ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DESARROLLO DE APLICACIONES DE SUPERVISIÓN INDUSTRIAL DE CÓDIGO ABIERTO PARA EL LABORATORIO DE REDES INDUSTRIALES

DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE GATEWAY IOT INDUSTRIAL PARA ADQUISICIÓN DE DATOS Y SU MONITOREO DESDE UN NAVEGADOR WEB

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y AUTOMATIZACIÓN

DIEGO GABRIEL MANCHENO BARBA

diego.mancheno@epn.edu.ec

DIRECTOR: DRA.- ING. SILVANA DEL PILAR GAMBOA BENÍTEZ

silvana.gamboa@epn.edu.ec

DMQ, enero 2022

CERTIFICACIONES

Yo, DIEGO GABRIEL MANCHENO BARBA declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

Sr. Diego Gabriel Mancheno Barba

Certifico que el presente trabajo de integración curricular fue desarrollado por DIEGO GABRIEL MANCHENO BARBA, bajo mi supervisión.

Dra.- Ing. Silvana del Pilar Gamboa Benítez
DIRECTOR

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

Sr. Diego Gabriel Mancheno Barba

Dra.- Ing. Silvana del Pilar Gamboa Benítez

DEDICATORIA

A mi querida familia: Mayra, Mario y Javier por todo su cariño, apoyo y motivación en este largo proceso.

AGRADECIMIENTO

A mis padres: Mayra y Mario, que me han apoyado incondicionalmente, por siempre motivarme y guiarme mediante su ejemplo.

A mi hermano: Javier, que siempre ha estado a mi lado respaldándome y trayéndome alegrías.

A mis abuelitos, que me han enseñado que con esfuerzo y trabajo se puede alcanzar cualquier meta que uno se proponga.

A mis tíos y primos, por todo su cariño y energía positiva.

A mis amigos, por siempre estar conmigo en las buenas y en las malas.

A mis panas del prepo, por todas sus locuras y su gran amistad que ha perdurado a pesar del tiempo y la distancia.

A mis estimados compañeros de la carrera por su esfuerzo en todas nuestras madrugadas de estudios y proyectos.

A los profesores de la Escuela Politécnica Nacional por impartirme todos sus conocimientos.

A la Doctora Silvana Gamboa por su apoyo en el desarrollo de este trabajo.

ÍNDICE DE CONTENIDO

RESUMEN

Actualmente, múltiples industrias han empezado a utilizar distintos dispositivos que posibilitan la integración de datos de un proceso industrial a través de internet para incorporar los procesos de fabricación en un esquema de Industria 4.0. Bajo este contexto, el presente trabajo involucra el desarrollo de un gateway IoT industrial para adquirir datos de dispositivos industriales mediante entradas y salidas estándar digitales o analógicas. El prototipo proporciona interacción bidireccional con una estación remota a través del protocolo de comunicaciones seriales Modbus RTU sobre interfaz RS-485. La implementación se basa en una tarjeta Arduino UNO que incorpora un módulo ethernet. En esta tarjeta se implementan las funciones necesarias para establecer la comunicación y permitir la gestión de datos en capas superiores bajo el protocolo MQTT. También se desarrolló una interfaz de monitoreo complementaria utilizando Node-RED. Se integraron varias características dentro de esta interfaz que incluye una visualización organizada de datos, generación de gráficos en tiempo real y alarmas que pueden ser configuradas y visualizadas en función de los valores y estados de algunas variables obtenidas a través del gateway desarrollado. El prototipo y la interfaz fueron sujetos a pruebas para validar su funcionamiento.

PALABRAS CLAVE: gateway, industrial, IoT, MQTT, Node-RED

ABSTRACT

Nowadays, many industries have started to use different devices that allow the integration of process data through the internet to incorporate manufacturing into an Industry 4.0 scheme. In this regard, the present work involves the development of an IIoT Gateway to acquire data from industrial devices and through standard digital or analog inputs and outputs. The prototype provides bi-directional interaction with a remote station through Modbus RTU serial communications protocol under RS-485 interface. The implementation is based on an Arduino UNO board that incorporates an Ethernet module. The necessary functions are implemented in this board to establish communication and allow data management in higher layers under the MQTT protocol. A complementary monitoring interface was also developed using Node-RED. Several features were integrated within this interface, which includes an organized data displaying, real-time charts generation, and alarms that can be set and visualized based on the values of certain variables obtained through the developed gateway. The prototype and interface were tested to validate their operation and performance. The gateway is based on low-cost technologies and allows the integration of multiple protocols and industrial devices for possible future projects.

KEYWORDS: gateway, industrial, IoT, MQTT, Node-RED

1. INTRODUCCIÓN

La cuarta revolución industrial impulsa la idea de la integración de todas las tecnologías disponibles hasta la fecha, equipadas con sensores y unidades de control capaces de comunicarse e intercambiar información de manera automática con máquinas, plantas e incluso productos a lo largo de la cadena de producción [1]. La Industria 4.0 se basa en un conjunto de tecnologías habilitadoras para su desarrollo, una de estas tecnologías lo constituye el internet de las cosas que es el uso de sistemas electrónicos en red para adquirir datos de sensores y actuadores embebidos en máquinas y otros objetos físicos [1].

En el centro del internet de las cosas se encuentra el gateway que es un dispositivo encargado de proporcionar conectividad entre las cosas que corresponden a objetos encargados de recolectar información y realizar cambios en el entorno, y la nube que corresponde a una red de equipos informáticos que posibilitan el uso de los datos y el comando de las cosas [2]. El objetivo del gateway dentro de un proceso industrial corresponde a posibilitar el flujo bidireccional de información entre la nube que corresponderían a los operadores locales y remotos y las cosas que corresponderían a los dispositivos en campo, aplicaciones e incluso a las personas involucradas directamente con el proceso industrial generando beneficios como el incremento de la eficiencia y productividad, la creación de nuevas oportunidades de negocios, el incremento de la seguridad de los trabajadores, el mejoramiento del proceso de innovación de productos, reducir costos de los recursos y mejorar el entendimiento de las demandas de los consumidores [3]. Todos estos beneficios pueden resultar bastante convenientes para una empresa, sin embargo, la adquisición de esta tecnología constituye una inversión económica que no todas las industrias pueden afrontar como es el caso de las PYMES, razón por la cual es importante la disponibilidad de soluciones desarrolladas con tecnología de bajo costo que posibiliten una opción alternativa para el acceso a dichos beneficios.

Bajo este contexto, el trabajo propuesto consiste en el desarrollo de un gateway que pueda ser utilizado en el laboratorio de redes industriales como un prototipo de pruebas para la supervisión y control de un proceso industrial, de tal manera que sirva como modelo para futuros trabajos que puedan generar beneficios para las pequeñas y medianas industrias. Se propone diseñar e implementar un prototipo de gateway industrial con soporte para internet de las cosas basado en hardware de bajo costo y software de código abierto que pueda adquirir información de variables y dispositivos industriales de campo, procesar la información y gestionar su comunicación en una red para su posterior uso en una aplicación de monitoreo y control, que será desarrollada mediante el uso de herramientas de software de libre acceso y que además pueda ser accedida de manera local o remota.

1.1 Objetivos

El objetivo general del presente trabajo es:

Diseñar e Implementar un prototipo de Gateway IoT Industrial para adquisición de datos de campo y desarrollar una interfaz para el monitoreo y control accesible mediante un navegador web.

Los objetivos específicos del presente trabajo son:

- Realizar una investigación bibliográfica sobre el funcionamiento y las características de gateways industriales con soporte IoT y su uso en el monitoreo y control de procesos industriales, así como los componentes de hardware y software necesarios para su implementación.
- Establecer los requerimientos para el diseño e implementación del prototipo de gateway industrial IoT y los requerimientos para el desarrollo de la interfaz de monitoreo y control.
- Seleccionar los protocolos de comunicaciones y tecnologías de hardware y software mediante las cuales se desarrollará el prototipo de gateway IoT industrial y la interfaz de monitoreo y control.
- Diseñar e implementar el prototipo de gateway industrial IoT y la interfaz de monitoreo y control.
- Validar el funcionamiento del prototipo y la interfaz implementada mediante la emulación de un proceso industrial.

1.2 Alcance

Se realizará una síntesis bibliográfica acerca de la aplicación del internet de las cosas en la industria y sus componentes.

Se realizará una síntesis bibliográfica acerca de gateways industriales comerciales y prototipos desarrollados con soporte para internet de las cosas, haciendo un énfasis en sus características y funcionalidades como la adquisición de datos de variables de campo, comunicaciones con dispositivos industriales y las aplicaciones comunes para las que son utilizados con enfoque hacia las plataformas de monitoreo y control, así como los protocolos de comunicaciones que podrían ser utilizados y las tecnologías de hardware y software que podría considerarse para la implementación.

Se establecerá el conjunto de requerimientos que debe incorporar el prototipo de gateway y la interfaz de monitoreo y control basándose en los equipos investigados y las aplicaciones comunes de las plataformas de monitoreo disponibles.

Se seleccionará los protocolos de comunicaciones que serán necesarios para que el prototipo pueda establecer las comunicaciones en una red informática y comunicarse con un dispositivo de campo.

Se seleccionará la tarjeta embebida y los dispositivos adicionales necesarios para que el prototipo sea capaz de establecer comunicaciones con una red informática y para posibilitar la adquisición datos de campo.

Se seleccionará las distintas herramientas de software de libre acceso que permitan: implementar un servicio que gestione las comunicaciones en la red, desarrollar la interfaz de monitoreo y control incluyendo el desarrollo de aplicaciones del internet de las cosas, e implementar una red privada virtual para realizar pruebas de acceso remoto a la interfaz de monitoreo y control.

Se realizará el diseño e implementación del prototipo en su etapa de hardware, incluyendo la selección de la circuitería necesaria para acondicionar las señales de entrada/salida, así como los módulos electrónicos necesarios para implementar las comunicaciones con el dispositivo de campo, además de módulos y sistemas electrónicos que permitan adquirir información de otros dispositivos electrónicos.

Se desarrollará la programación necesaria para adquirir datos de campo, procesar la información y gestionar su comunicación en la red para posibilitar un intercambio bidireccional entre la interface de monitoreo y el sistema implementado.

Se diseñará e implementará una interfaz de monitoreo y control que permita al usuario interactuar de manera bidireccional con todas las variables de campo de entrada y salida que serán consideradas en los requerimientos con el fin de simular una aplicación industrial del prototipo además de incorporar algunas funcionalidades de las interfaces de monitoreo.

Se montará el servicio requerido que pueda gestionar las comunicaciones del prototipo en la red, garantizando que los datos puedan ser utilizados por el usuario mediante la interfaz de monitoreo y control, además de un servicio de red privada virtual que posibilite el acceso remoto a la interfaz que será desarrollada.

Se realizarán pruebas mediante la emulación de un proceso industrial que permitan verificar el correcto funcionamiento del prototipo al adquirir información de datos de campo y su adecuada visualización en la interfaz de monitoreo, la cual puede ser accedida de manera remota mediante el uso del servicio de red privada virtual, también se verificará el funcionamiento de las distintas aplicaciones adicionales del internet de las cosas que puedan ser implementadas.

1.3 Marco Teórico

1.3.1 Internet de las Cosas en la Industria 4.0

A lo largo de la historia el desarrollo tecnológico ha permitido marcar revoluciones en la industria (Figura 1.1.) que han transformado la economía y han generado nuevas eras de crecimiento y competitividad, dichas tecnologías se conocen como tecnologías habilitantes.

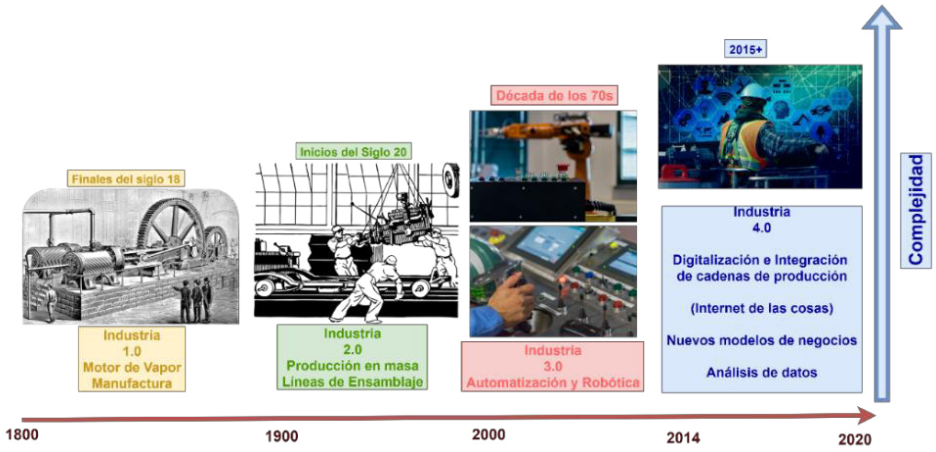


Figura 1.1. Evolución de la Industria. [2]

En la emergente cuarta revolución industrial se promueve la implementación de sistemas ciber-físicos (CPS) que involucran ingeniería computacional, y sistemas de comunicaciones como interfaces al mundo físico impulsando la idea de la integración de toda la cadena de suministro mediante la conexión de todos los dispositivos, aplicaciones y operadores de los diferentes niveles de la pirámide de automatización [1] [2].

La Industria 4.0 brinda varias posibilidades y beneficios como, por ejemplo [4]: Producción flexible, Fábrica convertible, Soluciones Orientadas al consumidor, Logística optimizada, Uso y análisis de datos, Economía Circular y uso eficiente de recursos.

Internet de las cosas

Entre las tecnologías habilitantes de la cuarta revolución industrial destaca el internet de las cosas que suele ser definido como una red de dispositivos y sistemas interconectados (basados en tecnologías de hardware y software) que adquieren información de sensores y actuadores embebidos en máquinas y objetos físicos que intercambian dicha información posibilitando la conexión del mundo físico de sensores, dispositivos y máquinas al internet.

El internet industrial de las cosas corresponde a una extensión y uso del internet de las cosas en aplicaciones del sector industrial con enfoque hacia las comunicaciones de máquina-a-máquina (M2M), Big data y machine learning, permitiendo a las industrias incrementar su eficiencia y confiabilidad [2].

Una de las aplicaciones del internet de las cosas lo constituye la recolección de datos de distintos dispositivos y procesos y su acceso desde cualquier lugar (por medio del internet) mediante un dashboard posibilitando un monitoreo y control de las variables de campo presentes en líneas de producción, esta aplicación es considerada para el presente trabajo.

Bloques Constitutivos del internet de las cosas [2]:

- **Cosas:** Son objetos equipados con sensores/actuadores que recolectan información que será trasmitida a la red.
- **Gateways:** Proporciona conectividad entre las cosas y la nube, se encarga del preprocesamiento y filtrado de datos previamente a su envío hacia la nube y el envío de comandos de control de la nube hacia las cosas. Garantiza la compatibilidad de varios protocolos de comunicaciones de campo.
- **Lago de datos:** Permite almacenar los datos de todos los dispositivos conectados.
- **Análisis de datos:** Corresponde al uso de los datos extraídos del lago de datos para la búsqueda de tendencias que permitan identificar desempeños, ineficiencias y maneras de mejorar el sistema. También suelen ser utilizados para crear algoritmos y modelos para aplicaciones de control.

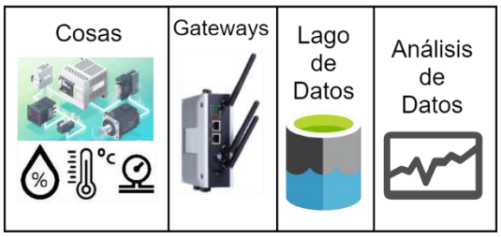


Figura 1.2. Bloques constitutivos del internet de las cosas

Arquitectura del internet industrial de las cosas

El consorcio del internet industrial (IIC) esquematiza la arquitectura del internet de las cosas en tres niveles (Figura.1.3.) Cada nivel se especifica a continuación [2] [5]:

- **Nivel de borde (Edge tier):** Recolecta datos de los nodos de borde (sensores, actuadores, dispositivos, sistemas de control y demás objetos) encargados de adquirir información de variables físicas o actuar sobre procesos industriales mediante protocolos de comunicaciones cableados o inalámbricos usados para transmitir los datos a un gateway que cumple la función de puente hacia otras redes.
- **Nivel de plataforma (Platform tier):** También se lo conoce como nivel de gateway y actúa como nivel intermediario facilitando las comunicaciones de comandos de control y de datos al consolidar, procesar y analizar los flujos de datos de los otros niveles, en ocasiones permite realizar consultas de datos y análisis.

- **Nivel de empresa (Enterprise Tier):** Corresponden a sistemas computacionales que realizan complejas tareas de procesamiento con el fin de realizar análisis de datos, proporcionar sistemas de soporte de toma de decisión, interfaces de monitoreo y control para especialistas, entre otras aplicaciones específicas. También se encarga de emitir comandos de control a los dos niveles inferiores.

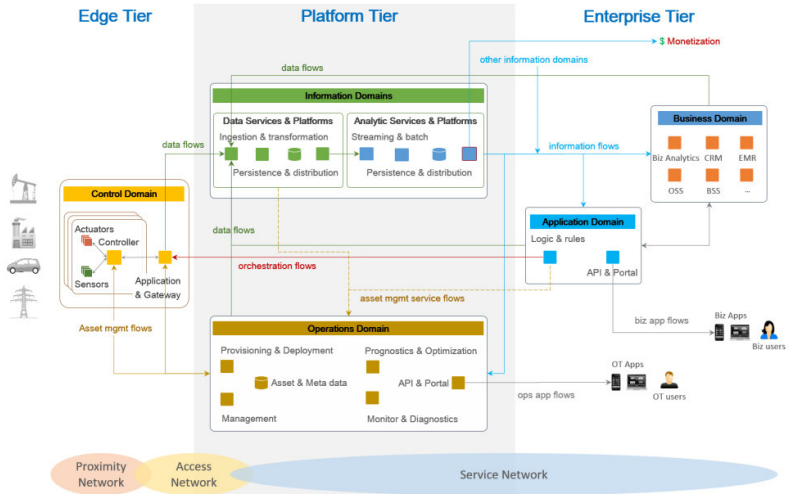


Figura 1.3. Arquitectura referencial del internet de las cosas [5].

1.3.2 Gateways Industriales en aplicaciones IoT

Equipos Comerciales

En el Anexo A, se presentan las funcionalidades y características respecto a interfaces, protocolos de comunicación y adquisición de datos de cuatro gateways comerciales de tipo industrial que se ha podido encontrar en la documentación de los fabricantes [6]-[9].

Se ha observado que todos los modelos implementan protocolos de comunicaciones de campo, generalmente Modbus RTU o Modbus TCP por defecto, y en ocasiones cuentan con entradas y salidas analógicas y digitales. A niveles superiores los protocolos que son más utilizados lo constituyen los protocolos OPC UA y MQTT. En cuanto a las características de los dispositivos cabe destacar la alternativa que presenta el gateway RevPi Connect que se basa en software de código abierto, además es importante recalcar sus características modulares, el sistema implementa buses para comunicarse con módulos de expansión que posibilitan ampliar las funcionalidades del gateway como más interfaces y protocolos de campo, mayor cantidad de entradas y salidas analógicas y digitales, o incluso comunicaciones inalámbricas.

Los protocolos de comunicaciones utilizados tanto de campo como de niveles superiores y la posibilidad de integrar variables analógicas o digitales de entrada y salida serán parámetros a tomarse en cuenta en el establecimiento de los requerimientos del prototipo.

Prototipos de gateways IIoT

En los últimos años se han desarrollado múltiples prototipos de gateways IIoT con varias de las funcionalidades que se han implementado en equipos comerciales como los presentados previamente, a continuación, se revisará brevemente las características, funcionalidades y esquemas de varios trabajos investigados basados en hardware y software.

En [10] se presenta una implementación basada en Raspberry Pi que permite el intercambio de datos entre el usuario y una red de sensores implementada utilizando una tarjeta Arduino UNO, en esta implementación una plataforma Web permite interactuar al usuario con la información recolectada de la red de sensores, para ello la tarjeta Arduino se encarga de adquirir la información de campo, comandar sus actuadores, y comunicarse vía USB a la tarjeta Rasberry Pi, encargada de funcionar como cliente MQTT con el fin de publicar datos de los sensores de campo y subscribirse a tópicos de comandos de la interfaz del usuario. En [11] se propone un gateway basado en raspberry Pi que posibilita el monitoreo y control de una planta a escala, el gateway se encarga de gestionar las comunicaciones de los datos en campo mediante los protocolos S7 y Modbus TCP. Aguas arriba la tarjeta funciona como un cliente MQTT encargado de publicar los datos de campo y subscribirse a tópicos de comandos, para gestionar las comunicaciones en niveles superiores se hace uso de un servidor alojado en la plataforma de IBM. El dashboard es implementado mediante una aplicación desarrollada en Node-RED. Entre otros desarrollos destaca la alternativa de [12] en la que se propone un gateway de código abierto que posibilita la programación de equipos industriales de acuerdo a IEC 61131-3, además de ofrecer una solución de supervisión y control industrial. El sistema consiste en un módulo principal recolector de información de nodos remotos mediante los protocolos MQTT y CoAP y comunicaciones de campo mediante Modbus TCP/IP.

Se han analizado tres implementaciones de prototipos de gateways IoT cada una con sus distintas aplicaciones y arquitecturas, se pudo notar que el protocolo que es mayormente utilizado corresponde al protocolo MQTT para gestionar las comunicaciones con niveles superiores. También cabe destacar el uso de Node-RED del trabajo desarrollado en [11] como una herramienta que permite hacer uso de la información gestionada por el gateway y que posibilita el desarrollo de una interfaz de monitoreo y control.

Dashboards de Monitoreo y Control

En el anexo B, se presentan varias plataformas de desarrollo de tableros de monitoreo y control que servirán de pauta para la implementación de la interfaz del presente trabajo.

Entre las distintas soluciones investigadas en [13]-[15] se puede notar que todas implementan widgets de visualización y de comandos, también implementan gráficas de tendencias en tiempo real y sistemas de gestión de alarmas, estas serán las principales funcionalidades a ser detalladas en la sección de requerimientos.

1.3.3 Protocolos de Comunicaciones

Como se lo pudo observar en las secciones previas de investigación de gateways industriales y de prototipos, los sistemas suelen implementar al menos un protocolo de comunicaciones industriales de campo y un protocolo de mensajería utilizado en la implementación del internet de las cosas, en esta sección se pretende realizar una investigación de los protocolos elegidos para ser utilizados en el desarrollo del trabajo.

De acuerdo con un estudio conducido por HMS Networks en 2020 [16], la tendencia se encuentra marcada por el uso de protocolos como Ethernet/IP, Profinet, Modbus TCP, EtherCAT, entre otros basados en conexión Ethernet. Sin embargo, las tecnologías basadas en buses de campo como Modbus RTU, Profibus DP o DeviceNet aún utilizadas en la industria requieren de su integración a los nuevos estándares de las últimas tecnologías de la información y comunicación. Por esta razón es necesario el desarrollo de dispositivos que posibiliten su integración a los estándares de conectividad desarrollados para el IIoT. Bajo este contexto, el protocolo elegido para su implementación corresponde a Modbus RTU debido a que es uno de los protocolos más utilizados en la industria.

En cuanto al protocolo de mensajería para aplicaciones del internet de las cosas se ha optado por elegir el protocolo MQTT que es uno de los más utilizados, como se lo pudo observar en las secciones previas, por sus características de comunicaciones bidireccionales confiables, sus clientes de bajo consumo de recursos (ideal para su implementación en microcontroladores) y su óptimo uso del ancho de banda como se lo menciona en [17]. A continuación, se presentan los protocolos mencionados:

1.3.3.1 Modbus RTU en Interfaz RS-485

Protocolo Modbus [18]

Es un protocolo abierto de comunicaciones seriales industriales de bajo consumo de recursos de procesamiento y almacenamiento, y su confiabilidad y sencillez [19]. El protocolo establece el mecanismo de comunicación de los dispositivos mediante la técnica de maestro/esclavo en la cual, solo el dispositivo maestro puede inicializar una transacción con otro dispositivo esclavo (a esto se lo denomina consulta). El esclavo con el cuál se está comunicando responde con la solicitud enviando datos o ejecutando una acción que el maestro solicite como se representa en la figura 1.4.

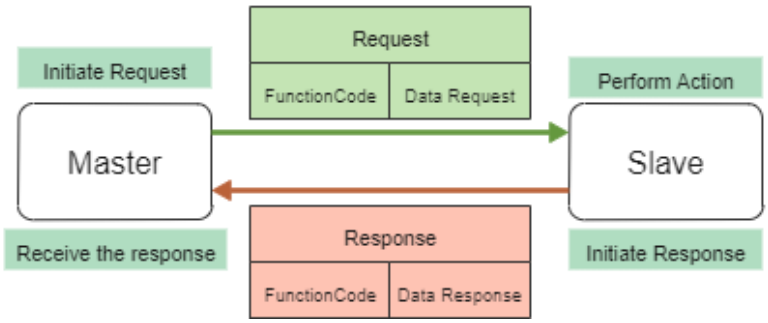


Figura 1.4. Técnica Maestro/Esclavo

El protocolo también define la estructura estándar de los mensajes, la cual debe ser reconocida e implementada por todos los dispositivos comunicantes de manera independiente al tipo de red. También define los formatos para solicitar información y generar respuestas, la disposición y contenidos de los campos de los mensajes, como es que un equipo conoce su dirección, y como reconoce que un mensaje esté destinado para sí mismo, o la acción que le es solicitada que realice y el formato de los datos y la información contenida en el mensaje.

El mensaje de consulta cuenta con los parámetros de dirección, código de función, datos y campo de chequeo de errores. En cuanto al mensaje de respuesta cuenta con los parámetros de confirmación o realimentación de error, datos y campo de chequeo de errores.

En el caso de Modbus RTU se envía los datos en paquetes de 8 bits. La trama de mensajes del protocolo Modbus RTU comienza con un intervalo de silencio de 3.5 caracteres, posteriormente se transmite la dirección del dispositivo, luego el código de función y los datos, un campo de chequeo de errores, y finaliza con un periodo de silencio de 3.5 caracteres como se ilustra en la figura 1.5 y se resume a continuación.

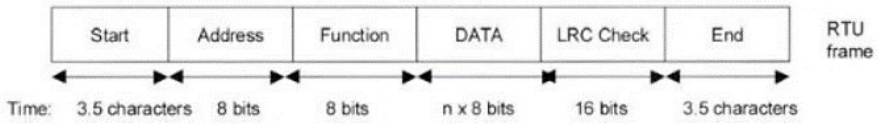


Figura 1.5. Trama Modbus [18].

- Campo de dirección: Se puede tener direcciones de 0 a 247 donde 0 corresponde a la dirección de broadcast y las restantes corresponden a las direcciones de los esclavos.
- Código de función: Campo de 8 bits con un código que permite a los dispositivos informar acerca de la acción que se requiere que se desarrolle. A continuación, se enlistan algunos de los códigos de función más importantes:

Tabla 1.1. Códigos de Función

Code	Nombre	Descripción
01	Read coil status	Solicitud de lectura de status booleano. En la consulta se especifica el registro Coil inicial y la cantidad de registros Coil contiguos a leer.
02	Read input status	Solicitud de estatus de entradas discretas. La función se utiliza de manera similar a la anterior.
03	Read holding registers	Posibilita la lectura de holding Registers de la misma manera que en los casos anteriores, especificando un registro inicial y un número de registros.
04	Read input registers	De la misma manera que en los casos previos posibilita la lectura de varios Input Registers contiguos.
05	Force single coil	Permite cambiar el estado lógico de un solo registro Coil para lo cual es necesario especificar su dirección y estado.
06	Preset single Register	Esta función permite colocar un valor predefinido en un holding register.
07	Read exception status	Permite leer el estado actual de uno de los esclavos.
08	Diagnostics	Permite realizar un diagnóstico de un esclavo.
11	Fetch comm. event counter	Retorna una palabra de estatus y una cuenta de eventos en las comunicaciones del esclavo.
12	Fetch comm. event log	Retorna la cuenta de eventos de comunicaciones, y un campo donde se enlistan los eventos de comunicación.

- Datos: Corresponde a información de direcciones, valores numéricos número de datos.
- Chequeo de errores: Su funcionalidad es efectuar un chequeo de redundancia cíclica.

Interfaz RS-485 [20]

Standard de comunicaciones seriales industriales que posibilita la transmisión de datos mediante cable par trenzado diferencial. Especifica las características de los transmisores y receptores y suele ser utilizado en redes locales, y enlaces de comunicaciones multidrop. A continuación, se presenta las características más relevantes de la interfaz:

Tabla 1.2. Especificaciones del standard RS-485.

Especificación	Detalle
Modo de Operación	Diferencial
Número de transmisores y receptores	32 trasmisores y 32 receptores
Máxima distancia de transmisión	1200 metros
Máxima velocidad de transmisión	10 Mbps
Máximos voltajes del transmisor/receptor	-7V a 12 V
Resistencias terminadoras	120 ohmios
Sensibilidad del receptor	+/- 200mV

La velocidad máxima que se puede alcanzar depende de la longitud entre los dispositivos, a mayor longitud, menor la velocidad y viceversa como se puede observar en la figura 1.6.

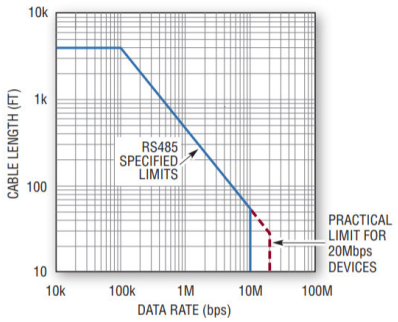


Figura 1.6. Curva de velocidad de transmisión de datos vs. longitud [20].

Los voltajes diferenciales del par A y B representan 3 valores lógicos [21]:

- 1 lógico si (VA-VB) se encuentra entre -1.5 y 6 voltios.
- 0 lógico si (VA-VB) se encuentra entre +1.5 y +6 voltios.
- Alta impedancia: El dispositivo transmisor no toma corriente de la línea y aparenta no estar conectado, para activar el dispositivo se requiere de un pin de control.

El uso del tercer estado posibilita el modo de transmisión half-dúplex en la que se utiliza un solo par trenzado compartido para la transmisión y recepción de información de hasta 32 dispositivos. También es posible la transmisión en modo full-dúplex mediante el uso de 2 pares de cables, uno utilizado para transmitir datos y otro utilizado para recepción de datos.

1.3.3.2 Protocolo MQTT

Es un protocolo ligero y sencillo de mensajería estándar comúnmente usado para el internet de las cosas. Como se lo mencionó previamente, los mensajes tienen cabeceras que optimizan el uso del ancho de banda, está basado en una arquitectura ideal para incorporar dispositivos con recursos limitados como por ejemplo microcontroladores, que pueden interactuar de manera bidireccional. Su capacidad de escalabilidad permitiría incorporar millones de dispositivos de características variadas. Otra de las características de este protocolo consiste en la confiabilidad de la entrega de los mensajes la cual puede ser definida según tres niveles de calidad de servicio QoS (Quality of Service) [17].

La arquitectura Pub/Sub permite desacoplar el cliente que envía el mensaje (Publisher) de los clientes que reciben el mensaje (Subscribers), esto quiere decir que los clientes no interactúan entre sí directamente. De hecho, no conocen la existencia el uno del otro. La conexión se establece mediante un componente adicional llamado Broker cuyo trabajo consiste en filtrar los mensajes y distribuirlos a los subscriptores que los necesiten [22].

También cabe destacar que el protocolo MQTT se encuentra definido en las capas de aplicación, presentación y sesión según el modelo OSI. El cliente y el Broker deben poseer las funcionalidades de las capas de red y transporte TCP/IP [22].

En la figura 1.7 se presenta un esquema básico de implementación de este protocolo:

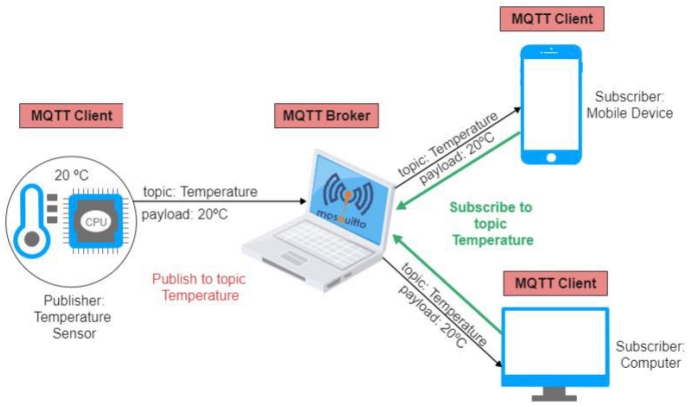


Figura 1.7. Protocolo MQTT.

Como se puede ver se tiene tres clientes MQTT, uno de ellos consiste en un cliente MQTT "Publisher" denominado sensor de temperatura, y los otros dos consisten en 2 clientes MQTT "Subscribers": un dispositivo móvil y un sistema informático. El cliente "Publisher" publica el valor de la temperatura bajo un tópico "temperatura", el dispositivo intermediario Broker permite filtrar el mensaje que contiene el valor de temperatura (20ºC) bajo el tópico "temperatura" y entregarlo a los dos subscriptores que pueden hacer uso de este valor de temperatura ya que se encuentran subscritos al tópico, de ser el caso que uno de ellos no esté subscrito al tópico, el Broker no entregará el mensaje, a este método de filtrado de mensajes se lo denomina filtrado basado en tópico, y será adoptado para este trabajo.

El Broker constituye en un componente fundamental para la implementación, razón por la cual se ha seleccionado la alternativa de código abierto Mosquitto Broker que implementa el protocolo de mensajería MQTT de modelo Pub/Sub. Es generalmente implementado en sistemas embebidos de baja potencia, también en computadoras domésticas, o incluso en servidores de gamas altas. Además, posee clientes MQTT para poder realizar pruebas de conexión y de funcionalidad del Broker desde la consola del sistema [23].

1.3.4 Node-Red

Se ha seleccionado Node-RED para el desarrollo de la interfaz de monitoreo debido a que es una herramienta de libre acceso sencilla de utilizar que posibilita el desarrollo de aplicaciones del internet de las cosas permitiendo integrar desarrollos de hardware y software a servicios web y otros tipos de software. La herramienta se fundamenta en programación visual basada en flujo de datos, los cuales se movilizan a través de varios bloques con distintas funcionalidades, dichos bloques son denominados nodos y permiten procesar información o desarrollar tareas con parámetros de entrada y salidas [24]. Los nodos básicos pueden ser utilizados directamente desde una paleta de nodos en la

herramienta, pero también se puede utilizar desarrollos de la comunidad de Node-RED que pueden realizar múltiples tareas como nodos de acceso a internet, generación de páginas Web, mensajería SMS, e-mail, publicación automática en redes sociales, entre otros.

Los mensajes son objetos JavaScript que contienen al menos un parámetro denominado "payload" que generalmente contiene la información más importante del mensaje. A este parámetro se hace referencia mediante la denominación msg.payload, también se puede añadir otras propiedades manualmente que proporcionen mucha más información de relevancia para su posterior procesado, visualización o uso general.

Otra característica del entorno consiste en el almacenamiento de información. El sistema permite almacenar información que puede ser compartida entre nodos de un mismo flujo, a esto se lo denomina "contexto". Cabe destacar que se almacena en la memoria del sistema mientras Node-RED se encuentre activo en intervalos configurables [25].

1.3.5 La plataforma de Arduino

Arduino es una plataforma de código abierto utilizada en el desarrollo de prototipos electrónicos mediante tarjetas embebidas de bajo costo basadas en microcontroladores que pueden ser programadas mediante el ambiente de desarrollo integrado de Arduino (IDE) [26]. Una de las más importantes características de la plataforma es el acceso libre a múltiples contribuciones y proyectos colaborativos que posibilitan a los usuarios desarrollar aplicaciones con facilidad mediante el uso de librerías para extender las funcionalidades de las tarjetas con módulos de expansiones, de comunicaciones, de almacenamiento, entre otros, tal es el caso de las múltiples librerías desarrolladas para comunicaciones bajo el protocolo Modbus o librerías de clientes MQTT. Es por esta razón que se ha optado por elegir a la plataforma de Arduino para el desarrollo del trabajo, a continuación, se presentará la tarjeta embebida de bajo costo Arduino UNO R3 que ha sido considerada debido a sus características y a la gran cantidad de desarrollos y módulos de expansión disponibles en el mercado para su distribución de pines y periféricos.

Arduino UNO R3 [27] [28]

Tarjeta embebida basada en el microcontrolador de baja potencia y bajo costo Atmega328P, posee 14 pines destinados para entradas y salidas digitales (6 PWM), 6 entradas analógicas, reloj de 16MHz, conector USB-B para programarlo, implementar comunicaciones seriales y alimentación, un conector para alimentación de 9 voltios y acceso a los periféricos disponibles del microcontrolador. Una de las ventajas más importantes consiste en el bajo costo de la tarjeta, sus características técnicas más relevantes se resumen en el anexo C.

1.3.6 Dispositivos adicionales

Para implementar el protocolo MQTT es necesario incorporar las funcionalidades de las capas de transporte y de red TCP/IP, por esta razón es necesario la utilización un módulo Ethernet, cuyas características serán presentadas posteriormente. De la misma manera para implementar las comunicaciones con dispositivos de campo se ha considerado implementar una arquitectura modular que permita implementar más protocolos de comunicaciones industriales en futuros trabajos mediante el uso de sistemas modulares de procesamiento que utilicen el bus I2C, bajo este contexto, también se presentará la tarjeta embebida Arduino Nano utilizada para gestionar las comunicaciones en campo.

Módulo Ethernet Shield R3 [29]

Posibilita la conexión de una tarjeta Arduino a una red informática, se basa en el chip ethernet Wiznet W5100 que implementa las funcionalidades de las capas de red (IP) y transporte (TCP y UDP) soportando hasta 4 conexiones con velocidades de 10/100Mb. El fabricante proporciona una librería para poder trabajar con el módulo, la comunicación con la tarjeta es manejada mediante bus SPI, en Arduino UNO usa los pines 10, 11, 12, 13 (SCK, MISO, MOSI, SS) los cuales ya no pueden ser utilizados como entradas o salidas generales.

Arduino Nano [30] [31]

Es una tarjeta embebida de pequeña dimensión diseñada para su uso en prototipos sobre protoboards, requiere alimentación cableada ya que no cuenta con un conector como Arduino UNO. Se basa en el microcontrolador ATMega328 que utiliza un reloj de 16 MHz, cuenta características similares que ATmega328P utilizado en Arduino UNO, por lo que la programación y acceso a las entradas y salidas analógicos y digitales, así como los periféricos de comunicaciones UART, I2C y SPI se implementan mediante conexiones a todos los mismos pines previamente mencionados, de la misma manera cuenta con la misma capacidad de memoria. Las características técnicas se resumen en el anexo C.

1.3.7 Sensores

Como se lo menciona en el apartado de requerimientos, es necesario incorporar al menos dos sensores que permitan obtener la información de campo, razón por la cual se ha optado por utilizar dos sensores I2C que serán presentados a continuación.

Sensores de Presión Digital BMP180 [32]

Es un sensor de bajo costo utilizado para medir presión barométrica y temperatura.

Puede ser utilizado para determinar la altura sobre el nivel del mar. La placa cuenta con un regulador de 3.3 voltios, un intercambiador de nivel I2C (puede operar en niveles de 5 voltios) y resistencias de pull-up integradas. El sensor consiste en un elemento primario con principio de transducción piezo-resistivo, un conversor A/D y una unidad de control que contiene Interface I2C y EEPROM. El sensor presenta valores no compensados ni interpretados de presión y temperatura por lo que es necesario el uso de un algoritmo de cálculo que hace uso de datos de calibración que permiten compensar offsets, dependencia de temperatura y otros parámetros, además de posibilitar la interpretación de las variables [33]. Las características del sensor se detallan en el anexo C.

Acelerómetro y Giroscopio MPU6050 [34]

Es un dispositivo de tracking de movimiento de 6 grados de libertad, combina un acelerómetro de 3 ejes, un giroscopio de 3 ejes, y una unidad digital de procesamiento de movimiento en el mismo empaquetado. Suele ser utilizado para determinar inclinaciones y orientaciones, además de su uso para estimar desplazamientos lineales. Posee un bus I2C para transmisión de datos.

Cabe destacar que de requerirse valores precisos es necesario implementar un algoritmo de calibración que permita modificar los offsets del sensor. Las características del sensor se resumen en el anexo C.

1.3.8 Red Privada Virtual

Consiste en una conexión encriptada a través del internet de un dispositivo y una red informática. La conexión garantiza que la información crítica sea transmitida de manera segura. Previene accesos no autorizados en el tráfico y posibilita el acceso remoto de los usuarios a sus redes de empresa bajo procedimientos de autentificación. [35]

LogMeIn Hamachi [36]

Permite crear redes privadas virtuales bajo demanda mediante un servicio gestionado y alojado por la misma empresa que garantiza una incorporación segura de equipos como si se tratase de una red de área local, el desarrollador provee múltiples opciones de conexión entre las que cabe destacar para este trabajo, “Mesh networking” que permite implementar de manera práctica y sencilla una red tipo malla que posibilita que las máquinas conectadas en la red interactúen unas con otras, garantizando el acceso a todos los recursos disponibles de los dispositivos. En cuanto a las conexiones, de manera gratuita, se tiene acceso de hasta 5 dispositivos. También cuenta con un control de acceso centralizado, incluyendo manejo de contraseñas, autentificación, entre otras funcionalidades.

2. METODOLOGÍA

2.1 Requerimientos del Sistema

Los requerimientos del sistema a implementarse han sido establecidos en base a las investigaciones de gateways comerciales, prototipos de investigación y aplicaciones de monitoreo y control analizadas en la sección previa. Los requerimientos del prototipo se presentan en la tabla 2.1, los de la interfaz se presentan en la tabla 2.2 y finalmente los requerimientos adicionales se resumen en la tabla 2.3.

Tabla 2.1. Requerimientos del prototipo

Requerimiento	Detalle	Objetivo en la implementación
Capacidad de interacción bidireccional con información de variables digitales y analógicas de campo.	**Entradas Digitales:** 2 entradas digitales (5 o 10 V)	Adquirir los estados de las 2 entradas digitales de voltaje para su representación en la interfaz de monitoreo y control.
	Salidas Digitales: 4 salidas digitales (tipo relé)	Accionar 4 salidas tipo relé según los comandos efectuados por el usuario desde la interfaz de monitoreo y control.
	Entradas Analógicas: 2 entradas analógicas (0-10v y 4-20ma)	Adquirir los valores analógicos de corriente y voltaje para su representación en la interfaz de monitoreo y control.
	Salidas analógicas: 1 salida analógica (0-10V)	Modificar el valor de la salida de voltaje según la consigna del usuario ingresada desde la interfaz de monitoreo y control.
Comunicación con un dispositivo de campo que trabaje con el protocolo Modbus RTU	El prototipo debe ser capaz de interactuar con los registros del dispositivo Modbus de: **Escritura:** Coil (00001-00008) Holding (40001-40004) **Lectura:** Contact (10001-10008) Input (30001-30004) La interface utilizada es RS-485 en modos half-dúplex y full-dúplex	Adquirir los estados y valores de los registros de lectura para su representación en la interfaz. Modificar los registros de escritura según los parámetros ingresados en la interfaz Posibilitar la trasmisión y recepción de información mediante la interfaz RS-485 en modos half-dúplex y full-dúplex
Integración de sensores de variables físicas.	Se puede obtener valores de variables físicas de dos sensores	Adquirir información de los sensores para su representación en la interfaz
Funcionalidad de cliente MQTT Pub/Sub.	El prototipo cuenta con el hardware y software requerido para funcionar como cliente MQTT con el fin de publicar estados y valores de variables y suscribirse a tópicos de comandos de la interfaz.	Implementar las funcionalidades de las capas de transporte y red TCP/IP. Gestionar las comunicaciones con el Broker de más de 40 tópicos de publicación y subscripción de las variables de entrada y salida de campo.

Tabla 2.2. Requerimientos de la interfaz de monitoreo y control

Requerimiento	Detalle	Objetivo en la implementación
Desarrollo de un dashboard de monitoreo y control de todas las variables consideradas.	Funcionalidad principal, permite la representación de variables de entrada mediante luces indicadoras, medidores, entre otros, y el comando de las salidas mediante interruptores, sliders, entre otros.	- Permitir la representación de estados booleanos y valores numéricos de entradas A/D, sensores y de los registros Contact e Input del dispositivo Modbus. - Posibilitar el accionamiento de las salidas A/D y de los registros Coil y Holding del dispositivo Modbus.
Gráficas de tendencias en tiempo real de varias variables	Aplicación de la interfaz de monitoreo y control cuya funcionalidad corresponde a la visualización en tiempo real de algunas variables de entrada en una gráfica de tendencias.	Representar mediante gráficas de tendencias en tiempo real los estados y valores numéricos de variables de entrada con intervalos de tiempo mayores a 10 segundos.
Sistema básico de gestión de alarmas	Aplicación de la interfaz de monitoreo que posibilita la configuración, activación, registro y notificación de varias alarmas de estado de algunas variables de entrada.	-Posibilitar la configuración (activación, registro y notificación vía correo electrónico) de varias alarmas. -Representar y registrar en una tabla las alarmas generadas. -Generar notificaciones mediante correo electrónico.

Tabla 2.3. Requerimientos adicionales de la implementación

Requerimiento	Detalle	Objetivo en la implementación
Instalación y configuración de Broker MQTT	La implementación requiere del uso de un Broker en un sistema que pueda gestionar las comunicaciones entre los clientes MQTT	Posibilitar la comunicación del prototipo como cliente MQTT
Instalación y configuración de un servicio de VPN para acceso remoto	Se debe instalar y configurar el software necesario para que pueda accederse a la interfaz de manera remota	Posibilitar el acceso remoto a la interfaz mediante el uso de un servicio de red privada virtual

2.2 Arquitectura del Sistema

Las funcionalidades del prototipo se ilustran en la arquitectura presentada en la figura 2.1 y corresponden a la adquisición de datos de campo a través del gateway IIoT de manera directa e indirecta. En el primer caso, la adquisición de datos es efectuada a través de entradas y salidas analógicas/digitales de la misma tarjeta. En el segundo caso, se hace uso de un bus I2C que permite incorporar sensores y dispositivos de campo por medio de una tarjeta intermediaria que trabaje como esclavo I2C. En cualquiera de los casos, una interfaz de monitoreo posibilita el acceso y visualización de los datos. Los componentes de la arquitectura son analizados a mayor detalle a continuación.

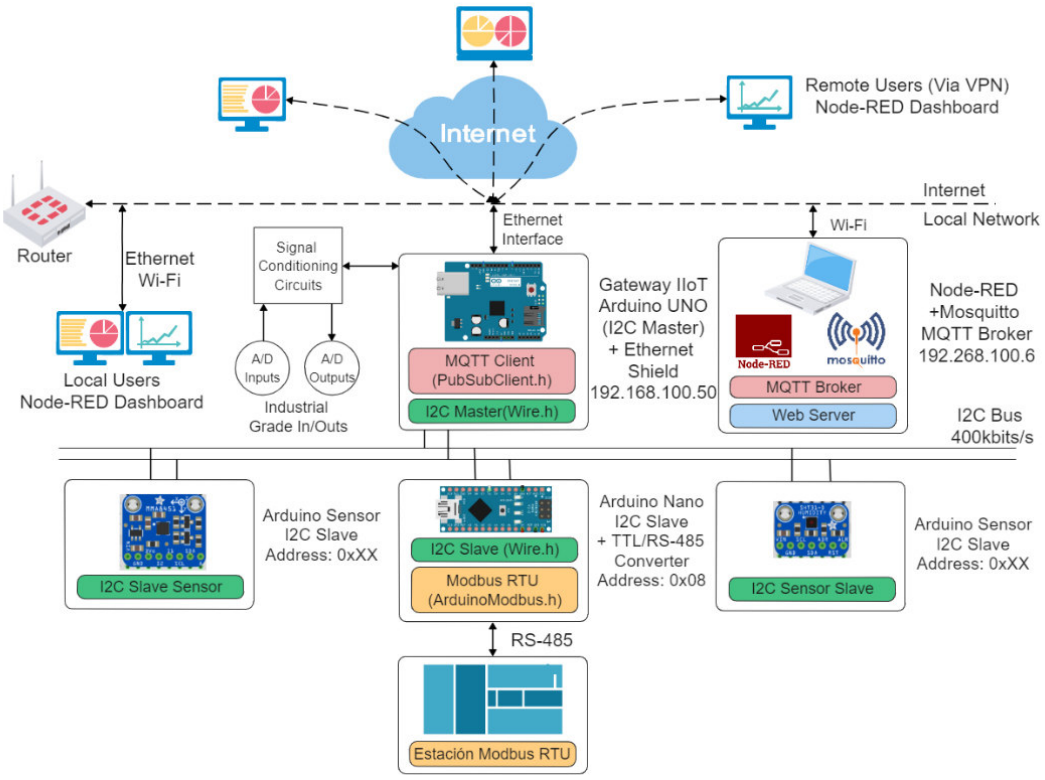


Figura 2.1. Arquitectura del Sistema.

- Gateway IIOT Arduino UNO (I2C Master) + Ethernet Shield: Su función es actuar como cliente MQTT capaz de publicar la información de campo y subscribirse a tópicos de comandos. Se encarga de procesar, gestionar y actuar de acuerdo al flujo de información bidireccional hacia la misma, para ello incorpora las funcionalidades de las capas de transporte y red TCP/IP mediante el módulo Ethernet, el uso de la interface implica la necesidad de las librerías Ethernet y SPI. Para implementar las funcionalidades del cliente MQTT se debe hacer uso de la librería “PubSubClient”. Cada uno de los mensajes se encuentra sujeto a un tópico de publicación o subscripción. La publicación de datos se realiza de manera continua como es el caso de las variables de entrada, garantizando que la interface se mantenga actualizada. La subscripción a los tópicos es realizada al principio del programa, la librería permite la recepción de los datos al hacer referencia a una función de callback que permite procesar la información cada que existe datos, esto permite actuar sobre variables o registros de salidas.

  Para obtener datos del dispositivo Modbus y de los sensores la tarjeta hace uso de las funcionalidades del bus I2C como maestro que permite la recolección de los datos. Las comunicaciones son inicializadas a una velocidad de transmisión de 400kbits/s. Los datos de los esclavos son procesados almacenados y enviados aguas arriba y la información de salida desde el maestro permite actualizar los registros de los esclavos en el lazo principal al enviar y recibir datos mediante el bus I2C (Figura 2.2.)

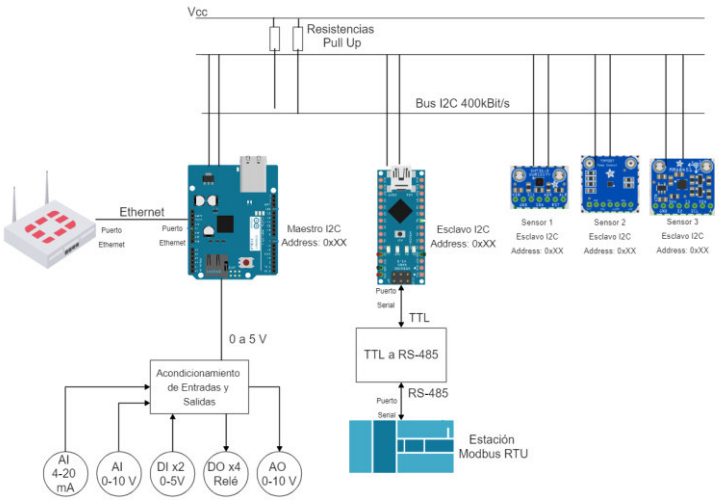


Figura 2.2. Bus I2C.

El uso del bus I2C posibilitaría la integración de más dispositivos que puedan obtener información de otros tipos de equipos que pueden manejar distintos protocolos o interfaces, diferentes sensores o posibilitaría extender la cantidad de entradas y salidas disponibles. Permitiendo al prototipo adaptarse de manera modular a las necesidades que puedan presentarse. El número de dispositivos está limitado por el número de direcciones que el maestro puede manejar, en el caso de Arduino los esclavos pueden utilizar las direcciones 8 a 127 a la velocidad máxima de 400kbits/s. El desacople o falla de uno de los esclavos no comprometería las funcionalidades del sistema. El número de tópicos se ve limitado por la memoria del sistema. En la figura 2.3 se ilustra las características modulares previamente mencionadas, se podría utilizar cualquier tipo de tarjeta embebida para la adquisición de datos.

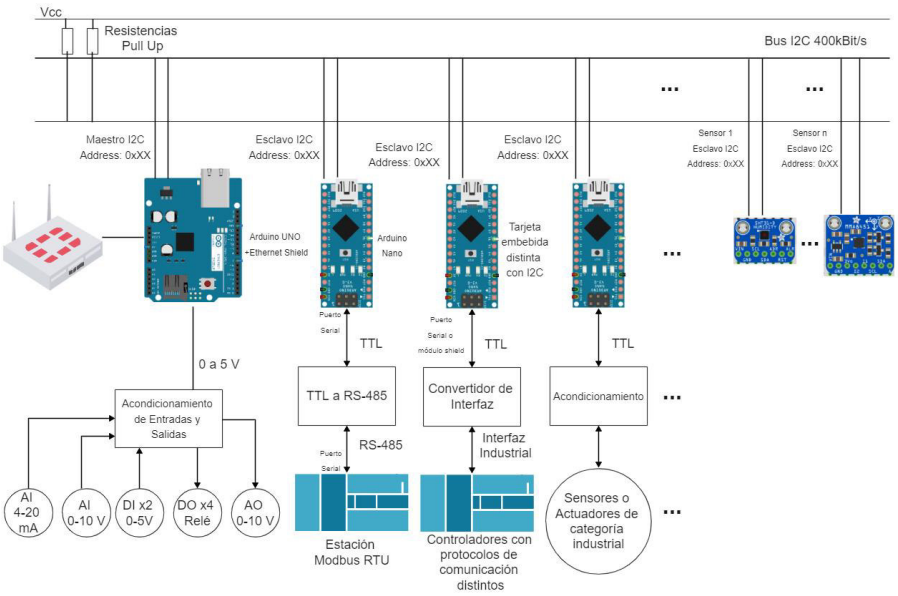


Figura 2.3. Dispositivos Adicionales en Bus I2C

- Arduino Nano (Esclavo I2C) + convertidor TTL/RS-485: Permite gestionar los registros modbus de entrada y salida del dispositivo de campo a través de una interfaz serial. La comunicación de los registros aguas arriba se la realiza mediante I2C hacia el maestro encargado de funcionar como cliente MQTT. Cabe destacar que es necesario cierta circuitería para transformar las señales seriales de voltaje TTL a las requeridas para la interfaz RS-485 considerando los modos de transmisión half-dúplex y full-dúplex.
- Node-Red + Mosquitto MQTT Broker: Son implementados en una computadora convencional que se encuentra en la misma red que el gateway. En primera instancia el Broker se encarga de gestionar las comunicaciones de los mensajes filtrándolos según sus tópicos. Node-Red permite hacer uso de los datos mediante nodos MQTT y nodos que permiten incorporar Widgets de visualización y funcionalidades adicionales como la visualización de datos en tiempo real y la gestión de alarmas.
- Acondicionamiento de Entradas y Salidas: Corresponde a circuitería y módulos complementarios que permiten adaptar las señales eléctricas a los niveles de la tarjeta.
- Sensores Esclavos I2C: Sensores que permiten obtener datos de variables físicas.
- Router: Corresponde a un dispositivo que permite interconectar distintos equipos informáticos en una misma red local, también posibilita el acceso al internet.
- Usuarios Locales dispositivos informáticos con acceso local a la interfaz de monitoreo.
- Usuarios remotos: Usuarios en distintas redes remotas que pueden interactuar con la interfaz mediante el uso de una VPN gestionada mediante el servicio de Hamachi.

2.3 Diseño de Hardware

2.3.1 Alimentación

El dispositivo puede ser alimentado mediante conexión USB o mediante una fuente de alimentación de 7 a 12 voltios, dependiendo de la corriente que sea utilizada en los demás circuitos. Mediante la conexión USB la corriente máxima que puede ser suministrada se limita a 400 mA, para consumos superiores es recomendable utilizar un suministro externo, por esta razón se procuró que se alimente a los circuitos de acondicionamiento mediante un suministro externo, en especial para los sistemas de consumo considerable.

2.3.2 Módulo Principal

En la figura 2.4 se presenta el diagrama de entradas y salidas del módulo principal cuyas funciones se detallan en la tabla 2.4.

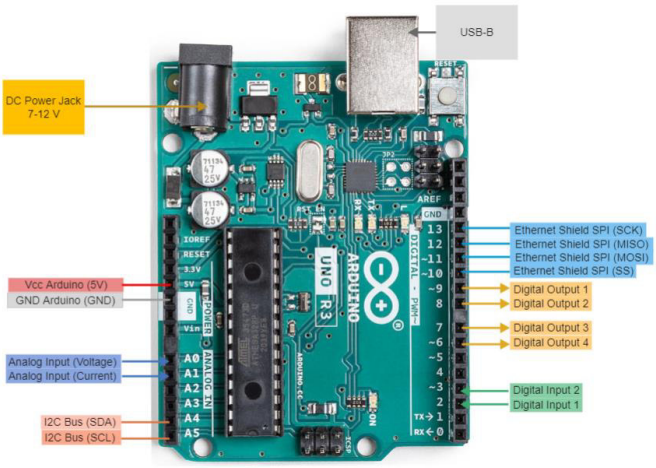


Figura 2.4. Diagrama de entradas y salidas

Tabla 2.4. Señales de entrada y salida del módulo principal

Señal	Tipo	Pin	Detalle
DC Power 7-12 V	Suministro de voltaje	N/A	Alimentación de la tarjeta con una fuente de voltaje externa opcional de 7 a 12 voltios
Vcc Arduino		5V	Permite el acceso al suministro de la tarjeta para alimentar otros circuitos.
GND Arduino		GND	Permite el acceso al suministro de energía de la tarjeta para alimentar otros circuitos.
USB-B	Suministro, Program. y Depuración.	N/A	Permite alimentar a la tarjeta, posibilitando el consumo de corriente hasta 400 mA, también es utilizado para programar la tarjeta o para realizar depuraciones en el código.
An. Input (Voltage)	Entradas analógicas	A0	Adquisición de señal analógica de voltaje ya acondicionada (0 a 5 voltios)
An. Input (Current)		A1	Adquisición de señal analógica de la entrada de corriente acondicionada a voltaje (0 a 5 voltios.)
Digital Output 1-4	Salida Digital	10-13	Permite generar la señal de activación digital que controla el módulo de relés.
Digital Input 1-2	Entrada Digital	2-3	Permite ingresar los estados lógicos de las entradas conectadas al prototipo.
SDA	Bus I2C	A4	Permite a la tarjeta embebida comunicarse con dispositivos esclavos I2C
SCL		A5	
MOSI	SPI Ethernet Shield	11	Permite la comunicación con el módulo Ethernet Shield
MISO		12	
SCK		13	
SS		10	

2.3.3 Acondicionamientos y Circuitos de Pruebas

Las entradas digitales son acondicionadas mediante dos optoacopladores al voltaje de 5 voltios de la tarjeta embebida. El circuito de prueba consiste en dos pulsadores que permiten abrir o cerrar un circuito resistivo, permitiendo cambiar los estados lógicos de la entrada entre los niveles de voltaje de 10 y 0 (12 y 0 voltios por condiciones del suministro). Del datasheet del dispositivo [37] se puede notar que uno de los valores normales de

operación del dispositivo constituye la corriente forward de 20 mA y un voltaje forward de 1.2 voltios la resistencia limitadora sería estimada de la siguiente manera,

$$R = \frac{10-1.2\ V}{20mA} = 440\ ohmios \tag{2.1}$$

Considerando el valor standard de 470 ohmios la corriente que circularía constituiría:

$$I_F = \frac{10-1.2V}{470} = 18.723\ mA \tag{2.2}$$

Considerando que la tensión del circuito de prueba es de 12 voltios:

$$I_F = \frac{12-1.2V}{470} = 22.979\ mA \tag{2.3}$$

Se puede notar que no se supera los límites máximos de 1 amperio, el sistema funcionaría sin inconvenientes para el rango de 0 a 10 voltios y tampoco presentaría inconvenientes para la prueba con suministro de 0 a 12 voltios.

Del lado del transistor se puede notar que el voltaje aplicado no supera los límites establecidos en la hoja de datos y que la corriente que circula por el transistor en estado de saturación ($V_{CE(SAT)} = 0.1\ V$) es:

$$I_c = \frac{5-0.1\ V}{10k\ ohmios} = 0.49\ mA \tag{2.4}$$

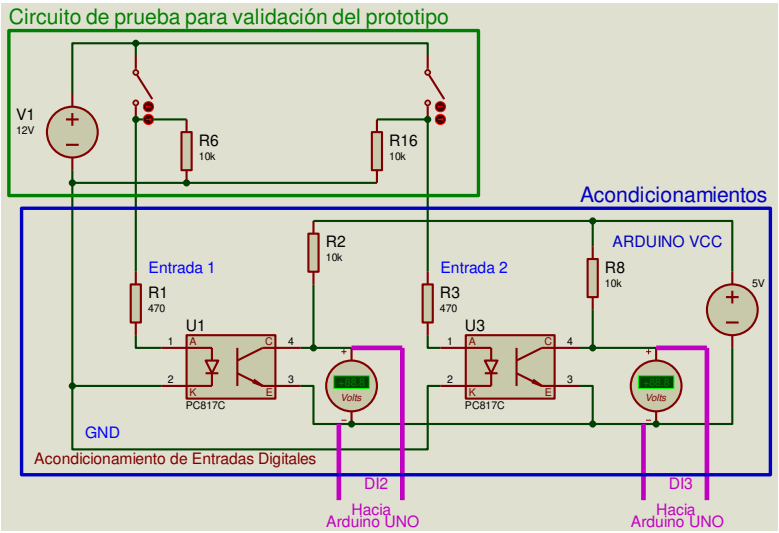


Figura 2.5. Acondicionamiento y Circuito de pruebas de entradas digitales.

Para el acondicionamiento de salidas digitales se ha considerado adquirir un módulo de relés de cuatro canales. El módulo consiste en 4 relés de 5 voltios activados mediante 4 señales de control ópticamente aisladas, Los relés pueden manejar cargas de hasta 250VAC o 30VDC con corrientes de hasta 10 amperios. Los pines del módulo son enlistados en la tabla 2.5 junto a una breve descripción de su función [38]:

Tabla 2.5. Pines de conexión del módulo de relés

Pin	Descripción
GND	Pin de referencia a tierra (GND Arduino)
IN1 a IN4	Señal de control de relés 1 al 4
Vcc	Alimentación para del módulo (5 voltios)
Vcc y JD-Vcc	Jumper de selección de suministro

Se utilizará una fuente externa para proporcionar el voltaje de activación de los relés por lo que se debe quitar el jumper para alimentarlos de manera independiente mediante el pin JD-Vcc. Para las señales de control es necesario alimentar al relé en Vcc con el suministro de Arduino, las señales de control de los pines IN1-4 permiten cambiar el estado del relé respectivo. El módulo junto a sus partes constitutivas se presenta en la figura 2.6.

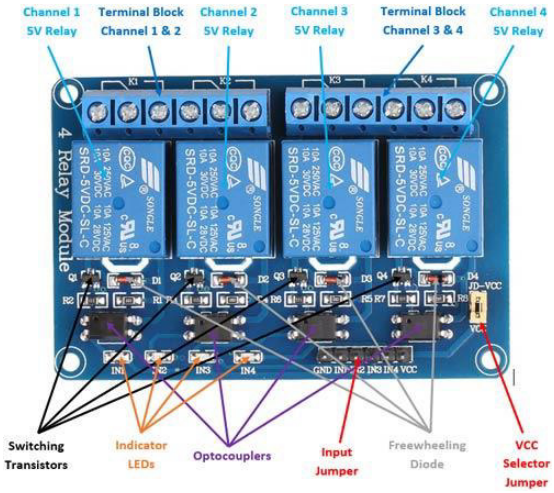


Figura 2.6. Partes del módulo de relés [38]

El circuito de acondicionamiento de la entrada analógica de voltaje consiste en un amplificador diferencial que permite tomar la señal analógica de entrada de hasta 10 voltios y disminuirla al nivel máximo de 5 voltios. Para el circuito de prueba se usa un divisor de tensión que permite restringir la tensión de la alimentación de 12 voltios hasta 10 voltios sobre un potenciómetro. El diseño se presenta a continuación:

La función de transferencia de la configuración responde a la siguiente función:

$$V_{out} = \frac{R2}{R1} * (V2 - V1) \tag{2.5}$$

Para condiciones máximas Vout=5 V2-V1=10

$$10 = \frac{R2}{R1} * 5 \tag{2.6}$$

Considerando R2=10k ohmios, la resistencia R1 debería ser 20k ohmios. El circuito de acondicionamiento se presenta en la figura 2.7 junto al circuito de prueba:

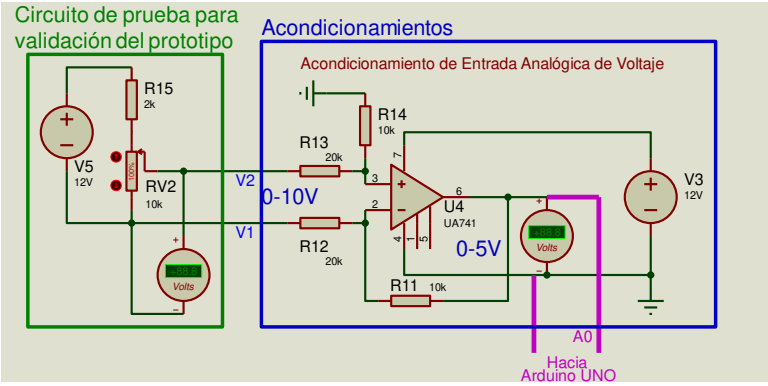


Figura 2.7. Acondicionamiento y circuito de pruebas de la entrada analógica de voltaje.

Para el acondicionamiento de la entrada analógica de corriente se consideró la adquisición de un módulo que permita convertir la señal de corriente de 4 a 20 mA a una señal de voltaje, el módulo adquirido corresponde a XY-ITOV, el cual es presentado en la figura 2.8.

Figura 2.8. Módulo conversor de corriente a voltaje [39]

El módulo cuenta con dos potenciómetros de calibración, el primero (ZERO) se lo utiliza para corregir el valor de voltaje a la salida cuando la corriente es la mínima, el segundo (SPAN) permite modificar el rango máximo a la salida cuando la entrada es 20mA.

El voltaje a la salida depende del voltaje de alimentación colocado en los extremos de la bornera de tres tornillos y la adecuada configuración de las conexiones de los terminales 1 a 4 mediante jumpers, para la salida requerida de hasta 5 voltios es necesario conectar los pines 1 y 2; y los pines 3 y 4. Para el máximo rango la alimentación debe ser de 7 a 36 voltios, sin embargo, si se utiliza voltajes menores se mantiene una relación lineal de voltaje y corriente ajustable mediante los potenciómetros. Otra de las características del módulo constituye el uso de una resistencia de 100 ohmios de precisión y de baja desviación por temperatura para la medición de corriente.

Para poder testear el circuito se implementó una topología basada en el circuito encontrado en [40] que permite modificar la corriente circulante en una carga según el voltaje a la entrada, cabe destacar que la configuración fue modificada para su funcionamiento ante el valor de la resistencia de medición del módulo que es de un valor de 100 ohmios. El circuito implementado se presenta en la figura 2.9.

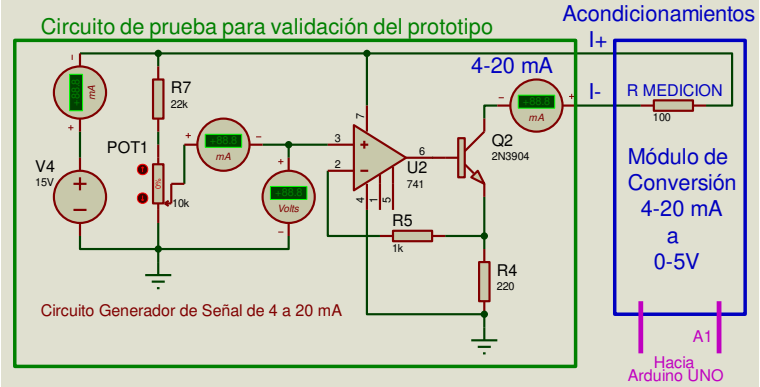


Figura 2.9. Circuito de prueba de la entrada analógica de corriente.

Para generar la salida analógica se hizo uso del conversor digital a analógico MCP4725 (presentado en la figura 2.10) del fabricante Microchip, el conversor es de un solo canal con una resolución de 12 bits, también cuenta con una interface I2C, la dirección del dispositivo es 0x60 por lo que no causa conflictos con los demás dispositivos del bus.

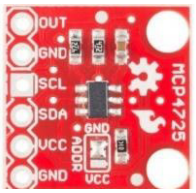


Figura 2.10. Conversor MCP4725. [41]

Cabe destacar que la salida del conversor depende del voltaje de referencia mediante el cual se alimenta el dispositivo, el fabricante proporciona una ecuación en la hoja de datos [42] que permite obtener dicho valor de voltaje:

$$V_{out} = \frac{V_{ref} * D_n}{4096} \tag{2.7}$$

Si el voltaje de referencia es menor, la salida del convertidor también puede disminuir. La salida del conversor es posteriormente acondicionada a un voltaje de 0 a 10 voltios mediante un amplificador operacional en configuración no inversor.

En primera instancia se considera el voltaje de entrada de 0 a 5 voltios y el voltaje de salida de 0 a 10 voltios, de la configuración del amplificador no inversor se puede obtener la siguiente expresión,

$$\frac{V_{out}}{V_{in}} = 1 + \frac{RB}{RA} \tag{2.8}$$

Para los valores máximos de voltaje se tiene,

$$\frac{10}{5} = 1 + \frac{RB}{RA} \tag{2.9}$$

Por lo tanto, las resistencias deben ser iguales, garantizando que la ganancia sea el doble, RA=RB=10k ohmios. Para la prueba de funcionamiento simplemente se coloca un voltímetro a la salida del amplificador operacional. El circuito se presenta en la figura 2.11.

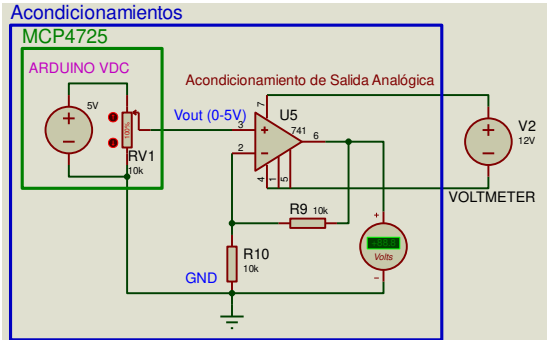


Figura 2.11. Acondicionamiento de la salida analógica de voltaje

2.3.4 Bus I2C

Como se lo mencionó en las secciones previas los dispositivos se comunicarán entre sí haciendo uso del bus I2C a velocidades de 400kHz, los dispositivos que se encuentran integrados en el sistema se presentan en la figura 2.12 junto a sus direcciones.

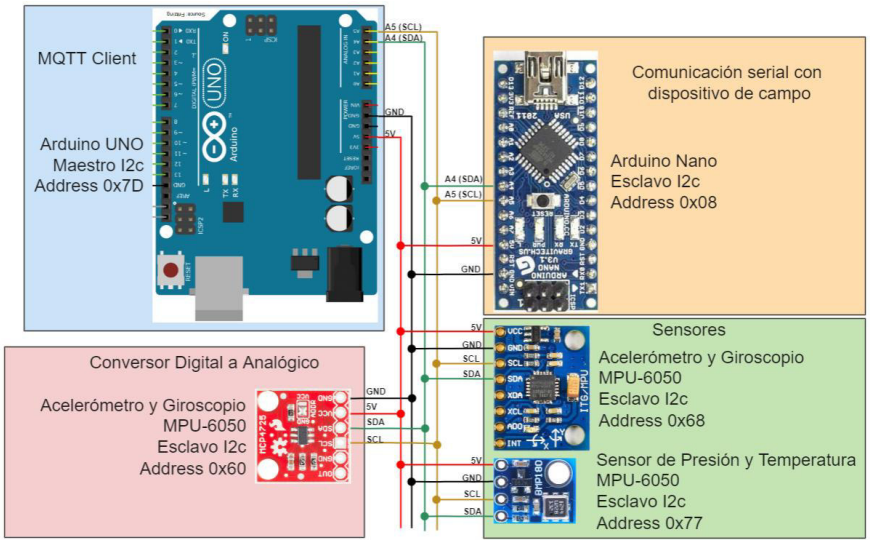


Figura 2.12. Dispositivos y conexiones del Bus I2C

2.3.5 Interfaz RS-485

Para implementar la interfaz RS-485 es necesario utilizar un conversor de nivel de voltaje que permita adaptar los voltajes de transmisión y recepción TTL (0-5 voltios) a señal diferencial Va-Vb, por esta razón es preciso la adquisición del módulo convertidor RS-485 a TTL basado en el circuito integrado MAX485, el módulo integra la circuitería requerida para poder utilizar las funcionalidades del integrado sin mayores complicaciones. El detalle de los pines se presenta el datasheet [43] del cual se puede concluir que el dispositivo puede trabajar como transmisor o receptor según los estados de los pines RE y DE, si

ambos se encuentran en bajo se encontrará en modo de recepción, si ambos se encuentran en alto se encontrará en modo de transmisión. Esta característica nos permitiría implementar los modos de transmisión full dúplex y Half dúplex al cortocircuitar dichos terminales y ponerlos a un solo estado o al controlarlos simultáneamente con dos salidas de la tarjeta, dichos modos son detallados a continuación:

Para modo de transmisión full-duplex (Figura 2.13) se requiere módulos independientes para transmisión y recepción, de la tarjeta embebida se requiere los pines TX y RX, los terminales DE y RE del módulo convertidor deben ser conectados a tierra para el receptor RX y se encontrarían conectados a 5 voltios para la transmisión de TX. Para realizar las pruebas se ha considerado utilizar un simulador por lo que resulta conveniente el uso de un conversor de USB a TTL que se encuentre conectado en la misma configuración que el prototipo hacia dos módulos independientes para transmisión y para recepción, los pares AB de transmisión de la tarjeta son conectados a los terminales AB de recepción de los módulos de prueba, de manera análoga se conecta el par AB de recepción de la tarjeta hacia los terminales AB de transmisión del módulo de prueba.

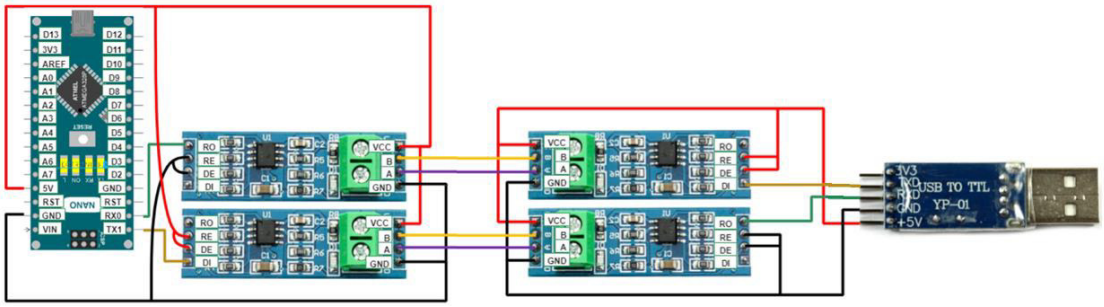


Figura 2.13. Modo de transmisión full-dúplex

Para modo de transmisión half-duplex (Figura 2.14) se requiere un solo módulo para transmitir y recibir datos, pero se requeriría de una o dos salidas digitales adicionales que controlen ambos pines de habilitación DE y RE. El prototipo posee la posibilidad de trabajar en ambos modos de transmisión con tan solo cambiar la configuración del hardware. Para el desarrollo de las pruebas se ha considerado el uso de un conversor USB a RS-485.

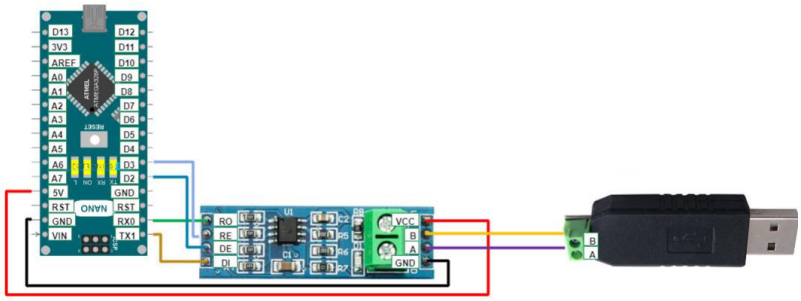


Figura 2.14. Modo de Transmisión half-dúplex.

2.4 Diseño de Software

El software fue implementado sobre las tarjetas embebidas Arduino UNO y Arduino Nano, los siguientes diagramas de flujo fueron las pautas para el desarrollo de la programación, detallados y esquematizados en orden, se presentan a continuación.

2.4.1 Módulo Principal

En el diagrama de flujo general se esquematiza las funciones del módulo principal encargado de gestionar las comunicaciones con los esclavos I2C, leer y actualizar sus entradas y salidas y funcionar como cliente MQTT encargado de publicar datos o subscribirse a comandos, cabe destacar que la librería utilizada permite procesar los datos ligados a tópicos subscritos solo al ser actualizados, mientras no llega una actualización de un dato el sistema seguirá publicando mensajes, receptando o enviando información a los dispositivos I2C o de sus entradas y salidas A/D, razón por la cual el proceso de actualización de datos ligados a tópicos subscritos debe ser esquematizado como un proceso similar a una interrupción como se presenta en el diagrama de la figura 2.15.

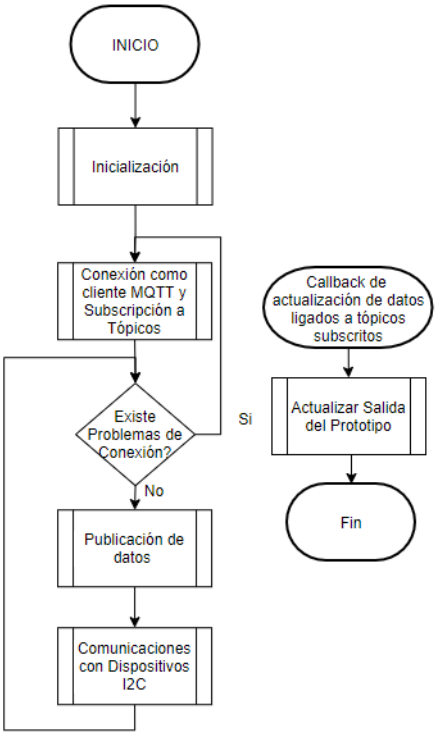


Figura 2.15. Diagrama de flujo general del módulo principal

2.4.1.1 Inicialización

El proceso de inicialización es esquematizado en cinco procesos en los cuales se realiza las configuraciones iniciales de las comunicaciones, las entradas y salidas y los sensores. Como se presenta en el diagrama de flujo de la figura 2.16

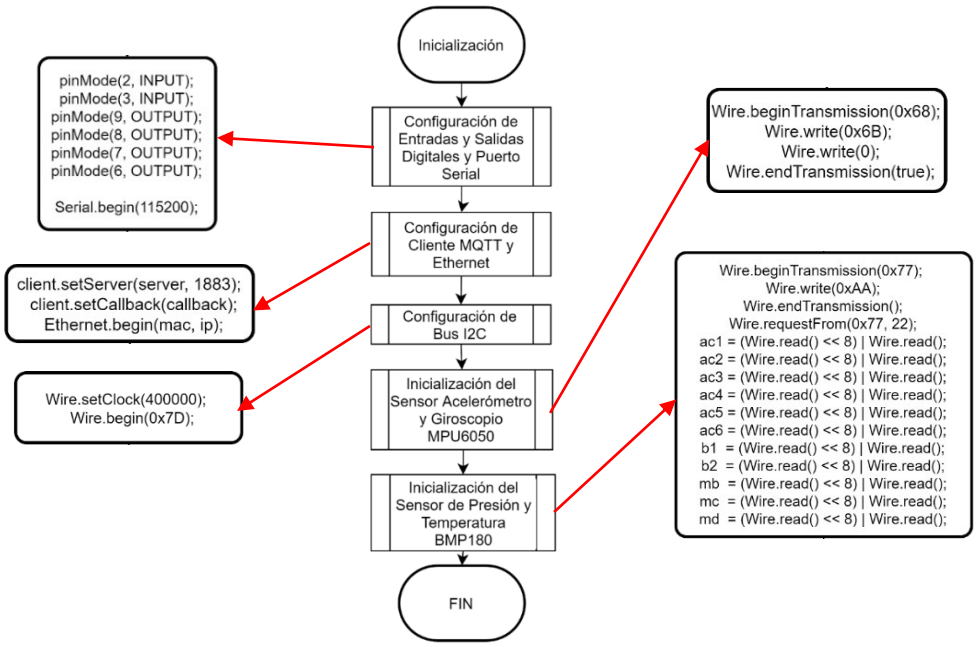


Figura 2.16. Inicialización del módulo principal.

Las entradas y salidas digitales del Arduino son configuradas y se inicializa comunicaciones seriales (115200 baudios) para poder realizar depuraciones en el código o para visualizar ciertas operaciones realizadas por el sistema. Posteriormente se configura la dirección IP del Broker, la dirección IP y dirección MAC del dispositivo, y el puerto mediante el cual el cliente se comunica con el Broker. Luego se realiza la configuración del Bus I2C, el maestro es configurado con una dirección 0x7D y la velocidad es establecida en 400kbit/s. Finalmente se realiza las inicializaciones de los dos sensores del bus I2C:

- Sensor MPU6050: Para despertar a la unidad de procesamiento (Dirección 0x68) se accede al registro PWR_MGMT_1 (0x6B) y se escribe un 0, el sensor podría ser calibrado para obtener resultados más precisos de ser requerido.
- Sensor BMP180: Se adquiere las constantes necesarias para implementar el algoritmo presentado del fabricante para adquirir valores interpretables de presión, temperatura y altura, la obtención de las constantes se presenta en el diagrama de la figura 2.16, donde se inicia las comunicaciones con el dispositivo 0x77 y se escribe la dirección del registro 0xAA desde el cual se van a leer 22 registros contiguos correspondientes a los 11 coeficientes como se presenta en el mapa de memoria de la tabla 2.6

Tabla 2.6. Mapa de Memoria de coeficientes de calibración del sensor BMP180.

	Parámetro	**AC1**	**AC2**	**AC3**	**AC4**	**AC5**	**AC6**	**B1**	**B2**	**MB**	**MC**	**MD**
Reg.	**MSB 0x..**	AA	AC	AE	B0	B2	B4	B6	B8	BA	BC	BE
	LSB 0x..	AB	AD	AF	B1	B3	B5	B7	B9	BB	BD	BF

2.4.1.2 Conexión como Cliente MQTT y Subscripción a Tópicos

El cliente debe realizar un proceso de conexión al servicio del Broker, también se debe realizar la subscripción del dispositivo a los tópicos de comandos, esto se lo realiza una sola vez, los tópicos son declarados como variables globales en memoria y serán detallados posteriormente, el diagrama de flujo del proceso se presenta en la figura 2.17. En la tabla del anexo D se detalla todos los tópicos de subscripción considerados.

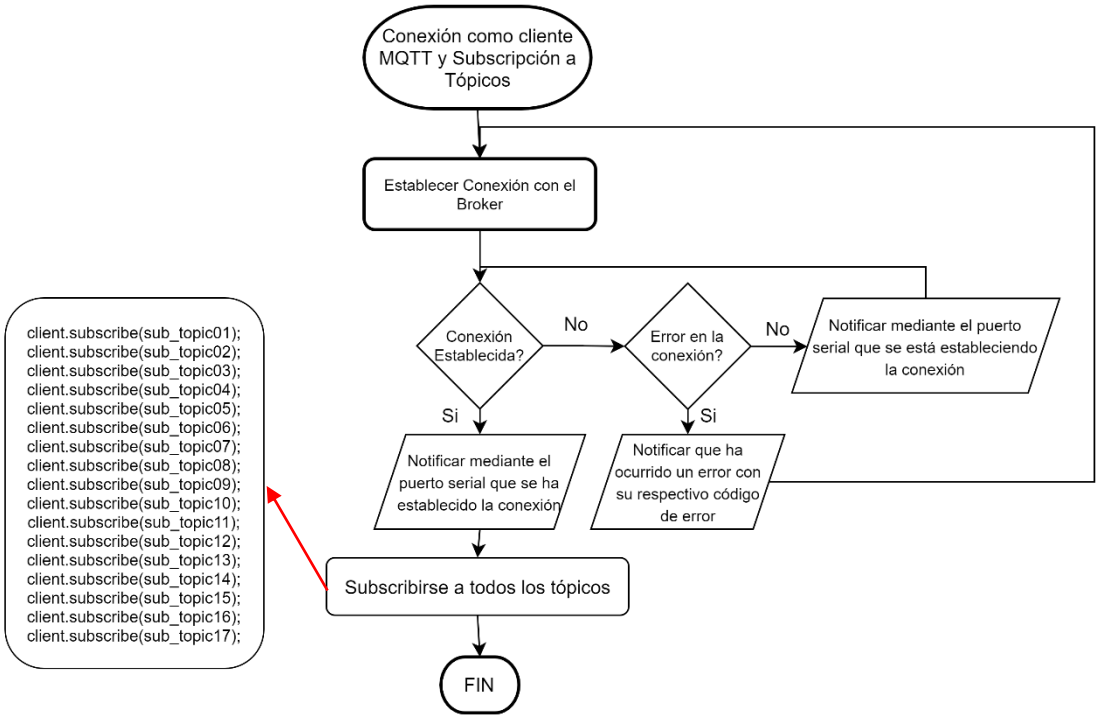


Figura 2.17. Proceso de conexión del cliente y subscripción a tópicos.

2.4.1.3 Publicación de los datos

Para la publicación se hará referencia mediante variables globales a los tópicos de publicación (resumidos en el anexo D), sus datos asociados son publicados continuamente dentro del lazo principal para garantizar que los datos sean actualizados frecuentemente y que su evolución sea registrada en la interface de monitoreo, los datos de los esclavos I2C son adquiridos y almacenados en variables globales de igual manera que las entradas digitales y las entradas analógicas cuyos estados son adquiridos y procesados previamente a que la función de publicación sea ejecutada. Para este proceso es requerido transformar los datos a cadenas de caracteres. Cabe destacar que es necesario implementar la función client.loop() que garantiza que se mantenga la funcionalidad de cliente MQTT. En la figura 2.18. Se presenta el diagrama de flujo del proceso de publicación de los datos como cliente MQTT, para publicar un dato en la función se debe especificar el tópico asociado y el mensaje que correspondería a la cadena de caracteres.

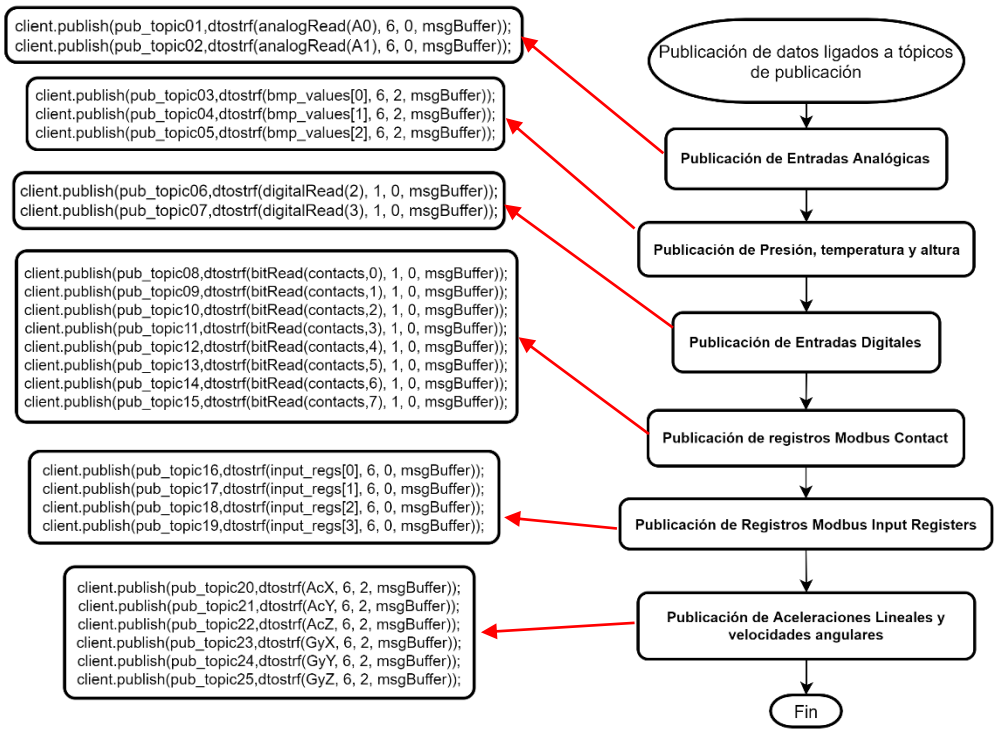


Figura 2.18. Proceso de Publicación de datos.

2.4.1.4 Comunicación con Dispositivos I2C

Para las comunicaciones con los dispositivos esclavos se debe tomar en cuenta que se envía un byte cada vez, si se requiere enviar un registro con un número mayor de bytes estos deben ser concatenados en el registro objetivo, de manera similar, los registros de escritura deben ser enviados separando cada uno de los bytes. Otras de las consideraciones que se debe tomar en cuenta lo constituyen las direcciones de los dispositivos y las direcciones de los registros que van a requerirse para su lectura o escritura. En el diagrama de flujo de la figura 2.19 se identifica cuatro subprocesos: el primero describe la comunicación del maestro I2C con el Esclavo Arduino Nano, luego dos subprocesos de obtención de datos del sensor MPU6050 y del sensor BMP180 y el subproceso de actualización de la salida del conversor digital a analógico MCP4725.

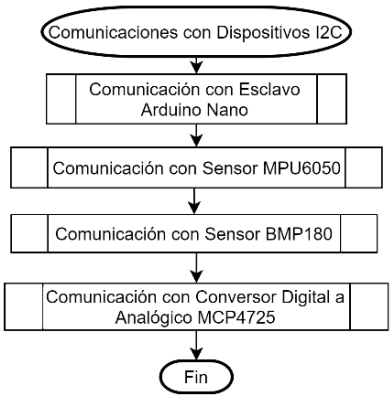


Figura 2.19. Proceso de Comunicaciones con Dispositivos I2C.

Para el caso del subproceso de comunicaciones con el esclavo Arduino Nano (presentado en la figura 2.20.) se inicializa la transmisión de 9 bytes de datos con el dispositivo de dirección 0x08, de estos bytes el primero corresponde a los estados de los 8 registros Coil binarios adquiridos por el cliente MQTT, los 8 bytes restantes corresponden a 4 holding Registers de 2 bytes cada uno. Finalizada la transmisión de los registros Coil y Holding el maestro solicita al esclavo de 9 bytes, el primero de los bytes corresponde a un byte de registros Contact binarios, los 8 bytes restantes corresponden a 4 registros input del dispositivo Modbus, debido al protocolo de transmisión los registros holding e input deben ser concatenados formando el valor numérico correcto, en el caso del dispositivo maestro se debe procesar los registros input, del lado del esclavo se deben procesar los registros holding. Los valores de todos los registros Modbus que procesa el maestro son almacenados en variables globales que solo se ven modificadas cuando hay una actualización de información del cliente MQTT para el caso de Coils y registros Holding. Para el caso de Contacts y registros Input se solicita la información en el lazo principal garantizando que se actualice de manera constante los registros en la interfaz de monitoreo siempre y cuando el esclavo mantenga la conexión en el bus I2C.

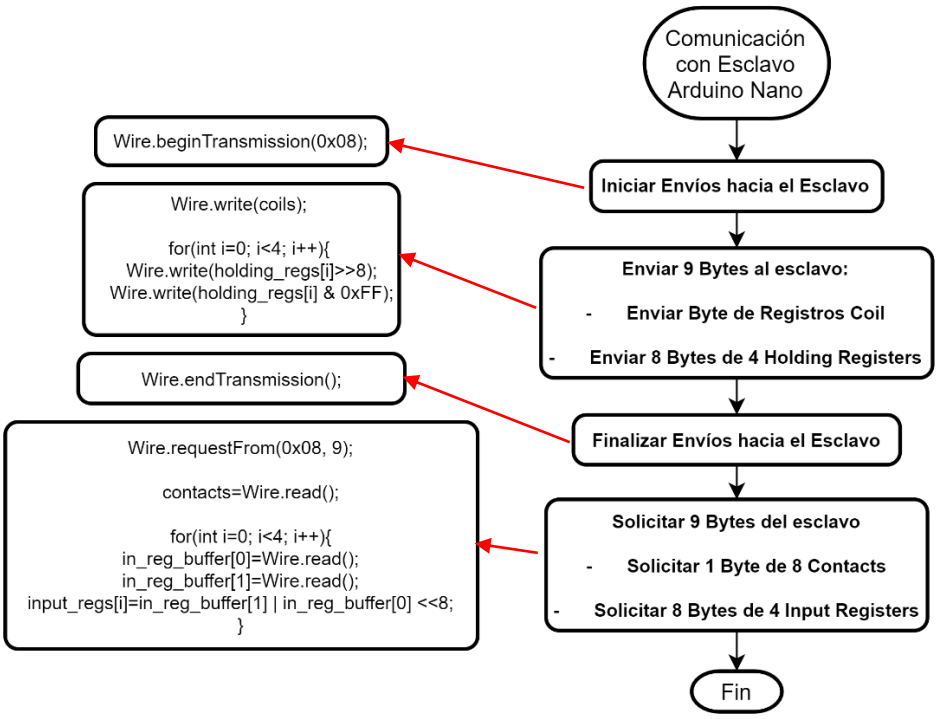


Figura 2.20. Proceso de Comunicaciones con Esclavo Arduino Nano.

Para las comunicaciones con el Sensor MPU6050 se empieza transmitiendo la dirección de los registros del mapa de memoria (tabla 2.9) que se quiere leer, la primera dirección a leerse corresponde al registro 0x3B, esto le indicará al dispositivo que se va a requerir uno o varios bytes desde ese registro, al solicitar información se requiere los valores de los 14

bytes contiguos que forman los valores de las 7 variables que el sensor es capaz de adquirir como lo son: las 3 aceleraciones lineales, la temperatura, y 3 velocidades angulares. Finalmente es necesario procesar los datos de las variables de 16 bits mediante las operaciones establecidas por el fabricante [44] para obtener valores interpretables.

Tabla 2.9. Mapa de memoria de los registros del sensor MPU6050

	Parámetro	Aceleraciones lineales			Temp.	Velocidades angulares		
		Eje X	Eje Y	Eje Z		Eje X	Eje Y	Eje Z
Registros	**MSB 0x..**	3B	3D	3F	41	43	45	47
	LSB 0x..	3C	3E	40	42	44	46	48

El valor de temperatura debe ser calculado utilizando la siguiente expresión:

$$T[ºC] = \frac{TEMP\_OUT}{340} + 36.53 \tag{2.10}$$

De la misma manera para obtener los valores reales de las expresiones de aceleraciones y velocidades angulares se debe utilizar las siguientes expresiones (tomando en consideración la configuración por defecto del sensor):

$$Aceleración = \frac{ACCEL\_OUT}{16384\frac{LSB}{g}} \tag{2.11}$$

$$Velocidad\ angular = \frac{GYRO\_OUT}{131\frac{LSB}{º/s}} \tag{2.12}$$

El diagrama de flujo del proceso se presenta en la figura 2.21.

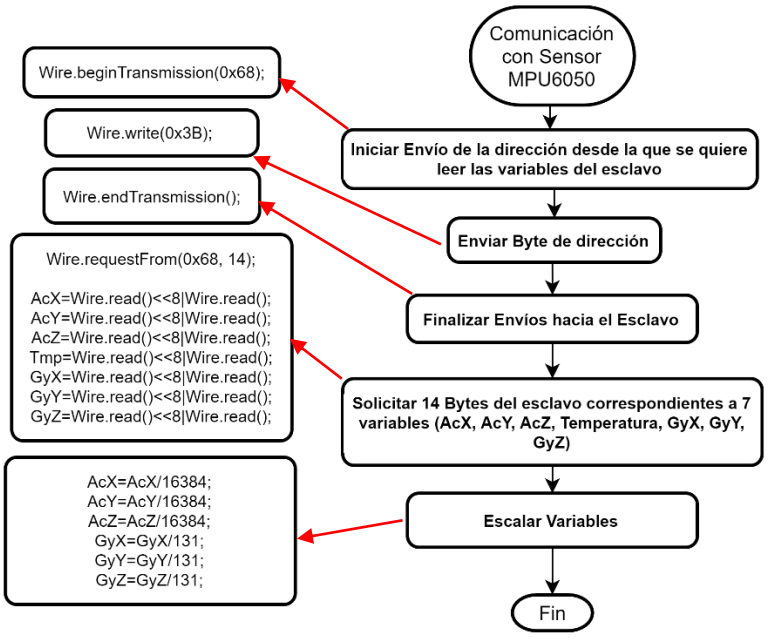


Figura 2.21. Proceso de comunicaciones con el sensor MPU6050

Para el caso del sensor de presión barométrica BMP180 se implementa el algoritmo desarrollado por el fabricante en la hoja de datos [33] (figura 2.22) almacenando en un vector de valores numéricos en punto flotante los valores de presión, temperatura y altura. Los valores de presión y temperatura en crudo se presentan en los registros UP (16 a 19 bits) y UT (16 bits) respectivamente y deben ser utilizados para calcular los valores físicos de las variables. Cabe destacar que para obtener el valor de altura se utilizó la fórmula barométrica internacional,

$$altitude = 44330 * \left(1 - \left(\frac{p}{1013.25}\right)^{\frac{1}{5.255}}\right) \tag{2.13}$$

Donde p_0 corresponde a la presión al nivel del mar 1013.25hpa, p corresponde a la presión medida.

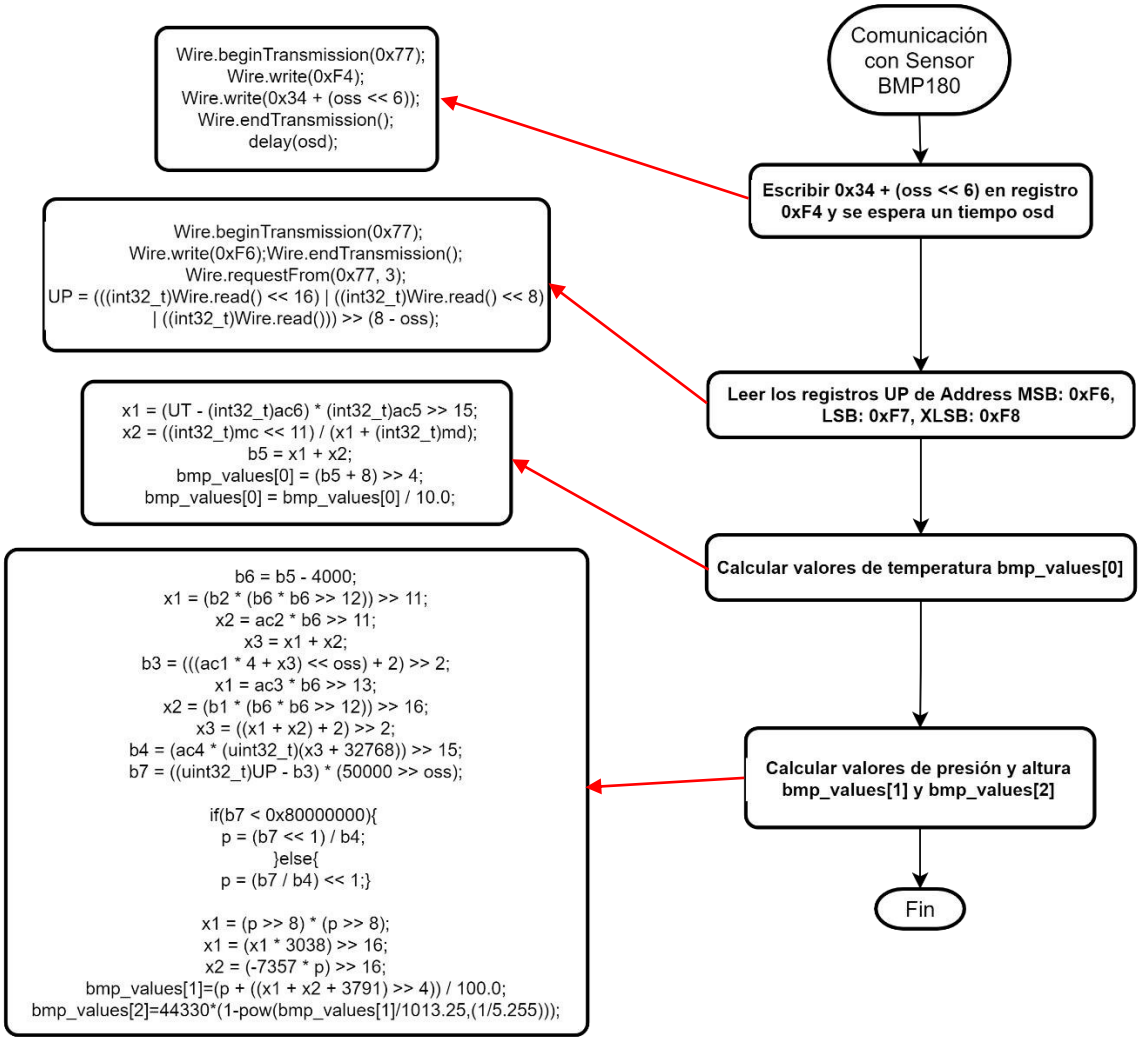


Figura 2.22. Proceso de comunicaciones con el sensor BMP180

Finalmente, para las comunicaciones con el conversor D/A MCP4725 se envía el comando de actualización seguido del setpoint establecido en la variable auxiliar analog_output la cual permite almacenar la información obtenida por el cliente MQTT (Figura 2.23).

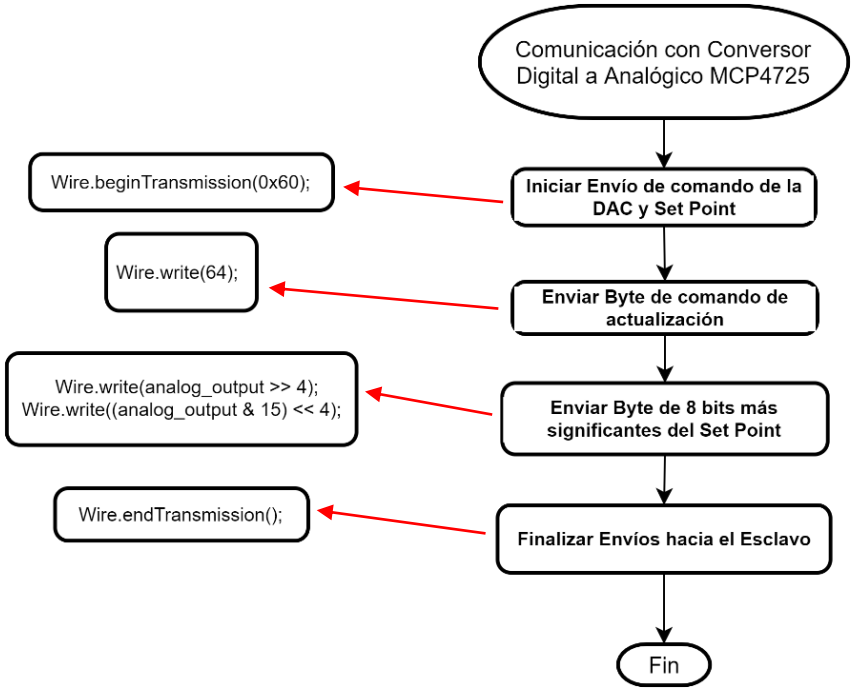


Figura 2.23. Proceso de comunicaciones con conversor digital a analógico MCP4725

2.4.1.5 Callback de actualización de datos ligados a tópicos de subscripción

Como se lo mencionó previamente, este callback (Figura 2.24) es ejecutado cada que un dato asociado a un tópico de subscripción debe ser actualizado, permite procesar la información, para hacerlo se compara el tópico recibido con la lista de tópicos almacenada en la memoria, esto permite interpretar el dato asociado a dicho tópico y procesarlo según corresponda, en este caso se debe contar con múltiples procesamientos para los datos según su tipo, en total para 17 tópicos de subscripción.

Para el caso de los tópicos de salidas digitales, si se recibe un dato asociado con un estado lógico de bajo o alto se pone en 0 ó 1 la salida digital según corresponda. Para el caso del tópico de la salida analógica se almacena directamente el dato en la variable analog_out que permite enviar el setpoint al conversor DA MCP4725.

Para el caso de los Coils el dato asociado permite poner en 1L o 0 L uno de los bits de la variable auxiliar de un byte “coils” que permite actualizar los datos del dispositivo Modbus por medio del esclavo I2C. Finalmente, para los registros holding el dato es almacenado en un vector auxiliar que almacena los valores de los 4 holding registers.

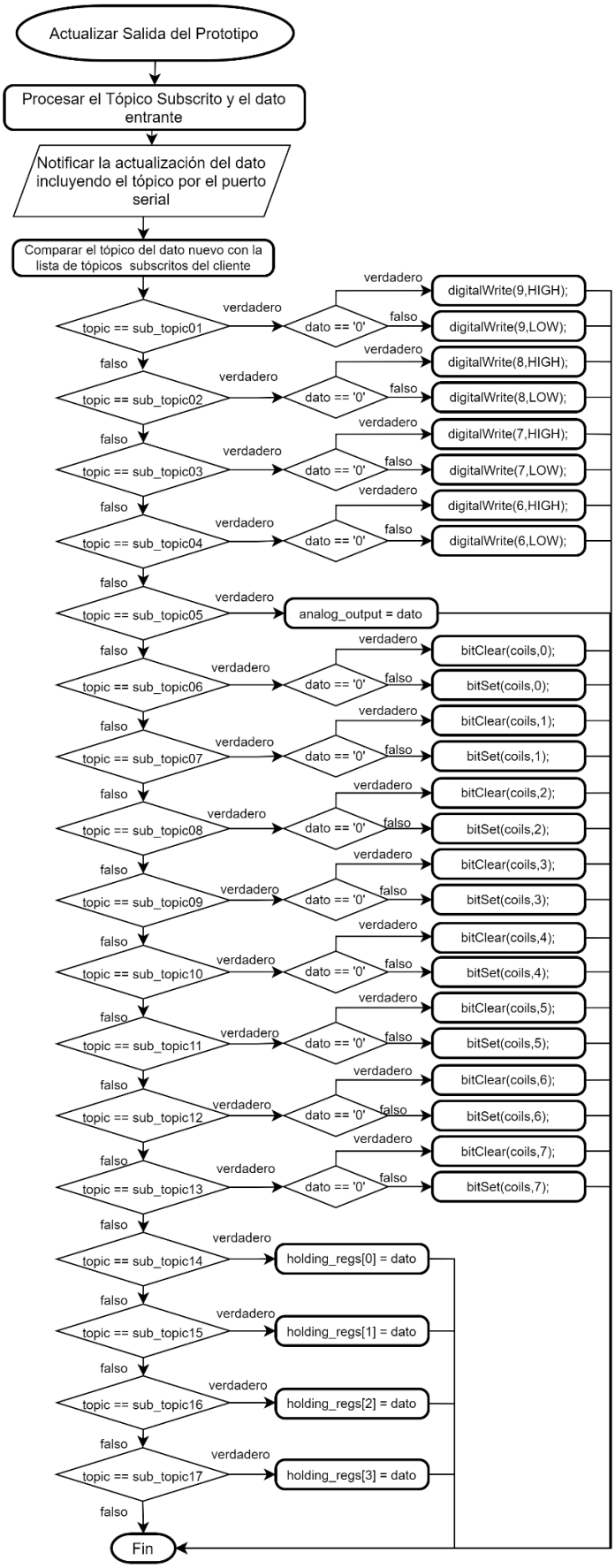


Figura 2.24. Callback de actualización de variable ligada a tópico de subscripción

2.4.2 Módulo Esclavo

El diagrama de flujo presentado en la figura 2.25 esquematiza las funciones del módulo encargado de manejar las comunicaciones con el dispositivo de campo bajo el protocolo Modbus RTU en interfaz RS-485, además es capaz de comunicarse con el módulo principal como esclavo I2C, permitiendo enviar la información del controlador al cliente MQTT y viceversa, para que este pueda publicar o subscribirse a tópicos bajo los cuales se gestiona el intercambio de datos de los registros Modbus Coil (000001-000008), Contact (100001-100008), Input Register (300001-300004) y Holding Register (400001-400004).

Para gestionar las comunicaciones con el dispositivo de campo en el modo de transmisión half-duplex es necesario controlar los pines de control del módulo de conversión RE y DE a diferencia del modo full-duplex en el cual no es necesario controlar los pines, para ello la librería utilizada ModbusMaster.h permite inicializar dos callbacks de pre-transmisión y post-transmisión que permiten cambiar los estados de los pines cada que se requiera, este proceso es esquematizado como una interrupción. De la misma manera la librería Wire.h utilizada para implementar las comunicaciones I2C permite inicializar dos callbacks que permiten la gestión de los eventos de recepción o solicitud de datos, estos callbacks también son esquematizados como interrupciones.

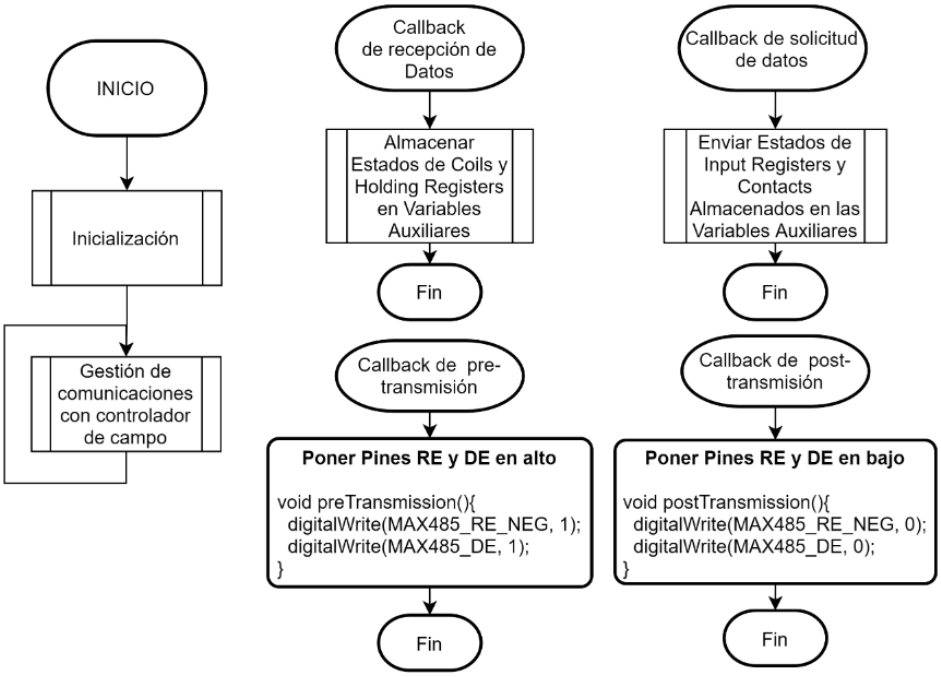


Figura 2.25. Diagrama de flujo general del módulo esclavo Arduino Nano

2.4.2.1 Inicialización

El proceso de inicialización es esquematizado en dos secciones (figura 2.26).

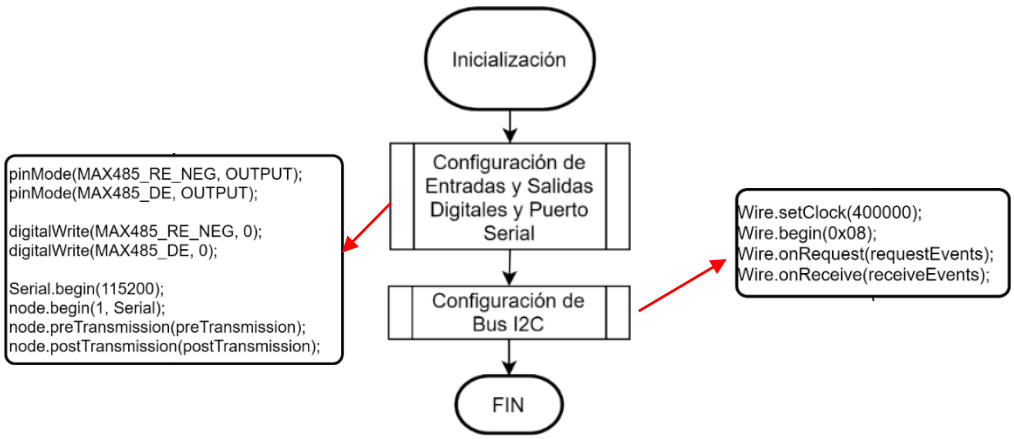


Figura 2.26. Proceso de Inicialización del módulo esclavo.

Para la inicialización el programa permite configurar los pines de control DE y RE como salidas y los coloca en un estado lógico 0, posteriormente se configura las comunicaciones seriales a una velocidad de 115200 baudios utilizada por el dispositivo de campo, luego se inicializa las comunicaciones con el dispositivo de campo especificando su ID y se declara los callbacks de pre-transmisión y de post-transmisión. Finalmente se inicializa las comunicaciones con el módulo maestro I2C a una velocidad de 400 kHz, la dirección del dispositivo corresponde a 0x08.

2.4.2.2 Gestión de comunicaciones con controlador de campo

Para poder transmitir y recibir datos de los dispositivos de campo la librería hace uso de un buffer de transmisión, que permite colocar la información que será enviada al dispositivo, y uno de recepción, que permite obtener la información del dispositivo. Para utilizarlos es necesario especificar la dirección en el buffer del dato que requiere ser procesado.

Para transmitir datos primero se debe colocar el/los datos en el buffer setTransmitBuffer(dirección en buffer (byte), dato (word)) e implementar posteriormente la función que se requiera, ya sea writeMultipleCoils(dirección Modbus, cantidad de coils) o writeMultipleRegisters(dirección Modbus, cantidad de holding registers) que implementan los códigos de función FC=0x0F y FC=0x10 respectivamente. Para leer los datos se debe primero especificar la función requerida, ya sea readDiscreteInputs(dirección Modbus, cantidad de contacts) y readInputRegisters (dirección Modbus, cantidad de input registers) que implementan los códigos FC=0x02 y FC=0x04 respectivamente para luego extraer la información del buffer de recepción mediante la función getResponseBuffer(dirección del dato). Cabe destacar que para enviar y recibir los datos se hace uso de variables auxiliares.

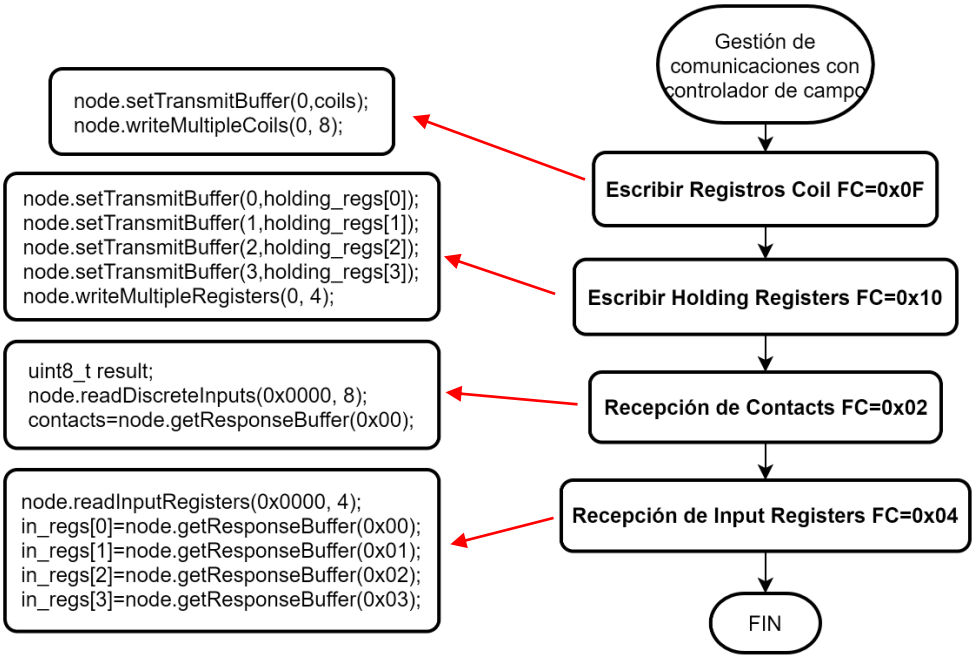


Figura 2.27. Comunicaciones con dispositivo Modbus RTU.

2.4.2.3 Callbacks de comunicaciones I2C

Los callbacks de recepción y solicitud (figura 2.28) permiten gestionar las comunicaciones con el dispositivo maestro I2C, en el caso de un evento de recepción de datos se empieza por almacenar byte a byte la información, para el caso de registros holding es necesario concatenar dos bytes para formar una palabra completa, en total el esclavo almacena 9 bytes, un primer byte de 8 registros Coil seguido de 8 bytes de 4 registros holding. En el caso de un evento de solicitud de datos el esclavo enviará byte a byte los datos solicitados por el maestro que corresponden a 9 bytes, un primer byte de 8 registros contact, seguido de 8 bytes correspondientes a los 4 registros input obtenidos mediante comunicación serial.

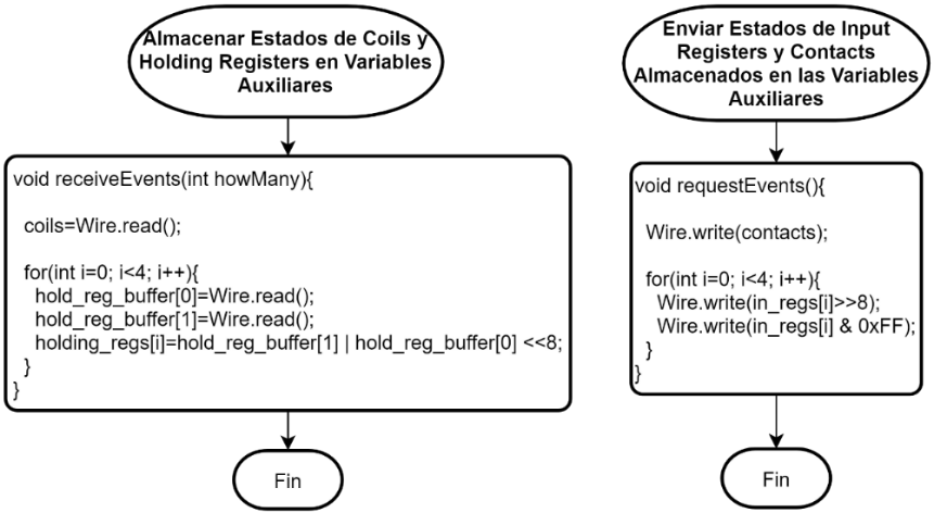


Figura 2.28. Callbacks de recepción y solicitud de datos del esclavo

2.5 Desarrollo en Node-RED

En esta sección se presentará el desarrollo de la interfaz de monitoreo desarrollada en Node-RED. Cabe destacar que los procesos de instalación, configuración y pruebas de las herramientas de software necesarias como lo constituyen el Broker MQTT, Node-RED y el cliente para la implementación de la red privada virtual se presentan en los anexos E-G.

La interfaz de monitoreo presenta cuatro ventanas desplegables mediante pestañas que serán presentadas a continuación: Dispositivos, Alarmas, Tópicos MQTT, Datos en Tiempo Real, como se lo representa la estructura del Dashboard de la figura 2.29.

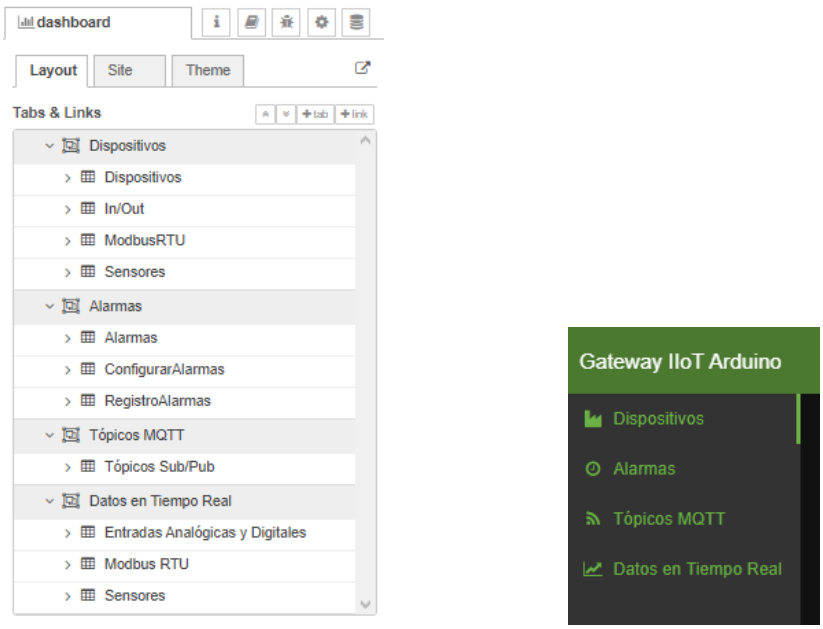


Figura 2.29. Estructura de Ventanas y Pestañas de la Interfaz de monitoreo y control

2.5.1 Ventana de Dispositivos

Permite representar los datos de campo mediante widgets de visualización como luces y medidores, o widgets de interacción como sliders e interruptores. De manera adicional también se incorporó una tabla resumen de todas las variables tomadas en cuenta por cada sección. En esta ventana se clasifican los dispositivos por su tipo, y pueden ser desplegados al dar clic en el botón de la sección respectiva las cuales se presentan a continuación:

Dispositivos: Esta sección se encuentra disponible siempre y cuenta con 4 botones, tres de los botones permiten presentar las distintas secciones una por una, el cuarto botón permite mostrar todas las secciones en la misma ventana.

Para que los botones posibiliten el intercambio entre secciones se hace uso del nodo “ui control” (como se presenta en el flujo de la figura 2.30) que permite definir cual sección se

muestra según el mensaje que se envíe en el campo payload mediante uno de los botones como se puede ver en la figura 2.31 para el botón “Mostrar Entradas y Salidas A/D”.

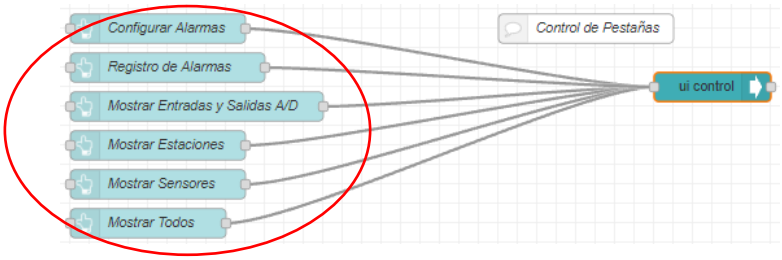


Figura 2.30. Flujo de control de secciones.

En la configuración de los botones es necesario enviar un objeto JSON como mensaje que posibilite que el nodo “ui control” cambie las configuraciones de la ventana, en este caso ocultando las secciones del dispositivo Modbus y de los sensores, “ModbusRTU” y “Sensores” respectivamente, pero que se mantengan las secciones “Dispositivos” e “In/Out” de los 4 botones de control y de entradas y salidas analógicas y digitales respectivamente. Cabe destacar que también se implementaron botones para su uso en la pestaña de alarmas mediante el mismo nodo “ui control” siguiendo el mismo proceso permitiendo intercambiar entre las ventanas “Registro de Alarmas” y “Configurar Alarmas”.

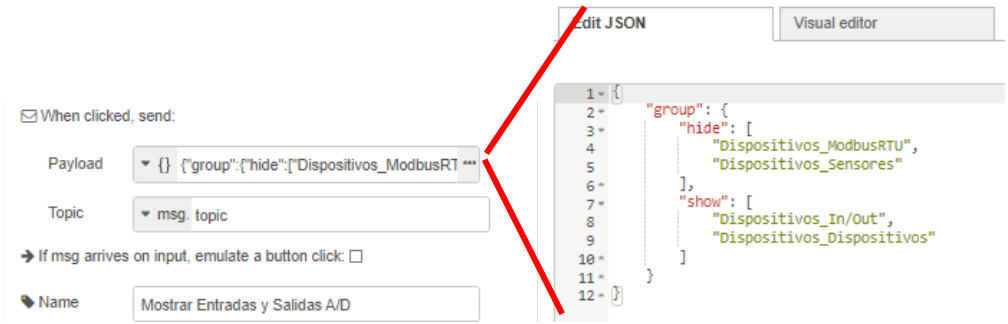


Figura 2.31. Campo payload del botón “Mostrar Entradas y Salidas A/D”.

ENTRADAS Y SALIDAS A/D (In/Out): En esta sección se representa los estados de las dos entradas digitales con luces (1L Verde, 0L Rojo) y los valores de las dos entradas analógicas con dos medidores (0-1024). También permite interactuar con las 4 salidas digitales mediante 4 interruptores, y con la salida analógica mediante un slider. Para su desarrollo se envía y recibe los mensajes de los nodos MQTT que permiten implementar las funcionalidades de publicación y subscripción, finalmente para montar los datos en la interfaz basta con conectar las salidas y entradas de los nodos MQTT directamente a los nodos de monitoreo y control, también se almacena los mensajes de entrada y salida mediante nodos de función para su posterior uso. El flujo desarrollado se presenta en la figura 2.32 junto a las configuraciones de algunos nodos relevantes.

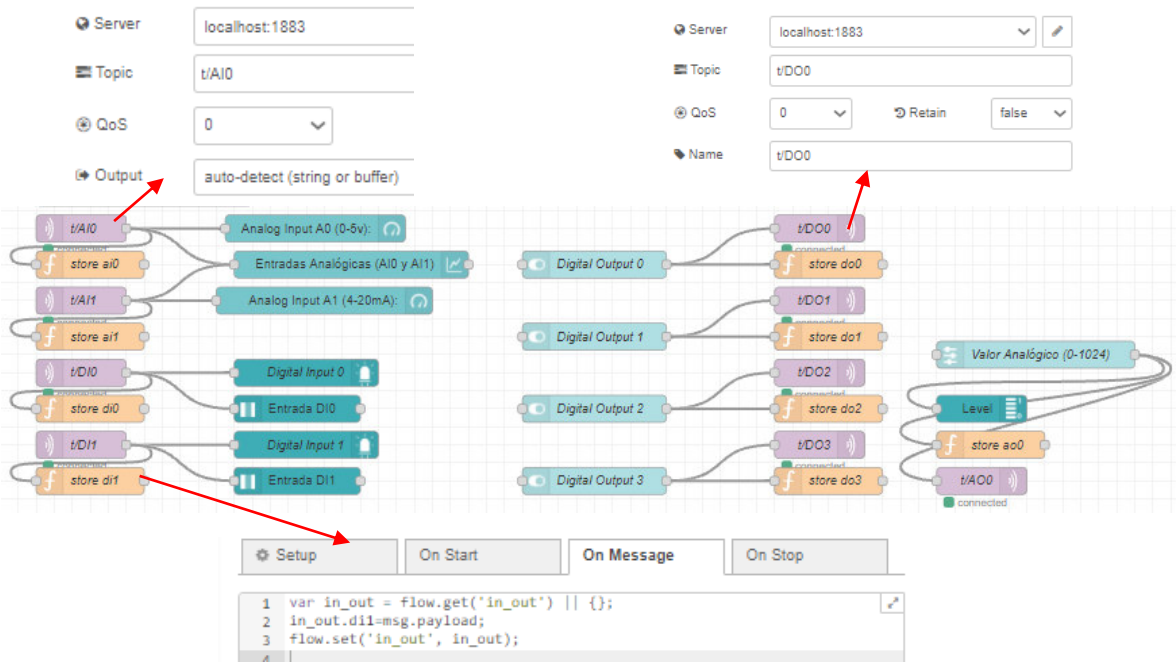


Figura 2.32. Flujo de la sección de “ENTRADAS Y SALIDAS A/D”

Los nodos para tópicos sujetos a datos entrantes como las entradas A/D, datos de sensores y registros Modbus contact e input mantienen configuraciones similares a la presentada en la figura 2.32 desarrollada para la entrada analógica AI0. En el campo Server se debe colocar la IP del Broker con su puerto, en este caso se ejecuta en la misma computadora que Node-RED por lo que la dirección es localhost y el puerto corresponde a 1883. Posteriormente se especifica el tópico para que el Broker pueda filtrar el mensaje, en el caso de las propiedades en la figura 2.32 “t/AI0” que corresponde al mismo tópico bajo el cual el prototipo publica la entrada analógica AI0, finalmente el campo QoS es establecido en 0, ya que la librería solo permite publicar mensajes bajo una calidad de servicio QoS=0 [45] que implica que los dispositivos no garantizan que el mensaje sea entregado.

Los nodos para tópicos sujetos a datos de salida como las salidas A/D y los registros Modbus coil y holding son configurados de manera similar al nodo implementado en la figura 2.32 para el envío del estado de la salida analógica DO0. De la misma manera se configura la dirección del Broker con su puerto, y el tópico “t/DO0” necesario para que el Broker lo filtre y lo entregue al prototipo subscrito al tópico como cliente MQTT. La calidad de servicio es mantenida en 0 a pesar de que para tópicos de subscripción la librería soporta hasta QoS=1 [45], también se mantiene deshabilitada la retención de los mensajes puesto a que no se requiere retener mensajes para nuevos tópicos no contemplados.

Los nodos de almacenamiento corresponden a nodos de función que hacen uso de las funcionalidades de “contexto” que posibilitan el almacenamiento de información para su

uso en el flujo. La programación del nodo de función es similar en todos los casos, el código se presenta en la figura 2.32 para la entrada digital DI1. En este caso se almacena y actualiza el mensaje de la entrada digital DI1 cada que el prototipo lo publica, para ello toma el mensaje del flujo “flow.in_out” (utilizado para recopilar, clasificar y usar los datos de las entradas y salidas A/D almacenadas) y los guarda en una variable auxiliar (del mismo nombre “in_out”), luego se almacena el mensaje msg.payload en una propiedad (.di1) de la variable auxiliar, para finalmente almacenar el dato nuevo de la variable auxiliar en el flujo “flow.in_out” posibilitando la actualización del dato almacenado para su uso.

En cuanto a los widgets de visualización e interacción, como se lo presentó en el flujo de Node-RED se utilizó distintos tipos de nodos: medidores, luces indicadoras, nivel, interruptores, y un slider. Para su uso simplemente se debe conectar al nodo dependiendo si es de entrada o salida y modificar los parámetros de configuración (grupo, tamaño, tipo, etiqueta, formato, unidades, rango, entre otros).

Finalmente, la tabla resumen de todas las variables es generada mediante el nodo “table” como se puede ver en el flujo presentado en la figura 2.33 junto a las configuraciones de los nodos de función y de tabla.

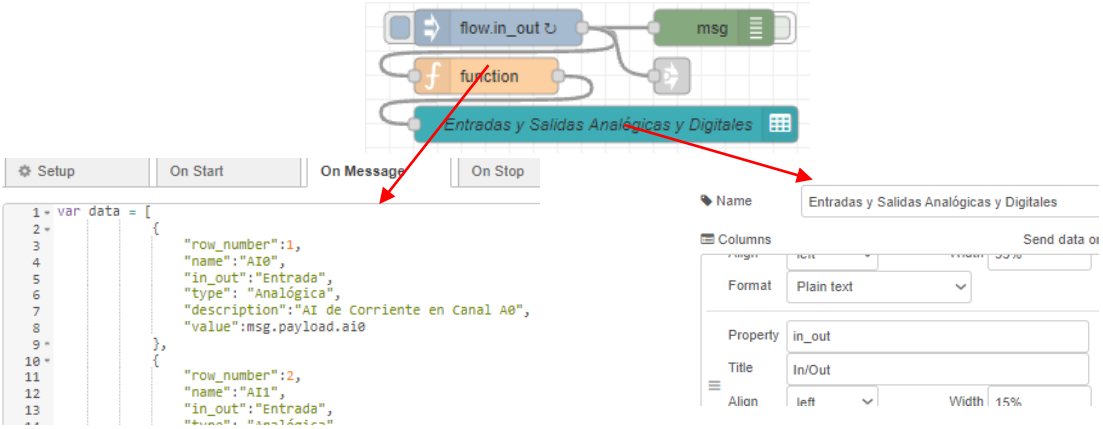


Figura 2.33. Flujo de generación de tabla resumen de variables Entradas y Salidas A/D

El nodo inject (azul) permite ingresar las variables de entradas/salidas A/D que fueron almacenadas en una tabla. Para clasificar las variables e incorporarlas adicionalmente a información complementaria en la tabla, se hace uso de un nodo de función que permite asignar a cada columna creada del nodo de tabla (row_number [#], name [Nombre], description [Descripción], in_out [In/Out], value [Valor]) un dato.

Como se puede notar en el código del nodo de función el dato es una propiedad del mensaje entrante designada mediante un punto (Ejemplo msg.payload.ai0 permite extraer el valor almacenado de la variable analógica AI1).

MODBUS RTU (ModbusRTU): En esta sección representa los estados de los 8 registros contact mediante luces indicadoras (1L Verde, 0L Rojo), los valores de 4 registros input mediante medidores (0-32768), y se puede interactuar con 8 registros coil mediante interruptores y 4 registros holding mediante sliders. De la misma manera que en la sección anterior se envía y recibe los mensajes mediante los nodos MQTT que permiten implementar las funcionalidades de publicación y subscripción cuya configuración y uso ya han sido detallados previamente. De igual manera los datos son almacenados mediante el uso de la funcionalidad de contexto para su posterior uso. El flujo desarrollado para la implementación de esta sección se presenta en la figura 2.34.

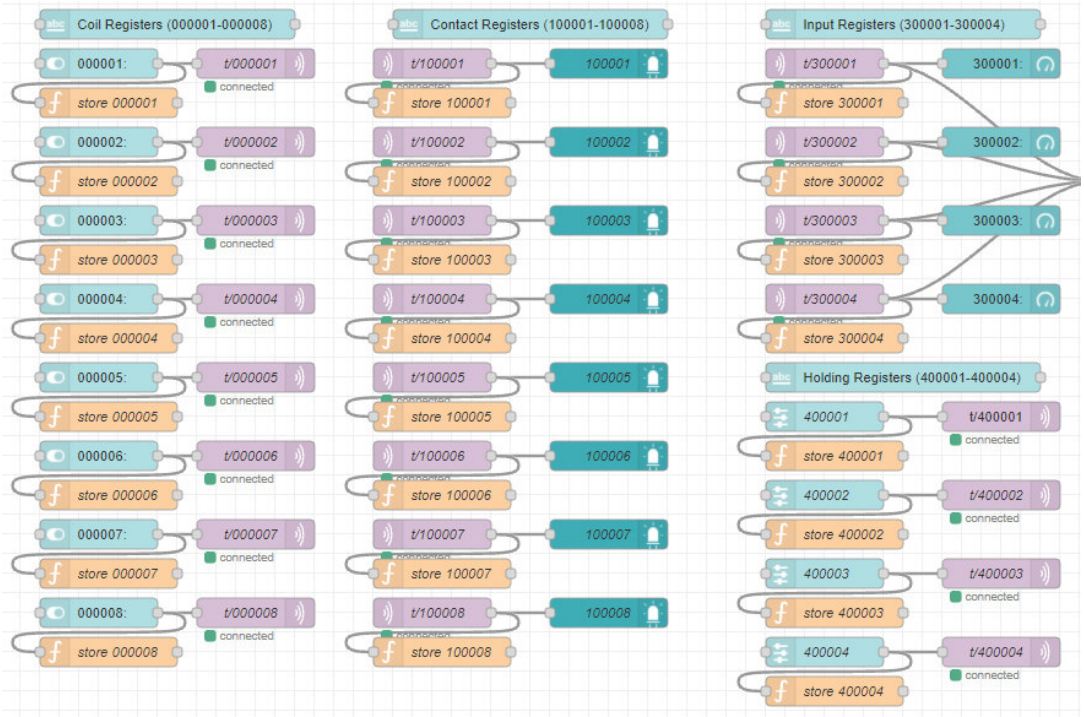


Figura 2.34. Flujo de la sección de “MODBUS RTU”

De manera similar que en el caso anterior se generó una tabla resumen con los registros tomando en consideración las columnas: row_number [#], station [Estación], address [Dirección], type [tipo], value [Valor]. El flujo se presenta a continuación,

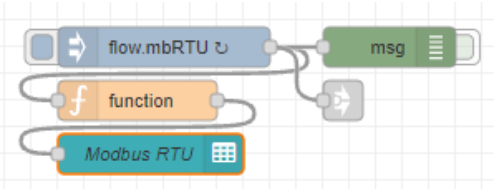


Figura 2.35. Flujo de generación de tabla resumen de registros Modbus RTU.

SENSORES (Sensores): En esta sección se representa los valores de presión, temperatura y altura obtenidos por el sensor BMP180 en tres medidores personalizados, y

los valores de aceleraciones lineales y velocidades angulares obtenidos por el sensor MPU6050 mediante texto no editable. De la misma manera que en las secciones anteriores se recibe mensajes de los nodos MQTT que permiten implementar las funcionalidades de subscripción. Los datos son almacenados mediante el uso de la funcionalidad de contexto para su posterior uso. El flujo implementado se presenta en la figura 2.36.

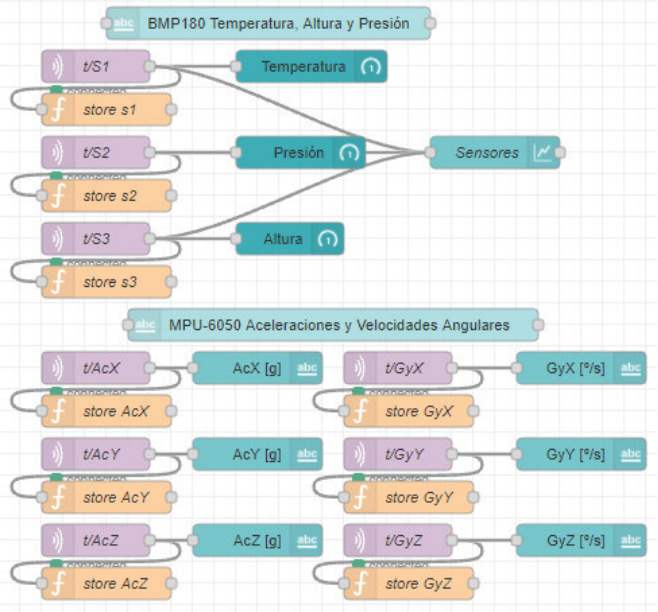


Figura 2.36. Flujo de la sección de “SENSORES”

Los datos son resumidos en una tabla utilizando el mismo método que en los casos anteriores, el flujo es el siguiente:

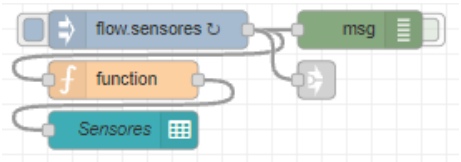


Figura 2.37. Flujo de generación de tabla resumen de valores de sensores.

Las columnas tomadas en cuenta para la tabla son las siguientes: row_number [#], name [Nombre], description [Descripción], type [Tipo de Variable], value [Valor]

2.5.2 Ventana de Alarmas

La finalidad de esta ventana corresponde a generar alarmas configurables que puedan ser registradas visualmente y notificadas mediante un correo electrónico. Las variables que han sido consideradas para la aplicación son de dos tipos, digitales y numéricas.

En el caso de las variables digitales se incorpora una alarma que notifique acerca del cambio de un estado, en el caso de las variables numéricas se considera el valor de la variable fuera de los límites establecidos para que el sistema notifique al usuario.

Las variables de prueba que se han considerado son: entradas analógicas de voltaje y de corriente (AI0 y AI1), valores de las variables presión, temperatura y altura del sensor BMP180 (S1, S2, S3), los registros del dispositivo Modbus 30001 y 10001 y los estados de las entradas digitales (DI0 y DI1)

En esta ventana se presenta 3 secciones, las cuales serán descritas a continuación:

Alarmas: Esta sección se encuentra disponible en todo momento y cuenta con 2 botones que permiten desplegar las secciones “Configurar Alarmas” y “Registro de alarmas”.

Como se lo mencionó previamente en el apartado de “Ventana de Dispositivos”, para poder intercambiar entre las dos secciones los nodos de botones envían objetos JSON como mensajes hacia al nodo “ui control”

Configuración de Alarmas: En esta sección se configuran las alarmas de las variables digitales y numéricas que se consideraron previamente.

Para el caso de las variables digitales se presenta las siguientes configuraciones:

- Habilitación de alarma “Alarma X”: Permite modificar la habilitación de la alarma
- Registro de alarma “Log Enable”: Permite habilitar el registro de alarma en una tabla resumen presentada en la sección de “Registro de alarmas” mediante un switch.
- Habilitación de E-mail “Email Enable”: Permite habilitar el envío de un correo electrónico automatizado dada la incidencia de una alarma mediante un switch.

Para las variables numéricas se presentan dos sliders adicionales a las configuraciones para variables digitales que permiten configurar los límites superior (Lim. Sup.) e inferior (Lim. Inf.) de la variable numérica por fuera de los cuales se generará una alarma.

Para almacenar las configuraciones de las alarmas de cada variable se hace uso de las funcionalidades de contexto mediante nodos de función (al igual que en los casos anteriores) con el fin de poder representarlas en una tabla resumen de configuraciones en la sección de “Registro de alarmas”. El flujo implementado para clasificar y almacenar las variables se presenta en la figura 2.38.

Los tres o cinco nodos de función conectados a los widgets de interruptores y sliders permiten almacenar las configuraciones en un sub-flujo exclusivo para una sola variable.

El nodo inject (azul) permite almacenar todas las configuraciones de la alarma de una sola variable en un solo flujo de configuraciones generales mediante un nodo de función adicional.

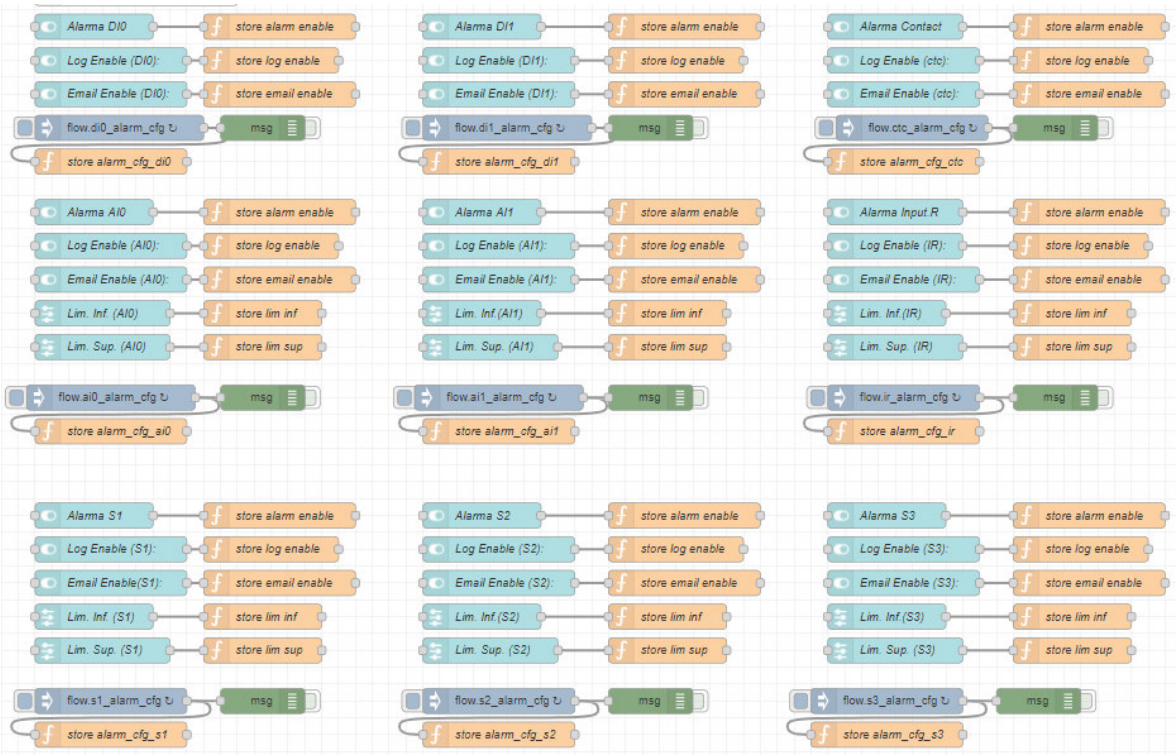


Figura 2.38. Almacenamiento de las configuraciones de las alarmas.

Para la generación de correo electrónico en el caso de las variables digitales se toma el estado de la variable obtenida por el prototipo, y se hace uso del nodo de reporte por excepción (RBE) que permite el flujo de datos si y solo si existe un cambio de estado, posteriormente se hace uso del nodo switch que permite inyectar un mensaje al nodo de función usado para llenar los campos del correo electrónico solo si el estado de la variable es 1. En la figura 2.39 se presenta el flujo para la entrada digital DI0 junto a las configuraciones relevantes de cada nodo.

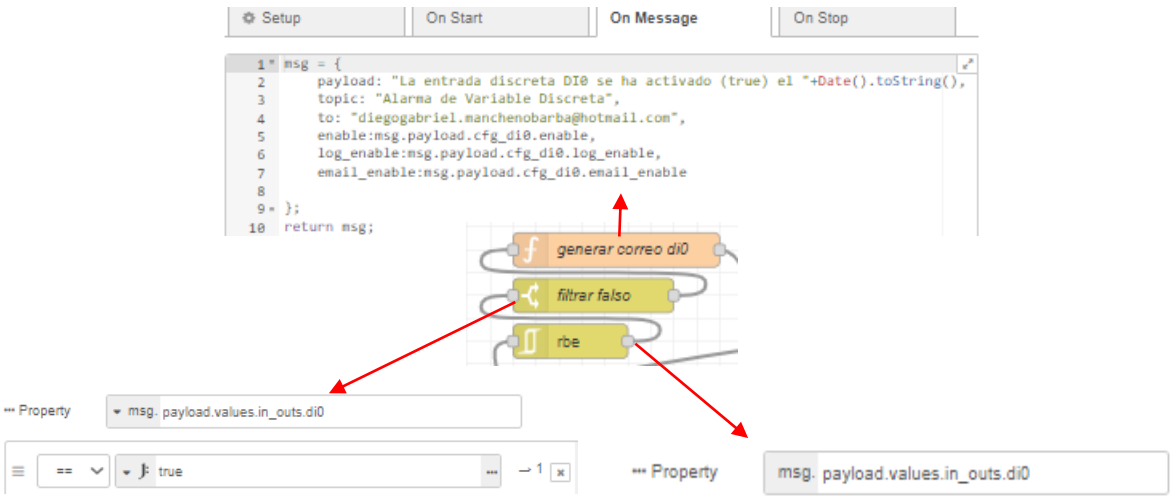


Figura 2.39. Generación de correo electrónico ante cambio de estado de DI0

En el caso de las variables numéricas se hace uso del nodo switch para comparar el valor de la variable con sus límites configurados, si se encuentra fuera de los límites el flujo pasa a un nodo de retardo (delay) que permite restringir la tasa de mensajes, esto con el fin de que ante la incidencia de la alarma no se generen correos de manera ininterrumpida sin un intervalo de tiempo entre correos, finalmente el nodo de función permite llenar los campos del correo electrónico. A continuación, se presenta el flujo de generación de correo para la variable del sensor S1 junto a la configuración de cada uno de los nodos en la figura 2.40.

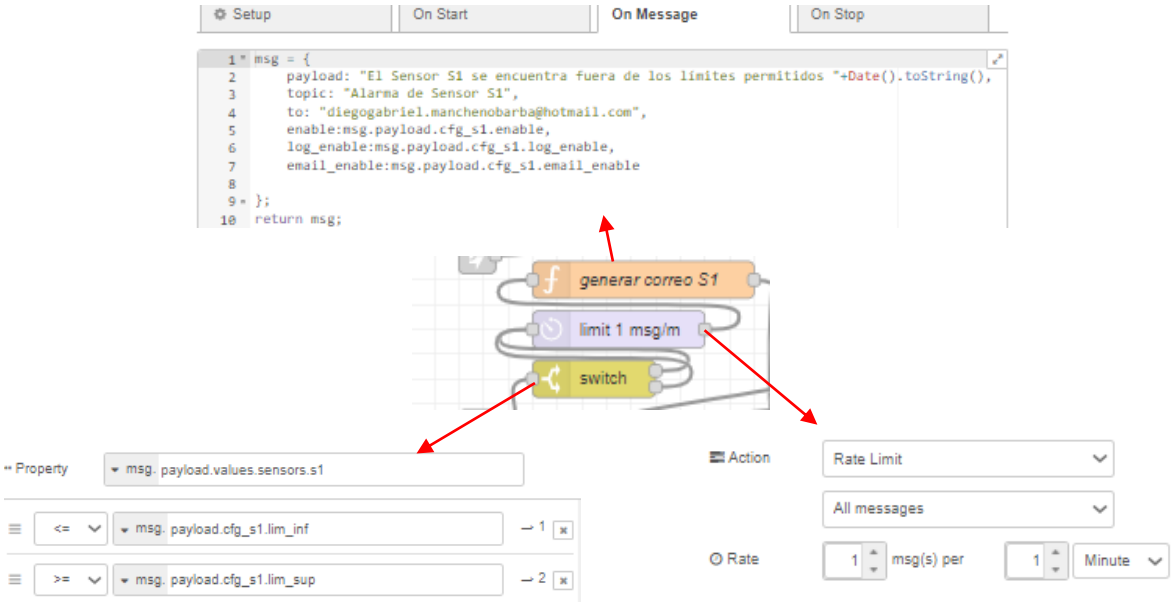


Figura 2.40. Generación de correo electrónico de valor fuera de límites de S1

En la figura 2.41. se presenta todos los nodos utilizados para la generación de correos de todas las variables consideradas.

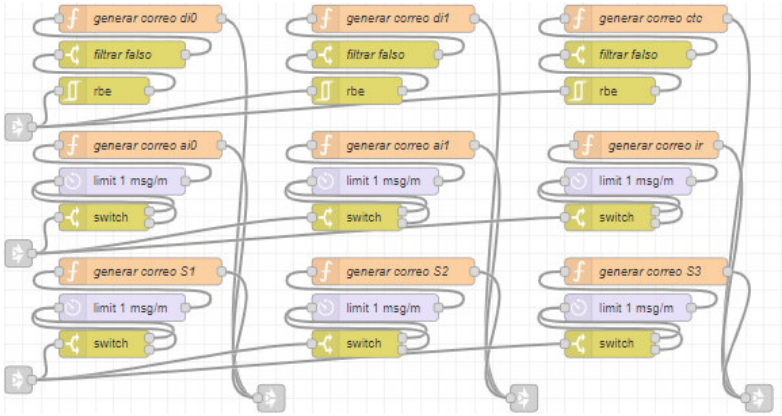


Figura 2.41. Generación de correo de todas las variables.

Finalmente, todos los nodos de generación de correo ingresan al último flujo presentado en la figura 2.42 junto las configuraciones relevantes de cada variable que permiten el envío automatizado del correo, la generación de una notificación pop-up y la generación de una entrada para el flujo de registro de incidencia de alarma de la sección posterior.

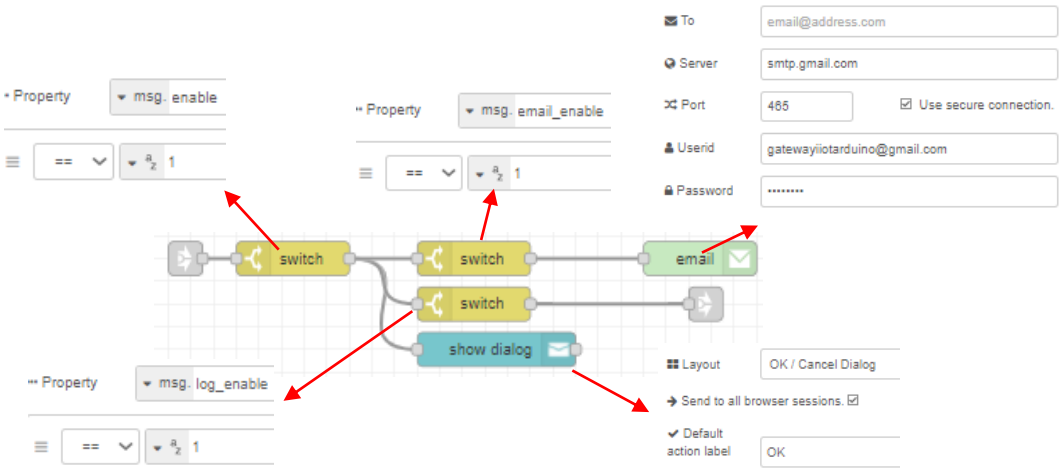


Figura 2.42. Flujo de envío de correo electrónico, registro y de notificación pop-up.

El primer nodo switch verifica si la alarma se encuentra habilitada, si no lo está el mensaje no continua, los siguientes dos switches permiten verificar si las habilitaciones de correo y de registro se encuentran activadas para permitir que el mensaje continue en el flujo, si no lo están solo se genera una notificación pop-up, por último, se encuentra el nodo email que permite hacer uso de una cuenta de correo electrónico para generar la notificación.

Registro de Alarmas: Como se lo mencionó previamente en esta sección se presenta una tabla resumen de las configuraciones de alarmas realizadas en la sección previa, también presenta una tabla generada mediante código HTML que presenta un registro de la incidencia de las últimas 20 alarmas. Para la tabla de configuraciones se almacena los valores de las variables en el flujo de configuraciones mediante el nodo de función “store values” de manera similar a las anteriores tablas, el flujo de configuraciones es inyectado a un nodo de función que permite llenar los campos de la tabla cuyas columnas consideradas son: row_number [#], name [Nombre], description [Descripción], enable [Habilitación de alarma], enable_reg [Habilitación de Registro, enable_mail [Habilitación de correo], lim_inf [Límite inferior], lim_sup [Límite Superior], y value [Valor].

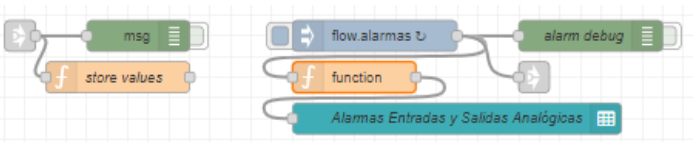


Figura 2.43. Flujo de generación de tabla resumen de configuraciones de alarmas.

Como se lo presentó en la figura 2.42 se condiciona la generación del registro de la alarma en el flujo final de envío de correo electrónico, esto con el fin de reutilizar los nodos de condición de incidencia de una alarma, a la salida del switch de habilitación de registro se presenta el siguiente flujo presentado en la figura 2.44.

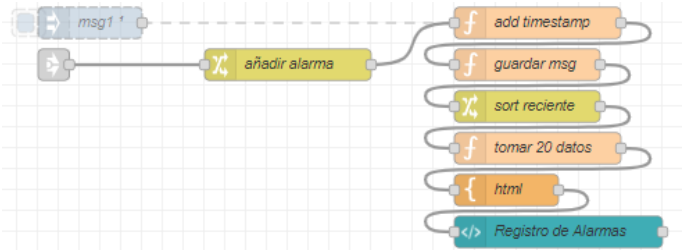


Figura 2.44. Flujo de generación de tabla de registro de alarmas.

El flujo se basa en la contribución de Node-RED de [46] y empieza con el nodo de cambio "añadir alarma" que permite mover el mensaje que se generó para el correo electrónico y el asunto del correo electrónico a dos propiedades distintas, posteriormente el nodo de función "add timestamp" permite añadir una nueva propiedad de fecha y hora para su registro en la tabla, luego el nuevo evento es almacenado en un arreglo del flujo mediante la funcionalidad de contexto en el nodo de función "guardar msg", esto permite almacenar los nuevos eventos de manera continua, posteriormente se ordena los elementos según la propiedad de fecha en el nodo de cambio "sort reciente" para extraer los primeros 20 datos mediante el nodo de función "tomar 20 datos", finalmente se genera una plantilla mediante el nodo "html" que permite generar el código para la generación de una tabla mediante el nodo de plantilla de visualización "Registro de Alarmas".

2.5.3 Ventana de Tópicos MQTT

En esta ventana se implementa una tabla resumen con todas las variables adquiridas mediante el prototipo, en sus columnas se especifica el nombre del tópico, el tipo (Publish/Subscribe), la categoría (Entradas y Salidas A/D, Registro Modbus o Sensores), el nombre de la variable y su valor. La estructura de esta ventana es la más sencilla ya que solo presenta una sola sección. Para implementar la tabla fue conveniente almacenar todas las variables en un solo flujo para extraer los datos de una manera más ordenada mediante el nodo inject y el nodo de función que permite añadir información a la tabla como se puede ver en la siguiente figura:

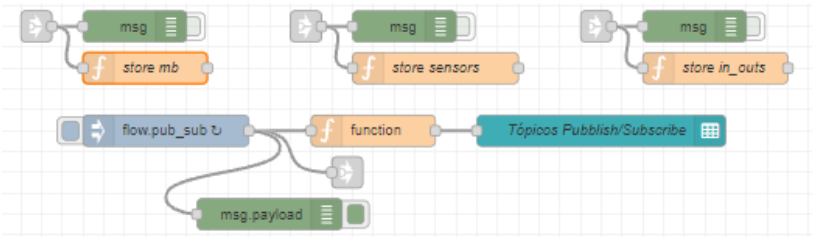


Figura 2.45. Flujo de generación de tabla resumen de tópicos MQTT.

Para acceder a los valores de las variables ahora se debe tomar en cuenta la propiedad de tipo a la que pertenecen las variables ya sea entradas/salidas, registros Modbus o valores

de los sensores (Ejemplo: msg.payload.in_outs.ai0), junto a las propiedades (columnas) de la tabla al igual que en las tablas previas: row_number [#], topic [Tópico MQTT], type [Tipo de Tópico], category [Tipo de variable], register [Nombre de la variable], description [Descripción de la variable], value [Valor de la variable].

2.5.4 Ventana de Datos en Tiempo Real

Esta ventana permite representar la evolución de los datos en función del tiempo, para ello se ha considerado las entradas digitales DI0 y DI1, las entradas analógicas AI0, AI1, los registros Modbus 300001-300004 y los datos de temperatura, presión y altura para intervalos de tiempo definidos.

Para el caso de las entradas digitales se hizo uso del nodo state-trail que permite implementar una línea continua de estados digitales cuyo color cambia según el estado de la entrada digital, su uso en el flujo se presenta en la figura 2.46.

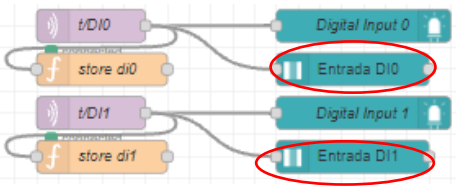


Figura 2.46. Nodo state-trail para DI0.

Para las entradas analógicas AI0 y AI1, las variables físicas adquiridas por el sensor BMP180 y los registros Modbus se hizo uso del nodo chart, representándolas en la misma gráfica para cada caso como se presenta en la figura 2.60. Los valores se representan en un intervalo de 15 segundos para las entradas analógicas y los registros Modbus y en un intervalo de 20 segundos para las variables físicas.

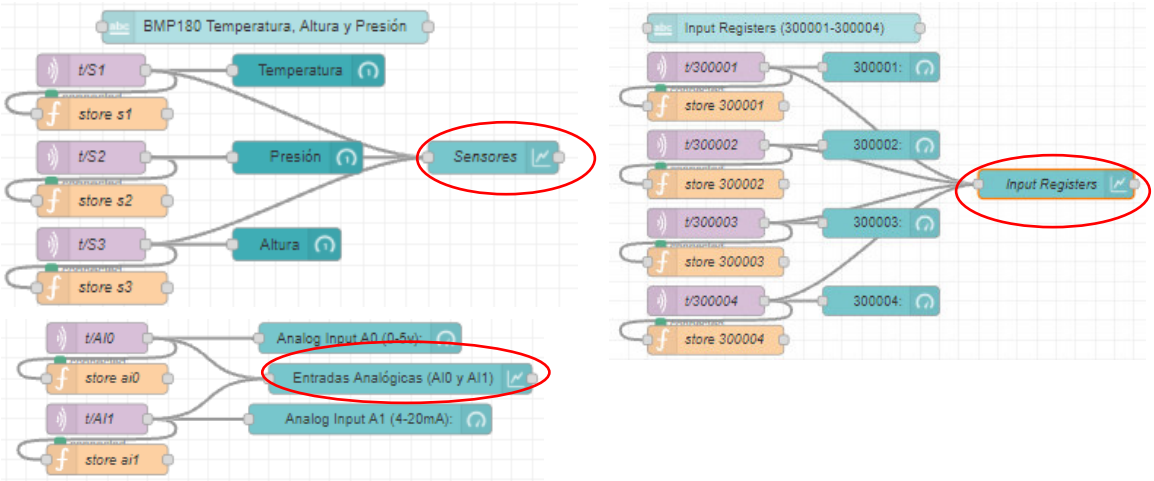


Figura 2.47. Nodos chart.

3. RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

3.1 Resultados

3.1.1 Prototipo de Gateway IoT

En esta sección se procederá a presentar el prototipo final desarrollado con sus partes constitutivas y las capturas de pantalla de cada una de las ventanas de la interfaz de monitoreo y control desarrollada en Node-RED con los datos obtenidos por medio del prototipo con el fin de verificar su funcionamiento bajo los circuitos de pruebas presentados en la sección previa.

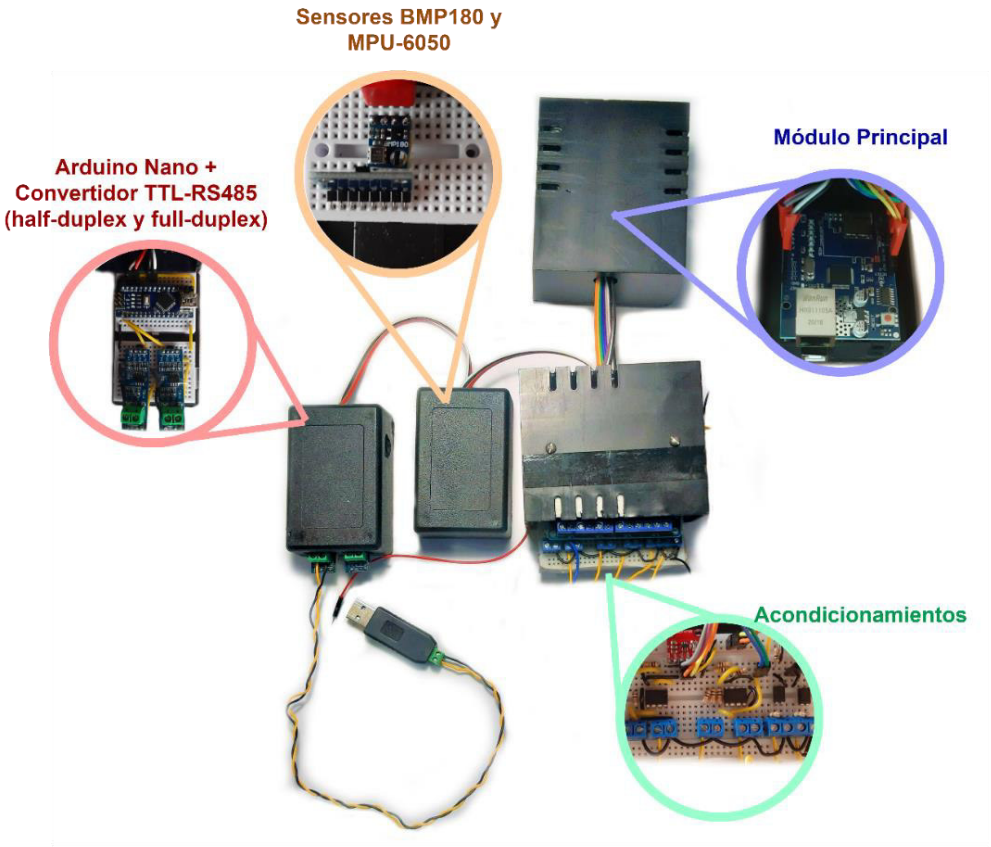


Figura 3.1. Prototipo de gateway IIoT

En la figura 3.1 se presenta el prototipo final del gateway industrial IoT desarrollado. Como se puede observar el módulo principal cuenta con un conector ethernet RJ-45 necesario para conectarse a la red informática en la cual se ha montado el sistema, también se puede observar el circuito de acondicionamiento de las entradas y salidas analógicas y digitales, así como los sensores de baja potencia BMP180 y MPU-6050 y el módulo basado en la tarjeta Arduino Nano encargado de recolectar información del dispositivo Modbus RTU de

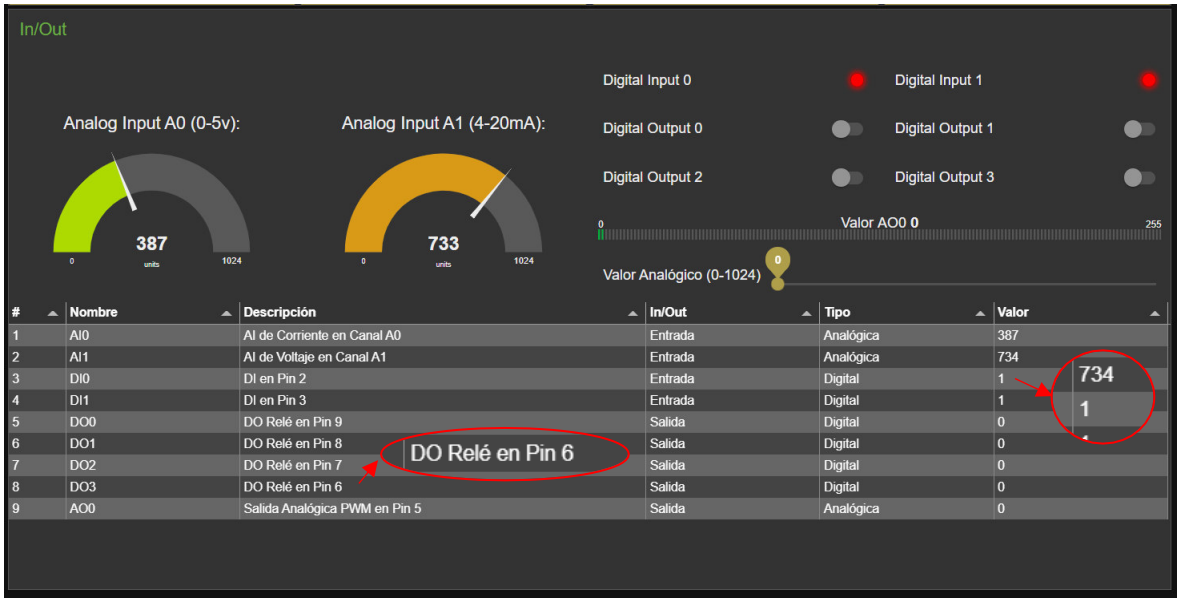
campo bajo interface serial RS-485 en los modos de transmisión half-duplex y full-duplex (configurables mediante el hardware).

Se verificó que los tiempos de actualización de los valores obtenidos mediante el prototipo de gateway IoT como lo son las entradas analógicas y digitales, las variables físicas obtenidas mediante los sensores BMP180 y MPU-6050, así como los registros contact e input del dispositivo Modbus son sumamente cortos y menores a un segundo al igual que los tiempos de activación o actualización de las salidas como lo corresponden a las salidas digitales y analógica y los registros Modbus coil y holding.

3.1.2 Ventana de Dispositivos

En primera instancia se presenta una captura de pantalla en la figura 3.2 de la sección de entradas y salidas digitales y analógicas de la ventana de dispositivos. Se puede verificar que los valores de las entradas de voltaje de 0 a 10 voltios y de corriente de 4 a 20 mA son representados sin inconvenientes en los widgets de medidores en valores de 0 a 1024, de la misma manera que las entradas digitales de 10 voltios en las luces indicadoras. También se puede verificar la activación de los relés mediante los widgets de interruptores y el cambio de voltaje a la salida mediante el voltímetro según el comando numérico ingresado mediante el widget de slider.

En la tabla resumen se presenta todas las variables consideradas con una breve descripción y clasificación que podría ser utilizada para describir su función en una aplicación industrial, además de su valor actual.



#	Nombre	Descripción	In/Out	Tipo	Valor
1	AI0	AI de Corriente en Canal A0	Entrada	Analógica	387
2	AI1	AI de Voltaje en Canal A1	Entrada	Analógica	734
3	DI0	DI en Pin 2	Entrada	Digital	1
4	DI1	DI en Pin 3	Entrada	Digital	1
5	DO0	DO Relé en Pin 9	Salida	Digital	0
6	DO1	DO Relé en Pin 8	Salida	Digital	0
7	DO2	DO Relé en Pin 7	Salida	Digital	0
8	DO3	DO Relé en Pin 6	Salida	Digital	0
9	AO0	Salida Analógica PWM en Pin 5	Salida	Analógica	0

Figura 3.2. Ventana de Dispositivos (Entradas y Salidas Analógicas y Digitales).

En la sección de sensores presentada en la figura 3.3 se verifica la representación de la temperatura ambiente en grados centígrados, la altura en metros sobre el nivel del mar y la presión en milibares obtenidas mediante el sensor BMP180 que corresponden a los valores promedio de temperatura y altura de la ciudad de Machachi en donde se ha puesto a prueba el prototipo, también se puede observar los valores de velocidades angulares y aceleraciones lineales medidas por el sensor acelerómetro y giroscopio MPU6050. También se presenta en una tabla resumen todas las variables con sus valores y sus descripciones al igual que en el caso anterior.

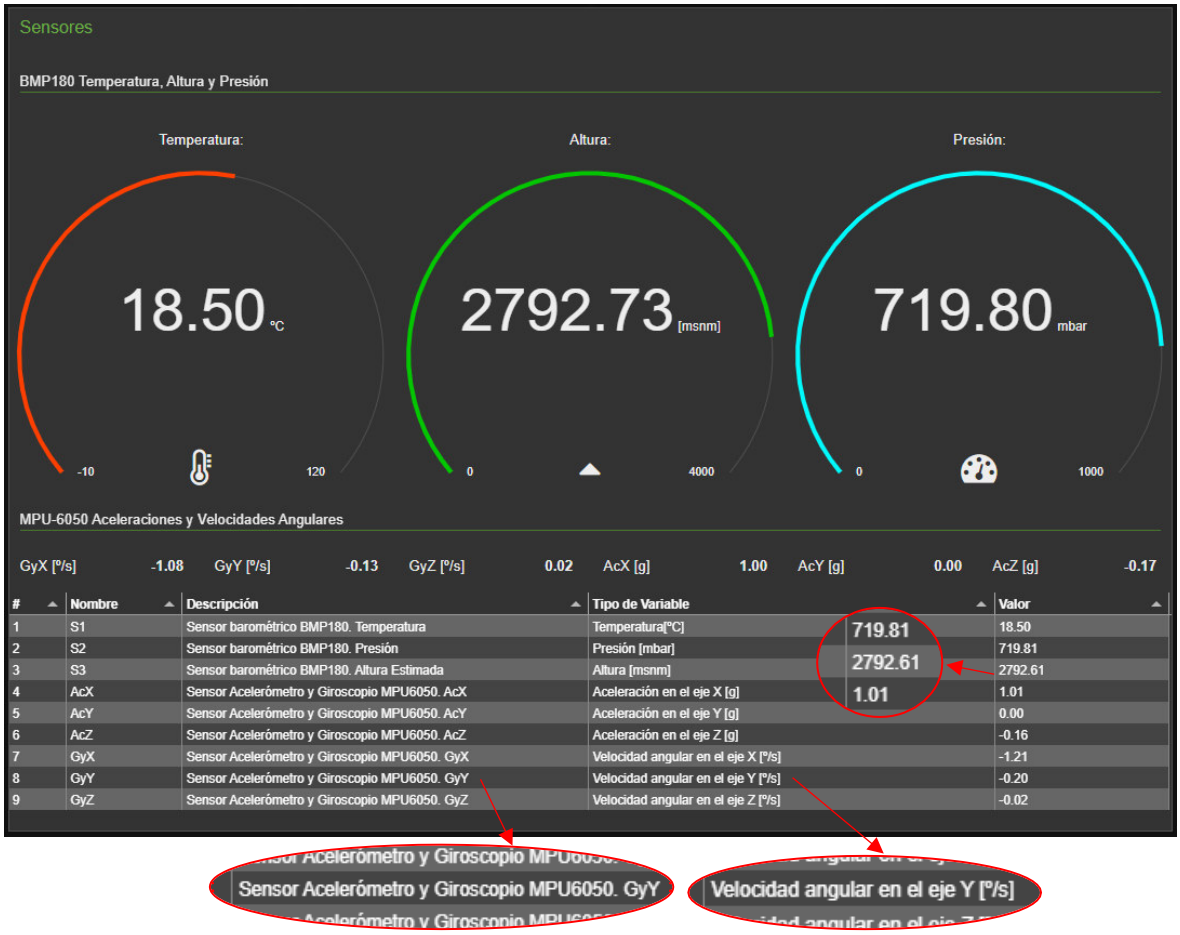


Figura 3.3. Ventana de Dispositivos (Sensores).

En la figura 3.4 se presenta la sección de registros Modbus gestionados por el prototipo con una estación remota simulada.

Como se lo mencionó en secciones previas el prototipo puede funcionar en ambos modos de transmisión (half-dúplex y full-dúplex) bajo la interface de comunicaciones seriales RS-485 con tan solo realizar modificaciones en el hardware del módulo.

Se puede verificar la representación de los valores de los registros contact (10001-10008) e input (30001-30004) en widgets de luces indicadoras y medidores respectivamente,

también se puede verificar el cambio de estado de los registros coil (00001-00008) y variaciones numéricas de los registros holding (40001-40004) mediante interruptores y sliders respectivamente.

De la misma manera se incorporó una tabla resumen de las variables con el fin de proporcionar más información acerca de una potencial aplicación industrial que podría funcionar bajo el comando de un dispositivo modbus de campo como podría ser por ejemplo un controlador lógico programable.

Figura 3.4. Ventana de Dispositivos (Modbus RTU).

Los valores y estados son exactamente iguales tanto en la interfaz de monitoreo y control desarrollada como en la aplicación de simulación que se presenta en la figura 3.5.

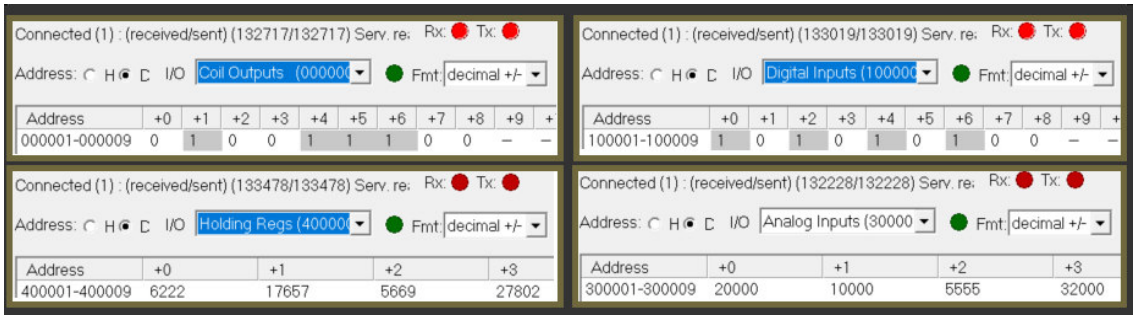


Figura 3.5. Registros de la estación Modbus simulada

3.1.3 Ventana de Tópicos MQTT

Esta pantalla presentada en la figura 3.6 permite verificar en una tabla resumen los valores de todas las variables consideradas para la implementación, en total se presentan 42 variables que el prototipo es capaz de gestionar como cliente MQTT para su comunicación con la interfaz de monitoreo y control desarrollada, permitiendo cumplir con uno de los objetivos propuestos para la implementación.

Tópicos Sub/Pub

#	Tópico	Tipo	Categoría	Variable	Descripción	Valor
1	t/AI0	Publish	Entradas y Salidas A/D	AI0	Entrada Analógica AI0	294
2	t/AI1	Publish	Entradas y Salidas A/D	AI1	Entrada Analógica AI1	286
3	t/DI0	Publish	Entradas y Salidas A/D	DI0	Entrada Digital DI0	1
4	t/DI1	Publish	Entradas y Salidas A/D	DI1	Entrada Digital DI1	1
5	t/DO0	Subscribe	Entradas y Salidas A/D	DO0	Salida Digital DO0	0
6	t/DO1	Subscribe	Entradas y Salidas A/D	DO1	Salida Digital DO1	0
7	t/DO2	Subscribe	Entradas y Salidas A/D	DO2	Salida Digital DO2	0
8	t/DO3	Subscribe	Entradas y Salidas A/D	DO3	Salida DIgital DO3	0
9	t/AO0	Subscribe	Entradas y Salidas A/D	AO0	Salida Analógica AO0	0
10	t/s1	Publish	Sensores	Sensor 1	Temperatura [°C]	18.50
11	t/s2	Publish	Sensores	Sensor 2	Presión [mbar]	719.81
12	t/s3	Publish	Sensores	Sensor 3	Altura [msnm]	2792.61
13	t/000001	Subscribe	Registro Modbus	Coil (000001)	Contacto de Motor 1	0
14	t/000002	Subscribe	Registro Modbus	Coil (000002)	Contacto de Motor 2	1
15	t/000003	Subscribe	Registro Modbus	Coil (000003)	Contacto de Motor 3	0
16	t/000004	Subscribe	Registro Modbus	Coil (000004)	Contacto de Motor 4	0
17	t/000005	Subscribe	Registro Modbus	Coil (000005)	Contacto de Motor 5	1
18	t/000006	Subscribe	Registro Modbus	Coil (000006)	Contacto de Motor 6	1
19	t/000007	Subscribe	Registro Modbus	Coil (000007)	Contacto de Motor 7	1
20	t/000008	Subscribe	Registro Modbus	Coil (000008)	Contacto de Motor 8	0
21	t/100001	Publish	Registro Modbus	Contact (100001)	Contacto de Falla 1	1
22	t/100002	Publish	Registro Modbus	Contact (100002)	Contacto de Falla 2	0
23	t/100003	Publish	Registro Modbus	Contact (100003)	Contacto de Falla 3	1
24	t/100004	Publish	Registro Modbus	Contact (100004)	Contacto de Falla 4	0
25	t/100005	Publish	Registro Modbus	Contact (100005)	Contacto de Falla 5	1
26	t/100006	Publish	Registro Modbus	Contact (100006)	Contacto de Falla 6	0
27	t/100007	Publish	Registro Modbus	Contact (100007)	Contacto de Falla 7	1
28	t/100008	Publish	Registro Modbus	Contact (100008)	Contacto de Falla 8	0
29	t/300001	Publish	Registro Modbus	Input Register (300001)	Velocidad del Motor 1	20000
30	t/300002	Publish	Registro Modbus	Input Register (300002)	Velocidad del Motor 2	10000
31	t/300003	Publish	Registro Modbus	Input Register (300003)	Velocidad del Motor 3	5555
32	t/300004	Publish	Registro Modbus	Input Register (300004)	Velocidad del Motor 4	32000
33	t/400001	Subscribe	Registro Modbus	Holding Register (400001)	SetPoint de Velocidad 1	6222
34	t/400002	Subscribe	Registro Modbus	Holding Register (400002)	SetPoint de Velocidad 2	17657
35	t/400003	Subscribe	Registro Modbus	Holding Register (400003)	SetPoint de Velocidad 3	5669
36	t/400004	Subscribe	Registro Modbus	Holding Register (400004)	SetPoint de Velocidad 4	27802
37	t/AcX	Publish	Sensores	AcX	Aceleración Lineal eje X [g]	1.01
38	t/AcY	Publish	Sensores	AcY	Aceleración Lineal eje Y [g]	0.00
39	t/AcZ	Publish	Sensores	AcZ	Aceleración Lineal eje Z [g]	-0.17
40	t/GyX	Publish	Sensores	GyX	Velocidad Angular eje X [°/s]	-1.12
41	t/GyY	Publish	Sensores	GyY	Velocidad Angular eje Y [°/s]	-0.37
42	t/GyZ	Publish	Sensores	GyZ	Velocidad Angular eje Z [°/s]	-0.10

Figura 3.6. Ventana de Tópicos MQTT.

3.1.4 Ventana de Datos en tiempo Real

En esta ventana de la interfaz de monitoreo y control se presentan datos de variables de entrada al prototipo en gráficos de tendencias en función del tiempo, las gráficas cuentan con una leyenda interactiva que permite la visualización de las variables que el usuario elija, también cabe recalcar que los periodos de visualización de los valores de las variables son de tiempos mayores o iguales a 15 segundos, garantizando que se cumpla los objetivos propuestos para el desarrollo de la interfaz de monitoreo y control presentados en la sección previa.

En la figura 3.7 se presenta una primera sección que permite representar la evolución de las entradas analógicas de voltaje y corriente de valores de 0 a 1024 y los estados de las entradas digitales del prototipo.

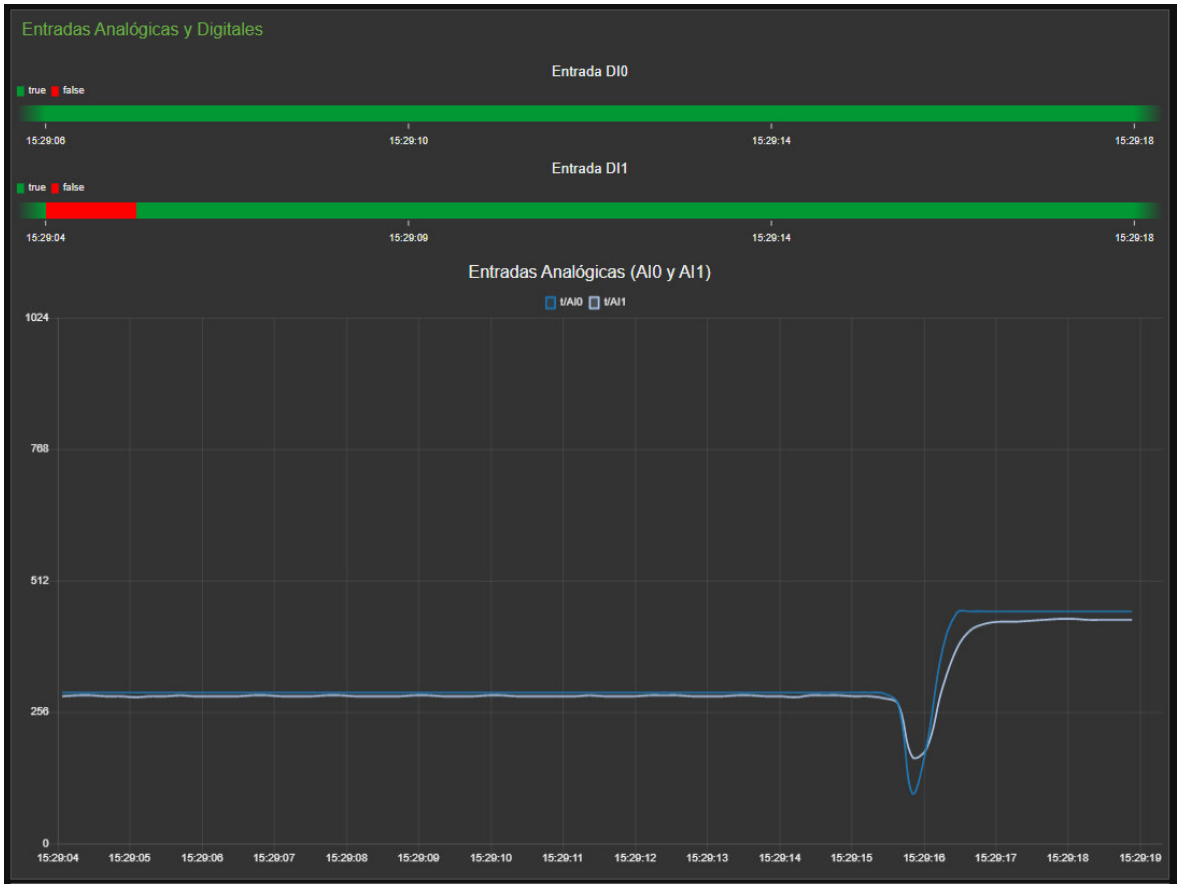


Figura 3.7. Ventana de Datos en Tiempo Real (Entradas A/D).

De igual manera en la figura 3.8 se presenta la sección de registros Modbus en la cual se representa la evolución en tiempo real de los valores numéricos de los registros input de la estación Modbus de campo (30001-30004).

Figura 3.8. Ventana de Datos en Tiempo Real (Input Registers).

Para finalizar en la figura 3.9 se representa la sección de tendencias de las variables físicas de presión, temperatura y altura obtenidas mediante el procesamiento de datos del sensor BMP180. Como se puede observar los valores son prácticamente constantes debido a que las condiciones ambientales como la presión atmosférica son variables físicas que no presentan cambios tan representativos en intervalos tan cortos de tiempo.

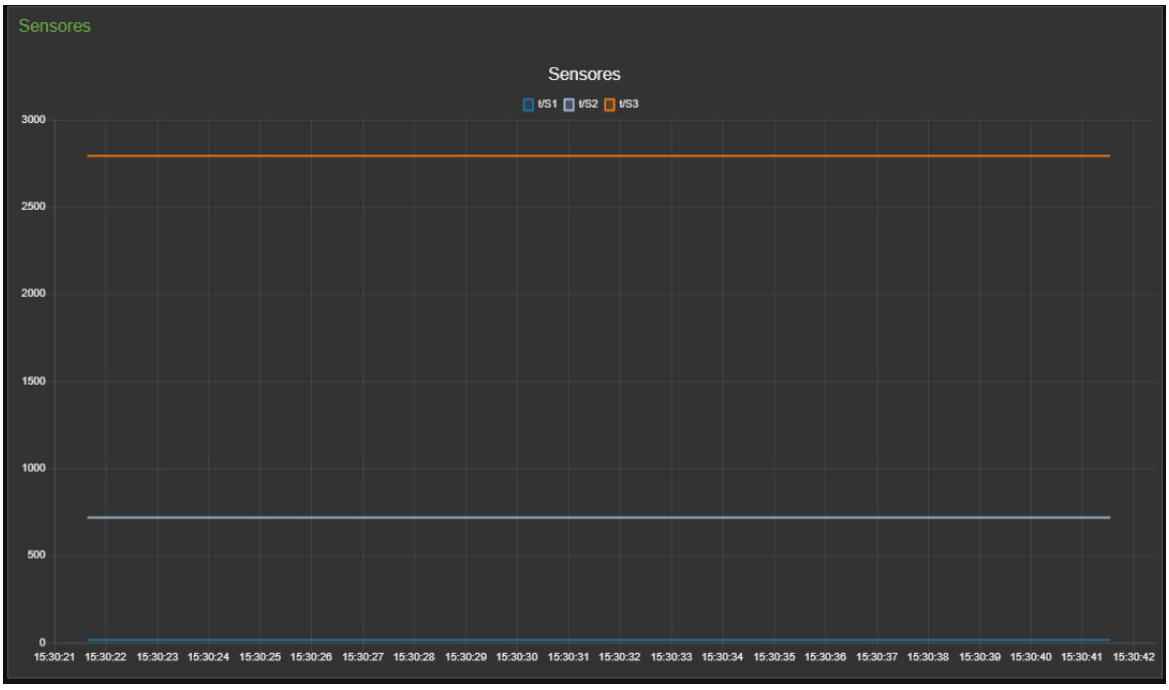


Figura 3.9. Ventana de Datos en Tiempo Real (Sensores).

Se puede verificar también que se puede realizar cambios pequeños en la medición de temperatura al calentar con la piel el sensor como se puede ver en la figura 3.10.

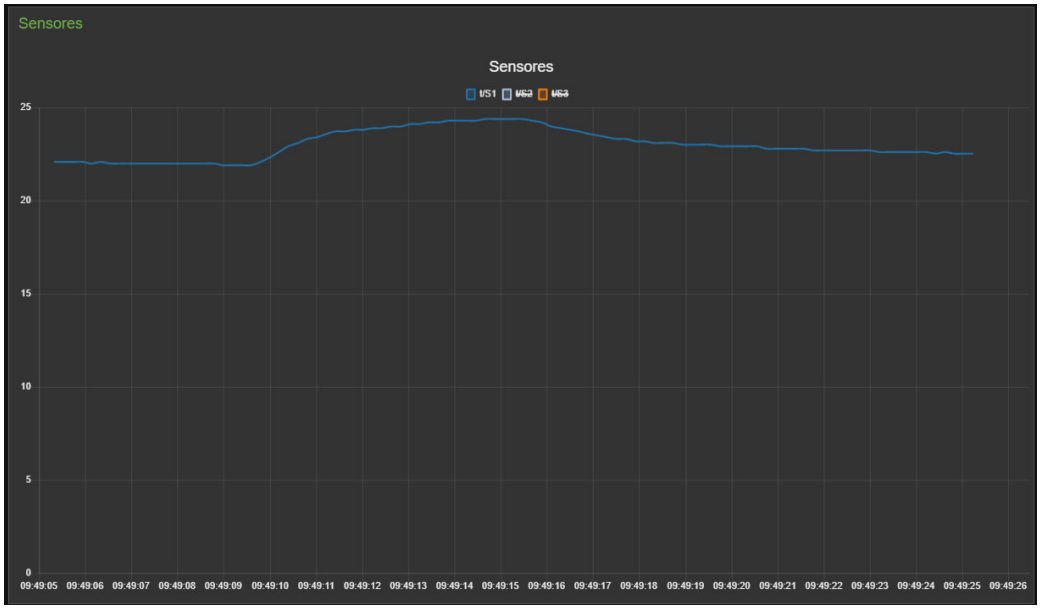


Figura 3.10. Ventana de Datos en Tiempo Real (Sensores/Temperatura).

3.1.5 Ventana de Alarmas

En primera instancia se presenta la sección de configuraciones en la figura 3.11 en la cual se ha habilitado alarmas para la entrada analógica AI0 y la entrada digital DI1. En el caso de ambas variables se ha habilitado su registro y la notificación por medio de correo electrónico, en el caso de la entrada AI0 se configurado los límites inferior de 17 y superior de 971 por fuera de los cuales la alarma será activada.

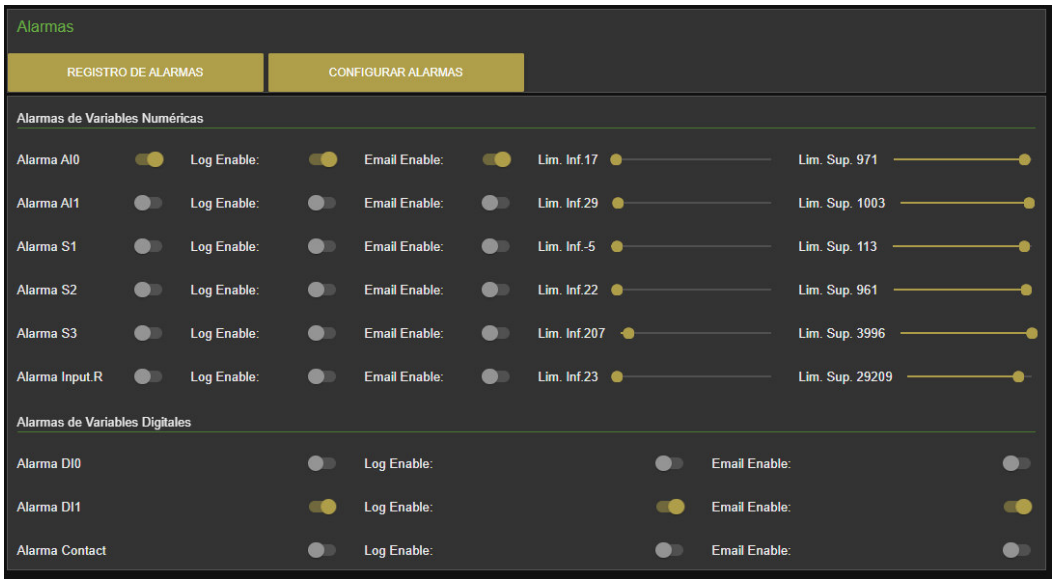


Figura 3.11. Ventana de Alarmas (Configuración de Alarmas).

Dada la incidencia de la alarma configurada para la entrada digital DI1 se presenta en la figura 3.12 el mensaje pop-up generado.

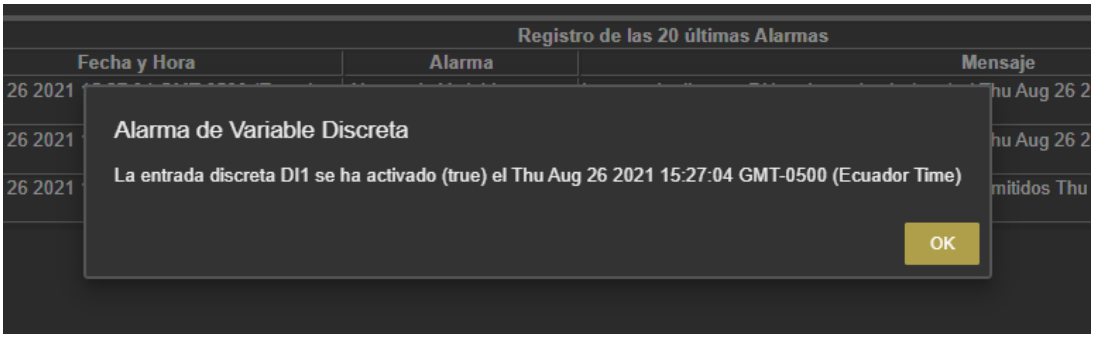


Figura 3.12. Mensaje Pop-Up

En la sección de registro de alarmas presentada en la figura 3.13 se puede notar en primera instancia la tabla resumen de las configuraciones de alarmas, las cuales se encuentran de acuerdo a las configuraciones de la figura 3.11.

Posteriormente se presenta la tabla resumen de eventos de alarmas, en este caso se ha generado tres alarmas, en primera instancia un cambio de la entrada analógica fuera del rango establecido y posteriormente el cambio de estado de 1L a 0L de la entrada digital DI1

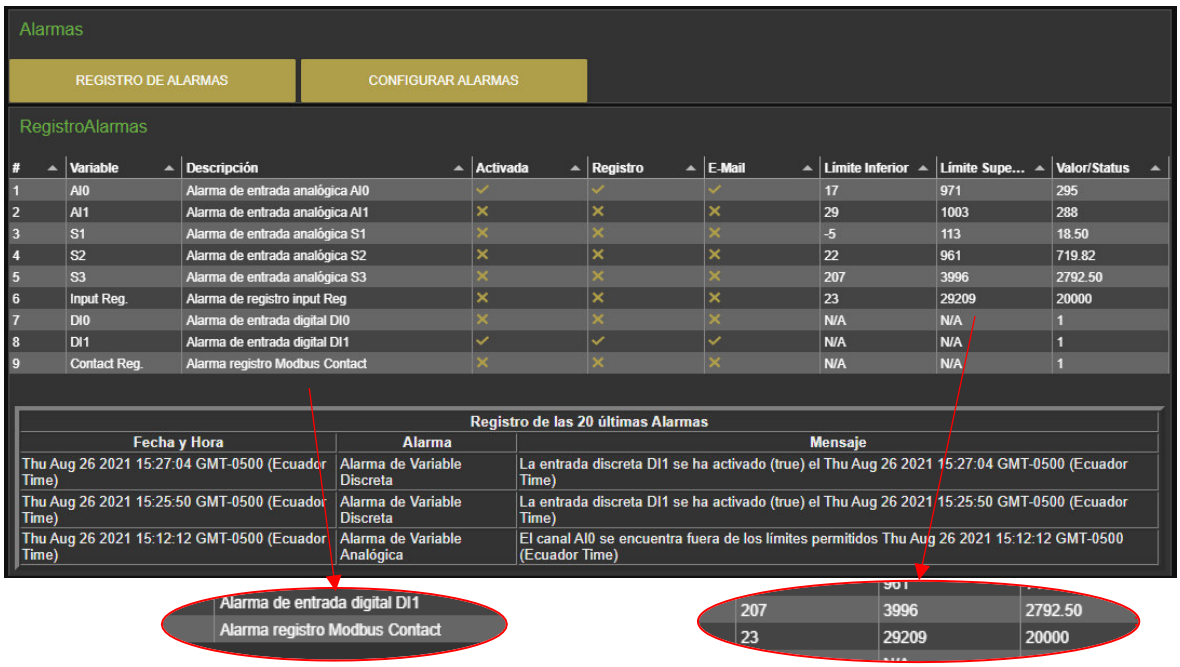


Figura 3.13. Ventana de Alarmas (Registro de Alarmas).

En la figura 3.14 se puede observar la notificación por medio de correo electrónico dado el evento de alarma.

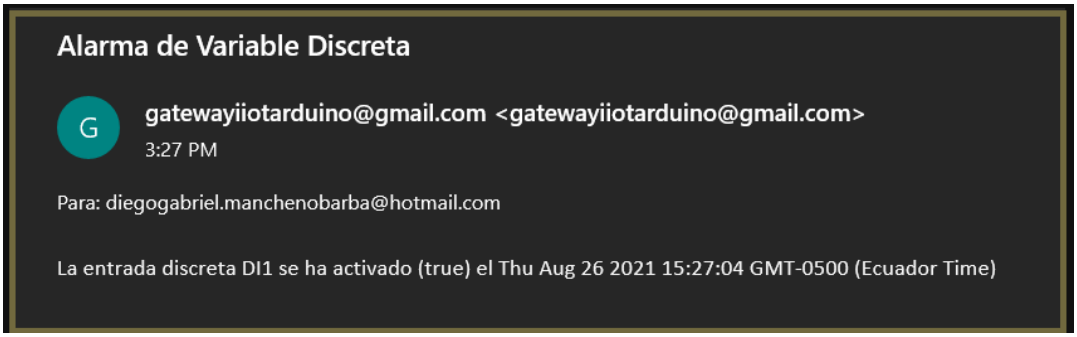


Figura 3.14. Notificación Vía Correo Electrónico.

Como se pudo verificar, la interfaz también incorpora la funcionalidad de un sistema de gestión de alarmas que posibilite las notificaciones de los eventos por medio de un correo electrónico automatizado y su registro en una tabla resumen, garantizando el cumplimiento de los objetivos propuestos para la implementación de la interfaz de monitoreo y control.

3.2 Conclusiones

Conforme a las pruebas realizadas del funcionamiento del prototipo de gateway IIoT y de la interfaz de monitoreo y control desarrollada se ha considerado las siguientes conclusiones y recomendaciones.

- Se ha desarrollado un prototipo de gateway industrial con soporte para internet de las cosas que posibilite la comunicación de información de campo de entradas y salidas analógicas y digitales industriales y de un dispositivo Modbus bajo comunicaciones seriales RS-485 con una interfaz de monitoreo y control que implementa aplicaciones básicas de monitoreo y del internet de las cosas.

- Se analizó las distintas características funcionales de varios gateways comerciales, prototipos en desarrollo y aplicaciones del internet de las cosas con el fin de establecer un conjunto de requerimientos que sirvieron como pauta para seleccionar las tecnologías requeridas y plantear la arquitectura necesaria para la implementación del sistema.

- Se ha propuesto una arquitectura bajo la cual se ha procedido a montar los dispositivos de hardware y los servicios de software que garanticen la comunicación del prototipo en la red, la gestión de los datos, el funcionamiento de la interfaz de monitoreo y control y el acceso remoto a la aplicación.

- Se seleccionó y realizó un breve estudio de los protocolos de comunicaciones adecuados para las comunicaciones tanto en el nivel de campo como en niveles superiores, así como las tecnologías de hardware y software necesarias para el

desarrollo del prototipo, el montaje del sistema y el desarrollo de la interfaz de monitoreo y control.

- Se realizó el diseño de hardware y software del módulo principal del prototipo de gateway IIoT considerando: sus funcionalidades como cliente MQTT Publish/Subscribe, la gestión de datos de campo de variables de entrada y salida digitales y analógicas y la implementación de una arquitectura modular que posibilite la integración de múltiples dispositivos de adquisición de datos de campo mediante el uso de comunicaciones I2C.

- Se realizó el diseño de hardware y software de un módulo esclavo en comunicación constante con el módulo principal mediante el bus I2C que posibilita la gestión de los datos de campo de un dispositivo Modbus RTU simulado bajo interfaz RS-485 en modos de transmisión half-dúplex y full-dúplex (configurados mediante el hardware).

- Se desarrolló la interfaz de monitoreo y control mediante la herramienta de libre acceso de Node-RED considerando dos aplicaciones del internet de las cosas como lo constituyen la visualización de tendencias en tiempo real y la gestión de alarmas de varias variables adquiridas mediante el prototipo.

- Se realizó los procesos de instalación y configuración de los componentes adicionales del sistema como lo constituyen el software de cliente de red privada virtual “Hamachi” utilizado para proporcionar acceso remoto a la interfaz de monitoreo y el broker Mosquitto MQTT utilizado para gestionar las comunicaciones entre la interfaz de monitoreo y control y el cliente MQTT que fue implementado en el prototipo.

- Por medio de las pruebas realizadas en la sección previa se pudo validar el funcionamiento del prototipo y de la interfaz de monitoreo y control desarrollada, el módulo principal constituye en un cliente MQTT que es capaz de manejar múltiples tópicos de publicación y subscripción por medio de los cuales los usuarios pueden interactuar mediante la interfaz de monitoreo con las entradas y salidas de manera bidireccional.

- La cantidad de tópicos MQTT considerados para el desarrollo de este trabajo es bastante considerable y puede ser suficiente para aplicaciones de pequeña escala, sin embargo, la implementación de más tópicos se encuentra limitada por la

memoria del módulo principal y podría involucrar la utilización de una tarjeta de almacenamiento microSD incorporada en el módulo Ethernet.

- El prototipo ha sido desarrollado mediante hardware de bajo costo accesible para las industrias de bajo poder adquisitivo y constituye una alternativa viable que podría incorporar muchas más funcionalidades debido a sus características modulares en un futuro.

3.3 Recomendaciones

- Se recomienda realizar desarrollos de módulos de expansión basados en tarjetas embebidas de bajo costo que permitan expandir la cantidad de entradas y salidas, además de incorporar muchos más protocolos de comunicaciones industriales con distintos dispositivos de campo, protocolos tales como Modbus TCP, Profinet, Profibus, Ethernet/IP entre otros.

- Se recomienda realizar tarjetas de acondicionamiento con circuitería robusta que sea mucho más adecuada en ambientes industriales y que pueda cumplir con el mismo objetivo de adaptar las señales de entrada y salida a los niveles de voltaje del módulo principal de manera aislada.

- Sería importante realizar desarrollos de hardware y software que permitan incorporar las funcionalidades de Broker MQTT con el fin de proporcionar una alternativa dedicada y aislada basada en hardware de bajo costo y software de libre acceso que pueda gestionar las comunicaciones de los mensajes de múltiples clientes.

- Sería importante implementar mejoras en el hardware del sistema como lo constituirían: un bus con mayor facilidad de conexión para módulos de expansión adicionales, así como mejoras constructivas y en el hardware que garanticen su funcionamiento bajo estándares y certificaciones internacionales.

- Se recomienda desarrollar aplicaciones basadas en software libre que hagan uso de la información obtenida mediante el prototipo como desarrollos de software que permitan obtener índices de desempeño, tendencias y aplicaciones de análisis de datos con el fin de posibilitar que las PYMES obtengan muchos más beneficios de la aplicación del internet de las cosas en la industria sin la implicación de una inversión económica considerable.

4. REFERENCIAS BIBLIOGRÁFICAS

[1] A. Awouda, K. Aliev, P. Chiabert, and D. Antonelli, “Practical Implementation of Industry 4.0 Based on Open Access Tools and Technologies,” IFIP Advances in Information and Communication Technology. Springer International Publishing, pp. 94–103, 2019. doi: 10.1007/978-3-030-42250-9_9.

[2] S. Munirathinam, “Industry 4.0: Industrial Internet of Things (IIOT),” Advances in Computers. Elsevier, pp. 129–164, 2020. doi: 10.1016/bs.adcom.2019.10.010.

[3] G. Alessandrini, “Internet Industrial de las Cosas (IIoT)”. Instituto de Calidad Industrial (INCALIN), Universidad Nacional de San Martín, Buenos Aires, May 2021.

[4] “What is Industrie 4.0?,” Plattform Industrie 4.0 - What is Industrie 4.0? [Online]. Available: https://www.plattform-i40.de/IP/Navigation/EN/Industrie40/WhatIsIndustrie40/what-is-industrie40.html. [Accessed: 11-Nov-2021].

[5] “The industrial Internet of Things Volume G1: Reference Architecture” Industrial Internet Consortium [Online] Available: https://www.iiconsortium.org/IIC_PUB_G1_V1.80_2017-01-31.pdf [Accessed: 17-Nov-2021].

[6] “SIMATIC IOT2000 The intelligent interface for the Industrial Internet of Things” Siemens [ONLINE] Available: https://new.siemens.com/us/en/products/automation/pc-based/iot-gateways/iot2000.html [Accessed: 17-Nov-2021].

[7] “IIoT Gateway RevPiConnect” KUNBUS GmbH [ONLINE] Available: https://revolutionpi.com/revpi-connect/ [Accessed: 17-Nov-2021].

[8] “IIOT GATEWAYS” Wieland [ONLINE] Available: https://www.wieland-electric.com/en/products/industrial-communication/iiot-gateways/ [Accessed: 18-Nov-2021].

[9] “Anybus Communicator IIoT - MQTT-OPC UA” Anybus by HMS Networks [ONLINE] Available: https://www.anybus.com/products/gateway-index/anybus-communicator/detail/anybus-iiot-communicator-mqtt-and-opc-ua-gen2 [Accessed: 18-Nov-2021].

[10] A. Glória, F. Cercas, and N. Souto, “Design and implementation of an IoT gateway to create smart environments,” Procedia Computer Science, vol. 109. Elsevier BV, pp. 568–575, 2017. doi: 10.1016/j.procs.2017.05.343

[11] F. Tietz, D. Brandão and L. F. Alves, "Development of an Internet of Things Gateway Applied to a Multitask Industrial Plant," *2018 13th IEEE International Conference on Industry Applications (INDUSCON)*, 2018, pp. 917-923, doi: 10.1109/INDUSCON.2018.8627273.

[12] I. V. Ferreira, J. A. Bigheti, and E. P. Godoy, “Development of a Wireless Gateway for Industrial Internet of Things Applications,” IEEE Latin America Transactions, vol. 17, no. 10. Institute of Electrical and Electronics Engineers (IEEE), pp. 1637–1644, Oct. 2019. doi: 10.1109/tla.2019.8986441.

[13] “Industrial IoT Application Development” Bridgera Monitoring Available: https://bridgera.com/industrial-iot/ [Accessed: 18-Nov-2021].

[14] “Industrial Remote Monitoring” Webee [ONLINE] Available: https://www.webee.io/industrial-remote-monitoring.html [Accessed: 18-Nov-2021].

[15] “ThingsBoard Open-source IoT Platform” ThingsBoard [ONLINE] Available: https://thingsboard.io [Accessed: 18-Nov-2021].

[16] “Industrial network market shares 2020 according to HMS Networks” HMS Networks [ONLINE] Available: https://www.hms-networks.com/news-and-insights/news-from-hms/2020/05/29/industrial-network-market-shares-2020-according-to-hms-networks

[17] “MQTT: The Standard for IoT Messaging” MQTT Org. [ONLINE] Available: https://mqtt.org [Accessed: 23-Nov-2021].

[18] W. J. Buchanan, “Modbus,” The Handbook of Data Communications and Networks. Springer US, pp. 677–687, 2004. doi: 10.1007/978-1-4020-7870-5_40.

[19] “An Introduction to Modbus RTU” Real Time Automation [ONLINE] Available: https://www.rtautomation.com/technologies/modbus-rtu/ [Accessed: 23-Nov-2021].

[20] “RS-485 Quick Guide TIA/EIA-485-A Standard” Analog Devices [ONLINE] Available: https://www.analog.com/media/en/technical-documentation/product-selector-card/rs485fe.pdf [Accessed: 23-Nov-2021].

[21] Mackay, S., Wright, E., Reynders, D. and Park, J. (2004). *Practical Industrial Data Networks*. Burlington: Elsevier.

[22] “MQTT and MQTT 5 Essentials A comprehensive overview of MQTT facts and features for beginners and experts alike.” 1st edn. HiveMQ GmbH, Ergoldinger Str. 2A 84030, Landshut Germany (2020)

[23] R. A. Light, “Mosquitto: server and client implementation of the MQTT protocol,” The Journal of Open-Source Software, vol. 2, no. 13. The Open Journal, p. 265, May 26, 2017. doi: 10.21105/joss.00265.

[24] “Node-RED Programming Guide” Sense Tecnic Systems Inc [ONLINE] Available: http://noderedguide.com [Accessed: 24-Nov-2021].

[25] “Working with context” OpenJS Foundation and Node-RED contributors [ONLINE] Disponible en: https://nodered.org/docs/user-guide/context [Accessed: 24-Nov-2021].

[26] “What is Arduino?” Arduino [ONLINE] Available: https://www.arduino.cc/en/Guide/Introduction [Accessed: 25-Nov-2021].

[27] “UNO R3” Arduino [ONLINE] Available: https://docs.arduino.cc/hardware/uno-rev3 [Accessed: 25-Nov-2021].

[28] “ATmega328P Datasheet” Arduino [ONLINE] Available: https://content.arduino.cc/assets/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf [Accessed: 25-Nov-2021].

[29] “Arduino Ethernet Shield V1” Arduino [ONLINE] Available: https://www.arduino.cc/en/Main/ArduinoEthernetShieldV1 [Accessed: 25-Nov-2021].

[30] “Nano” Arduino [ONLINE] Available: https://docs.arduino.cc/hardware/nano [Accessed: 25-Nov-2021].

[31] “ATmega48A/PA/88A/PA/168A/PA/328/P Datasheet” Arduino [ONLINE] Available: http://ww1.microchip.com/downloads/en/DeviceDoc/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061A.pdf [Accessed: 25-Nov-2021].

[32] “BMP180 Barometric Preassure/Temperature/Altitude Sensor-5V” Adafruit [ONLINE] Available: https://www.adafruit.com/product/1603 [Accessed: 25-Nov-2021].

[33] “BMP180 Digital Pressure sensor” Bosch Sensortec [ONLINE] Available: https://cdn-shop.adafruit.com/datasheets/BST-BMP180-DS000-09.pdf [Accessed: 25-Nov-2021].

[34] “MPU-6000 and MPU-6050 Product Specification Revision 3.4” Invensense [ONLINE] Available: https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf [Accessed: 25-Nov-2021].

[35] “What is a VPN? Virtual Private Network” Cisco [ONLINE] Available: https://www.cisco.com/c/en/us/products/security/vpn-endpoint-security-clients/what-is-vpn.html#~types-of-vpns [Accessed: 25-Nov-2021].

[36] “Create virtual private networks on-demand” LogMeIn [ONLINE] Available: https://www.vpn.net [Accessed: 25-Nov-2021].

[37] “Datasheet PC817X Series” SHARP [ONLINE] Available: https://www.farnell.com/datasheets/73758.pdf [Accessed: 27-Nov-2021].

[38] “5V Four-Channel Relay Module” Components 101 [ONLINE] Available: https://components101.com/switches/5v-four-channel-relay-module-pinout-features-applications-working-datasheet [Accessed: 27-Nov-2021].

[39] “Módulo Convertidor De Señal Corriente A Voltaje 4-30ma A 0-3.3V 5V 10V” Electronilab [ONLINE] Available: https://electronilab.co/tienda/modulo-convertidor-de-senal-corriente-a-voltaje-4-20ma-a-0-3-3v-5v-10v/ [Accessed: 27-Nov-2021].

[40] “Circuito de salida de 4-20 mA” Programmer Clic [ONLINE] Available: https://programmerclick.com/article/61451543491/ [Accessed: 27-Nov-2021].

[41] “SparkFun I2C DAC Breakout - MCP4725” SparkFun [ONLINE] Available: https://www.sparkfun.com/products/12918 [Accessed: 27-Nov-2021].

[42] “Datasheet MCP4725” Microchip [ONLINE] Available: https://cdn.sparkfun.com/datasheets/BreakoutBoards/MCP4725_2009.pdf [Accessed: 27-Nov-2021].

[43] “Datasheet MAX485” Maxim Integrated [ONLINE] Available: https://datasheets.maximintegrated.com/en/ds/MAX1487-MAX491.pdf [Accessed: 27-Nov-2021].

[44] “MPU-6000 and MPU-6050 Product Specification Revision 3.4” Invensense [ONLINE] Available: https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf [Accessed: 27-Nov-2021].

[45] N, O’Leary “Arduino Client for MQTT” [ONLINE] Available: https://github.com/knolleary/pubsubclient/blob/master/README.md [Accessed: 28-Nov-2021].

[46] “Example of a scrolling HTML Table based on incoming JSON objects” IoTPlay [ONLINE] Available: https://flows.nodered.org/flow/6f164cfd4b548d603c7387b29ed54027 [Accessed: 28-Nov-2021].

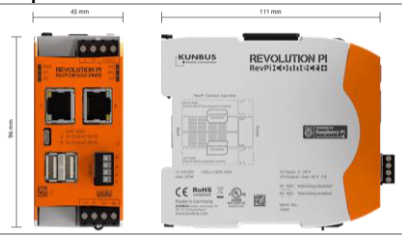
ANEXOS

Anexo A: Gateways IoT Comerciales

Tabla A.1. Características de los gateways IOT2040 e IOT2050 [6]

Característica	IOT2040	IOT2050
Descripción	Series de Gateways IoT Industriales desarrollados por Siemens que posibilitan la interconexión de dispositivos de campo a redes de empresa o a la nube, permitiendo incorporar la producción a un esquema de industria 4.0. Entre sus funcionalidades se encuentra la recolección de datos de campo, el análisis y almacenamiento de datos, y su transferencia a una red de nivel superior (red de empresa) o a la nube.	
Gateway		
Características técnicas		
Procesador	Intel Quark X1020	ARM TI AM6528 GB
Almacenamiento	1 slot tarjeta microSD	1 slot tarjeta microSD
Memoria	DDR3-SDRAM (1GB)	DDR4 (1GB)
Suministro	24 V	12/24 V
Características Funcionales		
Protocolos e Interfaces de comunicación	- Interface Serial COM: x2 (RS-232, RS-485) - Ethernet (RJ45): x2 100 Mbps - USB: x1 tipo A, x1 tipo micro-B	- Interface Serial COM: x1 (RS-232/422/485 configurable por software) - Ethernet (RJ45): x2 100-1000 Mbps - Puerto para Display - USB: x1 tipo A
	- WLAN: Tarjeta WLAN instalable en puerto MiniPCIe - Protocolos de campo: Modbus TCP, Modbus RTU, S7, Profinet - Protocolos de Red: MQTT, OPC UA, TCP/IP - Periféricos de comunicaciones Arduino: UART, SPI e I2C	
Entradas y Salidas A/D	- Entradas y Salidas de la interface de Arduino: Entradas/Salidas Digitales: 14 Entradas Analógicas: 6 PWM: 6 de las Salidas Serial UART: RX, TX I2C: SCL SDA SPI: SCK, MISO, MOSI, SS - Módulos Shield de Siemens: SIMATIC IOT2000 IO module 5x DI (24V), 2x DO (0-10V), 2x AI (0-10V o 4-20mA) SIMATIC IOT2000 Input Module Sink/Source 10 x DI	

Tabla A.2. Características de los gateways RevPiConnect [7]

Característica	RevPi Connect	RevPi Connect+
Descripción	Gateway IoT de categoría industrial de código abierto basado en Raspberry Pi 3, maneja protocolos comunes de IIoT como MQTT y OPC UA los cuales posibilitan la transferencia de datos directamente de campo hacia la nube. Es de características modulares cuenta con múltiples módulos de expansión como de entradas y salidas analógicas, comunicaciones industriales a nivel de campo cableadas o inalámbricas. Los distintos módulos de expansión le permiten adaptarse a otros protocolos industriales adicionales de acuerdo a la necesidad y se presentan a continuación en conjunto con todas las características del dispositivo.	
Gateway		
Características técnicas		
Procesador	Broadcom BCM2837 with Quad-Core ARM Cortex-A53	Broadcom BCM2837B0 with Quad-Core ARM Cortex-A53
Almacenamiento	eMMC 4GB	eMMC 8-32GB
Memoria	1GB	
Suministro	12-24 V	
Características Funcionales		
Protocolos e Interfaces de comunicación	**Interfaces del gateway:** - Ethernet (RJ45): x2 10-100 Mbps - Interface Serial: RS-485 - USB: x2 tipo A - HDMI: x1 Micro HDMI - Micro-USB: x1 Para transferir imágenes del sistema. - Buses PiBridge y ConBridge - Protocolos de Campo: Modbus RTU y Modbus TCP - Protocolos de Red: MQTT, OPC UA, TCP/IP **Comunicaciones con Dispositivos de Campo (expansiones):** - PROFINET IRT: Esclavo x2 RJ45, Velocidad de hasta 100Mbit/s - Ethernet/IP: Esclavo x2 TJ45, Velocidad de hasta 100Mbit/s - POWERLINK: Esclavo x2 RJ45, Velocidad de hasta 100Mbit/s - EtherCAT: Esclavo x2 RJ45, Velocidad de hasta 100Mbit/s - Sercos III: Esclavo x2 RJ45, Velocidad de hasta 100Mbit/s - Modbus TCP: Esclavo x2 RJ45, Velocidad de hasta 10/100Mbit/s - PROFIBUS: Esclavo mediante conector D-SUB 9F, Velocidad de hasta 12 Mbit/s - DeviceNet: Esclavo conector de 5 polos, Velocidad de hasta 500Kbit/s - CANopen: Esclavo mediante conector D-SUB 9M, Velocidad de hasta 1 Mbit/s	

	- Modbus RTU: Esclavo conector de 8 polos, Velocidad de hasta 115.2Kbit/s - Serial: Esclavo conector de 8 polos, Velocidad de hasta 115.2Kbit/s RS232/422/485 - DMX: Maestro/Esclavo conector de 8 polos - CAN: Conector de 4 polos, Velocidad de hasta 1Mbit/s **Comunicaciones inalámbricas:** - M-Bus: Módulo de Radiofrecuencia (Banda de 868MHz) hasta 500m con línea de vista, velocidades de 4.8, 32, 100kbps - M-Bus VHP: Módulo de Radiofrecuencia (Banda de 169MHz) hasta 8000m con línea de vista, velocidades de 2.4, 4.8, 19.2kbps
Entradas y Salidas A/D	**Entradas y Salidas A/D:** - RevPi MIO: x8 AI de voltaje (0…10 V), x8 AO de voltaje (0…10 V) x4 DIO (24V) **Entradas y Salidas Digitales:** • RevPi DIO: x14 DI (24V y 2.5mA) y x14 DO (Tipo Transistor 24VDC y 0.5A) • RevPi DI: x16 DI (24V y 2.5mA) • RevPi DO: x16 DO (Tipo Transistor 24VDC y 0.5A) **Entradas y Salidas Analógicas:** • RevPi AIO: x4 AI de Voltaje (+/- 10V, +/-5V, 0…10V, 0…5V) o de Corriente (0…20mA, 0…24mA, 4…20mA), x2 AO de Voltaje (+/-10V, +/-11V, +/-5V, 0…10V,0…11V, 0…5V, 0…5.5V) o de Corriente 0…20mA, 0…24mA, 4…20mA.) o x2 canales para RTD

Tabla A.3. Características de los gateways WIENET IOT [8]

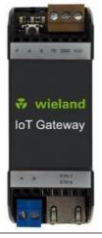
Característica	WIENET IOT SK 115-W	WIENET IOT SK 100-DIO8-3G-W
Descripción	El fabricante Wieland ofrece modelos de gateways que integran funcionalidades de gateway IIoT como recolectar procesar y filtrar datos de campo y transmitirlos a la nube y funcionalidad de gestión de VPN en un mismo dispositivo. Posibilita el diagnóstico desde cualquier lugar del mundo y el acceso a los dispositivos de campo de manera remota. Posee funcionalidades de servidor OPC UA y cliente MQTT.	
Gateway		
Características Funcionales		
Protocolos e Interfaces de comunicación	- Interface Serial: x1 RS485/232 - Ethernet (RJ45): x2 10/100 Mbps	- Interface Serial: x1 RS485/232 - Ethernet (RJ45): x1 10/100 Mbps - Modem Mobile 3G
	- Protocolos de campo: Modbus TCP, Modbus RTU - Protocolos de niveles superiores: MQTT, OPC UA	
Entradas y Salidas A/D	Ninguna	4 entradas digitales, 4 salidas digitales

Tabla A.4. Características del gateway Anybus Communicator IIoT [9]

Característica	Anybus Communicator IIoT
Descripción	Permite integrar dispositivos de automatización viejos que manejan comunicaciones seriales RS-232/422/485 (Por defecto equipos que manejan protocolos Modbus RTU y ASCII) con la posibilidad de conectar hasta 31 nodos a sistemas de monitoreo basados en los protocolos de comunicaciones del internet industrial de las cosas OPC UA o MQTT también integra un cliente de email, un servidor FTP y un servidor web embebido para diagnóstico y visualización de datos.
Gateway	
Características Funcionales	
Protocolos e Interfaces de comunicaciones	Interfaces del gateway: - Ethernet (RJ45): x2 - Interface Serial: Configurable RS-232/422/485 Puede monitorear hasta 31 estaciones (RS-422/485) con velocidades de 1, 2-57.6 kbit/s Protocolos de Campo: Modbus RTU (Por defecto) Protocolos de niveles superiores: OPC UA y MQTT
Entradas y Salidas A/D	Ninguna

Anexo B: Plataformas de monitoreo y control

- **Industrial IoT Application Development – Bridgera [13]**

El desarrollador ofrece una solución llamada Bridgera Monitoring que se encuentra enfocada al monitoreo de dispositivos en tiempo real y la toma de decisiones en función de los datos obtenidos. Entre las funcionalidades se tiene:

- Monitoreo Industrial y de equipos de manufactura en tiempo real mediante un dashboard.
- Compatibilidad con todos los dispositivos
- Acceso a tendencias históricas y análisis de desempeño bajo demanda.
- Alertas y notificaciones SMS y e-mail bajo cambios de estados

El ejemplo que proporciona el desarrollador corresponde a un dispositivo con múltiples sensores que permite medir la temperatura, presión, humedad y estado de carga de la batería del dispositivo, también cabe destacar que se puede tener distintos niveles de acceso que pueden ser configurados desde una vista jerárquica.

Figura B.1. Dashboard de Bridgera Monitoring [13].

Por cada dispositivo se tiene varias pestañas cuya funcionalidad será resumida a continuación:

Live Status: Permite analizar en tiempo real los estados de los sensores, cuenta con valores numéricos y gráficas de tendencias en tiempo real, en el ejemplo del fabricante los datos son actualizados con una frecuencia de 1 segundo, pero la frecuencia de actualización de los datos se encuentra determinada por la velocidad de actualización del dispositivo.

Response Log: Permite retener la información de los datos en una base de datos que cuenta con una marca temporal para todas las variables.

Insight: En esta pestaña se muestran curvas de análisis de los datos en periodos de tiempo especificados por el usuario, como pueden ser curvas de campanas, contabilización de incidencias de alarmas o eventos, u otros gráficos de análisis estadísticos.

Alarm Settings: Permite configurar alarmas para cada una de las variables, entre los campos que son configurables se tiene, nombre de la alarma, los parámetros de activación de la alarma (límites), las notificaciones que pueden ser generadas (Vía SMS, Email o Pop-up), la configuración de los destinatarios de la alarma y la habilitación de la alarma.

Actions: Permite obtener los estados del dispositivo, configurarlo o reiniciarlo

Map: Permite visualizar la locación de un dispositivo en el mundo y verificar su estado, si el dispositivo cuenta con un sistema incorporado de posicionamiento GPS puede visualizarse su posición en tiempo real.

Alarm history: Permite visualizar un registro de las alarmas generadas

- **Industrial Remote Monitoring – Webee [14]**

Consiste en una solución IIoT en la que se posibilita el monitoreo operacional de los procesos, entre los dispositivos que pueden ser monitoreados se menciona máquinas en contacto directo con un proceso industrial como lo pueden ser controladores lógicos programables o redes OPC UA, o SCADA cuya información puede ser accedida en tiempo real.

El sistema permite dar seguimiento a valores críticos del proceso, condiciones ambientales, uso de recursos, y generar alertas de anomalías o análisis de tendencias para mejorar la eficiencia de los procesos industriales.

En la Figura B.2 se presenta algunos de los widgets de monitoreo presentados por el desarrollador.

Figura B.2. Widgets de monitoreo de Webee [14].

ThingsBoard – Open-source IoT Platform [15]

Es una Plataforma de recolección de datos, procesamiento, visualización y manejo de dispositivos de código abierto. Posibilita la conectividad de dispositivos con protocolos standard de la industria del internet de las cosas como MQTT, CoAP y HTTP. El desarrollador asegura una alta tolerancia al fallo, escalabilidad y desempeño.

En cuanto a la gestión de dispositivos se proporciona una plataforma de monitoreo y control basada en APIs (Application Programming Interface) que permiten el acceso seguro a las distintas entidades IoT ya sean dispositivos, recursos, consumidores, entre otros.

Otra de las características de la plataforma corresponde a la recolección, almacenamiento y visualización de datos mediante dashboards, algunos de los widgets se presentan en la figura B.3.

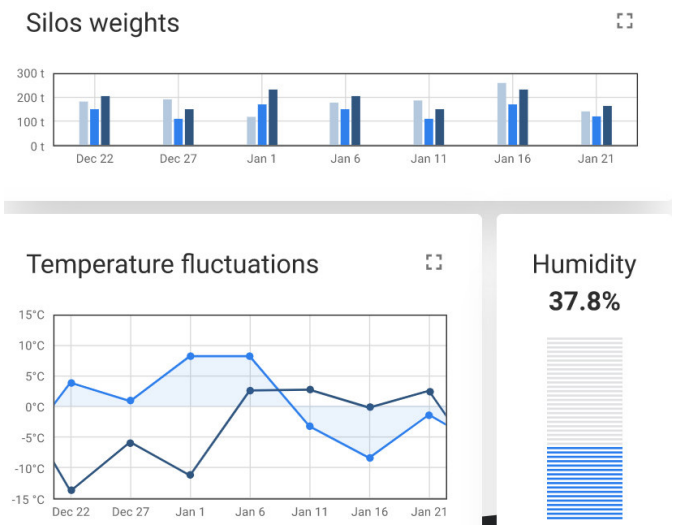


Figura B.3. Widgets de monitoreo de ThingsBoard [15].

También se puede definir reglas de procesamiento de datos que permiten transformar y normalizar la información proveniente de los dispositivos incluyendo la funcionalidad de generación de alarmas ante eventos entrantes de telemetría, actualización de atributos, inactividad de dispositivos o acciones del usuario.

Anexo C: Características de las tarjetas embebidas Arduino UNO R3 y Arduino Nano y de los sensores BMP180 y MPU-6050

Tabla C.1. Especificaciones Arduino UNO R3 [27] [28]

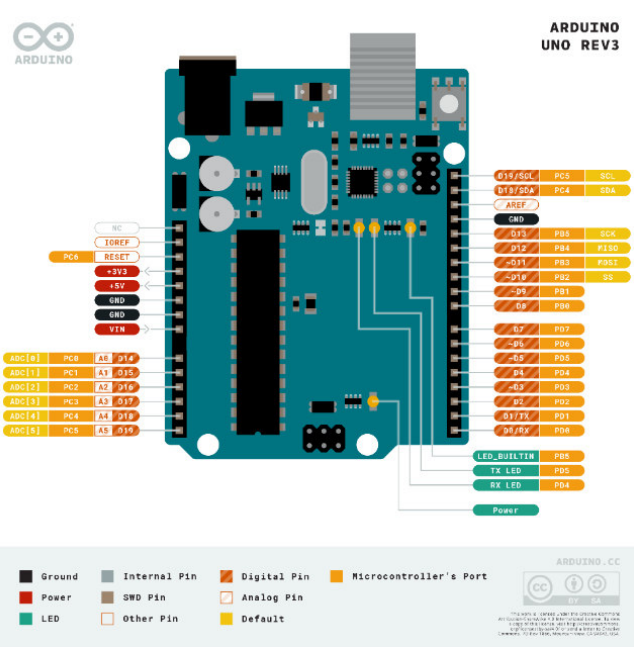
Tarjeta	**Nombre**	**Arduino UNO R3**
Microcontrolador	ATmega328P	
Conector USB	USB-B	
Pines	Entradas/Salidas Digitales	14
	Entradas Analógicas	6
	Pines PWM	6
Distribución de pines		
Comunicaciones	Serial UART (Universal Asynchronous receiver/transmitter)	Si, 2 líneas Rx, Tx en pines digitales 0 y 1 respectivamente
	I2C	Si, 2 líneas SCL, SDA en los pines analógicos A5 y A4 respectivamente, con una velocidad máxima de hasta 400 kBit/s
	SPI	Si, 4 líneas SCK, MISO, MOSI y SS en los pines digitales 13,12,11,10 respectivamente hasta la velocidad máximo del reloj del microcontrolador de 16MHz
Potencia	Voltaje	De operación: 5 [V] Recomendado: 7-12 [V] Límites: 6-20 [V] Conector: Jack 9 [V] Corriente por pin: 20 [mA]
Velocidad	Procesador Principal	ATmega328P 16MHz
	Procesador USB-Serial	ATmega16U2 16MHz
Memoria	ATmega328P	SRAM: 2KB FLASH: 32KB EEPROM: 1KB

Tabla C.2. Especificaciones Arduino Nano [30] [31]

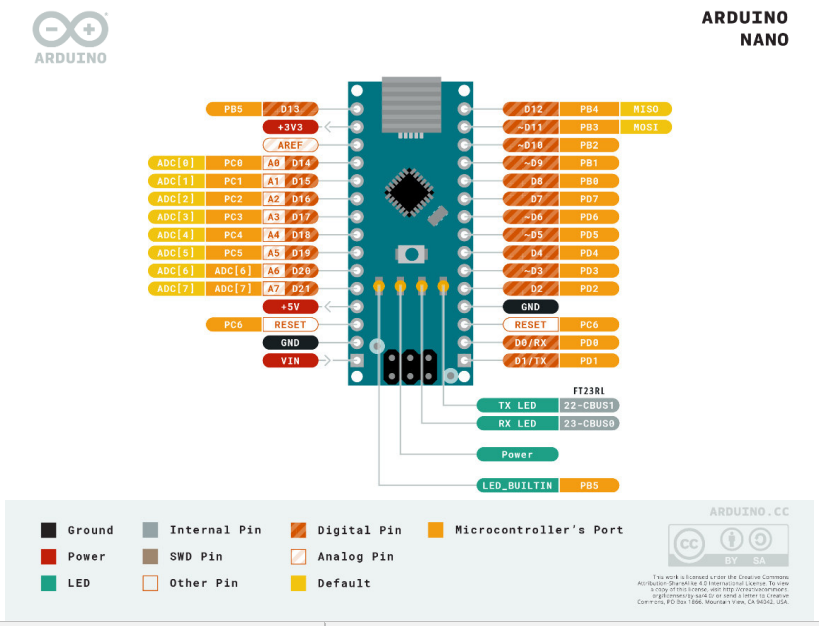
Tarjeta	**Nombre**	**Arduino Nano**
Microcontrolador	ATmega328P	
Conector USB	Mini USB-B	
Pines	Entradas/Salidas Digitales	14
	Entradas Analógicas	8
	Pines PWM	6
Distribución de pines		
Comunicaciones	Serial UART (Universal Asynchronous receiver/transmitter)	Si, 2 líneas Rx, Tx en pines digitales 0 y 1 respectivamente
	I2C	Si, 2 líneas SCL, SDA en los pines analógicos A5 y A4 respectivamente, con una velocidad máxima de hasta 400 kBit/s
	SPI	Si, 4 líneas SCK, MISO, MOSI y SS en los pines digitales 13,12,11,10 respectivamente hasta la velocidad máximo del reloj del microcontrolador de 16MHz
Potencia	Voltaje	De operación: 5 [V] Recomendado: 7-12 [V] Corriente por pin: 40 [mA]
Velocidad	Procesador	ATmega328 16MHz
Memoria	ATmega328P	SRAM: 2KB FLASH: 32KB EEPROM: 1KB

Tabla C.3. Características del sensor BMP180 [33].

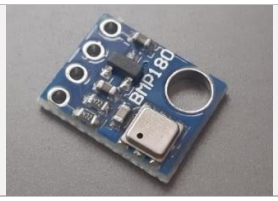
Especificación	Detalle
Sensor	
Rango de Presión	300 a 1100 hPa
Resolución de Presión	0.01 hPa=0.01mbar
Rango de temperatura	0 a 65 ºC
Resolución de temperatura	0.1 ºC
I2C Fast and High Speed	Velocidad de hasta 3.4Mbit/s
Dirección I2C de 7 bits	0x77
Pines	SDA: Datos SCL: Señal de Reloj VCC: Alimentación 5V GND

Tabla C.4. Características del sensor MPU6050 [34].

Especificación	Detalle
Sensor	
Rangos seleccionables (Giroscopio)	+/- 250, 500, 100, 2000 º/s
Rangos seleccionables (Acelerómetro)	+/- 2, 4, 8, 16 g
Rango de temperatura	-40 a 85 ºC
I2C Fast and High Speed	Velocidad de hasta 3.4Mbit/s
Dirección I2C de 7 bits	0x68 o 0x69
Pines	SDA: Datos SCL: Señal de Reloj VCC: Alimentación 5V GND AD0: Dirección en bus I2C (si 0L 0x68, si 1L 0x69) INT: Interrupción XDA: Pin auxiliar de datos XCL: Pin auxiliar de señal de reloj

Anexo D: Tópicos de publicación y Subscripción gestionados por el prototipo

Tabla D.1. Tópicos de Publicación y Subscripción.

Tipo	Alias	Tópico	Detalle del dato asociado
Subscribe	sub_topic01	t/DO0	Salida digital DO0
	sub_topic02	t/DO1	Salida digital DO1
	sub_topic03	t/DO2	Salida digital DO2
	sub_topic04	t/DO3	Salida digital DO3
	sub_topic05	t/AO0	Salida analógica AO0
	sub_topic06	t/000001	Registro Coil de dirección 000001
	sub_topic07	t/000002	Registro Coil de dirección 000002
	sub_topic08	t/000003	Registro Coil de dirección 000003
	sub_topic09	t/000004	Registro Coil de dirección 000004
	sub_topic10	t/000005	Registro Coil de dirección 000005
	sub_topic11	t/000006	Registro Coil de dirección 000006
	sub_topic12	t/000007	Registro Coil de dirección 000007
	sub_topic13	t/000008	Registro Coil de dirección 000008
	sub_topic14	t/400001	Holding Register de dirección 400001
	sub_topic15	t/400002	Holding Register de dirección 400002
	sub_topic16	t/400003	Holding Register de dirección 400003
	sub_topic17	t/400004	Holding Register de dirección 400004
Publish	pub_topic01	t/AI0	Entrada analógica AI0
	pub_topic02	t/AI1	Entrada analógica AI1
	pub_topic03	t/S1	Variable Temperatura
	pub_topic04	t/S2	Variable Presión
	pub_topic05	t/S3	Variable Altura
	pub_topic06	t/DI0	Entrada Digital DI0
	pub_topic07	t/DI1	Entrada Digital DI1
	pub_topic08	t/100001	Registro Contact de dirección 100001
	pub_topic09	t/100002	Registro Contact de dirección 100002
	pub_topic10	t/100003	Registro Contact de dirección 100003
	pub_topic11	t/100004	Registro Contact de dirección 100004
	pub_topic12	t/100005	Registro Contact de dirección 100005
	pub_topic13	t/100006	Registro Contact de dirección 100006
	pub_topic14	t/100007	Registro Contact de dirección 100007
	pub_topic15	t/100008	Registro Contact de dirección 100008
	pub_topic16	t/300001	Input Register de dirección 300001
	pub_topic17	t/300002	Input Register de dirección 300002
	pub_topic18	t/300003	Input Register de dirección 300003
	pub_topic19	t/300004	Input Register de dirección 300004
	pub_topic20	t/AcX	Aceleración Lineal en X
	pub_topic21	t/AcY	Aceleración Lineal en Y
	pub_topic22	t/AcZ	Aceleración Lineal en Z
	pub_topic23	t/GyX	Velocidad angular en X
	pub_topic24	t/GyY	Velocidad angular en Y
	pub_topic25	t/GyZ	Velocidad angular en Z

Anexo E: Implementación, Configuración y Pruebas de Mosquitto Broker MQTT

Descarga e Instalación

1. En la página principal de Eclipse Mosquitto (presentada en la figura E.1): https://mosquitto.org se debe seleccionar la sección “Download” que permitirá desplegar la página de descargas presentada en la figura E.2 donde se seleccionará el instalador adecuado, en este caso mosquitto-2.0.14-install-windows-x64.exe, al dar clic en la opción el navegador empezará a descargar el software.

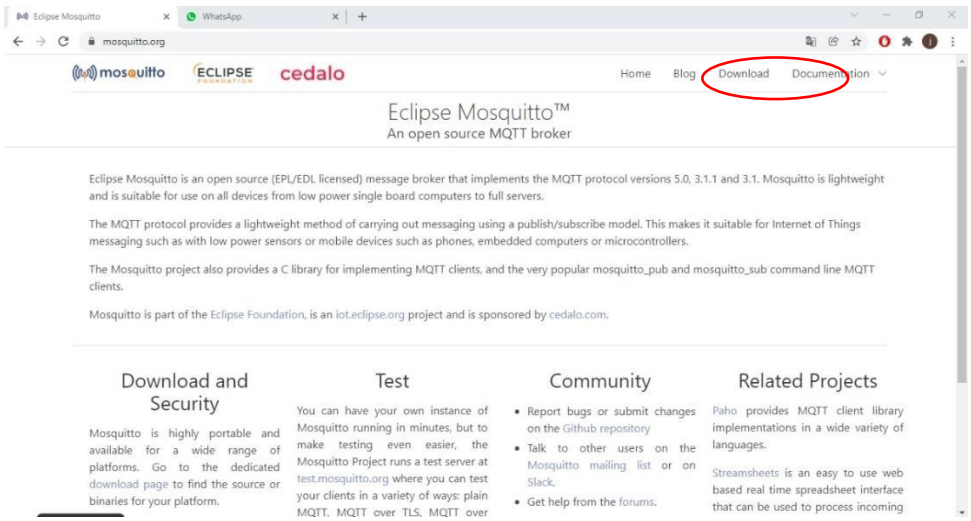


Figura E.1. Página principal de Eclipse Mosquitto.

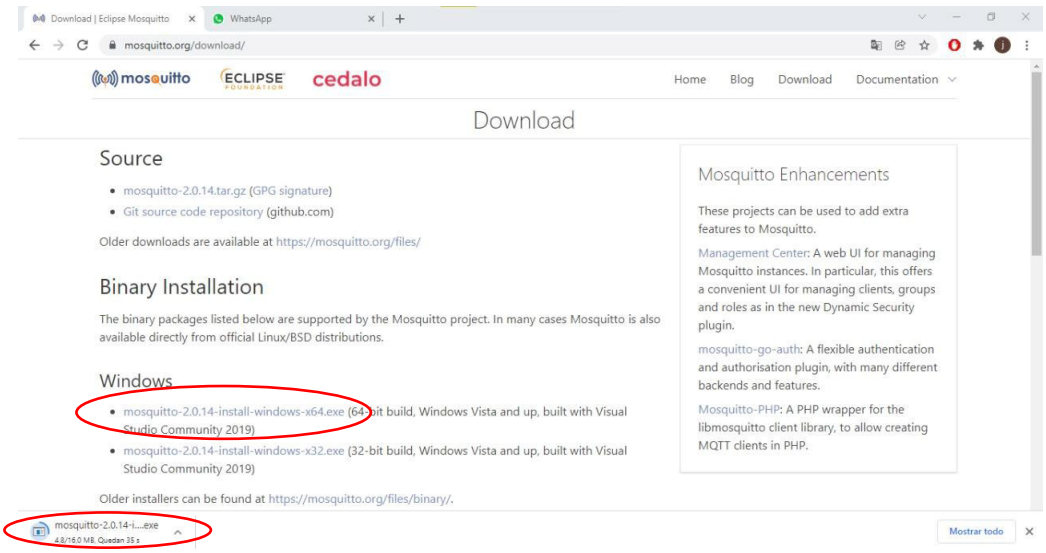


Figura E.2. Descarga de Mosquitto MQTT Broker.

2. Posteriormente a la descarga se deberá ejecutar el instalador y seguir los pasos de instalación presentados en la figura E.3: Instalar todos los componentes, seleccionar carpeta de instalación y esperar a que el proceso finalice.

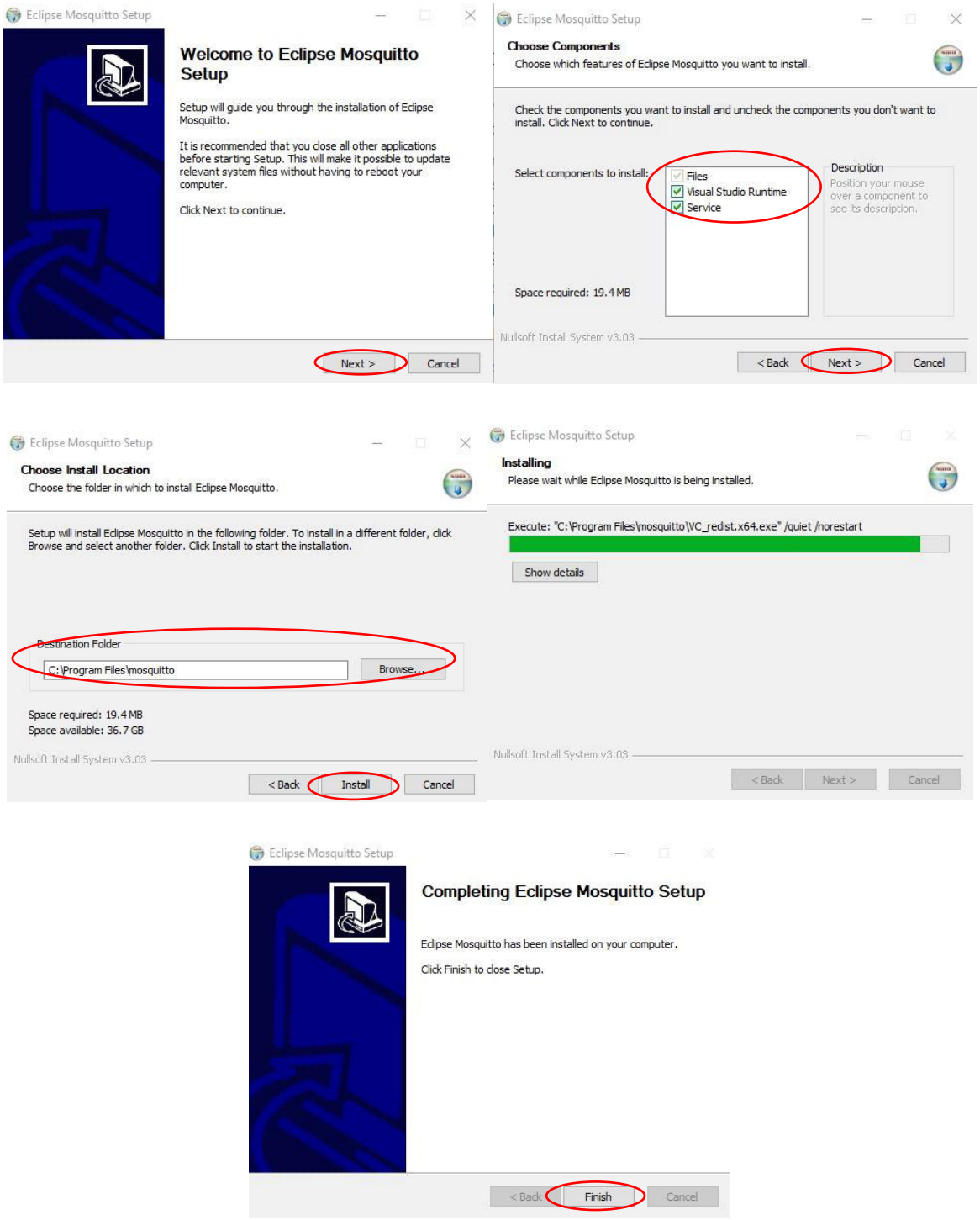


Figura E.3. Instalación de Mosquitto MQTT Broker

3. Finalmente, para poner en marcha el Broker se accede a los servicios de Windows, se localiza el servicio “Mosquitto Broker” (figura E.4), se da clic derecho y se lo ejecuta dando clic en iniciar.

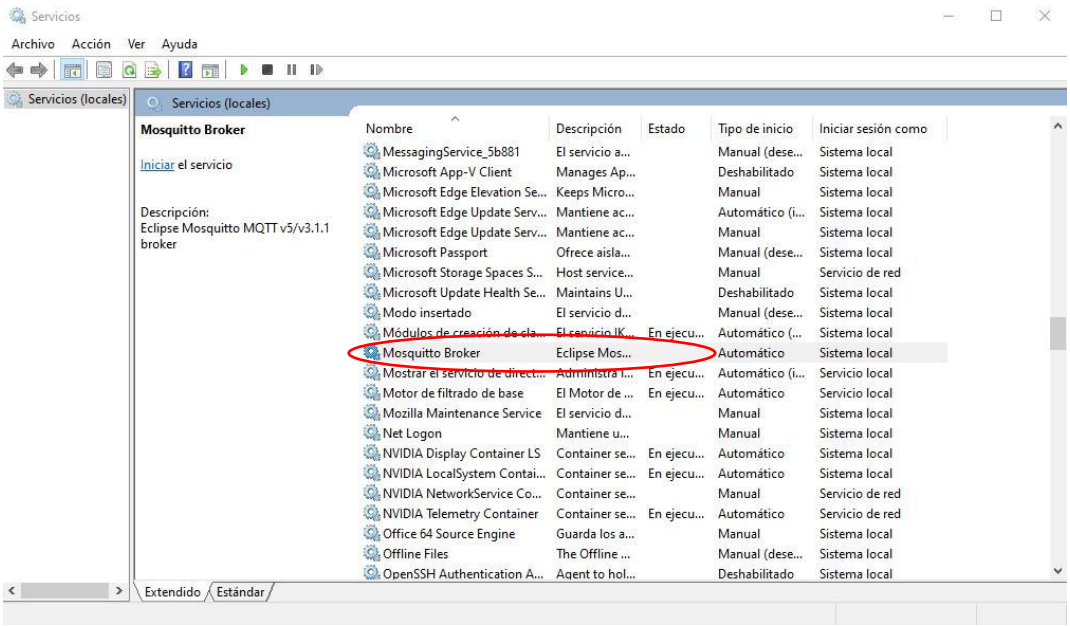


Figura E.4. Ejecución de Broker MQTT.

Configuración del Broker

1. Para configurar al Broker se debe acceder a la carpeta de instalación y se debe generar un archivo .conf mediante el bloc de notas con los parámetros:

allow_anonymous true (Permite conexión de clientes MQTT al broker sin especificar un nombre usuario ni contraseña)

listener 1883 (Permite configurar el puerto mediante el cual el broker escuchará a los clientes MQTT)

En la figura E.5 se presenta la ruta típica de instalación y el archivo de configuración.

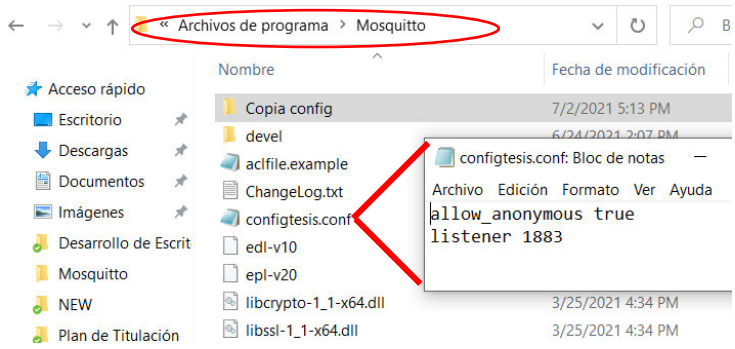


Figura E.5. Archivo de Configuración del Broker.

2. Para que la configuración del broker sea efectuada es necesario abrir el símbolo del sistema de Windows, acceder al directorio de instalación mediante los comandos “cd ..” y “cd C:\Program Files\Mosquitto” e ingresar la línea de comando: mosquitto -c configtesis.conf que permitirá carga la configuración tal y como se muestra en la figura E.6.

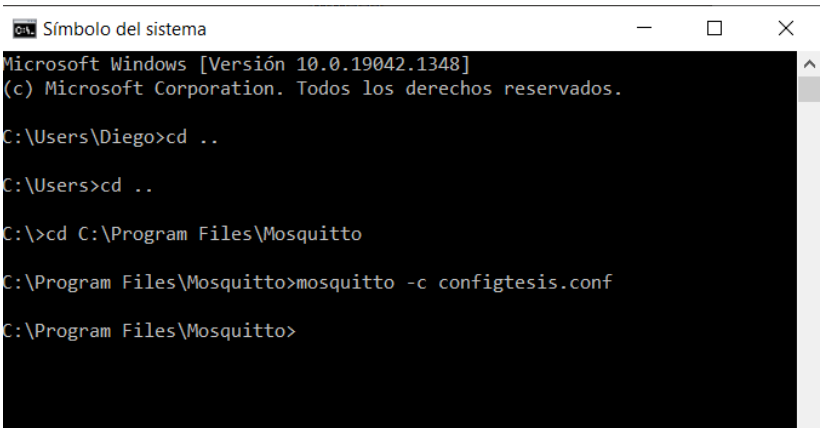


Figura E.6. Carga de la configuración del broker.

Pruebas de publicación y subscripción mediante mosquitto_pub y mosquitto_sub

Como se lo mencionó en el marco teórico el software posee clientes de publicación y subscripción para testear las funcionalidades del broker, para la prueba se procederá a abrir dos símbolos del sistema y acceder a la carpeta de instalación del broker de la misma manera que en la sección anterior, el objetivo es utilizar el cliente mosquitto_sub en una de las ventanas para subscribirse al tópico “prueba” y utilizar el cliente mosquitto_pub en la otra ventana para publicar la cadena “hola mundo” bajo el tópico “prueba”.

Para la subscripción se debe ingresar la línea:

mosquitto_sub -h localhost -t prueba

Permite especificar que el broker se encuentra en la misma máquina (localhost) y el tópico de subscripción es (prueba)

Para la publicación se debe ingresar la línea:

mosquitto_pub -h localhost -t prueba -m “hola mundo”

Permite especificar que el broker se encuentra en la misma máquina (localhost), el tópico de publicación es (prueba) y el mensaje es “hola mundo”.

En la figura E.7 se presenta las líneas ingresadas y las salidas del proceso de prueba de ambas ventanas utilizadas.

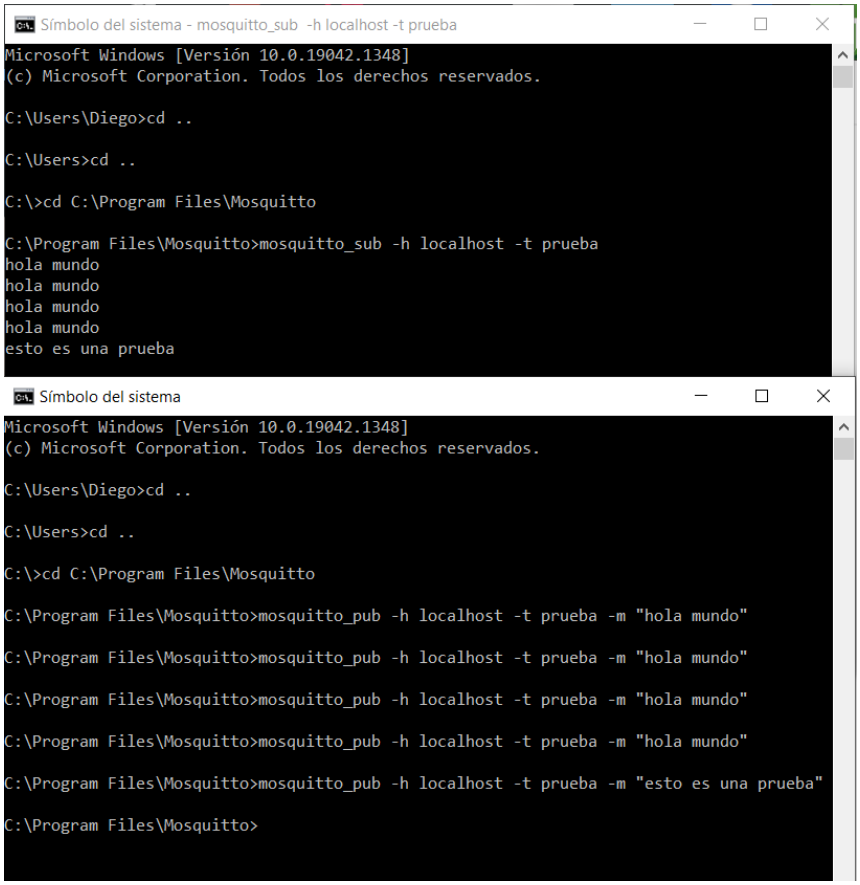


Figura E.7. Prueba de subscripción y publicación.

Anexo F: Descarga e Instalación de Node-RED

1. Descargar el instalador de Node.js de la página de descargas de: https://nodejs.org/es/

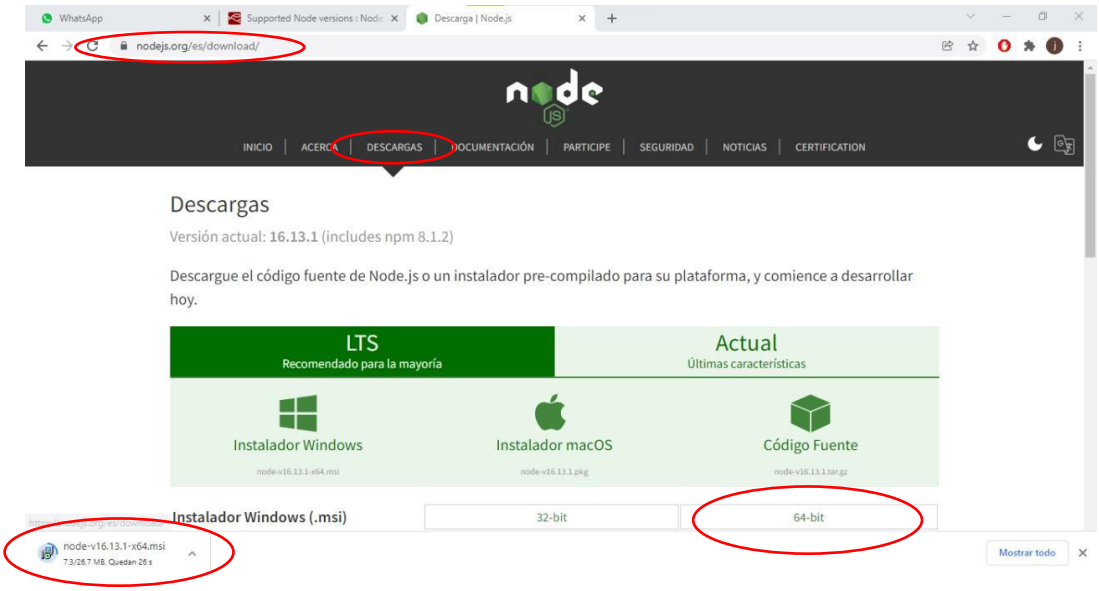


Figura F.1. Página de descargas de Node.js

2. Ejecutar el instalador y seguir los pasos de instalación: aceptar términos y condiciones, seleccionar ruta de instalación y seleccionar todas las características como en la figura F.2.

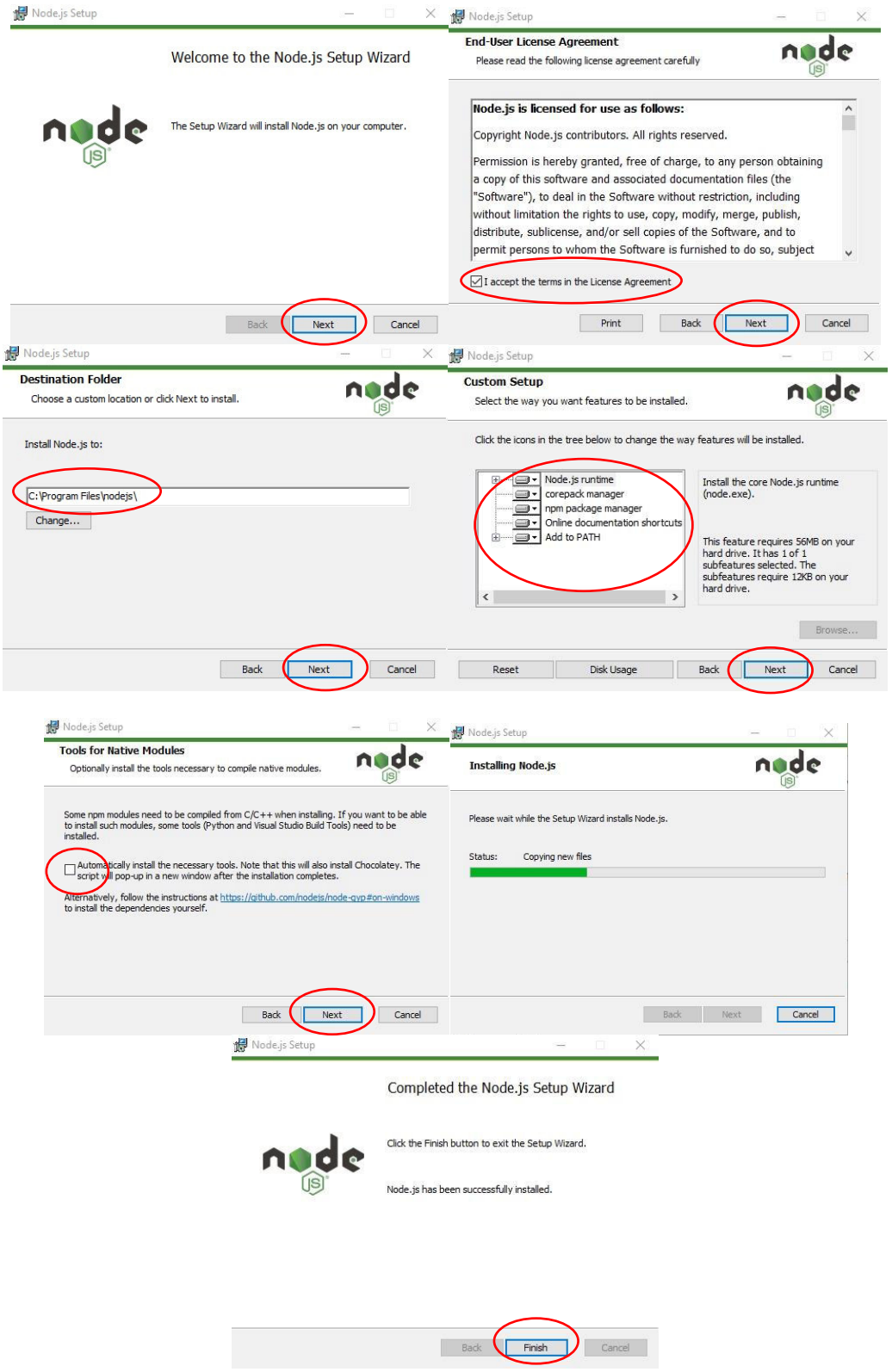


Figura F.2. Pasos de instalación de Node.js

3. Finalizada la instalación de Node.js se debe abrir el símbolo del sistema e ingresar la línea node –version && npm –version para verificar que la instalación fue correcta como se muestra en la figura F.3.

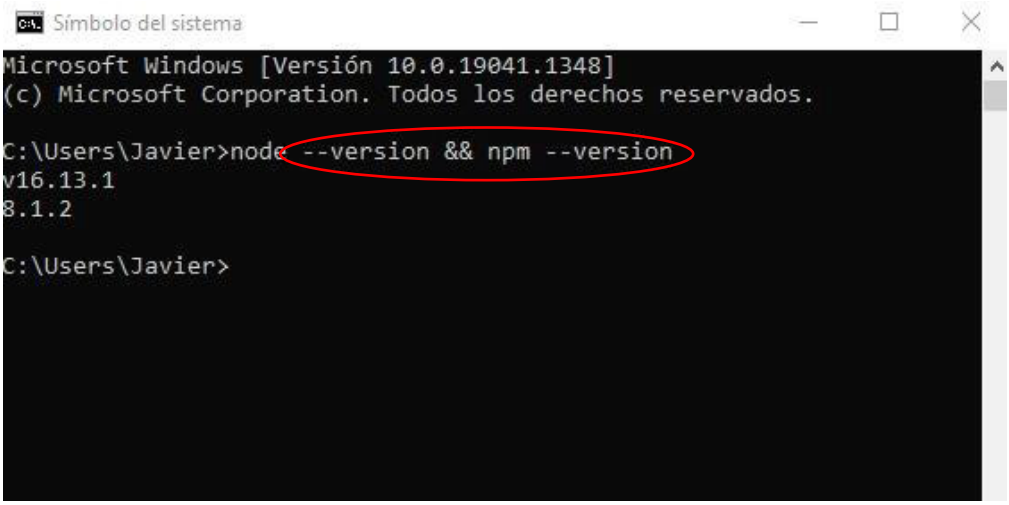


Figura F.3. Verificación de instalación de Node.js

4. Ahora se procede a instalar Node-RED ingresando la línea de comandos npm install -g –unsafe-perm node-red

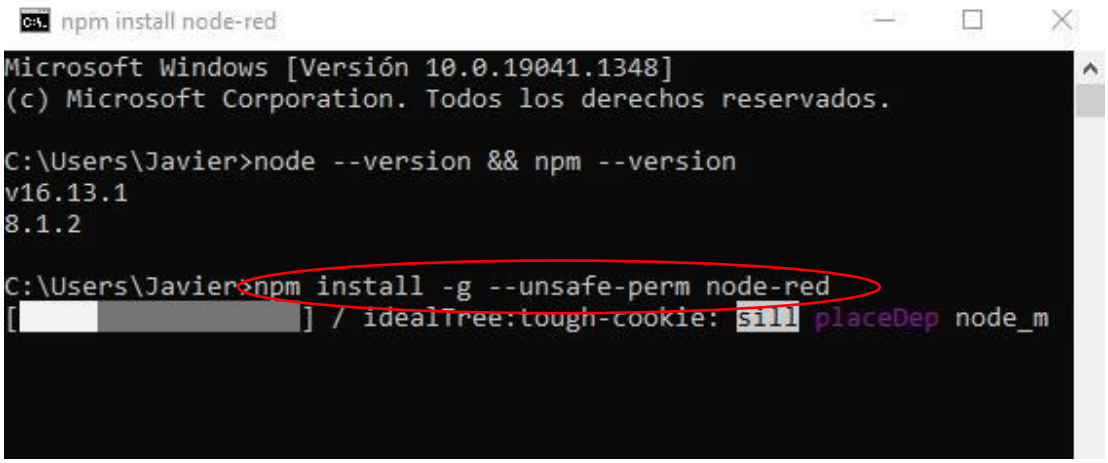


Figura F.4. Instalación de Node-RED.

5. Finalmente se procede a ejecutar node-red ingresando en el símbolo del sistema node-red como se puede ver en la figura F.5.

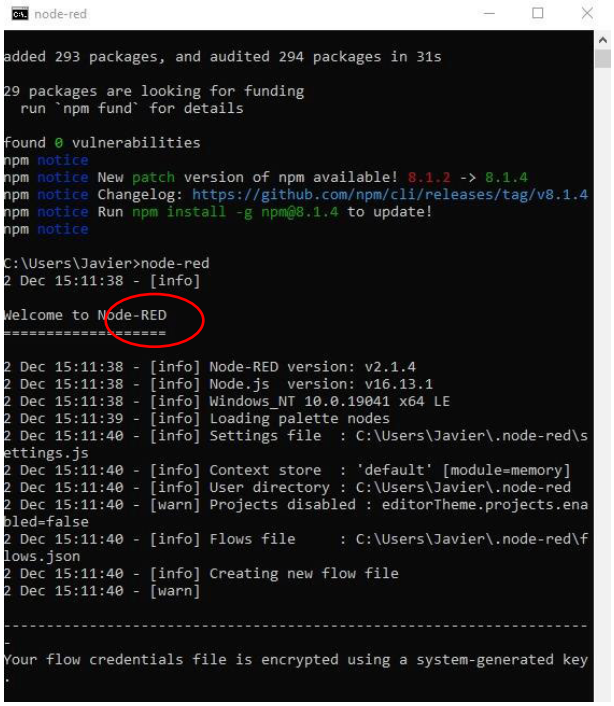


Figura F.5. Ejecución de Node-RED

Ahora se podrá ingresar al entorno de edición de flujos al ingresar en un navegador la dirección IP y el puerto, en este caso se ejecuta en la misma computadora por lo que se ingresa mediante localhost:1880 como se puede ver en la figura F.6.

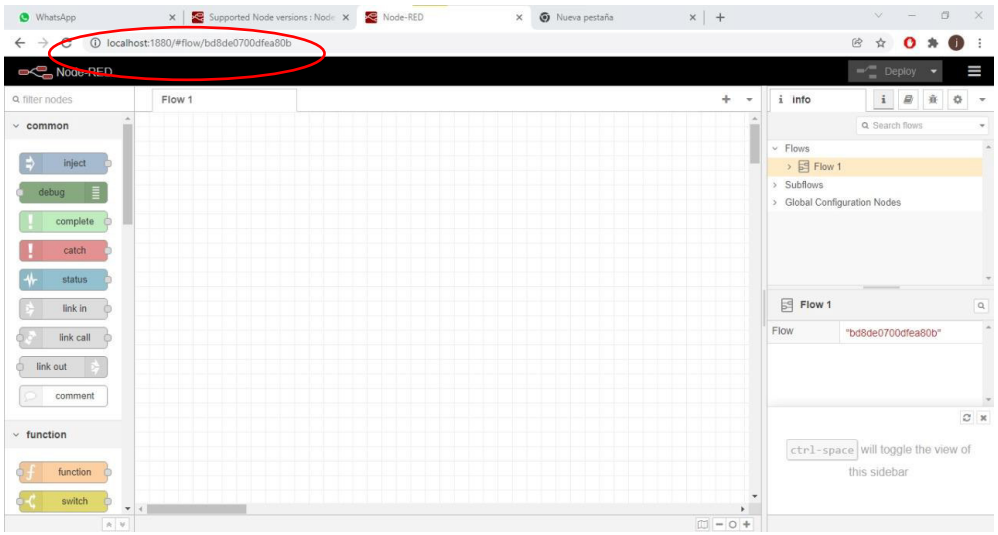
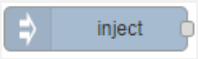
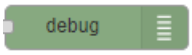
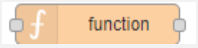
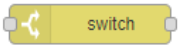
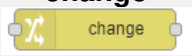


Figura F.6. Entorno de Node-RED

En el entorno se puede notar que los nodos mediante los cuales se desarrollará las aplicaciones se encuentran a la izquierda en la paleta de nodos.

En la siguiente tabla se resumen distintos nodos instalados por defecto que fueron utilizados para el trabajo:

Tabla F.1. Nodos por defecto de Node-RED.

Nodo	Descripción
Inject 	Permite inyectar un mensaje en un flujo de manera manual o automática a intervalos regulares, el mensaje puede ser de distintos tipos incluyendo cadenas de caracteres, objetos JavaScript o el tiempo actual. A la salida del nodo se obtendrá un mensaje con los parámetros msg.payload y msg.topic que pueden contener información.
debug 	Permite visualizar en la pestaña de depuración el mensaje en el punto del flujo al que el nodo se encuentra conectado. Puede ser configurado para mostrar solo el parámetro msg.payload o todo el objeto u arreglo JavaScript. También incorpora el tiempo en el que el mensaje fue recibido.
link in 	Permite crear una conexión entre los nodos. Es utilizado para ingresar mensajes cuando los nodos en la interfaz de programación se encuentran demasiado alejados
link out 	Permite crear una conexión entre los nodos. Es utilizado para enviar mensajes a otros nodos en la interfaz de programación que se encuentran demasiado alejados
function 	Permite generar una función JavaScript con los mensajes que se recibe en la entrada del nodo. Se espera que la función siempre retorne un mensaje que podría tener varios parámetros. Por convención el mensaje de entrada se encuentra en la propiedad msg.payload
switch 	Permite enrutar los mensajes según condiciones de valor de una de las propiedades del mensaje o su secuencia. Cuando un mensaje llega, el nodo evalúa las condiciones y enruta a determinada salida el mensaje recibido. Las condiciones pueden ser resultado de una comparación con una cadena de caracteres, un valor, o una expresión JSONata.
change 	Permite cambiar, borrar o mover las propiedades de un mensaje, flujo u objeto global. Pueden aplicarse múltiples reglas de tratamiento de las propiedades simultáneamente.
delay 	Permite generar retardos en cada mensaje que llega al nodo o limitar la tasa a la que los mensajes pueden ir al siguiente nodo del flujo.
template 	Permite generar una plantilla cuyos parámetros sean función de los mensajes de la entradas. Puede ser configurado para programar la generación de platillas mediante HTML
mqtt in 	Posibilita la conexión con un MQTT Broker como un cliente que se encuentre subscrito a un tópico en específico. Entre las salidas del nodo se tiene: payload: cadena con el valor asociado al tópico. (Ejemplo: 20ºC) topic: Permite al Broker identificar cual mensaje debe llegar al nodo (Ejemplo: topic/Temperatura) qos: permite definir el tipo de calidad de servicio de mensajería entre Broker y el nodo. Cabe destacar que en el nodo se debe configurar la dirección IP del servidor (Broker) y el puerto que generalmente es 1883.
mqtt out 	Análogamente al nodo previo permite establecer la conexión con un MQTT Broker solo que en este caso se trata de un cliente tipo Publisher que se encarga de publicar un mensaje para su uso bajo un tópico asociado para el filtrado de datos. Posee las mismas características que el nodo previamente mencionado.

También se incorporó nodos adicionales en la paleta mediante el siguiente procedimiento:

Se da clic en “Manage palette” en el menú de la plataforma (figura F.7) para desplegar una ventana de configuraciones cuya pestaña “Install” permite añadir nodos adicionales desarrollados por los contribuyentes de Node-RED como se muestra en la figura E.8. Los nodos de las contribuciones pueden ser encontrados con descripciones y detalles de su funcionamiento en la página de la comunidad: https://flows.nodered.org

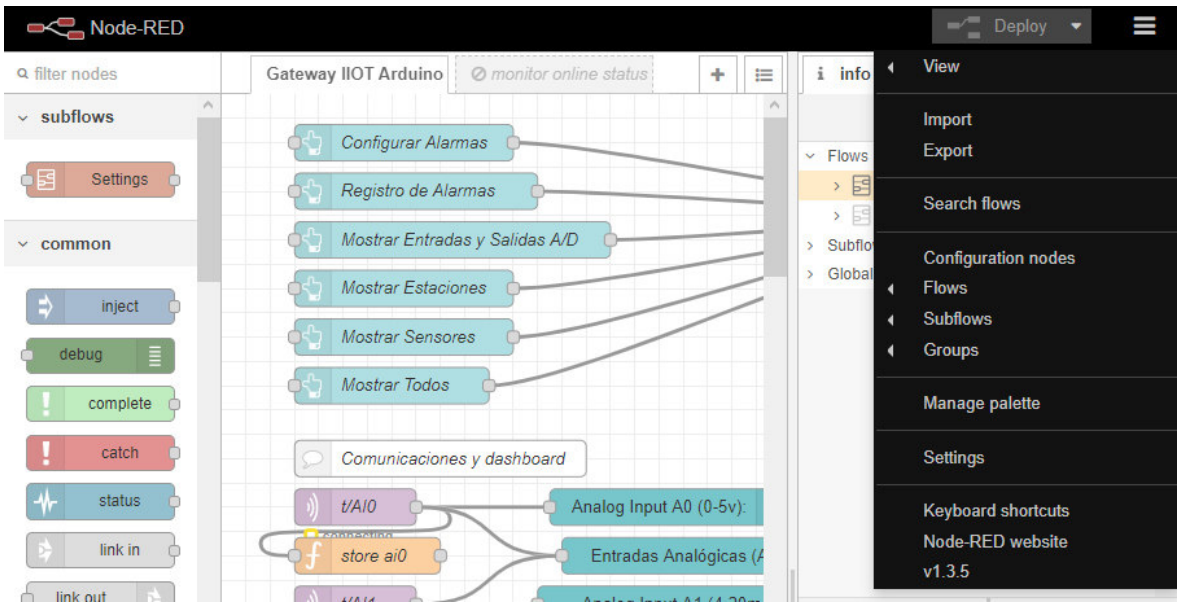


Figura F.7. Menú de la plataforma

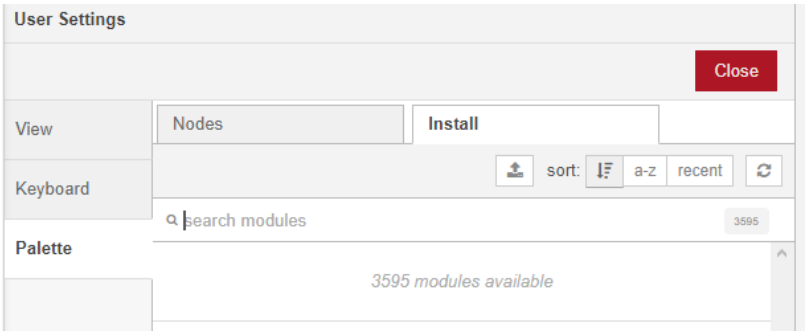


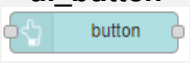
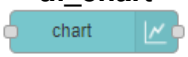
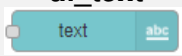
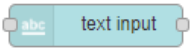
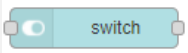
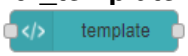
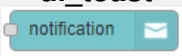
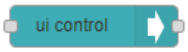
Figura F.8. Instalación de nodos

A continuación, se resumirá los que podrían ser utilizados para la generación de la interfaz de monitoreo.

Paquete: node-red-dashboard

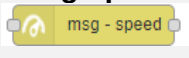
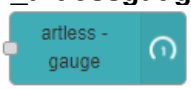
La interfaz de monitoreo puede ser accedida mediante un navegador web al ingresar la dirección IP de la pc seguido de /ui (Ejemplo: localhost:1883/ui)

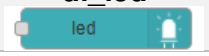
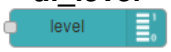
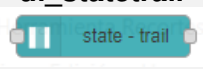
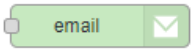
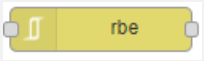
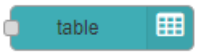
Tabla F.2 Nodos de Dashboard

Nodo	Descripción
ui_button 	Permite añadir un botón a la interfaz, al dar clic en el botón se generará un mensaje con un parámetro msg.payload configurado en la programación que puede tener una cadena o valor numérico.
ui_chart 	Posibilita la visualización de tendencias en tiempo real en una gráfica, puede ser configurado para que sean ejes, gráficos de barras o de tipo pastel cuyos valores límites son configurables (de ser el caso). El parámetro msg.payload del mensaje entrante es convertido a un valor numérico previamente a la generación de un nuevo punto, un valor erróneo que no pueda ser convertido será descartado.
ui_text 	Permite generar en la interfaz texto no editable. Podría ser configurado para mostrar valores de variables.
ui_text_input 	Posibilita al usuario ingresar información que puede ser enviada como mensaje cada que el texto es modificado.
ui_gauge 	Permite incorporar un medidor circular con aguja en la interfaz de visualización cuyos límites pueden ser configurados manualmente. El parámetro msg.payload del mensaje entrante es convertido a un valor numérico previamente a la generación de un nuevo punto, un valor erróneo que no pueda ser convertido será descartado.
ui_slider 	Permite incorporar un slider con límites en la interface de monitoreo. Cada que el slider cambie de valor se generará un msg.payload con el valor.
ui_switch 	Permite añadir un interruptor a la interfaz, al producirse un cambio de estado se generará un mensaje con un parámetro msg.payload con el estado nuevo del interruptor
ui_template 	Posibilita la creación de elementos dinámicos que cambien su apariencia según los mensajes de entrada. Podría ser utilizado para incorporar código HTML para generar contenido como por ejemplo tablas
ui_toast 	Permite generar una notificación pop up en la interfaz del usuario.
ui_ui_control 	Permite controlar de manera dinámica el dashboard, por ejemplo, ocular o mostrar pantallas o segmentos de una pestaña.

Finalmente se mencionan distintos nodos de otras contribuciones que fueron utilizados para generar la interfaz:

Tabla F.3. Nodos adicionales.

Paquete	Nodo	Descripción
node-red-contrib-msg-speed	**msg-speed** 	Permite contabilizar la cantidad de mensajes que llegan al nodo según intervalos de tiempo, puede ser utilizado para visualizar el desempeño del sistema
node-red-contrib-ui-artless-gauge	**ui_artlessgauge** 	Permite incorporar medidores sencillos en la interfaz de visualización

node-red-contrib-ui-led	**ui_led** 	Incorpora en la interfaz un visualizador de un led booleano.
node-red-contrib-ui-level	**ui_level** 	Permite la visualización de una variable numérica sobre una regla limitada
node-red-contrib-ui-state-trail	**ui_statetrail** 	Permite visualizar en la interfaz de monitoreo en tiempo real los cambios de estado de variables booleanas.
node-red-node-email	**email** 	Permite enviar un mensaje vía email a varios destinatarios configurables. Requiere de una cuenta en Gmail que sea la encargada de enviar los correos de manera automatizada cuando exista un mensaje en su entrada. Podría incorporar incluso archivos adjuntos.
node-red-node-rbe	**rbe** 	Permite el paso de datos solo si el parámetro msg.payload ha sufrido un cambio de estados, caso contrario se ignora el mensaje
node-red-node-uitable	**ui_table** 	Permite incorporar un widget de visualización en la interfaz de monitoreo que pueda mostrar datos en una entrada, requiere que se ingrese un arreglo de datos.

Anexo G: Descarga, Instalación y Configuración de la aplicación de cliente VPN LogMeIn Hamachi

1. Descargar el programa de la página principal de Hamachi: https://www.vpn.net

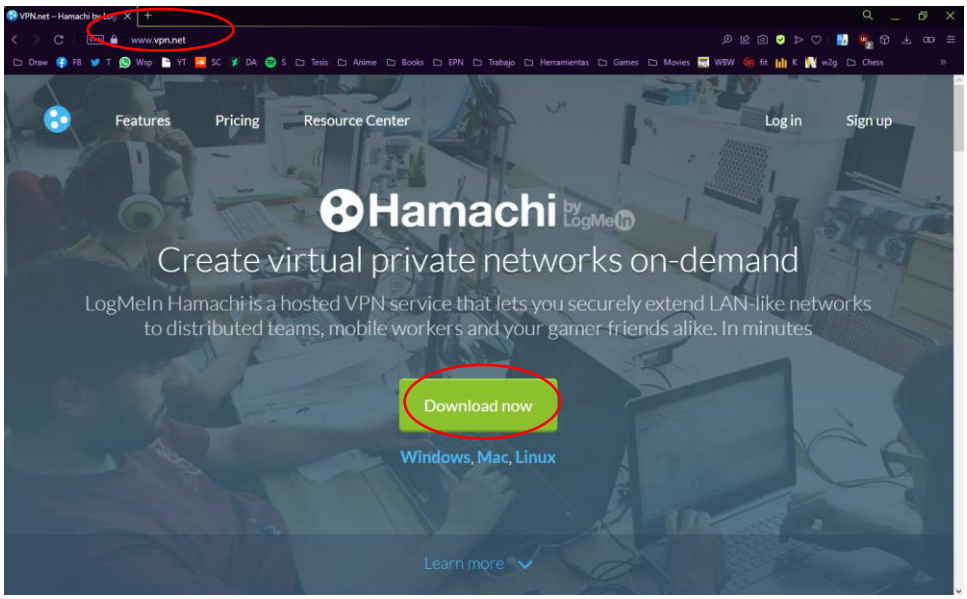


Figura G.1. Página principal de Hamachi

2. Al ejecutar el instalador se deben seguir cada uno de los pasos de instalación: seleccionar idioma, marcar la casilla de aceptación de términos y condiciones, elegir la carpeta de destino y esperar a que el proceso termine. El instalador suele ofrecer software adicional no requerido para la aplicación por lo que no debe ser instalado.

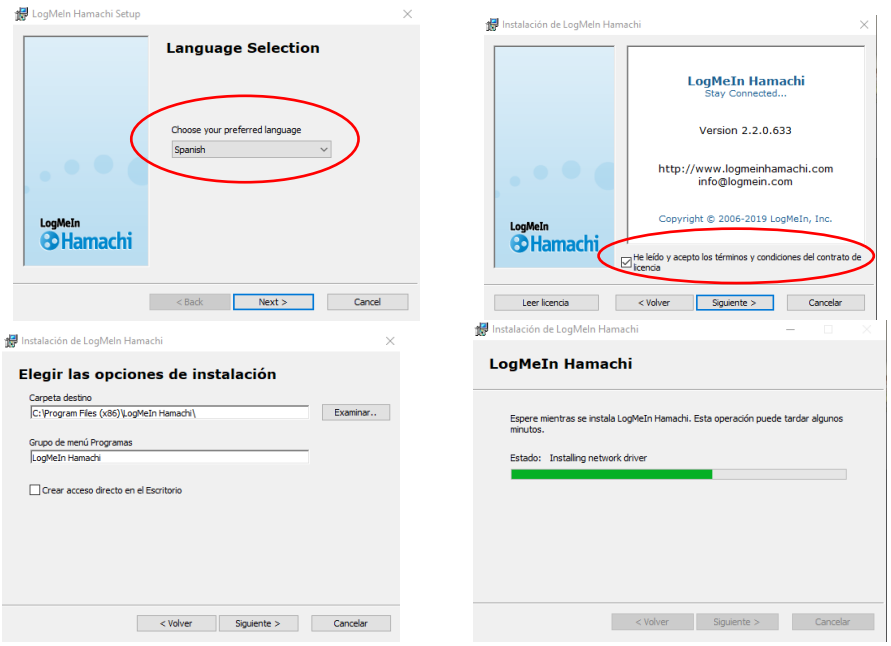


Figura G.2. Proceso de Instalación

3. Una vez instalado, se debe dar clic en el botón de encendido el cual desplegará una ventana de inicio de sesión y registro presentada en la figura G.3, se debe crear una ID mediante un correo electrónico y una contraseña.

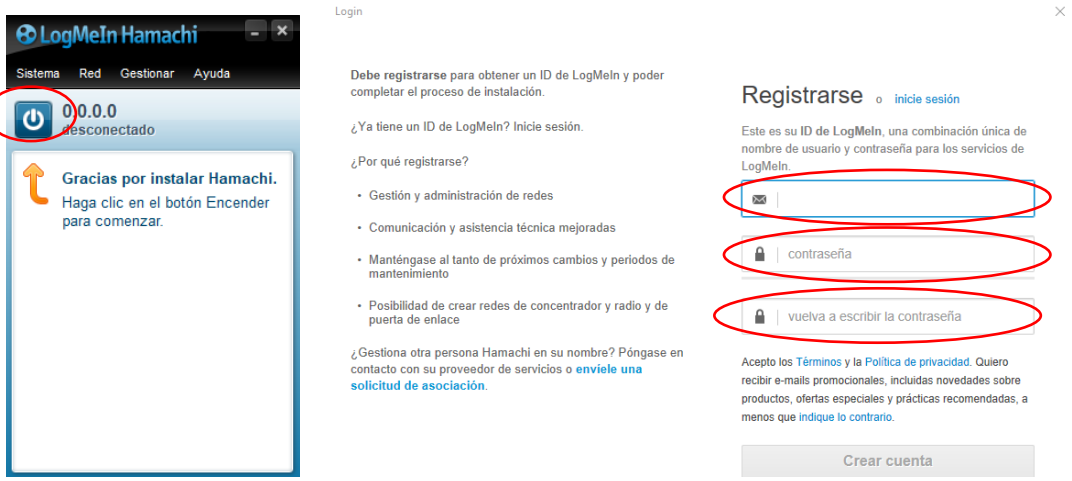


Figura G.3. Registro de usuario.

4. En caso de que se esté creando una nueva red privada virtual por primera vez se debe seleccionar “Crear una nueva red” e ingresar los parámetros de ID y contraseña mediante las cuales los clientes podrán acceder a la red como se muestra en la figura G.4.

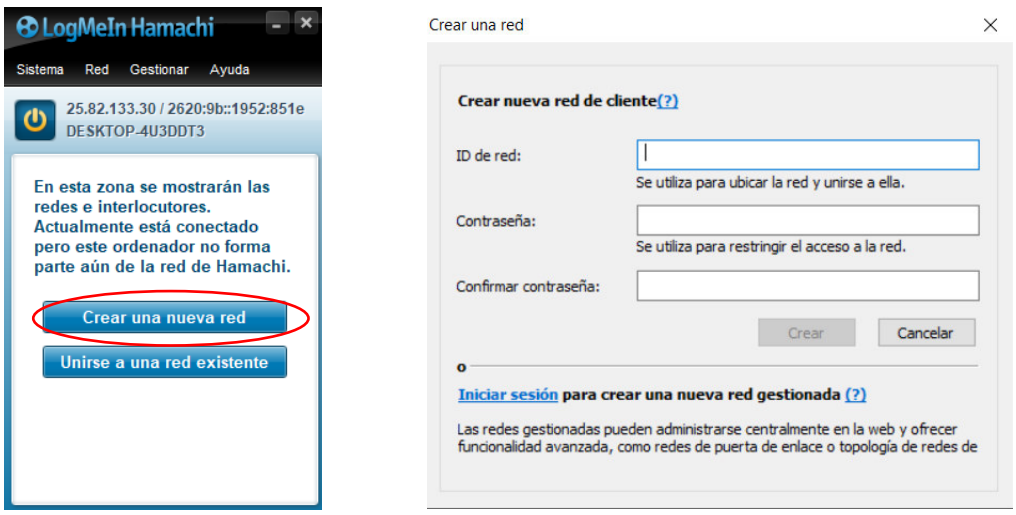


Figura G.4. Creación de una nueva red.

5. Para que un cliente remoto pueda unirse a la VPN se debe seleccionar “Unirse a una red existente” y se debe ingresar el nombre de la red y la contraseña mediante los cuales se creó la red, en este caso la red fue creada mediante la PC principal con los parámetros ID: IIOT-DIEGO y Contraseña: IIOT-DIEGO.

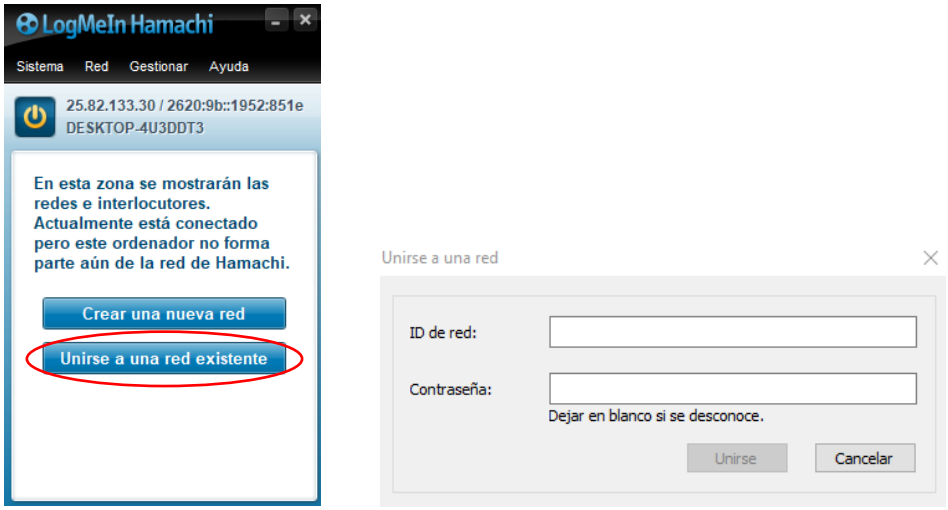


Figura G.5. Proceso de acceso a una red existente.

6. Posteriormente se podrá visualizar todos los dispositivos conectados a la red creada (como se puede ver en la figura G.6), en este caso: “DESKTOP-7HLURN6” corresponde a un cliente remoto que permite testear el acceso por medio de internet a la interfaz de monitoreo y control implementada en el cliente “LaptopDiego” que corresponde a la computadora donde se gestionan los servicios de Mosquitto Broker y Node-RED que se encuentran en ejecución.

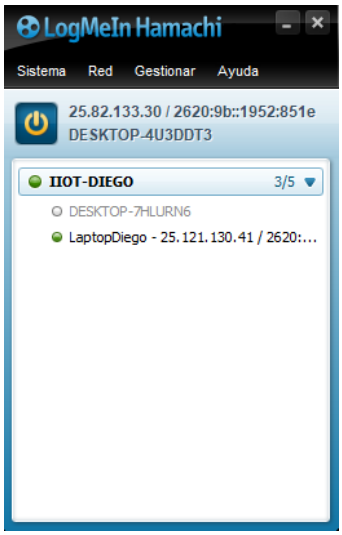


Figura G.6. Red privada virtual.

7. Finalmente se debe verificar la habilitación en el firewall del programa del cliente y el motor de tunelamiento que permite establecer la conexión a la red, por lo que se debe acceder a la configuración/firewall de Windows defender y seleccionar la opción “Permitir una aplicación o una característica a través de Firewall de Windows Defender” como se presenta en la figura G.7.

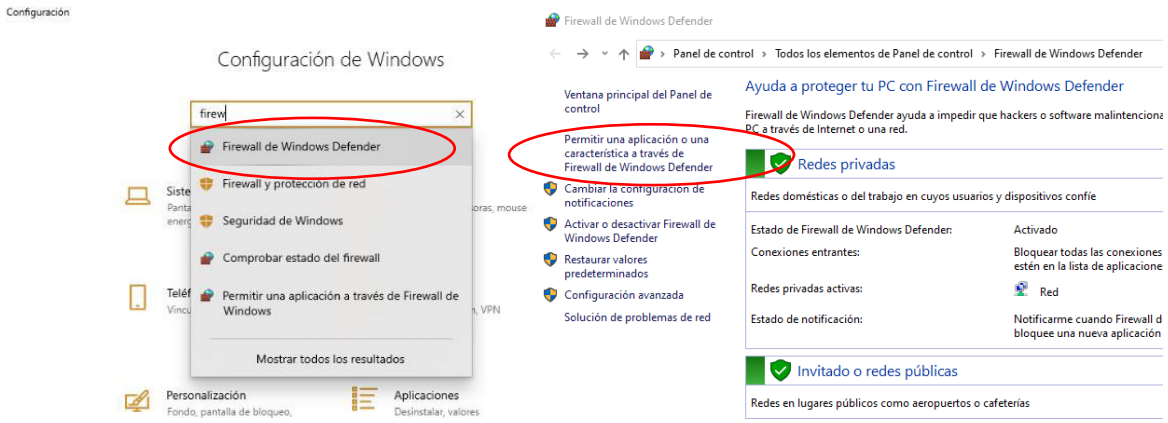


Figura G.7. Permitir aplicación a través del Firewall de Windows Defender.

8. Al acceder a la ventana se debe dar clic en “Cambiar Configuración” y luego “Permitir otra aplicación…”, en la pantalla desplegable se debe dar clic en examinar y localizar la carpeta de instalación del software como se puede observar en la figura G.8.

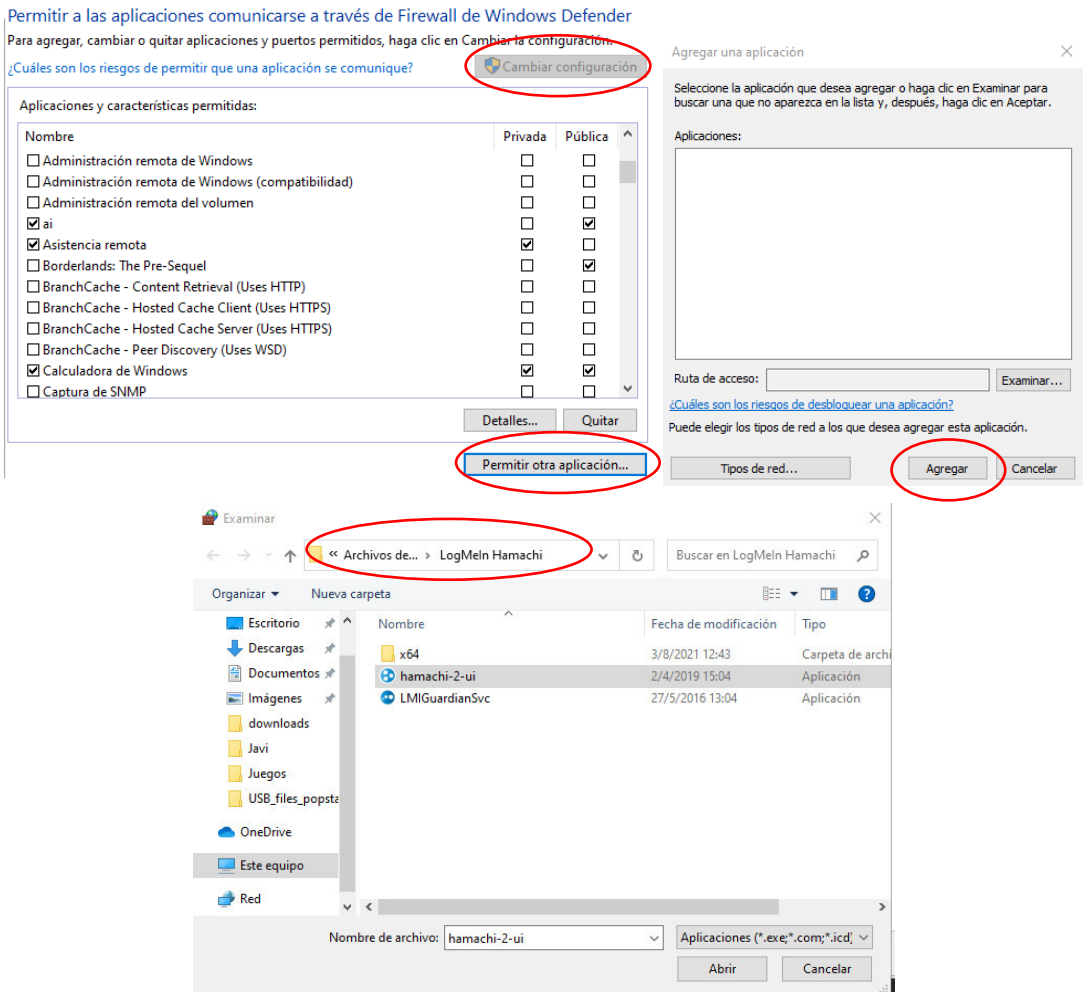


Figura G.8. Búsqueda de aplicaciones por habilitar

9. Una vez localizado se añade la aplicación del Cliente de Hamachi “hamachi-2-ui”, y la aplicación “hamachi-2” de la capeta x64 de la misma ruta una por una.

En la figura G.9 se presenta el proceso de inclusión de “hamachi-2-ui”

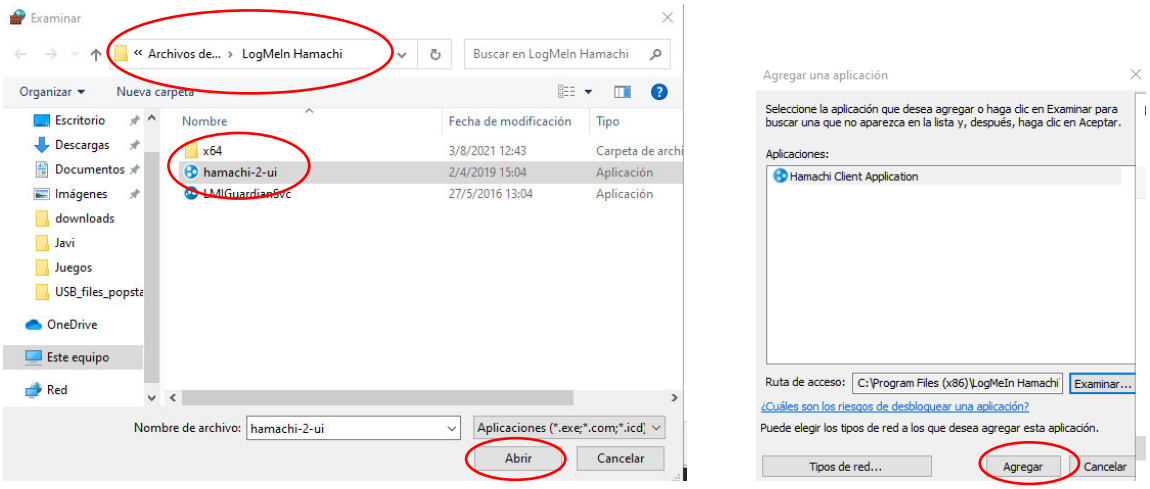


Figura G9. Aplicación “hamachi-2-ui” añadida

En la figura G.10 se presenta la carpeta x64 de la aplicación “hamachi-2” que también debe ser añadida de la misma manera.

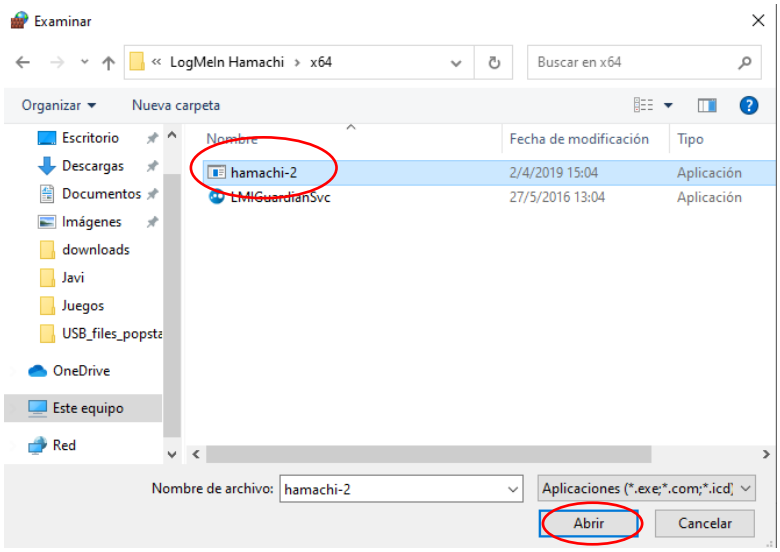


Figura G.10. Carpeta de la aplicación “hamachi-2”

Finalmente, ambas aplicaciones añadidas se enlistarán en la ventana (figura G.11), para habilitar su funcionamiento se debe marcar las casillas privada y pública y se deberá dar clic en Aceptar.

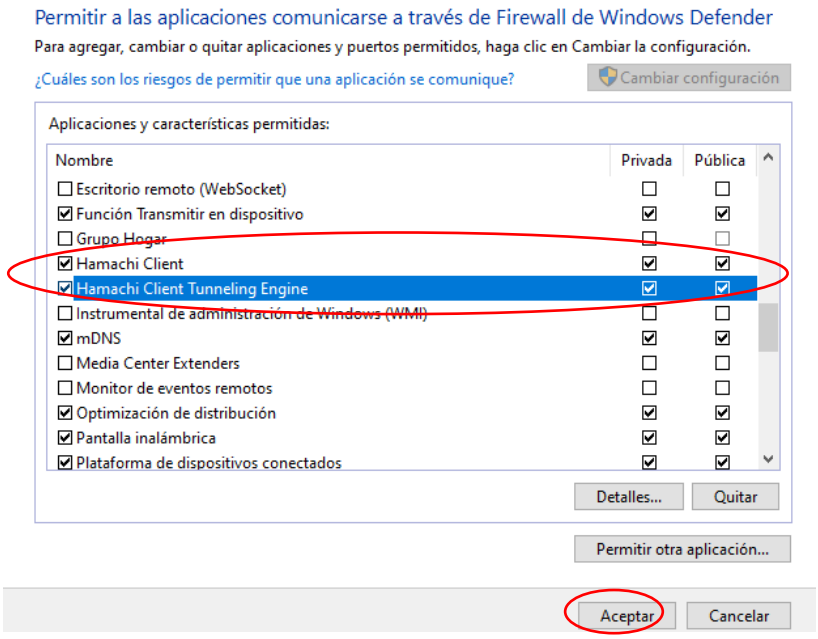


Figura G.11. Habilitación de las aplicaciones

10. Para acceder a la interfaz desde el equipo remoto basta con escribir la ip Hamachi en el navegador, en este caso de LaptopDiego 25.121.130.41:1880/ui (figura G.12)

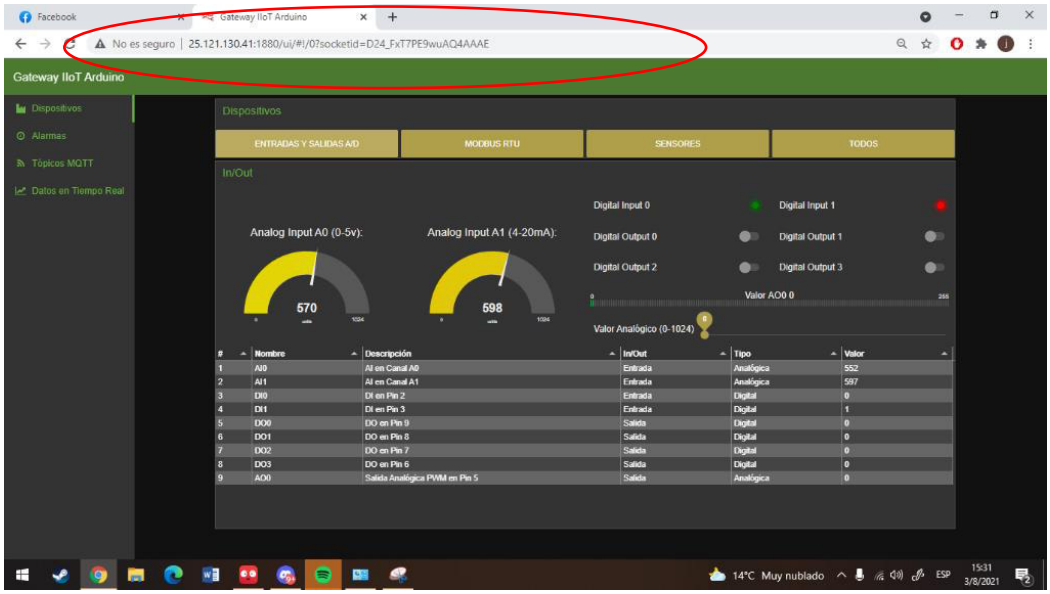


Figura G.12. Acceso remoto a la interfaz de monitoreo y control.