EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Ciudad Guzmán

**INSTITUTO TECNOLÓGICO DE CD. GUZMÁN**

TITULACIÓN INTEGRAL
TESIS

**TUNA SHIELDS S DE RL DE CV**

TEMA:
# "IMPLEMENTACIÓN DE UN PLC SIEMENS LOGO EN PLATAFORMA DE APLICACIONES IoT"

QUE PARA OBTENER EL TÍTULO DE:
**INGENIERO ELECTRÓNICO**

PRESENTA:
**CARLOS ÁNGEL PATRICIO MARTÍNEZ**

ASESOR INTERNO:
**MIP. JOSE DE JESUS GARCIA CORTES**

ASESOR EXTERNO:
**ING. ALEJANDRO ELIZALDE TORRES**

CD. GUZMÁN JALISCO, MÉXICO, AGOSTO DE 2020

"2020, Año de Leona Vicario, Benemérita Madre de la Patria"

Cd. Guzmán, Jal. a 30 /noviembre/2020

ASUNTO: Liberación de proyecto para titulación integral

**C. M.I.E. FAVIO REY LUA MADRIGAL**
**JEFE DEL DEPTO. DE DIV. DE EST. PROF.**
**PRESENTE**

Por este medio informa que ha sido liberado el siguiente proyecto para titulación integral:

| | |
|---|---|
| Nombre del estudiante y/o egresado: | CARLOS ANGEL PATRICIO MARTINEZ |
| Carrera: | INGENIERIA ELECTRONICA |
| No. de Control: | 15290370 |
| Nombre del Proyecto | IMPLEMENTACION DE UN PLC SIEMENS LOGO EN PLATAFORMA DE APLICACIONES IOT |
| Producto: | TESIS |

Agradezco de antemano su valioso apoyo en esta importante actividad para la formación profesional de nuestros egresados.

**A T E N T A M E N T E**

**M.E.H. MARCO ANTONIO SOSA LÓPEZ**
**JEFE DEL DEPTO. ELÉCTRICA Y ELECTRÓNICA**

| | | |
|---|---|---|
| M.I.P. JOSE DE JESUS GARCIA CORTES | M.E.H. GUSTAVO CHAVEZ CRENDAIN | M.I.E. JOSE MARIA HERNADEZ OCHOA<br>José Ma Hernández O. |
| Nombre y firma del asesor | Nombre y firma del revisor | Nombre y firma del revisor |

C.p. Archivo
MASL/adc

# AGRADECIMIENTOS

Escribo estas líneas para expresar mis más profundo y sincero agradecimiento a todas aquellas personas que con su ayuda han colaborado en la realización del presente trabajo.

**A mis padres,** Lorena y Carlos, por brindarme su apoyo en todo momento a lo largo de este recorrido, por sus consejos, sus valores, por la motivación constante que me han permitido ser la persona que soy, por los ejemplos de perseverancia y constancia que me han mostrado siempre, por el valor mostrado para salir adelante, pero más que nada, por su amor.

**A mis hermanos,** Emmanuel, Moisés, y Juan, por ser parte importante de mi vida y representar la unidad familiar.

**A mi asesor interno,** Jesús García Cortes, por orientarme durante el proceso de desarrollo del proyecto.

**A mis asesores externos,** Alejandro y Arturo, por su apoyo técnico en todo el proceso de la elaboración del proyecto, así mismo, por el conocimiento compartido.

A todos ellos, muchas gracias.

# IMPLEMENTACIÓN DE UN PLC SIEMENS LOGO EN PLATAFORMA DE APLICACIONES IoT

Carlos Ángel Patricio Martínez
Instituto Tecnológico de Ciudad Guzmán
Av. Tecnológico No 100 Cd. Guzmán, Jalisco, C.P. 49100, México
carang_05@live.com

**RESUMEN**

El crecimiento de la industria 4.0 está impulsando el desarrollo del internet de las cosas industrial, permitiendo que muchos dispositivos dentro de la industria sean conectados al internet para el intercambio de datos.
Este documento esta enfocado en la evaluación de los protocolos que el PLC LOGO! ofrece (Modbus TCP/IP y S7)para transmisión de datos con el Gateway (Raspberry Pi 3 integrada con Node-RED), y posteriormente con los demás dispositivos conectados al mismo, tales como dashboard local, dashboard remoto y sensores. Para esto se consideraron dos entornos de prueba: uno basado en dispositivos externos conectados al PLC LOGO! y la utilización de Modbus TCP/IP para el envío de datos al gateway, así como la conexión de una celda de carga al gateway a través de una placa desarrollada de conversión de protocolo cableado a wifi. El segundo entorno está basado en la simulación de un sistema de dispensando, el cual fue desarrollado en el software Fatory I/O. Para este segundo entorno, de utilizo el protocolo S7 para la comunicación entre el PLC LOGO! y el gateway, además el mismo protocolo fue utilizado para la comunicación entre el PLC LOGO! y Factory I/O.
El desarrollo de los dashboard locales para ambos entornos se realizó en Node-RED. Y para el desarrollo de los dashboards remotos fue utilizada una plataforma comercial (Ubidots), con la cual a través del gateway se estableció la comunicación con el protocolo MQTT, utilizando el bróker que la misma plataforma proporciono.

**Palabras Clave: PLC, IoT, Gateway, Node-RED, S7, Modbus, MQTT.**

## INDICE GENERAL

## INDICE FIGURAS

## INDICE TABLAS

# CAPÍTULO 1
# INTRODUCCIÓN

# Capítulo 1 – Introducción

## 1.1 Objetivo general

Se tiene como objetivo general la implementación del PLC LOGO! a una plataforma remota y una local, a través de una red IoT. Además de probar los dos protocolos que este PLC LOGO! trabaja, con el fin de encontrar la mejor opción para la implementación con el gateway (raspberry pi), y el posterior envío de datos a las dos plataforma, o demás dispositivos que se encuentren en la red. También se tiene como objetivo general el diseño e implementación de una placa electrónica que convierta protocolos cableados a wifi, logrando una conexión de estos sensores con el gateway para usar sus datos en diferentes dispositivos conectados a este.

## 1.2 Objetivos específicos

- Programar PLC LOGO para intercambio de datos con el gateway a través de los dos diferentes protocolos que este contiene, como lo son Modbus TCP/IP y S7.
- Programar gateway (raspberry pi) para intercambio de datos con el PLC LOGO! a través de los dos diferentes protocolos que PLC LOGO! contiene, como lo son Modbus TCP/IP y S7.
- Implementación de un dashboard local en Node-RED, para cada caso de prueba.
- Implementación de un dashboard remoto en Ubidots, para cada caso de prueba.
- Transmisión de datos con el dashboard remoto con MQTT.
- Desarrollo de placa electrónica para conversión de protocolos cableados a wifi, y el envió de los datos al gateway (raspberry)
- Elaborar informe técnico del proyecto.
- Elaborar manual de implementación de un PLC LOGO! con una raspberry para envío de datos a Ubidots.
- Elaborar guía de configuración de Raspberry.
- Elaborar guía de configuración de Ubidots y dashboards.
- Elaborar manual de uso de software de LOGO.

## 1.3 Definición del problema

En la actualidad existen diferentes gateways para poder realizar el envío de datos de un PLC simple a la nube, o bien, ya existen PLCs que tienen integrado esto, sin embargo, el costo de estos es muy alto, además de que están limitados a la

implementación de otros posibles cambios, como el posible intercambio de datos con sensores a través de wifi y demás protocolos.

### 1.4 Hipótesis

El protocolo Modbus TCP/IP funcionara de manera adecuada puesto que es un protocolo antiguo y universal. Dando la posibilidad de implementación no solamente con este PLC, sino con la mayoría de ellos independiente de la marca.
El protocolo S7 será más fácil de trabar, y más rápido, puesto que este protocolo es de la marca de PLC, mostrando una posible mejora con respecto a los otros.
La implementación de la raspberry como un Gateway nos dará la oportunidad de mayor amplitud de trabajo con diferentes protocolos y dispositivos, así mismo, el coste sería muy bajo.
La placa electrónica podrá convertir cualquier protocolo cableado a wifi, y los datos llegaran al Gateway de manera oportuna, permitiendo la distribución de estos datos por los dispositivos necesarios.
El intercambio de datos entre los dashboards tanto local como remoto se dará en tiempo real, dando la posibilidad a futuros análisis de datos, o la posible implementación de toma de decisiones automáticas.

### 1.5 Justificación

Se creará la prueba de los dos protocolos que el PLC LOGO! tiene, para mandar datos al Gateway, para de esta manera saber cuál de los dos funciona de una manera estable y amplia. Además de la implementación de dashboard tanto local como remoto, para la prueba de envío de datos con el protocolo MQTT desde el Gateway. Se creará una placa electrónica, que convierta diferentes protocolos cableados a wifi, para la implementación de diferentes sensores en una red IoT, con el envío de datos al Gateway.

### 1.6 Limitaciones

A continuación, se mencionan las causas que pudieron ocasionar que el proyecto no se terminara a tiempo:

- Componentes electrónicos defectuosos o de mala calidad.
- Que los dispositivos electrónicos requeridos para la investigación no se recibieran a tiempo.
- Que los softwares a utilizar no sean libres. Y tener que recurrir a opciones poco amigables.

## 1.7 Delimitaciones

Se utilizará un PLC LOGO! de SIEMENS que cuenta con conexión Ethernet para la comunicación local con una RASPBERRY PI y visualización de datos de manera loca, a través de un dashboard local implementado en Node-RED y un dashboard remoto con Ubidots.

## 1.8 Metas

Las metas que se plantearon al planear el proyecto fueron las siguientes:

- Trasmitir datos del PLC LOGO! al gateway (raspberry) con los dos protocolos que el mismo tiene.
- Crear para PLC LOGO! dashboard local dentro de Node-RED.
- Crear para el PLC LOGO! dashboard remoto en Ubidots.
- Placa para convertir protocolos cableados a wifi.
- Implementación de un sensor cableado al gateway a través de la placa desarrollada.
- Implementar sistemas de prueba para cada uno de los protocolos.

## 1.9 Metodología

I Especificaciones del proyecto
II Investigación documental
III Diseño
IV Construcción
V Pruebas
VI Elaboración de informes técnicos
VII Elaboración del reporte final del proyecto

## 1.11 Lista de actividades

**I Especificaciones del proyecto**

1. Definir el diagrama a bloques del sistema para prueba uno.
2. Definir el diagrama a bloques del sistema para prueba dos.
3. Definir qué dispositivos se utilizarán dentro del sistema de acuerdo con las características propuestas para la prueba uno.
4. Definir qué dispositivos se utilizarán dentro del sistema de acuerdo con las características propuestas para la prueba dos.
5. Definir contenido de informe técnico.

**II Investigación documental**

6. Investigar en internet.
7. Investigar en libros.
8. Investigar en revistas técnicas.
9. Investigar en portales de investigación.
10. Consultar experto.

**III Diseño**

11. Diseño de código a bloques en software de PLC LOGO! para primer prueba.
12. Diseño de placa electrónica para conversión de protocolos.
13. Diseño de código con nodos en Node-RED para primer prueba.
14. Diseño de código a bloques en software de PLC LOGO! para segunda prueba.
15. Diseño de código con nodos en Node-RED para segunda prueba.
16. Diseño de entorno industrial dentro del simulador.

**IV Construcción**

17. Realizar conexión de la primera prueba para el protocolo Modbus TCP/IP.
18. Ensamble de placa para conversión de protocolos.
19. Conexión de placa de conversión de protocolos a Node-RED, así como la conexión con el sensor.
20. Configuración del dashboard local en Node-RED para primer prueba.
21. Configuración del dashboard remoto en Ubidots para primer prueba.
22. Realizar conexión de la segunda prueba para el protocolo S7.
23. Configuración del dashboard local en Node-RED para segunda prueba.
24. Configuración del dashboard remoto en Ubidots para segunda prueba.
25. Conexión por protocolo S7 del PLC LOGO! y el simulador industrial.

**V Pruebas**

26. Prueba para el protocolo Modbus TP/IP entre la conexión PLC-Gateway.
27. Prueba para el protocolo Modbus S7 entre la conexión PLC-Gateway.
28. Prueba de placa de conversión de protocolos.
29. Prueba de envió de datos de placa electrónica al Gateway
30. Prueba de publicación y suscripción al bróker de Ubidots.
31. Prueba de envío y recepción de datos en dashboard local.
32. Prueba de envío y recepción de datos en dashboard remoto.

**VI Elaboración de informes técnicos**

33. Guía de configuración/instalación de Node-RED en raspberry pi.
34. Guía de creación de entornos gráficos en simulador Factory I/O.
35. Guía de configuración de Ubidots para la implementación de dashboards.

36. Guía de programación para Node-RED y el uso de las librerías para los protocolos Modbus TCP/IP y S7.
37. Guía de programación en software LOGO! Soft Comfort para la programación con bloques de funciones FUP.

**VII Elaboración del reporte final del proyecto**

38. Elaboración del reporte final del proyecto.

# CAPÍTULO 2
# MARCO TEÓRICO

# Capítulo 2 Marco Teórico

## 2.1 Internet de las cosas (IoT)

El Internet de las Cosas o Internet of Things (IoT) es un concepto más allá de la interconexión de miles de dispositivos a Internet, es el potencial al que se puede llegar no solo en el área de la domótica, sino en la industria y la seguridad. El IoT es una tendencia que trae consigo nuevas ideas y posibilidades que van desde los hogares inteligentes hasta las fábricas inteligentes [1].

El IoT es el concepto de la interconexión entre distintos dispositivos a través del Internet, para ello es necesario hardware especializado que permita realizar la comunicación y control de estos dispositivos o bien mediante un servidor que recabe los datos que éstos recolecten. Mediante el uso de etiquetas de radio frecuencia se podrá tener todo tipo de dispositivos identificados lo que permitiría saber su ubicación, el tiempo que éstos lleven en ese lugar, si están siendo utilizados o si es necesario desecharlos [1].

### 2.1.1 ¿Qué es IoT?

El IoT es una infraestructura de red global, que enlaza los objetos físicos y objetos virtuales (bases de datos, registros, imágenes, audios, etc.) a través de la explotación de la captura de datos y capacidades de comunicación. Esta infraestructura incluye desarrollos existentes, la evolución del Internet y las redes. Ofrecer la identificación especifica de objetos, sensores y la capacidad de conexión como base para el desarrollo de servicios cooperativos y aplicaciones independientes como se muestra en la Figura 2.1.

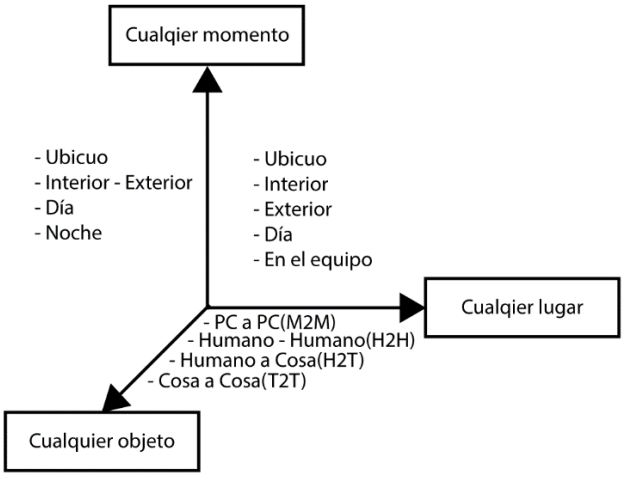



**Figura 2.1** Dimensiones del IoT [1].

El IoT se caracteriza por un alto grado de captura de datos de forma autónoma, transferencia de eventos, conectividad de red e interoperabilidad, cuya función principal es transportar información de un punto a otro de forma rápida, fiable y segura [1].

El concepto de IoT fue propuesto por Kevin Ashton en 1999 en el centro AUTO-ID del MIT, donde se realizaban investigaciones en el campo de identificación de radiofrecuencia en red y tecnología de sensores. El IoT estima codificar hasta 100 billones de objetos y seguir el movimiento de estos; se calcula que todo ser humano está rodeado por lo menos por 1000 objetos. Los analistas estiman que para el 2020 habrá aproximadamente 50 mil millones de dispositivos que estarán conectados a Internet como se muestra en la Figura 2.2. Los usuarios, los objetos y los servicios en la nube se conectan a través de Internet para permitir nuevas aplicaciones de uso [3].

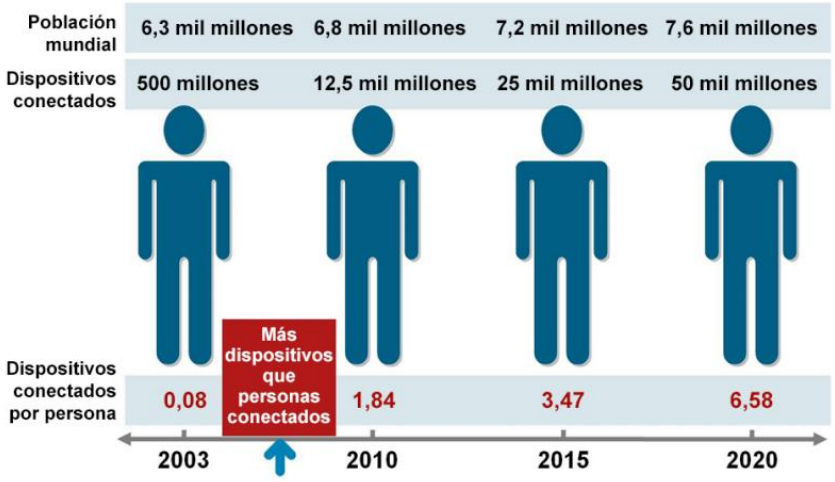



**Figura 2.2** Nacimiento del IoT [3].

Actualmente mediante la aplicación del protocolo Ipv6 se puede identificar por medio de un código a cualquier tipo de objeto, lo que permitirá el desarrollo de aplicaciones, como la apertura de tiendas completamente controladas y automatizadas con IoT, mediante las identificaciones de los productos por medio de etiquetas RFID se puede tener registro del número total de productos en bodega y en exhibición, así como la identificación de productos caducados. Al estar conectados todos los productos en la tienda al IoT, es posible tener la ubicación de todos los productos y el cobro mediante terminales automáticas sin la necesidad de la intervención de un vendedor [2].

### 2.1.2 Descripción técnica del IoT

Un objeto físico puede estar representado en el mundo de la información por uno o varios objetos virtuales (registros de sensores, listas de contenido, ubicación, etc.), pero el objeto virtual también puede existir sin tener asociado ningún objeto físico [3].

Un dispositivo es una pieza de equipo con capacidades obligatorias de comunicación y capacidades opcionales de detección, accionamiento, adquisición, almacenamiento y procesamiento de datos. Los dispositivos recaban diversos tipos de información y la suministran a las redes de la información y la comunicación para su posterior procesamiento, algunos dispositivos también ejecutan operaciones en función de la información recibida de las redes de la información y la comunicación. Los dispositivos se comunican con otros dispositivos: a través de la red de comunicaciones por medio de una pasarela (caso a), por medio de una red sin pasarela (caso b) o directamente, esto es, sin utilizar la red de comunicación (caso c) como se muestra en la Figura 2.3 [3].

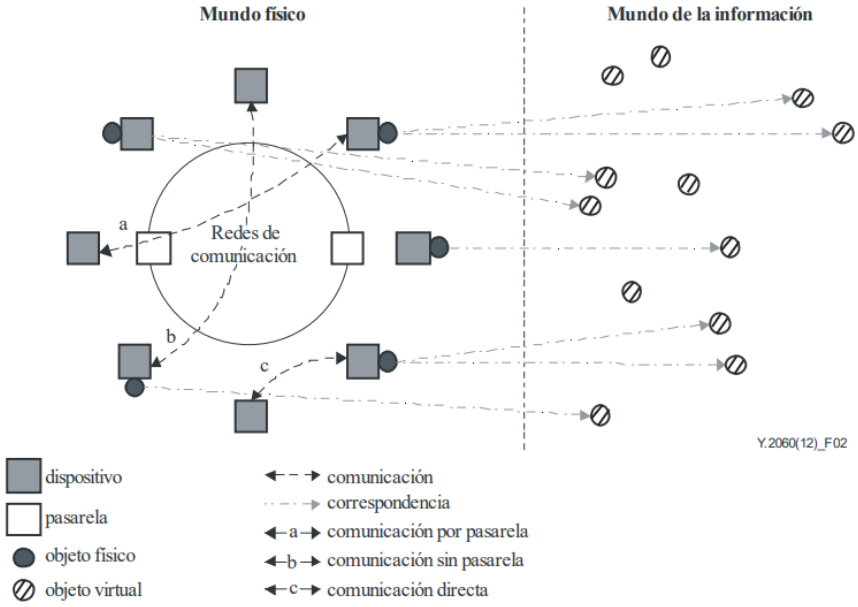



**Figura 2.3** Representación técnica del IoT [2].

### 2.1.3 Características fundamentales y requisitos del IoT

Las características fundamentales del IoT son las siguientes [3]:

- **Interconectividad:** En el contexto de IoT, todo puede estar interconectado con la infraestructura mundial de la información y la comunicación.

- **Servicios relacionados con objetos:** El IoT es capaz de suministrar servicios relacionados con los objetos dentro de las restricciones de los objetos, como protección de la privacidad y coherencia semántica entre los objetos físicos y sus correspondientes objetos virtuales. Para ofrecer servicios relacionados con objetos dentro de las restricciones de objetos, las tecnologías en el mundo físico y en el de la información evolucionarán.

- **Heterogeneidad:** Los dispositivos en IoT son heterogéneos dado que se basan en diferentes plataformas de hardware y redes. Pueden interactuar con otros dispositivos o plataformas de servicios a través de redes diferentes.

- **Cambios dinámicos:** El estado de los dispositivos varía dinámicamente, por ejemplo, del modo reposo al activo, conectado y/o desconectado, así como el contexto del dispositivo, como la ubicación y velocidad. Además, el número de dispositivos también puede cambiar dinámicamente.

- **Escala de gran dimensión:** El número de dispositivos que ha de gestionarse y que se comunican entre sí puede ser incluso un orden de magnitud mayor que el número de dispositivos conectados actualmente a Internet. El porcentaje de comunicación que requerirán estos dispositivos será muchísimo mayor que el de la comunicación entre humanos. Será incluso más esencial la gestión de los datos generados y su interpretación para fines de aplicación, aspectos que guardan relación con la semántica de datos y la manipulación eficiente de datos.

El IoT como todo sistema en desarrollo presenta cualidades especifica como [3]:

- **Conectividad basada en la identificación:** El IoT necesita que se establezca conectividad entre un objeto y el identificador del objeto. Para ello puede ser necesario además procesar de manera unificada identificadores posiblemente heterogéneos.

- **Compatibilidad:** Es indispensable garantizar la compatibilidad entre sistemas heterogéneos y distribuidos para el suministro y consumo de diversos tipos de información y servicios.

- **Redes automáticas:** Es necesario que las funciones de control de red del IoT soporte las redes automáticas (en particular técnicas y/o mecanismos de autogestión, autoconfiguración, auto restablecimiento, auto optimización y autoprotección), a fin de adaptarse a los diferentes dominios de aplicación, diferentes contextos de comunicación y diferentes tipos de dispositivos.

- **Configuración automática de servicios:** Es preciso poder configurar los servicios a partir de los datos de los objetos adquiridos, comunicados y procesados automáticamente con seguimiento a las reglas configuradas por los operadores o personalizadas por los clientes. Los servicios automáticos pueden depender de las técnicas de fusión y minería de datos automáticas.

- **Capacidades basadas en la ubicación:** El IoT debe dar soporte a capacidades basadas en la ubicación. Las comunicaciones y servicios relacionados con objetos dependientes de la información sobre la ubicación de los objetos y/o los usuarios. Es necesario detectar y rastrear automáticamente la información sobre la ubicación. Las comunicaciones y servicios basados en la ubicación pueden estar limitados por leyes y reglamentos y deben cumplir los requisitos de seguridad.

- **Seguridad:** En IoT, todo 'objeto' está conectado a Internet, lo que conlleva considerables amenazas de seguridad, en ámbitos tales como la confidencialidad, autenticidad e integridad de datos y servicios. Un ejemplo esencial de los requisitos de seguridad es la necesidad de integrar diferentes técnicas y políticas de seguridad para la diversidad de dispositivos y redes de usuario en IoT.

- **Protección de la privacidad**: Muchos objetos tienen sus propietarios y usuarios. Los datos detectados de los objetos pueden contener información privada acerca de sus propietarios o usuarios. El IoT tiene que dar soporte a la protección de la privacidad durante la transmisión, combinación, almacenamiento, minería y procesamiento de datos. La protección de la privacidad no debe ser un obstáculo a la autentificación de las fuentes de datos.

- **Autoconfiguración (plug y play):** El IoT debe soportar la autoconfiguración que permite generar, componer o adquirir sobre la marcha configuraciones semánticas para la integración paulatina y la cooperación de los objetos interconectados con aplicaciones, y atender las necesidades de las aplicaciones.

- **Capacidad de administración:** El IoT debe dar soporte a la capacidad de administración para garantizar el funcionamiento normal de la red. Las aplicaciones IoT suelen trabajar automáticamente sin la intervención humana, pero el proceso global de funcionamiento debe poder gestionar las partes pertinentes.

### 2.1.4 Tecnologías del IoT

Hay algunas tecnologías mejoradas utilizadas en el desarrollo y despliegue de productos de Internet de las cosas que ha tenido un gran éxito. Se trata de la tecnología RFID, la comunicación de redes de sensores inalámbricos, la computación en nube, Big-data, middleware y el software de aplicaciones IoT. La elección de los dominios antes mencionados se basa en el estado moderno de los asuntos de la idoneidad de IoT [4].

- **RFID:** La identificación de frecuencia por radio (RFID) inicia el reconocimiento automatizado así como la captura de datos con ondas de radio y una etiqueta. Basado en la habilidad RFID la frase "Internet de las cosas" fue propuesta para asociar la identificación absoluta de dispositivos/objetos conectados interoperables.

- **Wireless Sensor Network Transportation:** WSN es uno de los componentes clave a favor del éxito del esquema de IoT. WSN es una compilación de nodos de sensores, que comprende un nodo de principio apropiado llamado fregadero o cabeza de cúmulo. Escalabilidad, robustez, fiabilidad y parámetros de eficiencia energética son necesarios al diseñar un sistema WSN impulsado por energía.

- **Cloud Computing:** La computación en nube funciona como tecnología o representación del procesamiento de datos distribuidos bajo demanda, con la ayuda de Internet unos pocos recursos de información escalables y capacidades proporcionadas como servicio a usuarios/dispositivos externos. Los dispositivos conectados con IoT crean una enorme cantidad de datos compartidos a través de Internet. Muchos de los aparatos IoT requerían un espacio de almacenamiento de datos gigantesco y velocidad de procesamiento para tomar decisiones inmediatas, visualización de datos, gestión de dispositivos, análisis de datos y búsqueda de banda ancha para transmitir datos de audio y vídeo. La computación en la nube funciona en el backend supremo para manejar datos creados por dispositivos IoT y procesarlos para procedimientos IoT.

- **Big Data:** Big Data tiene enormes responsabilidades en IoT, donde introduce un soporte de almacenamiento de datos para almacenar e integrar con datos de IoT preestablecidos y sin forma. El diseño de Hadoop en Big Data junto con varias bases de datos complementarias crea un área de almacenamiento distribuida de archivos para almacenar, así como gestionar recursos variados, tipos de datos compuestos por sensores y lectores de RFID. El trabajo de big-data en IoT es increíble, aunque las aplicaciones más

observables serán en análisis, seguridad de datos y espacios de almacenamiento de datos.

- **Middleware:** Internet de las cosas (IoT) era el elemento colectivo de la Internet potencial con el cómputo omnipresente. La IoT exige conexiones por medio de actuadores mixtos, sensores en bruto, agregadores y aparatos diferentes con conciencia de contexto, conservan la protección y la confidencialidad. Consecutivamente para satisfacer las demandas anteriores, IoT debe necesitar la plataforma de software definitivo como middleware. Middleware, una hoja de software introduce entre los aparatos de software que formulan en nombre de los desarrolladores de software para lograr una transmisión más fácil y entrada/ salida. Simplemente middleware hizo la conexión entre diferentes programas complejos que no fueron construidos originalmente para conectar con otros.

### 2.1.5 Modelo de referencia del IoT

El modelo de referencia del IoT consta de cuatro capas que se relacionan con las capacidades de gestión y de seguridad, ver Figura 2.4.

Las capas involucradas son las siguientes [3]:

- **Capa de aplicación:**
  La capa de aplicación contiene las aplicaciones IoT.

- **Capa de soporte de servicios y aplicaciones:**
  La capa de soporte de servicios y aplicaciones consiste en los dos siguientes grupos de capacidades:
  - **Capacidades de soporte genéricas:**
    Son capacidades comunes que pueden utilizar diferentes aplicaciones IoT, tales como procesamiento o almacenamiento de datos.
  - **Capacidades de soporte especificas:**
    Son capacidades para atender las necesidades particulares de diversas aplicaciones. En realidad, pueden consistir en diversos grupos de capacidades precisas que ofrecen distintas funciones de apoyo a las diferentes aplicaciones IoT.

- **Capa de red:**
  Consiste en dos tipos de capacidades:
  - **Capacidades de red:**
    Las capacidades de red ofrecen funciones de control de la conectividad en red, tales como funciones de control de acceso y de recursos de transporte, gestión de la movilidad y autentificación, autorización y administración.

- **Capacidades de transporte:**
  Las capacidades de transporte se centran en suministrar conectividad para el transporte de información y datos específicos de servicios y aplicaciones IoT, así como el transporte de información de control y gestión relacionada con IoT.

- **Capa de dispositivo:**
  Se puede efectuar una clasificación lógica de las capacidades de la capa de dispositivo, en dos tipos:
  - **Capacidades del dispositivo:**
    - **Interacción directa con la red de comunicaciones:**
      Los dispositivos pueden recabar y cargar información directamente (es decir, sin recurrir a capacidades de pasarela) en la red de comunicación y pueden recibir información directamente (por ejemplo, instrucciones) de la red de comunicación.
    - **Interacción indirecta con la red de comunicación:**
      Los dispositivos pueden recabar y cargar información indirecta en la red de comunicación, es decir, mediante capacidades de pasarela. Además, los dispositivos pueden recibir información indirecta (por ejemplo, instrucciones) de la red de comunicación.
    - **Redes ad-hoc:**
      Los dispositivos pueden construir redes de manera ad-hoc en algunas circunstancias cuando sea necesario para aumentar la capacidad evolutiva y la velocidad de despliegue.
    - **Modo reposo activo:**
      Las capacidades de dispositivo deben disponer de mecanismos para pasar a los modos "reposo" y "activo" a fin de ahorrar energía.

  - **Capacidades de pasarela:**
    Las capacidades de pasarela son, entre otras:
    - Soporte de interfaces múltiples: En la capa de dispositivo, las capacidades de pasarela soportan dispositivos conectados mediante diferentes tipos de tecnologías alámbricas e inalámbricas, tales como el bus de red de control de zona (CAN), Zigbee, Bluetooth o Wix|-Fi. En la capa de red, las capacidades de pasarela pueden comunicarse a través de diversas tecnologías, tales como la red telefónica pública conmutada (PSTN), las redes de segunda o tercera generación (2G o 3G), las redes LTE

(evolución a largo plazo), Ethernet o las líneas digitales de abonado (DSL).

- **Capacidades de gestión:**
  IoT abarca las clases tradicionales de fallos, configuración, administración, rendimiento y seguridad (FCAPS), es decir, la gestión de fallos, de la configuración, de la administración, del rendimiento y de la seguridad.

- **Capacidades de seguridad:**
  Hay dos tipos de capacidades de seguridad: genéricas y específicas. Las capacidades de seguridad genéricas son independientes de la aplicación. Las capacidades de seguridad específicas están estrechamente relacionadas con los requisitos específicos de la aplicación, por ejemplo, los requisitos de seguridad para el pago con un dispositivo móvil.

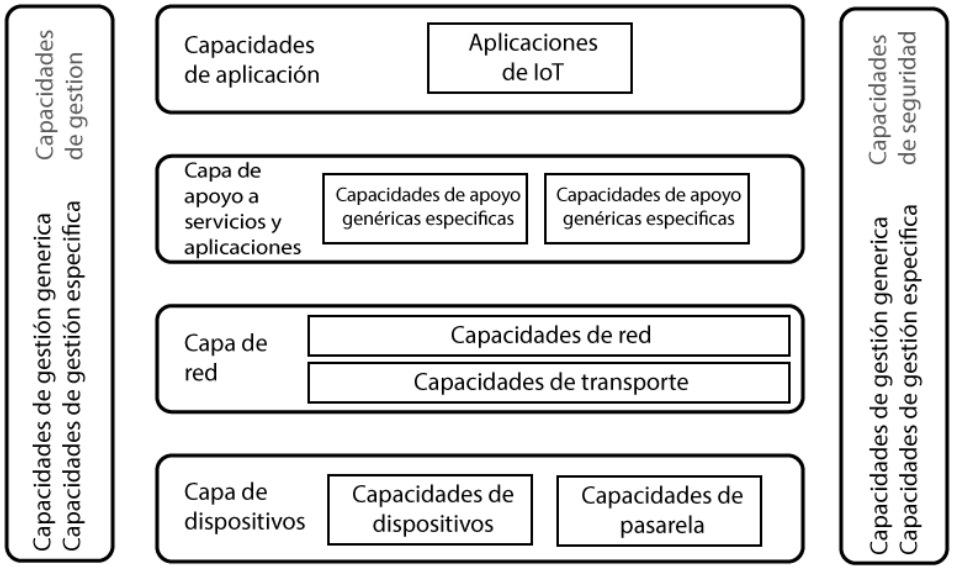



**Figura 2.4** Modelo de referencia del IoT [1].

### 2.1.6 Aplicaciones del IoT

En la actualidad el IoT ha tomado muchas formas, permitiendo aplicaciones como:

- **Automoción** [5]**:**
  Aplicaciones en la industria automovilística que incluyen el uso de "Smart things" para supervisar e informar de todo, desde la presión de los neumáticos hasta la proximidad de otros vehículos. La tecnología RFID se utiliza para optimizar la producción de vehículos, mejorar la logística, aumentar el control de calidad y mejorar el servicio al cliente. Los usos de dispositivos inalámbricos ayudan a identificar mejor donde están las cosas, lo que hace posible acelerar los procesos de montaje y localizar vehículos o

componentes en una fracción del tiempo. Por lo tanto, comunicaciones de corto alcance (DSRC) tendrán la posibilidad de tener tasas de bits más altas y así reducir la posibilidad de interferencia con otros equipos, generando con esto que las comunicaciones de vehículo-a-vehículo (V2V) y Vehículo-a-Infraestructura (V2I) avancen significativamente en los sistemas inteligentes de transporte, como, por ejemplo; servicios de seguridad de vehículos.

- **Telecomunicaciones** [5]**:**
  El IoT creará la posibilidad de fusión de diferentes tecnologías de telecomunicaciones para crear nuevos servicios. Un ejemplo es el uso de GSM, la tecnología NFC Comunicación de campo), Bluetooth de bajo consumo, WLAN, redes hop múltiples, GPS y sensores de redes, junto con la tecnología de la tarjeta SIM. NFC permite la comunicación entre los objetos de una manera sencilla y segura sólo por tener ellos uno del otro. Por consiguiente, el teléfono móvil puede ser utilizado como un lector de NFC y transmitir los datos leídos a un servidor central. Cuando se utiliza en un teléfono móvil, la tarjeta SIM tiene un papel importante como el tiempo de almacenamiento para los datos de la NFC y las credenciales de autenticación (como números de los billetes, los recuentos de tarjetas de crédito, información de ID, etc). Se podría decir que en un futuro no muy lejano las fronteras de la IoT y las redes clásicas de telecomunicaciones convergerán haciendo un cruce de dominios y así apoyar la creación de servicios de información, y al mismo tiempo garantizar la protección.

- **Edificios inteligentes** [5]**:**
  Tecnologías de automatización de edificios o casas han sido desarrolladas para oficinas y departamentos de lujo. Muchas investigaciones actuales se están desarrollando para lograr tener una "casa inteligente", pero como esta tecnología es cara se están desarrollando aplicaciones acordes a las necesidades humanas y a su factor económico, como, por ejemplo: tener contadores inteligentes es cada vez más popular para medir el consumo de energía y permitir transmitir esta información al proveedor del servicio eléctrico. El uso de automatización de dispositivos del hogar con las tecnologías de comunicación inalámbrica (por ejemplo, Zigbee, 6LoWPAN, etc) permitirían que todas las cosas del edificio puedan tener una comunicación bidireccional con los demás. Por ejemplo, el monitor de la pantalla táctil en la nevera se puede utilizar para cambiar la configuración del termostato o un teléfono móvil que entra en el edificio puede activar preferencias de configuración en base al perfil de la persona.

- **Salud y Tecnología médica** [5]**:**
  El IoT tendrá muchas aplicaciones en el sector de la salud, con la posibilidad de utilizar el teléfono móvil con capacidades de RFID con sensores como una

plataforma para la monitorización de parámetros médicos y la administración de fármacos. Las grandes ventajas de estos sistemas serían la prevención, monitoreo de accidentes y el diagnóstico rápido de algún paciente. La combinación de sensores, RFID, NFC (Near Field Communication), Bluetooth, Zigbee, 6LoWPAN, Wirelesshart, ISA100, Wifi permitirá mejorar significativamente la medición y los métodos de seguimiento de las funciones vitales (temperatura, presión arterial, frecuencia cardíaca, niveles de colesterol, glucosa en la sangre, etc.). Adicionalmente, se espera que la tecnología de sensores esté disponible a un costo mucho menor con el apoyo incorporado para la conectividad y monitorización remota.

- **Vida Independiente** [5]**:**
  Aplicaciones y servicios de IoT tendrán un gran impacto en cada una de las vidas independientes ofreciendo apoyo a una población que envejece, detectando las actividades de la vida diaria mediante sensores de monitorear interacciones sociales, además de esto, se tendrá un seguimiento de enfermedades crónicas mediante sensores portátiles de signos vitales y sensores localizados en el cuerpo. Con la aparición de los algoritmos de detección de patrones y aprendizaje automático, las "cosas" en el entorno de un paciente serían capaces de realizar el cuidado de este, debido a que las cosas pueden aprender las rutinas regulares y generar alertas o enviar notificaciones en situaciones de anomalía.

- **Farmacéutica** [5]**:**
  En IoT, particularmente en el dominio de los productos farmacéuticos, al igual que cualquier tecnología a ser utilizada directamente sobre el ser humano, una adecuada aplicación es de suma importancia para prevenir el comprometer la salud de los pacientes. La posibilidad de monitoreo mediante sensores del estatus de productos farmacéuticos que deben pasar a través de una cadena de suministros hasta llegar a los pacientes tiene múltiples beneficios como:

  - Identificación de productos que requieren condiciones especiales de almacenamiento o trasporte.
  - Seguimiento de medicamentos originales que cumplen estándares de calidad, de forma que se posibilite la detección de productos falsificados y mantener la cadena de suministro libre de estafadores.
  - Así también, las etiquetas inteligentes sobre medicamentos pueden beneficiar a los pacientes informando la dosis, fecha de caducidad, y asegurando la autenticidad del medicamento. A través de etiquetas inteligentes conjuntamente con un gabinete de medicina también inteligente, que lee la información transmitida por las etiquetas de los medicamentos, los pacientes pueden recibir un recordatorio para

tomar su medicamento a intervalos apropiados, en las dosis prescritas y el cumplimiento del paciente puede ser monitoreado.

- **Industria** [6]**:**
  La aplicación de la IoT a la industria manufacturera se llama IIoT (o Internet Industrial o Industria 4.0). El IIoT revolucionará la fabricación al permitir la adquisición y accesibilidad de cantidades mucho mayores de datos, a velocidades mucho mayores y de manera mucho más eficiente que antes.
  El IIoT puede mejorar enormemente la conectividad, la eficiencia, la escalabilidad, el ahorro de tiempo y el ahorro de costos para las organizaciones industriales. Los líderes empresariales pueden utilizar los datos IIoT para obtener una visión completa y precisa de cómo su empresa está haciendo, lo que les ayudará a tomar mejores decisiones.

### 2.1.7 Seguridad en IoT

La seguridad incluye todas las técnicas que tienen por objeto proteger, ya sea proteger la información y estar pendientes contra los ataques maliciosos. IoT es el mayor arreglo con una amalgama de pequeños dispositivos bien vestidos dotados de datos personales sensibles si cualquier fuga de esta información causa muchos problemas por lo que la seguridad de IoT toma lugar anticipado antes que su diseño y despliegue. En general, los sistemas de seguridad con respecto a IoT consisten en ofrecer los siguientes servicios [4]:

- **Privacidad:**
  Garantiza que los datos de identidad del cliente no sean rastreables o identificables de sus acciones y comportamiento anteriores en el sistema.
- **Autentificación:**
  Los dispositivos deben autenticarse entre sí antes de transferir mensajes o información. La autenticación asegura que la fuente de datos es segura.
- **Confidencialidad:**
  Garantiza que la información puede ser alteradas por fuentes legitimadas solamente; y que esté preparado para fuentes no oficiales.
- **Integridad:**
  Garantiza que los datos no son cambiados por ilegitimidad o por entidades de terceros ya se intencional o accidentalmente.

## 2.2 Controlador Lógico Programable

Los controladores lógicos programables (ver Figura 2.5) son ahora la tecnología de control de procesos industriales más utilizada. Un controlador lógico programable (PLC) es un ordenador de grado industrial que es capaz de ser programado para realizar funciones de control. El controlador programable ha eliminado gran parte

del cableado duro asociado a los circuitos de control de relés convencionales. Otros beneficios incluyen respuesta rápida, fácil programación e instalación, alta velocidad de control, compatibilidad de red, resolución de problemas y conveniencia de pruebas, y alta fiabilidad [7].


**Figura 2.5** Controladores lógicos Programables [7].

Inicialmente el PLC fue utilizado para reemplazar la lógica de relé, pero su creciente gama de funciones le permite estar inmerso en muchas aplicaciones complejas. Debido a que la estructura de un PLC se basa en los mismos principios que los empleados en la arquitectura informática, es capaz no sólo de realizar tareas de conmutación de relés, sino también de realizar otras aplicaciones como comparación y procesamiento de señales analógicas [7].

### 2.2.1 ¿Qué es un PLC?

Un controlador lógico programable (PLC) es un ordenador industrial que recibe entradas de dispositivos de entrada y luego evalúa esas entradas en relación con la lógica del programa almacenado y genera salidas para controlar dispositivos de salida periférica. Los módulos de E/S y un diagrama de bloque funcional PLC se muestran en la Figura 2.6. Se toman muestras de los dispositivos de entrada y se actualiza en tiempo real la tabla de imágenes de entrada del PLC correspondiente. El programa del usuario, cargado en la memoria del PLC a través del dispositivo de programación, resuelve la lógica de aplicación predefinida y actualiza la tabla lógica interna de salida. Los dispositivos de salida se manejan en tiempo real de acuerdo con los valores actualizados de la tabla de salida [8].


**Figura 2.6** Arquitectura de las entradas-salidas de un PLC [8].

Las interfaces estándar para dispositivos de entrada y salida están disponibles para la automatización de cualquier aplicación existente o nueva. Estas interfaces son factibles con todo tipo de PLCs independientemente del proveedor seleccionado. Los sensores y actuadores permiten al PLC interactuar con todo tipo de dispositivos analógicos y ON/OFF mediante el uso de módulos digitales de E/S, convertidores analógicos a digitales (A/D), convertidores digitales a análogos (D/A) y circuitos de aislamiento adecuados. Aparte de la entrada de alimentación y las interfaces de E/S, todas las señales dentro del PLC son digitales y de baja tensión [8].

PLCs se utilizan en muchas industrias y máquinas. A diferencia de los ordenadores comunes, el PLC está diseñado para múltiples disposiciones de entrada y salida, rangos de temperatura extendidos, inmunidad al ruido eléctrico, y resistencia a la vibración y el impacto. Los programas para controlar el funcionamiento de la máquina se almacenan normalmente en una memoria de respaldo de batería o no volátil. Un PLC es un ejemplo de sistema cableado en tiempo real porque los resultados de salida deben ser producidos en respuesta a las condiciones de entrada dentro de un tiempo limitado; de lo contrario, el funcionamiento no deseado resultará. La mayoría de los componentes electromecánicos necesarios para los sistemas de relés de control cableados están completamente eliminados, resultando en una gran reducción en espacio, consumo de energía y requisitos de mantenimiento [8].

### 2.2.2 Un poco de historia del PLC

Antes de la introducción de los PLC, todas las tareas de producción y control de procesos se realizaban mediante sistemas basados en relés. Las industrias no tuvieron más opción que hacer frente a este sistema de control inflexible y costoso. La modernización de un sistema de producción de control de máquinas basado en relés supone un cambio en todo el sistema de producción, que es muy caro y lleva mucho tiempo. En la década de 1960, General Motors (GM) publicó una propuesta para la sustitución de máquinas basadas en relés. La historia del PLC comenzó con un Ingeniero llamado Richard E. Morley, quien también fue uno de los fundadores de Modicon Corporation en respuesta a la propuesta de GM. Morley finalmente creó el primer PLC en 1977 y lo vendió a Gould Electronics, que lo presentó a General Motors. Este primer PLC se mantiene ahora de forma segura en la sede de la empresa [8].

La historia de PLCs se muestra en categorías de tiempo a partir de los primeros sistemas introducidos desde 1968 hasta 1971. Esto es seguido por un lapso de 6 años etiquetados como la primera generación de PLC. La segunda generación comenzó en 1979 y cubrió un período de 7 años, terminando en 1986. Este período mostró un mayor número de vendedores en su mayoría de las empresas existentes de EE.UU. además de empresas alemanas y japonesas. La tercera generación

comenzó en 1987 y duró 10 años, seguido de un período de crecimiento y avance continuos en hardware y software, lo que llevó a un amplio despliegue de PLC en la mayoría de las actividades de automatización de fabricación y control de procesos [8].

### 2.2.3 Arquitectura del PLC

Un PLC típico consiste principalmente en una unidad de procesamiento central (CPU), fuente de alimentación, memoria, módulo de comunicación y circuitos apropiados para manejar datos de E/S. El PLC puede ser visto como una caja inteligente con cientos o miles de relés separados, contadores, temporizadores y lugares de almacenamiento de datos. Estos contadores, temporizadores y relés no existen físicamente, pero son entidades internas simuladas por software. Los relés internos se simulan a través de ubicaciones de bits en los registros de memoria. La Figura 2.7 muestra un diagrama de bloques simplificado de una arquitectura de hardware de PLC genérica típica [8].

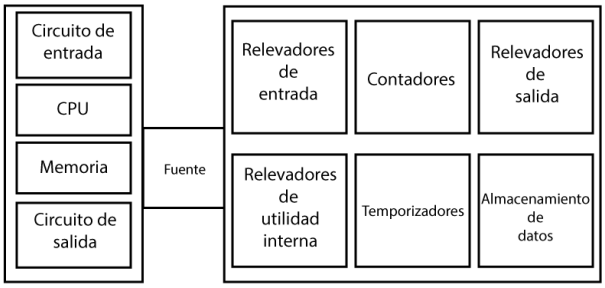



**Figura 2.7** Arquitectura general del PLC [8]

Los módulos de entrada de PLC se implementan típicamente usando transistores y estos sí existen físicamente. Reciben señales de interruptores externos y sensores a través de contactos. Estos módulos permiten al PLC interactuar con el exterior y obtener una sensación en tiempo real del estado del proceso. Los módulos de salida se implementan típicamente usando transistores y el uso de tríodos para la corriente alterna (Triacs) para conmutar la potencia conectada a la bobina de salida cuando el bit de referencia de salida es TRUE. Envía señales ON/OFF a solenoides externos, luces, motores y otros dispositivos. Estos módulos permiten al PLC interactuar y regular en tiempo real el proceso controlado. Los contadores son simulados por software y no existen físicamente. Pueden ser programados para contar arriba, abajo, o ambos arriba y abajo eventos/pulsos. Estos contadores simulados son limitados en su velocidad de conteo, pero adecuados para la mayoría de las aplicaciones en tiempo real [8].

La mayoría de los proveedores de PLC proporcionan módulos de contador de alta velocidad que están basados en hardware y pueden acomodar eventos extremadamente rápidos. Los contadores típicos incluyen contadores ascendentes, descendentes y ascendentes/descendentes. Los temporizadores también son

simulados por software y no existen físicamente. Los tipos más comunes son los temporizadores on-delay, off-delay y retentive. Los incrementos de tiempo varían bastante pero típicamente son mayores que 1/1000 de segundo [8].

El almacenamiento de datos es una memoria/registro de alta velocidad asignado simplemente para almacenar datos. Estos registros se utilizan generalmente en matemáticas o manipulación de datos como almacenamiento temporal. También se utilizan para almacenar valores asociados con temporizadores, contadores, señales de entrada/salida y parámetros de interfaz de usuario. Los búferes de comunicación y las tareas relacionadas con la red y la interfaz de usuario también hacen uso del almacenamiento de alta velocidad. También típicamente se pueden utilizar para almacenar datos y programas cuando se elimina la energía del PLC. En el encendido, los mismos contenidos que existían antes de que se eliminara la energía todavía estarán disponibles [8].

#### 2.2.3.1 El CPU

La unidad de procesamiento central (CPU) está integrada en el PLC fijamente en una sola unidad, mientras que los tipos de rack modular suelen utilizar un módulo plugin. CPU, controlador y procesador son términos utilizados por diferentes fabricantes para denotar el mismo módulo que realiza básicamente las mismas funciones. Los procesadores varían en velocidad de procesamiento y opciones de memoria. Un módulo de procesador se puede dividir en dos secciones: la sección de CPU y la sección de memoria, de la cual se hablará más adelante (Figura 2.8). La sección CPU ejecuta el programa y toma las decisiones necesarias por el PLC para operar y comunicarse con otros módulos. La sección de memoria almacena electrónicamente el pro- grama PLC junto con otra información digital recuperable [7].

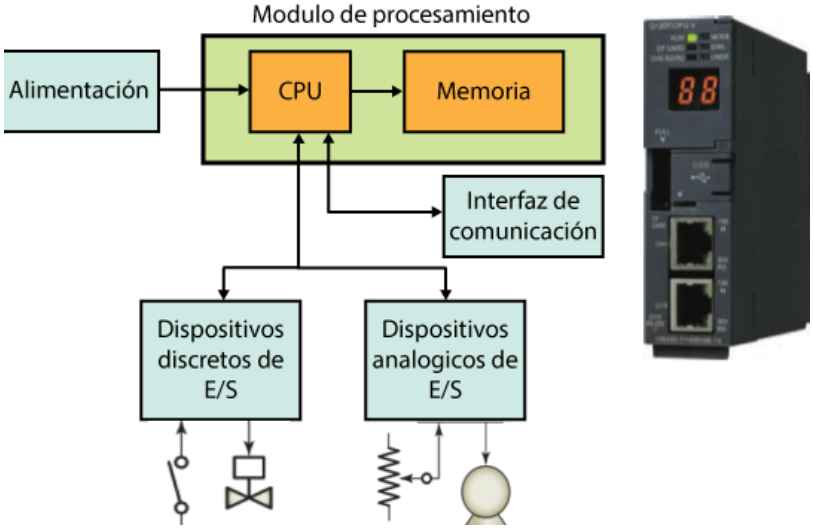



**Figura 2.8** Secciones del módulo de procesamiento de un PLC [7].

El CPU contiene un microprocesador similar al encontrado en un ordenador personal. La diferencia es que el programa utilizado con el microprocesador está diseñado para facilitar el control industrial en lugar de proporcionar computación de propósito general. La CPU ejecuta el sistema operativo, administra la memoria, monitorea las entradas, evalúa la lógica del usuario (programa de escalera), y activa las salidas apropiadas [7].

El CPU dentro de un PLC puede contener más de un procesador. Una ventaja de usar multiprocesamiento es que la velocidad de operación general se mejora. Cada procesador tiene su propia memoria y programas, que operan simultánea e independientemente. En estas configuraciones el escaneo de cada procesador es paralelo e independiente, reduciendo así el tiempo total de respuesta. Los sistemas de PLC tolerantes a fallas soportan procesadores duales para procesos críticos. Estos sistemas permiten al usuario configurar el sistema con procesadores redundantes (dos), lo que permite transferir el control al segundo procesador en caso de fallo del procesador [7].

### 2.2.3.2 La sección I/O

La sección de entrada/salida (E/S) de un PLC es la sección a la que todos los dispositivos de campo están conectados y proporciona la interfaz entre ellos y la CPU. Los arreglos de entrada/salida están integrados en un PLC fijo mientras que los tipos modulares usan módulos externos de E/S que se conectan al PLC [7].

La Figura 2.9 ilustra una sección de E/S basada en rack compuesta por módulos individuales de E/S. Los módulos de interfaz de entrada aceptan señales de la máquina o de los dispositivos de proceso y las convierten en señales que pueden ser utilizadas por el controlador. Los módulos de interfaz de salida convierten las señales del controlador en señales externas utilizadas para controlar la máquina o el proceso. Un PLC típico tiene espacio para varios módulos de E/S, lo que permite personalizarlo para una aplicación particular mediante la selección de los módulos apropiados. Cada ranura en el rack es capaz de acomodar cualquier tipo de módulo de E/S.

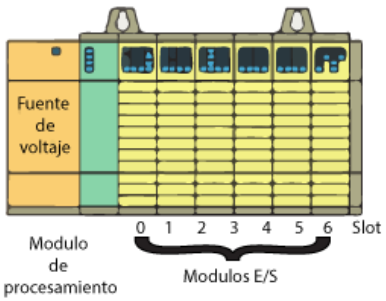



**Figura 2.9** Sección de E/S basada en Racks [7].

El sistema de E/S proporciona una interfaz entre los componentes cableados en el campo y la CPU. La interfaz de entrada permite comunicar a la CPU la información de estado relativa a los procesos, y por lo tanto permite a la CPU comunicar señales de funcionamiento a través de la interfaz de salida a los dispositivos de proceso bajo su control [7].

El sistema de memoria del PLC almacena información sobre la interfaz a los dispositivos de proceso bajo su control. el estado de todas las entradas y salidas. Para llevar un registro de toda esta información, utiliza un sistema llamado direccionamiento. Una dirección es una etiqueta o número que indica dónde se encuentra cierta información en la memoria de un PLC [7].

#### 2.2.3.3 Módulos I/O análogos

Los módulos analógicos representan cantidades físicas que pueden tener un número infinito de valores. Las entradas y salidas analógicas típicas varían de 0 a 20 mA, 4 a 20 mA, o 0 a 10 V. [7]

Los módulos de entrada analógicos normalmente tienen múltiples canales de entrada que permiten que 4, 8 o 16 dispositivos estén conectados al PLC. Los dos tipos básicos de módulos de entrada analógica son la detección de tensión y la detección de corriente. Los módulos de entrada tienen ajustes de cambio de inmersión seleccionables por el usuario para elegir si cada entrada será una entrada de corriente o tensión [7].

La transición de una señal analógica a valores digitales se realiza mediante un convertidor analógico a digital (A/D), el elemento principal del módulo de entrada analógica. Los módulos de entrada de tensión analógica están disponibles en dos tipos: unipolar y bipolar [7].

El módulo de interfaz de salida analógica recibe de los datos digitales del procesador, que se convierten en una tensión o corriente proporcional para controlar un dispositivo de campo analógico. La transición de una señal digital a valores analógicos se realiza mediante un convertidor digital-a-analógico (D/A), el elemento principal del módulo de salida analógica. Una señal de salida analógica es una señal continua y cambiante que es variada bajo el control del programa PLC. Los dispositivos comunes controlados por un módulo de salida analógica PLC incluyen instrumentos, válvulas de control, registrador de cartas, unidades electrónicas y otros tipos de dispositivos de control que responden a señales analógicas. Emplean rangos de salida analógicos estándar como 5 V, 10 V, 0 a 5 V, 0 a 10 V, 4 a 20 mA, o 0 a 20 mA [7].

### 2.2.3.4 Diseño de memoria

La memoria es el elemento que almacena información, programas y datos en un PLC. La memoria del usuario de un PLC incluye espacio para el programa del usuario, así como ubicaciones de memoria direccionables para el almacenamiento de datos. Los datos se almacenan en lugares de memoria mediante un proceso llamado escritura. Los datos se recuperan de la memoria por lo que se conoce como lectura [7].

La complejidad del programa determina la cantidad de memoria requerida. Los elementos de memoria almacenan piezas individuales de información llamadas bits (para dígitos binarios). La cantidad de capacidad de memoria se especifica en incrementos de 1000 o en incrementos "K", donde 1 K es 1024 bytes de almacenamiento de memoria (un byte es 8 bits) [7].

El programa se almacena en la memoria como 1s y 0s, que normalmente se ensamblan en forma de palabras de 16 bits. Los tamaños de memoria se expresan comúnmente en miles de palabras que se pueden almacenar en el sistema; así 2 K es una memoria de 2000 palabras, y 64 K es una memoria de 64.000 palabras. El tamaño de la memoria varía de tan pequeño como 1 K para sistemas pequeños a 32 MB para sistemas muy grandes, ver figura 2.10. La capacidad de memoria es un prerrequisito importante para determinar si un procesador particular manejará los requerimientos de la aplicación específica [7].

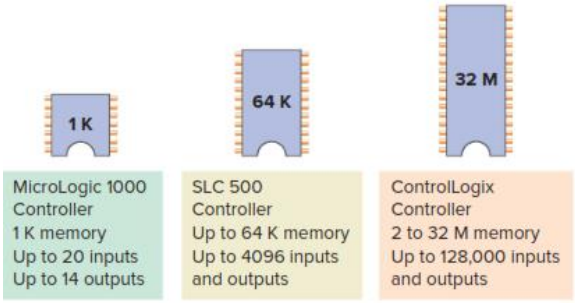



**Figura 2.10** Capacidades típicas de memoria en un PLC [7].

La ubicación de la memoria se refiere a una dirección en la memoria de la CPU donde se puede almacenar una palabra binaria. Una palabra generalmente consiste en 16 bits. Cada pieza binaria de datos es un bit y ocho bits componen un byte, ver Figura 2.11. La utilización de memoria se refiere al número de lugares de memoria necesarios para almacenar cada tipo de instrucción. Una regla general para ubicaciones de memoria es una ubicación por bobina o contacto. Un K de memoria permitiría almacenar en memoria un programa que contiene 1000 bobinas y contactos [7].

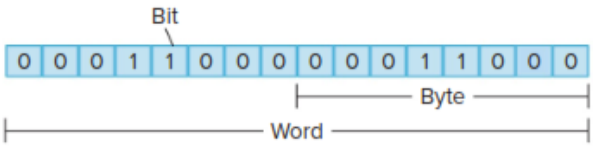



**Figura 2.11** Bit, byte y word [7].

La memoria de un PLC puede dividirse en secciones que tengan funciones específicas. Las secciones de memoria utilizadas para almacenar el estado de entradas y salidas se denominan archivos o tablas de estado de entrada y archivos o tablas de estado de salida, ver Figura 12 [7].

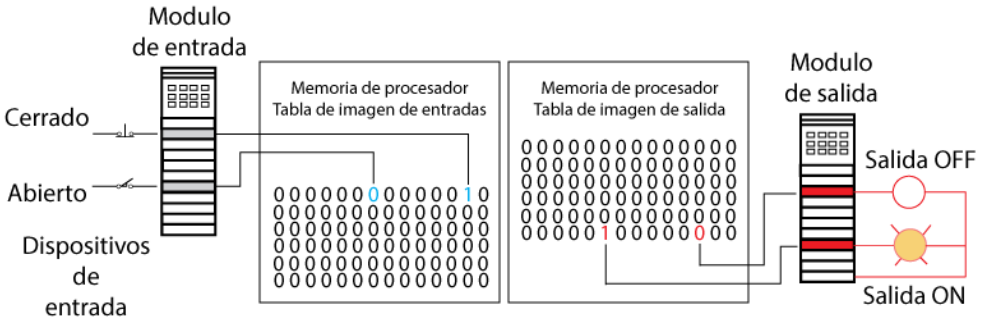



**Figura 2.12** Tabla de registros de memoria de entradas y salidas [7].

Estos términos simplemente se refieren a una ubicación donde se almacena el estado de un dispositivo de entrada o salida. Cada bit es un 1 o un 0, dependiendo de si la entrada está abierta o cerrada. Un contacto cerrado tendría un binario 1 almacenado en su ubicación respectiva en la tabla de entrada, mientras que un contacto abierto tendría un 0 almacenado. Una lámpara que está ENCENDIDA tendría un 1 almacenado en su ubicación respectiva en la tabla de salida, mientras que una lámpara que está APAGADA tendría un 0 almacenado. La CPU revisa constantemente las tablas de entrada y salida de imágenes. Cada vez que se examina una ubicación de memoria, la tabla cambia si el contacto o bobina ha cambiado de estado [7].

El PLC ejecuta rutinas de comprobación de memoria para estar seguro de que la memoria del PLC no se ha dañado. Esta comprobación de memoria se realiza por razones de seguridad. Ayuda a asegurar que el PLC no se ejecutará si la memoria está dañada [7].

### 2.2.3.5 Tipos de memoria

La memoria se puede colocar en dos categorías generales: volátil y no volátil. La memoria volátil perderá la información almacenada si se pierde o se elimina toda la energía operativa. La memoria volátil se altera fácilmente y es bastante adecuada

para la mayoría de las aplicaciones cuando se apoya en la copia de seguridad de la batería [7].

La memoria no volátil tiene la capacidad de retener la información almacenada cuando la energía se elimina accidental o intencionalmente. Como su nombre lo indica, los controladores lógicos programables tienen memoria programable que permite a los usuarios desarrollar y modificar programas de control. Esta memoria se hace no volátil de modo que si se pierde energía, el PLC mantiene su programación [7].

- **Read Only Memory(ROM):**
  Read Only Memory (ROM) almacena programas, y los datos no se pueden cambiar después de que el chip de memoria ha sido fabricado. La ROM se utiliza normalmente para almacenar los programas y datos que definen las capacidades del PLC. La memoria ROM no es volátil, lo que significa que su contenido no se perderá si se pierde energía. La memoria ROM es utilizada por el PLC para el sistema operativo. El sistema operativo es grabado en la memoria ROM por el fabricante del PLC y controla el software del sistema que el usuario utiliza para programar el PLC [7].

- **Random Access Memory(RAM):**
  La memoria de acceso aleatorio (RAM), a veces conocida como memoria de lectura-escritura (R/W), está diseñada para que la información pueda ser escrita o leída desde la memoria. La memoria RAM se utiliza como área de almacenamiento temporal de datos que pueden necesitar ser cambiados rápidamente. La RAM es volátil, lo que significa que los datos almacenados en RAM se perderán si se pierde energía. Una copia de seguridad de la batería es necesaria para evitar la pérdida de datos en caso de una pérdida de energía. La mayoría de los PLC utilizan la tecnología CMOS-RAM para la memoria del usuario. Los chips CMOS-RAM tienen un consumo de corriente muy bajo y pueden mantener la memoria con una batería de litio durante un tiempo prolongado, de dos a cinco años en muchos casos. Algunos procesadores tienen un condensador que proporciona al menos 30 minutos de copia de seguridad de la batería cuando la batería está desconectada y la energía está APAGADA [7].

- **Erasable Programmable Read-Only Memory(EPROM):**
  La memoria de sólo lectura programable borrable (EPROM) proporciona cierto nivel de seguridad contra cambios no autorizados o no deseados en un programa. Las memorias EPROM están diseñadas para que los datos almacenados en ellas puedan ser leídos, pero no se pueden alterar fácilmente sin un equipo especial. La memoria EPROM se usa para respaldar, almacenar o transferir programas de PLC.

- **Electrically erasable programmable read-only memory(EEPROM):**
  La memoria de sólo lectura programable borrable eléctricamente (EEPROM) es una memoria no volátil que ofrece la misma flexibilidad de programación que la RAM. La EEPROM se puede sobreescribir eléctricamente con nuevos datos en lugar de ser borrado con luz ultravioleta. Debido a que la EEPROM es la memoria no volátil, no requiere copia de seguridad de la batería. Proporciona almacenamiento permanente del programa y se puede cambiar fácilmente usando dispositivos de programación estándar. Típicamente, un módulo de memoria EEPROM se usa para almacenar, respaldar o transferir programas PLC [7].

- **Flash EEPROMs:**
  Las memorias Flash EEPROM son similares a las EEPROM que sólo se pueden utilizar para almacenamiento de copia de seguridad. La principal diferencia viene en la memoria flash: son extremadamente rápidos en el ahorro y la recuperación de archivos. Además, no necesitan ser retirados físicamente del procesador para la reprogramación; esto se puede hacer usando el circuito dentro del módulo del procesador en el que residen. La memoria flash a veces también está integrada en el módulo del procesador, donde copia de seguridad automáticamente partes de la RAM. Si la energía falla mientras un PLC con memoria flash se está ejecutando, el PLC volverá a funcionar sin haber perdido ningún dato de trabajo después de restaurar la energía [7].

### 2.2.4 Lenguaje de programación FUP

El término lenguaje de programación de PLC se refiere al método por el cual el usuario comunica información al PLC.

La programación de diagramas de bloques funcionales utiliza instrucciones que se programan como bloques conectados en pantalla para realizar ciertas funciones. Los bloques típicos de funciones incluyen lógica, temporizadores y contadores. Los diagramas funcionales de bloques son similares en su diseño a los diagramas eléctricos/electrónicos de bloques utilizados para simplificar sistemas complejos mostrando bloques de funcionalidad. El concepto principal detrás de un diagrama de bloque funcional es el flujo de datos. Los bloques de funciones se unen para completar un circuito que satisface un requisito de control. Flujo de datos en una ruta desde las entradas, a través de bloques de funciones o instrucciones, y luego a las salidas, ver figura 2.13 [7].

El uso de bloques de funciones para la programación de controladores lógicos programables (PLC) está ganando mayor aceptación. En lugar de la clásica representación de contacto y bobina del diagrama de escalera o la programación

lógica de la escalera de relé, los bloques de funciones presentan una imagen gráfica al programador con algoritmos subyacentes ya definidos. El programador simplemente completa la información necesaria dentro del bloque para completar esa fase del programa [7].

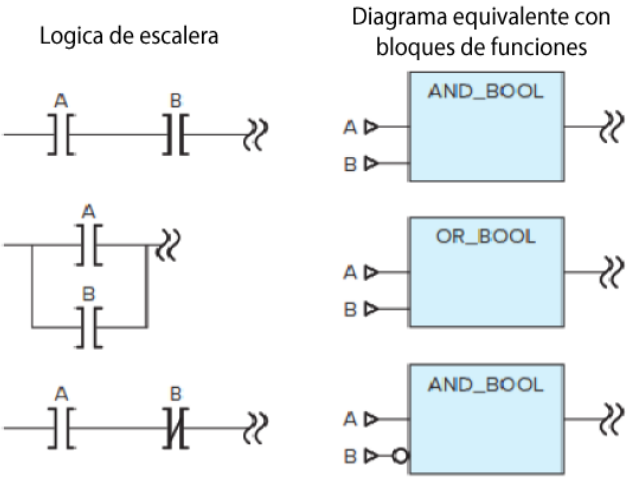



**Figura 2.13** Equivalencia de un diagrama de funciones a bloques a funciones en un diagrama de escalera [7].

### 2.2.5 Direccionamiento de instrucciones

Para completar la entrada de una instrucción tipo relé, debe asignar una dirección a cada instrucción. Esta dirección indica qué entrada de PLC está conectada a qué dispositivo de entrada y qué salida de PLC conducirá qué dispositivo de salida [7].

El direccionamiento de entradas y salidas reales, así como de los internos, depende del modelo de PLC utilizado. Los formatos de direccionamiento pueden variar de una familia de PLC a otra, así como para diferentes fabricantes. Estas direcciones pueden ser representadas en decimal, octal, o hexadecimal dependiendo del sistema numérico utilizado por el PLC. La dirección identifica la función de una instrucción y la vincula a un bit en particular en la tabla de datos de la parte de la memoria. La Figura 2.14 muestra el formato de direccionamiento de un controlador Allen-Bradley SLC 500. Las direcciones contienen el número de ranura del módulo donde los dispositivos de entrada o salida están conectados. Las direcciones están formateadas como tipo de archivo, número de archivo, número de ranura y bit [7].

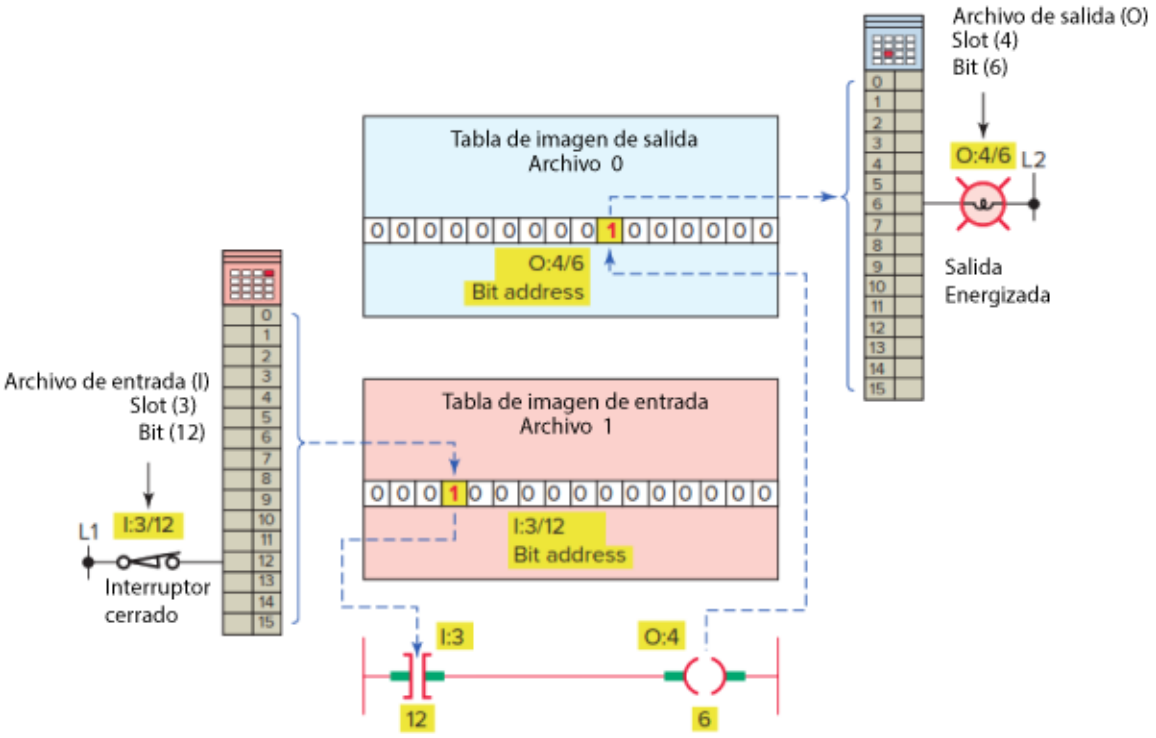



**Figura 2.14** Formato de direccionamiento para un PLC [7].

### 2.2.6 Comunicación de datos

Las comunicaciones de datos se refieren a las diferentes formas en que los sistemas basados en microprocesadores PLC se comunican entre sí y con otros dispositivos. Los dos tipos generales de enlaces de comunicación que pueden establecerse entre el PLC y otros dispositivos son enlaces punto a punto y enlaces de red [7].

La figura 2.15 ilustra un enlace de comunicación en serie punto a punto. Las comunicaciones en serie se utilizan con dispositivos tales como impresoras, estaciones de trabajo del operador, unidades motrices, lectores de código de barras, computadoras u otro PLC. Las interfaces de comunicación en serie están integradas en el módulo del procesador o vienen como módulos separados. Un módulo serie instalado en cada controlador es normalmente todo lo que se necesita para que dos PLC del mismo fabricante establezcan un enlace punto a punto [7].

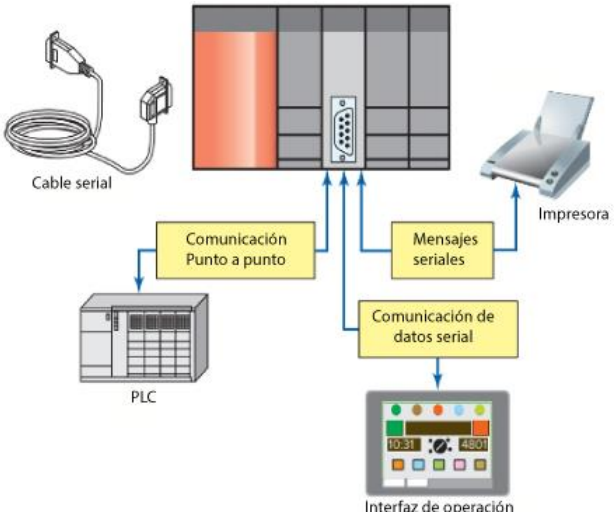



**Figura 2.15** Comunicación serial punto a punto [7].

A medida que los sistemas de control se hacen más complejos, requieren sistemas de comunicación más eficaces entre los componentes del sistema. Una red de área local o LAN es un sistema que interconecta componentes de comunicaciones de datos dentro de una zona geográfica limitada, normalmente no más de una o dos millas. Esencialmente, una LAN es un sistema privado de comunicaciones in situ que permite la comunicación entre computadoras, PLC, robots y similares [7].

La velocidad a la que se pueden transmitir caracteres a lo largo de una línea de comunicación depende del número de bits de información binaria que se pueden enviar en un momento dado. Esta transferencia de información se mide en bits por segundo, o baudios. La Figura 2.16 ilustra un enlace de comunicación LAN. Las comunicaciones de red soportan la comunicación entre múltiples PLCs y otros dispositivos. Las redes de PLC permiten [7]:

- Compartir información tal como el estado actual de los bits dentro del PLC, los cuales podrían determinar otra acción dentro de otro dispositivo conectado.
- Monitorear la información de una locación central.
- La transferencia de datos a través de diferentes PLC para realizar un objetivo global.

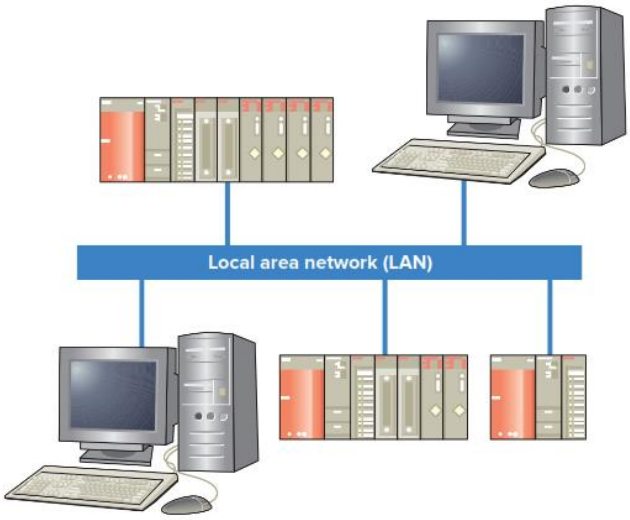



**Figura 2.16** Comunicación por la red de área local (LAN) [7].

Los medios de transmisión son el cable a través del cual fluyen los datos y las señales de control en una red. Los medios de transmisión utilizados en los sistemas de comunicación de datos incluyen cable coaxial, par trenzado o fibra óptica, ver Figura 2.17. Cada cable tiene diferentes capacidades eléctricas y puede ser más o menos adecuado para un entorno específico o tipo de red. No todas las redes transmiten información por cable. Las redes inalámbricas Wi-Fi Ethernet, como el módem de radio DF1, se comunican a través de ondas de radio, que se transmiten a través del aire. El medio de transmisión de fibra óptica es cada vez más popular en aplicaciones basadas en PLC. En comparación con los pares trenzados de cables o cables coaxiales, los cables de fibra óptica tienen ventajas tales como su pequeño tamaño, nulo efecto de interferencia electromagnética, y rangos de transmisión largos. Típicamente, un cable de fibra óptica puede llevar una señal de 1.000 megabaudios a 10 km. Estos tipos de cables tienen anchuras de banda muy grandes por lo que son muy adecuados para la transmisión de alta velocidad en ambientes ruidosos [7].

En aplicaciones industriales, las LAN se han utilizado con mayor frecuencia como sistema de comunicación para sistemas de control distribuidos (DCS). Recuerde que un sistema DCS utiliza controladores individuales para controlar los subsistemas de una máquina o proceso. Este enfoque contrasta con el control centralizado en el que un único controlador gobierna toda la operación. Un segundo uso importante de las redes de área local es el control de supervisión y adquisición de datos (SCADA). Una LAN permite la recopilación y el procesamiento de datos para un grupo de controladores a realizarse utilizando un ordenador host como punto central para la recogida de datos [7].


**Figura 2.17** Medios de transmisión de datos.

Hay tres niveles generales de funcionalidad de las redes industriales. La Figura 2.18 muestra una ilustración de los tres niveles, que pueden resumirse de la siguiente manera [7]:

- **Nivel del dispositivo:** el nivel del dispositivo implica varios dispositivos de sensores y actuadores de máquinas y procesos. Estos pueden incluir dispositivos tales como sensores, interruptores, accionamientos, motores y válvulas.
- **Nivel de control:** el nivel de control involucra a los controladores industriales de la red. Este nivel puede incluir controladores tales como PLCs y controladores de robots. Las comunicaciones a nivel de control incluyen compartir E/S y datos de programas entre los controladores.
- **Nivel de Información**: El nivel de información es una red de planificación compuesta típicamente de redes de negocios y computadoras de la compañía. Este nivel puede incluir programación, ventas, gestión e información corporativa.

Cada dispositivo conectado a una red se conoce como nodo o estación. Como las señales viajan a lo largo de un cable de red, se degradan y se distorsionan en un proceso que se llama atenuación. Si un cable es lo suficientemente largo, la atenuación finalmente hará una señal irreconocible. Un repetidor es un dispositivo que amplifica una señal a su potencia original para permitir que sus señales viajen más lejos. Los diferentes tipos de red tendrán especificaciones diferentes para longitud y tipo de cable sin repetidor [7].

La topología de red es la disposición física de los dispositivos en una red formada por los cables de red cuando se unen nodos. La topología tipo estrella ilustrada en la Figura 2.19 y su funcionamiento puede resumirse como sigue [7]:

- Un conmutador de red o hub está conectado a varios nodos de red PLC.
- Actualmente, la mayoría de las redes Ethernet utilizan interruptores en lugar de centros. Un conmutador realiza la misma función básica que un

concentrador, pero aumenta efectivamente la velocidad, el tamaño y la capacidad de manejo de datos de la red.

- La configuración permite la comunicación bidireccional entre el switch/hub y cada PLC.
- Los PLC se pueden añadir o eliminar de la red sin interrumpir la comunicación con otros dispositivos dentro de la red.
- Un problema con la topología tipo estrella es que si el interruptor/concentrador se cae, toda la LAN se cae.
- Este tipo de sistema funciona mejor cuando la información se transmite principalmente entre el controlador principal y los PLC remotos. Sin embargo, si la mayor parte de la comunicación se produce entre los PLC, la velocidad de operación se ve afectada.
- Además, el sistema tipo estrella puede utilizar cantidades sustanciales de conductores de comunicación para conectar todos los PLC remotos a una ubicación central.

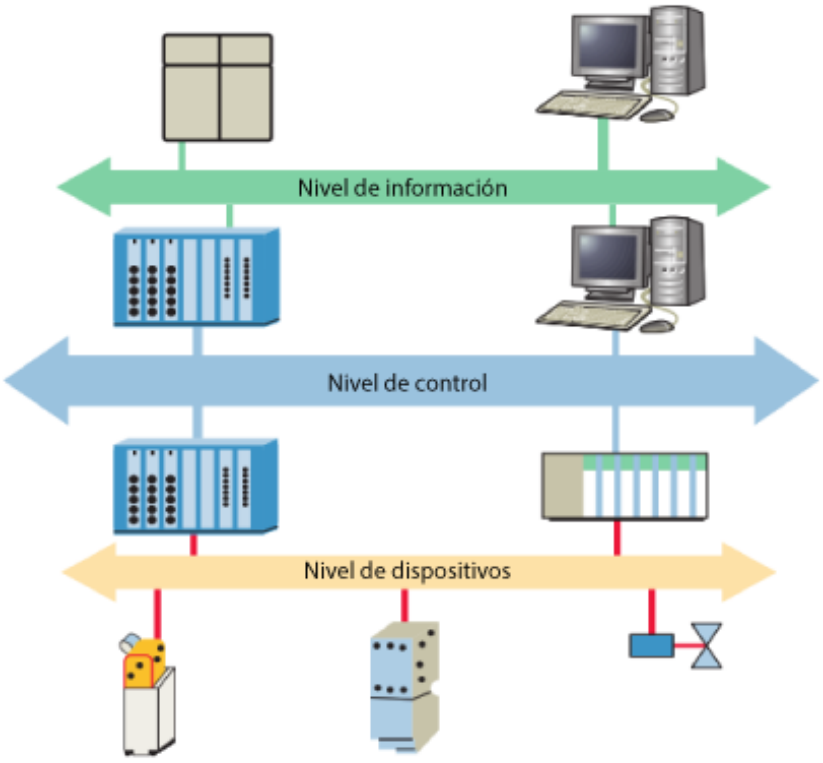



**Figura 2.18** Niveles de funcionalidad de las redes industriales [7].

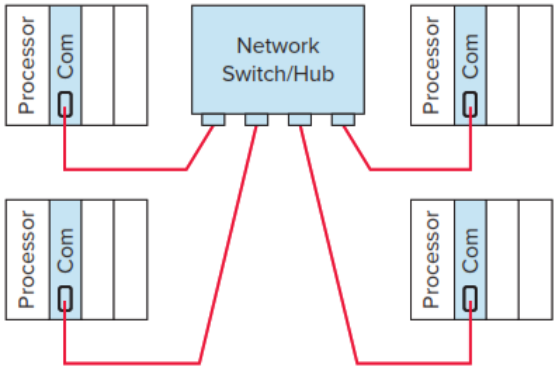



**Figura 2.19** Tipología de red tipo estrella [7].

La topología del bus de comunicación, ilustrada en la Figura 2.20, es una configuración de red en la que todas las estaciones están conectadas en paralelo con el medio de comunicación y todas las estaciones reciben información de cualquier otra estación de la red. El funcionamiento de una red de topología de bus puede resumirse en lo siguiente [7]:

- Utiliza un único cable troncal de bus al que se conectan diferentes PLCs mediante un cable que corta el cable principal.
- Cada PLC está interconectado al bus mediante un módulo de interfaz de red que se conecta mediante un cable o conector.
- Debido a la naturaleza de la tecnología de bus, y a la forma en que los datos se transmiten en la red, cada extremo del bus debe ser terminado con una resistencia de terminación.
- A medida que los datos se mueven a lo largo del bus total, cada nodo del PLC está escuchando su propia dirección de identificación del nodo y acepta sólo la información enviada a esa dirección.
- Debido a la disposición lineal simple, las redes tipo bus requieren menos cable que todas las otras topologías.
- Ninguna estación controla la red y las estaciones pueden comunicarse libremente entre sí.
- Las redes tipo bus son muy útiles en los sistemas de control distributivo, porque cada estación o nodo tiene la misma capacidad de control independiente y puede intercambiar información en cualquier momento dado.
- Otra ventaja de la red de autobuses es que puede añadir o eliminar estaciones de la red con una cantidad mínima de reconfiguración del sistema.
- La principal desventaja de esta red es que todos los nodos dependen de una línea troncal de bus común, y una ruptura en esa línea común puede afectar a muchos nodos.

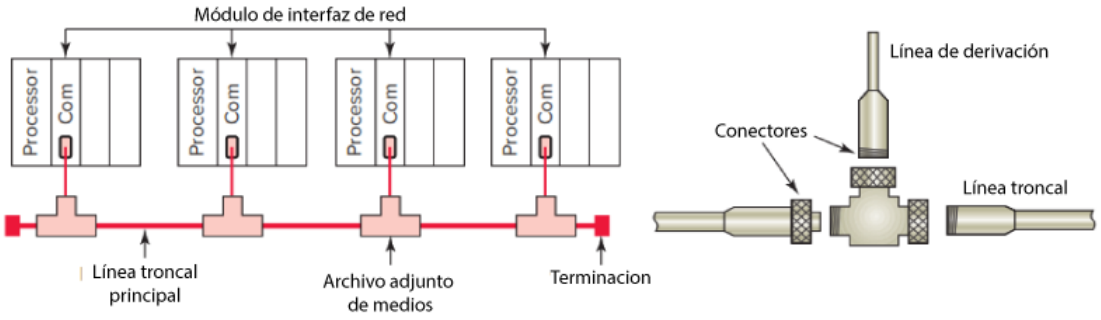



**Figura 2.20** Topología de red tipo bus [7].

Las redes de bus E/S se pueden dividir en dos categorías: redes de bus de dispositivos y redes de bus de proceso. Las redes de bus de dispositivos interactúan con dispositivos de información de bajo nivel como pulsadores e interruptores de límite que transmiten principalmente datos relativos al estado de encendido/apagado del dispositivo y su estado de funcionamiento. Las redes de bus de dispositivos pueden ser clasificadas como buses de bit-wide o byte-wide [7].

Un protocolo de red define cómo se ordenan y codifican los datos para su transmisión en una red. En el pasado, las redes de comunicaciones eran a menudo sistemas patentados diseñados según las normas de un proveedor específico; los usuarios se veían obligados a comprar todos sus componentes de control a un único proveedor. Esto se debe a los diferentes protocolos de comunicación, secuencias de comandos, esquemas de comprobación de errores, y medios de comunicación utilizados por cada fabricante [7].

La mayoría de los PLC se adhieren a los protocolos establecidos por la Organización Internacional de Normalización (ISO). La Interconexión de Sistemas Abiertos (modelo OSI) es un conjunto de siete capas que definen las diferentes etapas que deben atravesar los datos para viajar de un dispositivo a otro a través de una red. El modelo OSI tiene en cuenta el hardware, software, protocolos y arquitectura de red que son necesarios para que dos máquinas se comuniquen. Las conexiones reales entre máquinas son físicas, mientras que las conexiones entre capas en el modelo son lógicas [7].

### 2.2.7 PLC LOGO!

LOGO! Es un pequeño autómata que lleva integradas las siguientes funciones:

- fuente de alimentación.
- unidad de operación y visualización.
- circuitos internos de control.
- interfaz para módulos del programa y cable de conexión del PLC.

Existen numerosos modelos de LOGO! unos y otros se diferencian el número de entradas y salidas, las funciones que pueden soportar, etc. en cuanto a las funciones que el PLC es capaz de controlar, depende de la generación del autómata, siendo los 0aB0 de primera generación, los 0AB1 de segunda y así sucesivamente hasta los 0AB4 que son la quinta generación.


**Figura 2.21** PLC LOGO! de Siemens.

### 2.2.7.1 Información técnica de PLC LOGO!

En base a la página del fabricante del PLC las características técnicas son las siguientes:

- 1 display
- Soporta 1/24 V DC
- El voltaje menor con el que puede ser energizado es 10.8 V
- El voltaje mayor con el que puede ser realizado es 28.8 V
- 8 entradas digitales de las cuales 4 pueden ser usadas como entradas analógicas (0 – 10 V)
- 4 salidas de relevador
- No cuente con fusible interno, requiere un externo.
- la corriente de salida es de 10 A para cargar resistivas, para cargas inductivas 3 A.
- Puede trabajar correctamente entre -20 °C y 55 °C

### 2.2.7.2 Conexionado con el PLC LOGO!

Las entradas se definen por I1, I2, etc. las salidas se denominan por Q1, Q2, etc. El PLC LOGO! y sus entradas deben alimentarse por separado, para evitar funcionamientos erróneos. Vea la figura 2.22.

Las entradas del PLC LOGO! no poseen separación galvánica, por lo que requieren el mismo potencial de referencia que la tensión de alimentación.

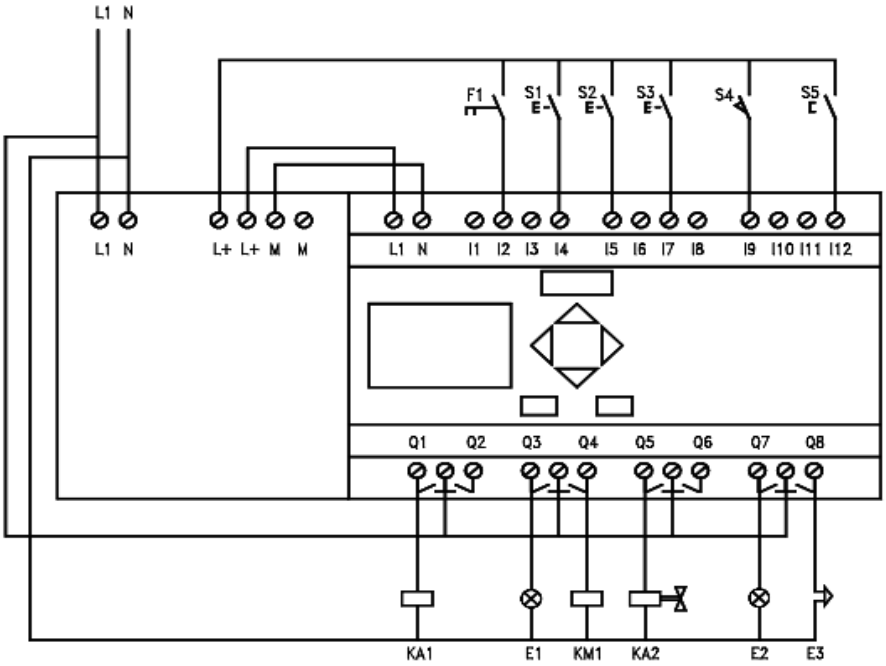



**Figura 2. 22** Conexionado de un PLC LOGO!

### 2.2.7.3 Programación

La programación del PLC LOGO! puede efectuarse directamente desde las teclas situadas en la parte frontal del propio PLC, o también a través de un ordenador. cuando se programa desde el PC se emplea el software LOGO! Soft Comfort, del cual existen varias versiones, ya que el fabricante las actualizaciones constantemente.

**Programación básica en el propio PLC:**
Al trabajar desde las teclas del LOGO! podemos acceder a cuatro menús de trabajo, y dentro de cada uno de ellos hay varias opciones, vea a figura la figura 2.23.

**Programación mediante LOGO! Soft Comfort:**
Con el software LOGO! Soft Comfort pueden elaborarse los programas de forma más rápida, confortable y clara. la obtención del esquema consiste en colocar los bloques de programación libremente en una plataforma de programa y unirlos entre sí. el software facilita la labor del usuario, entre otras cosas, mediante la simulación del programa que este realizó realizando.

La pantalla de operación del software LOGO! Soft Comfort consta de varias barras de herramientas, cómo podemos ver en la figura 2.24.

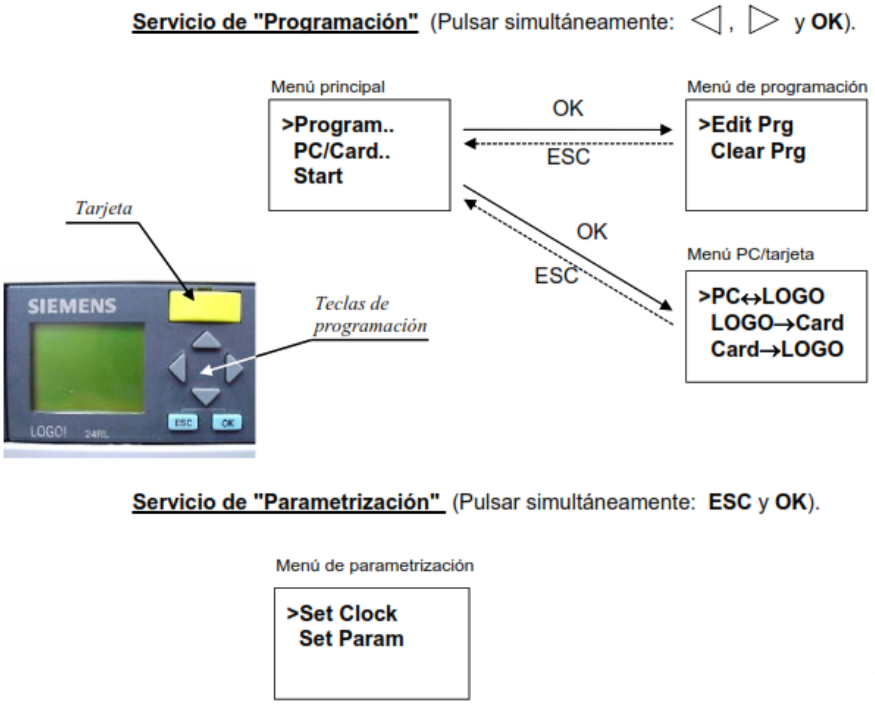



**Figura 2.23** Menús de programación y parametrización del PLC LOGO!

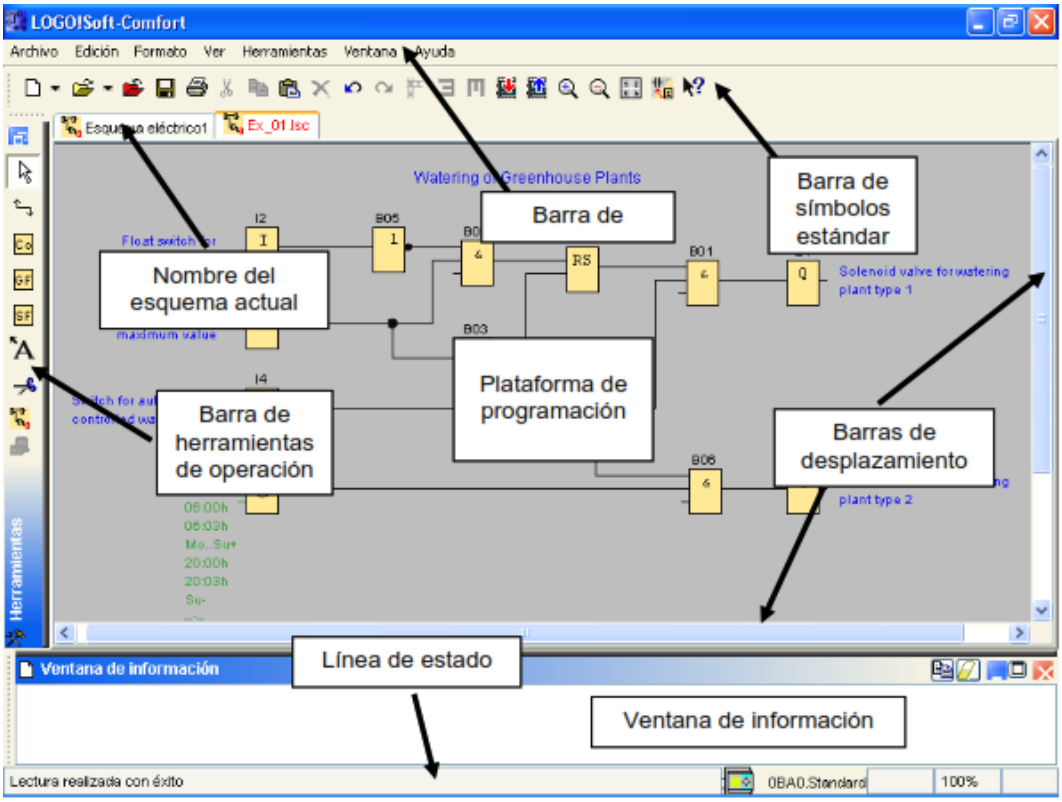



**Figura 2.24** Pantalla principal del software LOGO! Soft Comfort [9].

Entre otras de las ventajas que tiene este software, es que se puede realizar visualización en tiempo real del programa que has desarrollado.

Para poder realizar todo esto antes mencionado, así como, la descarga del programa en el PLC es necesario que conectes con anticipación el PLC con este software a través de la red local. lo único que se tiene que realizar es conectar el PLC LOGO! al modem, y después introducir IP del PLC al software logo.

#### 2.2.7.4 Bloques funcionales dentro del LOGO Soft Confort

Un bloque funcional es un elemento que convierte la información que les llega a sus entradas en información de salida.

Dentro de las funciones básicas, todos los bloques corresponden a las compuertas lógicas elementales, tales como: AND, NAND, OR, NOR, XOR y NOT.

Dentro de las funciones especiales, se trata en bloques más complejos, como: reloj, marcha-paro, diversos tipos de temporizadores, entradas analógicas, etc.

## 2.3 Raspberry

El Raspberry Pi (ver figura 2.25) es un ordenador de bajo coste, de tamaño tarjeta de crédito que se conecta a un monitor de ordenador o TV, y utiliza un teclado y un ratón estándar. Es un pequeño dispositivo capaz que permite a personas de todas las edades explorar la informática, y aprender a programar en idiomas como Scratch y Python. Es capaz de hacer todo lo que se espera que haga una computadora de escritorio, desde navegar por Internet y jugar vídeo de alta definición, hasta hacer hojas de cálculo, procesamiento de textos y juegos [10].

Es más, el Raspberry Pi tiene la capacidad de interactuar con el mundo exterior, y se ha utilizado en una amplia gama de proyectos de fabricantes digitales, desde máquinas de música y detectores de padres a estaciones meteorológicas y pajareras tweeting con cámaras infrarrojas. Queremos ver el Raspberry Pi ser utilizado por los niños de todo el mundo para aprender a programar y entender cómo funcionan las computadoras [10].

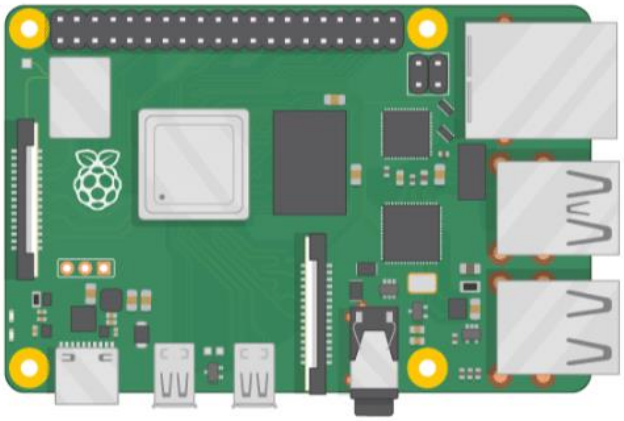


**Figura 2.25** Raspberry Pi 3 [10].

### 2.3.1 Un poco de historia de las tarjetas Raspberry

La Raspberry Pi Foundation es una organización benéfica educativa registrada (número de registro 1129409) con sede en el Reino Unido. El objetivo de nuestra Fundación es promover la educación de adultos y niños, en particular en el ámbito de la informática, la informática y materias conexas [10].

### 2.3.2 Partes de la tarjeta Raspberry

De manera general las partes de una raspberry son las siguientes (ver figura 2.26):

- **Puerto USB:** este se utiliza para conectar un ratón y un teclado. También puede conectar otros componentes, como una unidad USB.
- **Tarjeta Micro SD:** usted puede insertar una tarjeta SD aquí. Aquí es donde el software del sistema operativo y sus archivos se almacenan.
- **Puerto Ethernet:** esto se utiliza para conectar Raspberry Pi a una red con un cable. Raspberry Pi también puede conectarse a una red a través de LAN inalámbrica.
- **Conector de audio:** puedes conectar auriculares o altavoces aquí.
- **Puerto HDMI:** aquí es donde se conecta el monitor (o proyector) que se está utilizando para mostrar la salida de la Raspberry Pi. Si su monitor tiene altavoces, también puede utilizarlos para escuchar el sonido.
- **Conector de energía Micro USB:** aquí es donde se conecta una fuente de alimentación. Siempre debe hacer esto último, después de haber conectado todos los demás componentes.
- **Puertos de propósito general:** estos permiten conectar componentes electrónicos como Leds y botones a Raspberry Pi.

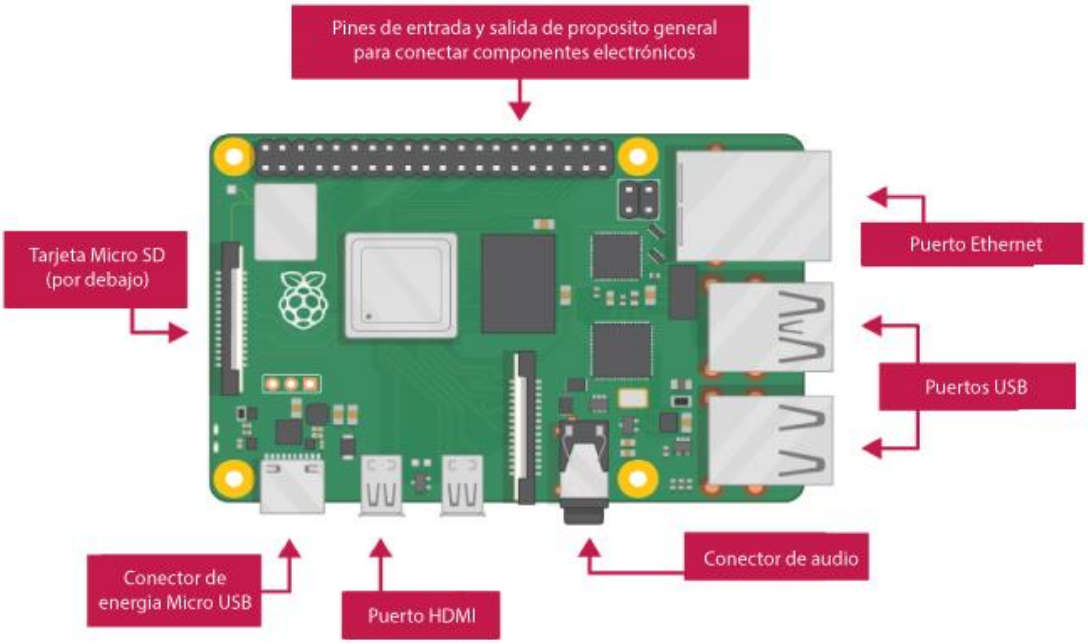



**Figura 2.26** Partes de una Raspberry Pi [10].

### 2.3.3 Características de la Raspberry Pi 3

La Raspberry Pi 3 será la que esteré utilizando dentro de mi proyecto, es por eso que en seguida enlisto las características principales de la misma [10]:

- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
- 1GB RAM
- BCM43438 wireless LAN y Bluetooth Low Energy (BLE)
- 100 Base Ethernet
- 40-pin extended GPIO
- 4 USB
- Puerto HDMI y de audio.
- Puerto para conectar una cámara de Raspberry Pi
- DSI para conectar una pantalla.
- Slot para tarjeta Micro SD.
- Puerto de alimentación Micro USB

## 2.4 NodeRed

Node-RED es una herramienta de programación basada en flujos, originalmente desarrollada por el equipo de Servicios de Tecnología Emergente de IBM y ahora parte de la Fundación JS [3].

### 2.4.1 ¿Qué es NodeRed?

Node-RED es una herramienta de programación para conectar dispositivos de hardware, API y servicios en línea. Principalmente, es una herramienta visual diseñada para la Internet de las Cosas, pero también se puede utilizar para otras aplicaciones para ensamblar muy rápidamente flujos de diversos servicios [11].

Node-RED es de código abierto y fue creado originalmente por la organización IBM Emerging Technology. Está incluido en el paquete de aplicaciones de inicio de IoT de IBM Bluemix (Plataforma-as-a-Service o Paas). Node-RED también puede desplegarse por separado utilizando la aplicación Node.js. Actualmente, Node-RED es un proyecto de la Fundación JS [11].

Node-RED permite a los usuarios unir servicios Web y hardware mediante la sustitución de tareas comunes de codificación de bajo nivel (como un servicio simple hablar con un puerto serie), y esto se puede hacer con una interfaz visual arrastrar-soltar. Varios componentes de Node-RED están conectados entre sí para crear un flujo. La mayor parte del código necesario se crea automáticamente [11].

### 2.4.2 Historia de NodeRed

Node-RED comenzó su vida a principios de 2013 como un proyecto paralelo de Nick O'Leary y Dave Conway-Jones del grupo de servicios de tecnología emergente de IBM. Lo que comenzó como una prueba de concepto para visualizar y manipular asignaciones entre temas MQTT, rápidamente se convirtió en una herramienta mucho más general que podría extenderse fácilmente en cualquier dirección. Fue de fuente abierta en septiembre de 2013 y se ha desarrollado en código abierto desde entonces, siendo uno de los proyectos fundadores de la Fundación JS en octubre de 2016 [3].

### 2.4.3 Características

Las principales características de Node-RED se enumeran a continuación [11]:

- Es compatible con la edición de flujo basado en el navegador.
- Como está construido en Node.js, es compatible con un entorno de ejecución ligero junto con el evento impulsado y el modelo que no bloquea.
- Los diversos flujos creados en Node-RED se almacenan utilizando JSON, que puede ser fácilmente importado y exportado para compartir con otros. Puede ejecutarlo localmente (soporte Docker, etc).
- Puede caber fácilmente en los dispositivos más utilizados como Raspberry Pi, Beaglebone Negro, Arduino, dispositivos basados en Android, etc.

- Se puede ejecutar en el entorno de la nube como Bluemix, AWS, MS-Azure, etc.

### 2.4.4 Arquitectura

Algunos de los aspectos clave de la arquitectura Node-RED se enumeran a continuación [11]:

- Node-RED (última versión 0.16) es rápido porque es impulsado por la última versión soportada de Node.js (LTS 6.x, 6.x).
- Arquitectura asíncrona y basada en eventos. Cola de eventos de un solo hilo, que admite la simplicidad.
- Soporte de un solo idioma para Javascript (tanto de front-end como back-end).
- La arquitectura completa se ha construido usando express, d3, jquery y ws.

### 2.4.5 Aplicaciones

Node-RED se puede utilizar en una gama de aplicaciones. Los principales se enumeran a continuación [11]:

- En Bluemix, para conectarse a IoT (con Rest y MQTT).
- Para enlazar y conectar a bases de datos (Mongodb).
- Para almacenar datos de IoT para el cómputo presente y futuro.
- Para las redes sociales, cuando se toman medidas y cuando se necesitan aplicaciones basadas en eventos (como Twitter).

Node-RED tiene más de 225.000 repositorios de paquetes, a los que es fácil extender y añadir nuevos paquetes. También tiene una comunidad dedicada, y está construido con arquitectura robusta usando Node.js (que es muy popular en estos días).

Node-RED se puede utilizar en aplicaciones y servicios impulsados por eventos y de rápido acceso al mercado, con pasos fácilmente implementables.

## 2.5 Ubidots

Ubidots una plataforma de IoT (Internet de las cosas) que habilita la toma de decisiones a empresas de integración de sistemas a nivel global (ver figura 2.27). Este producto permite enviar datos de sensores a la nube, configurar tableros y

alertas, conectarse con otras plataformas, usar herramientas de analítica y arrojar mapas de datos en tiempo real [13].

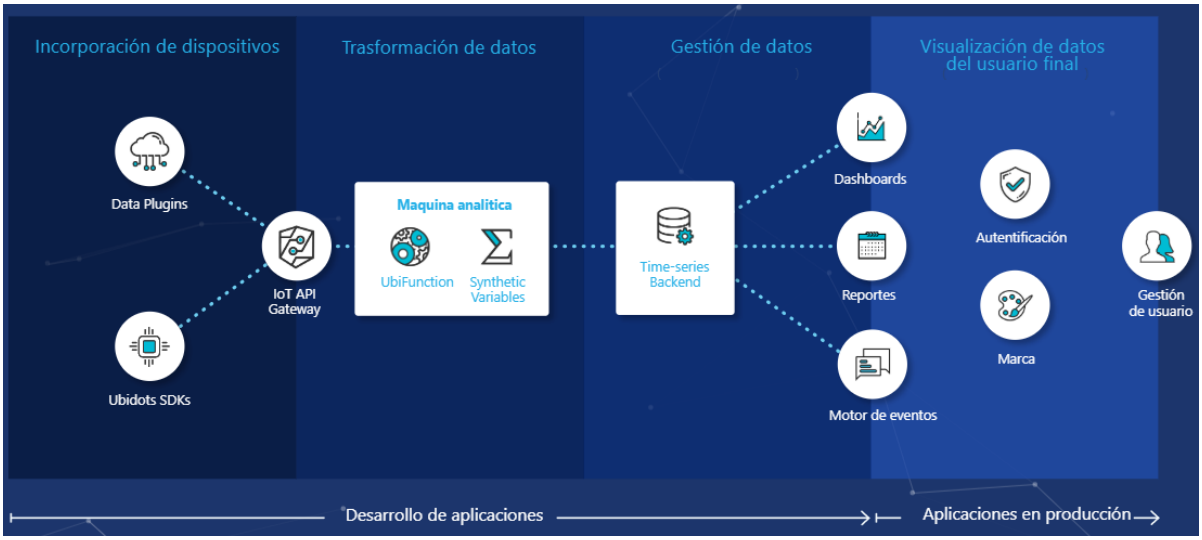



**Figura 2.27** Servicios de Ubidots *[13]*.

### 2.5.2 Historia de Ubidots

Ubidots nació como una firma de servicios de ingeniería en 2012, entregando soluciones IoT de extremo a extremo en conjunto con su socio y cofundadora compañía Netux, para monitorizar, controlar y automatizar remotamente procesos para clientes sanitarios, así como startups financiadas y Fortune 1,000s en el sureste americano y a través de América Latina [13].

Entre 2012 y 2014, Ubidots logró innumerables proyectos conectados a Internet a través de Salud, Energía / Servicios públicos, Fabricación, Transporte, y Retail [13]. Con una sólida columna vertebral y una firme determinación de convertirse en una empresa liderada por productos, Ubidots se unió al Acelerador Boston Masschallenge en 2013, girando hacia una startup global de tecnología, ganando emprendimiento y apoyo estadounidense, y dejando atrás el negocio de servicios locales [13].

Ubidots se ha convertido en conocido dentro del hardware, software, ingeniería integrada, y los círculos de fabricantes como la plataforma asequible, fiable y más utilizable en el ecosistema de IoT Application Enablement [13].

### 5.5.3 ¿Cómo funciona Ubidots?

Cada vez que un dispositivo actualiza un valor de sensor en una variable, se crea un punto de datos o "dot". Ubidots almacena los puntos que vienen de sus dispositivos dentro de las variables, y estos puntos almacenados tienen las *timestamps* correspondientes [13]:

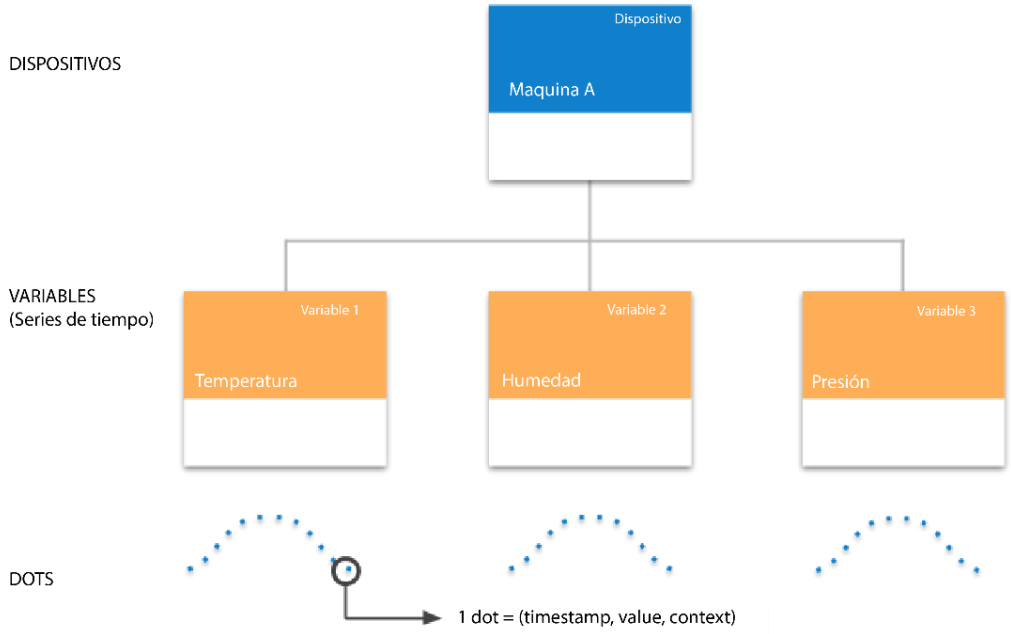



**Figura 2.28** Jerarquía de datos en Ubidots [13].

Cada “dot” contiene los siguientes elementos [13]:

**Tabla I** Elementos de un "dot" en Ubidots [13].

| Elemento | Descripción | Mandatorio |
|---|---|---|
| Value | Un valor numérico. Ubidots acepta hasta 16 números de longitud de coma flotante. | SI |
| Timestamp | Tiempo Unix Epoch, en milisegundos. Si no se especifica, nuestros servidores asignarán uno en la recepción. | NO |
| Context | Una colección arbitraria de pares clave-valor. Utilizado principalmente para almacenar las coordenadas de latitud y longitud de los dispositivos GPS. | NO |

**Values**

Un valor numérico. Ubidots acepta hasta 16 números de longitud de coma flotante [13].

{"value" : 34.87654974}

**Timestamps**

Un “timestamps”, como se describe aquí, es una manera de rastrear el tiempo como un total de segundos. Este conteo comienza en la época de Unix el 1 de enero de 1970 en UTC. Por lo tanto, la marca de tiempo unix es meramente el número de segundos entre una fecha en particular y la época Unix. Por favor, tenga en cuenta que cuando envíe datos a Ubidots, debe establecer la marca de tiempo en milisegundos; además, si obtiene la marca de tiempo de un punto, será en milisegundos [13].

"timestamp" : 1537453824000

La marca de tiempo anterior corresponde al jueves, 20 de septiembre de 2018 2:30:24 PM.

**Context**

Los valores numéricos no son el único tipo de datos soportado; también puede almacenar tipos de cadena o caracteres dentro de lo que llamamos “context”. El “context” es un objeto de valor clave que le permite almacenar no sólo valores numéricos sino también valores de cadena. Un ejemplo de uso del “context” podría ser [13]:

"context" : {"status" : "on", "weather" : "sunny"}

Un “context” se usa comúnmente para almacenar las coordenadas de latitud y longitud de su dispositivo para casos de uso de aplicaciones de GPS/rastreo. Todos los mapas de Ubidots utilizan las teclas lat y lng del “context” de un punto para extraer las coordenadas de su dispositivo, De esta manera sólo necesita enviar un solo punto con los valores de coordenadas en el contexto variable para trazar un mapa en lugar de enviar por separado tanto latitud como longitud en dos variables diferentes. A continuación puede encontrar un “context” típico con valores de coordenadas [13]:

"context" : {"lat":-6.2, "lng":75.4, "weather" : "sunny"}

## 2.6 Protocolo MODBUS

El protocolo de comunicaciones industriales MODBUS fue desarrollado en 1979 por la empresa norteamericana MODICON y debido a que es público, relativamente sencillo de implementar y flexible se ha convertido en uno de los protocolos de comunicaciones más populares en sistemas de automatización y control. A parte de que muchos fabricantes utilizan este protocolo en sus dispositivos, existen también versiones con pequeñas modificaciones o adaptadas para otros entornos (como por ejemplo, JBUS o MODBUS II) [1].

MODBUS especifica el procedimiento que el controlador y el esclavo utilizan para intercambiar datos, el formato de estos datos, y cómo se tratan los errores. No específica el tipo de red de comunicaciones a utilizar, por lo que se puede implementar sobre redes basadas en Ethernet, RS-485, RS-232, etc [1].

### 2.6.1 Descripción

MODBUS funciona siempre en modo maestro-esclavo (cliente - servidor), siendo el maestro (cliente) quien controla en todo momento las comunicaciones con los esclavos que pueden ser hasta 247. Los esclavos (servidores) se limitan a retornar los datos solicitados o a ejecutar la acción indicada por el maestro. La comunicación del maestro hacia los esclavos puede ser de dos tipos [1]:

- “Peer to peer”: en que se establece comunicación “maestro - esclavo”, el maestro solicita información y el esclavo responde (se envía el comando a un dispositivo comprendido entre las direcciones 1- 247).
- “Broadcast”: en que se establece comunicación “maestro - todos los esclavos”, el maestro envía un comando a todos los esclavos de la red sin esperar respuesta (se envía a la dirección 0).

Como se puede ver, la secuencia básica en las comunicaciones MODBUS consiste siempre en una trama de pregunta, seguida de su correspondiente trama de respuesta [1]:

- Pregunta: con el código de función que indica al esclavo que operación ha de realizar, y los bytes necesarios (datos, comprobación) para su ejecución.
- Respuesta: con la confirmación o datos resultantes de la ejecución de la función.

Existe algún caso concreto, en que hay más de una trama de respuesta para una trama de pregunta, como por ejemplo, cuando el maestro envía una operación cuya respuesta puede llevar al esclavo un tiempo elaborar. En estas situaciones el esclavo envía una primera respuesta indicando que aún no tiene los datos y tardará

un tiempo en disponer de ellos, y otra segunda con los datos o confirmación de la operación [1].
Además las comunicaciones MODBUS se pueden realizar en modo ASCII, en modo RTU o en modo TCP/IP. En modo ASCII los bytes se envían codificados en ASCII, es decir, que por cada byte a transmitir se envían dos caracteres ASCII (2 bytes) con su representación hexadecimal (esto permite leer las tramas con un simple editor de texto). En modo RTU se envían en binario, tal cual. En el modo ASCII las tramas comienzan por 3AH (carácter ':'), y terminan en 0DH-0AH (CR LF Carrier Return Line Feed) y cada byte se envía como dos caracteres ASCII. En modo RTU no se utiliza indicador de inicio y final de trama. En modo TCP/IP es simplemente MODBUS sobre Ethernet. En lugar de usar direcciones de dispositivos para comunicarse con dispositivos esclavos, se utilizan direcciones IP. Con MODBUS / TCP, Los datos MODBUS simplemente se encapsulan dentro de un paquete TCP/IP. Por lo tanto, cualquier red Ethernet que admita TCP/IP debe admitir inmediatamente MODBUS / TCP.

### 2.6.2 Campos de las tramas MODBUS

El número de campos de las tramas MODBUS varía ligeramente dependiendo de si utilizamos la codificación ASCII, RTU o TCP/IP [1]:

Codificación ASCII:

- Inicio de trama: 2 caracteres ASCII (que representan 1 byte) codificando el carácter “:” (0x3A)
- Número de esclavo: 2 caracteres ASCII (que representan 1 byte) codificando la dirección del esclavo destino (u origen) de la trama
- Código Operación: 2 caracteres ASCII (que representan 1 byte) con el código de operación
- Dirección, datos y subfunciones Datos: con los parámetros necesarios para realizar la operación.
- LRC (16): H L
- Final de trama: 4 caracteres ASCII (que representan 2 bytes) con los caracteres CR (0x0D) - LF (0x0A)

Codificación RTU:

- Número de esclavo: 1 byte con la dirección del esclavo destino (u origen) de la trama
- Código Operación: 1 byte con el código de operación
- Subfunciones Datos: con los parámetros necesarios para realizar la operación.
- CRC (16): H L.

Codificación TCP/IP

- Identificador de transacción: 2 bytes.
- Identificador del protocolo: 2 bytes.
- Longitud del mensaje: 2 bytes.
- Numero de esclavo: 1 byte.
- Código de operación: 1 byte.
- Posición del registro inicial: 2 bytes.
- Numero de registros requeridos: 2 bytes.

### 2.6.3 Descripción de los campos de las tramas MODBUS

**Número de Esclavo (1byte):** En el caso de las tramas enviadas por el máster, el campo de número de esclavo indica la dirección del destinatario de esta trama. Permite direccionar hasta 247 esclavos, con las direcciones de 1d a 247d (0x00 a 0xF7). El 0x00 es para los mensajes de Broadcast, así el primer esclavo comienza con la dirección 1 (de 1 a 247). En el caso de las tramas enviadas por los esclavos, este byte sirve para indicar al máster a quién pertenece la respuesta. Es decir, cada vez que un esclavo responde, sitúa su propia dirección en el byte de dirección lo que permite saber al maestro a que equipo corresponde cada respuesta. Las tramas broadcast, no tienen asociada respuesta, y algunas implementaciones de MODBUS no admiten la trama de broadcast [1].

**Código de Operación o Función (1byte):** Indica el tipo de operación que queremos realizar sobre el esclavo. Las operaciones se pueden clasificar en dos tipos [1]:

- De lectura / escritura en memoria: para consultar o modificar el estado de los registros del mapa de memoria del esclavo.
- Ordenes de control del esclavo: para realizar alguna actuación sobre el esclavo.

El código de operación puede tomar cualquier valor comprendido entre el 0 y el 127 (el bit de más peso se reserva para indicar error). Cada código se corresponde con una determinada operación. Algunos de estos códigos se consideran estándar y son aceptados e interpretados por igual por todos los dispositivos que dicen ser compatibles con MODBUS, mientras que otros códigos son implementaciones propias de cada fabricante. Es decir que algunos fabricantes realizan implementaciones propias de estos códigos “no estándar” [1].
Es también mediante el código de función que el esclavo confirma si la operación se ha ejecutado correctamente o no. Si ha ido bien responde con el mismo código de operación que se le ha enviado, mientras que si se ha producido algún error, responde también con el mismo código de operación pero con su bit de más peso

a 1 ( 0x80 ) y un byte en el campo de datos indicando el código de error que ha tenido lugar [1].

**Dirección, datos y subfunciones (n bytes):** Este campo contiene la información necesaria para realizar la operación indicada en el código de operación. Cada operación necesitará de unos parámetros u otros, por lo que el número de bytes de este campo variará según la operación a realizar. En el caso del esclavo, este puede responder con tramas con o sin campo de datos dependiendo de la operación. En los casos en que se produzca algún error es posible que el esclavo responda con un byte extra para especificar el código de error [1].
Al establecer la dirección de una variable u otro elemento en el mapa de direcciones MODBUS, direccionamos con 1 unidad menos a la del registro al que queremos acceder, de manera que si p.ej. quisiéramos acceder al relé @ 127d, lo haríamos situando el valor 126d en el byte del campo de dirección. Otros ejemplos [1]:

- El relé número 1 de un controlador se direccionaría con el valor 0000 en el campo de dirección de un mensaje MODBUS
- El relé 0x007F (127d) de un controlador se direccionaría con el valor 0x007E (126d) en el campo de dirección de un mensaje MODBUS
- El Holding Register 40001 se accedería situando el valor 0000 en el campo de dirección del mensaje. Como se puede ver el código de función de acceso a los Holding Registers lleva implícito el acceso a la dirección '4XXXX'.
- El Holding Register 40108 es accedido leyendo de la dirección 0x006B (107d)

Generalmente en MODBUS cada tipo de dato se mapea en un rango de memoria concreto [1]:
**@1-10000 (DOs – salidas digitales):** 1 bit por dirección para indicar el estado de una salida, mando o relé (0 desactivado, 1 activado). Las direcciones de este rango se suelen acceder mediante las funciones 1 (lectura), 5 (escritura), 15 (escritura múltiple).
**@10001-20000 (DIs – entradas digitales):** 1 bit por dirección para leer el estado de una entrada digital (0 desactivada, 1 activada) también denominadas DIs (Digital Inputs). Las direcciones de este rango se suelen acceder con la función 2 (lectura) y llevan implícita la dirección 10001 como dirección base (para acceder a una dirección bastará con especificar la distancia entre esta y la dirección base).
**@20001-30000:** el protocolo MODBUS estándar no hace uso de este rango de direcciones.
**@30001-40000 (AIs – entradas analogicas):** 16 bits por dirección con el estado de las medidas o entradas analógicas también denominadas AIs (Analog Inputs). Dependiendo del dispositivo este puede hacer uso de más de un registro para almacenar la información de la medida, así con 2 registros consecutivos podríamos almacenar medidas de 32 bits. Las direcciones de este rango se acceden mediante la función 4 (lectura) y llevan implícita la dirección 30001 como dirección base (para

acceder a una dirección bastará con especificar la distancia entre esta y la dirección base).

**@40001-50000 (AOs – salidas analogicas):** 16 bits con los registros de salidas analógicas o de propósito general (Output Registers – Holding Registers). Se acceden con las funciones 3 (lectura), 6 (escritura) o 16 (escritura múltiple) y llevan implícita la dirección 40001 como dirección base (para acceder a una dirección bastará con especificar la distancia entre esta y la dirección base).

Algunos fabricantes expresan la dirección de forma compuesta, separando la dirección de palabra y la dirección de bit: por ejemplo la word 0x30 bit 1

Como se cita en el apartado de “Código de Operación o Función”, cuando se produce un error en la ejecución de un comando en el esclavo, este responde poniendo a 1 el bit de más peso del código de función (0x80). Con este bit el maestro sabe que se ha producido un error, pero para obtener más detalle sobre el tipo de error, ha de comprobar el campo de datos:

**Tabla II** Código de errores en protocolo Modbus.

| Código | Nombre | Significado |
|---|---|---|
| 01 | ILLEGAL FUNCTION | El código de función recibido no se corresponde a ningún comando disponible en el esclavo |
| 02 | ILLEGAL DATA ADDRESS | La dirección indicada en la trama no se corresponde a ninguna dirección válida del esclavo |
| 03 | ILLEGAL DATA VALUE | El valor enviado al esclavo no es válido |
| 04 | SLAVE DEVICE FAILURE | El esclavo ha recibido la trama y la ha comenzado a procesar, pero se ha producido algún error y no ha podido termina la tarea. |
| 05 | ACKNOWLEDGE | El esclavo ha recibido la trama y la está procesando pero esto le llevará un periodo un poco largo. Mediante esta respuesta se evita que el máster considere un error de timeout. El máster podrá enviar más tarde una trama una trama de tipo Poll Program Complete para verificar si ha completado el comando |
| 06 | SLAVE DEVICE BUSY | El esclavo está ocupado realizando otra tarea y no puede atender a esa petición en ese instante por lo que el máster tendrá que reintentarlo más adelante. |

**Control de errores LRC o CRC:** Se utiliza un sistema de detección de errores diferente dependiendo del tipo de codificación utilizado (ASCII o RTU). En el caso de la codificación ASCII es el checksum (o Longitud Redundancy Check LRC) en módulo 16 expresado en ASCII (2 caracteres representan 1 byte), sin considerar el ":" ni el “CR LF” de la trama. En la codificación RTU se utiliza el método de CRC (Cyclical Redundancy Check) codificado en 2 bytes (16 bits) [1]. En el caso TCP/IP no existe un control de errores.

### 2.1.4 Descripción de los códigos de función más frecuentes

Los siguientes códigos son algunos de los códigos de función MODBUS más extendidos, soportados por todos los dispositivos que cumplen con las especificaciones del estándar:

- Función 1 Read Coil Status
- Función 2 Read Input Status
- Función 3 Read Holding Registers
- Función 4 Read Input Registers
- Función 5 Force Single Coil
- Función 6 Preset Single Register
- Función 15 Force Multiple Coils
- Función 16 Force Multiple Registers

**Función 1 o 2 (1 Leer estado de bobina - 2 Leer estado de entrada):** Permite realizar la lectura del estado de las DIs (el comando 2-Read input status) o DOs (el comando 1-Read Coil Status). Para ello el maestro solicita el número de bits que desea leer a partir de una determinada dirección. Cada dirección se corresponde con un registro de 1 bit con el estado de la entrada digital. El esclavo responde indicando el número de bits que retorna y sus valores. En la trama de respuesta se aprovechan todos los bits del byte, y puede haber hasta 256 bytes [1].

**Función 3 o 4 (3 Leer registros Holding – 4 Leer registros de entrada):** Permite realizar la lectura del valor de las AIs (@4XXXX el comando 3 Read Holding Registers) o AOs (@3XXXX el comando 4 Read Input Registers). El máster indica la dirección base y número de palabras a leer a partir de esta, mientras que el esclavo indica en la respuesta el número bytes retornados, seguido de estos valores. Aunque en realidad se está escribiendo en el rango de registros o valores numéricos, los registros son direccionados a partir de la dirección 0 (así el registro @40001 se direcciona 0) [1].

**Función 5 (Fosar Single Coil):** Permite modificar el estado de una DO del esclavo (mando o relé). Es decir mediante este comando podemos modificar algún bit de alguna de las variables internas del esclavo u ordenar la ejecución o activación de

un mando. Actúa sobre la zona de memoria de los DOs. El Maestro especifica la dirección del bit o mando que quiere modificar seguido de 0x00 para ponerlo a 0 o 0xFF para ponerlo a 1. El esclavo responde con una trama similar indicando la dirección que ha modificado y el valor que ha establecido en el bit o mando [1].

**Función 6 (Registros simples prestablecidos):** Permite la escritura en las AOs del esclavo (ya sea una señal o valor interno del equipo), y por tanto actúa sobre la zona de memoria de las AOs. Deberemos indicar la dirección del valor que queremos modificar y la magnitud que queremos asignarle. Luego el esclavo debería responder con la dirección del dato que ha modificado y el valor que le ha asignado, que debería coincidir con el enviado. Aunque en realidad se está escribiendo en el rango de AOs, los registros son direccionados a partir de la dirección 0 (así el registro se direcciona 0) [1].

**Función 15 (Forsar múltiples bobinas):** Permite la modificación simultanea de varios bits de DOs en el esclavo, pasándolos a OFF (‘0’) o a ON (‘1’) según convenga. Actúa sobre la zona de memoria de las DOs. Así en el comando se pasan la dirección inicial (dirección del primer bit o mando a modificar) y la cantidad y estado de cada uno de los sucesivos mandos (bits) a modificar [1].

**Función 16 (Múltiples registros prestablecidos):** Permite realizar la escritura en un grupo de AOs, y por tanto actúa sobre la zona de AOs. Se debe especificar la dirección a partir de la que queremos comenzar a actualizar valores, el número de valores que queremos actualizar, y la lista de valores que queremos asignar a estos registros. Aunque se está escribiendo en el rango de registros o valores numéricos, los registros son direccionados a partir de la dirección 0 (es decir el registro 40001 se direcciona 0) [1].

### 2.7 Protocolo MQTT

MQTT fue inventado y desarrollado por IBM a finales de los 90. Su aplicación original era vincular sensores en oleoductos de petróleo a satélites. Tal como sugiere su nombre, se trata de un protocolo de mensajería con soporte para la comunicación asíncrona entre las partes. Un protocolo de sistema de mensajes asíncrono separa al emisor y al receptor del mensaje tanto en el tiempo como en el espacio y, por lo tanto, es escalable en ambientes de red que no sean de confianza. Pese a su nombre, no tiene nada que ver con colas de mensajes; en realidad, utiliza un modelo de publicación y suscripción [2].

#### 2.7.1 ¿Qué es MQTT?

MQTT es un protocolo de red leve y flexible que ofrece el equilibrio ideal para los desarrolladores de IoT [2]:

- Un protocolo liviano permite la implementación en hardware de dispositivos altamente restringidos y en redes de ancho de banda limitada y de alta latencia.
- Su flexibilidad hace posible el suporte a diversos escenarios de aplicación para dispositivos y servicios de IoT.

### 2.7.2 ¿Cómo funciona MQTT?

El protocolo MQTT define dos tipos de entidades en la red: un bróker de mensajería y numerosos clientes. Bróker es un servidor que recibe todos los mensajes de los clientes y, en seguida, redirige estos mensajes a los clientes de destino relevantes. Un cliente es cualquier cosa que pueda interactuar con el bróker y recibir mensajes. Un cliente puede ser un sensor de IoT en campo o una aplicación en un centro de datos que procesa datos de IoT [2].

1. El cliente se conecta al bróker. Puede suscribirse a cualquier "tema" de mensajería del bróker. Esta conexión puede ser una conexión TCP/IP simple o una conexión TLS cifrada para mensajes sensibles.
2. El cliente publica los mensajes en un tema, enviando el mensaje y el tema al bróker.
3. Después, el bróker remite el mensaje a todos los clientes que se suscriben a este tema.

Como los mensajes de MQTT se organizan por temas, el desarrollador de aplicaciones tiene la flexibilidad de especificar que determinados clientes solo pueden interactuar con determinados mensajes. Por ejemplo, los sensores publicarán sus lecturas en el tema "sensor_data" y se suscribirán al tema "config_change". Las aplicaciones de procesamiento de datos que guardan los datos del sensor en una base de datos de backend se suscribirán al tema "sensor_data". Una aplicación de consola administrativa podría recibir comandos del administrador del sistema para ajustar las configuraciones de los sensores, tales como la sensibilidad y la frecuencia de muestreo, y publicar dichos cambios en el tema "config_change" (ver figura 2.28) [2].
Al mismo tiempo, MQTT es liviano. Tiene un encabezado simple para especificar el tipo de mensaje, un tema basado en texto y, posteriormente, una carga útil binaria arbitraria. La aplicación puede tener cualquier formato de datos para la carga útil, como JSON, XML, binario cifrado o Base64, siempre que los clientes de destino puedan analizar la carga útil [2].

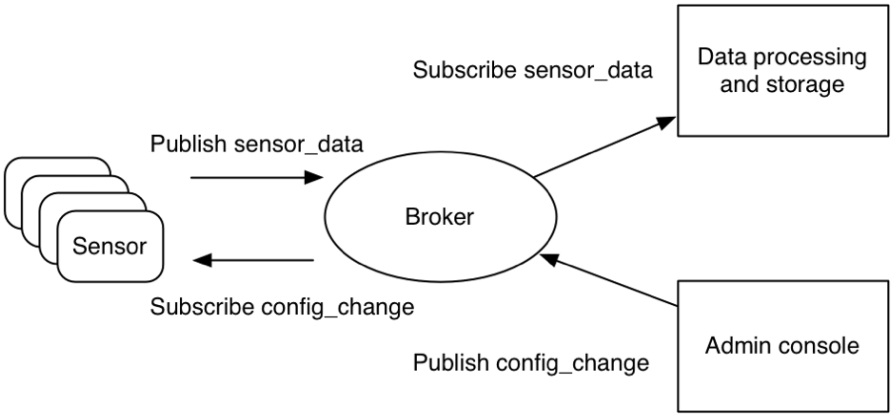



**Figura 2.29** El modelo de publicación y de suscripción de MQTT para sensores de IoT [2].

## 2.8 Protocolo S7

S7comm (S7 Communication) es un protocolo propietario de Siemens que funciona entre controladores lógicos programables (PlCs) de la familia Siemens S7-300/400 [16].

Se utiliza para la programación de PLC, el intercambio de datos entre PICs, el acceso a datos PLC desde SCADA (control de supervisión y adquisición de datos) sistemas y fines de diagnóstico [16].

### 2.8.1 Breve historia del s7

El protocolo es utilizado por Siemens desde el lanzamiento de la serie de productos Simatic S7 en 1994. El protocolo también se utiliza sobre otras capas físicas/de red, como RS-485 con MPI (Interfaz multipunto) o Profibus [16].

### 2.8.2 Estructura de trama de datos

Los datos en el protocolo S7comm vienen como carga útil de paquetes de datos COTP. El primer byte es siempre 0x32 como identificador de protocolo. Los procesadores de comunicación especiales para la serie S7-400 (CP 443) pueden utilizar este protocolo sin las capas TCP/IP.

**Tabla III** Protocolo S7 [16].

| | **Capa OSI** | **Protocolo** |
|---|---|---|
| 7 | Capa de aplicación | Comunicación S7 |
| 6 | Capa de presentación | Comunicación S7 |
| 5 | Capa de sesión | Comunicación S7 |
| 4 | Capa de transportación | ISO en TCP (RFC 1006) |
| 3 | Capa de red | IP |
| 2 | Capa de enlace de datos | Ethernet |
| 1 | Capa fisca | Ethernet |

### 2.8.3 Pasos generales para establecer comunicación a un PLC por S7

De manera general, los pasos para establecer un conexión a un PLC por el protocolo S7, son 3 y se muestran en seguida:

1. Conectar al PLC mediante TCP por el puerto 102.
2. Conectarte a la capa ISO (Requerimiento de conexión COTP)
3. Conectarse a la capa de comunicación S7.

Paso 1) usa la dirección IP del PLC

Paso 2) utiliza una destinación TSAP con una longitud de dos bytes. El primer byte del código del TSAP del destinatario contiene el tipo de comunicación (1 = PG, 2 = OP). El segundo byte contiene el número del rack y el slot: Este es la posición del CPTU del PLC. El número del slot está codificado en los bits 0-4, y el numero el rack está codificado en los bits 5-7.

Paso 3) en esta parte se realiza la negociación de los detalles específicos del protocolo, como el tamaño de la trama de los datos de comunicación.

## 2.7 Simulador Factory IO

### 2.7.1 ¿Qué es Factory IO?

Factory IO es una simulación de fábrica en 3D para tecnologías de automatización del aprendizaje (ver figura 2.29). Diseñado para ser fácil de usar, permite construir rápidamente una fábrica virtual utilizando una selección de piezas industriales comunes. La E/S de Factory también incluye muchas escenas inspiradas en aplicaciones industriales típicas, que van desde niveles de dificultad principiantes hasta avanzados. El escenario más común es utilizar Factory IO como una plataforma de entrenamiento de PLC ya que PLC son los controladores más comunes que se encuentran en aplicaciones industriales. Sin embargo, también se puede utilizar con microcontroladores, Softplc, Modbus, entre muchas otras tecnologías [16].

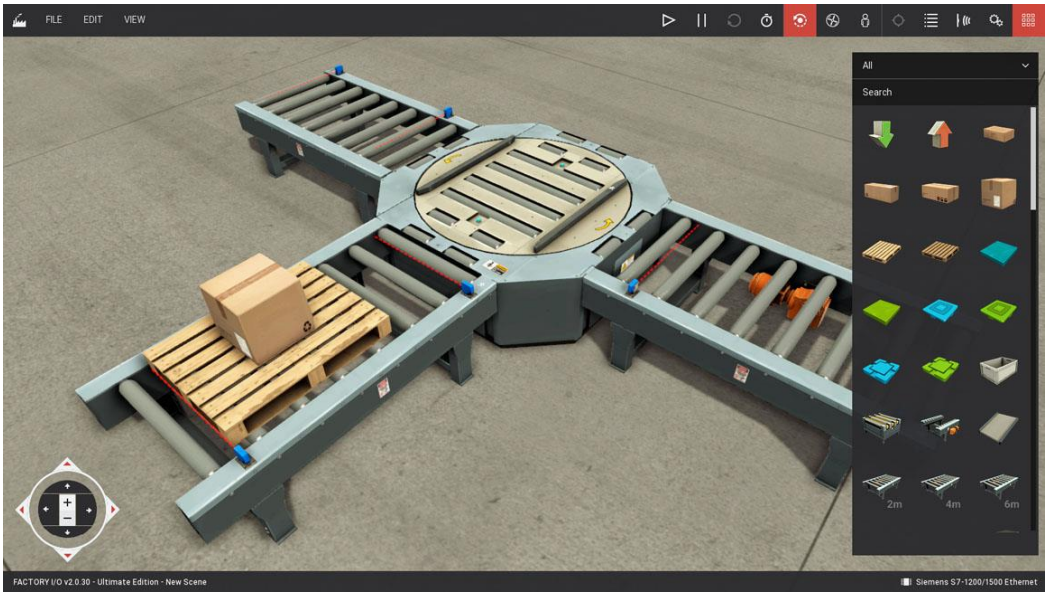



**Figura 2.30** Ambiente de Factory IO [16].

### 2.7.2 características de Factory IO

- Escenas industriales típicas predeterminadas.
- Librería de partes industriales
- Creación de escenarios propios
- Dispositivos de entrada y salida, tanto analógicos como digitales.
- Controladores de entrada y salida.
- Posibilidad de inyectar errores a los escenarios.
- Conexión con el mundo exterior a partir de diferentes protocolos de comunicación.

# CAPÍTULO 3
# ESTADO DE LA TÉCNICA Y DEL ARTE

# Capítulo 3 Estado de la técnica y del arte.

## 3.1 Estado de la técnica

### Nacional

**Título de la invención:** sistema de comunicaciones inalámbricas industriales [16].
**Inventores:** kazuhiro Ishikawa [jp]; Koji Kunii [jp]; Norimasa ozaki [jp]; Shengcong wu [cn]; tomohiko aki [jp]; toshiaki kuwahara [jp]; yoshihiro nozaki [jp]
**Resumen de la invención:** Un sistema de comunicaciones inalámbricas industriales incluye un controlador lógico programable (PLC) que lleva a cabo al menos monitorización dentro de un instalación industrial, al menos un dispositivo inalámbrico principal conectado al PLC por un bus de campo, una pluralidad de dispositivos inalámbricos esclavo, que se instalan correspondientemente a dispositivos de hardware respectivos, y llevan a cabo comunicaciones inalámbricas con el dispositivo inalámbrico principal, una unidad de procesamiento de conexión que lleva a cabo un proceso de conexión de forma inalámbrica entre el dispositivo inalámbrico principal y los dispositivos inalámbricos esclavo, y una unidad de procesamiento de transmisión/recepción que transmite y recibe datos de forma inalámbrica entre el dispositivo inalámbrico principal y los dispositivos inalámbricos esclavo.

### Internacional

**Título de la invención:** Distributed industrial performance monitoring and analytics [17].
**Inventores:** Nixon; Mark John, Enver; Alper Turhan, Bell; Noel Howard, Kidd; Joshua Brian, Muston; Paul R.
**Resumen de la invención:** Se proporcionan sistemas y métodos de monitoreo y análisis de procesos industriales distribuidos para su operación dentro de una planta de proceso. Una pluralidad de motores de datos distribuidos (DDE) puede ser incorporada dentro de la planta de proceso para recoger y almacenar datos generados por fuentes de datos, tales como PLC's. Por lo tanto, los datos pueden almacenarse de manera distribuida en los DDE integrados en toda la planta de proceso. Los DD pueden conectarse mediante una red de análisis de datos para facilitar la transmisión de datos mediante suscripción o consulta. Los DD pueden configurarse como una pluralidad de clusters, que pueden incluir clusters locales y centralizados. Las agrupaciones locales pueden obtener datos en streaming de fuentes de datos y transmitir datos seleccionados a un consumidor de datos. El grupo centralizado puede registrar los grupos locales, recibir datos de ellos y realizar funciones de análisis de datos sobre los datos recibidos. Los datos analizados pueden enviarse posteriormente a un consumidor de datos.

## 3.2 Estado del arte

### Nacional

**Título de la tesis:** Propuesta de un sistema de monitoreo de energía eléctrica para transformadores de 20 MVA [18].
**Autores:** Daza de Jesús, José Luis; Ubaldo Romero, Marcos Uriel
**Resumen:** En esta tesis se propone un sistema de monitoreo para transformadores de potencia en un industrias que le permita conocer las variables de mediciones eléctricas, el aprovechamiento de la energía suministradas con respecto a la otorgada por CFE. Para proponer el sistema de monitoreo fue necesario conocer los instrumentos auxiliares los cuales son los transformadores de corriente y tensión dado, que a esta subestación se encuentra alimentada a altos niveles de tensión y corriente, en donde un analizador de redes n puede conectarse directamente. La selección de estos dispositivos se realizó en base a normas nacionales e internacionales estandarizadas, para la parte eléctrica se tomó en cuenta diferentes factores tales como altitud, nivel de contaminación, temperatura y lugar del diseño. A través de la conexiones de los instrumentos de medición se configuraron los parámetros a medir más importantes dependiendo las características de relación de transformación, así como los ajustes de demanda de potencia, energía y la parte de monitoreo de la energía en valores mínimo, instantáneo, medio y máximo. Para complementar este proyecto, se plantea que es la parte de protocolo de comunicación, la cual consistió en interconectar los equipos de medición a un ordenador PC para realizar el monitoreo desde la oficina dedicada para esto, sin la necesidad de estar en la subestación realizando mediciones. La comunicación se logró a través de una red Ethernet o red Modbus, operando las 24 horas o el tiempo de operación que demande esta industria el trabajo.

**Título de la tesis:** Internet de las cosas aplicado a la manipulación remota de un variador de frecuencia [19].
**Autores:** Mariscal Organista, Adonay Omar; Manjarrez Sánchez, Ailyn
**Resumen:** Se realizó la comunicación entre un variador de frecuencia y una computadora de placa reducida para manipular desde una interfaz web un variador de frecuencia, esto con el fin de aplicar Internet de las cosas en el Banco de pruebas y tuberías del Laboratorio de Ingeniería Térmica e Hidráulica Aplicada (LABINTHAP) y adaptar nuevas tecnologías. Primero se entendió el funcionamiento del banco de tuberías que consta de una bomba centrífuga que succiona agua limpia de un contenedor y lo lleva a través de las tuberías. En la tubería principal está una placa de orificio para la obtención del flujo. Se comprobó la relación proporcional entre las rpm y el flujo generado. Posteriormente se reconoce el funcionamiento del variador

de frecuencia para ubicar los puertos y los protocolos de comunicación con los que cuenta. Una vez identificados, se adaptó un módulo RS-485 al variador para obtener la comunicación vía Modbus. En la computadora de placa reducida Orange Pi, utilizando el lenguaje de programación llamado Python, el cual cuenta con características como ModBus y acceso a la web, se desarrolló lo necesario para generar la comunicación y lograr una manipulación remota. La manipulación del variador de frecuencia desde una distancia de hasta 67 kilómetros metros fue exitosa. Mandando la información por medio de una interfaz web donde se solicita el flujo deseado y se muestra la frecuencia a la que el variador se encuentra para generar el flujo solicitado. Se demuestra como el Internet de las cosas integra dispositivos a la red antes no incluidos, comprobando que se tendrán más de 50 billones de dispositivos conectados para el 2020, según los estudios realizados por Cisco IBSG en el 2011.

**Internacional**

**Título de la tesis:** Monitoring System In Industry Using IoT [20].
**Autores:** Jagadesh M, Saravan M, Narayanan V, Priya Vadhana.
**Resumen:** En el mundo web de hoy la innovación de IoT se ha ampliado definitivamente. IoT se utiliza para asociar es una mezcla de marco de correspondencia y marco insertado. IoT se utiliza para transmitir y aceptar información. Estos marcos se utilizan para cribar máquinas mecánicas. En este procedimiento se observan a pequeña escala aplicaciones modernas como control de dimensión de fluidos, control de vitalidad, etc. El pensamiento fundamental de este documento es esbozar el significado de IoT para cribar aplicaciones mecánicas.

**Título de la tesis**: monitoring and control of plc based automation system Parameters using iot [21].
**Autores:** Somesh Bakshi, Gaurav Khairmode, Nilesh Varkhede, Swapnil Ayane.
**Resumen**: Las máquinas de proceso industrial trabajan principalmente en sistemas basados en PLC y SCADA. Los PLC están equipados con muchos sensores, guardia de seguridad electrónica, conectores. PLC es un ordenador digital industrial adaptado para el control de procesos de fabricación como líneas de montaje o dispositivos robóticos. Toda la información recogida de los sensores se acepta como señales digitales a los PLC. Esta información está disponible en el control de supervisión y adquisición de datos (SCADA) a través del protocolo RS485. Internet de las cosas (Iot) está aumentando rápidamente la tecnología. Iot es la red de objetos físicos o cosas embebidas con electrónica, software, sensores y conectividad de red, que permite a estos objetos recoger e intercambiar datos. En este proyecto, estamos desarrollando un sistema que automáticamente monitoreará

las aplicaciones industriales y generará alertas/alarmas o tomará decisiones inteligentes usando el concepto de Iot. Iot nos ha dado una manera prometedora de construir sistemas y aplicaciones industriales de gran alcance mediante el uso de dispositivos inalámbricos y sensores. Diseña e implementar un sistema de monitoreo de datos de controladores lógicos programables (PLC) basado en web en tiempo real. Una contribución principal de este proyecto es que resume los usos de la Iot en industrias con Inteligencia Artificial para monitorear y controlar la Industria.

**Título de la tesis:** IOT Application for Real-time Monitor of PLC Data using EPICS [22].
**Autores:** Ramesh Joshi, H M Jadav, Aniruddh Mali, S V Kulkarni.
**Resumen:** Un gran interés reciente en la comunicación Máquina a Máquina se conoce como Internet de las cosas (IOT), para permitir la posibilidad de que los dispositivos autónomos utilicen Internet para intercambiar los datos. Internet y la World Wide Web han causado una revolución en la comunicación entre las personas. Nacieron de la necesidad de intercambiar información científica entre instrumentos. Control System Studio (CSS) (Best Operator Yet Operator Interfaz) BOY OPI no es sólo una aplicación, sino también un marco de trabajo que se puede ampliar con widgets y fuentes de datos. El framework proporcionó implementaciones para todas las funcionalidades comunes, como lector y escritor de archivos XML, manejo de conexiones PV, widgets abstractos, propiedades, OPI Runtime y OPI Editor. El marco puede ampliarse utilizando los puntos de ampliación que ofrece el marco. Los sistemas de monitoreo CSS son extremadamente importantes en Física Experimental y Sistemas de Control Industrial (EPICS). La mayoría de ellos se basan en cliente/servidor (C/S). Este trabajo diseña e implementar un sistema de monitoreo de datos de controladores lógicos programables (PLC) basado en web en tiempo real sobre datos EPICS. Este sistema se basa en el navegador y el servidor (B/S). Utilizando MODBUS/TCP los datos de comunicación se han archivado en EPICS. Entonces todos los datos se muestran en una tabla de tiempo real en el navegador (Internet Explorer o Firefox/Mozilla). El gráfico se actualiza cada intervalo regular y se puede ampliar y ajustar. También, proporciona consejos de datos mostrando y modo de pantalla completa. La adquisición de los datos sería manejada por la tarjeta de adquisición de datos múltiples que ha sido cableado comunicación con PLC utilizando 24 VDC a 5 VDC y viceversa circuito electrónico.

**Título de la tesis:** Development of an Internet of Things Gateway Applied to a Multitask Industrial Plant [23].
**Autores:** Fabricio Tietz, Dennis Brandao, Luiz Ferreira Alves.

**Resumen:** La evolución y difusión de las tecnologías de la información y la comunicación ha llegado al entorno industrial y ha permitido el intercambio de información entre unidades de fabricación completas, la integración de la cadena de producción, la agilidad en la adquisición de datos y la decisión automáticahacer - este paradigma es etiquetado como la Cuarta Revolución Industrial. En este contexto, este trabajo introduce una pasarela Internet of Things aplicada a una planta industrial multitarea que contiene dispositivos con diferentes protocolos de comunicación, como PROFIBUS, HART, Ethernet y MODBUS. La pasarela se basa en hardware y software que se comunican con el equipo industrial a través de los protocolos Siemens S7 y Modbus y con un servidor desarrollado en una plataforma de computación en nube a través del protocolo de telemetría de consultas de mensajes. Esta solución permite el monitoreo y control de las variables de proceso de la planta a través de una interfaz gráfica en línea. Por otra parte, la medición del tiempo dedicado a leer los datos de los Controladores Lógicos Programables y su actualización en la interfaz online muestra que la comunicación entre el servidor y el sistema físico presenta baja latencia.

# CAPÍTULO 4
# INFORME TÉCNICO

## Capítulo 4 Informe técnico

En este capítulo se mencionarán las diferentes pruebas que se realizaron con el PLC LOGO!, así mismo, se describirá brevemente cada uno de los programas realizados para cada prueba, y al igual de lo que se haya desarrollado para las pruebas. De igual manera se mostrarán las diferentes dashboards que se realizaron en cada prueba.

### 4.1 Conexión del PLC LOGO! a Node-RED mediante Modbus TCP/IP, envío de datos a Ubidots por MQTT y prueba de conexión con dispositivos externos.

La primera prueba fue realizada con el protocolo Modbus TCP/IP entre PLC LOGO! y Gateway (raspberry). Se tomaron 3 diferentes tipos de dispositivos para la prueba de intercambio de datos, entre los cuales fueron focos, para la prueba de las salidas de relevador del PLC LOGO!, y para las entradas, unos simples interruptores. Además se conectó una celda de carga para el envió de datos al PLC LOGO! desde las raspberry, esto a través de una placa desarrollada la cual se encarda de enviar los datos por medio de Wifi al Gateway, para que después este los mande al PLC LOGO!, y a las diferentes dashboards. El intercambio de datos también se dio entre el dashboard local (Dashboard en Node-RED) y la dashboard remota (Ubidots), la siguiente figura 4.1.

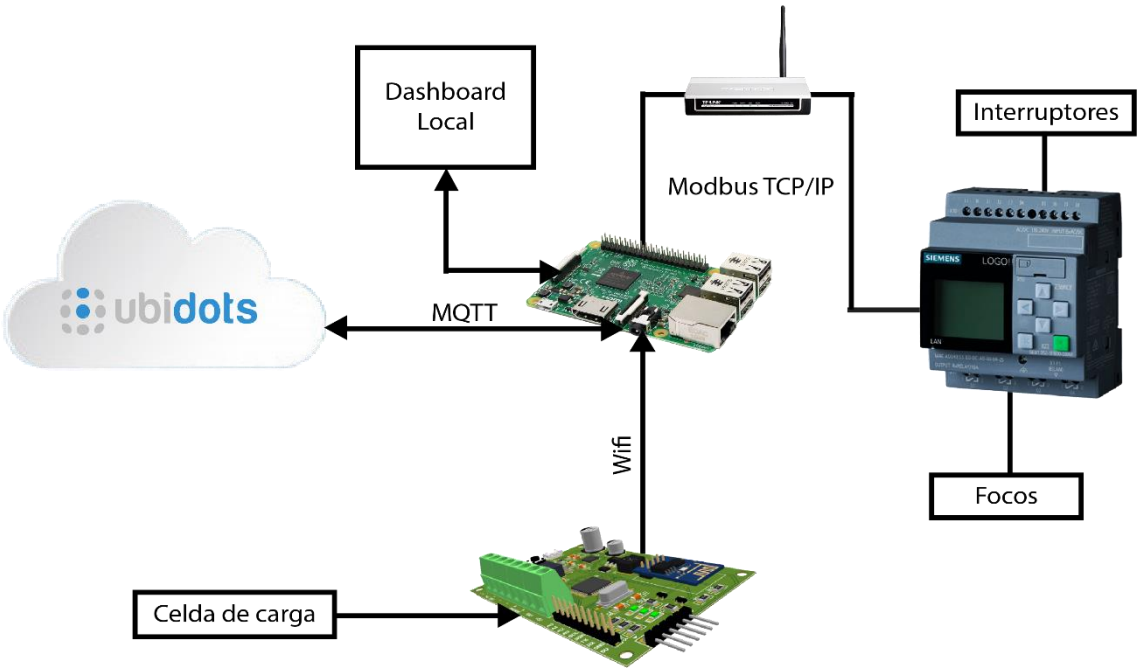



**Figura 4.1** Diagrama de la prueba con el protocolo Modbus TCP/IP.

### 4.1.1 Placa de conversión de protocolos a Wifi.

Para la aplicación de la primer prueba fue necesario la realización de la siguiente placa electrónica (ver figura 4.2), la cual tiene como propósito fundamental la conversión de diferentes protocolos cableados a wifi, esto con la finalidad de poder mandar los datos al Gateway, para de esta manera distribuirlos a donde sea necesario.
Este dispositivo consta en un microcontrolador de microchip, sus respectivos botones de boot y reset, así como un módulo de wifi ESP8266. Puede ser alimentado de 0  a 24 V máximo. Se han dejado 10 pines hembra de 2.54 mm en los cuales se encuentran 4 pines de propósito general (A0-A3), SCL, SDA, TX, DX y alimentación de 5V. Hay 4 terminales, en las cuales una es para la alimentación, otra para VIN, para alimentación de 5V y finalmente una para el MAX485. La placa consta de los pines de ICSP para poder hacer pruebas al microcontrolador.

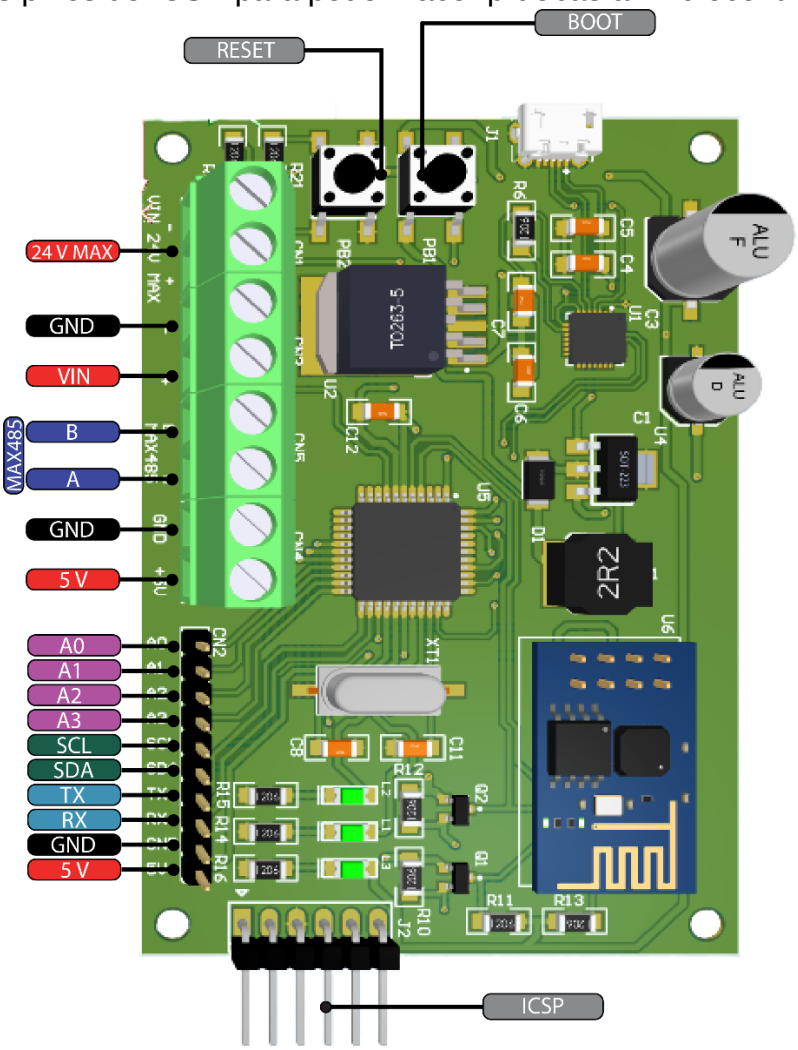



**Figura 4.2** Placa conversora de protocolos a Wifi.

### 4.1.2 Descripción general del programa realizado en LOGO! Soft Comfort para la prueba de Modbus TCP/IP.

Para poder utilizar el protocolo Modbus TCP/IP dentro del PLC LOGO!, fue necesario realizar un proyecto de red, en el cual se da dio de alta tanto a PLC LOGO! como al Gateway (raspberry) ambos con su respectiva IP (ver figura 4.3). Al tener registrados ambos dispositivos, fue necesario hacer la conexión entre ambos, la cual se denota por la línea amarilla debajo de ellos. Al dar doble clic sobre esta línea amarilla, aparece una pantalla en la cual se tiene que dar de alta los registros de donde obtienes datos y a donde los mandarás, al igual es necesario especificar otros requisitos necesarios para poder utilizar adecuadamente el protocolo Modbus TCP/IP,

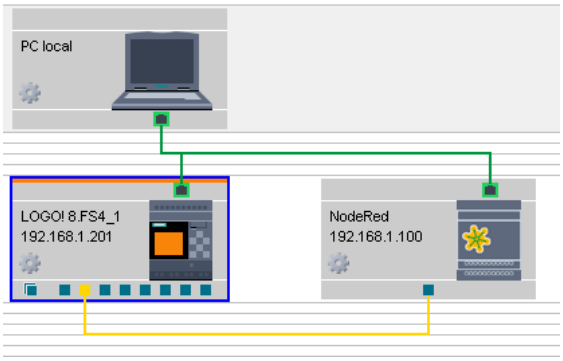



**Figura 4.3** Proyecto de red.

Una vez dado de alta todos los registros necesarios, ya es posible el uso de los datos. Estos datos que vendrán o que se mandarán, tienen que pasar por un registro específico del PLC LOGO!, el cual previamente ya registraste. Teniendo en cuenta el tipo y número de registro se puede utilizar el dato, más la ayuda de los bloques de red.
Dentro de la figura 4.4 se encuentran los bloques de funciones utilizados para esta primer prueba. El programa está constituido por lo siguiente:

- Lectura del estado de las entradas I1, I2, e I3, y el almacenamiento de las mismas en registros V, para el envió al Gateway.
- Lectura de registros tipo VW, en el cual llega la señal de los diferentes dashboard para la activación de las salidas de relevador, o activación de la luminaria conectada a esas salidas.
- Aplicación de una simple secuencia de la salidas al alcanzar un valor umbral, proveniente del valor de la celda de carga que viene del Gateway, y que leemos en registros tipo VW.

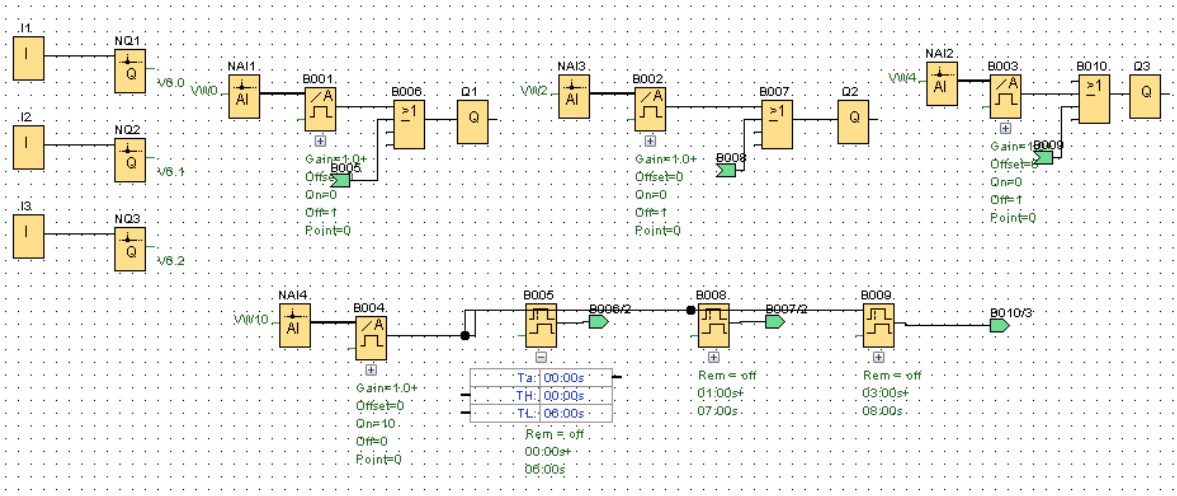



**Figura 4.4** Diagrama de funciones en LOGO! Soft Comfort para prueba uno.

### 4.1.3 Descripción general del Código a bloques en Node-RED para la función como Gateway y la transmisión de datos por Modbus TCP/IP.

Para poder obtener los datos del PLC LOGO! con Modbus TCP/IP fue necesario utilizar una paquetería especial en Node-RED. Esta paquetería contiene un nodo llamado Modbus Flex Server (nodo rojo), el cual se encarga de gestionar los registros para el envío y recepción de datos. Para poder realizar el envió de datos es necesario especificar el tipo y número de registro, el ID, y el dato a almacenar. Para recibir es necesario saber en qué tipo de registros está almacenado el dato, de esta manera se utiliza la salida correspondiente. Esta salida mandara todos los datos en forma de buffer lo cual se tiene que cambiar a un array para poder hacer uso de los datos. Punto importante, los datos vendrán desfasados por 8 posiciones dentro del buffer.

Con las configuraciones necesarias en el nodo para el envío y recepción de datos, ya se pudo tratar los datos provenientes de las diferentes dashboards (local y remota).

Todos los bloques naranjas corresponden a funciones escritas en JavaScript, con diferentes propósitos entre los cuales se encuentra la conversión de datos recibidos y la adaptación de formatos para el caso de envío por MQTT.

Lo bloques azules representan el envío de datos a cada elemento dentro del dashboard local.

Los bloques morados, son parte de la paquetería de MQTT en Node-RED, en los cuales se publica información, y se recibe de los tópicos a los cuales se está subscrito.

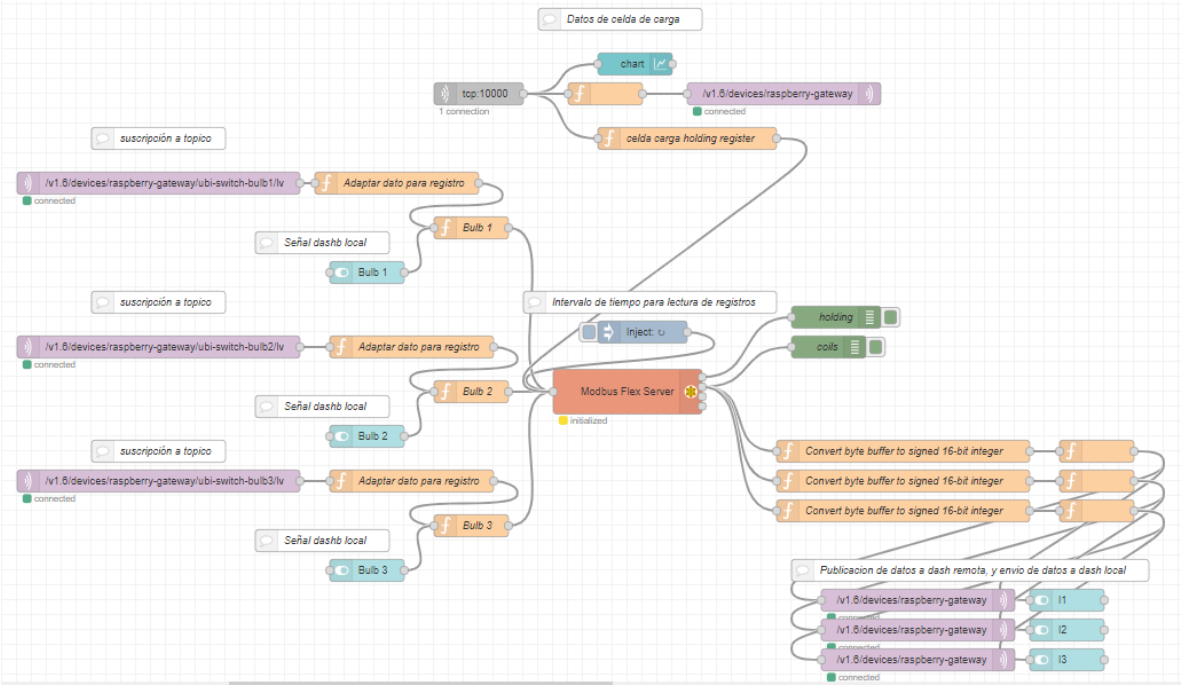



**Figura 4.5** Diagrama de nodos en Node-RED para prueba uno.

### 4.1.4 Dashboard local realizado para prueba con Modbus TCP/IP.

El dashboard local fue realizado con una paquetería en Node-RED.
En la figura 4.6, se muestra 3 recuadros diferentes, el primero muestra interruptores con los cuales se puede controlar los focos conectados a las salidas de relevador del PLC LOGO!. El segundo recuadro muestra un gráfico donde se ve el historial de datos que la celda de carga ha estado mandando a través de wifi. Por último se muestra un recuadro donde se puede ver el estado de las entradas del PLC LOGO! I1, I2, I3.

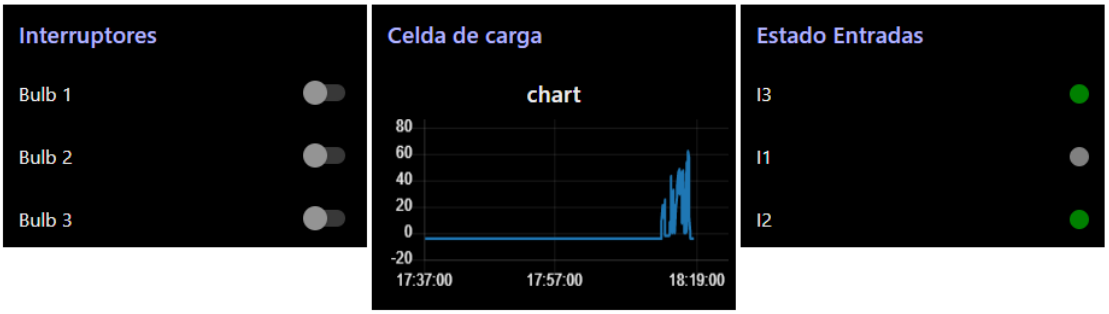



**Figura 4.6** Dashboard local de prueba uno.

### 4.1.4 Dashboard remoto en Ubidots para prueba uno.

El dashboard remoto fue implementado en la plataforma Ubidots. Los datos intercambiados entre la plataforma y Gateway fueron realizados por el protocolo MQTT. El bróker encargado fue el proporcionado por la misma plataforma. Así que simplemente se realizaba la publicación y suscripción a un tópico desde Node-RED, y en lo que respecta a la plataforma, tuve que crear un dispositivo nuevo, en el cual se iban a estar almacenando los tópicos a los que podía publicar o subscribirme. Después fue necesario realizar el dashboard (ver figura 4.7), en el cual se encuentra tres indicadores que se encargan de mostrar el estado de las entradas del PLC LOGO!, y tres interruptores para el control de las salidas del mismo. También se aprecia un gráfico en el cual se mostró los diferentes datos provenientes de la celda de carga.

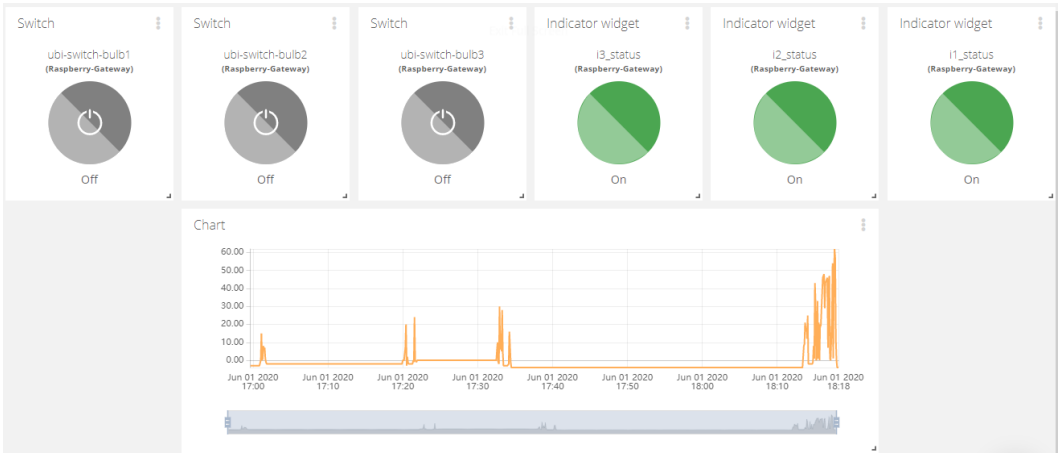



**Figura 4.7** Dashboard remoto de prueba uno.

### 4.2 Conexión del PLC LOGO! a Node-RED mediante protocolo S7, envío de datos a Ubidots por MQTT y prueba de conexión mediante simulación de proceso en Factory I/O

Para está segunda prueba pero ahora con el protocolo S7 para el intercambio de datos entre Gateway y PLC (ver figura 4.8), se implementó la simulación de un dispensado de material, sin embargo, el simulador usado no conto con los dispositivos necesarios gráficamente para realizar la implementación, por lo cual se hicieron adaptaciones con tanques de agua, usando todos los elementos que contaba cada uno, como válvulas de entrada, válvulas de salida, y el nivel de los mismos.

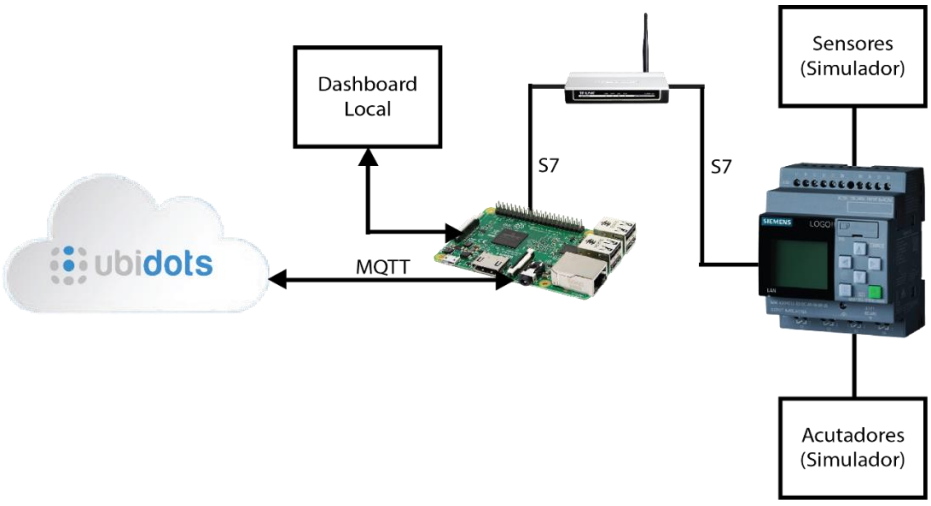



**Figura 4.8** Diagrama de la prueba con el protocolo S7.

### 4.2.1 Entorno de la simulación industrial

Como ya se mencionó anteriormente, el propósito de este entorno era diferentes, sin embargo, el simulador no tuvo los dispositivos necesarios, para lo cual se tuvo que hacer adaptación con los dispositivos que se encontraban. En este caso, como se puede apreciar en la figura 4.9, se presentan 5 tanques, de los cuales los tres que están completamente juntos, simulan tolvas de diferentes materiales para construcción, los dos tanques al fondo, uno simula agua como tal, y el otro un silo de cemento. La variable nivel se adaptó para simular la variable de cada dispositivo real.

Al igual se creó un tablero en el cual se encuentran luces indicadoras, que representan la activación de un elemento dentro del sistema. Así mismo, se presenta un selector para cambiar de modo manual a modo automático. En modo manual se puede tener control de todos los dispositivos de manera separada a través de un selector. En modo automático, se realiza la interacción con los dispositivos desde los dashboards.


**Figura 4.9** Entorno industrial dentro del simulador Factory I/O.

### 4.2.2 Programa en LOGO! Soft Comfort para el intercambio de datos entre Gateway-PLC-Simulador.

Ya que el programa estaba algo extenso este será mostrado en dos partes las cuales se muestran en la figura 4.10 y 4.11.
Dentro de la primera parte (ver figura 4.10) se encuentran marcadas 3 áreas especificas A, B, C. En el área A se encuentra la lectura de registros proveniente de los botones de inicio, paro, y aborto, tanto de Node-RED como del simulador. Así mismo en esta sección, están los registros del selector automático-manual, de donde se toman conexiones a los bloques de funciones necesarios.
En el área B, se encuentra el intercambio de datos entre registros para el posterior envío de datos al Gateway.
En la sección C, se encuentra la lectura de registros de los diferentes botones del simulador de procesos, después se encuentra el enclavamiento de estos, y también la configuración especifica de cada salida. Además se encuentran unos bloques lógicos para el prendido por secciones de cada una de las salidas.

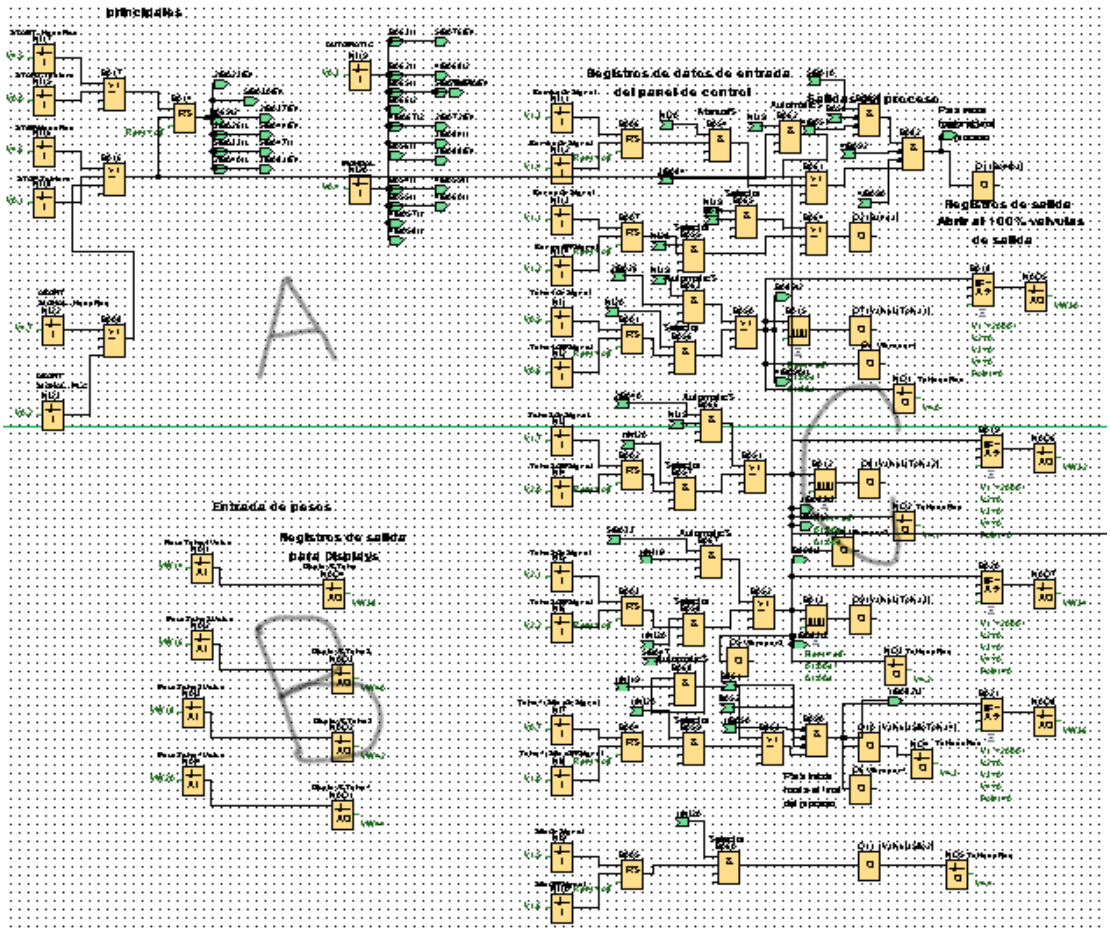



**Figura 4.10** Primera parte del programa con bloque de funciones para la prueba dos.

En la figura 4.11 se muestra la segunda parte del código a bloques de funciones.
En esta segunda parte no se enmarcaron áreas, puesto que la función de cada uno de los bloques es la misma. Esta función se encarga de hacer la entrega del pedido.
Lo primero que se realiza es la lectura de los registros en los cuales se encuentra el valor a dispensar, y la cantidad que se encuentra al momento en cada uno de los contenedores. Después cada uno de estos datos es pasado por una función matemática para determinar la cantidad a dispensar, sabiendo el dato a dispensar, se genera el valor umbral para los bloques de control de la salida. Este bloque solo espera la señal de iniciar, para arrancar el proceso de entrega del pedido.
Al final se agrega un bloque de lectura de datos, y la escritura de los mismos en registros para su posterior envío al Gateway.

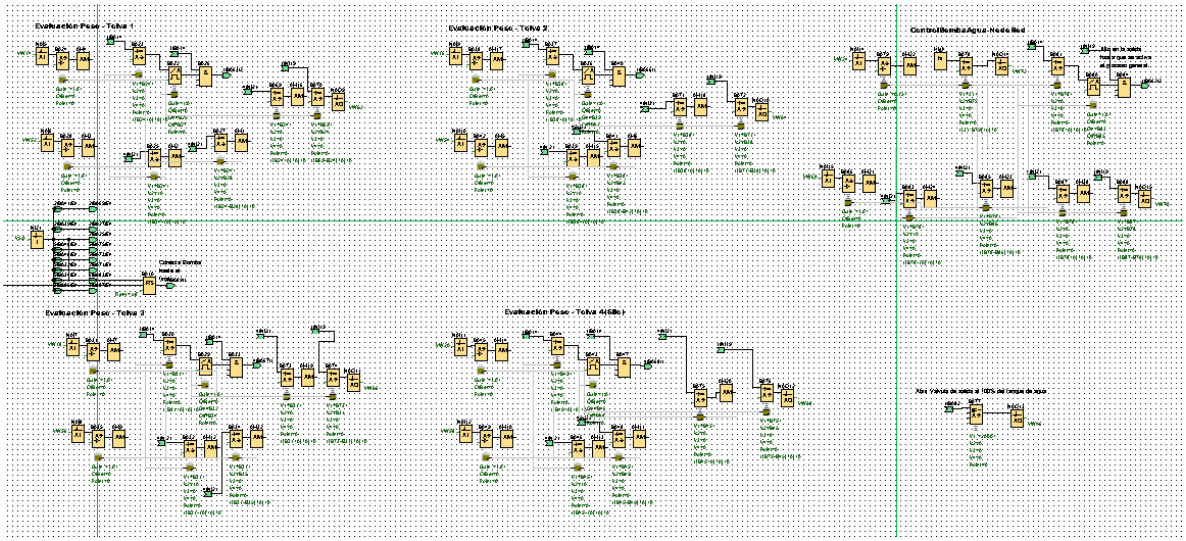


**Figura 4.11** Segunda parte del programa con bloque de funciones para la prueba dos.

### 4.2.3 Programa en Node-RED para el envío y recepción al PLC LOGO! y al dashboard local.

Para poder realizar la conexión entre la raspberry y el PLC LOGO! con el protocolo S7, fue necesario instalar la paquetería correspondiente para este protocolo.
La figura 4.12 muestra el código, el cual se ha seccionado para una mejor referencia.
En la sección A se encuentra la lectura de los registros encargados de proporcionar el estado de la bomba y banda. Estos datos son enviados al dashboard local y al dashboard remoto, este último se publica a un tópico en el bróker de Ubidots. Para lo cual se requiere de una función, para adaptar el formato.
En la sección B, tenemos las adquisición de datos de los pesos de material en cada tolva, al igual que en la sección anterior, estos datos son enviados a ambos dashboards, con sus funciones para adaptar el valor al formato correspondiente.

En la sección C, se encuentra los nodos que subscriben a un tópico, en este caso provenientes del bróker de Ubidots, que corresponden a los botones de inicio, alto y aborto. Este dato se manda por S7 al PLC LOGO!. También se tiene otro nodo enlazado con el dashboard local, por el cual se envía el estado del interruptor.
Las secciones D, E, G, H, realizan la misma función que la mencionada en la sección A y B. Se lee el valor del registro en la raspberry, previamente enviado del PLC LOGO! por el protocolo S7. Con el valor en el Gateway, este se manda a los dos dashboards. Para el dashboard local simplemente se conecta un nodo, que manda directo el dato. Para el dashboard remoto se tiene que realizar un publicación a un tópico, a través del nodo en color morado, como se ilustra en la figura. Previamente es necesario adaptar el valor al formato requerido a través de un nodo de función.
La sección F, se encarga de estar subscrito a los tópicos correspondientes al contexto de pedido. Esta suscripción es realizada con los bloques color morado. El dato recibido cuando existe una publicación es mandado al registro del Gateway, para posteriormente ser enviado al PLC LOGO! por medio del protocolo S7.
También se encuentra otro bloque azul, correspondiente al dashboard local, este proporciona el valor al bloque que hace el almacenamiento en el registro correspondiente. Al final se muestra un nodo tipo función, con el propósito de generar una variable global, para el análisis de los datos.

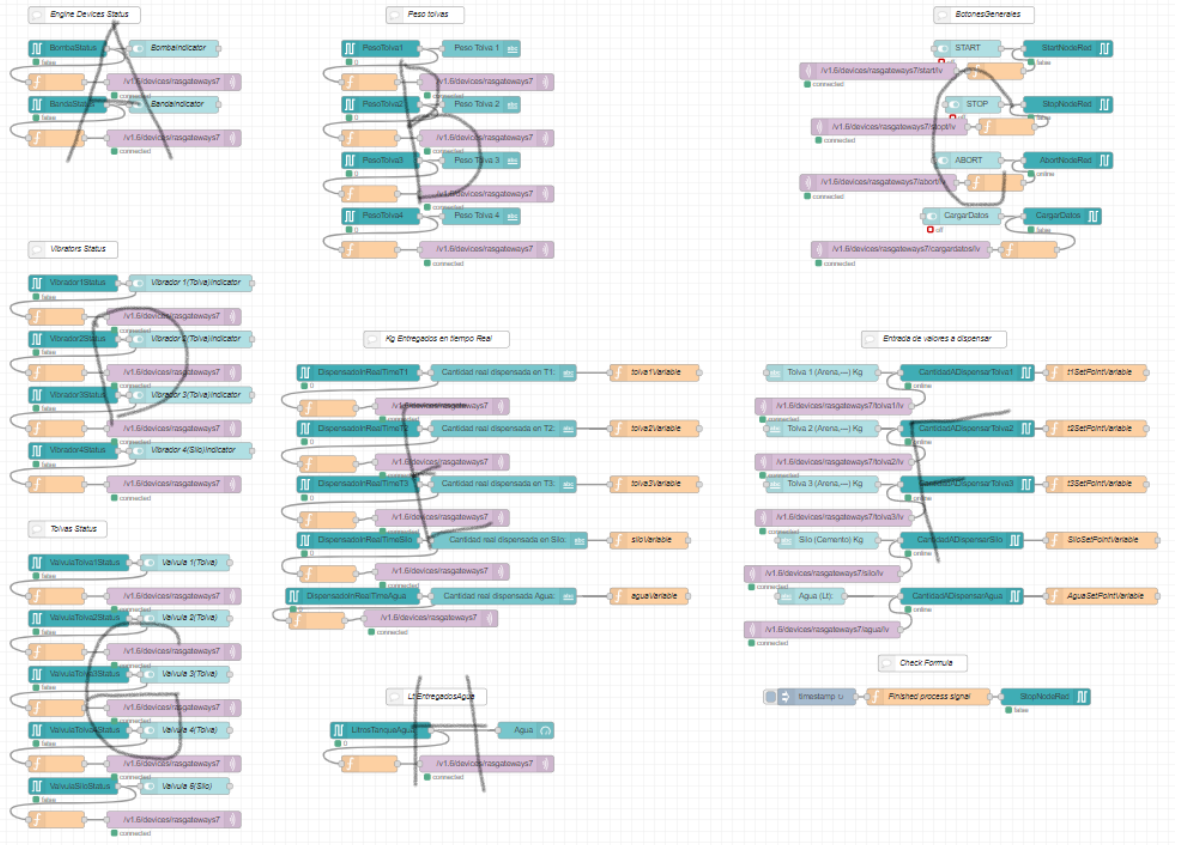



**Figura 4.12** Programa en Node-RED para prueba dos.

### 4.2.4 Dashboard local en Node-RED para prueba dos.

En la figura 4.13 se encuentra una imagen de lo que es el dashboard local, para esta prueba. En él se encuentran seis secciones diferentes: Estado de máquinas, estado de vibradores, estado de tolvas, la cantidad de material en cada contenedor, creación de un pedido, y pedido.
En la sección de estado de máquinas se encuentra el estado de la bomba y la banda, cuando estás están activas, el circulo cambia a un color verde. Lo mismo sucede con las otras dos secciones, que son estado de vibradores y tolvas.
La cuarta sección muestra la cantidad de materia que se encuentra en cada contenedor, para así poder saber, de cuanto se dispone para hacer algún pedido.
La sección de crear pedido permite anotar los datos requeridos de cada tipo de materia. Ahí mismo se encuentran los botones, CargarDatos, el cual tiene como propósito cargar los datos a los registros del PLC LOGO!, y también se encuentran los botones de iniciar, parar y abortar el proceso.
La última sección que corresponde a pedido, se encarga de mostrar el valor real entregado de cada uno de los materiales después de haber sido entregados.


**Figura 4.13** Dashboard local para prueba dos.

### 4.2.5 Dashboard remoto en Ubidots para prueba dos.

Dentro de Ubidots se desarrollaron dos dashboards diferentes, uno para el estado de los dispositivos y la cantidad de material que se encuentra en los diferentes depósitos, y otro para realizar el pedido de ciertos materiales.
En la figura 4.14 encontramos el primer dashboard, el cual contiene once indicadores, cada uno correspondiente a los dispositivos en el sistema, al igual se encuentra una tabla en la cual se muestra la cantidad de materia en las tres tolvas y el silo, y finalmente un pequeño circulo que representa un tanque, y que muestra el nivel de agua del tanque del sistema.
En la figura 4.15 se muestra el segundo dashboard, encargado de realizar los pedidos de manera remota. En primera instancia, tenemos los tres interruptores:

inicio, paro y aborto. Posteriormente se muestran unos recuadros en los cuales, se es necesario escribir la cantidad a dispensar de cada tipo de material. Al escribir el dato, es necesario presionar el botón enviar, para que estos datos lleguen a Node-RED, y al tener todos es necesario cargar los datos a través de presiona el botón que se encuentra a la derecha de estos cuadros de llenado. Finalmente, tenemos una lista en la cual se muestra los nombres de los contenedores de cada material. Ahí se podrá mostrar la cantidad real entregada después de haber dispensado el pedido proporcionado.

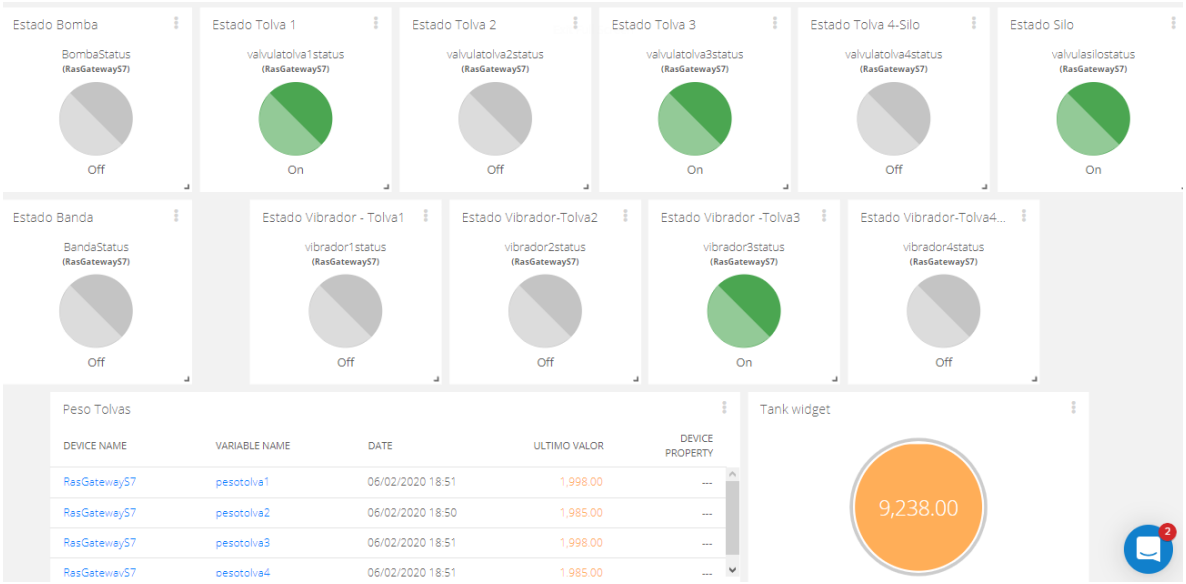



**Figura 4.14** Dashboard 1 remoto para prueba dos.

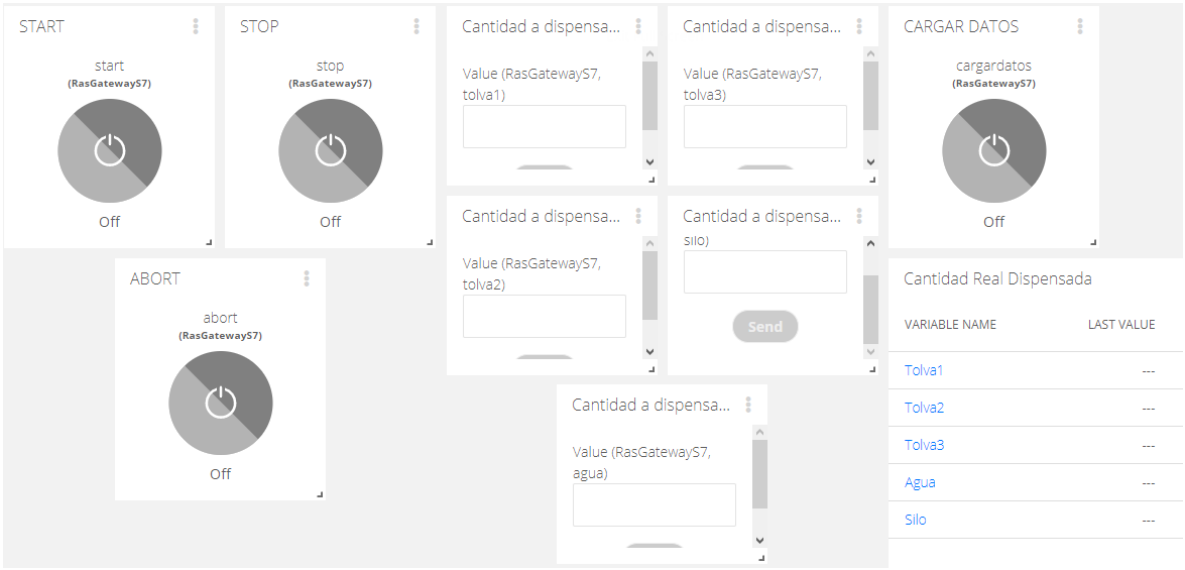



**Figura 4.15** Dashboard 2 remoto para prueba dos.

# CAPÍTULO 5
# CONCLUSIONES

# Capítulo 5 Conclusiones

## 5.1 Análisis de resultados

Durante el proceso de conexión del PLC LOGO! a plataforma IoT, se realizaron las pruebas con los dos protocolos que el dispositivo ofrece (S7 y Modbus TCP/IP). Para ambos casos se utilizó como Gateway una raspberry utilizando como entorno de programación Node-RED. En el cual se hacia la recolección de datos y el envió a la plataforma IoT (Ubidots) mediante el protocolo MQTT. De igual manera, se realizó un dashboard local para cada caso, dentro del entorno de programación de Node-RED.

**Prueba uno realizada con el protocolo Modbus TCP/IP y dispositivos externos conectados al PLC LOGO!.** En lo que respecta al PLC LOGO! el trabajo con los datos dentro de este protocolo fue muy bueno, sin embargo presento una limitante. La cantidad de datos a transferir es muy reducida, complicando la utilización del mismo en proyectos de gran magnitud. Otro de los problemas enfrentados, fue la superposición de datos en los registros, específicamente con los registros VW (2 bytes) y V(1 bit). La solución para esto fue aplicar un desplazamiento de la cantidad de registros usada en VW más uno, y desde ahí empezar a utilizar el otro tipo de registros.

El trabajo por parte del Gateway (Raspberry) utilizando los nodos para el protocolo Modbus TCP/IP, no fue bueno, mostro muchas complicaciones al momento de utilizar los nodos para este protocolo, debido a su poca documentación y al poco desarrollo de la librería. Al momento de la implementación, surgieron diversos problemas tales como, que el nodo utilizado para el envío y recepción de datos a los registros, presento una saturación aproximadamente de 4 minutos después de estar activo, creando una desconexión con el PLC LOGO!, y este tiempo se vio reducido conforme las cantidad de registros aumentaba. Para volver hacer la conexión con el PLC LOGO! tenía que ser reiniciado el PLC, creando tiempos de retraso.

El intercambio de datos entre las dashboard local tuvo una latencia muy baja, provocando un aprovechamiento óptimo de visualización de datos y envío de datos, sin embargo, el hecho de que ya esté predefinidos los bloques hace que tengas que adaptarte a un entono grafico no muy amigable.

El dashboard remoto mostro una buena fluidez, aunque la latencia fue un poco mayor comparada con la del dashboard local. El entorno grafico de los componentes dentro del entorno, fue de mejor visualización.

La placa realizada para la implementación de este sensor, funciono correctamente, los datos llegaron correctamente al Gateway. Sin embargo, el concepto de envió de datos desde el Gateway si es un poco lento, porque se suma el tiempo de envío de

la placa al Gateway más la latencia entre el Gateway y el PLC LOGO!, haciendo que el dato tarde más en llegar y poder trabajar con él.

**Prueba dos realizada con el protocolo S7 y una simulación de proceso.**

El PLC LOGO! de desempeño muy bien con este protocolo, además de que las configuraciones para usar el mismo dentro del software LOGO! Soft Comfort fueron más sencillas. La limitante de la cantidad de datos a transferir presentada con el uso del protocolo Modbus no se presentó para este caso, sin embargo, si presenta un límite, el cual es considerable para realizar un proyecto no tan grande, sin tener que recurrir a otro modulo adicional.

En lo que corresponde al Gateway (raspberry), presento un de desempeño muy bueno con el envío y recepción de datos con este protocolo S7, puesto que el desarrollo de la paquetería para este protocolo está más desarrollado, presentado más herramientas, con diferentes configuraciones, y un tratado de datos más simple.

La implementación del dashboard local en Node-RED funciono correctamente, el envío de datos y la recepción se presentó sin ningún retraso. Sin embargo, la parte visual deja mucho que desear aun, ya que no existe manera de que muevas los objetos en la posición que tu deseas, o la medida, para esto se requiere de la implementación de otros nodos en los cuales se tiene que programar cada elemento con Angular y HTML.

El dashboard remoto implementado en Ubidots, por lo contrario, mostro muchas fallas. Este funcionaba correctamente, mientras no se hicieran intercambio de datos con el protocolo S7, cuando el protocolo S7 hacia intercambios se desconectaba aproximadamente 6 segundos y después por uno 2 o 3 segundos se volvía conectar, y así actuaba mientras hubiera intercambio de datos con el otro protocolo. Tal vez esto se debe al incremento exponencial de los registros utilizados en las raspberry para el intercambio de datos con el PLC, y así mismo, el incremento de tópicos en el bróker de Ubidots. Así que fue imposible realizar, un pedido a mi proceso puesto que al iniciar el mismo, empezaba el intercambio de datos entre el PLC y el Gateway, tumbando la conexión al bróker inmediatamente.

El simulador de procesos realizo muy buen trabajo, sin embargo, aún le falta la implementación de más dispositivos industriales, para la creación de otros ambientes. En este caso del proyecto, fue necesario adaptar con lo que se tenía, tratando de recrear el proceso requerido, que sin duda alguno puedo asegurar que se acercó a un 80%. El intercambio de datos con el PLC se realizó por el mismo protocolo, S7, y la conexión fue muy simple, así mismo, el enlace de variables del sistema con el PLC.

El sistema implementado de dosificación, funciono correctamente desde el dashboard local, logrando realizar pedidos de dosificaciones específicas y mostrando al final, la cantidad exacta de producto. Al final esto es lo que se quiere, pero remotamente también, para cual será necesario hacer pruebas con otras plataformas, y de igual manera con otros protocolos.

## 5.2 Trabajos futuros

De acuerdo con las nuevas perspectivas visualizadas así como las debilidades encontradas durante el desarrollo de este trabajo se presentan algunas sugerencias para trabajos a futuro a fin de mejorar dicho proyecto.

**Utilización de otra plataforma para visualización remota.** Para la dashboard remota se utilizó Ubidots, una plataforma que si bien es muy buena, y presenta muchas funcionalidades, la cuestión de latencia es un poco alta en comparación con otros servicios del mismo tipo.

**Desarrollo de nodos propios del protocolo Modbus TCP/IP en Node-RED.** Dentro de las pruebas con los dos protocolos de comunicación (Modbus TCP/IP y S7), fue notoria la diferencia de las librerías en Node-RED. S7 tenía una librería más completa y mejor desarrollada. Es por eso por lo que para un mejor desarrollo de proyectos con Modbus TCP/IP, sería de mucho mejora la creación o mejora de las librerías existentes. Estas mejoras se desarrollarían en un lenguaje de programación llamado JavaScript.

**Creación de Dashboard local dinámico en Node-RED.** La implementación de un dashboard más personalizado y dinámico a través de la implementación de un bloque llamado *template*, el cual permite escribir código html y Angular. Esto permitirá de pantallas con una parte grafica mejor desarrollada, y una mejor distribución de los componentes dentro del entorno. Todo esto para una mayor experiencia para el usuario final.

**Placa conversora de Modbus RTU a Modbus TCP/IP.** La placa desarrollada en este proyecto ha sido de Modbus RTU (y otros protocolos más) a Wifi, sin embargo, como pudieron notar en el desarrollo de esta sección, los datos del sensor llegan primeramente al Gateway (Raspberry) y después se trasmiten los datos al PLC. Para este caso de prueba funciono apropiadamente, más en un proceso en donde la latencia sea muy importante, surgiría un problema este traslado de datos de esta manera.

## 5.3 Conclusiones generales

El desarrollo de este trabajo permitió la puesta en marcha de un PLC LOGO! en una red IoT y así mismo el desarrollo de una placa para la conversión de protocolos cableados a wifi, donde a través de las pruebas con los diferentes, protocolos y dispositivos se permitió saber como se comportaba la red completa, desde la transmisión de datos entre PLC y Gateway, hasta la transmisión de datos con los diferentes dashboard y dispositivos conectados. Sin duda alguna, el protocolo que mejor se desempeño fue en S7, tal vez por el hecho de que es el protocolo de la

empresa fabricante. En lo que respecta a Modbus TCP/IP, a pesar de ser un protocolo antiguo, este no cuenta con el desarrollo pertinente de los nodos en Node-RED aún, al igual que presenta una limitante en este PLC en particular, que es la cantidad de registros que pueden utilizarse para el intercambio de datos.

El dashoard local, creado en Node-RED funciono perfectamente si ninguna latencia grande. Sin embargo, el dashboard remoto, implementado en Ubidots, no cumplió su función completamente, ya que este presentaba desconexiones continuas con el bróker de la misma empresa, creando que el intercambio de datos no se presentara en tiempo real, o inclusive se perdiera.

Este trabajo tiene mucho por recorrer aun, probando con diferentes plataformas, y protocolos, buscando la manera de que los datos estén en tiempo real, tanto de manera local como remota, para en un futuro poder indagar otros temas dentro del concepto de IoT, tales como análisis de datos para la prevención de fallas o inclusive la administración más concreta de todo el sistema.

# BIBLIOGRAFÍA

## Bibliografía


[1] J. L. . M. Peña y G. E. C. Suqillo. (2016). “Estudio del modelo de referencia del Internet de las Cosas (IoT), con la implementación de un prototipo domótico”. Escuela Politecnica Nacional. Volumen I.

[2] Evans, Daves. (2011). Internet de las cosas: Cómo la prócima evolución de internet lo cambia todo. En línea. Disponible en: https://www.cisco.com/c/dam/global/es_mx/solutions/executive/assets/pdf/internet-of-things-iot-ibsg.pdf. Consultada en mayo de 2020.

[3] ITU-T Grupo de estudio 16. (2012). Infraestructura mundial de la información, aspectos del protocolo internet y redes de la próxima generación. UIT-T. Suiza, Ginebra.

[4] S.-L. Peng, L. Huang y S. (2019). Principles of Internet Of Things(IoT) Ecosystem: Insight Paradigm. Springer Nature Switzerland AG. Warsaw, Poland.

[5] L. A. P. Jurado, W. A. V. Velásquez y N. F. E. Vinueza. (2014). Estado del Arte de las Arquitecturas deInternet de las Cosas (IoT). En linea. Disponible en: https://www.academia.edu/7197061/Estado_del_Arte_de_las_Arquitecturas_de_Internet_de_las_Cosas_IoT_. Consultado en abril de 2020.

[6] J. Ladew. (2018). Inductive Automation. En linea. Disponible en: https://inductiveautomation.com/resources/article/what-is-iiot. Consultada en abirl de 2020.

[7] F. D. Petruzella. (2017). Programmable Logic Controllers. Mc Graw Hill Education. New York.

[8] K. Kamel y E. Kamel. (2014). Programmable logic controllers. Mc Graw Hill Education. an Francisco.

[9] A. R. Santillan. (2014). Instituto nacional de tecnoligóas educativas y de formación del profesorado. En linea. Disponible en: http://fresno.pntic.mec.es/jyanez/domotica/introduccionlogosoft.pdf. Consultada en marzo de 2020.

[10] R. P. Foundation. (2019). Raspberry Pi. En linea. Disponible en: https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/. Consultada en mayo de 2020.

[11] N. O'Leary y D. C. Jones. (2018). Node-RED. En linea. Disponible en: https://nodered.org. Consultada en marzo de 2020.

[12] S. Soppin. (2017). OpenSource For U.com. En linea. Disponible en: https://opensourceforu.com/2017/09/node-red/. Cosultada en abril de 2020.

[13] Palacios, Albaro. (2018). Ubidots documentation. En línea. Disponible en: https://ubidots.com/. Consultada en mayo de 2020.

[14] J. Bartolomé. (2011). Protocolo Modbus TCP/IP. En linea. Disponible en: http://www.tolaemon.com/docs/modbus.htm#introduccion. Consultada en abirl de 2020.

[15] M. Yuan. (2018). ¿Por qué el MQTT es bueno para IoT?. En linea. Disponible en: https://www.ibm.com/developerworks/ssa/library/iot-mqtt-why-good-for-iot/index.html. Consultada en marzon de 2020.
[16] T. Wiens. (2016). S7 Comm. En linea. Disponible en: https://wiki.wireshark.org/S7comm. Consultada en mayo de 2020.
[17] I.O, Frank. (2019). Factory I/O documentation. En línea. Disponible en: https://docs.factoryio.com. Consultada en mayo de 2020.
[18] K. I. K. K. N. O. S. W. T. A. T. K. y Y. N. (2020). Sistema de comunicaciones inalambricas industriales: MX 370841 B. Oficina de patentes. Mexico.
[19] M. J. Nixon, A. T. Enver, N. H. Bell, J. B. Kidd y P. R. Moston. (2016). Distributed industrial performance monitoring and analytics: US 20170102678 A1. Oficina de patentes. Estados unidos.
[20] J. L. Daza de Jesus y M. Ubaldo Romero. (2020). “Propuesta de un sistema de monitoreo de energía eléctrica para transformadores de 20 MVA”. Academia Journarls del IPN. http://tesis.ipn.mx/handle/123456789/27971. México, D.F. Mexico.
[21] A. O. Mariscal Organista y S. A. Manjarrez. (2019). “Internet de las cosas aplicado a la manipulación remota de un variador de frecuencia.”. Academia Journarls del IPN. https://tesis.ipn.mx/handle/123456789/27662. México, D.F. México.
[22] J. S. N. y P. V. (2019). “Monitoring System In Industry Using IoT”. International Conference on Advanced Computing & Communication Systems. Volumen I; Aticulo No. 5.
[23] S. Bakshi, G. Khairmode, N. Varkhede y S. Ayane. (2019). “Monitoring and control of plc based automation system”. International Research Journal of Engineering and Technology (IRJET). Volumen VI. Articulo No. 3.
[24] R. Joshi, Jadav, A. Mali y Kulkarni. (2016). “IOT Application for Real-time Monitor of PLC Data”. International Conference on Internet of Things and Applications (IOTA). Volumen V. Articulo No. 1.
[25] F. Tietz, D. Brandao y L. Ferreira Alves. (2018). “Development of an Internet of Things Gateway”. IEEE International Conference on Industry Applications. Volumen XIII.
[26] B. A. a. Drivers. (2003). PLC LOGO! Siemens. En linea. Disponible en: www.siemens.com/logo/. Consultada en abirl de 2020.
[27] S. S. Michael. (2018). Introducción a los controladores logicos progamables. En linea. Disponible en: https://www.allaboutcircuits.com/technical-articles/what-is-a-plc-introduction-to-programmable-logic-controllers/. Consultada en marzo de 2020.

# GLOSARIO DE TÉRMINOS

## Glosario de términos

**API REST:** Es cualquier interfaz entre sistemas que use HTTP para obtener datos o generar operaciones sobre esos datos en todos los formatos posibles, como XML y JSON.
**API:** Interfaz de programa de aplicación (Application Program Interface).
**ASCII:** Código estándar americano para el intercambio de información (American Standard Code for Information Interchange).
**Device ID:** Documento de identidad del dispositivo (Device Identity document).
**Gateway:** Dispositivo físico o un programa de software que sirve como punto de conexión entre la nube y los controladores, sensores y dispositivos inteligentes.
IIoT: Internet de las cosas Industrial (Industrial Internet Of Things).
**RTU:** Unidad terminal remota (Remote Terminal Unit).
**IoT:** Internet de las cosas (Internet of Things)
**IoT:** Internet de las cosas (Internet Of Things).
**IP:** Protocolo de internet (Internet Protocol).
**JSON:** es un formato de texto sencillo para el intercambio de datos (JavaScript Object Notation).
**M2M:** Maquina a máquina (Machine to Machine).
**MQTT:** Transporte de telemetría de cola de mensajes (Message Queue Telemetry Transport).
**PLC:** Controlador Lógico Programable (Progamming Logic Controller).
**RAM:** Memoria de acceso aleatorio (Random Access Memory).
**ROM:** Memoria de solo lectura (Read Only Memory).
**SoC:** Sistema en un chip (System on a Chip).
**Trama:** Es una serie sucesiva de bits, organizados en forma cíclica, que transportan información.

# ANEXO 1
## Dictamen

## Anexo 1. Dictamen

TunaShields

**Asunto:**
**Informe Técnico**

**A quien corresponda:**

Por medio de la presente se hace constar que el **MIP. José de Jesús García Cortés** participó como ASESOR por parte del Instituto Tecnológico de Cd. Guzmán en el proyecto denominado: **"Implementación de un PLC Siemens LOGO en una plataforma de aplicaciones IoT"** en el periodo comprendido del 20 de enero al 31 de mayo de 2020, en dicho proyecto participó el estudiante del programa de Ingeniería Electrónica del Instituto Tecnológico de Cd. Guzmán: **C. Carlos Angel Patricio Martínez** con N. de control **15290370**.

Con el desarrollo del mencionado proyecto, la empresa **Tuna Shields S. de R.L. de C.V** logró los siguientes beneficios:

1. Abrir una línea de investigación y desarrollo relacionada a la automatización industrial y el desarrollo del concepto industria 4.0
2. Desarrollo de un prototipo de tarjeta electrónica para integración de sensores modbus.
3. Se desarrollaron vías de integración de protocolos iot con un PLC.

Se extiende la presente para los fines legales que al interesado convengan, en la Ciudad de Colima, Colima a los 31 días del mes de mayo de 2020.

Alejandro Elizalde Torres
Director General

TunaShields

www.tunashields.com
Mauna Loa 1214 | Nvo. Milenio
Colima, Col. México | C.P. 28048

# ANEXO 2
## ¿Cómo referenciar bibliografía?

## Anexo 2. ¿Cómo referencia la bibliografía consultada?

a) **Libros;** Autor. (Año). Título subrayado. Editorial. Lugar de impresión. Número de las páginas consultadas. Ejemplo:

[01] Ruelas-Lepe, Rubén y Bernal-Casillas, José de Jesús. (2010). Desarrollo y publicación de la tesis. Amate editorial. Zapopan, Jalisco, México. Páginas 98-112.

b) **Artículos;** Autor. (Año). Título. Nombre del artículo entre comillas. Subrayar el evento en el cual se presentó el artículo. Tomo, número o cualquier otra indicación que contenga la portada de la revista, tal y como se indica en la mismo.

[02] García-Cortés, José de Jesús y otros. (2017). "Modelado y control de la variable temperatura de un invernadero por medio de un controlador PID clásico y un controlador PID difuso de 9 reglas". Academia Journals Cd. Juárez 2016. Volumen II; Artículo No. J179; ISSN 1946-5351. Cd. Juárez,

c) **Internet;** Autor. (año). Título subrayado. Dirección. Fecha de consulta. Ejemplo:

[03] Juma, N. G. (1999). The Pedosphere and its Dynamics: Ecological Functions of Soil, 2.4 Recycles Wastes. En línea. Disponible en: www.pedosphere.com. Consultada en diciembre de 1999.

d) **Conferencias o clases;** Autor. (Año). Indicar si es conferencia o clase. "tema". Materia especifica subrayada. Lugar, fecha. Ejemplo:

[04] Molina-Gaudo, Pilar. (2005). Conferencia. "La mujer en la ingeniería". Educación superior. Universidad de Zaragoza; Zaragoza, España. 14 de abril.

e) **Experiencia;** Autor. Experiencia. Lugar donde se adquirió la experiencia. Materia impartida o proyecto realizado. Periodo. Ejemplo:

[05] Pérez-Quiroz, Raúl. Experiencia. Instituto Tecnológico de San Luis Potosí. Asignatura de programación de enero a junio de 1986

f) **Patentes;** Autor. (Año). Patente. Nombre subrayado. Número de patente. Oficina d patentes. Lugar. Ejemplo:

[06] Pérez-Ulloa, Juan. (1986). Patente. Pomada para eliminar la comezón: 4567456. Oficina de Patentes. México, D.F., México.

# ANEXO 3
## Node-RED

# Anexo 3. Node-RED en Raspberry Pi 3

## Instalación de Node-RED en Raspberry Pi 3

Para poder correr Node-RED en una Raspberry se Requiere de diferentes paqueterías, como Node.js, npm y Node-RED. Sin embargo, la página oficial provee de un script qué ayuda la instalación de estos mismos.

Corriendo el siguiente comando, se descargará y correrá el script:

```
bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered)
```

Nota: El siguiente script solamente correrá en sistemas operativos basados en Debian, así como Ubuntu y Diet-Pi. Para verificar qué los paquetes se pueden instalar corre el siguiente comando:

```
 sudo apt install build-essential git
```

## Corriendo Node-RED en Raspberry Pi 3

Ya que se han instalado las paqueterías, para poder correr Node-RED tienes que abrir la terminal y escribir

```
node-red
```

Puede ser parado presionando las teclas Ctrl  + C.

## Corriendo Node-RED como un servicio

Node-RED También se puede correr como un servicio. esto significa qué puede estar funcionando en segundo plano y también inicia automáticamente al conectar la raspberry.

Los siguientes comandos son los utilizados para trabajar al Node-RED como un servicio:

```
node-red-start -> Inicia Node-Red como un servicio y muestra en la terminal la salida de registro. Precionar Ctrl + C no para el servicio, lo sigue corriendo en segundo plano.

node-red-stop -> Para el Node-RED como servicio.
```

```
node-red-restart -> Para y reinicia Node-RED como servicio.

node-red-log -> Muestra en terminal la salida de registro.
```

**Autoarranque al encender**

Si tú quieres qué Node-Red inicie cuando la raspberry Pi arranque o cuando sea reiniciada, necesitas activar el servicio de autoarranque corriendo el siguiente comando:

```
sudo systemctl enable nodered.service
```

para desactivar el servicio de autoarranque, corre el siguiente comando:

```
sudo systemctl disable nodered.service
```

**Abriendo el editor**

Una vez no red este corriendo tú puedes acceder al editor en un navegador.

Si tú estás usando el navegador en el escritorio de la raspberry Pi, puedes abrir el editor de la siguiente manera:

http://localhost:1880.

Nota: es recomendable usar un navegador fuera de la raspberry Pi.

Cuando estás usando el navegador fuera de la raspberry Pi, tienes que usar la dirección IP de tu raspberry Pi, y escribir lo siguiente:

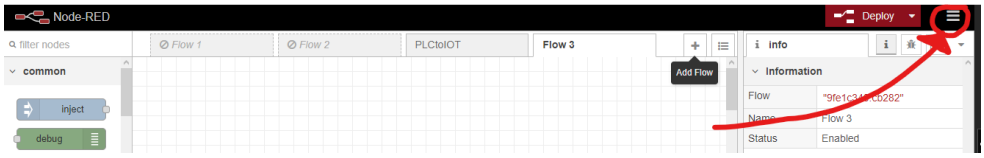
`http://<hostname>:1880 -> Rempllaza <hostname> por la ip`

Si no sabes la ip, teclea esto en la terminal:

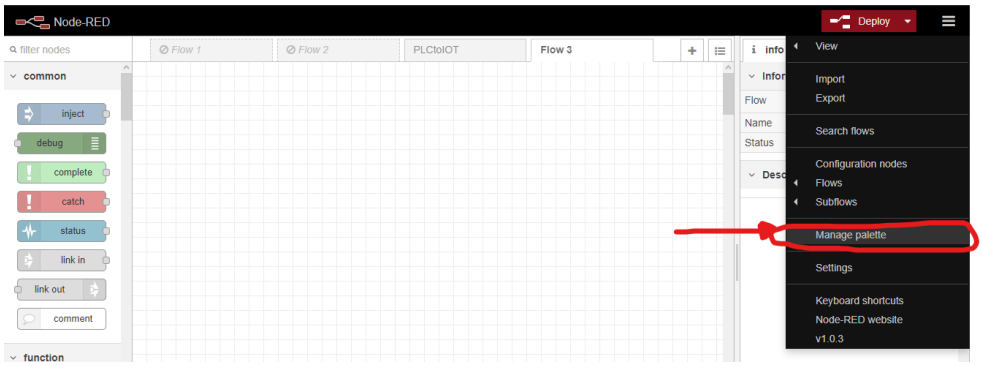
`hostname -I`

## Instalación de librerías en Node-RED

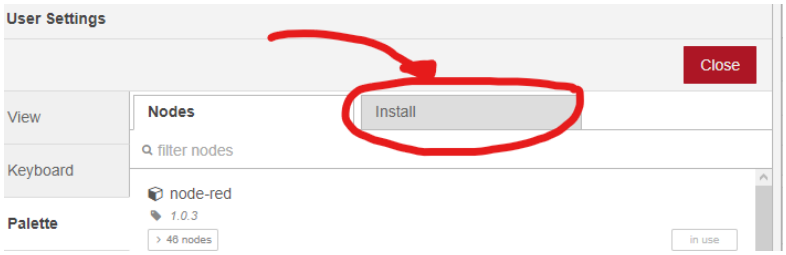
1. Dentro del entorno del editor, da clic donde indica la siguiente imagen:




2. Después de dar clic, aparecerá un menú como el mostrado en la siguiente imagen, en el cual irás a “Manage palette”.




3. Dentro de la nueva ventana emergente, ir a la pestaña “Install”.

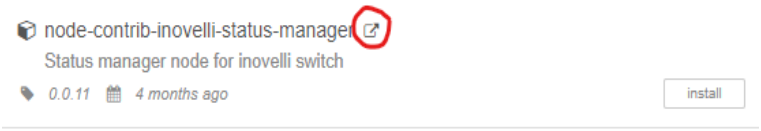



4. Automáticamente el cursor, estará en un buscador, en el cual harás la búsqueda del paquete requerido.

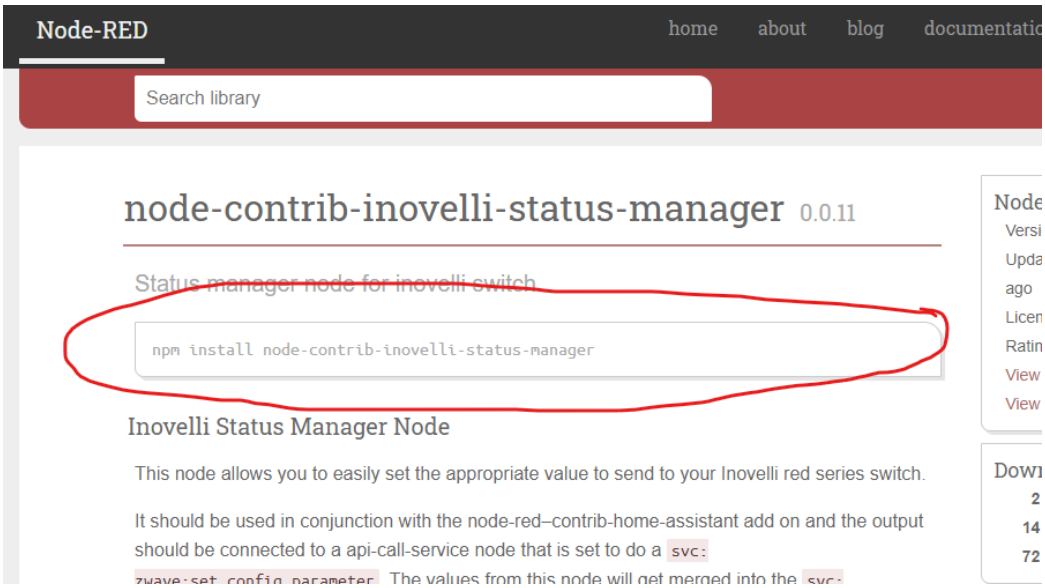
5. Al encontrar el paquete a instalar, tienes dos opciones para realizar la instalación. La primera opción es simplemente dar clic en el botón “Install” que se encuentra en la parte inferior derecha de cada paquete. Realizando esta primer opción simplemente es esperar a que se instale. Cuando esto suceda en lugar de la frase “Install”, será “Installed”.

6. Existe una segunda opción, debido a que en ocasiones hay problemas realizando la instalación de ciertos paquetes desde el editor.

Esta se trata de escribir una línea de código en la terminal, para lo cual tienes que ir al sitio web de la librería. Para poder encontrar esta línea de código, da clic en el icono que se encuentra circulado con rojo, como se muestra en la imagen:



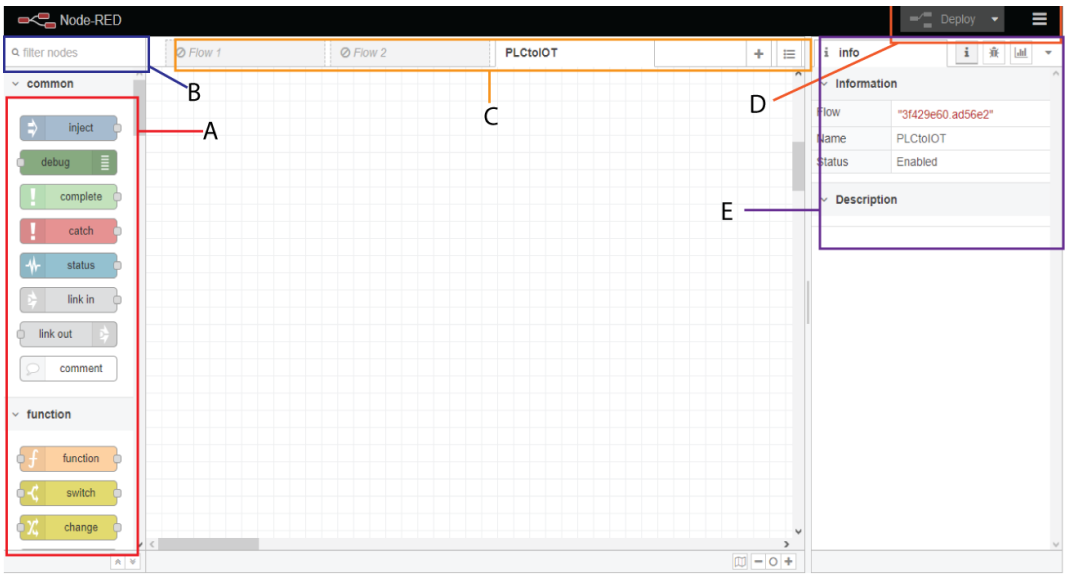

7. Después de haber dado clic, te abrirá una nueva pestaña en tu navegador, en el cual podrás ver al iniciar la página, la línea de código que nos interesa a nosotros:




8. Ya identificada la línea de código, procedemos a escribirla en la terminal de nuestra raspberry.

**Node-RED y su entorno**

En la siguiente imagen se encuentra el ambiente general que vas a encontrar en Node-RED. De manera general te explicaré qué es lo que cada parte hace:




A) En esta área Encontrarás todos los nodos Organizados de acuerdo con el nombre de su paquetería. Para buscar a través de este espacio es necesario girar el scroll de tu mouse.
Para poder utilizar los notos simplemente seleccionas y arrastras al área de trabajo.

B) En este apartado podrás hacer la búsqueda del nodo que tú requieras, simplemente escribiendo el nombre de este, o una palabra relacionada.

C) En este lugar encontraras las diferentes pestañas del proyecto, en cada uno puedes tener diferentes códigos, de igual manera puedes ocultar las pestañas necesarias, para poder cargar sólo el que requieres.

- Para crear una nueva pestaña simplemente se da clic en el signo de + que se encuentra en la parte derecha.
- Para eliminar la pestaña, das doble clic sobre la pestaña, lo cual te arrojará una ventana emergente, estando ahí, en la parte superior izquierda de esa ventana emergente encontraras un botón que diga “Delete”, presiónalo y listo.
- Para ocultar pestaña, es igual que para eliminar, sin embargo, el botón se encuentra en la parte inferior izquierda de la ventana emergente, como “Enabled”, simplemente presiónalo y listo. Para poderla utilizar, realizas los mismos pasos.

D) Dentro de esta sección encontrarás dos botones diferentes.
En el primero, será el que te ayude a cargar tu programa a la raspberry, así mismo, para poder reiniciar todo el programa o volver a cargar.
En el segundo botón, está todo lo relacionado a configuración de Node-RED (Configuration), así como la exportación e importación de trabajos (Import &

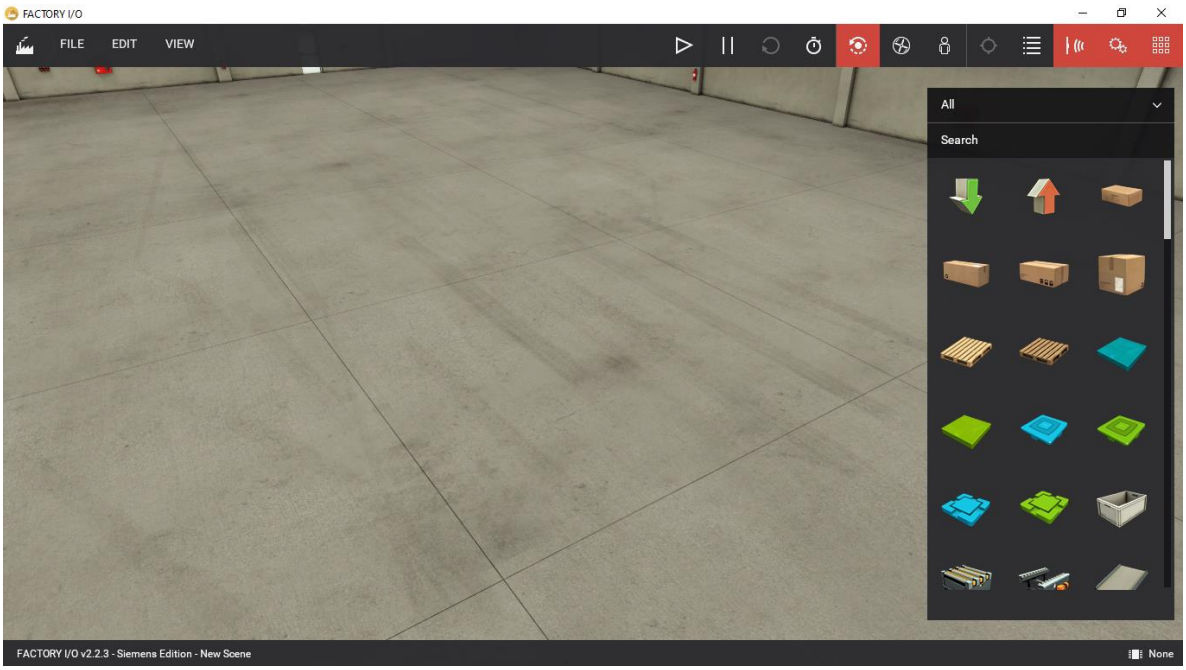
Export). Punto importante para poder acceder a otros trabajos encontrados en la web, o bien para compartir tu proyecto con alguien más.
Como ya se había mencionado antes en esta sección se encuentra el apartado para poder cargar nuevas paqueterías (Manage Palette).
¿Problemas con datos fantasmas? Hay ocasiones en las que al agregar ciertos nodos, se crean variables, y después de eliminar el nodo, estas variables no se eliminan, la mejor solución es ir a la configuración de nodos (configuration nodes). Ahí tienes que buscar el nombre de la paquetería y podrás eliminar las variables creadas automáticamente.

E) En esta última Sección encontrarás 3 ventanas diferentes.
En la sección de información, representada con una i, encontrarás la descripción de cada nuevo seleccionado, así que si tienes dudas sobre qué hace ese nodo, solo selecciónalo Y vea esa ventana.
En la pestaña dos encontraras la parte de debug, aquí es donde podrás ver todos los datos mandados al nodo debug.
Por último se encuentra la ventana de Dashboard, aquí es donde podrás cargar las propiedades del dashboard que estés utilizando, así como hacer ediciones a las mismas.

# ANEXO 4
## Factory IO

# Anexo 4. Factory I/O

## Entorno de trabajo en Factory IO




**Menú de bienvenida:** Puedes acceder rápidamente a el entorno principal en donde encontraras documentación, tutoriales, escenas, el chequeador de actualizaciones, etc.

**Correr o Editar:** Con este botón podrás intercambiar entre dos modos, el correr tu escena, o pararla para editar.

**Pausa:** Pausa la simulación.

**Resetea:** Resetea la simulación y los parámetros configurados dentro de la escena.

**Escala de tiempo:** En este apartado podrás modificar el la velocidad en que la simulación se correrá.

**Cámara orbitaste:** Con esta cámara podrás moverte por toda la escena si tener un punto fijo de visión.

 **Cámara voladora:** Con esta cámara podrás girar a 360 grados desde un solo punto a una altura cualquiera.

 **Cámara en primera persona:** Podrás tener la visión a la altura de una persona, pero al igual podrás ver a 360 grados.

 **Seguir objeto:** Con esto puedes seleccionar algo para estar siguiendo.

 **Ventana de vistas:** Aquí te desplegara una ventana en la cual tendrás la opción de guardar posiciones específicos de ciertas vistas con las cámaras, así podrás acceder fácilmente a ese punto de visualización previamente visto con mayor facilidad.

 **Etiquetas de sensores:** Muestra u oculta las etiquetas de los sensores que se encuentren en tu escena.

 **Etiquetas de actuadores:** Muestra u oculta las etiquetas de los actuadores que se encuentren en tu escena.

 **Paleta de librerías de dispositivos:** Muestra u oculta la paleta de las librerías. Dentro de esta ventana podrás encontrar los diferentes, actuadores, sensores y demás dispositivos para poder construir tu propia escena. Para poder utilizar lo que se encuentra dentro de esta ventana, basta con solo seleccionar y arrastrar al ambiente de trabajo.

**Icono de estado:** En este espacio encontraras el controlador seleccionado, así como el estado de este (Conectado o desconectado).

**Creación de escena personalizada**

1. Abre Factory IO, y dentro de la ventana busca la opción que diga nuevo y presiona.

2. Automáticamente te abrirá un entorno de un almacén grande, es aquí donde colocaras los diferentes componentes de tu proceso, desde actuadores hasta sensores y demás.

3. Para poder colocar objetos simplemente activa la ventana de librerías, que se encuentra en la parte superior derecha. Es el primer icono.

Cuando ya se haya abierto esto, busca lo necesario.
Para poder colocarlo, simplemente seleccionas y arrastras hasta la posición que tu desees.

4. Cuando ya hayas colocado todos los objetos necesarios, necesitas ir a File>Drivers. Ahí podrás seleccionar cómo quieres controlar todo tu sistema.

5. Al momento de seleccionar el driver a utilizar, te aparecerá una imagen referente al dispositivo que seleccionaste, en el cual se mostraran los puertos de salida y entrada.

6. Ahora que ya se ha seleccionado el driver, tienes que ir a configuración que se encuentra en la parte superior derecha, y colocar los parámetros necesarios para establecer una comunicación entre la escena y el controlador seleccionado.

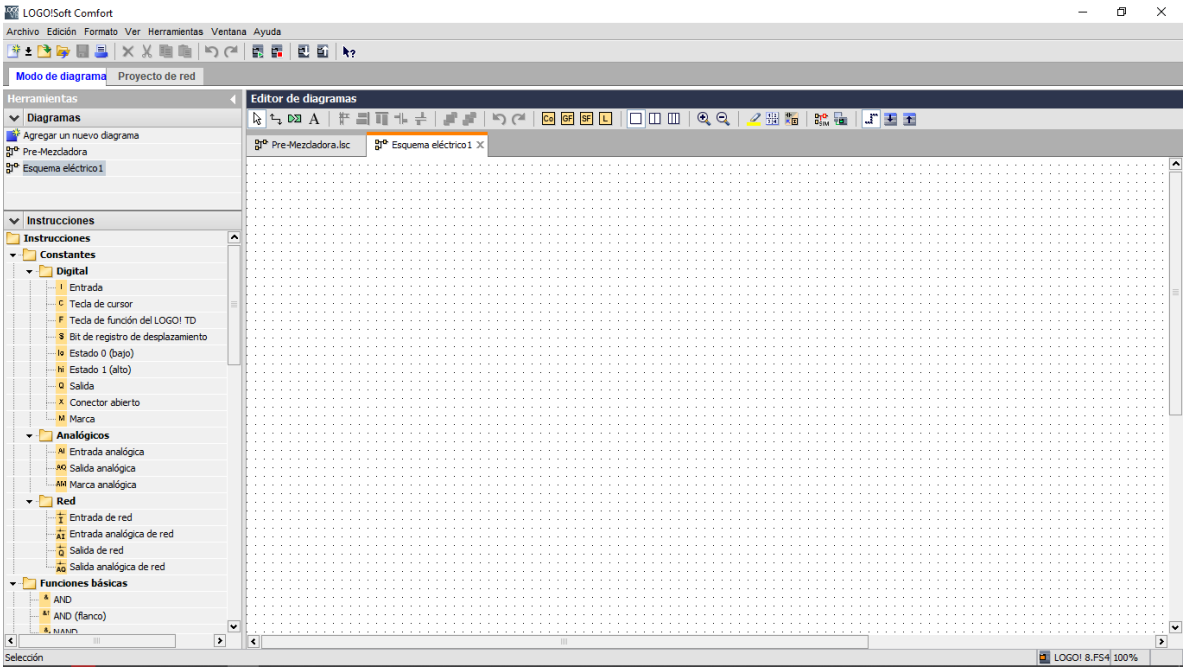
7. En esa misma sección encontraras otro botón que dice conectar, el cual servirá para saber si realmente la configuración realizada previamente funciona o no.

8. En el lado izquierdo encontraras todas las variables de sensores y por la parte derecha las de actuadores, a través de arrastrar y colorar las variables en la imagen del controlador que apareció al seleccionar, se hará la conexión con el controlador para el intercambio de datos o bien el control de dispositivos.

# ANEXO 5
## LOGO! Soft Comfort

# Anexo 5. LOGO! Soft Comfort

## Entorno del software LOGO! Soft Comfort




 Crea un espacio de trabajo nuevo, para realizar diagrama tipo FUP.

 Crea espacio de trabajo de los 3 diferentes diagramas, FUP, KOP, UDF.

 Abre archivo.

 Cierra espacio de trabajo.

 Borrar, Cortar, Copiar, Pegar. En ese Orden.

 Retroceder cambios, reestablecer cambios.

 Conectar Software y PLC.

 Desconectar comunicación entre software y PLC.

 Cargar programa a PLC.

 Carga programa del PLC al software.

 Selección de bloques.

 Conectar entre bloques.

 Deshacer conexión, y crear puntos aislado de conexión.

 Insertar texto.

 Ordenar grupo de bloques.

 Traer adelante – Enviar al fondo.

 Activar ventana de funciones comunes: Constantes/Conectores, Funciones básicas, Funciones especiales, perfil de registros de datos.

 Dividir espacio de trabajo para trabajar con diferentes diagramas a la vez.

 Zoom.

 Selecciona una línea en tu diagrama y la colorea desde inicio hasta fin.

 Aumenta el tamaño de tu espacio de trabajo.

 Convertir a diagrama KOP.

 Iniciar imulación.

 Iniciar prueba online.

 Mostrar líneas de referencia.

 Mostrar/Ocultar referencias de los bloques.

**Diagramas:** Encontrarás todos los diagramas creados y abiertos. De igual manera, podrás crear un nuevo diagrama.

**Instrucciones:** Aquí estarán todas los bloques de funciones organizadas por tipo de función, propiedades, y demás.
Para poder hacer uso de los diferentes bloques simplemente seleccionas el bloque que necesitas, y ve a tu espacio de trabajo a colocarlo.

**Realizar comunicación entre software LOGO! Soft Comforty el PLC**

1. Presiona el siguiente icono  . Al realizar la acción, se abrirá un ventana emergente, en la cual se realizará la configuración.
2. Dentro de la ventana emergente, encontrarás lo que se ve en la imagen siguiente. Aquí podrás hacer dos cosas, insertar la IP del PLC, o bien buscarlo directamente, simplemente presionado en el siguiente icono  .
   Al presionarlo el software buscará dentro de la red local, la PLC, y automáticamente aparecerá en la ventana.

3. Cuando ya se haya realizado la acción anterior, ya sea escribir directamente la dirección IP o bien buscarla. Se presionará el botón “Probar”, con este botón se probará que la comunicación haya sido exitosa, de ser así, se pondrá verde la Lilea sobre el botón, y una palomita verde.
4. Después de haber probado simplemente se da aceptar, y todo listo. Ahora ya es posible cargar o extraer el programa en el PLC.

**Crear un proyecto de red en PLC LOGO!**

Dentro del apartado de comunicaciones, el PLC LOGO! tiene la posibilidad de realizar transmisión de daros por protocolos como ModBus TCP/IP o bien su protocolo de la marca, el S7.

Como el S7 es parte importante de la empresa, este está muy bien desarrollado, así que es mucho mayor la facilidad de crear una transmisión de datos. En seguida se plasmarán brevemente los pasos para realizar transmisión de datos con este protocolo:

1. Dentro del modo diagrama, se crea un nuevo diagrama:
2. En la ventana “Diagramas”, podrás encontrar el diagrama que acabas de realizar, ve ahí y da clic derecho sobre el diagrama.
3. Dentro de la ventana que apareció, busca la opción “Conexiones Ethernet”, y da clic.
4. Aparecerá una ventana pequeña en la cual colocaras la dirección IP de LOGO, y en la mascará de subred, coloca lo siguiente: 255.255.255.0
5. Con esto ya estás habilitado el uso de bloques de comunicación, ya sólo faltaría dar de alta al PLC en el dispositivo con el que quieres intercambiar datos.
6. Los bloques que utilizarás serán los que se encuentran en la carpeta de Red, dentro de la ventana “Instrucciones”.
   Simplemente coloca las que vayan con el dato a trasmitir, ya sea booleano o uno número más grande.
7. Ya colocados los bloques tienes que seleccionar el espacio de memoria que estarán usando, simplemente da doble clic sobre el bloque y selecciona.