VOLUMEN 16
XXVII Verano De la Ciencia
ISSN 2395-9797
www. jóvenesenlaciencia.ugto.mx

Implementación de sistema de control IoT basado en Edge Computing

Rosa Ivonne Peñaran Prieto [1]
Francisco Javier Moreno Vazquez [2]
Juan Manuel Cano Gallardo [3]
Alejandro Pizano Martínez[4]
Luis Ramon Merchan Villalba[5]
[1]Departamento de Ingeniería Comunicaciones y electrónica Universidad de Guanajuato
ri.penaranprieto@ugto.mx[1]
[2]Departamento de Ingeniería Mecatrónica Universidad de Guanajuato
fj.morenovazquez@ugto.mx[2]
[3]Departamento de Ingeniería Eléctrica Universidad de Guanajuato
jm.canogallardo@ugto.mx[3]
[4]Departamento de Ingeniería Eléctrica Universidad de Guanajuato
apizano@ugto.mx[4]
[5]Departamento de Ingeniería Eléctrica Universidad de Guanajuato
Lr.merchan@ugto.mx[5]

Resumen

Mediante el desarrollo de este proyecto se plantea la implementación de sistema de control IoT (*Internet of Things*) basado en cómputo en el borde (*Edge Computing*). A través de este sistema de busca establecer el control de diversos dispositivos IoT sin la necesidad de acceder a servicios de cómputo en la nube (*Cloud Computing*). De esta forma se podrán controlar algunos dispositivos en el entorno de una red de área local wifi bajo el manejo dado por un sistema informático ejecutándose desde un equipo de cómputo de bajo presupuesto (Raspberry Pi), donde se contempla el uso del protocolo MQTT y un servidor HTTP para permitir a los usuarios acceder al control de los dispositivos de forma sencilla.

Palabras clave: Edge Computing, IoT, Área local.

Introducción

Con el pasar de las décadas, la humanidad ha modificado la manera sustancial en que se comunica. Hace millones de años los humanos se comunicaban con gestos y señas, hoy en día gracias a los avances tecnológicos es posible comunicarse por medio de envío de datos a través de una red global la cual permite enviar y recibir información desde el otro lado del mundo en cuestión de segundos. Cada día se utilizan más y más dispositivos electrónicos los cuales tienen la capacidad de enviar y recibir datos de la red. En cuanto a los usuarios de internet en dispositivos móviles, en enero de 2022 alcanzaron al 67,1% de la población, es decir, 5,310 millones de personas (Galeano, 2022). Según el portal DOMO en un minuto en internet se puede dar lo siguiente (ADN40, 2022):

- En Twitter se pueden generar hasta 575 mil tuits por minuto
- En Facebook se pueden generar más de 44 millones de vistas en videos y posteos
- En TikTok se pueden reproducir 167 millones de videos
- En YouTube se transmiten hasta 694 mil vídeos
- En Instagram los usuarios comparten hasta 65 mil fotos por cada 60 segundos

Para llevar a cabo el procesamiento de los datos en la red y otras acciones se pueden identificar 3 tipos de labores de cómputo: en la nube (*Cloud*), en la niebla (*Fog*) y en el borde (*Edge*) (ver Figura 1.), los cuales se describen a continuación:

- *Cloud computing*: En este tipo de procesamiento de datos se envía la información a un servidor remoto, el cual puede verse como un servidor principal, que mediante software o algoritmos

complejos procesa la información que recibe. Este servicio lo usan generalmente las grandes compañías informáticas que ofrecen servicio web cómo Facebook, Google, Twitter entre otras.

- *Fog computing*: A diferencia del *cloud computing*, este tipo de procesamiento no envía a los datos a un servidor principal, sino que envía los datos a servidores secundarios a una distancia menor a la que está el servidor de *cloud computing*, teniendo así un sistema de cómputo descentralizado con una respuesta más rápida.
- *Edge computing*: A diferencia de los dos anteriores, el *edge computing* no necesita enviar datos a un servidor de grandes prestaciones ya que el procesamiento de datos se lleva a cabo en una red local la cual no tiene que estar necesariamente conectada a internet lo cual abre una oportunidad a trabajar con dispositivos inalámbricos sin la necesidad de una conexión a la red global (Rodal, 2021).

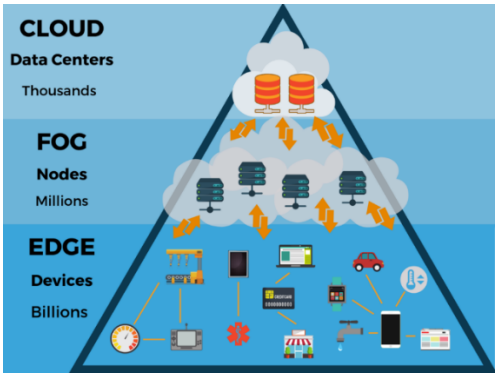


Figura 1. Cloud computing, Fog computing y Edge computing

Por otro lado, se tiene otro conjunto de aplicaciones basados en el internet llamada internet de las cosas, conocido también como IoT por sus siglas en inglés *Internet of Things*. IoT es la interconexión de los objetos del mundo físico a través de internet, estando estos equipados con sensores, actuadores y tecnología de comunicación. La versatilidad que ofrece esta tecnología le permite ubicarse en distintos rubros de la vida diaria. El IoT cada vez es más utilizado en sectores como la salud, la energía e industria dando pie a lo que se conoce como Industria 4.0. Se está en la era de la interconexión de las cosas u objetos, esto permite crear un ambiente de mayor confort para los usuarios, así como, mejorar el procesamiento de los datos obtenidos a través de sensores lo cual permite, según la aplicación, tener un mayor control sobre determinados dispositivos. Dentro del IoT hay tres componentes o elementos básicos que interactúan entre sí, por un lado se tiene el hardware, como sensores, actuadores (dispositivos que controlan los sistemas) y otros dispositivos de comunicación alojados en los objetos; también se tiene la plataforma de middleware, que es el software que permite el intercambio de información entre las aplicaciones, así como las herramientas computacionales que permitan el análisis de datos; finalmente se tienen las herramientas que en forma fácil permiten la visualización e interpretación de la información y que deben ser diseñadas para permitir el acceso de diferentes aplicaciones y dispositivos.

Dentro de la componente de hardware Arduino es una plataforma electrónica abierta utilizada para la creación de prototipos (Arduino, 2022). Está basada en software y hardware flexibles, que han sido diseñados buscando aproximar la programación y la robótica a usuarios no expertos, es decir, su entorno es “fácil” de usar (Isaias, 2016).

Adicionalmente se tienen protocolos de comunicación especialmente diseñados para aplicaciones IoT, entre ellos el MQTT. El protocolo MQTT se ha convertido en uno de los principales pilares del IoT por su sencillez y ligereza. Ambos son condicionantes importantes dado que los dispositivos de IoT, a menudo, tienen limitaciones de potencia, consumo, y ancho de banda. MQTT son las siglas *Message Queing Telemetry Transport*. Es un protocolo de comunicación M2M (machine-to-machine), está basado en la pila TCP/IP como base para la comunicación. En el caso de MQTT cada conexión se mantiene abierta y se "reutiliza" en cada comunicación. Es una diferencia, por ejemplo, a una petición HTTP 1.0 donde cada transmisión se realiza a través de conexión. (Luis Llamas, 2019)

En este proyecto se busca desarrollar una plataforma *Edge Computing* de bajo costo la cual dispone del control de dispositivos IOT mediante el protocolo de comunicación MQTT, la cual podrá proporcionar la información al usuario a través de una interfaz en una página web. La información la generan los dispositivos conectados a la red local a través de los sensores u actuadores conectados en ellos. Para llevar esto a cabo,

se utiliza un ordenador de placa reducida de bajo costo “Raspberry Pi” la cual funge como bróker de la plataforma *Edge computing*, a través de esta, fluyen los datos de cliente a cliente como se muestra en la Figura 2. Los clientes de este bróker son 3 tarjetas de desarrollo ESP32, estas últimas tienen conectados dispositivos de los que obtienen información o pueden cambiar el estado de actuadores según la información que será recibida del bróker además de un cliente que se ejecuta en la página web.

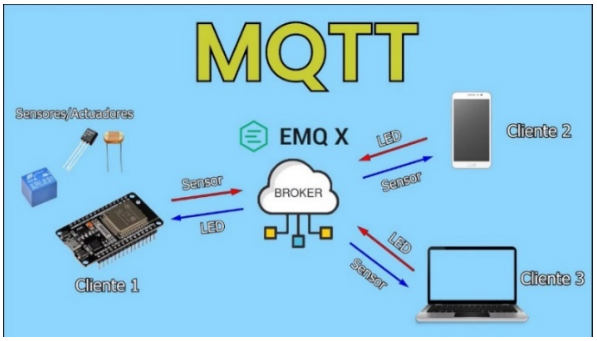


Figura 2. Flujo de datos del protocolo MQTT

Gracias a que el protocolo de comunicación MQTT no exige que los dispositivos utilizados, como la tarjeta Raspberry Pi, tengan una gran capacidad de procesamiento es posible desarrollar este proyecto con los elementos antes mencionados ya que brindan la capacidad necesaria para realizar las tareas que conllevan la comunicación MQTT lo que hacen de esta plataforma de *Edge computing* una opción viable para el control de los dispositivos y procesamiento de la información.

Plataforma Edge Computing

La plataforma *Edge Computing* abordada en este trabajo corresponde a un sistema con la capacidad de manejar algunos dispositivos IoT dentro del entorno de una red de área local. Esta red corresponde a una conexión WIFI de 2.4 Ghz, banda que es soportada por la mayoría de los dispositivos IoT basados en microcontroladores. Los dispositivos por considerar son tarjetas de desarrollo que incorporan el microcontrolador ESP32 a los cuales se les han adaptado actuadores y sensores. Además, estos dispositivos van a poder ser manejados por un usuario a nivel local accediendo a una página web con una interfaz gráfica adecuada para poder monitorizar y controlar los dispositivos. Finalmente, la plataforma se va a basar en el protocolo MQTT para el flujo de la información. Para esto se va a utilizar un bróker MQTT que se va a ejecutar localmente dentro de la Raspberry PI, misma que va a ejecutar el servidor http que servirá la página web. Estos elementos se describen a continuación.

Broker MQTT

El bróker también conocido como “servidor” es el software encargado de la gestión de los mensajes, su función principal es la de distribuir los mensajes entre los dispositivos conectados a él. El funcionamiento del MQTT es un servicio de mensajería con patrón publicador/suscriptor(pub-sub), por lo que este tipo de infraestructuras los clientes deben conectarse con un servidor central conocido como bróker.

Para filtrar los mensajes que son enviados a cada cliente los mensajes se disponen en tópicos organizados jerárquicamente. Un cliente puede publicar un mensaje en un determinado tópico. Otros clientes pueden suscribirse a este tópico, y el bróker le hará llegar los mensajes suscritos. Los clientes inician una conexión TCP/IP con el bróker, el cual mantiene un registro de los clientes conectados. Esta conexión se mantiene abierta hasta que el cliente la finaliza. Por defecto, MQTT emplea el puerto 1883 y el 8883 cuando funciona sobre TLS. (Luis Llamas, 2019). Uno de los componentes más importantes del protocolo MQTT es la definición y tipología de los mensajes, ya que son una de las bases de la agilidad en la que radica su fortaleza. Cada mensaje consta de 3 partes (ver Figura 3.):

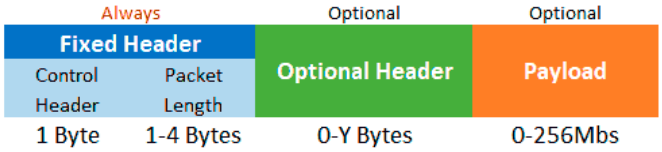


Figura 3. Estructura de un mensaje enviado en el protocolo MQTT

- Cabecera fija. Ocupa 2 a 5 bytes, obligatorio. Consta de un código de control, que identifica el tipo de mensaje enviado, y de la longitud del mensaje. La longitud se codifica en 1 a 4 bytes, de los cuales se emplean los 7 primeros bits, y el último es un bit de continuidad.
- Cabecera variable. Opcional, contiene información adicional que es necesaria en ciertos mensajes o situaciones.
- Contenido(payload). Es el contenido real del mensaje. Puede tener un máximo de 256 Mbs aunque en implementaciones reales el máximo es de 2 a 4 kB.

Dentro de este proyecto, se utilizará de manera local una Raspberry Pi como servidor bróker para que los dispositivos se conecten a ella y de esa forma controlar los eventos y las acciones que se produzcan en ellos.

Para el desarrollo de este trabajo se utilizó el bróker MQTT EMQX (EMQX, 2022) ejecutándose dentro de un contenedor de Docker (Docker, 2022) mediante Docker Compose. El bróker EMQX MQTT es un bróker de mensajería MQTT, tratándose de un código total y parcialmente abierto, altamente escalable y de alta disponibilidad para todo tipo de aplicaciones IoT, M2M y móviles que manejan variados clientes simultáneos.

Se utilizo Docker ya que este es una plataforma abierta para realizar diversas actividades como, por ejemplo, crear, implementar, ejecutar y compartir aplicaciones en contenedores. Un contenedor prácticamente es un entorno aislado, y como resultado del aislamiento y seguridad que existe, Docker nos entrega su software inmediatamente teniendo requerimientos mínimos. Los beneficios más importantes a la hora de usar Docker es que acelera básicamente el proceso de desarrollo. La mayor velocidad se debe a que puede separar las aplicaciones de la infraestructura, además permite migrar un desarrollo fácilmente a otro equipo de una forma sencilla de una forma compacta. Además, se tiene Docker-Compose que es una herramienta para definir y ejecutar aplicaciones Docker de varios contenedores. En comparación con las máquinas virtuales, los contenedores Docker son livianos, lo cual es vital para la Raspberry Pi que se está utilizando.

Servidor HTTP local

En cuanto a la parte de la comunicación entre el usuario y los dispositivos, al estar utilizando una red local, la forma más intuitiva para poder lograr esta comunicación es a través de una página web. Esta página web desarrollada podría ser visualizada por todos los usuarios que estén conectados en la red local, la cual estaría alojada en un servidor http simple corriendo a través de Python por medio de la Raspberry Pi.

Para este proyecto se utilizó la jerarquía de tópicos mostrada en la Figura 4, en la que se puede observar que los dispositivos que pueden cambiar su estado, de encendido a apagado y viceversa, tienen un tópico de SET y de STATUS, el primero permite enviar la señal de cambiar el estado del dispositivo mientras que el segundo devuelve el estado del dispositivo una vez que se haya realizado el cambio.

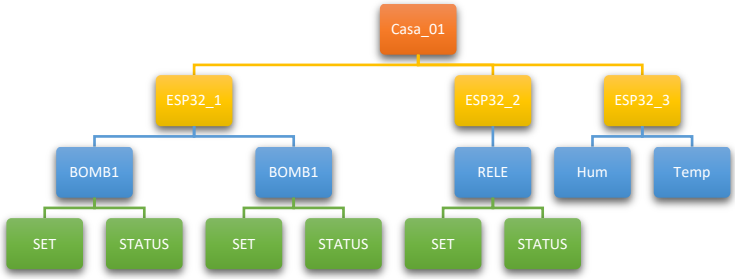


Figura 4. Jerarquía de tópicos MQTT

En la Figura 5. Se muestran los dispositivos que operan como clientes MQTT entre los cuales existe un canal de comunicación, como puede observarse no existe una conexión directa entre la página web y las tarjetas ESP32, es necesario que antes pase la información a través del bróker. La Figura 6. muestra el flujo de datos entre los elementos de la plataforma *Edge computing*, además, se indican cuáles son emisores y cuáles son los receptores de información.

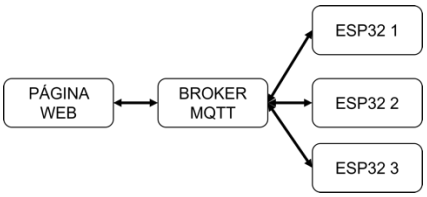


Figura 5. Esquema de Comunicación con la Página Web

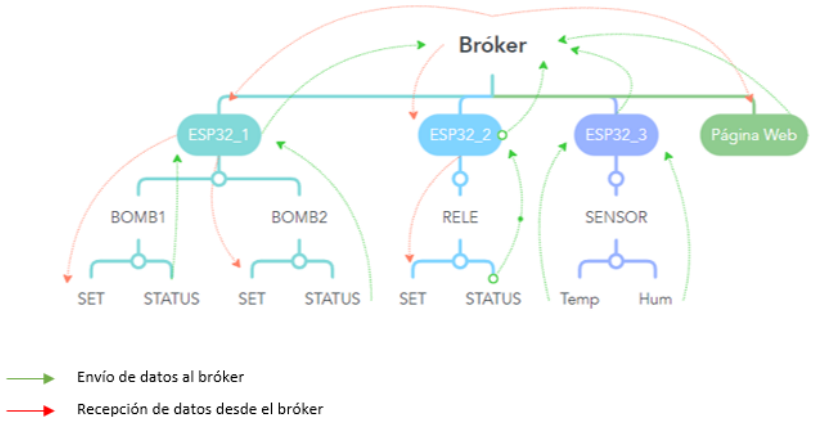


Figura 6. Flujo de datos a partir del bróker

En la Figura 7. se puede observar el envío de datos, desde el origen hasta el receptor, los cuales como se muestra en el diagrama son publicados en tópicos específicos, además, se puede ver que el tópico por el cual fluye un dato es el mismo en todo el recorrido, sin embargo, el estado de los clientes en el bróker discrepa entre sí ya que mientras uno está suscrito y toma el dato del bróker, el otro publica el dato en el bróker.

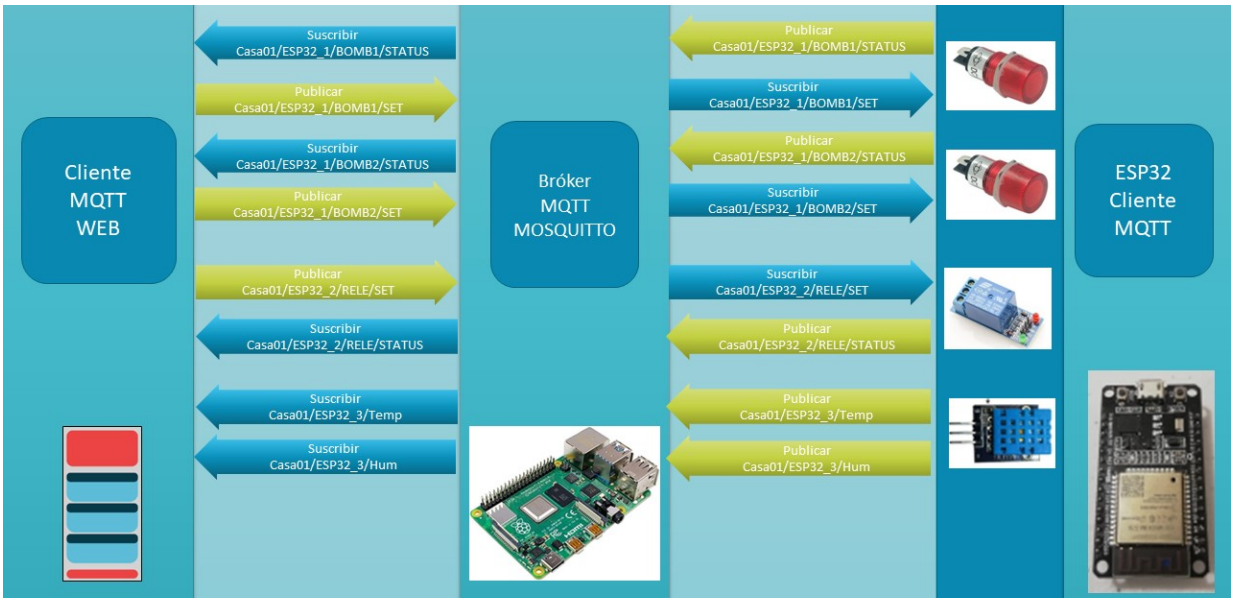


Figura 7. Sistema IoT MQTT

Como se ha mencionado en el documento, los dispositivos conectado al bróker y al servido HTTP se consideran como clientes una vez que han realizado una conexión exitosa y estable, en la Figura 8 se muestra un diagrama en el cual se observa de manera más clara el estado de los dispositivos.

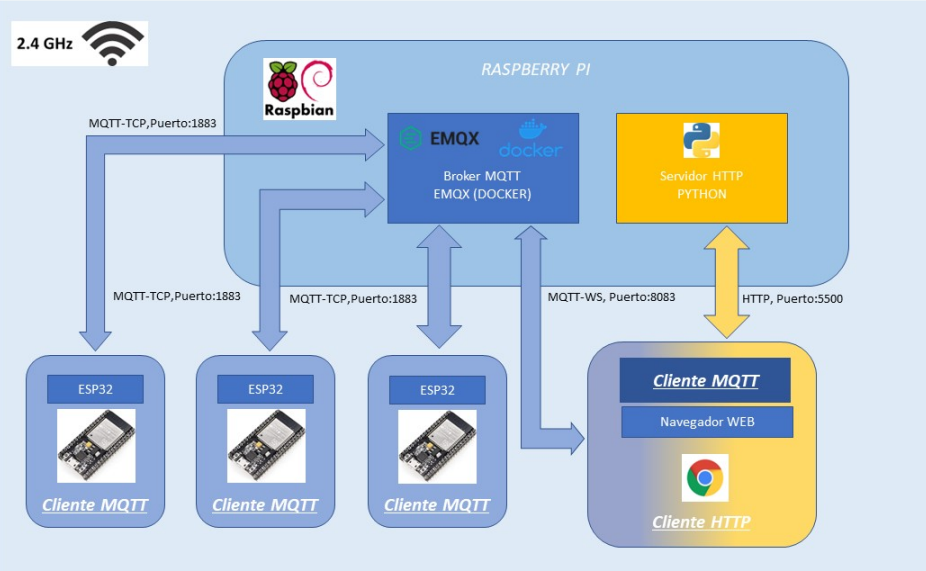


Figura 8. Diagrama de clientes MQTT y HTTP

El proceso completo de desarrollo y deploy (puesta en marcha) de la página web se dividió en 3 fases:

1. Servidor Python. En esta etapa, la más corta de todas, mediante un simple comando en el lenguaje de programación Python se levanta un servidor en la red local para que la página esté disponible para todos nuestros usuarios. El servidor, en este caso, se levantó en el puerto 5500.
2. Mockup. Para esto se usaron herramientas de desarrollo web para poder dar un buen diseño visual a la página que se ajustara a estándares actuales de desarrollo web tales como el responsive design (diseño responsivo).
    a. El primer "borrador" de la página (ver Figura 9.) se desarrolló en el software gratuito *Lunacy*, en este programa se bocetaron los diseños a partir de los cuales maquetaríamos nuestra página web .

a) *b)*

Figura 9. Diseño de la página para a) ordenador, b) dispositivo móvil

    b. Una vez teniendo el boceto, se realizó el diseño utilizando HTML5 para dar las etiquetas a cada elemento y CSS3 para agregar un estilo propio y algunos comportamientos a la página.
3. Implementación MQTT. Por último, es agregada la parte lógica y de programación para poder realizar una comunicación e intercambio de datos con el Bróker MQTT implementado en nuestra

Raspberry Pi. La comunicación se realizó mediante el lenguaje de programación JavaScript haciendo uso de la librería Paho MQTT.

a. Primero se declara un puerto, dirección IP, usuario y contraseña para poder realizar la comunicación con el Bróker.
b. Cada dispositivo contiene un identificador y tópico único el cual se tiene que declarar en el programa.
c. Después de establecer la conexión con las credenciales declaradas anteriormente se debe realizar la implementación de dos funciones muy importantes para el funcionamiento del programa:
    i. *Transmisión.* La página permite accionar una serie de elementos tales como bombillas o relés mediante el uso de botones, por lo que nuestra página hace un proceso de transmisión de datos, específicamente el identificador y tópico antes mencionado, para poder indicarle a las tarjetas qué acción debe tomar.
    ii. *Recepción.* En esta función, los dispositivos, mediante el bróker, envían constantemente datos a nuestra página web y con un pequeño filtrado y selección de estos datos podemos elegir qué acción tomar, tal como cambiar el estatus de un led (encendido/apagado) o mostrar la temperatura o cualquier otro valor de un sensor.
d. En el proceso de transmisión y recepción de datos se tuvo especial cuidado en que tanto los datos de la página web como de los sensores en el bróker estuviesen sincronizados y ambos recibiendo la misma indicación.

Dispositivos MQTT

Se utilizaron 3 tarjetas de desarrollo ESP32 (Ver Figura 10) como clientes MQTT. Estas tarjetas son adecuadas debido a su versatilidad ya que cuentan con conectividad wifi la cual es necesaria para realizar la conexión a la red local en la que se encuentra el bróker al que se enviarán los datos recabados y del que se recibirán los datos. Con este intercambio de información puede informar al usuario del estado (encendido o apagado) de un bombillo, motor o cualquier dispositivo que esté conectado y configurado con la ESP32, además, es posible enviar datos a la tarjeta con la finalidad de cambiar el estado de los dispositivos antes mencionados. (MECTRONICA, 2022)

Figura 10. Tarjeta de desarrollo ESP32

La programación de las tarjetas puede hacerse con distintos softwares, en este proyecto se utilizó Arduino IDE. Para poder programar las tarjetas es necesario añadir el modelo de las tarjetas a Arduino IDE ya que no vienen por defecto en el software. La conexión a internet se puede realizar añadiendo la librería 'WiFi' al inicio del código que será cargado en la tarjeta de desarrollo, una vez conectada a una red WiFi la tarjeta puede interactuar con un bróker MQTT, para ello, es necesario utilizar una librería la cual permite realizar esta interacción. Existen distintas librerías que pueden añadirse a Arduino IDE para poder interactuar con el bróker MQTT, por ejemplo, PubSubClient y MQTT, por mencionar algunas, en este caso se optó por PubSubClient (ESPRESSIF, 2022).

La Tabla 1. Muestras los elementos que están conectados a las tarjetas ESP32. Los dispositivos conectados como las bombillas y el relé pueden cambiar su estado de apagado a encendido dependiendo del valor de entrada que lea la tarjeta en el tópico al cual está suscrita, la configuración de las tarjetas está hecha de tal manera que el bróker recibe una retroalimentación del estado del relé y las bombillas, para la programación de las tarjetas con estos dos dispositivos no es necesario agregar una librería extra en el código de Arduino IDE. En el caso del sensor de humedad y temperatura, éste toma mediciones de la temperatura y humedad del lugar donde se encuentra, posteriormente envía los datos a la tarjeta y esta los envía al bróker, el código que es cargado en la tarjeta que es conectada con el sensor es necesario añadir la librería DHT (beegee_tokyo, 2022).

Tabla 1. *Distribución de dispositivos*

Tarjeta	Sensor	Bombilla	Relé
ESP32_1	Ninguno	2	Ninguno
ESP32_2	Ninguno	Ninguna	1
ESP32_3	Temperatura/humedad	Ninguno	Ninguno

Resultados

Como resultado de este proyecto se obtuvo una plataforma de *Edge computing* funcional que, aunque no es de grandes prestaciones, es capaz de llevar a cabo los procesos de los datos que son requeridos por los dispositivos para poder mostrar los mensajes al usuario. Todos los elementos físicos que tienen una conexión WIFI se conectaron de manera exitosa a la red local creada en el ordenador como de muestra en la Figura 11.

Figura 11. Red WIFI local creada desde un ordenador.

Se obtuvo una página web (ver Figura 12.) con botones los cuales funcionan de tal manera que cuando son accionados (en el caso de las bombillas y el relevador), dependiendo la tarjeta, el usuario cambia el estado de los dispositivos, este cambio de estados es confirmado con la lógica de programación de las tarjetas la cual envía una retroalimentación del estado de los dispositivos así el usuario puede ver el estado de los dispositivos cuando estos tienen un cambio.

.

VOLUMEN 16
XXVII Verano De la Ciencia
ISSN 2395-9797
www. jóvenesenlaciencia.ugto.mx

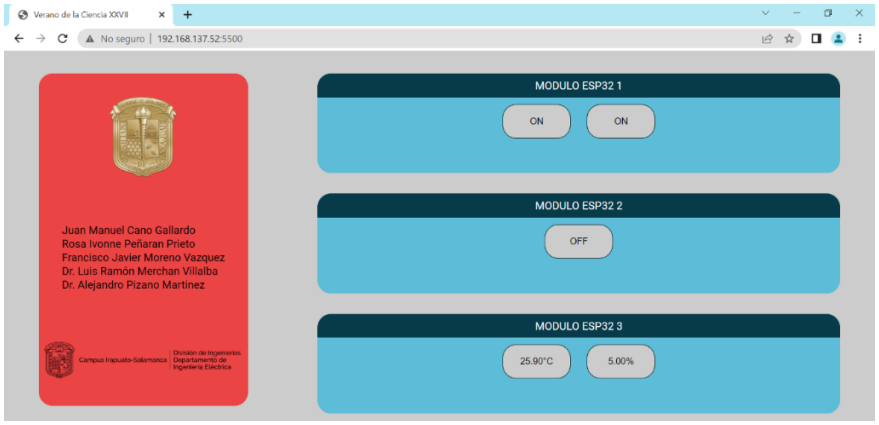


Figura 12. Página web tomando datos del bróker

Fue posible conectar los dispositivos al bróker de manera autónoma, es decir, solamente conectando las tarjetas a la alimentación, estas automáticamente se conectaron a la red local WIFI y al bróker MQTT. En la Figura 13 se puede apreciar una aplicación del bróker EMQX que permite ver los tópicos que se encuentran en bróker, identificando aquí los tópicos de la plataforma desarrollada.

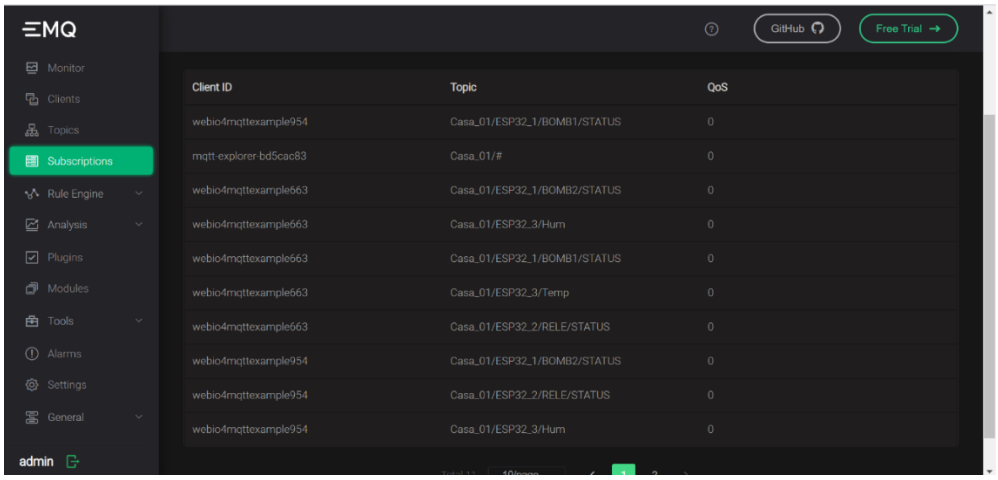


Figura 13. Clientes subscritos al bróker

La Figura 14 muestra un cliente MQTT de un ordenador desde el cual se corroboró que la información proveniente de los dispositivos fuera la esperada, es decir, que cuando se realizara un cambio de estado de los dispositivos desde la página web, los clientes MQTT correspondientes regresaran una retroalimentación del cambio realizado. Aunque dispositivo que tiene el sensor no recibe datos, éste solo envía datos al bróker, esta acción la lleva a cabo solamente cuando el valor de la temperatura o el valor del porcentaje de humedad cambian con la finalidad de evitar una saturación en el tráfico de datos.

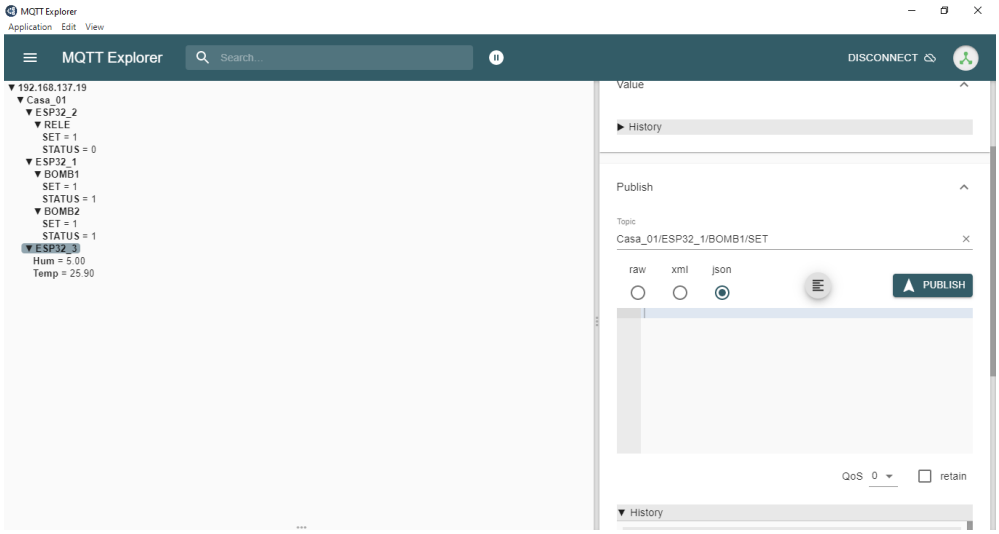


Figura 14. Información publicada en el bróker por las tarjetas

En la Figura 15, se muestran los dispositivos programados para conectarse como clientes MQTT al ser conectados a la alimentación, además, se muestra que están conectados a las bombillas, al relé y al sensor según sea el caso.

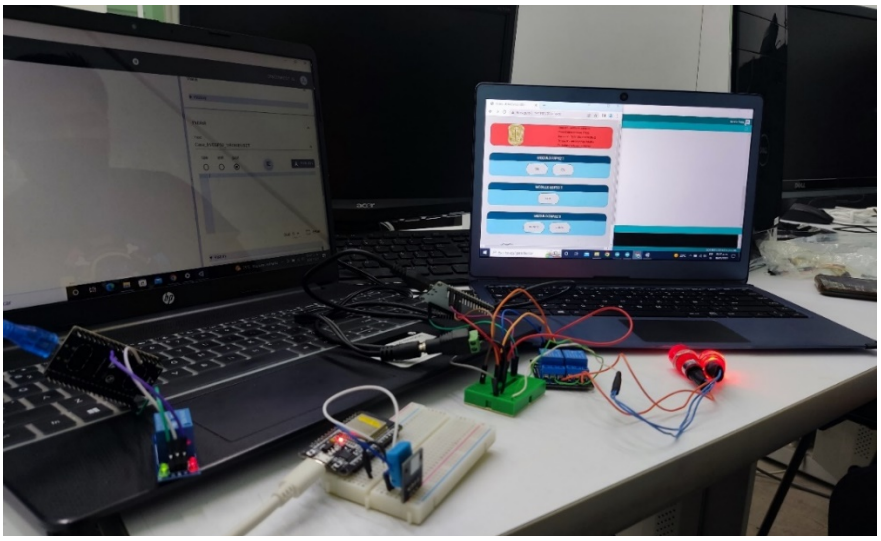
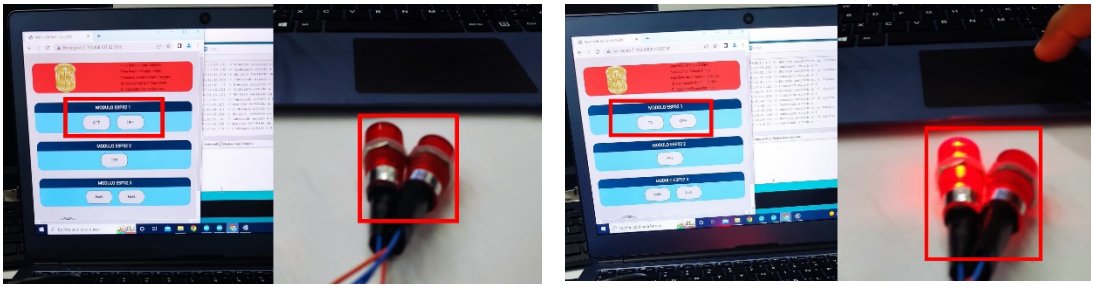
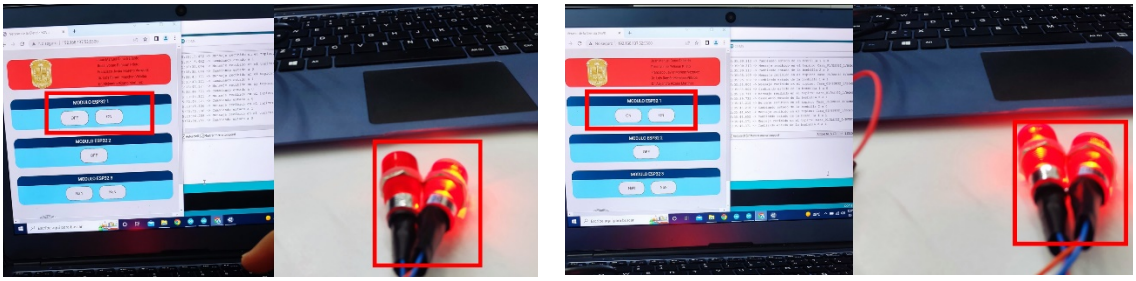


Figura 15. Montaje de dispositivos conectados al bróker

Como se muestra en la Figura 16 la página web indica el estado correcto en el que se encuentran las bombillas, además, en la Figura 17. se muestra por medio del monitor serie de Arduino IDE que en la tarjeta también se ve relejado el estado de las bombillas y que éste estado coincide con el mostrado en la página web.



a) *b)*



c) *d)*

Figura 16. Bombillas controladas por el cliente MQTT ESP32_1 a) bombillas 1 y 2 apagadas, b) bombilla 1 encendida, c) bombilla 2 encendida, d) bombillas 1 y 2 encendidas

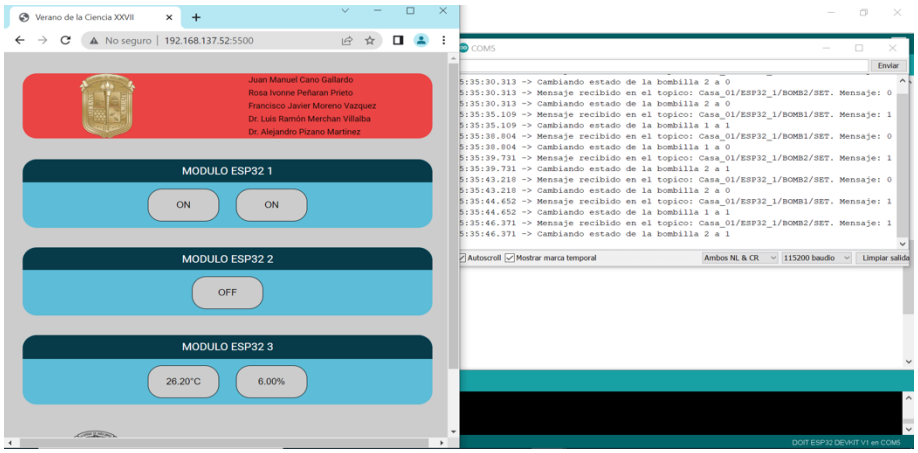
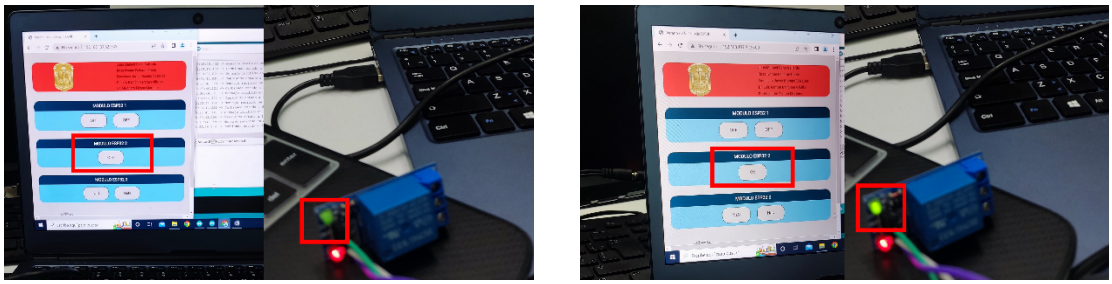


Figura 17. Cambio de estado de las bombillas visto en la página web y en la tarjeta ESP32 desde monitor serie de Arduino IDE

En la Figura 18. se muestra el estado del relevador el cual es indicado físicamente con un led verde el cual está encendido cuando el relé está desactivado (estado=0) y se apaga cuando el relé está activado (estado=1). En la Figura 19. Se corrobora la información anterior por medio del monitor serie de Arduino IDE el cual muestra el último estado del relé que fue cambiado por la tarjeta de desarrollo.



a) b)

Figura 18. Relevador controlado por cliente MQTT ESP32_2 a) relevador desactivado, b) relevador activado

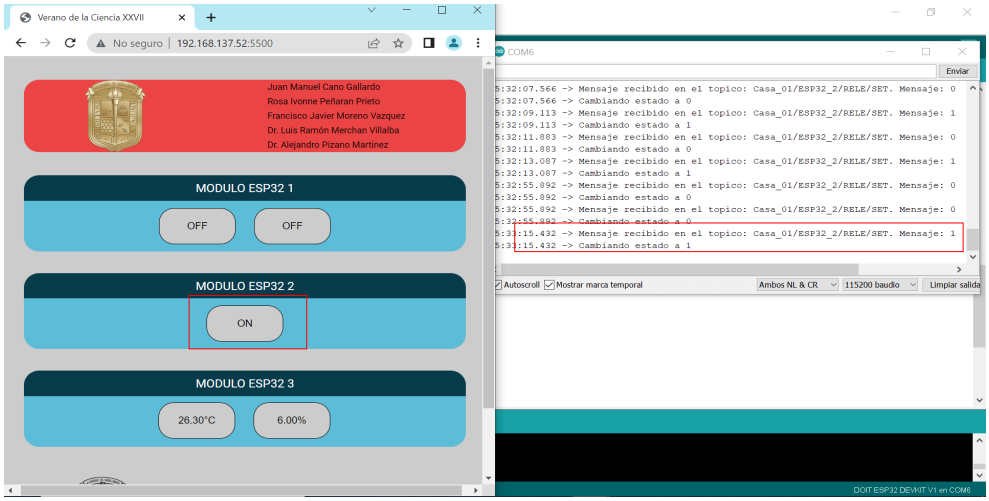


Figura 19. Cambio de estado del relevador en la página web y en la tarjeta ESP32 desde monitor serie de Arduino IDE

En la Figura 20 se observa el sensor de humedad conectado a la tarjeta de desarrollo, físicamente no hay manera de corroborar que esté en funcionamiento, sin embargo, en el monitor serie de Arduino IDE se pueden mostrar las lecturas del sensor, en la Figura 21. se muestran las lecturas del sensor en el monitor serie las cuales corresponden con la información mostrada en tiempo real en la página web, por lo que se corrobora que el usuario puede saber en tiempo real estado de las lecturas del sensor.

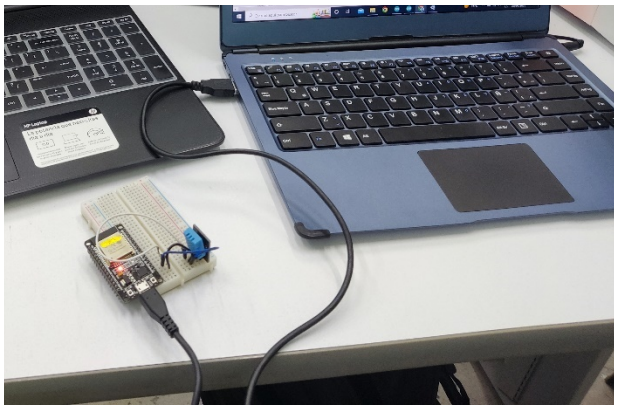


Figura 20. Sensor de humedad conectado a la tarjeta de desarrollo

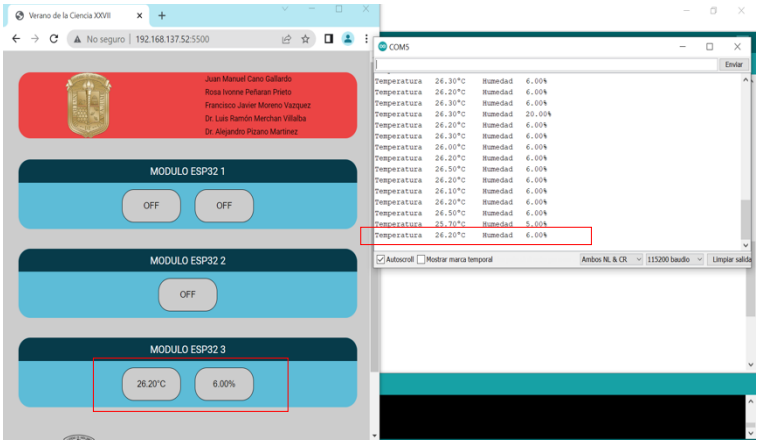


Figura 21. Verificación de que la información mostrada en la página web sea la misma que la recibida por la tarjeta ESP32 vista desde monitor serie de Arduino IDE

Conclusiones

Es posible desarrollar una plataforma de control con elementos de bajo costo que puedan trabajar de manera óptima en conjunto según las necesitades del usuario. El procesamiento de los datos, así como la distribución de estos a los dispositivos suscritos en cada uno de los tópicos del bróker es fundamental ya que un dato enviado a un dispositivo erróneo puede desencadenar una serie de errores en el funcionamiento que controlan las tarjetas de desarrollo, además, es de suma importancia verificar que los datos obtenidos por las tarjetas, en este caso los datos del sensor de temperatura sean los mismos que se están presentando en la página web.

Al tratarse de envío de datos en tiempo real es fundamental corroborar que efectivamente los datos envidados entre dispositivos se vean reflejados en un lapso muy corto de tiempo, pues pueden crearse conflictos en esta parte si la programación de las tarjetas no es la adecuada, por esto fue necesario implementar una lógica en las tarjetas de tal manera que realizaran una acción solamente cuando el valor de entrada fuera distinto al valor anterior que se tenía, ya que si se manejan retardos en el código de programación, las tarjetas tardan un tiempo extra en realizar una acción, lo que hace que la comunicación entre dispositivos no sea optima ya que no se estaría realizando un intercambio de datos en tiempo real.

Esta plataforma de *Edge computing* puede extenderse teniendo comunicación con otro bróker ampliando el sistema a esquemas de *Fog* y *Cloud Comuting*; y es gracias a esto que podría implementarse en distintas industrias, por ejemplo, en la ganadería o la agricultura para conocer las condiciones de temperatura y humedad del lugar donde se coloque un sensor, así mismo podría dar alertas a los usuarios cuando se alcancen determinados valores, gracias a la capacidad de recibir una retroalimentación también podría tener aplicación en actividades que requieran saber constantemente el estado de válvulas, interruptores, actuadores, entre otros.

Referencias

ADN40. (17 de Mayo de 2022). *ADN40*. Recuperado el 2022 de Julio de 13, de https://www.adn40.mx/ciencia/mundo-un-minuto-internet-ytc

Arduino. (20 de 07 de 2022). *Arduino*. Obtenido de https://www.arduino.cc/

beegee_tokyo. (2022). *Arduino*. Recuperado el 14 de Julio de 2022, de Arduino Web Site: https://www.arduino.cc/reference/en/libraries/dht-sensor-library-for-espx/

Docker. (20 de 07 de 2022). *Docker*. Obtenido de https://www.docker.com/

EMQX. (20 de 07 de 2022). *The Most Scalable MQTT Broker for IoT*. Obtenido de https://www.emqx.io/

ESPRESSIF. (2022). *ESPRESSIF*. Recuperado el 14 de Julio de 2022, de Arduino-ESP32: https://docs.espressif.com/projects/arduino-esp32/en/latest/getting_started.html

Galeano, S. (27 de Enero de 2022). *MARKETING4ECOMMERCE*. Obtenido de https://marketing4ecommerce.net/usuarios-de-internet-mundo/#:~:text=En%20cuanto%20a%20los%20usuarios,en%20los%20%C3%BAltimos%2012%20meses

Isaias, A. M. (2016). IOT, EL INTERNET DE LAS COSAS Y LA INNOVACIÓN DE SUS APLICACIONES. *VinculaTégica, II*(1), 2313-2337.

Luis Llamas. (17 de Abril de 2019). *Luis Llamas*. Recuperado el 17 de Julio de 2022, de https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/

MECTRONICA. (2022). *MECTRONICA*. Recuperado el 13 de Julio de 13, de https://www.mactronica.com.co/tarjeta-de-desarrollo-esp32-wifi-bluetooth#:~:text=El%20ESP32%20es%20un%20SoC,de%20velocidad%20que%20el%20ESP8266.

Rodal, E. (24 de Mayo de 2021). *Podcast industria 4.0*. Recuperado el 13 de 07 de 2022, de https://www.podcastindustria40.com/fog-computing/