circuito de la celda industrial.

III | Tipo de Producto

El Tipo de Producto representa qué tipo de producto se encuentra dentro de una bandeja de la celda industrial, siempre y cuando haya más de un tipo de producto, este variará en función del tipo.

IV | Estado

El Estado representa el estado actual del producto en cada momento donde se encuentra dentro del circuito de la célula industrial. Este valor a pesar de ser un valor *integer*, únicamente tiene una representación binaria donde se indica si el producto está acabado o no está acabado.

V | Estado

Este Estado, a diferencia del anterior, representa el estado en el que se encuentra el retenedor o la plataforma, estos pueden adquirir varios valores que dependen de si el equipo está en reposo, o si se le da orden para dejar paso a una bandeja en una salida o en la otra (en caso de tener varias salidas disponibles). Los valores que esta variable tendrá en este proyecto serán los siguientes:

- Estado = 1: Implica que el equipo está en reposo. En este estado las bandejas que se encuentren con el equipo serán entradas hacia el equipo.
- Estado = 2: Implica que el equipo está en petición. En este estado las bandejas que se encuentren con el equipo serán paradas por el mismo.
- Estado = 5: Implica que el equipo está en reposo y la bandeja avanza hacia la salida 1. En este estado a las bandejas que se encuentren en el equipo se les dará avance recto.

- Estado = 9: Implica que el equipo está en reposo y la bandeja avanza hacia la salida 2. En este estado a las bandejas que se encuentren en el equipo se les dará avance de cambio de dirección.

Cabe destacar que no todos los equipos tienen dos tipos de avance, solo algunas plataformas. Estas son las caracterizadas por una intersección entre las líneas, y se considera el Estado = 5 cuando la bandeja que recorre el equipo no tiene que cambiar de dirección, y Estado = 9 cuando la bandeja sí que cambia de dirección.

3.1.5. Magelis Edge Box

El equipo Magelis Edge Box es desarrollado y comercializado por Schneider Electric. Este equipo actúa de pasarela permitiendo la comunicación desde equipos de campo hasta el punto donde se encuentran las aplicaciones software que se requieran en cada caso. Algunos de estos softwares son, por ejemplo, el Azure, Eco Truxure Machine Advisor, InfluxDB, o cualquier tipo de almacenamiento en la nube.

Este equipo permite un diseño de red industrial muy versátil y con una gran cantidad de diseños de hardware y software, que se pueden adaptar a la perfección a las soluciones que se deban aplicar.

3.2. Softwares utilizados

Se conoce como software de red aquellos programas necesarios para relacionar y comunicar varios equipos informáticos. Gracias a ellos se permite la transferencia de información necesaria para el correcto uso de los sistemas. En los siguientes puntos se expondrán los diferentes programas utilizados para la implementación de este proyecto en la célula industrial.

3.2.1. Software Node-RED

Node-RED es una herramienta para el desarrollo de comunicaciones que se basa en el flujo de información a partir de una programación visual. Nos permite la creación de una conexión de varios medios hardware hacia múltiples servicios en línea como parte del IIoT.

Para el desarrollo de este trabajo se usará la versión 1.2.3. Cabe destacar la necesidad de la descarga del entorno de ejecución de JavaScript, NodeJS. Para este caso de estudio se usará la versión recomendada por los desarrolladores, siendo esta la 14.15.0[GLE14].

Una vez se tiene instalado el software en un equipo, Node-RED se ejecuta en el mismo equipo en forma de servidor local al que podrá acceder cualquier dispositivo que se encuentre en la misma red. La IP a través de la cual se puede acceder al software es la IP local 127.0.0.1, a través de su puerto 1880. Para ejecutar el programa se debe ejecutar el comando *node-red* a través de la consola

de comandos del equipo donde esté instalado, y acceder a la dirección IP y puerto anteriormente mencionados a través de un navegador web.

La programación en Node-RED se basa en la unión de nodos a través de un flujo de datos. Se pueden entender estos nodos como un bloque ya predefinido, donde se ubica una programación en JavaScript y está diseñado para facilitar el trabajo al programador gracias a un ahorro considerable en el código de programación en la mayoría de los casos. Estos nodos se dividen en librerías, donde en cada una de ellas se implementa la solución a un problema en concreto que se pueda manejar dentro del entorno del software. Estas librerías se encuentran dentro de la opción de *Manage Palette – Install*, y posteriormente buscar la librería necesaria que se necesite en cada caso. Para este desarrollo, serán necesarias las siguientes librerías:

- node-red-contrib-influxdb: Permitirá la comunicación con la base de datos que se implementará mediante el software InfluxDB.
- node-red-contrib-modbus: Permitirá establecer una comunicación mediante protocolo Modbus TCP hacia equipos. [GL15]

La lógica de Node-RED es el establecimiento de un flujo de programa que va pasando de nodo a nodo mediante la unión de éstos. Esta unión simboliza el traspase de información de un nodo a otro, donde la propiedad que se irá cambiando en función de las necesidades será el mensaje (msg).

Este mensaje contiene una cantidad de información dividida en variables, donde por defecto se encuentran las variables msg.payload y msg.topic. Estas variables no son únicas ni excluyentes, por lo que se pueden añadir o borrar otras en función de lo que se necesite. Existen tres tipos de variables en Node-RED en función de su disponibilidad dentro del programa, las de contexto Node, de Flow y de Global. Las variables de contexto Node son las que se encuentran más encapsuladas y únicamente serán visibles y accesibles desde el mismo nodo donde sean declaradas. Las variables de contexto Flow darán acceso a ellas desde un mismo flujo de programa. Y por último las variables de contexto Global a las que pueden acceder todos los flujos de programa y todos los nodos que haya dentro de ellos.

3.2.2. Software InfluxDB

InfluxDB es considerado un gestor para bases de datos de series temporales. Esto se entiende como un almacenamiento de datos organizados a lo largo del tiempo. Este gestor es comúnmente utilizado para el monitoreo de operaciones, análisis en tiempo real de sistemas y almacenamiento de datos para equipos utilizados en el Internet de las cosas. Para este proyecto se usará la versión 1.8.3[GLE16].

Las bases de datos que se suelen usar para almacenar y evaluar datos de sensores o protocolos con marcas temporales durante un rango de tiempo determinado. Dentro de este software, las bases de datos son muy compactas y se componen por valores almacenados en un instante de tiempo determinado y en columnas.

Para el desarrollo de este proyecto, se utilizarán los comandos listados a continuación dentro de la consola de comandos de InfluxDB:

- CREATE DATABASE "nombre_database": Creará una base de datos con el nombre que le indiquemos, que almacenará sus valores dentro del mismo equipo donde esté ejecutándose el programa.
- SHOW DATABASES: Mostrará las bases de datos que se hayan creado.
- DROP DATABASE "nombre_database": Eliminará la base de datos y todo su contenido de la memoria del equipo donde esté ejecutándose el programa.
- USE DATABASE "nombre_database": Seleccionará la base de datos correspondiente al nombre que se le indique, para poder ver o modificar los valores que contenga.
- SHOW MEASUREMENTS: Comando condicionado a la previa ejecución de USE DATABASE. Mostrará las medidas que se hayan ido almacenando dentro de la base de datos que se haya seleccionado.
- SELECT * FROM "nombre_measurement": Permite mostrar la serie de tiempo que se haya almacenado dentro de la base de datos seleccionada, junto a los valores correspondientes para cada instante de tiempo donde se ha recogido el valor mostrado.

Para el desarrollo de esta aplicación, se usará la base de datos llamada "TFG_DB", donde se almacenarán las medidas (*measurements),* que se vayan recogiendo del sistema. Las medidas pueden entenderse como variables independientes de series de datos que se almacenarán dentro de la base de datos.

3.2.3. Software Grafana

Grafana es una aplicación que permite la visualización de datos de series temporales. A partir de un conjunto de mediciones recolectadas a lo largo del tiempo se obtiene una visión gráfica de la situación concreta de un sistema[GLE17]. Esto lo hace una herramienta perfecta para la visualización de datos que serán almacenados dentro de la base de datos de InfluxDB.

Tal y como ocurre en Node-RED, Grafana se ejecuta en el mismo equipo en forma de servidor local. La IP a través de la cual se puede acceder al software es la IP local 127.0.0.1, a través de su puerto 3000. Cabe destacar que, por defecto, para acceder a Grafana se necesita acceder mediante un usuario y contraseña. Estos serán:

- Usuario: admin
- Contraseña: admin

Esta contraseña para el acceso a la aplicación deberá ser cambiada la primera vez que se acceda al software.

Una vez se ejecuta ya la aplicación a través de un buscador web, se debe añadir una fuente de datos para poder ser posteriormente visualizados. Por defecto, este programa soporta una fuente de datos, pero mediante *plugins* se pueden recopilar desde varias.

El siguiente paso es la generación de los conocidos como *dashboards*. Estos serán las pantallas compuestas por paneles que permitirán la visualización de los datos que se recojan desde la base de datos. Estos paneles tienen gran cantidad de formas de representación de los datos que nos interese, yendo desde los gráficos lineales o valores numéricos hasta tablas y listas.

Cada uno de estos paneles deberá ser configurado para indicarle la fuente de datos de la cual se requiere la visualización de los datos, y la medida que queremos representar. Estos datos tienen multitud de forma de representarse y agruparse, pudiendo ser por intervalo de tiempo, por formato, o también por ejemplo por valor del propio dato.

Este programa permite una gran versatilidad para la representación de los datos y de una forma muy intuitiva, permitiendo fácilmente generar la representación de datos sencillos hasta representaciones mucho más complejas que posiblemente requieran de *plugins* para su correcta visualización.

Uno de los *plugins* que se deberá instalar es el *FlowCharting*. Este adicional permite ver y utilizar diferentes graphicos de librerías *Draw.io*. Teniendo en cuenta que este formato de diagrama es con el que se ha realizado el diagrama de la célula industria, por primera vez mostrado en la Figura 9, este añadido permitirá insertar el diagrama pudiendo tratar independientemente cada elemento para así ver modificaciones en ellos.

4. Método experimental

4.1. Simulación del sistema

Para simular el sistema de la célula industrial explicado en puntos anteriores, se generará una tabla en formato CSV para poder simular el recorrido de varias bandejas a través de la célula industrial. Para ello, se tendrán en cuenta las estructuras de datos que tienen los retenedores y plataformas, explicados en el apartado 3.1.4 Estructura de datos de la célula industrial, y los valores que irán variando dentro de esta estructura en función del avance de las bandejas a través de las líneas de la célula industrial.

Este archivo de simulación estará anexado al presente documento como ANEXO II: SIMULACIÓN RETENEDORES DE LA CÉLULA.

Este archivo simulará los datos de las 17 plataformas y 20 retenedores que forman el conjunto de la célula industrial. El orden que seguirán estos componentes va siguiendo la lógica del circuito que puede realizar una bandeja a través del sistema, tal y como se muestra en la Figura 15.[GL18]

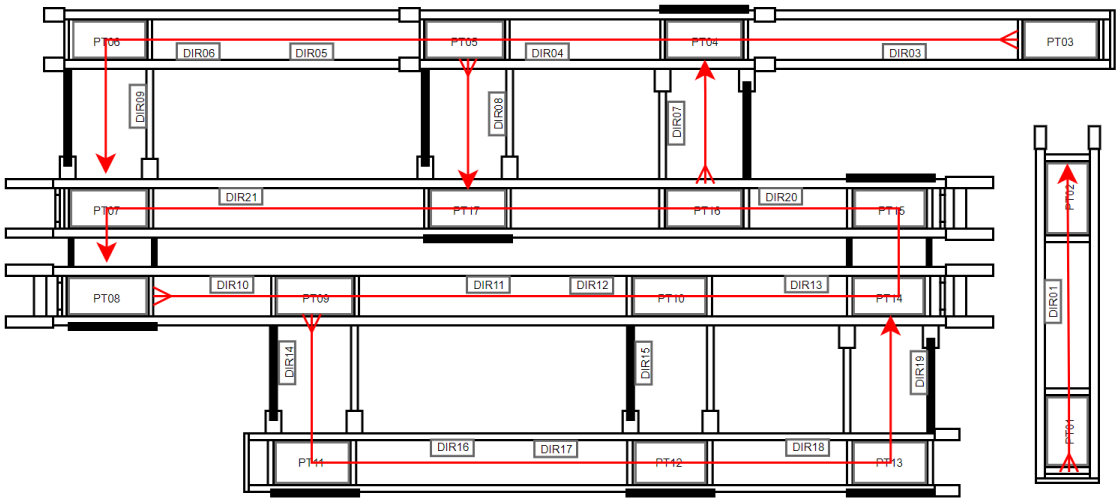

Figura 15: Posibles recorridos de una bandeja dentro de la célula industrial. Fuente: Elaboración propia

Así pues, y teniendo en cuenta la estructura de datos de las plataformas y retenedores dentro del programa del PLC explicada anteriormente en el apartado de la estructura de los datos[GL19], el archivo de simulación tendrá un total de 962 bytes con el que simularán estos componentes, estando formado cada uno de ellos por un total de 13 bytes. La ocupación de esta memoria, por simplicidad y poca implicación dentro de este estudio, ocupará desde el MW0 hasta el MW960.

Una vez conocida la disposición de los datos de cada componente dentro del archivo CSV, conviene explicar cómo se simulará el progreso de una bandeja dentro de la celda industrial. Este progreso irá marcado por el cambio de una fila a otra, en orden descendiente, y empezando desde la primera

petición. Se pueden entender estas peticiones como cada cambio importante generado dentro del sistema, donde se informará de los valores que cambian en cada plataforma y retenedor que se ve involucrado en estos cambios.

Para esta simulación, se ha seleccionado el recorrido de 2 bandejas totalmente independientes, donde cada una seguirá el recorrido mostrado en las siguientes tablas:

Bandeja 1	
Petición	**Ubicación de la bandeja**
1 / 2	Plataforma 3
3 / 4	Retenedor 3
5 / 6	Plataforma 4
7 / 8	Retenedor 4
9 / 10	Plataforma 5
11 / 12	Retenedor 5
13 / 14	Retenedor 6
15 / 16	Plataforma 6
17 / 18	Retenedor 9
19 / 20	Plataforma 7
21 / 22	Plataforma 8
23 / 24	Retenedor 10
25 / 26	Plataforma 9
27 / 28	Retenedor 14
29 / 30	Plataforma 11
31 / 32	Retenedor 16
33 / 34	Retenedor 17
35 / 36	Plataforma 12
37 / 38	Retenedor 18
39 / 40	Plataforma 13

Tabla 7: Recorrido de la Bandeja 1. Fuente: Elaboración propia

Bandeja 2	
Petición	**Ubicación de la bandeja**
15 / 16	Plataforma 3
17 / 18	Retenedor 3
19 / 20	Plataforma 4
21 / 22	Retenedor 4
23 / 24	Plataforma 5
25 / 26	Retenedor 8
27 / 28	Plataforma 17
29 / 30	Retenedor 21
31 / 32	Plataforma 7
33 / 34	Plataforma 8
35 / 36	Retenedor 10
37 / 38	Plataforma 9
39 / 40	Retenedor 11
41 / 42	Retenedor 12
43 / 44	Plataforma 10
45 / 46	Retenedor 13
47 / 48	Plataforma 14

Tabla 8: Recorrido de la Bandeja 2. Fuente: Elaboración propia

Como puede observarse, en cada posición donde se controla la bandeja existen 2 peticiones diferentes, eso es debido a que a primera petición que se genera en una posición se le transfiere la información de la bandeja entrante en ese punto, y la petición consecutiva a esta transfiere la información a la salida donde irá la bandeja. Esto permite ver la transición y los diferentes estados del sistema mientras una bandeja lo recorre y permitirá una mayor claridad a la hora de su visualización.

Para una mayor claridad, en la siguiente imagen se muestran los recorridos simulados indicados en las tablas anteriores. En esta imagen, el circuito que se genera a través de las indicaciones azules corresponden a la Bandeja 1, y las indicaciones de color anaranjado corresponden a la Bandeja 2.

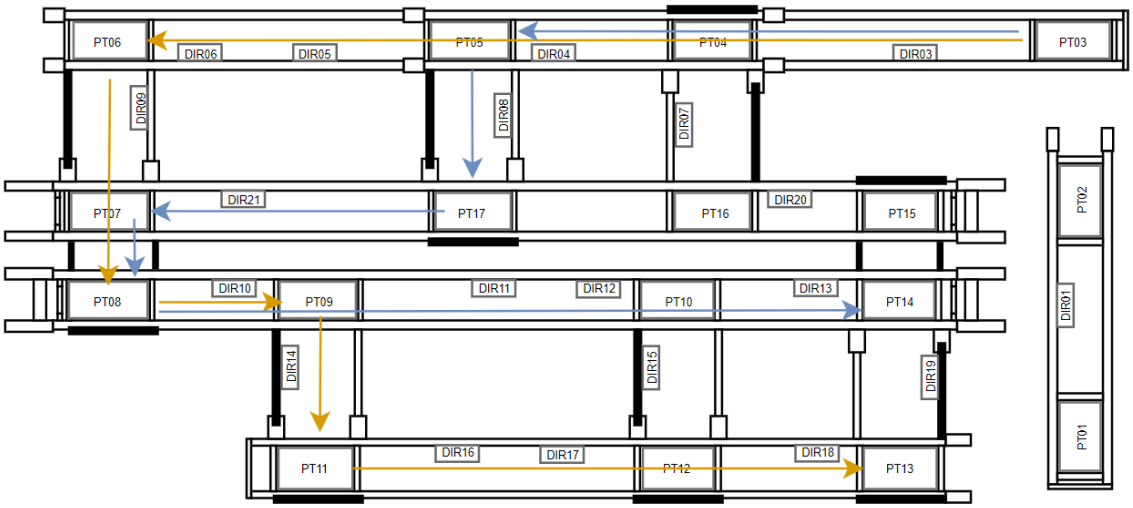

Figura 16: Recorrido de la Bandejas 1 y Bandeja 2 dentro de la célula industrial. Fuente: Elaboración propia

4.2. Estados de plataformas y retenedores

En este apartado se pretenderá explicar la metodología seguida para la adquisición de los datos de la célula industrial provenientes de las plataformas y de los retenedores, el almacenaje de los mismos en una base de datos, y su posterior visualización en un diagrama de la célula dentro de Grafana.

Este paso permitirá visualizar exactamente el recorrido que sigue una bandeja, simulando así el funcionamiento básico que realiza un SCADA dentro de un entorno industrial. Este seguimiento del recorrido se visualizará mediante el cambio de color en los elementos implicados, que indicará la posición donde se encuentre una de las bandejas dentro del sistema.

4.2.1. Adquisición de datos con Node-RED

Para llevar a cabo la implementación de esta arquitectura será necesario el software Node-RED. Éste permitirá la adquisición de los datos de la célula industrial.

Para este paso se usarán los siguientes bloques:

I | Inject

Este bloque permite inyectar un mensaje en un flujo manualmente o mediante intervalos regulares que pueden ser configurados. Este mensaje puede ser de varios tipos, pasando por ejemplo por una cadena de caracteres, un booleano o una expresión dada en JavaScript.

Para esta implementación se generará un mensaje booleano con una señal de *True*, y con una repetición en el tiempo en un intervalo configurable de tiempo. Esta parte del programa hará posible la lectura de todos los valores del Excel de simulación de la célula con un único flanco.

II | File in

Este nodo permite leer el contenido de un fichero como valores de string o *binary buffer*. Para esta aplicación los valores serán leídos como cadena de caracteres (string). Para configurar este nodo es necesario indicarle el directorio de nuestro archivo CSV a leer e indicar el formato de salida que tendrá el archivo y la codificación que tendrá:

- Output: a single utf8 string
- Encoding: utf8

III | CSV

Este nodo convierte el formato de un archivo de tipo CSV a objeto JavaScript en ambas direcciones. Este nodo debe ser configurado indicando el tipo de separación que tendrá el archivo CSV, la configuración de entrada para evitar o incluir valores del archivo, y el formato de salida que tendrá:

- Separator: semicolon
- Output: a message per row

Con esta configuración se le está indicando al programa que la separación de los valores que contiene el archivo CSV es a través de punto y coma, y que tiene que mostrar la salida separando el mensaje línea por línea.

IV | Function

Este nodo permite aplicar una función de JavaScript al flujo de datos que se recibe en él. El mensaje se pasa como un objeto de JavaScript llamado msg, y puede estar compuesto desde objetos, hasta *arrays* o no devolver nada para forzar la detención de un flujo de datos. Esta función será utilizada para adquirir una petición en concreto en cada iteración del programa, simulando así de una forma mayor un funcionamiento real de la célula industrial.

V | InfluxDB out

Nodo que permite la escritura de valores y tags hacia una medida de InfluxDB. Para la configuración de este nodo se necesita indicar el nombre de la base de datos a la que se apuntará, su dirección IP y puerto, y el nombre de la medida que se quiere almacenar en ella.

VI | Debug

Este nodo muestra las propiedades del mensaje. Con esto se permite ver el mensaje en un punto determinado del flujo para poder examinar qué datos tenemos en un punto determinado.

Una vez realizada una breve explicación sobre los nodos que van a usarse en esta parte de la implementación, a continuación se muestra la imagen del programa en conjunto con estos nodos y un diagrama de flujo para el correcto entendimiento de la secuencia del programa:

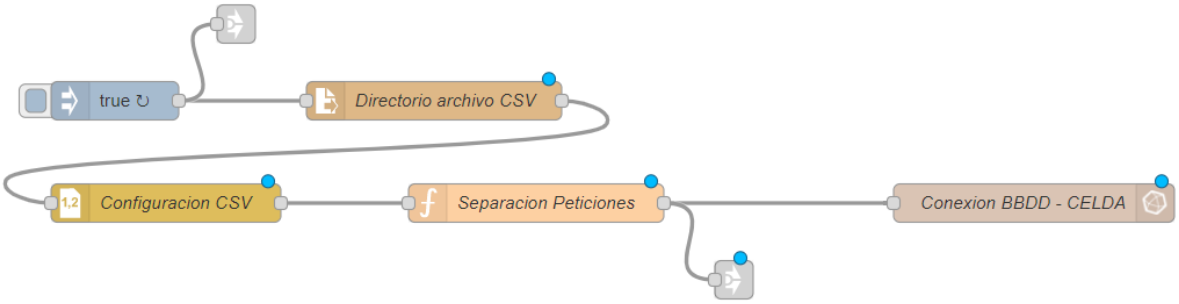

Figura 17: Código para la adquisición de datos CSV en Node-RED. Fuente: Elaboración propia

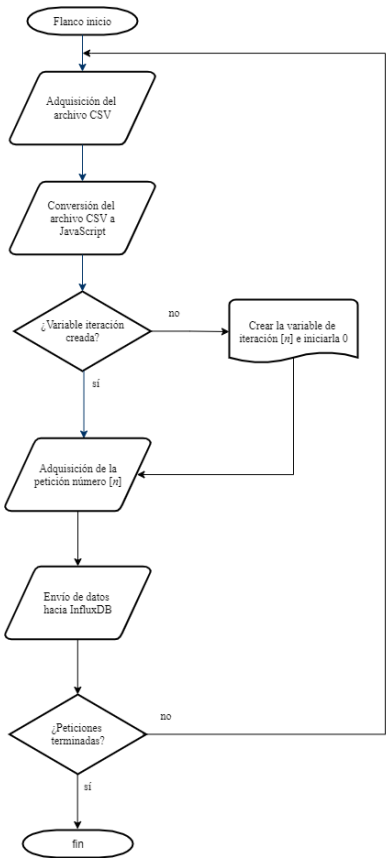

Figura 18: Diagrama de flujo programa Node-RED. Fuente: Elaboración propia

4.2.2. Almacenamiento de datos con InfluxDB

Para almacenar todos los datos provenientes de la adquisición del Node-RED, éstos se almacenarán dentro de un mismo *measurement*, en el cual se ubicarán diferentes *fields* dentro de el mismo *measurement*. Las *fields* son subconjuntos de datos dentro de un *measurement*, que pueden entenderse como una matriz de datos la cual tiene una etiqueta para la identificación de todos ellos.

En este caso, los datos almacenados lo harán en el siguiente *measurement:*

- TFG_DB\MEDIDAS

4.2.3. Visualización del sistema con Grafana

Como ya se ha comentado en apartados anteriores, para la visualización de este sistema se utilizará el *plugin FlowCharting*. Los detalles necesarios para la instalación de este *plugin* no serán explicados en la presente memoria ya que carecen del suficiente interés para la explicación del desarrollo de este proyecto.

El diagrama de la célula se ha generado mediante un archivo .drawio, el cual es compatible con este *plugin* mediante una fácil exportación del código de este hacia Grafana. Este contenido del diagrama será copiado hacia la propiedad de Flowchart/Source Content, obteniendo así un panel en el que en su interior se encontrará el diagrama de la célula.

Una vez se ha importado el diagrama al *dashboard* de Grafana, se procederá a su configuración para poder ver cambios en los estados de las plataformas y los retenedores. La finalidad de estos pasos es conseguir una localización en tiempo real de una bandeja dentro de la célula, la cual se mostrará mediante una forma que cambiará de color en función de si hay una bandeja en su posición o no. La lógica que seguirá el panel para mostrar la ubicación de una bandeja es la siguiente:

Color de la forma	Significado
Naranja	Bandeja no presente
Verde	Bandeja presente

Tabla 9: Lógica para la visualización de una bandeja en un elemento de la celda. Fuente: Elaboración propia

El *dashboard* final se muestra en la siguiente figura, seguido de los puntos necesarios para explicar el procedimiento seguido.

Figura 19: Dashboard de la vista global de la célula industrial. Fuente: Elaboración propia

I | Configuración de las formas

A la hora de importar un diagrama hacia Grafana, este programa adquiere también el nombre identificador de cada forma que se halla dentro del mismo. En este caso, las formas que se usarán son las formas circulares que pueden verse en la Figura 19[GL20], las cuales cambiarán de color siguiendo la lógica de la Tabla 9[GL21].

Por ejemplo, la forma presente en la plataforma 17 (PT17), tendrá el nombre identificador número 37. Esto se puede examinar en el archivo fuente .drawio, viendo las propiedades de datos del elemento que interese.

II | Configuración de los datos de interés

Para saber si hay una bandeja presente o no encima de un elemento, se usará la columna ID Producto del objeto Bandeja de Entrada. Para más información acerca de este dato, consultar la Tabla 6.[GL22] Este dato está configurado en el archivo de simulación para que siempre que haya un objeto en un elemento, se indique un dígito igual o mayor que 0. En caso de no haber ningún elemento presente, éste se volverá al valor -1.

Para la configuración de recogida de este dato, se deben generar peticiones para cada una de ellos llamadas *query* dentro del software Grafana. Esta *query* creará una petición para un elemento indicado dentro de la base de datos. Siguiendo el ejemplo de la PT17, el valor que nos interesa se ubica en la columna 458 dentro del *measurement* MEDIDAS de la base de datos TFG_DB. Se configurará esta petición para que se recoja este valor tal y como se muestra en la siguiente imagen:

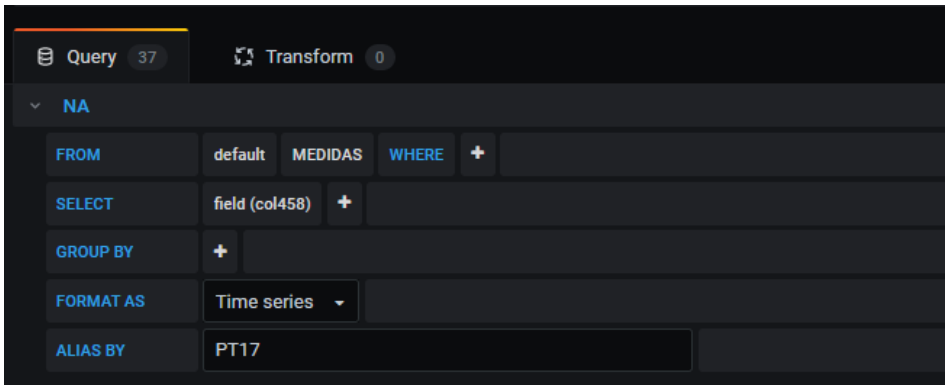

Figura 20: Configuración de la petición para la recogida de un dato. Fuente: Elaboración propia

Sintetizando estos elementos, en esta *query* se genera una adquisición del dato proveniente del *measurement* MEDIDAS, seleccionando la *field* con nombre col458, y sin ningún tipo de cálculo ni agrupación debido a que únicamente interesa el último valor que se ha recogido. El alias condicionará el siguiente punto que se indicará a continuación, y sirve para la identificación del propio *query*.

III | Configuración de las reglas

Se entienden como reglas a las condiciones que tienen que darse del dato recogido, para que la forma que tiene que indicar la posesión o no de una bandeja cambie su diseño para que pueda entenderse el cambio de estado de los elementos. En este caso, se generará una regla para cada elemento que nos interesa, siendo estos todas las bandejas y retenedores.

Siguiendo de nuevo el ejemplo de la PT17, se configurará esta regla teniendo en cuenta los valores explicados en los anteriores dos puntos. En la siguiente figura se puede observar la configuración de la regla utilizada.

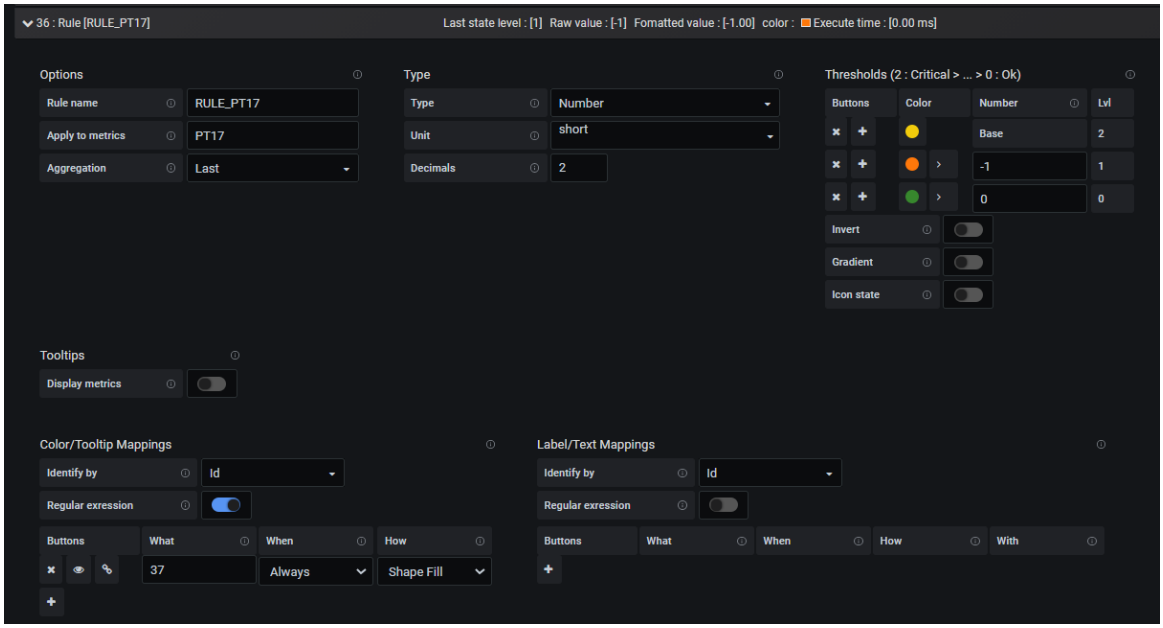

Figura 21: Configuración de la regla para la visualización del estado. Fuente: Elaboración propia

Esta esta regla, se necesitará configurar varios elementos dentro de las pestañas *Options*, *Thresholds* y *Color*. Con estas se conseguirá el cambio de color que se requiere en este apartado.

La pestaña *Options* se configura añadiendo un nombre a la regla, e indicando que *query* debe aplicarse para esta regla. Sintetizando, el valor que adquirirá esta regla será la que corresponda con la columna col458, ya que es la que se ha configurado dentro de la *query* PT17. La configuración de *Aggregation* permite configurar que el dato de interés sea el último que tiene esta variable.

En el apartado *Thresholds*, se configuran las posibles opciones de valores que se quieren evaluar, y los colores que se desean en las mismas opciones. Cabe destacar que este apartado sigue las reglas de un sistema condicional, y que se aplicará una u otra siempre y cuando el valor de la columna col458 sea igual o superior que el valor que se configure en este apartado. Tal y como se muestra, se ha configurado esta parte para que el color sea naranja en caso de que haya un valor -1 (bandeja no presente), y que el color se vuelva verde en caso de que haya un valor 0 o superior (bandeja presente).

Por último, el apartado *Color* permite aplicar esta regla a una forma determinada dentro de nuestro panel, y en unas condiciones indicadas. En este caso, se desea identificar el elemento por su ID (correspondiente al nombre indicador explicado puntos más arriba), y que esta regla sea constante. En los siguientes apartados se configura el elemento deseado, en este caso el 37, haciendo que la regla se aplique siempre y que el cambio que se observe en el elemento sea el color de relleno de la forma.

En la siguiente imagen, puede verse el cambio de estado del retenedor DIR06, que se torna de color verde para indicar la presencia de una bandeja en esa posición.

Figura 22: Cambio de estado en el panel del DIR06. Fuente: Elaboración propia

Se puede comprobar que este cambio sí que corresponde con el estado actual del retenedor DIR06, examinando el *query* que se ha realizado en este momento. Eso puede hacerse a través de *Query Inspector*, dentro de la edición del propio panel. Ahí se puede observar que en la última adquisición el valor de esta *query* es de 1000, indicando así la presencia de una bandeja encima de él.

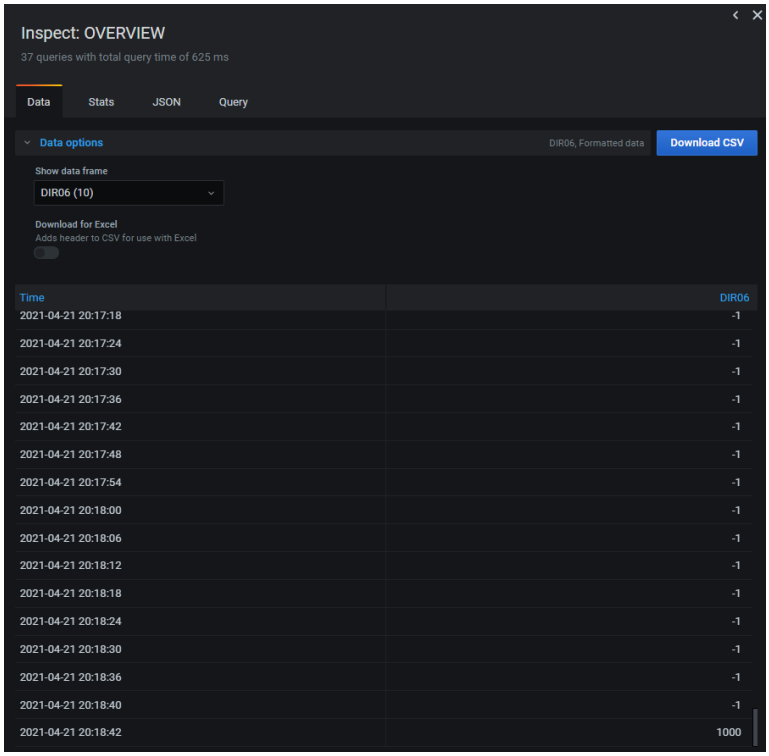

Figura 23. Inspección del *query* para la comprobación del estado de DIR06. Fuente: Elaboración propia

4.3. Sensor de vibración

En este apartado práctico se mostrará el procedimiento usado para la adquisición de datos de un sensor de vibración ubicado en un motor de la célula industrial del laboratorio, para su posterior almacenamiento en la base de datos y visualización en Grafana. A diferencia de la adquisición de datos de las plataformas y retenedores de la célula, anteriormente explicado, este apartado no simulará ningún valor y los datos recogidos serán a tiempo real.

Este sensor realiza cuatro cálculos basados en medidas calculadas a través de la vibración:

I | RMS

El RMS indica la longitud desplazada por segundos de vibración de un objeto. Su unidad son los mm/s, y se compone de una forma de onda sinusoidal a pesar de que el valor representado es a través del valor cuadrático medio.

II | Skewness

El Skewness representa la uniformidad de la distribución de los datos respecto la media aritmética. Es determinado a partir del cociente entre el tercer momento estadístico y el cubo de la raíz del segundo momento estadístico.

III | Kurtosis

También conocido como coeficiente de apuntamiento, Kurtosis permite conocer el grado de concentración de los datos con respecto a la media. Tener una mayor Kurtosis significa una mayor concentración de datos muy cerca de la media de la distribución, mientras que un valor pequeño denota una elevada frecuencia de datos alejados de la misma media.

IV | Mean

Se conoce como el valor promedio de una serie de datos, calculándose como el sumatorio de estos valores dividido por la cantidad de los mismos.

V | Indicator

Indicator es un valor que no devuelve el sensor de vibración y que será calculado a través del valor cuadrado de RMS

Para adquirir estos datos, será necesario implementar una comunicación vía Modbus para acceder al sensor. Éste tiene actualmente una IP pública 147.83.83.29 a través del puerto 20000.

4.3.1. Adquisición de datos con Node-RED

Como se ha comentado anteriormente, para adquirir los datos del sensor de vibración se gestionará una comunicación a través de un nodo Modbus. A continuación se detallarán los nodos utilizados para este fin.

I | Modbus Flex Getter

Permite gestionar una conexión Modbus TCP a partir de un nodo de entrada con unos parámetros de configuración. Estos parámetros de entrada, explicados en el siguiente punto, serán los que darán la indicación de qué datos debe el nodo coger dentro de la memoria del PLC.

II | Function

Esta función le dará al nodo de comunicación Modbus la dirección exacta y los valores que debe coger del PLC. El código usado para esta función se indica en la siguiente figura:

```
msg.payload = {
    value: msg.payload,
    'fc': 3,
    'unitid': 0,
    'adress':0,
    'quantity':4
}
return msg;
```

Figura 24: Código para solicitud de datos del sensor de vibración en el PLC. Fuente: Elaboración propia

Una vez se reciben los datos del sensor de vibración, éstos se encuentran en un formato de matriz y deben ser procesados para su posterior inducción a la base de datos. Para ello, se generan cinco funciones diferentes que procesaran cada dato para ponerlos finalmente en el formato correcto dentro de la base de datos.

III | InfluxDB out

Este nodo, explicado ya en puntos anteriores, será el en cargado de, individualmente, insertar cada valor enviado por el PLC y con su posterior procesado, dentro de la base de datos. Cada una de estas medidas estará almacenada como un *measurement* dentro de la base de datos.

4.3.2. Almacenamiento de datos con InfluxDB

Como ya se ha comentado, en este proyecto habrá únicamente una misma base de datos donde se almacenarán distintas medidas. En este caso, se añadirán los valores explicados en apartados anteriores referentes al sensor de vibración en misma base de datos y en *measurements* independientes, los cuales se llamarán:

- TFG_DB\RMS
- TFG_DB\SKEWNESS
- TFG_DB\KURTOSIS
- TFG_DB\MEAN
- TFG_DB\INDICATOR

4.3.3. Visualización del sistema con Grafana

Para la visualización en Grafana se usará un *dashboard* específico para el sensor de vibración que se llamará VIBRATION SENSOR, y se ubicará dentro de la carpeta TFG. Este *dashboard* estará compuesto por 5 valores correspondientes a los datos enviados y calculados por Node-RED del sensor de vibración a tiempo real, y 5 gráficas de los mismos valores que mostrarán. En la siguiente figura se muestra el panel generado con cada señal y sus valores de los últimos 15 minutos.

Figura 25: Dashboard de sensor de vibración. Fuente: Elaboración propia

4.4. Cálculos de datos de interés

Como se ha ido comentando a lo largo de esta memoria, una parte interesante de la monitorización de datos son las conclusiones que se pueden obtener a través de éstos. Es por ello, que en este apartado se expondrá el procedimiento seguido para obtener distintos valores, siguiendo el mismo hilo de procedimiento (Node-RED - InfluxDB - Grafana) que el resto de los valores de la planta.

4.4.1. OEE

El *Overall Equipment Effectiviness* (OEE por sus siglas en inglés), se trata de un cálculo que indica el aprovechamiento integral de una máquina o de un sistema industrial. Sintetizando, se trata de un porcentaje de ocupación a lo largo del tiempo de todos los equipos que hay disponibles.

Llevándolo dentro del alcance de la célula industrial, se entiende como OEE al porcentaje de tiempo que se tienen retenedores y plataformas en uso. Este dato permite ver una estadística más real del uso que se le está dando a los equipos, y permite detectar posibles vías de mejora dentro de la planta.

En este caso, se realizará el cálculo de un OEE global dentro de la célula industrial, entendiendo esto como el porcentaje del tiempo de ocupación de las plataformas y retenedores dentro de unos recorridos específicos, marcados por la simulación explicada en el apartado 4.1 Simulación del sistema[GL23].

Otra opción disponible sería el cálculo de un OEE por cada equipo dentro de la célula industrial, dando un valor individual que permitiría prever posibles malfuncionamientos o mantenimientos de algunos equipos a causa de un mayor uso de estos en comparación con otros.

Para realizar este cálculo de ocupación dentro de la célula industrial, se partirá del Excel de simulación donde se verán los valores ESTADO de cada retenedor y plataforma, donde se seguirá la siguiente lógica para ver si ocupación o disponibilidad:

Estado	Significado
= 1	Si el estado es igual a 1, se considera que el equipo no está en uso
!= 1	Si el estado es diferente a 1, se considera que el equipo sí está en uso

Tabla 10: Lógica para el cálculo de OEE. Fuente: Elaboración propia

Estos valores están almacenados dentro del decimotercer word de cada equipo. Esto puede verse explicado en el apartado 3.1.4 Estructura de datos de la célula industrial[GL24].

En los siguientes puntos se hará una explicación de los pasos seguidos hasta llegar a la visualización de los datos dentro del software Grafana.

I | Node-RED

El flujo para conseguir este valor se inicia a partir del nodo Object to Array, que se trata de una función con un pequeño código que transforma una variable de tipo objeto a una matriz. Posterior a esto, se inicia otro bloque de función que realizará ya el cálculo del valor de OEE para la célula industrial.

Para este cálculo, se declaran tres variables de tipo Flow, que se inicializan a valor 0 la primera vez que se ejecuta el programa, simulando así el primer instante de medida. Estas variables serán las siguientes:

- OEE: Variable que almacenará en memoria el valor constante en porcentaje del OEE.
- Ocupado: Esta variable almacenará en memoria la cantidad de veces en las peticiones que se ha encontrado una plataforma o un retenedor sin ocupación.
- Libre: Esta variable almacenará en memoria la cantidad de veces en las peticiones que se ha encontrado una plataforma o un retenedor con ocupación.

A partir de este punto, se tienen en cuenta la cantidad de datos totales que forman las plataformas y retenedores dentro del programa simulado de PLC. Como ya se comentó en el apartado 4.1 Simulación del sistema, las direcciones ocupadas van desde la MW0 hasta la MW961. Esto da en total de 481 columnas dentro del archivo de Excel, de las cuales nos interesa la columna múltiple de 13 para realizar el cálculo con el valor de ESTADO de cada equipo.

Se genera una iteración dentro de este vector de datos para almacenar dentro de las variables *libre* y *ocupado* si ese equipo en concreto se encuentra en uso o no. Una vez se finaliza toda la iteración a lo largo de la matriz, se realiza el cálculo del OEE realizando un porcentaje de la cantidad de ocupados que hay en referencia a la suma de los ocupados y libres. Este resultado se fija para que únicamente muestre dos decimales y se transmite el mensaje aguas abajo dirección al InfluxDB. Este mensaje también incluirá las variables de *ocupado* y *libre*, las cuales se procesan posteriormente a esta función para ser enviadas hacia InfluxDB.

En las siguientes imágenes se muestra el flujo implicado en toda esta explicación junto a la parte de código dentro del nodo Calculo OEE.

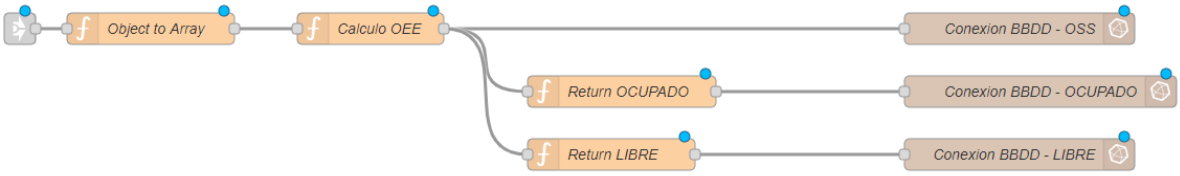

Figura 26: Flujo para el cálculo OEE. Fuente: Elaboración propia

```
    // inicializar OEE, ocupado y libre a 0
if (flow.get("OEE") === undefined) {
    flow.set("OEE", 0)
}
if (flow.get("ocupado") === undefined) {
    flow.set("ocupado", 0)
}
if (flow.get("libre") === undefined) {
    flow.set("libre", 0)
}

var col = 13;
    // iteraciones dentro del vector de datos
while (col<=482) {
    if (msg.payload[col] == 1) {
        flow.set("libre", flow.get("libre")+1);
    } else {
        flow.set("ocupado", flow.get("ocupado")+1);
    }
    col = col + 13;
}

    // cálculo y seguimiento del flujo
var s_OEE = (flow.get("ocupado")/(flow.get("libre")+flow.get("ocupado")))*100;
s_OEE = s_OEE.toFixed(2);
flow.set("OEE", s_OOS);
msg.payload = flow.get("OEE");

return msg;
```

Figura 27: Código para el cálculo de OEE. Fuente: Elaboración propia

II | InfluxDB

El almacenamiento de los datos se realizará en la base de datos TFG_DB, dentro de los siguientes *measurements*:

- TFG_DB\OEE
- TFG_DB\OCUPADO
- TFG_DB\LIBRE

III | Grafana

Para la visualización de los datos en Grafana, se añade un nuevo *dashboard* que se llamará OEE. En él se podrá observar el panel mostrado en la siguiente figura, que contendrá el valor actual de OEE, los valores acumulados que ha habido a lo largo del tiempo de las variables OCUPADO y LIBRE, y por último un gráfico temporal de los valores de OEE a lo largo del tiempo.

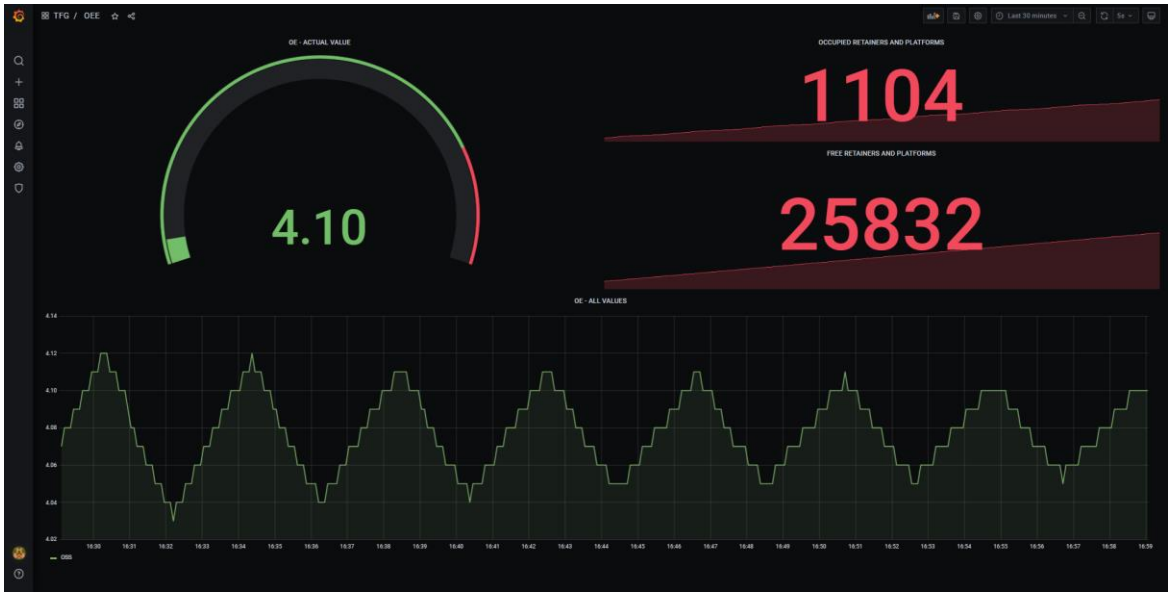

Figura 28: Dashboard del OEE. Fuente: Elaboración propia

Cabe destacar que la repetición de las variaciones en el gráfico histórico de OEE, y el crecimiento continuo de los valores de OCUPADO y LIBRE es debido a que se los datos recogidos son de la simulación en el archivo CSV, y el programa genera un escaneo cíclico de los datos una vez ya se hayan acabado las peticiones.

4.4.2. Productos terminados totales

A nivel de producción, es interesante conocer la cantidad de producto fabricado a lo largo de una jornada laboral o dentro de un intervalo determinado de tiempo. Este indicador puede dar ideas de los problemas que pueden estar generando una baja producción o una producción no constante, y

poder encontrar los motivos que los ocasionan para generar una solución que mejore la productividad en una fabricación.

Para conseguir el valor de las piezas producidas, se partirá de nuevo del Excel de simulación donde se verán los valores del ESTADO del producto dentro de los objetos de BANDEJA SALIDA 1 o de BANDEJA SALIDA 2. En esta aplicación, los valores que puede tener esta variable siguen la siguiente lógica:

Estado	Significado
= 0	Si el estado es igual a 0, se considera que el producto no está terminado
= 1	Si el estado es igual a 1, se considera que el producto sí está terminado

Tabla 11: Lógica para el cálculo de producción. Fuente: Elaboración propia

Estos valores están almacenados dentro del octavo y decimosegundo word de cada equipo. Esto puede verse explicado en el apartado 3.1.4 Estructura de datos de la célula industrial[GL25].

En los siguientes puntos se hará una explicación de los pasos seguidos hasta llegar a la visualización de los datos dentro del software Grafana.

I | Node-RED

El flujo para conseguir este valor se inicia a partir del nodo Object to Array, explicado ya en el apartado anterior para el cálculo del OEE.

El bloque para calcular la producción estará compuesto por una variable de tipo Flow para tener el dato almacenado, la cual se inicializará a 0 la primera vez que se ejecute el programa. Esta variable será la que incrementará su valor siempre que una pieza haya terminado dentro del proceso de producción.

El principio para el cálculo de este valor será el mismo seguido que en el apartado para el cálculo de OEE. Partiremos de los datos y columnas donde se almacenan los datos para poder ver los valores que nos interesan y saber si el producto está o no está acabado.

La función comprobará en cada iteración si el octavo o el decimosegundo word del equipo que se está examinando tienen el valor que indica que el producto está terminado. A continuación se pueden ver las figuras que muestran el flujo implicado en toda esta explicación junto a la parte de código dentro del nodo Calculo Producción.

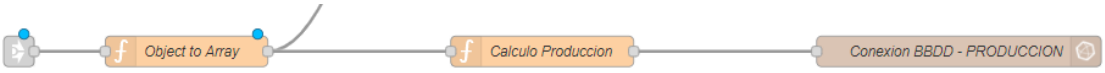

Figura 29: Flujo para el cálculo de producción. Fuente: Elaboración propia

```
    // inicializar Producción a 0
if (flow.get("produccion") === undefined) {
    flow.set("produccion", 0)
}
var col = 8;
    // iteraciones dentro del vector de datos
while (col<=482) {
    if (msg.payload[col] == 1) {
        flow.set("produccion", flow.get("produccion")+1);
    }
    col = col + 4;
    if (msg.payload[col] == 1) {
        flow.set("produccion", flow.get("produccion")+1);
    }
    col = col + 9;
}
msg.payload = flow.get("produccion");
return msg;
```

Figura 30: Código para el cálculo de Producción. Fuente: Elaboración propia

II | InfluxDB

Como ya se ha explicado, el almacenamiento de los datos se realizará en la base de datos TFG_DB, dentro del siguiente *measurement*:

- TFG_DB\PRODUCCION

III | Grafana

Para visualizar los datos en Grafana, se añade un nuevo *dashborad* que se llamará PRODUCCIÓN. En él únicamente se visualizará la cantidad de elementos terminado que se hayan detectado desde la inicialización de la adquisición y guardado de datos, pero en diferentes paneles que otorgarán diferentes vistas.

El primer panel contendrá el número total de elementos terminados desde el inicio de la adquisición de los datos. Este número devolverá de forma clara, rápida y concisa un valor que es altamente importante para cualquier planta de producción.

El otro panel indicará la producción agrupada por rangos de una hora, pudiendo así detectar posibles fallos en la organización de la producción en determinados rangos horarios.

Ambos paneles se muestran en la siguiente figura.

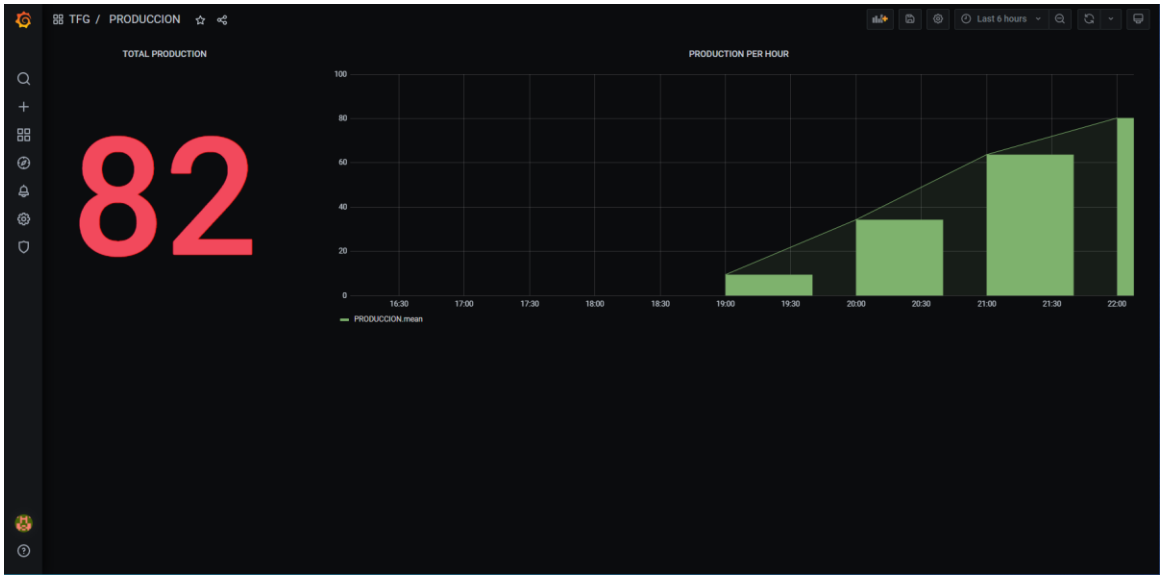

Figura 31: Dashboard de la producción. Fuente: Elaboración propia

Esta representación de valores es posible gracias a la funcionalidad que presta Grafana a la hora de recoger los datos de la base de datos, los cuales pueden ser agrupados dentro de rango configurable de tiempo.

5. Conclusiones y posibles mejoras

A lo largo de esta memoria se han explicado las importancias de una correcta monitorización de un sistema industrial para poder sacar conclusiones para la mejora del mismo, junto con una de las múltiples posibilidades de ejecución de esta tarea.

Como es lógico, todo sistema implementado está sujeto a posibles mejoras y cambios que pueden dar una mayor funcionalidad, y no hayan podido ser implementados por múltiples motivos. En este apartado se pretende dar varias posibilidades para mejorar esta solución.

En primera parte, ya se ha comentado la importancia del índice de OEE dentro de un sistema automatizado, y la implementación en él puede traer grandes conclusiones de mejora. A pesar de ello, un punto importante también sería la implementación de un índice OEE para cada elemento, dando así una imagen más individual de cada uno de ellos, y pudiendo mejorar el sistema realizando cambios de posicionamiento, o los circuitos que pueda seguir una bandeja para aprovechar al máximo todo el sistema.

Otra posible mejora podría ser el acceso desde el software Grafana hacia un elemento en concreto, y que este nos devuelva sus valores actuales de estado, si tiene un producto o no en su posición, o si en caso de tener un producto en su posición, poder ver información sobre él. Esto podría ayudar a la localización de los elementos de una forma clara y rápida en el sistema. Junto a esta mejora, se une la posibilidad de diferenciar individualmente cada bandeja entrante en el sistema, pudiendo así detectar el recorrido que una u otra están generando, ya que con la solución actual puede llegar a ocurrir que los indicadores se crucen entre ellos y se pierda el seguimiento de la bandeja.

Por otra parte, una buena mejora sería la visualización de la bandeja entre dos equipos, ya que actualmente este paso no devuelve ningún dato en especial para poder verlo en el software de visualización, y cabe la posibilidad que se dé el caso de que no se encuentra una bandeja en el sistema ya que se encuentra en una transición entre equipos.

A nivel de programación, una posible mejora sería la adquisición del archivo CSV que simula el sistema una única vez, permitiendo así una mayor optimización de los recursos en la adquisición de los datos, y evitando posibles problemas en caso de que el archivo sea variado o corrompido.

A nivel de las pantallas de visualización, un buen añadido sería la generación de una pantalla principal, desde la que se pueda acceder a partir de botones o enlaces a todas las demás pantallas. En esta pantalla principal se podría añadir, por ejemplo, si el sistema se encuentra en funcionamiento o parado, y algunos de los valores sin detalle para tener una vista rápida del estado de funcionamiento. Para el acceso a las pantallas, también podrían crearse unas credenciales para

dar acceso a un grupo en concreto de usuarios, generando así distintos niveles de permisos que limiten la visualización de ciertos datos.

Por último, la última mejora sería la implementación de la adquisición de los datos reales de la plataforma. Este punto permitiría comprobar la viabilidad y robustez del sistema conectándose con él a tiempo real. Por otra parte, la posibilidad de esta implementación queda demostrada con la adquisición de datos del sensor de vibración, donde los sistemas sí que se han conectado a tiempo real con la planta y han conseguido adquirir valores sin problema.

En conclusión, esta implementación demuestra que los sistemas de adquisición de datos, guardado de los mismos y visualización a tiempo real son de gran ayuda para la producción, ya que arrojan datos y cálculos a tiempo real que permiten una rápida reacción ante alguna incidencia. Además, todos los softwares implementados presentan una gran sencillez de uso y una gran comunidad de usuarios donde se plantean muchos temas que pueden tener relación con un problema y ayudan a buscar una solución.

Queda claro que para la implementación de un sistema como el expuesto en esta memoria actualmente no se necesitan grandes niveles de programación ni conocimiento de redes. Eso es gracias a la sofisticación de los programas, cada vez diseñados a más alto nivel, y la cantidad de información que se presta hoy en día en internet. Esta sofisticación también facilita enormemente la tarea de interconexión entre los mismos programas que, gracias a la sencilla instalación de un *plugins* o alguna otra característica, permite una rápida y sencilla comunicación entre los mismos.

Un gran mercado que se está abriendo hoy en día es el mercado de la domótica. Esta solución es fácilmente plausible en un sistema domótico de una casa personal, debido a los puntos explicados anteriormente y la libertad de uso de estos softwares.

6. Bibliografía y webgrafía

Basco, A., Beliz, G., Coatz, D., Garnero, P. (2018) *Industria 4.0, Fabricando el Futuro.* Inter-American Development Bank.

Bilbao y Ramón Lanza, L. *Historia económica (Tema 5: Los inicios de la Segunda Revolución Industrial, 1870-1914).* Universidad Autónoma de Madrid.

Chaves Palacio, J. (2004). *Desarrollo tecnológico en la primera revolución industrial.* Revista de Historia.

Damm, A. (25 de febrero de 2013). *Crecimiento del PIB: Causa y efectos.* Recuperado de http://www.asuntoscapitales.com/articulo.asp?ida=6602.

Fernández, A., Delgado, M. *Proyecto: Procesado de datos en Matlab. Adquisición, tratamiento y visualización de datos.* Edicions UPC.

Flowcharting-repository, Manuals and repo for plugin flowcharting. GitHub. Recuperado de https://algenty.github.io/flowcharting-repository/STARTED.html

Galbiatti, M. *Revolución industrial.* Atlantic International University.

Historia del mundo contemporáneo. La revolución industrial: de las sociedades agrarias a las industriales. (2009). Universidad Nacional de Educación a Distancia.

Inman, Daniel J. (1996) *Engineering vibration.* Upper Saddle River, NJ

Pérez Reverte, J. (junio de 2020). *Estudio e implementación de una plataforma digital para el despliegue de aplicaciones en el marco de la Industria 4.0.* Trabajo final de grado, UPC, Escola Tècnica Superior d'Enginyeria Industrial de Barcelona.

Principios de automatización. Pirámide CIM (5 de marzo de 2008). Recuperado de http://automatizacion2008.blogspot.com/2008/03/piramide-cim.html

Rodal, E. (7 de enero de 2020). *Qué son los Sistemas Ciber-Físicos en la Industria 4.0* [Audio podcast]. Recuperado de https://www.youtube.com/watch?v=Fb3okO9Aemo

Seitz, M. (17 de marzo de 2017). Qué países tienen más robots en sus fábricas y cuán cierto es que nos están robando los puestos de trabajo. *BBC Mundo*. Recuperado de https://www.bbc.com/mundo/noticias-39267567

The Future of Jobs Report 2018 (2018). World Economic Forum.

The Third Industrial Revolution: A Radical New Sharing Economy. Vice Documentary Films.

Wu, C. (5 de septiembre de 2017). *Computer Integrated Manufacturing.* Departamento de automatización de la Tsinghua University.
Recuperado de http://www.simflow.net/publications/books/*cimie*-part1.pdf

5 lenguajes de programación para PLC. Seika. Recuperado de https://www.seika.com.mx/5-lenguajes-de-programacion-para-plc/