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

Trabajo Final de Grado

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Romero Lazo, Luis Eduardo

ESTUDIO DEL PROCESO DE DISEÑO Y DESARROLLO DE APLICACIONES BASADAS EN REALIDAD AUMENTADA APLICADAS A UN PROCESO INDUSTRIAL

Director: Delgado Prieto, Miguel

Codirector: Fernández Sobrino, Ángel

Convocatoria: Junio, 2021

Declaración de honor

I declare that,

the work in this Degree Thesis is completely my own work, no part of this Degree Thesis is taken from other people's work without giving them credit, all references have been clearly cited.

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

Luis Eduardo Romero Lazo		22/06/2021
Student Name	Signature	Date

Title of the Thesis: ESTUDIO DEL PROCESO DE DISEÑO Y DESARROLLO DE APLICACIONES BASADAS EN REALIDAD AUMENTADA APLICADAS A UN PROCESO INDUSTRIAL

Agradecimientos

A Miguel Delgado y Ángel Fernández, por ser mis tutores y guiarme, aconsejarme, dedicarme su tiempo y animarme con este proyecto.

A todas las amistades que he hecho durante estos años de carrera y la han hecho más agradable.

A mi familia, por apoyarme, darme la oportunidad de estudiar algo que me gustaba y ha acabado gustándome mucho más y por brindarme todas las facilidades que han podido sea cuáles sean las circunstancias.

Gracias.

Resumen

En este documento se presenta de manera explicativa el proceso que se ha llevado a cabo con tal de realizar el proyecto. Dicho proyecto consiste en realizar una aplicación de realidad aumentada que interaccione con el laboratorio de Schneider Electric, ubicado en la Escuela Superior de Ingeniería Industrial, Aeroespacial y Audiovisual de Terrassa (ESEIAAT), de la Universidad Politécnica de Cataluña (UPC), que sea capaz de visualizar información que pueda considerarse relevante, así como poder acceder a guías de mantenimiento mediante esta.

Primeramente, se explicará cuál es el punto de inicio, así como algunos conceptos previos necesarios para entender el resto del documento.

Seguidamente se explicarán las decisiones tomadas a la hora de diseñar la aplicación, qué valores mostrar y cuáles no y la manera de hacerlo. Para ello se trabajará con *InfluxDB* y *NodeRed* como principales programas a tener en cuenta, seguido de *EcoStruxure Augmented Operator Advisor* como la aplicación principal de desarrollo de la app. Primeramente, se simulará el área de trabajo mediante un archivo CSV (valores separados por comas) con la intención de imitar los posibles cambios que se puedan dar.

En los siguientes dos capítulos se hablará sobre la implementación de la aplicación, es decir, cómo se ha desarrollado la aplicación en sí, consideraciones adicionales y su validación.

Finalmente, se darán las conclusiones del proyecto y se mencionarán posibles futuros pasos

Abstract

In this document you'll find, in an explanatory way, the process that has been carried out in order to make this project. This project consists of making an application of augmented reality that interacts with the Schneider Electric laboratory, located in the college of industrial, aerospace and audiovisual engineering of Terrassa (ESEIAAT), of Polytechnic University of Catalonia (UPC), that is capable of visualize data that may be relevant, as well as be able to access to maintenance guides through the app.

First, the starting point will be explained, as well as some previous concepts that are necessary to understand the document.

After that, the decisions made when designing the application, which values will be shown and which not and the way to do it will be explained. In order to do that, *InfluxDB* and *NodeRed* will be used as the main programs to consider, followed by *EcoStruxure Augmented Operator Advisor* as the main developing program for the app. First, the working area will be simulated by making use of a CSV file (comma separated values), aiming to imitate the possible changes that may occur.

The next two chapters are about the implementation of the app, that is to say, how the app has been developed, some additional considerations and its validation.

Finally, the last chapter are the conclusions of the project and possible next steps will be mentioned.

Índice de contenido

Índice de ilustraciones

Capítulo 1

1.1 Objetivos

El objetivo de este proyecto es realizar una aplicación capaz de integrar tecnología de realidad aumentada en un proceso industrial para mantenimiento y supervisión. En este caso se hará en el laboratorio de Schneider Electric de la ESEIAAT, donde se dispone de una célula industrial compuesta de varias estaciones de trabajo controladas por sus respectivos PLC's que trabajan con distintos buses de comunicación. Con esto, será necesario conocer la estructura de comunicaciones con la finalidad de realizar una conexión entre la app que se pretende desarrollar y los PLC. También será necesario decidir qué variables se desea supervisar en la app con la intención de optimizar tanto el diseño como la funcionalidad de la misma. El resultado final pretende ser una aplicación que permita la visualización de datos y otra capaz de dar soporte al mantenimiento, ya sea con procedimientos creados en la app o documentos adicionales.

1.2 Alcance

Para realizar este proyecto se han tenido que realizar las siguientes tareas:

- Comprender el caso de estudio sobre el que se trabajará
- Conceptualizar la estructura de comunicaciones necesarias
- Identificar las funcionalidades de la realidad aumentada y su uso en un proceso industrial
- Búsqueda de información sobre el software de *Schneider Ecostruxure Augmented Operator Advisor* y familiarización con el mismo
- Diseño y desarrollo de aplicación de prueba
- Búsqueda de información sobre protocolos de comunicación
- Familiarización con el software *Node-RED*
- Documentación sobre *InfluxDB*
- Creación de una base de datos haciendo uso de *InfluxDB*
- Integración de la base de datos con *Node-RED*
- Integración *EcostruxureAOA* con el software *Node-RED*

- Diseño de aplicación definitiva
- Desarrollo de aplicación definitiva
- Validación de aplicación final

Sin embargo, debe mencionarse que no se realizarán:

- Modificaciones en el sistema automatizado ya existente. Se trabajará con el hardware y el software que se dispone y no se efectuará ninguna modificación.
- Una aplicación que abarque la totalidad de la línea. En su lugar, se realizará una aplicación de visualización de datos y otra de soporte al mantenimiento sobre dos partes de la línea.

1.3 Requisitos

Para la realización de este proyecto será necesario:

- Licencia activa del software *Ecostruxure Augmented Operator Advisor* de *Schneider Electric*
- Hacer uso de una herramienta de programación, en este caso se utilizará *Node-RED*, un editor de flujo basado en navegador
- Servidor de base de datos de *timeseries*, en este caso *InfluxDB*
- Célula industrial con todos sus componentes (sensores, actuadores, PLC'S, etc) en este caso provisionada por el laboratorio de automatización de *Schneider Electric*
- Una tableta o móvil con la aplicación *Ecostruxure Augmented Operator Advisor App*
- Fotos de alta resolución de los distintos elementos que se desea supervisar. En su defecto, se podrá hacer uso de códigos QR pertenecientes a la aplicación con la finalidad de realizar la misma tarea.
- Adicionalmente y de forma excepcional, dada la situación causada por el COVID-19 se hará uso de *Postman* y *AnyDesk*, con la intención de trabajar remotamente en la medida de lo posible

1.4 Contexto

La realidad aumentada es una tecnología que lleva años en auge y actualmente ya hay varias empresas que hacen uso de esta tecnología. Dicha tecnología consiste en superponer información adicional (por ejemplo, texto, imágenes, vídeos, correos electrónicos o incluso aplicaciones como Google Maps o similares) al mundo real haciendo uso de un dispositivo tecnológico que disponga de cámara. Es decir, a través de un móvil, tableta o incluso otros dispositivos como gafas virtuales (*Google Glass, Hololens,* etc), el usuario puede visualizar contenido digital superpuesto a lo que ve realmente. De esta manera se pueden integrar elementos físicos con elementos virtuales.

Ilustración 1: HoloLens de Microsoft. Fuente: www.Haptic.al

Cada vez se explora más qué utilidades puede tener y sus beneficios, tales como poder tener control y supervisión de un proceso industrial. De esta manera, se tendría una conexión entre máquinas y datos, pudiendo el usuario acceder a ellos de una manera mucho más cómoda y rápida, optimizando así la productividad. Por este motivo, además, es especialmente útil en el sector del mantenimiento, ya que brinda la posibilidad de conocer el estado de distintas máquinas y en caso de ser necesaria algún tipo de intervención, el usuario podría disponer directamente de una guía de procedimiento.

Sin embargo, lo ideal sería hacer uso de gafas virtuales y estas no se usan tanto, sobre todo en empresas pequeñas. No obstante, es algo en crecimiento y cada vez se implementa más, de modo que con el paso del tiempo puede ser algo establecido.

1.5 Motivación

Me resulta una tecnología atractiva para la industria 4.0 y me gustaría ser partícipe en hacer uso de una tecnología en auge como esta, por ese motivo escogí este proyecto. Creo que puedo aprender mucho de su funcionamiento, así como puedo ganar más conocimiento sobre protocolos de comunicación e interacción con PLC's. Considero que independientemente de su funcionamiento, es algo llamativo a nivel visual y que a nivel de usuario es fácil entender cualquier cosa, ya que es un nivel de interacción mucho más entendible puesto que estás constantemente viendo una parte real.

Capítulo 2

2.1 Realidad aumentada e Industria 4.0

2.1.1 Industria 4.0

A mediados del siglo XVIII se inició la primera revolución industrial, una etapa de mecanización de procesos productivos y el cambio de una economía agraria a una industrial.

Entre 1850 y 1914 tuvo lugar la segunda revolución industrial, caracterizada por ser el inicio de la producción en cadena y la producción en masa de energía eléctrica.

La tercera revolución industrial no tiene fecha concreta pero el concepto se aceptó en 2006. Los avances en telecomunicaciones y el uso de energías renovables fueron los que la caracterizaron.

Finalmente llegamos a la cuarta revolución industrial, conocida como Industria 4.0, cuyos inicios son recientes, situándose en la década del 2020. En la Industria 4.0 la automatización industrial es de gran importancia.

Se busca la implementación de "*Smart factories",* fábricas inteligentes digitalizadas con la capacidad de tener una gestión eficiente de recursos y una mayor adaptabilidad a las necesidades de producción. Se introduce el concepto del "Internet de las cosas (IoT)", sistemas con la capacidad de comunicarse entre sí de manera inteligente. También se empiezan a gestionar mayores cantidades de datos gracias a distintos tipos de sensores. Aunque estas ideas son utilizadas en distintos sectores, son especialmente útiles en el sector de la producción ya que son muy relevantes en la automatización industrial.

2.1.2 Realidad aumentada

¿Qué papel juega la realidad aumentada en la industria 4.0? Bien, primero hay que introducir el concepto de realidad aumentada. La **realidad aumentada** consiste en la superposición de elementos virtuales sobre elementos reales (físicos), "aumentando" así la realidad. A través de un dispositivo con cámara y pantalla, se pueden ver estos elementos virtuales superpuestos sobre los elementos físicos que vemos en pantalla. No debe confundirse con la **realidad virtual** ya que esta última tiene como intención crear un espacio totalmente virtual. Tampoco debe confundirse con la **realidad mixta**, ya que esta es una combinación de ambas, aportando elementos adicionales a la realidad, pero llegando incluso a modificar tu entorno y crear espacios virtuales en función de tu entorno físico.

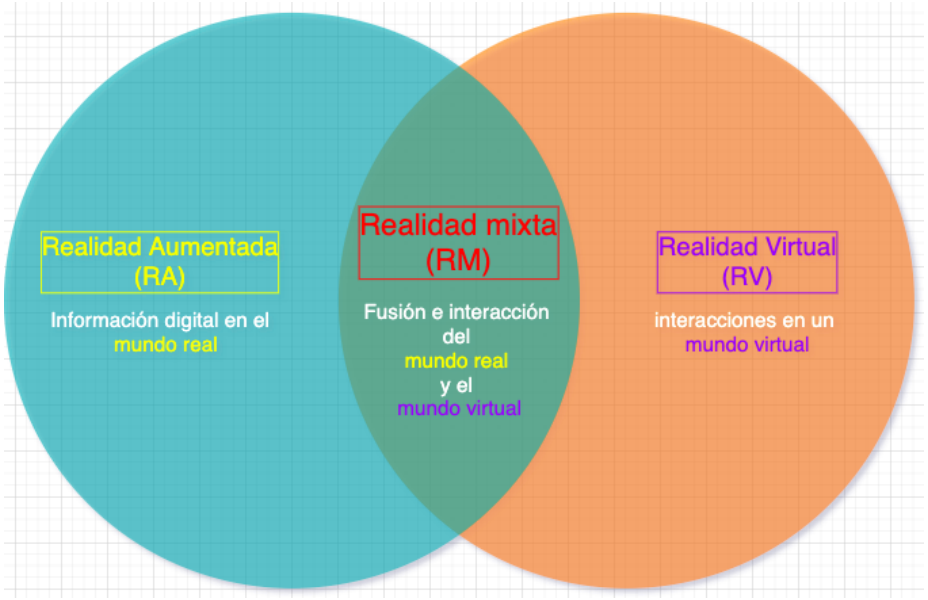


Ilustración 2:Diferencia entre Realidad Virtual, Mixta y Aumentada. [28]

Al tratarse de una superposición de información sobre la realidad, se puede entender por qué es una tecnología interesante; por la optimización. La posibilidad de visualizar información de una manera rápida y clara es realmente de utilidad en sectores donde se busca optimizar al máximo diferentes factores como el tiempo de producción o tener un control de stock, entre otros.

Ya existen empresas que utilizan esta tecnología con distintas finalidades, en este proyecto se verá cómo se puede aplicar en el ámbito de la optimización, pero se debe mencionar que, a día de hoy, algunas empresas utilizan realidad aumentada para dar soporte técnico de forma remota. De esta manera no sería necesario que un técnico se desplace al lugar donde se requiere ayuda y podría ayudar a través de lo que se muestra en pantalla.

2.2 Descripción del área de trabajo

El laboratorio de automatización industrial pretende simular una línea de producción donde se transportan, mediante el uso de bandejas o palés, materias primas por cintas transportadoras y son tratadas en distintas estaciones de trabajo. Para ello, la línea dispone de un conjunto de motores, sensores inductivos, electroválvulas y una serie de elementos que hacen posible el control de las bandejas, tales como accionar las cintas para mover las bandejas, detectar su llega a las distintas estaciones y detenerlas en caso de ser necesario, entre otras tareas.

Se pueden diferenciar dos partes: el pulmón y la célula industrial.

Ilustración 3: Línea de producción del laboratorio de automatización industrial

2.2.1 Pulmón

Ilustración 4: Pulmón del laboratorio de automatización

El pulmón es un sistema automatizado con la capacidad de almacenar bandejas. Está ubicado entre DIR05 y PT06, con la intención de guardar las bandejas vacías que finalicen el ciclo pasando por la estación de extracción, y también con la intención de proveer de bandejas a la línea a través de la estación de carga de materias primas, en función de las necesidades del proceso de producción.

Dispone de cuatro cadenas con una serie de sensores que permiten la carga y descarga de las bandejas. También dispone de sensores en la parte superior e inferior que indican si el pulmón ha alcanzado su capacidad máxima o, por el contrario, está vacío.

El pulmón está gobernado por un PLC propio, formando así la línea Ethernet.

2.2.2 Célula Industrial

La célula industrial se puede dividir en 4 zonas (líneas) en función del PLC que gobierna a dicha línea y su tipo de comunicación. Como se mencionó antes, el Pulmón compone la línea ethernet, pero también disponemos de la línea CAN, Profibus y AS-i.

Para realizar este proyecto es necesario conocer el flujo de trabajo que se lleva a cabo en el laboratorio. Para este proyecto, se considera el punto de partida la estación PT06, la estación de carga de materias primas, sin embargo, se debe mencionar que el punto de partida real es PT03 y DIR03, componiendo la estación de entrada de bandejas vacías.

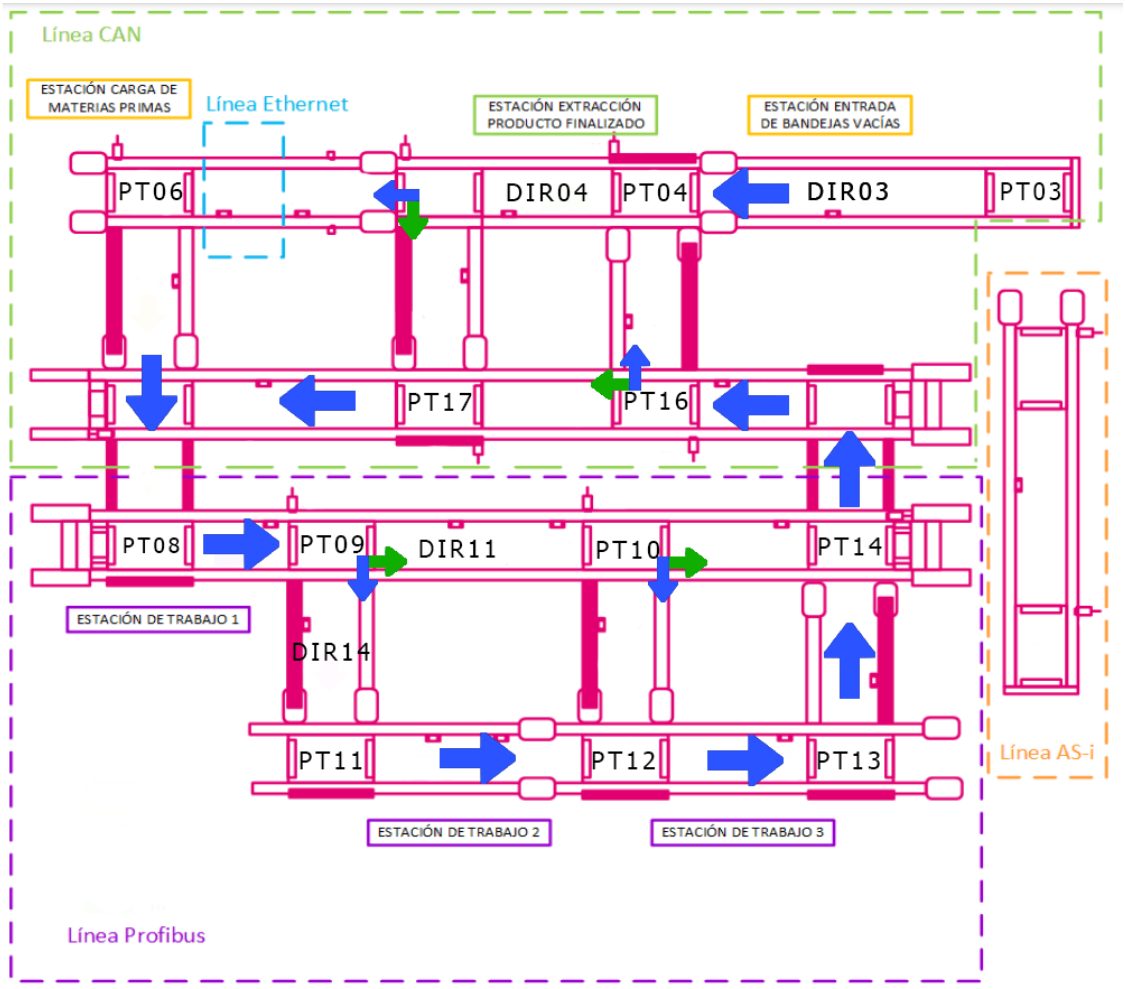


Ilustración 5: Plano Laboratorio Automatización [1]

Desde PT06 se moverá una bandeja o palé, cargado con materias primas, mediante cintas transportadoras hasta PT08, la estación de trabajo 1. Una vez aquí, se simula un proceso en esa estación y una vez terminado pasa a PT09 donde en función del tipo de producto que lleve, seguirá una dirección u otra, en este caso, DIR11 o DIR14. En caso de tomar DIR14, el palé pasará por la estación de trabajo 2, donde se simulará otro proceso, seguidamente de la estación de trabajo 3. Si, por otra parte, la bandeja tomó la dirección DIR11, esta nuevamente podrá ir en una dirección u otra en función del tipo de producto que tenga una vez llegado a PT10. Es decir, desde aquí podrá ir a la estación de trabajo 3 o seguir recto hasta PT14. Desde aquí avanzará hasta PT16 y en caso de que la bandeja tenga producto avanzará hasta DIR04, la estación de extracción de producto. Si no es el caso, avanzará recto volviendo a pasar por las estaciones de trabajo.

Ilustración 6: Bandeja línea de producción

Este proyecto sólo se centrará en el proceso de la línea CAN, ya que esta es la que determina la carga de materias primas en las bandejas y la extracción de productos finalizados. El resto de líneas no son de interés para la resolución de este proyecto.

2.2.3 Funcionamiento retenedores

Otro dato de importancia es cómo funciona cada plataforma o retenedor. Existen cuatro estados distintos:

- Reposo: El retenedor no tiene bandeja.
- Petición: El retenedor tiene una bandeja encima.
- Avance 1: La bandeja ha salido del retenedor sin cambio de dirección.
- Avance 2: La bandeja ha salido del retenedor y cambiando de dirección.

Sin embargo, es posible tener más de un estado al mismo tiempo. El ejemplo más claro es el de estar en reposo y avance al mismo tiempo, ya que en cuanto la bandeja sale del retenedor se volverá a activar el estado de reposo, pero el estado de avance permanecerá hasta que la bandeja llegue al siguiente retenedor.

El periodo de tiempo en el que el estado de avance está activo y el de reposo inactivo es bastante reducido, de unos 200 milisegundos, por lo que prácticamente el estado de avance y reposo estarán activos simultáneamente gran parte del tiempo.

Otro caso que se puede dar es el de estar en el estado de petición y avance, teniendo una bandeja en el retenedor y otra avanzando hacia el siguiente.

2.3 Conceptos previos

2.3.1 EcoStruxure Augmented Operator Advisor

Para implementar la realidad aumentada, Schneider Electric dispone de un software de desarrollo llamado EcoStruxure Augmented Operator Advisor (de ahora en adelante se le llamará EcoStruxureAOA en este documento). Según Schneider, la intención principal de este software es la optimización del tiempo; acceso rápido y efectivo a información deseada, localización de fallos más rápidos y mayor facilidad en el mantenimiento.

Ilustración 7: Tableta haciendo uso de EcoStruxureAOA. Fuente: www.cambiodigital-ol.com

En EcoStruxureAOA se pueden diferenciar 3 partes:

1. **Builder**

   Es un software accesible vía web (es decir, que no requiere descargar absolutamente nada) donde se desarrolla la aplicación final. Aquí es donde se añaden las áreas que serán controladas, las variables deseadas, procedimientos y todo tipo de información que se desee mostrar.

   Para que la realidad aumentada funcione, se realiza una comparación constante de imágenes previamente almacenadas (llamadas Escenas en el builder) con lo que se ve a través de la app mediante el uso de la cámara y es aquí, en el builder, donde se almacenarán.

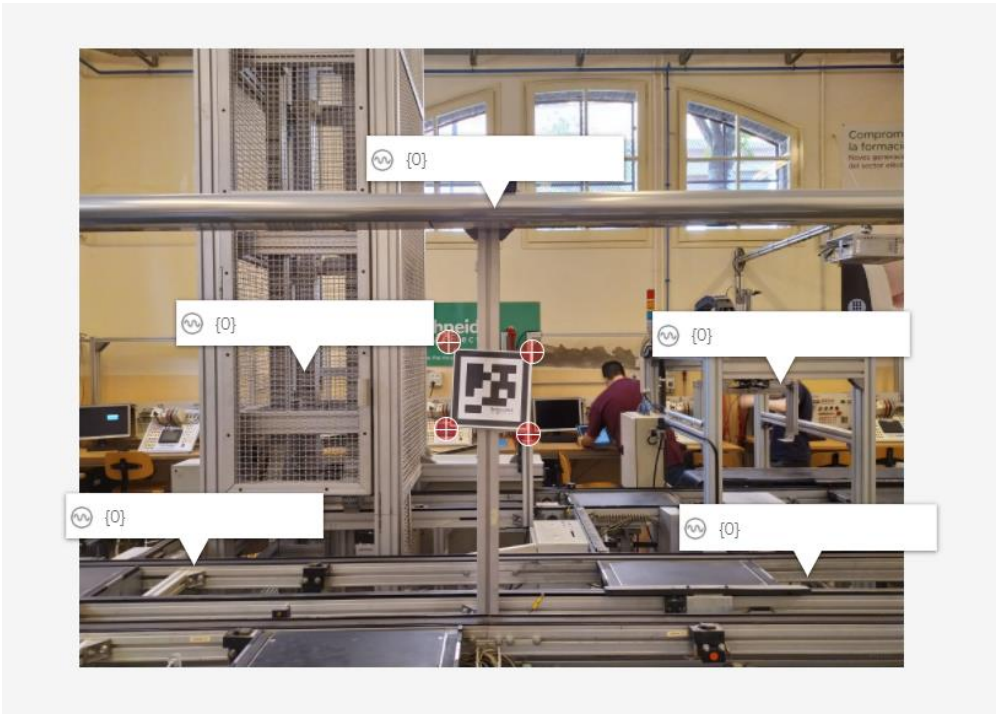


Ilustración 8: Ejemplo de Escena en el Builder con 5 puntos de interés

De forma alternativa y recomendable, si se trabaja en un entorno cuyas imágenes sean muy parecidas entre sí, se puede hacer uso de códigos bidi proporcionados por la propia aplicación. Los códigos bidi son un tipo de código de barras, en este caso bidimensionales, que contienen determinada información. Requieren licencia para ser creados, por lo que este proyecto se limita a los que proporciona Schneider Electric en su software. En este caso, cada código bidi tiene asociada una identificación, de manera que podemos asociar variables a cada código bidi como si de escenas distintas se tratara.

De esta manera, bastará con imprimir los códigos y colocarlos en las zonas de interés y en función de qué código bidi mostremos, se mostrará una información u otra, según el diseño que se haya llevado a cabo.

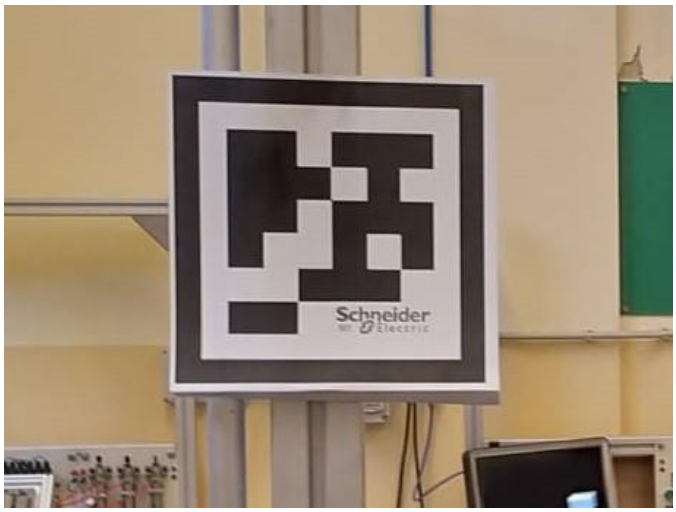


Ilustración 9: Ejemplo de código bidi de Schneider

Tanto si se usan códigos bidi como imágenes del área de trabajo, en cuanto la aplicación detecte similitudes mostrará la "parte virtual", es decir, los elementos superpuestos que haya decidido añadir el programador, llamados puntos de interés. Estos pueden ser:

- Variables
- Documentos
- Enlaces a páginas web
- Audios y Vídeos
- Procedimientos
- Aplicaciones Externas

Como este proyecto se centra en la supervisión, se usarán únicamente variables, con la intención de ver valores en tiempo real. Las variables se obtendrán gracias a Node-RED, que nos permitirá la comunicación con el PLC que gestiona la línea donde se pretende implementar la aplicación.

Una vez hecha la aplicación, tan solo habrá que compilar y llevarla al ordenador donde esté instalado el runtime.

2. Runtime

El runtime actuará de servidor de la app de la tableta. Recopila toda la información que se mostrará en cada punto de interés creado en el builder y los envía a la EcoStruxureAOA App de la tableta. En este caso y gracias a *Node-RED,* enviará las variables en tiempo real a la app. Las fotos de las

Se encarga también de gestionar la base de datos donde están las fotos de las distintas escenas del proyecto que se haya cargado.

Para ello, basta con tener la carpeta zip descomprimida, que se obtiene al compilar el proyecto en el builder, en el ordenador donde se ejecutará el runtime.

3. App

Finalmente se instala una aplicación en una tableta o móvil. Para que esta funcione, debe estar conectada a la misma red que el ordenador donde está instalado el runtime. La aplicación hará uso de la cámara del dispositivo buscando coincidencias con las fotografías hechas previamente (añadidas en el builder) y cuando encuentre una mostrará todos los puntos de interés. Desde aquí, el usuario podrá visualizar las variables, iniciar procedimientos, acceder a otras escenas y cualquier otra cosa que se haya añadido desde el builder.

Ilustración 10: Diagrama de conexión. Fuente: Guía de Inicio rápido EcoStruxureAOA

Como se puede ver en el diagrama, el PLC con el que se trabajará enviará información al ordenador donde estará instalado el runtime, donde previamente se habrá cargado el proyecto creado en el Builder. Mientras el runtime esté ejecutándose, se podrá hacer uso de la aplicación instalada en la tableta conectándose al runtime de manera inalámbrica y de esta manera ver información proveniente del PLC, procesada para que aparezca como se haya diseñado en el Builder.

2.3.2 Node-RED

Node-RED es una herramienta visual de programación con la que se puede programar prácticamente sin escribir código. Está basado en flujo, por lo que para programar simplemente se establecen relaciones entre los distintos nodos. Los nodos son bloques preprogramados, es decir, cada nodo que podemos encontrar tiene una función predefinida y no nos limitamos únicamente a los que incluye por defecto, gracias a la comunidad de programadores podemos obtener multitud de nodos con distintas funciones, incluso pudiendo utilizar protocolos como Modbus.

Ilustración 11: Ejemplo de un simple programa con Node-RED. La salida será un "Hello World!"

Node-RED puede enviar y recibir datos constantemente, por lo que su uso aquí será principalmente enviar datos a un servidor creado con InfluxDB y también enviar los valores de cada variable a visualizar en EcoStruxureAOA. Por otra parte, se encargará también de obtener cualquier dato que podamos necesitar. En este caso, recibirá datos vía Modbus del PLC que gestiona la línea de producción. EcoStruxureAOA ya incluye Node-RED en el runtime, por lo que no será necesario instalar nada adicionalmente.

2.3.3 InfluxDB

InfluxDB es un servidor basado en *timeseries*. Es decir, cada dato que se sube al servidor se visualiza en una fila con distintas columnas que normalmente suelen ser tres: el nombre del dato en cuestión, su valor y el momento exacto en el que fue subido, valor que ya añade el propio servidor por defecto. Es especialmente útil en proyectos como este, ya que al tener la fecha en la que ha ocurrido cada evento, podemos consultar cualquier posible cambio. Por poner un ejemplo, si la línea de producción está tardando más de lo habitual en la última hora, se podrían filtrar los datos para visualizar solamente un rango de tiempo y verificar si hay algo fuera de lo común.

Para este proyecto se subirá el estado de la línea de producción cada vez que este cambie y luego mediante Node-RED se realizarán las consultas necesarias.

Capítulo 3

3.1 Planificación

Con tal de llevar a cabo el desarrollo de la aplicación se han seguido una serie de pasos y se han tomado ciertas decisiones de diseño. Para empezar, en el siguiente diagrama se puede ver los pasos a seguir con tal de realizar dicha tarea.

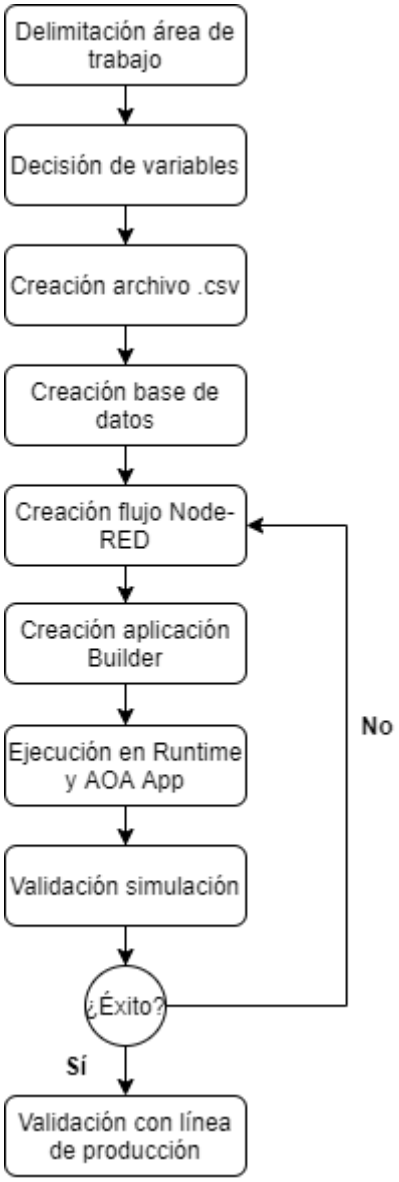


Ilustración 12: Diagrama de planificación

Se ha optado por decidir primero cuál será la sección de la línea con la que se trabajará para posteriormente hacer una elección de las variables necesarias y las que se mostrarán.

Después de eso, se creará el archivo en formato CSV con las variables seleccionadas para poder realizar la simulación. Esto se hace con la intención de probar la aplicación final con variables controladas y así conociendo el output que se debería obtener. Posteriormente se procederá ya con los softwares de InfluxDB y Node-RED. Primero se creará una base de datos en Influx y una vez creada se configurarán los nodos tipo Influx de Node-RED. Posteriormente se creará el flujo necesario para la aplicación y se creará la aplicación en el builder.

Una vez creada la aplicación en el *builder*, se compilará y se llevará al *runtime*, donde se ejecutará. También se conectará con la aplicación de la tableta y se procederá a validar la simulación.

En caso de éxito se pasará a una validación con la línea de producción operando de manera continua, de no ser el caso, habrá que revisar cada paso desde la creación del flujo en Node-RED.

3.1.1 Delimitación del área de trabajo

En este proyecto no se pretende abarcar toda la línea de producción sino más bien una parte de ella. Tal y como está programada la línea, es más fácil evitar los cambios de dirección. De esta manera, el recorrido que se va a controlar siguiente, empezando en la carga de materias primas y acabando en la estación de extracción:

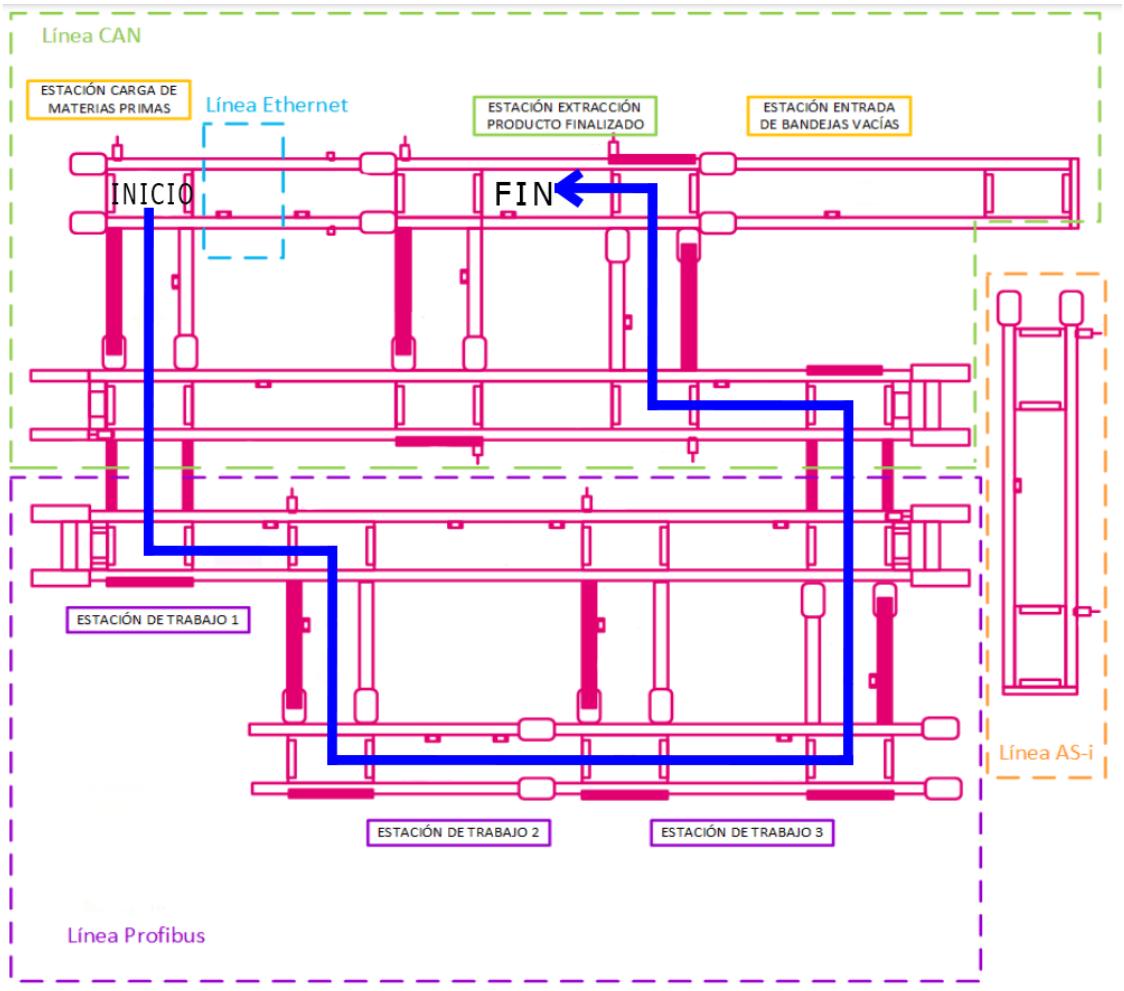


Ilustración 13: Recorrido a controlar [1]

3.1.2 Variables necesarias

Independientemente de los datos a mostrar, se deben filtrar las variables que serán de utilidad y las que no. Todas las variables aquí seleccionadas se han extraído del trabajo final de grado *"Estudio de las etapas de automatización de un proceso industrial",* y pueden diferir con las variables actuales de la línea de producción

Puesto que necesitamos controlar el tiempo que tarda en producirse cada pieza, será necesario saber qué recorrido ha hecho. Para ello, se usarán todas las variables de estado de cada retenedor. Es decir, se cogerán los estados de todos los retenedores, de PT03 a PT17, independientemente de que algunos retenedores se usen o no.

Puesto que durante la fabricación de un producto puede iniciarse la fabricación de otro antes de que el primero termine, se asocian distintos ID a cada bandeja. Estos también serán necesarios para desarrollar la aplicación, ya que a la hora de contabilizar el tiempo queremos asegurarnos de estar contando el tiempo de una bandeja en concreto y no mezclarlos entre sí. Estos están divididos en 3, **"ID_BANDEJA_CARGA"**, **"ID_BANDEJA_DIR1"** y **"ID_BANDEJA_DIR2".**

Adicionalmente, se utilizarán las variables de **Paro/Rearme** porque será uno de los datos que se pretenden visualizar en la aplicación final. Por el mismo motivo, las variables **"LOTE_CARGADO"** y **"PRODUCTOS_POR_CARGAR"** también se utilizarán.

También se utilizará la variable **"CONTADOR_BANDEJAS_VACIAS"** para tener un control de cuántas bandejas hay sin utilizar.

3.2 Datos a visualizar

3.2.1 OEE

Dado que la intención es optimizar los tiempos de producción, es vital tener controlados los tiempos que se tarda en realizar cada pieza y así poder detectar anomalías. Para ello, existen los llamados **KPI** (del inglés, ***K**ey **P**erformance **I**ndicator)* que son indicadores clave de rendimiento. La intención de estos es medir de forma porcentual hasta qué punto se cumple determinado objetivo y así poder determinar si se requiere tomar algún tipo de medida con tal de mejorar dicho resultado. Los KPI se pueden utilizar en distintos sectores, pero para este proyecto se usará únicamente uno, destinado a procesos de fabricación: el **OEE** (del inglés ***O**verall **E**quipment **E**fectiveness*), eficiencia general de los equipos.

La fórmula para calcular el OEE es la siguiente:

$$OEE = Disponibilidad * Eficiencia * Calidad$$

Viendo la fórmula se puede entender por qué es interesante utilizar este KPI; incluye tres parámetros que son realmente interesantes de evaluar a nivel de producción.

- Disponibilidad

  La fórmula para calcular la disponibilidad es la siguiente:

$$Disponibilidad = \frac{Tiempo\ de\ ejecución}{Tiempo\ Producción\ Estimado}$$

$$= \frac{Tiempo\ Producción\ Estimado - Tiempo\ de\ stop}{Tiempo\ Producción\ Estimado}$$

  En este caso, se supondrá que la máquina está disponible en todo momento.

- Eficiencia

  La fórmula para calcular la eficiencia es la siguiente:

$$Eficiencia = \frac{Tiempo\ de\ ciclo\ ideal * Total\ piezas\ producidas}{Tiempo\ de\ ejecución}$$

El tiempo de ciclo ideal será un valor estimado que se fijará haciendo simulaciones previamente.

- Calidad

La fórmula para calcular la calidad es la siguiente:

$$Calidad = \frac{Piezas\ bien\ hechas}{Total\ piezas\ producidas}$$

Sin embargo, en este proyecto se ignorará este parámetro dado que no se producen piezas realmente y no se dispone de una manera de simular la producción de piezas buenas o malas, así que se considera que todas las piezas son buenas, es decir, el parámetro calidad se considerará equivalente a 1.

Como se ha mencionado, al suponer calidad y disponibilidad equivalentes a 1, se obtiene que el OEE tendrá el mismo valor que la eficiencia, pero se debe mencionar que en un caso real esto no sería así y que por tanto el OEE que se obtendrá no tiene por qué reflejar la realidad ya que únicamente se va a tener en cuenta la velocidad de producción de las piezas y se omitirá su calidad y las pausas que muy probablemente habría en un caso real. De todos modos, si se tratase de un caso real, los siguientes valores pueden ser usados de guía:

- 40%, es común ver este OEE en empresas que están empezando. Es un OEE realmente bajo, pero suele ser fácilmente mejorable.
- 60%, es un OEE bastante común e implica que hay potenciales mejoras.
- 85%, es un valor bastante bueno y al que muchas empresas intentan llegar, ya que implica que hay pocas pausas, alta velocidad de producción y la mayoría de piezas son buenas.
- 100%, sería el OEE ideal, donde no hay lugar a más mejora. Implicaría que se está produciendo únicamente piezas buenas, lo más rápido posible y sin pausas de por medio.

3.2.2 Otras variables

Aparte del OEE hay otras variables que se han optado por mostrar.

- Estado de emergencia y tiempo de emergencia.

Pese a que se puede visualizar el estado de emergencia desde el PLC o incluso desde el pulsador de emergencia, se ha considerado que es interesante visualizarlo a través de la app a modo de alerta. De igual manera, en cuanto se dé el estado de emergencia, se mostrará un contador para contabilizar la duración de esta. Sin embargo, ambas variables permanecerán ocultas fuera del estado de emergencia ya que se trata de un caso excepcional y el espacio que podemos visualizar en pantalla es limitado, por lo que es preferible ahorrar espacio y no excederse con las variables en pantalla.

- Productos y lotes restantes

Como la intención de la aplicación es tener un control sobre la producción, es útil saber en todo momento cuántos lotes y productos quedan por producir.

- Último tiempo

Esta variable tiene como finalidad tener un control del tiempo de producción, otra manera de comprobar que los tiempos de cada producto se están cumpliendo. Aunque es cierto que con el OEE ya se tiene una idea de si se están cumpliendo o no los tiempos de producción, de esta manera se podría ver de una forma más rápida si hay algo fuera de lo común sin tener que recurrir a consultarlo en la base de datos.

- Contador de bandejas vacías

Esta variable también se mostrará en la app final. Pese a que en la línea de producción de este proyecto no es realmente necesaria porque prácticamente se podría contabilizar directamente mirando la línea, en líneas de producción de gran escala sería útil tener ese valor mostrándose.

3.3 Conceptos necesarios

Dado lo que se ha comentado en el capítulo anterior, se puede obtener el siguiente diagrama de conexiones:

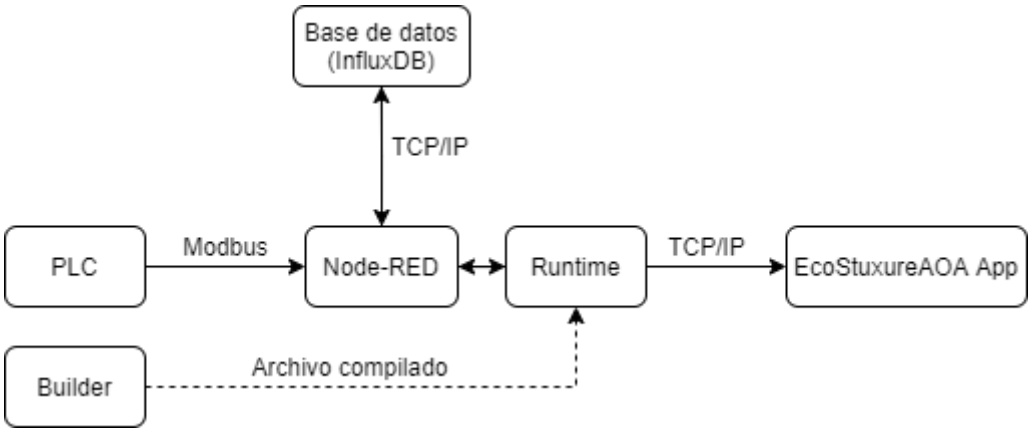


Ilustración 14: Diagrama de conexiones

El PLC con el que se trabajará, el que gobierna la línea CAN, enviará la información necesaria al ordenador donde estará instalado el runtime. Para ello, se enviará a Node-RED vía Modbus. Paralelamente, Node-RED estará enviando datos, vía TCP/IP, a la base de datos creada con InfluxDB con tal de almacenarlos, pero también podrá hacer solicitar datos almacenados.

El runtime estará ejecutando el proyecto compilado creado en el builder, para ello simplemente se tendrá que guardar el archivo en el ordenador donde esté instalado el runtime. Adicionalmente, el runtime estará ejecutando Node-RED.

Siempre y cuando esté el runtime activo, se podrá usar la EcoStuxureAOA App de la tableta conectándose al runtime vía TCP/IP.

3.3.2 TCP/IP

El modelo TCP/IP es un conjunto de protocolos de red basada en internet que permite transmitir datos entre dispositivos. Este modelo incluye 4 capas: la de aplicación, la de transporte, la de internet y finalmente la de interfaz de red.

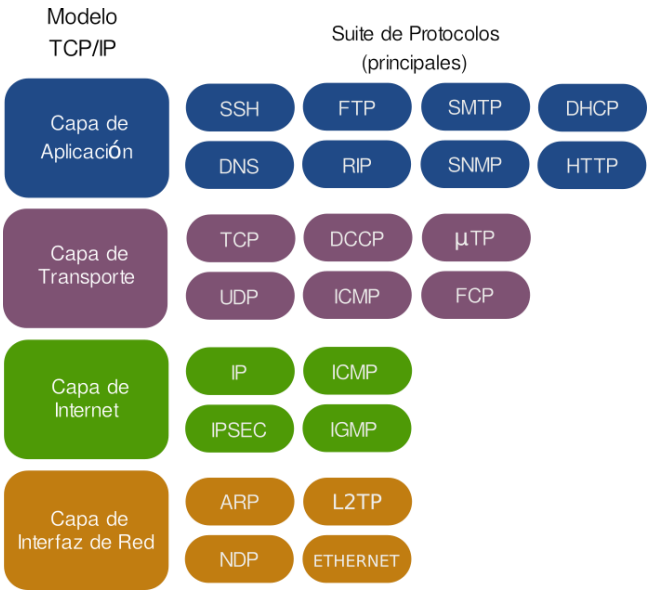


Ilustración 15: Capas modelo TCP/IP

Cada capa incluye a distintos protocolos, pero se mencionarán los que son relevantes para este proyecto:

- IP

  IP (**P**rotocolo de **I**nternet) es un protocolo de comunicaciones que se encuentra en la capa de Internet del modelo TCP/IP. Se encarga de portar paquetes (llamados datagramas) del origen a su destino. Durante su transmisión, estos datagramas pueden fragmentarse si es necesario y volver a unirse nuevamente en su destino.

  Principales características:

  - Direccionamiento mediante direcciones lógicas IP de 32 bits
  - Si un paquete no llega a su destino, permanecerá en la red durante un tiempo limitado
  - Para la distribución de paquetes, trata de buscar siempre la mejor ruta

  Para establecer comunicación entre dos dispositivos, estos tienen que tener una dirección IP. Una dirección IP es un número que identifica a una interfaz de un dispositivo dentro de una red que utilice el protocolo de internet. Esta

dirección, además de identificar la interfaz del dispositivo, también identifica la red a la que pertenece.

- TCP

TCP es un **P**rotocolo de **C**ontrol de **T**ransmisión cuya función es garantizar que los datos se transmitan a su destinatario, sin errores y en el orden correcto. El protocolo TCP da la seguridad que el protocolo IP no tiene, garantizando que lleguen a su destinatario. Es un protocolo orientado a la conexión, esto quiere decir que el origen y el destinatario deben aceptar la conexión previamente para poder empezar la transmisión. Una vez realizada la transferencia de datos, se finaliza la conexión.

Principales características:

- Permite monitorizar el flujo de datos, evitando de esta manera la saturación de la red
- Vuelve a ordenar los segmentos provenientes del protocolo IP
- Brinda la posibilidad de fragmentar los datos en segmentos de longitud variada para el protocolo IP

3.3.3 Modbus

Modbus es un protocolo de comunicaciones basado en la arquitectura maestro/esclavo, donde el maestro hace una solicitud y el esclavo responde a esta. Fue diseñado en 1979 para PLC's de Modicon y a día de hoy es uno de los protocolos de comunicaciones estándar en la industria.

Modbus es usado con frecuencia en la industria, permite controlar una red de dispositivos y la comunicación entre ellos. Dentro de una red modbus, todos los dispositivos disponen de una dirección única. Cada uno de ellos puede enviar datos, pero lo normal es que solo uno de ellos lo haga. Todos los dispositivos reciben cada trama enviada, pero esta tiene la dirección del dispositivo destinatario, de modo que solamente ese dispositivo ejecutará dicha orden.

Para este proyecto se usará la variante Modbus TCP/IP. Esta variante de Modbus utiliza el modelo TCP/IP haciendo uso del puerto 502. La trama modbus se encapsula en una trama TCP

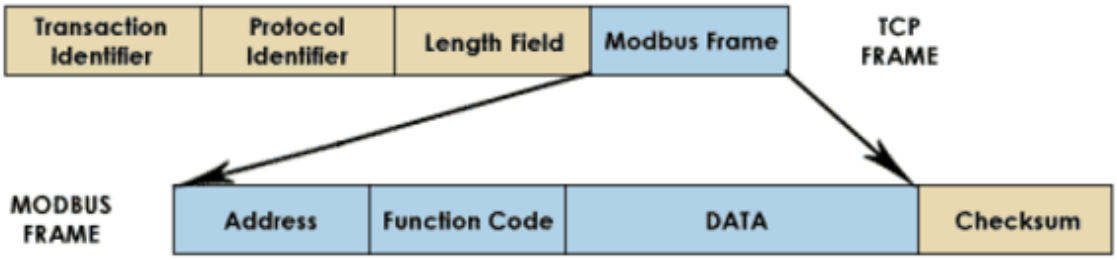


Ilustración 16: Encapsulamiento trama Modbus en TCP

Capítulo 4

4.1 Diseño del archivo de simulación

Como se mencionó en el capítulo anterior, en lugar de hacer el programa directamente con la línea de producción, se hará un programa que simule el funcionamiento de esta. Para ello, se creará un archivo CSV (*comma separated values)*. Estos archivos se pueden crear haciendo uso de Excel, y es exactamente lo que se ha hecho para crear dicho archivo.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	ID_BANDEJA_CARGA	ID_BANDEJA_DIR1	ID_BANDEJA_DIR2	PT03	PT04	PT05	PT06	PT07	PT08	PT09	PT10	PT11	PT12
2	5555	5555	5555	5555	5555	5555	5555	5555	5555	5555	5555	5555	5555
3	-1	-1	-1	1	1	1	1	1	1	1	1	1	1
4	123	-1	-1	1	1	1	2	1	1	1	1	1	1
5	-1	123	-1	1	1	1	5	1	1	1	1	1	1
6	-1	123	-1	1	1	1	5	1	1	1	1	1	1
7	123	-1	-1	1	1	1	1	2	1	1	1	1	1
8	-1	123	-1	1	1	1	1	5	1	1	1	1	1
9	-1	123	-1	1	1	1	1	5	1	1	1	1	1
10	123	-1	-1	1	1	1	1	1	2	1	1	1	1
11	-1	123	-1	1	1	1	1	1	5	1	1	1	1
12	-1	123	-1	1	1	1	1	1	5	1	1	1	1
13	456	-1	-1	1	1	1	2	1	1	1	1	1	1
14	-1	456	-1	1	1	1	5	1	1	1	1	1	1
15	-1	456	-1	1	1	1	5	1	1	1	1	1	1
16	456	-1	-1	1	1	1	1	2	1	1	1	1	1
17	-1	456	-1	1	1	1	1	5	1	1	1	1	1
18	-1	456	-1	1	1	1	1	5	1	1	1	1	1
19	456	-1	-1	1	1	1	1	1	2	1	1	1	1
20	-1	456	-1	1	1	1	1	1	5	1	1	1	1
21	-1	456	-1	1	1	1	1	1	5	1	1	1	1

Ilustración 17: Extracto del Excel desarrollado para el CSV

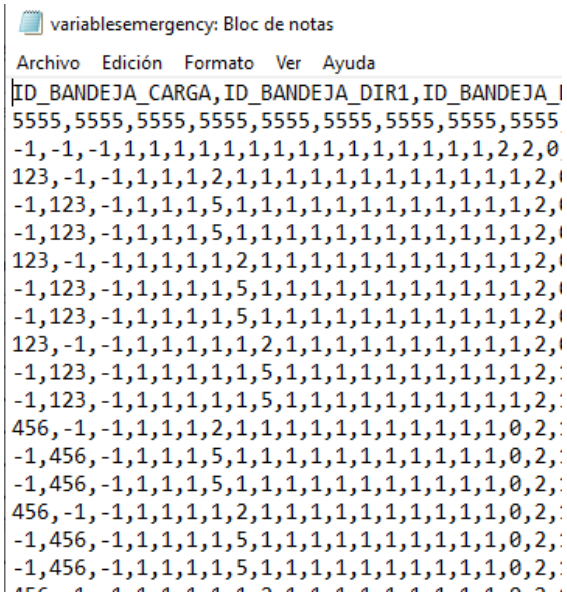
Cada fila del Excel representa un momento donde ha habido un cambio en la línea de producción, ya sea que la bandeja se ha movido o se está moviendo, o se ha completado un lote, etc. Por otra parte, cada columna representa una variable.

La primera fila está destinada a añadir los nombres de cada variable según aparecen en el trabajo final de grado *"Estudio de las etapas de automatización de un proceso industrial"* por Marc Torrecilla.

En la imagen se puede ver que la segunda fila contiene el valor "5555", el valor de por sí no tiene por qué ser ese ni la fila debería ser necesaria, pero se ha optado por dar formato a cada columna para evitar problemas con los números negativos de los ID's de las bandejas.

Adicionalmente, se han duplicado algunas filas con tal de simular una posible doble lectura de la línea de producción sin cambio alguno. Esto se quiere evitar puesto que cada fila se subirá a la base de datos creada en InfluxDB, de modo que no interesa estar constantemente llenando la base de datos si no se ha producido ni un solo cambio en el estado de la línea de producción, por lo que se ha optado por añadirlo y se buscará una manera de ignorar las filas duplicadas durante el desarrollo con Node-RED.

Una vez finalizado el documento en Excel se ha exportado como CSV.


variablesemergency: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
ID_BANDEJA_CARGA,ID_BANDEJA_DIR1,ID_BANDEJA_
5555,5555,5555,5555,5555,5555,5555,5555,5555
-1,-1,-1,1,1,1,1,1,1,1,1,1,1,1,1,1,2,2,0
123,-1,-1,1,1,1,2,1,1,1,1,1,1,1,1,1,1,1,2,
-1,123,-1,1,1,1,5,1,1,1,1,1,1,1,1,1,1,1,2,
-1,123,-1,1,1,1,5,1,1,1,1,1,1,1,1,1,1,1,2,
123,-1,-1,1,1,1,1,2,1,1,1,1,1,1,1,1,1,1,2,
-1,123,-1,1,1,1,1,5,1,1,1,1,1,1,1,1,1,1,2,
-1,123,-1,1,1,1,1,5,1,1,1,1,1,1,1,1,1,1,2,
123,-1,-1,1,1,1,1,1,2,1,1,1,1,1,1,1,1,1,2,
-1,123,-1,1,1,1,1,1,5,1,1,1,1,1,1,1,1,1,2,
-1,123,-1,1,1,1,1,1,5,1,1,1,1,1,1,1,1,1,2,
456,-1,-1,1,1,1,2,1,1,1,1,1,1,1,1,1,1,0,2,
-1,456,-1,1,1,1,5,1,1,1,1,1,1,1,1,1,1,0,2,
-1,456,-1,1,1,1,5,1,1,1,1,1,1,1,1,1,1,0,2,
456,-1,-1,1,1,1,1,2,1,1,1,1,1,1,1,1,1,0,2,
-1,456,-1,1,1,1,1,5,1,1,1,1,1,1,1,1,1,0,2,
-1,456,-1,1,1,1,1,5,1,1,1,1,1,1,1,1,1,0,2,
```

Ilustración 18: Recorte del archivo CSV

4.2 Diseño de la base de datos

Se ha creado la base de datos **"tfg_romero"** en InfluxDB. Dentro de esta base de datos se creará automáticamente un *measurement* (la parte donde se almacenan los datos y define su estructura) llamado **"pruebacsv"** en cuanto se intente acceder a la base de datos desde Node-RED. Aparte del *timestamp*, no se añadirá ninguna información desde aquí y su uso se limita a almacenar cada fila del archivo CSV y poder consultarlo cuando se requiera. Dicho de otra manera, se guardará en la base de datos cada cambio que se produzca en la línea de producción y el *timestamp* registrará en qué momento se ha producido cada cambio; la llegada de la bandeja a cada retenedor, el momento en el que sale del mismo, etc.

4.3 Diseño del flujo de Node-RED

4.3.1 Acondicionamiento simulación

Para poder implementar la simulación de la línea de producción, hay que tratar la información que llega a través del archivo CSV. Node-RED carga el archivo completo y la salida es la lectura de este archivo en su totalidad, siendo cada fila un Objeto con todas las columnas y un valor por objeto. Para simular la línea de producción se necesita que la lectura se haga por filas, así que para eso se ha usado el nodo **"Delay"**, que tiene la opción de limitar la salida a un mensaje con el intervalo de tiempo que se desee. En este caso se ha optado por 3 segundos, ya que será un tiempo similar al de cada estado en la línea de producción.

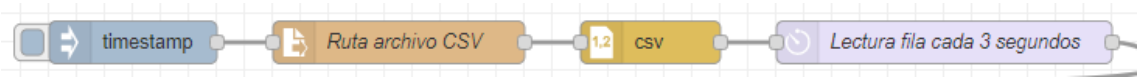


Ilustración 19: Lectura de archivo csv

De esta manera, se obtendrá cada 3 segundos un objeto con un estado nuevo y, por tanto, viendo la evolución de los estados, se puede simular el movimiento de las bandejas a lo largo de la línea de producción.

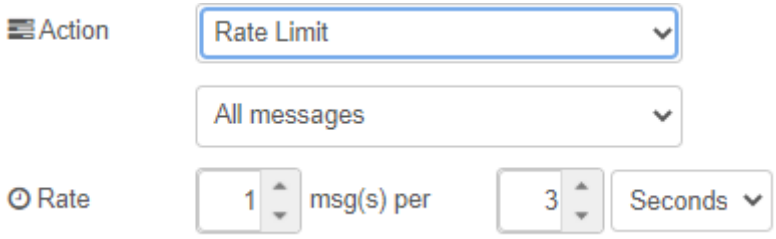


Ilustración 20:Configuración Nodo Delay

4.3.2 Cálculos del tiempo de ciclo

Para calcular el tiempo que se tarda en producir cada pieza, en cuanto una pieza llega al final de todo (estación de extracción), se hace una petición a la base de datos con tal de obtener el momento en el que llega al final, y también se solicita el ID de la bandeja.

Ilustración 21: Nodos para el cálculo del tiempo de ciclo

Una vez se obtienen los datos de llegada a la estación de extracción, se tienen que seleccionar en un nodo de función para poder realizar la trabajar con ellos. Al estar el ID dividido en 3, hay que seleccionar únicamente el que verdaderamente contiene el ID y no un "-1".

En la siguiente ilustración se puede observar el código escrito para cumplir dicho objetivo. Se crea la variable **"end"** y se guarda el tiempo proveniente de Influx. Después, se crean distintos condicionales cuya función principal es buscar el ID que es diferente a "-1", para después poder guardarlo en una variable de Node-RED y tratarla. Finalmente, se guarda el valor de las variables **"identificador"** y **"end"** en **"ID"** y **"endTime"**, respectivamente. Esto último se hace para poder trabajar con los valores almacenados en dichas variables en otros nodos del mismo flujo.

```
var end = msg.payload[0].time;   // current time
var identificador;
if (msg.payload[0].ID_BANDEJA_CARGA>1){
    identificador = msg.payload[0].ID_BANDEJA_CARGA;
}

if (msg.payload[0].ID_BANDEJA_DIR1>1){
    identificador = msg.payload[0].ID_BANDEJA_DIR1;
}

if (msg.payload[0].ID_BANDEJA_DIR2>1){
    identificador = msg.payload[0].ID_BANDEJA_DIR2;
}

flow.set("endTime", end);
flow.set("ID", identificador);
return msg;
```

Ilustración 22: Contenido del nodo función "TiempoFInal + ID"

Posteriormente se solicitan todas las salidas de la estación de carga de materias primas que ha habido durante el tiempo de ejecución. Se solicitan todas puesto que desde los nodos de Influx no se pueden hacer peticiones para un valor determinado que desconocemos. Por poner un ejemplo, si se sabe que la bandeja que acaba de llegar a la estación de extracción tiene **"ID:123",** bastaría con hacer la petición del momento en el que esa bandeja ha salido de la estación de carga, eso sí es posible y completamente viable, sin embargo, no sería un proceso automático porque tendríamos que solicitarlo específicamente para

cada bandeja que llegue al final y hacer una petición manual. Para automatizar el proceso, se ha optado por solicitar todos los inicios y luego en un nodo de función filtrar y seleccionar el que se desea.

Una vez filtrados los datos y se obtiene el momento en el que salió la bandeja de la estación de carga con el mismo ID que la que acaba de llegar a la estación de extracción, se obtiene el tiempo de ciclo haciendo la diferencia de ambos tiempos.

Esto en código se traduce en guardar el contenido de la variable **"ID"**, proveniente del nodo **"TiempoFinal + ID"**, en una variable nueva, en este caso **"identificador"**. Luego se realiza un bucle que mira todas las veces que se ha guardado la salida de la bandeja de la estación de carga, incluyendo condicionales que comprueban que el ID sea el mismo. En cuanto coinciden, se realiza la diferencia de tiempos en la línea 15.

```
1  var identificador=flow.get("ID");
2  var length=msg.payload.length;
3  var inicio;
4  for (var i=0;i<length;i++){
5      if(msg.payload[i].ID_BANDEJA_CARGA==identificador){
6          inicio = msg.payload[i].time;
7      }
8      if(msg.payload[i].ID_BANDEJA_DIR1==identificador){
9          inicio = msg.payload[i].time;
10     }
11     if(msg.payload[i].ID_BANDEJA_DIR2==identificador){
12         inicio = msg.payload[i].time;
13     }
14 }
15 msg.payload=flow.get("endTime")-inicio;
16 return msg;
17
```

Ilustración 23: Cálculo del tiempo de ciclo

Este valor se ha optado por añadirlo a la base de datos de Influx para poder ver el histórico de tiempos de ciclo y detectar anomalías. Del mismo modo, esta variable se ha conectado a un nodo de EcoStruxureAOA con tal de usarlo en la aplicación final. A esa variable se la ha nombrado **"ultimotiempo"** puesto que calculará cada tiempo de ciclo y mostrará únicamente el último valor en la aplicación de EcoStuxureAOA.

4.3.3 Cálculo de OEE

Para calcular el OEE, que como se ha mencionado anteriormente en este proyecto será equivalente al rendimiento, se ha realizado lo siguiente:

```
1  var tcicloideal=18000;
2  var piezas=msg.payload[0].LOTE_CARGADO;
3  var tiempoactual=(new Date())
4  var tinicio;
5  var runtime=tiempoactual-flow.get("iniciociclo");
6  var performance=(tcicloideal*piezas)/runtime;
7
8  msg.payload=performance;
9
10
11 return msg;
```

Ilustración 24: Cálculo de OEE incluido en nodo función

Se ha fijado el tiempo de ciclo ideal como 18000 ya que durante pruebas anteriores el tiempo estaba ligeramente por encima de ese valor. Se ha guardado el valor de piezas terminadas y se calcula el tiempo de ejecución haciendo la diferencia entre el tiempo actual y el tiempo en el que ha comenzado la producción. Este último valor se ha recogido en cuanto se ha iniciado el flujo de Node-RED. Seguidamente en la línea 6 se calcula el rendimiento con la fórmula vista anteriormente.

Ilustración 25: Nodos para el cálculo del OEE

Posteriormente se ha conectado a un nodo de EcoStuxureAOA para poder visualizarla en la aplicación final.

4.3.4 Cálculo del tiempo de emergencia

El tiempo de emergencia es otro dato que se pretende mostrar en la aplicación final y que también se guardará en la base de datos. Para hacer el cálculo de esta, se hace constantemente una solicitud a la base de datos buscando el momento en el que se inicia el estado de emergencia. En caso de que no haya, esta parte del flujo no hará absolutamente nada.

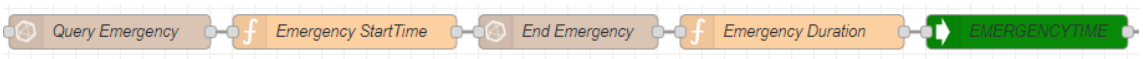


Ilustración 26: Flujo del cálculo del tiempo de emergencia

Una vez solicitado el momento en el que se inicia el estado de emergencia a Influx, se guarda ese valor en una variable interna de Node-RED y se solicita el momento en el que se activa el rearme. Una vez se obtiene dicho momento, se calcula la diferencia en un nodo de función y se envía a EcoStruxureAOA y a InfluxDB

```
1 var start = msg.payload[0].time;   // current time
2 flow.set("emergencystart", start);
3 return msg;
```

Ilustración 27: Almacenamiento del inicio del estado de emergencia

```
1 var end;
2 end=msg.payload[0].time;
3 msg.payload=end-flow.get("emergencystart");
4 return msg;
5
```

Ilustración 28: Cálculo del tiempo de emergencia

En la línea 2 de la ilustración 27 se guarda el inicio del estado de emergencia en la variable **"emergencystart"**, convirtiéndolo en una variable accesible fuera de ese nodo. Dicha variable vuelve a aparecer en el nodo **"Emergency Duration",** ilustración 28, línea 3, para realizar la diferencia de tiempo con el momento en el que finaliza dicho estado.

4.3.5 Otros datos

- Aparte de las variables mencionadas anteriormente, desde el flujo de Node-RED se envían las variables **"CONTADOR_BANDEJAS_VACÍAS", "PRODUCTOS_POR_CARGAR", "LOTE_CARGADO"** y **"EMERGENCY"** directamente desde el archivo CSV sin pasar por la base de datos de InfluxDB. Para ello simplemente se utilizan nodos del tipo EcoStruxure AOA.

Ilustración 29: Variables a visualizar en EcoStruxureAOA

- Las variables que han requerido algún cálculo y se han subido a InfluxDB como, por ejemplo, la variable **"ultimotiempo"** no se pueden subir directamente y se han tenido que acondicionar. Para ello, se ha optado por convertirlas en objetos de JavaScript. De esta manera lo que hacemos es añadir una columna más a la base de datos de InfluxDB con el nombre de la variable. En la línea 1 de la ilustración 31 se puede observar cómo se ha realizado la creación del objeto, guardando luego el valor en él y así poder subirlo sin problemas a InfluxDB.

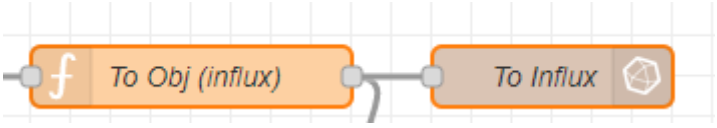


Ilustración 30: Nodo de conversión a objeto de JavaScript y subida a InfluxDB

```
1  var obj={ultimotiempo:""};
2  obj.ultimotiempo=msg.payload;
3  msg.payload=obj;
4  return msg;
5
```

Ilustración 31: Contenido nodo "To Obj (Influx)" de la variable ultimpotiempo

- Como se dijo anteriormente, nos interesa optimizar la información que se sube a la base de datos con tal de no sobrecargar la red y llenar la base de datos con información inútil o duplicada y por ello, durante la creación del CSV se pusieron filas con tal de simular esto. Durante la lectura del archivo CSV en Node-RED es inevitable hacer la lectura de esas filas puesto que Node-RED lee todo el archivo, aunque luego pongamos un nodo delay. Sin embargo, existe un nodo llamado rbe (del inglés, *report by exception*) que bloquea el flujo si el valor es exactamente el mismo que el anterior y gracias a este nodo se ha puesto solución a dicho problema.

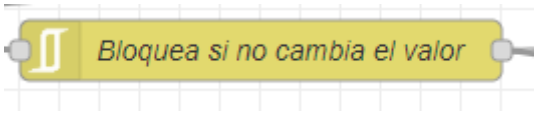


Ilustración 32: Nodo RBE

4.4 Diseño de la aplicación en Builder

4.4.1 Creación de escenas

Una vez finalizado el programa en Node-RED, toca hacer la aplicación en el builder. Para ello, se ha creado un área llamada **"Final"** que contendría todo el laboratorio de Schneider Electric, pero en este caso sólo se controlará la línea de producción, por lo que se ha creado un área para dicha línea. La intención inicial era hacer uso de varias fotos, sin embargo, debido a lo similares que son las distintas partes de la línea, se ha optado por hacer uso de un código bidi y así evitar posibles problemas.

Etiqueta	Tipo	Número de etiqueta
Celula	Reconocimiento de etiqueta	501

Ilustración 33: Escena con reconocimiento de la etiqueta 501

Para ello, se ha tomado una fotografía de la etiqueta disponible y al adjuntarla a la escena, se ha tenido que delimitar la etiqueta en la fotografía para que posteriormente el reconocimiento se realice correctamente.

Ilustración 34: Delimitación etiqueta

4.4.2 Creación de variables

Una vez creada las escenas, se han declarado las variables que se obtendrán desde Node-RED. Las variables **"CONTADOR_BANDEJAS_VACIAS", "EMERGENCY"** y **"EMERGENCYTIME"** se mantendrán ocultas en un inicio.

VARIABLES

Buscar: Escribir para buscar...

Nombre	Alias	Tipo	
PRODUCTOS_POR_CARGAR	Productos por cargar	Internal	☐
ultimotiempo	ultimotiempo	Internal	☐
OEE	OEE	Internal	☐
EMERGENCY	Emergency	Internal	☐
EMERGENCYTIME	Time	Internal	☐
LOTE_CARGADO	LOTE_CARGADO	Internal	☐
CONTADOR_BANDEJAS_VACIAS	CONTADOR_BANDEJAS_VACIAS	Internal	☐

Ilustración 35: Lista de variables a mostrar

Posteriormente, se ha ubicado cada variable en diferentes zonas. Pese a que el funcionamiento con etiquetas es distinto ya que lo único que busca reconocer es la etiqueta en sí y no su entorno, se han colocado las variables donde irían si se hubiera usado reconocimiento de imagen. Cada **"{0}"** será sustituido por el valor de la variable en cada momento.

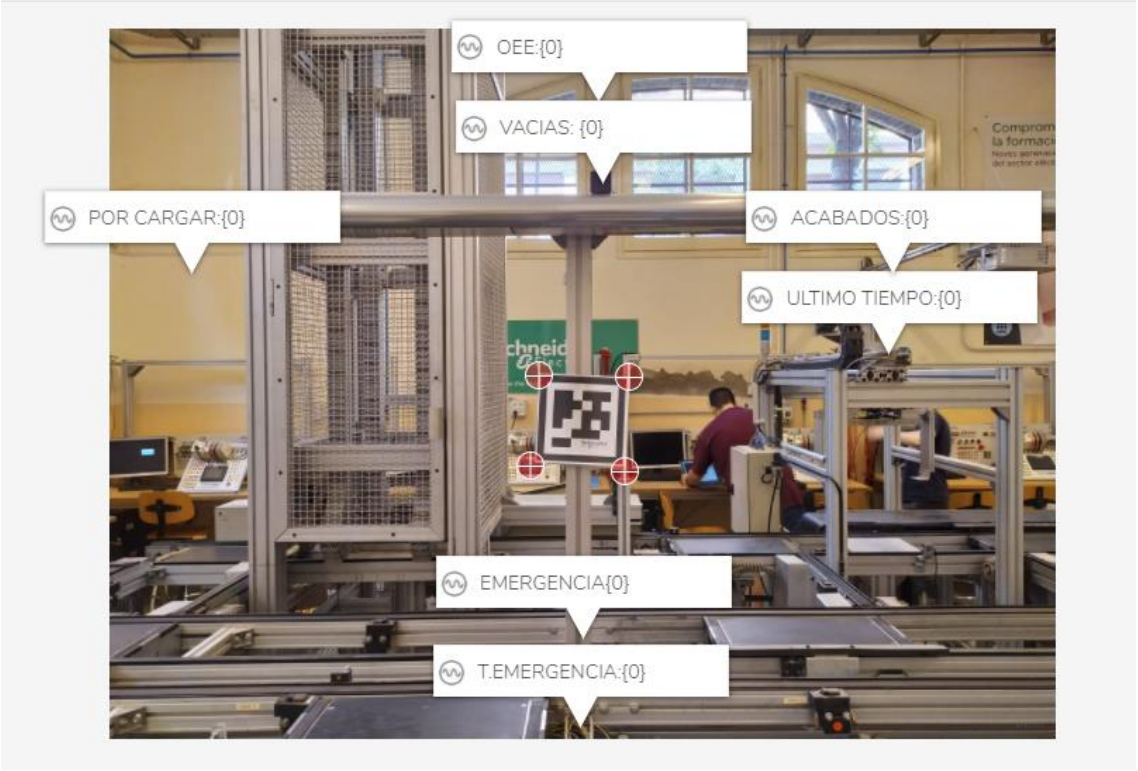


Ilustración 36: Ubicación de Variables

4.4.3 Desencadenadores

DESENCADENADORES

Nombre	Descripción	Objeto
ALERT	Emergency started	EMERGENCY
EMPTY	Oculta la variable si es 0	CONTADOR_BANDEJAS_VACIAS
Ultimo Tiempo	Muestra valor si es diferente a 0	ultimotiempo

Ilustración 37: Lista de desencadenadores

Los desencadenadores pueden llevar a cabo distintas acciones sobre las variables declaradas cuando se da una condición específica. Estas acciones pueden ser, por ejemplo, hacer que la variable parpadee, cambiarle el color, mostrarla u ocultarla, etc. Son acciones puramente estéticas y no afectan al valor de la variable. Principalmente se han usado para ocultar las variables que no requieren mostrar información de manera constante. En el caso de las variables **"EMERGENCY"** y **"EMERGENCYTIME"** se han ocultado y se mostrarán únicamente cuando nos encontremos en estado de emergencia, y también se han añadido una serie de acciones con la intención llamar la atención a modo de alerta.

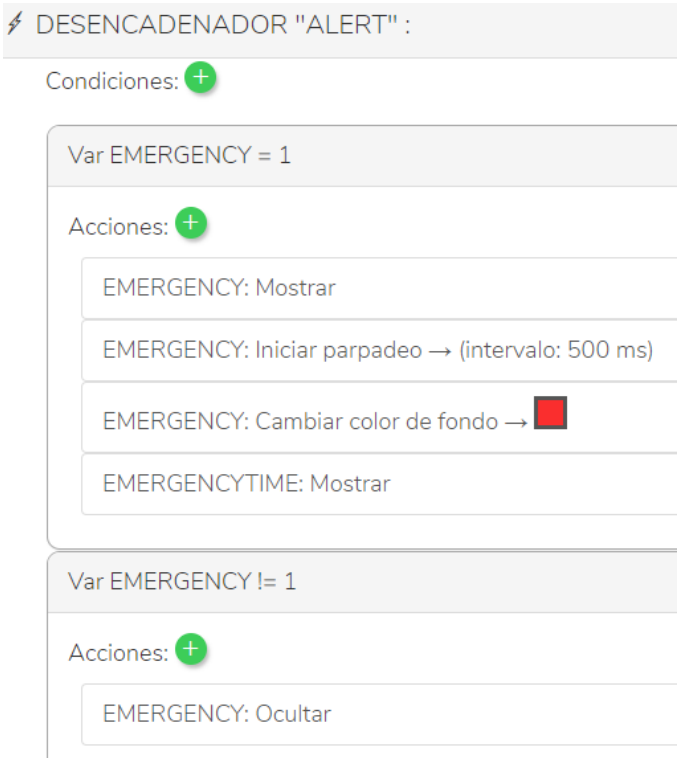


Ilustración 38: Acciones para las variables EMERGENCY y EMERGENCYTIME

Capítulo 5

5.1 Validación Node-RED

5.1.1 Validación lectura CSV

Objetivo: Asegurar una lectura correcta del archivo CSV para trabajar adecuadamente con las variables deseadas.

Proceso de validación: Se ha verificado cambiando la conexión a un nodo de depuración en Node-RED, lo que permite verificar la salida sin influir en el resto del flujo.

Resultado esperado: Se esperaba que la lectura del archivo se hiciera por filas y omitiera las filas que se repiten.

Resultado obtenido: Los resultados han sido satisfactorios.

5.1.2 Validación cálculos

Objetivo: Asegurar que los cálculos realizados corresponden a lo esperado y no se están mezclando valores que puedan alterar el resultado previsto.

Proceso de validación: Con la configuración realizada sobre el archivo CSV, la lectura de cada fila se realiza cada 3 segundos. De modo que se ha contabilizado de forma manual el tiempo que hay entre estados para asegurar el cálculo de las variables **"EMERGENCYTIME"** y **"ultimotiempo".** Para el **OEE,** se ha calculado de manera manual tomando como valores fijos el número de piezas y el tiempo de ciclo. El runtime se ha estimado en función del momento en el que se realiza cada cálculo del OEE en el flujo.

Resultado esperado: Se esperaba que la variable **"EMERGENCYTIME"** sea aproximadamente de 24000 milisegundos, que la variable **"ultimotiempo"** tenga valores alrededor de 18000 milisegundos salvo cuando aparece el estado de emergencia de por medio, es ese caso debería ser de 48000 milisegundos aproximadamente. Se esperaba también alcanzar un **OEE** elevado, cercano al 90%.

Resultado obtenido: Las variables **"EMERGENCYTIME"** y **"ultimotiempo"** cumplen perfectamente lo esperado. El **OEE** ha alcanzado el 90% en algunas pruebas, pero se mantiene cerca del 60% y baja al 40% cuando se da el estado de emergencia.

Corrección realizada: Se ha añadido un bloque tipo Influx para que haga el cálculo del OEE únicamente cuando llegue una bandeja al final del ciclo. Del mismo modo, se ha agregado un node tipo "block" que bloquee el cálculo si no ha cambiado ningún valor.

5.2 Validación EcoStruxureAOA

5.2.1 Validación visualización

Objetivo: Comprobar que las variables programadas en el builder se visualicen correctamente en la EcoStruxureAOA App.

Proceso de validación: Se ha usado la EcoStruxureAOA App enfocando al código bidi utilizado en el proyecto utilizando distintos dispositivos.

Resultado esperado: Se esperaba que las variables **"PRODUCTOS_POR_CARGAR"**, **"ultimotiempo", "OEE", "EMERGENCY", "EMERGENCYTIME"**, **LOTE_CARGADO"** y **"CONTADOR_BANDEJAS_VACIAS"** se visualizaran correctamente al enfocar la etiqueta.

Resultado obtenido: El reconocimiento de la etiqueta ha funcionado correctamente y las variables se han mostrado en pantalla.

5.2.2 Validación desencadenadores

Objetivo: Comprobar que las variables que hacen uso de desencadenadores se activen cuando se den lugar los acontecimientos programados.

Proceso de validación: Se ha ejecutado todo el proceso.

Resultado esperado: Se esperaba que la variable **"ultimotiempo"** se mantuviera oculta hasta que no se finalice como mínimo un producto, también que la variable **"CONTADOR_BANDEJAS_VACIAS"** se mantenga oculta cuando no haya bandejas vacías. Por otra parte, se esperaba que la variable **"EMERGENCY"** parpadease en rojo cada 0,5 segundos y permaneciera oculta mientras no se diera el estado de emergencia. También, la variable **"EMERGENCYTIME"** aparecería cuando se dé el estado de emergencia y permanecería visible una vez terminado.

Resultado obtenido: Las acciones aplicadas a **"CONTADOR_BANDEJAS_VACIAS", "EMERGENCY"** y **"EMERGENCYTIME"** funcionan correctamente. La acción aplicada a **"ultimotiempo"** funciona una única vez y luego queda visible permanentemente. Esto se debe a que a pesar de cargar de nuevo el proyecto, la variable **"ultimotiempo"** mantiene el último valor que ha tenido.

Corrección realizada: Reiniciar el runtime.

5.2.3 Validación valores variables

Objetivo: Asegurar que las variables muestran valores reales sin alteraciones.

Proceso de validación: Se ha ejecutado todo el proceso.

Resultado esperado: Se esperaba que las variables que no han pasado por cálculos en Node-RED, es decir, las variables **"CONTADOR_BANDEJAS_VACIAS"**, **"PRODUCTOS_POR_CARGAR"**, **"LOTE_CARGADO"** y **"EMERGENCY"** mantuvieran los valores sin alteraciones y se reiniciaran cada vez que se reinicia Node-RED.

Resultado obtenido: Todas las variables cumplen satisfactoriamente, pero sólo se reinician si se vuelven a inicializar a 0.

Corrección realizada: Reiniciar el runtime.

5.3 Validaciones adicionales

Una vez realizadas las anteriores validaciones, quedará una última parte por validar. Todas las validaciones han sido con el archivo CSV, que permite simular todo el proceso con un control de las señales y por tanto predecir los outputs que se obtendrán, esto no lo permite la máquina real. La línea de producción debe estar en funcionamiento de manera continua y las señales serían desconocidas. Esta última validación no ha podido realizarse por problemas técnicos en el laboratorio, sin embargo, las validaciones anteriores ya adelantan que, de realizarse la validación con la línea de producción en marcha, esta sería exitosa.

Para ello bastaría con cambiar en Node-Red todos los nodos relacionados con la simulación y colocar en su lugar nodos Modbus, acondicionándolos para tener el mismo principio de funcionamiento que la simulación. Puesto que las variables serían las mismas no habría que hacer cambios en el builder ni en Influx, así que bastaría con ejecutar el runtime y la EcoStruxureAOA App en la tableta. Para asegurarse de que los desencadenadores funcionan, habría que entrar en estado de emergencia de manera manual pero por lo demás, con tener la línea de producción en marcha sería suficiente.

Capítulo 6

6.1 Conclusiones

Este proyecto ha consistido en la implementación y desarrollo de una aplicación de realidad aumentada en un proceso industrial. Para llevar a cabo esto se ha trabajado con el software de Schneider Electric, EcoStruxure Augmented Operator Advisor que ha sido el software que ha permitido hacer uso de la realidad aumentada, no obstante, también se debe mencionar que se ha tenido que usar Node-RED e InfluxDB para poder integrarla al área de trabajo.

La validación dividida en dos partes, una simulada con datos controlados y que nos permiten predecir qué salida deberíamos obtener, y otra con la línea de producción funcionando de manera continua, se ha podido completar hasta la parte controlada. La parte con la máquina operativa no ha podido llevarse a cabo debido a problemas técnicos ajenos al proyecto.

No obstante, la validación simulada se aproxima lo suficiente como para determinar que se podría llevar a cabo una validación exitosa con la máquina operativa haciendo uso del programa desarrollado en este proyecto.

Cabe destacar también que la simulación ha sido más que suficiente para aprender sobre protocolos de comunicación y visualizar el impacto que puede tener la realidad aumentada a nivel industrial en el futuro.

6.2 Posible continuación

Como posible continuación, se podrían expandir las posibilidades de la aplicación de realidad aumentada, puesto que en este proyecto se ha centrado en la supervisión, es decir a la visualización de variables, pero se podrían crear guías de procedimiento, acceso a documentos y una serie de opciones que generen más interacción entre el usuario y la app.

Bibliografía

La siguiente bibliografía se ha elaborado siguiendo el estilo de citación del IEEE

[1] M. Torrecilla Matencio, "ESTUDIO DE LAS ETAPAS DE AUTOMATIZACIÓN DE UN PROCESO INDUSTRIAL," 2020.

[2] N. Munuera Cano, "ESTUDIO DE LAS ETAPAS DE AUTOMATIZACIÓN DE UN PROCESO INDUSTRIAL: COMUNICACIONES Y OPERACIÓN," 2020.

[3] P. Fraga-Lamas, T. M. Fernández-Caramés, Ó. Blanco-Novoa, and M. A. Vilar-Montesinos, "A Review on Industrial Augmented Reality Systems for the Industry 4.0 Shipyard," *IEEE Access*, vol. 6. Institute of Electrical and Electronics Engineers Inc., pp. 13358–13375, Feb. 20, 2018, doi: 10.1109/ACCESS.2018.2808326.

[4] "¿Qué es una API?" https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces (accessed Mar. 11, 2021).

[5] "Add a Timestamp to Function node - General - Node-RED Forum." https://discourse.nodered.org/t/add-a-timestamp-to-function-node/22610 (accessed May 04, 2021).

[6] "Augmented Reality – The Past, The Present and The Future | Interaction Design Foundation (IxDF)." https://www.interaction-design.org/literature/article/augmented-reality-the-past-the-present-and-the-future (accessed Jun. 14, 2021).

[7] "Calculate OEE – Definitions, Formulas, and Examples | OEE." https://www.oee.com/calculating-oee.html (accessed Jun. 13, 2021).

[8] "Creating your first flow : Node-RED." https://nodered.org/docs/tutorials/first-flow (accessed Feb. 23, 2021).

[9] "Data exploration using InfluxQL | InfluxDB OSS 1.7 Documentation." https://docs.influxdata.com/influxdb/v1.7/query_language/data_exploration/#the-into-clause (accessed May 24, 2021).

[10] "El Protocolo IP." https://neo.lcc.uma.es/evirtual/cdd/tutorial/red/ip.html (accessed Jun. 17, 2021).

[11] "How to append timestamp into msg.payload - General - Node-RED Forum." https://discourse.nodered.org/t/how-to-append-timestamp-into-msg-payload/6981 (accessed May 04, 2021).

[12] "Influxdb project: 08 query postman - YouTube." https://www.youtube.com/watch?v=oO3AdI9xgOM&list=PLtdl5xR_8RHTWaUVNA-RqARcT8b3jMs7V&index=8 (accessed Mar. 11, 2021).

[13] "InfluxDB Tutorial - Complete Guide to getting started with InfluxDB - YouTube." https://www.youtube.com/watch?v=Vq4cDIdz_M8&t=79s (accessed Feb. 25, 2021).

[14] "infoPLC_net_DLL_AOA_NodeRed.pdf - Google Drive." https://drive.google.com/file/d/1JCtpZgVFCSC3Yr6occHSjNIKH6Bm_rJB/view (accessed Feb. 25, 2021).

[15] "JavaScript primer – Node RED Programming Guide." http://noderedguide.com/javascript-primer/ (accessed May 24, 2021).

[16] "Las 4 revoluciones industriales y las tecnologías que usaron." https://www.solex.biz/noticias/revoluciones-industriales-tecnologias-impulsaron/ (accessed Jun. 14, 2021).

[17] "List of tz database time zones - Wikipedia." https://en.wikipedia.org/wiki/List_of_tz_database_time_zones (accessed May 04, 2021).

[18] "Moment.js | Docs." https://momentjs.com/docs/#/displaying/format/ (accessed Feb. 28, 2021).

[19] "Modelo TCP/IP - Wikipedia, la enciclopedia libre." https://es.wikipedia.org/wiki/Modelo_TCP/IP (accessed Jun. 17, 2021).

[20] "MODBUS TCP | Tutoría Virtual de A. Javier Barragán Piña." http://uhu.es/antonio.barragan/content/modbus-tcp (accessed Jun. 17, 2021).

[21] "Modbus - Wikipedia, la enciclopedia libre." https://es.wikipedia.org/wiki/Modbus (accessed Jun. 17, 2021).

[22] "Node RED - Nodos Modbus (Modbus Read) 1/3 | Schneider Electric - YouTube." https://www.youtube.com/watch?v=Ae97Fg-TMO8 (accessed Mar. 11, 2021).

[23] "node-red-contrib-increment (node) - Node-RED." https://flows.nodered.org/node/node-red-contrib-increment (accessed May 04, 2021).

[24] "node-red-contrib-influxdb (node) - Node-RED." https://flows.nodered.org/node/node-red-contrib-influxdb (accessed Mar. 02, 2021).

[25] "node-red-contrib-moment (node) - Node-RED." https://flows.nodered.org/node/node-red-contrib-moment (accessed Mar. 02, 2021).

[26] "node-red-contrib-simpletime (node) - Node-RED." https://flows.nodered.org/node/node-red-contrib-simpletime (accessed May 04, 2021).

[27] "OEE (Overall Equipment Effectiveness) - A Practical Guide." https://limblecmms.com/blog/oee-overall-equipment-effectiveness/ (accessed Jun. 13, 2021).

[28] "OEE Measures Improvements in Productivity | Lean Production." https://www.leanproduction.com/oee.html (accessed Jun. 13, 2021).

[29] "Operations Management - Google Books." https://books.google.es/books?hl=en&lr=&id=OmOfDwAAQBAJ&oi=fnd&pg=PA31&dq=oee+range&ots=Wj3nNWWuY-&sig=5vnJoIbsf3UtjGEUHAM7ehgx43w&redir_esc=y#v=onepage&q=oee range&f=false (accessed Jun. 13, 2021).

[30] "Performance indicator - Wikipedia." https://en.wikipedia.org/wiki/Performance_indicator#System_operations (accessed Jun. 13, 2021).

[31] "Query InfluxDB with Flux | InfluxDB OSS 2.0 Documentation." https://docs.influxdata.com/influxdb/v2.0/query-data/get-started/query-influxdb/ (accessed Mar. 02, 2021).

[32] "Realidad mixta - Wikipedia, la enciclopedia libre." https://es.wikipedia.org/wiki/Realidad_mixta (accessed Jun. 14, 2021).

[33] "Running Node-RED locally : Node-RED." https://nodered.org/docs/getting-started/local (accessed Feb. 21, 2021).

[34] "Sidebar: Information : Node-RED." https://nodered.org/docs/user-guide/editor/sidebar/info (accessed Feb. 25, 2021).

[35] "Una de las bases más asentadas de la Industria 4.0." https://www.automaticaeinstrumentacion.com/texto-diario/mostrar/2829069/bases-asentadas-industria-40 (accessed Jun. 13, 2021).

[36] "Uso de Node-Red con Augmented Operator Advisor - infoPLC." https://www.infoplc.net/descargas/176-schneider-electric/miscelánea/3120-node-red-augmented-operator-advisor-schneider-electric (accessed Jun. 01, 2021).

[37] "Write data with the InfluxDB API | InfluxDB OSS 1.8 Documentation." https://docs.influxdata.com/influxdb/v1.8/guides/write_data/ (accessed Mar. 10, 2021).