

DESARROLLO E IMPLEMENTACIÓN DE UNA RED DE MEDICIÓN DE PARÁMETROS AMBIENTALES EN RECINTOS DE UNA ORGANIZACIÓN APLICANDO TECNOLOGÍAS IOT.

Sergio F. Felissia¹, Jorge A. Bossio¹, Gastón C. Peretti¹, Rodolfo D. Podadera¹, Santiago Previotto¹

¹Grupo de Investigación y Desarrollo en Electrónica (GIDE), Departamento de Ingeniería Electrónica, Facultad Regional San Francisco, Universidad Tecnológica Nacional, Av. De la Universidad 501 San Francisco (Córdoba).
sfelissia@facultad.sanfrancisco.utn.edu.ar

Resumen

El presente trabajo describe el desarrollo de un nodo de medición de parámetros ambientales, específicamente, temperatura, humedad, presión atmosférica y concentración de CO₂ destinado a su utilización en aulas y otros recintos de acceso público. A partir de la pandemia de Covid-19, surgió la necesidad de evaluar la calidad del aire en los ambientes cerrados, procurando mejorar la ventilación, principalmente, cuando existe una cantidad importante de personas. Adicionalmente, la concentración de dióxido de carbono se controla mediante equipos de ventilación (S. Li, 2018).

El objetivo planteado en la ejecución del prototipo fue la construcción de un módulo embebido de medición con conectividad WiFi, incluyendo sensores adecuados para la medición de las variables mencionadas. La alimentación se realiza a través de una fuente switching similar a la de un cargador de dispositivos móviles. El proyecto se completa con el acceso a una red IoT y la posibilidad de mostrar a través de aplicaciones web, los valores actuales y el registro temporal de las variables ambientales, obtener gráficos históricos y otros procesamiento de la información almacenada en una base de datos en la nube.

Desde el punto de vista de hardware se utiliza un sistema embebido, con conectividad Wifi, a la cual se conectan los sensores. Los nodos se conectan a la red WiFi de la organización. Se ha creado una máquina virtual en los servidores de la institución, en la cual se instaló un servidor MQTT. Se creó una base de datos a la cual se accede a través de aplicaciones. Se crearon dashboard para mostrar los valores en tiempo real y realizar consultas del registro histórico de las variables visualizado a través de gráficos.

Se concluyó que el sistema puede tomar las mediciones, almacenarlas en una base de datos, mostrar los valores actuales y procesar la información.

Palabras Clave: Red IoT, protocolo MQTT, Concentración CO₂, Ventilación, Ambiente.

Introducción

Se describe en este trabajo el proceso de diseño de la red de medición y los nodos que se ubicarán en los espacios donde se requiere monitorear los parámetros ambientales de Temperatura, Humedad, presión atmosférica y concentración de CO₂. En este último caso, un parámetro importante para determinar el grado de ventilación de los ambientes, sobre todo ante la presencia de gran cantidad de personas. Se describe la selección de los componentes del ecosistema IOT y su fundamentación.

Para la implementación del acceso a una red, se utiliza la misma red WiFi de acceso libre, disponible en la institución. Se creó en un servidor externo contratado, una máquina virtual con recursos mínimos. En la Tabla 1 se puede observar que se trata de un servidor de un único núcleo, 1Gb de RAM, 10 Gb de disco de estado sólido donde se ha instalado una distribución del sistema operativo Linux.

Tabla 1: Recursos de hardware y software de la máquina virtual como servidor del sistema.

Hardware		Software	
Núcleos	1 (uno)	Sistema operativo	Ubuntu (Linux)
Memoria	1 Gb RAM	Versión	18.04
Almacenamiento	10 Gb SSD	Arquitectura	64 bits
Transferencia	1 TB	Aplicación	Instalación mínima

En el primer paso del despliegue de la red, se instaló el servidor MQTT. MQTT es un protocolo ligero de publicación y suscripción que se utiliza en Internet de las cosas. Su popularidad lleva a varias implementaciones en diferentes lenguajes (A. Al-Fuqaha, 2015).

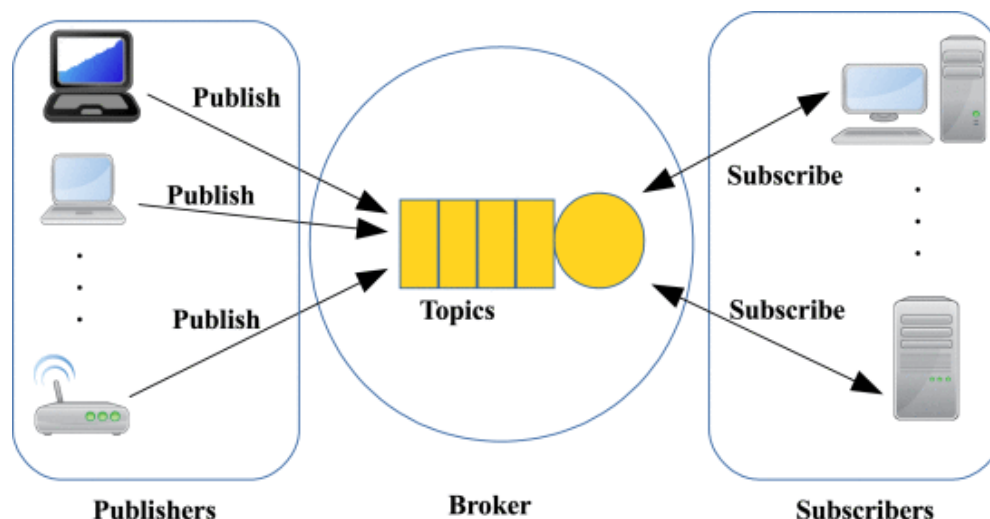


Figura 1: Estructura publicador suscriptor de MQTT (A. Al-Fuqaha, 2015).

Las implementaciones de MQTT de código abierto más populares son: Mosquitto, HiveMQ, EMQX, VerneMQ, MQTT.js y Paho (M. Bender, 2021). Para la selección de alguna de las plataformas de código abierto se analizaron algunos trabajos de evaluación de performance. La evaluación incluye escalabilidad, interoperabilidad, consumo de recursos y latencia. De acuerdo con los resultados presentados en estos trabajos, las principales implementaciones de clientes y servidores se ajustan a los requisitos básicos del estándar y, por lo tanto, pueden interoperar. El lenguaje de implementación es un factor crucial para el consumo de recursos y provoca diferencias considerables en la escalabilidad (EMQX, Página web: “Evaluation for popular online public MQTT broker”, 2022). Los resultados muestran que las latencias entre implementaciones y las pérdidas de paquetes difieren en los distintos sistemas.

Tabla 2: Latencia promedio en los distintos brokers MQTT de código abierto (M. Bender, 2021)

Name	Broker Address	TCP	TLS	WebSocket
EMQX (Global)	broker.emqx.io	1883	8883	8083,8084
EMQX (CN)	broker-cn.emqx.io	1883	8883	8083,8084
Eclipse	mqtt.eclipseprojects.io	1883	8883	80, 443
Mosquitto	test.mosquitto.org	1883	8883, 8884	80, 443
HiveMQ	broker.hivemq.com	1883	N/A	8000

A partir de estos análisis se instaló el servidor EMQX, que ofrece un dashboard de administración que se adecúa a las necesidades del proyecto. Se adoptó como software intermediario para recibir los datos transmitidos a través del protocolo MQTT y registrar la información recibida en una base de datos, la herramienta también de código abierto denominada Node-RED.

Node-RED es una plataforma de desarrollo de software gráfica que permite la programación en entorno web, a través de bloques, basada en el lenguaje Node.JS. Se ha extendido, de manera muy importante en el desarrollo de aplicaciones para IoT e IIOT (Node-RED, Página web: “Node-RED Cookbook”,2022).

Node-RED se usa para ser un suscriptor que se suscribe para recibir datos del broker MQTT (S. Chanthakit, 2018). Con Node-RED, podemos crear fácilmente un flujo para administrar y manejar los datos recibidos. Luego, Node-RED enviará datos al tablero de monitoreo, que es una aplicación web receptiva para mostrar datos en la interfaz de usuario de indicadores, texto y gráficos (S. Chanthakit, 2018)). Además, Node-RED se encarga de registrar la información de las variables relevadas en una base de datos MariaDB (SQL). El esquema del software instalado en el servidor virtual se muestra en la Figura 2.

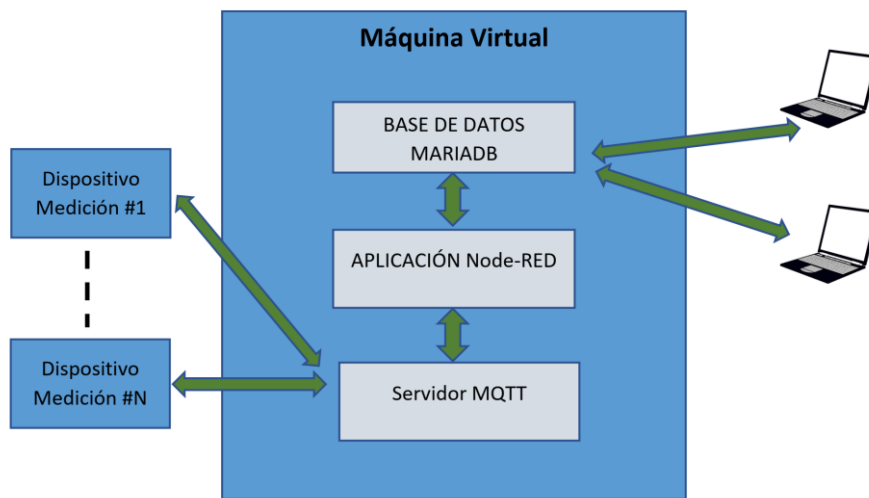


Figura 2: Esquema de software instalado en el servidor virtual.

Diseño del Sistema:

El sistema de monitoreo de parámetros ambientales consta de tres componentes principales (S. Chanthakit, 2018) que se muestran en la Figura 3 .

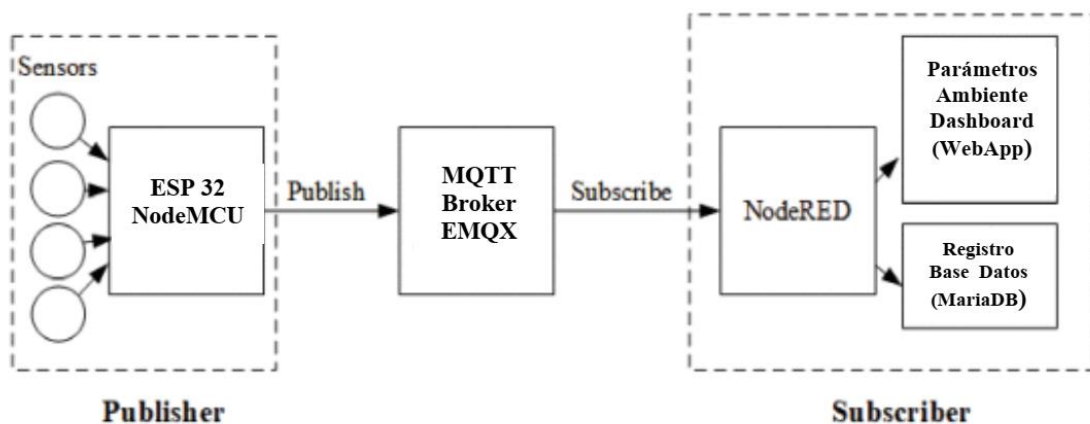


Figura 3. Arquitectura del Sistema de Monitoreo de la Calidad del Aire (S. Chanthakit, 2018).

El primer componente es el dispositivo nodo de medición. El nodo de medición de los parámetros ambientales es un hardware que utiliza ESP32 como sistema embebido que se conecta a

sensores para medir temperatura, humedad, presión atmosférica y concentración de monóxido de carbono. El firmware a bordo hace que el dispositivo actúe como un publicador que lee los datos del sensor y los envía al bróker MQTT. Se conectan sensores al ESP32 que recopila, analiza y transforma los datos de los sensores y luego los publica en un tópico previamente definido.

El segundo componente es el broker MQTT. Recibe los paquetes de datos publicados por los dispositivos y los distribuye a los suscriptores a los tópicos en donde se han publicado los datos.

El tercer componente es un panel de monitoreo de los parámetros ambientales y un servicio de registro de la información de los sensores, en una base de datos. Se utiliza una herramienta de programación: Node-RED, que actúa como suscriptor para recibir datos desde el broker MQTT. Se diseñaron flujos (bloques de software de la plataforma Node-RED) para administrar y manejar los datos recibidos y mostrarlos en forma de indicador, texto y gráfico en el tablero. Adicionalmente, otros flujos se encargan del almacenamiento de los datos en una base de datos Node-RED, (Página web: “Node-RED Cookbook”, 2022).

Dispositivo nodo de medición de parámetros ambientales.

La implementación del dispositivo de medición de parámetros ambientales se puede dividir en dos partes: hardware y software.

1) Hardware

El diagrama en bloques del hardware del dispositivo se muestra en la Figura 4. El principal componente de hardware del dispositivo es una placa ESP32 NodeMCU que se conecta a los sensores (Espressif, “ESP32 WROOM32E Datasheet”, 2022). En la Figura 5, se observa el prototipo experimental implementado y se indica la ubicación de los distintos componentes.

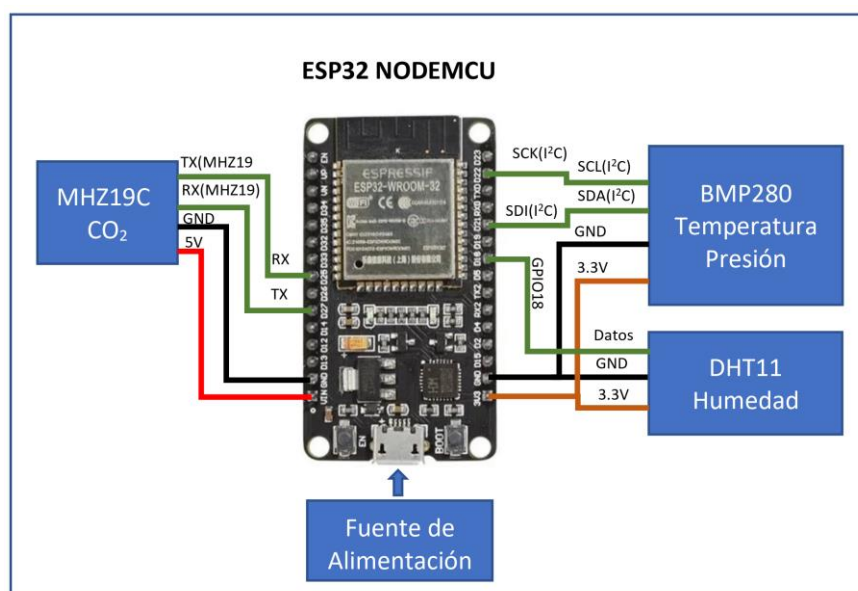


Figura 4: Estructura del hardware implementado en el dispositivo de medición de parámetros ambientales.

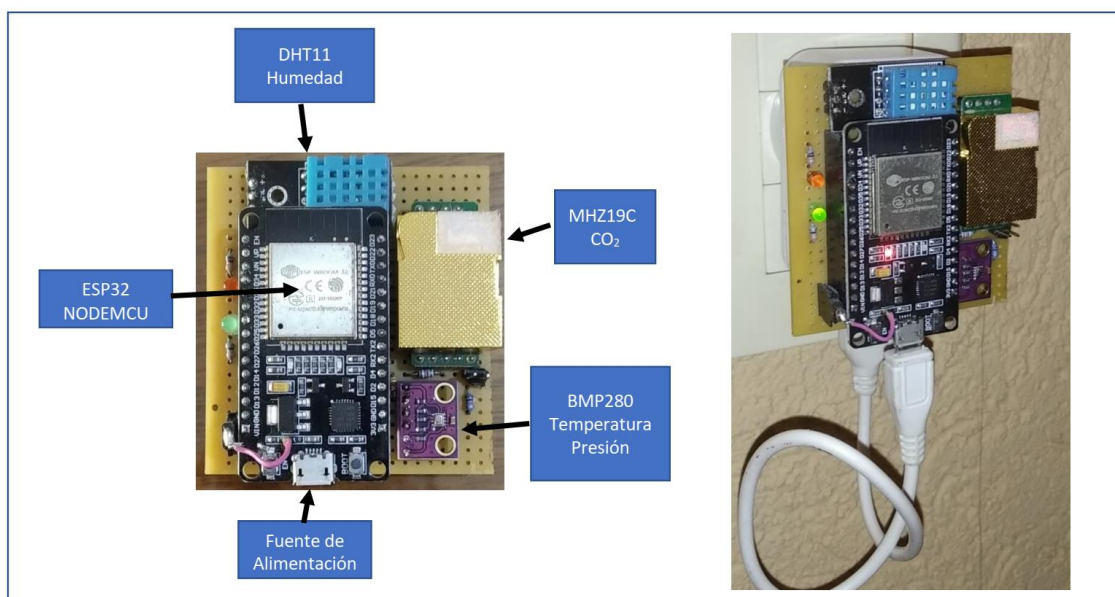


Figura 5: Prototipo del dispositivo de medición de parámetros ambientales.

En este dispositivo, se han instalado tres circuitos integrados que relevan cuatro variables.

- Sensor de humedad (DHT11), un sensor digital de temperatura y humedad de bajo costo. Utiliza un método de medición capacitivo y un termistor para medir el aire circundante y emite una señal digital en el pin de datos.
- Sensor temperatura y presión atmosférica, basado en el circuito integrado BMP280. El BMP280 es un termómetro y barómetro digital del fabricante Bosch Sensortech, que permite realizar lecturas de temperatura, presión atmosférica, y estimación de altitud sobre el nivel del mar. Para temperatura el rango es de -40° a 85°C , con una precisión de $\pm 1.0^{\circ}\text{C}$. Para presión atmosférica / altímetro es de 300hPa a 1110 hPa, equivalente a una altitud de -500m a 9000m sobre el nivel del mar. Con una precisión absoluta de $\pm 1.0^{\circ}\text{C}$ y de 1.0 hPa, y la relativa de 0.12 hPa, equivalente a una precisión en altitud de aproximadamente $\pm 1\text{m}$. Basado en tecnología BOSCH piezo-resistiva con gran robustez EMC, alta precisión y linealidad, así como con estabilidad a largo plazo. Se ha diseñado para ser conectado directamente a un microcontrolador a través de I2C o SPI.
- El módulo de sensor de CO_2 infrarrojo MH-Z19C es un sensor de tamaño pequeño de tipo común que utiliza el principio de infrarrojo no dispersivo (NDIR) para detectar la existencia de CO_2 en el aire, con buena selectividad, no dependiente de oxígeno y de larga duración. Tiene compensación de temperatura incorporada; y tiene salida UART y salida PWM. Tiene un rango de 400 ppm a 2000 ppm en un modelo y hasta 5000 ppm en otro modelo (Tabla 3).

Tabla 3: Rango de detección y precisión del sensor CO_2 MH-Z19C

Gas de detección	Fórmula	Rango de detección	Precisión
Dióxido de carbono	CO_2	400-2000 ppm	$\pm(50\text{ppm}+5\% \text{ valor de lectura})$
		400-5000 ppm	

Este módulo tiene dos métodos para la calibración del punto cero: método manual y auto-calibración. Todo el punto cero está a 400 ppm de CO2 (Winsen, "Infrared CO2 Sensor Module (Model: MH-Z19C) User's Manual", 2022).

Método manual:

Se conecta el pin HD del módulo a un nivel bajo (0V), con una duración mínima de 7 segundos. Antes de calibrar el punto cero, es necesario asegurar que el sensor esté estable durante más de 20 minutos en un entorno ambiental de 400 ppm.

Método de Autocalibración:

La función de autocalibración significa que después de que el sensor funciona de forma continua durante un período de tiempo, puede determinar de forma inteligente el punto cero de acuerdo con la concentración ambiental y calibrarse a sí mismo. El ciclo de calibración es una calibración automática cada 24 horas desde el encendido. El punto cero de la calibración automática es de 400 ppm.

2) Software

Firmware del dispositivo

Para la programación del firmware del ESP32 a bordo del dispositivo sensor de parámetros ambientales, se realizó utilizando Arduino IDE.

El software que se implementó en el dispositivo recopila los datos de las cuatro variables a través de los módulos sensores. Luego, se utiliza una librería de cliente MQTT para publicar datos en el broker.

Las librerías que suministran los fabricantes de los integrados, para la adquisición y el procesamiento de los datos digitales que envían los sensores, acondicionan las mediciones a los valores y medidas estándar, utilizando las funciones de lectura de los objetos correspondientes a los datos de temperatura, humedad y presión atmosférica son mostradas en el código fuente.

```
#include <Adafruit_BMP280.h>
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN 18           // Digital pin connected to the DHT sensor
#define DHTTYPE DHT11      // DHT 11
Adafruit_BMP280 bmp;       // use I2C interface
Adafruit_Sensor *bmp_temp = bmp.getTemperatureSensor();
Adafruit_Sensor *bmp_pressure = bmp.getPressureSensor();
DHT_Unified dht(DHTPIN, DHTTYPE);

-----
-----
void loop(){
-----
-----
//Lectura de los valores de Temperatura, Presión atmosférica y Humedad
sensors_event_t temp_event, pressure_event;
bmp_temp->getEvent(&temp_event);
bmp_pressure->getEvent(&pressure_event);
Temperatura=temp_event.temperature;
Presion=pressure_event.pressure;ht.humidity().getEvent(&event);
Humedad=event.relative_humidity;
// *****
```

Para la temperatura y la humedad, el sensor DHT11 da el valor de la temperatura en grados centígrados(°C) y el valor de la humedad en humedad relativa (% HR). En este caso solo se toma la humedad relativa ya que la temperatura y presión atmosférica se obtienen a través del sensor integrado BMP280. Este último, a través de su librería, da la temperatura en °C y la presión en hPa (hectopascal).

Para la medición de concentración de CO₂ se utiliza el sensor MHZ19 de la empresa Winsen. La interfase con el módulo ESP32 se realiza a través de un puerto serie, enviando comandos para realizar las lecturas las funciones de calibración y la configuración. La librería abstrae todas estas tareas de modo que para el programador es transparente la lectura del valor medido de concentración de CO₂, de acuerdo con la siguiente sección del código fuente.

```
#include "MHZ19.h"
#include <SoftwareSerial.h>

#define RX_PIN 25           // Rx pin which the MHZ19 Tx pin is attached to
#define TX_PIN 27           // Tx pin which the MHZ19 Rx pin is attached to
#define BAUDRATE 9600       // Device to MH-Z19 Serial baudrate

MHZ19 myMHZ19;              // Constructor for library
SoftwareSerial mySerial(RX_PIN, TX_PIN); // Create device to MH-Z19 serial

void loop(){

-----
-----
if (millis() - getDataTimer >= 5000) // Se toma una muestra cada 5 segundos.
{
    CO2 = myMHZ19.getCO2();          // Request CO2 (as ppm)
    getDataTimer = millis();
}
```

Se configura la conexión de red utilizando las funciones de la librería WiFi. Queda configurada la función callback que atenderá las solicitudes de conexión. En este caso, el argumento de setServer es la dirección IP de la máquina virtual creada. Para el proceso publicación/suscripción MQTT se utiliza el puerto 1883.

```
void setup(){
// Conecta al WiFi
  setup_wifi();
  client.setServer(MQTT_SERVER, 1883);
  client.setCallback(callback);
  digitalWrite(LED_CONEXION, HIGH);
//*****
```

Se diseñó una cadena con formato JSON para el payload enviado.

```
char buffer[80]=" ";
char formato[80]="{\"CO2\": %d, \"Temperatura\": %s, \"Presion\": %s, \"Humedad\": %s }";
//Payload del dato publicado en un tópico.
sprintf(buffer, formato, CO2, strTemp, strPresion, strHum);
```

JSON (JavaScript Object Notation) es un formato ligero de intercambio de datos. JSON es de fácil lectura y escritura para los usuarios. JSON es fácil de analizar y generar por parte de las máquinas. JSON se basa en un subconjunto del lenguaje de programación JavaScript. Es un formato de texto completamente independiente del lenguaje, pero que utiliza convenios que resultan familiares a los programadores de lenguajes de programación como C, C++, C#, Java™, JavaScript, Perl, Python y mucho otros. Estas características hacen de JSON un lenguaje de intercambio de datos ideal.

JSON se basa en dos estructuras:

- Una colección de pares de nombre/valor.
- Una lista ordenada de valores.

Estas son estructuras de datos universales. Prácticamente todos los lenguajes de programación modernos los admiten de una forma u otra. Tiene sentido que un formato de datos que sea intercambiable con los lenguajes de programación también se base en estas estructuras.

En JSON, estas estructuras toman las formas de objetos, arreglos, valores, cadenas, números y espacios en blanco (JSON.ORG, Página web: "Introducing JSON", 2022).

En este proyecto se utiliza una estructura de objeto JSON (Figura 6). Un objeto es un conjunto desordenado de pares nombre/valor. Un objeto comienza con { (*llave izquierda*) y termina con } (*llave derecha*). Cada nombre va seguido de : (*dos puntos*) y los pares de nombre/valor están separados por , (*coma*).

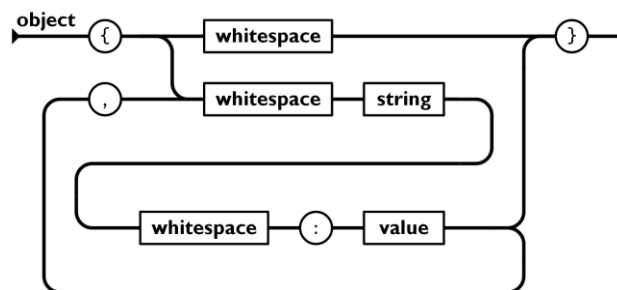


Figura 6: Objeto JSON.

El formato del payload utilizado se conforma de acuerdo con lo que se observa en la Figura 7.

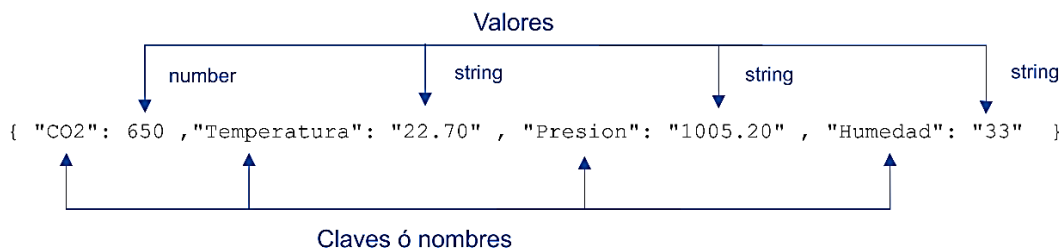


Figura 7: Ejemplo de cadena en el formato JSON.

Servidor virtual. Software en el Servidor

Los paquetes de software instalados en la máquina virtual que actúa como servidor son:

- EMQX: Servidor de MQTT.
- Node-RED: Entorno de programación basado en el framework de Node.JS para realizar aplicaciones que se ejecutan en servidor tanto como servicios, como interfaz de usuario ejecutable en navegador (paneles de monitoreo).
- XAMPP: paquete del servidor web Apache que incluye servidor de base de datos SQL(MariaDB) y phpMyAdmin.

Se configura Node-RED para que se ejecute como servicio, en segundo plano, de modo que permanentemente realice las tareas de conexión al servidor MQTT, obtenga el paquete de datos enviado por el dispositivo, separe los valores de las variables leídas, guarde en una base de datos esos valores y presente los datos en una pantalla de monitoreo (dashboard) (Figura 8):

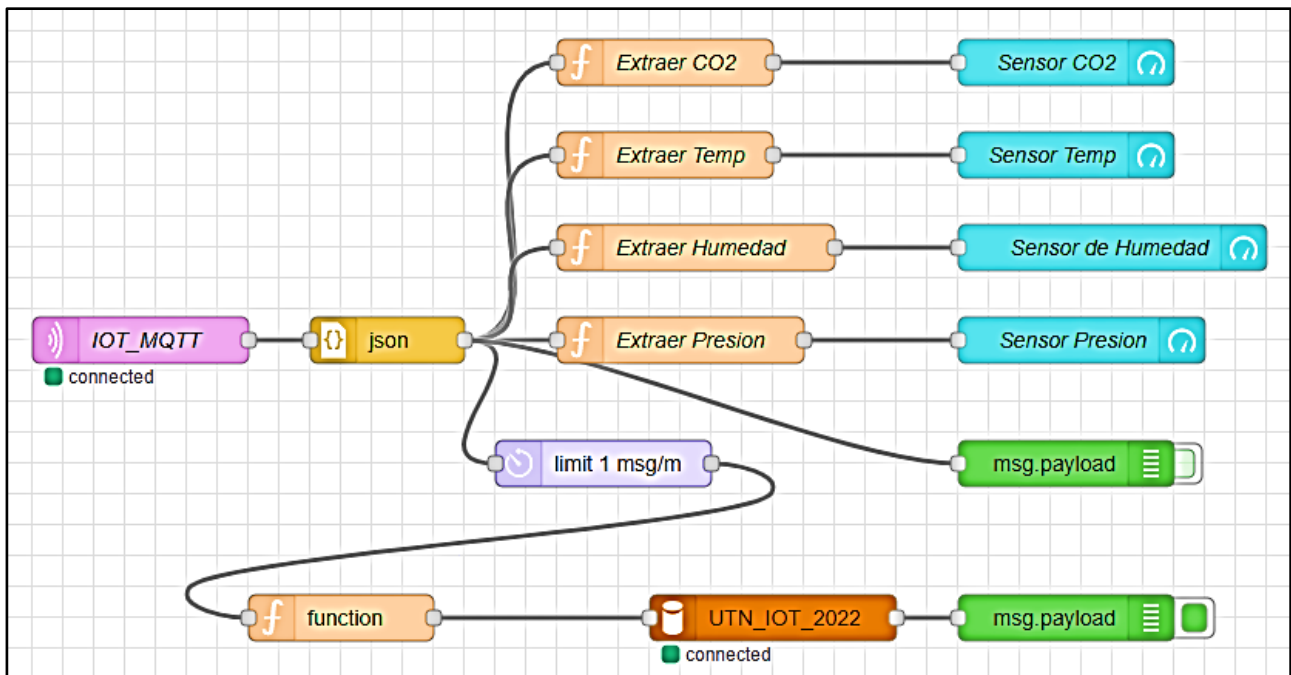


Figura 8: Diagrama de flujos de Node-RED para guardar los valores de las variables en la base de datos UTN_UOT_2022 y cada 1 minuto y mostrar valores en un dashboard.

A través de una operación de inserción en la base de datos, cuyo código mostrado a continuación, se encuentra en el bloque “function” que se conecta al bloque de base de datos, se almacenan los valores de temperatura, humedad, presión atmosférica y concentración de CO₂ en la tabla “Datos sensores” de la base de datos “UTN_IOT_2022”.

```

var presion=msg.payload.Presion;
var temp=msg.payload.Temperatura;
var humedad=msg.payload.Humedad;
var CO2=msg.payload.CO2;
msg.topic="INSERT INTO `Datos sensores`(Temperatura, Presión, Humedad, CO2,
Dispositivo)VALUES ('"+temp+"', '"+presion+"', '"+humedad+"', '"+CO2+"', 'SANFC01')";
return msg;

```

Resultados y discusiones

En la Figura 9, se puede observar un panel de visualización básico en tiempo real de las variables medidas, al que se puede acceder a través de cualquier navegador. Se agregan dos botones de comando para realizar funciones de activación/desactivación de cualquier parte de hardware. Estas funciones se manejan también por mensajes de publicación en tópicos del protocolo MQTT, en este caso desde la aplicación hacia el dispositivo. En este último, en el software, se escribe el código que incluye la suscripción a ese tópico, recibiendo los mensajes de activación y de desactivación.

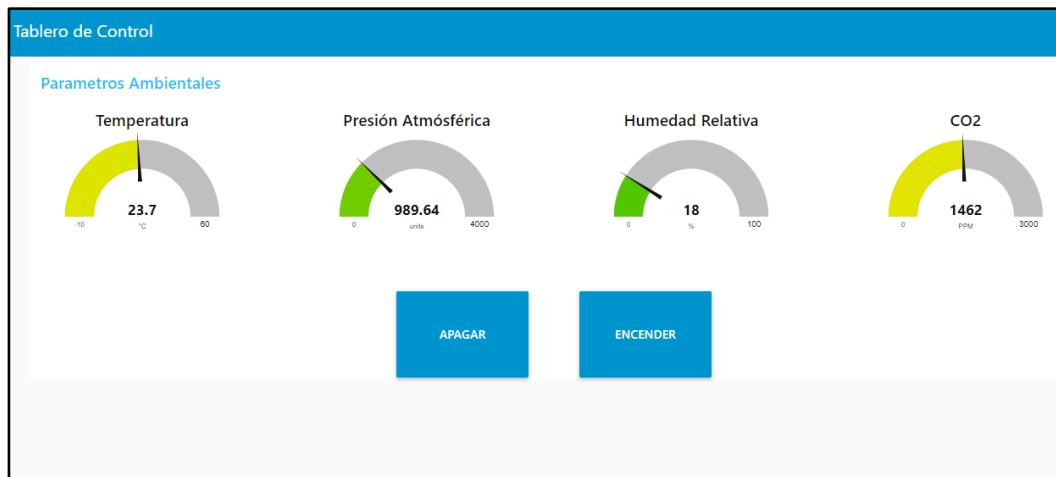


Figura 9: Dashboard básico de monitoreo en tiempo real de los parámetros ambientales.

En la Figura 10: se observa los registros obtenidos de las variables y almacenados en la base de datos UTN_IOT_2022, y en la tabla Datos_sensores. Visualización a través del panel de phpMyAdmin.

Servidor: localhost »		Base de datos: UTN_IOT_2022 »		Tabla: Datos sensores			
Examinar	Estructura	SQL	Buscar	Insertar	Exportar	Importar	
Id	Dispositivo	Humedad	fecha	Temperatura	Presión	CO2	Otros
351	SANFCO1	14	2022-08-29 15:53:26.020888	25.04	1015.08	817	NULL
352	SANFCO1	14	2022-08-29 15:54:26.362968	25.03	1015.06	815	NULL
353	SANFCO1	14	2022-08-29 15:55:26.665861	24.98	1015.04	809	NULL
354	SANFCO1	15	2022-08-29 15:56:27.010341	24.96	1015.04	807	NULL
355	SANFCO1	15	2022-08-29 15:57:27.335949	24.96	1015.05	807	NULL
356	SANFCO1	14	2022-08-29 15:58:27.690013	25.02	1015.05	812	NULL
357	SANFCO1	14	2022-08-29 15:59:28.007384	25	1015.04	818	NULL
358	SANFCO1	14	2022-08-29 16:00:28.339632	25.03	1015.04	822	NULL
359	SANFCO1	14	2022-08-29 16:01:28.661306	24.97	1014.97	828	NULL
360	SANFCO1	14	2022-08-29 16:02:29.001971	24.95	1014.98	839	NULL
361	SANFCO1	15	2022-08-29 16:03:29.323857	24.96	1014.95	842	NULL

Figura 10: Dashboard básico de monitoreo en tiempo real de los parámetros ambientales.

Conclusiones

Se concluyó que el sistema opera de la manera esperada, logrando una permanente captura de los datos que envía el nodo prototipo y los almacena en la base de datos creada. Los datos obtenidos durante períodos largos de tiempo pueden ser utilizados para el estudio de distintas condiciones que se pueden producir en los ambientes, en relación con la cantidad de personas que ocupan un determinado espacio, temperaturas, presión, humedad, ventilación de los ambientes, fijar alarmas en los momentos que la concentración de CO₂ se eleva por encima de niveles perjudiciales para la salud. Por el momento, este trabajo se enfocó en la implementación de un nodo prototipo que puede ser extendido a una gran cantidad de nodos de medición distribuidos en la mayor cantidad de espacios, principalmente los de acceso público.

Es de vital importancia realizar la calibración del medidor de concentración de CO₂ y analizar la metodología de calibración que actualmente es manual y luego automática. La calibración manual, implica retirar el dispositivo nodo de medición del lugar donde está instalado, llevarlo a un espacio al aire libre, supuesto libre de contaminación atmosférica, para que tome esa concentración como 400 ppm. Se han utilizado, distintos componentes del ecosistema IoT, tanto de hardware como de software. Podemos mencionar, sistemas embebidos, conectividad inalámbrica, protocolo MQTT, software de base de datos, máquinas virtuales. Para la lectura y procesamiento de los datos

registrados se realizó una programación a través de Node-RED que permite hacer consultas a la base de datos sobre las claves almacenadas, en períodos determinados y posteriormente hacer análisis gráficos de la información obtenida. Esta aplicación se describe en otros trabajos y por su extensión no se incorporó a este documento.

Referencias:

S. Chanthakit and C. Rattanapoka, "MQTT Based Air Quality Monitoring System using Node MCU and Node-RED," *2018 Seventh ICT International Student Project Conference (ICT-ISPC)*, 2018, pp. 1-5, doi: 10.1109/ICT-ISPC.2018.8523891.

M. Bender, E. Kirdan, M. -O. Pahl and G. Carle, "Open-Source MQTT Evaluation," *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*, 2021, pp. 1-4, doi: 10.1109/CCNC49032.2021.9369499.

A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," in *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347-2376, Fourthquarter 2015, doi: 10.1109/COMST.2015.2444095.

S. Li, "Research on Intelligent Monitoring System of Greenhouse Intensity and CO2 Concentration Based on STM32," *2018 IEEE International Conference on Mechatronics and Automation (ICMA)*, 2018, pp. 666-670, doi: 10.1109/ICMA.2018.8484450.

Node-RED, Página web: "Node-RED Cookbook", Accedido: 05/09/2022. Recuperado de: <https://cookbook.nodered.org/#mqtt>.

EMQX, Página web: "Evaluation for popular online public MQTT broker", Accedido 07/09/2022. Recuperado de: <https://www.emqx.com/en/blog/popular-online-public-mqtt-brokers>.

Winsen, "Infrared CO2 Sensor Module (Model: MH-Z19C) User's Manual", Accedido 07/09/2022. Recuperado de: https://www.winsen-sensor.com/d/files/infrared-gas-sensor/mh-z19c-pins-type-co2-manual-ver1_0.pdf

Espressif, "ESP32 WROOM32E Datasheet", Accedido 07/09/2022. Recuperado de: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf.

JSON.ORG, Página web: "Introducing JSON", Accedido 12/09/2022. Recuperado de: <https://www.json.org/json-en.html>