UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA:
INGENIERÍA DE SISTEMAS

Trabajo de titulación previo a la obtención del título de:
Ingenieros de Sistemas

TEMA:
IMPLEMENTACIÓN DE UNA RED CAN CON CONECTIVIDAD A DISPOSITIVOS IOT COMO SOLUCIÓN DE INTERCONEXIÓN DESDE UNA PLATAFORMA ABIERTA

AUTORES:
ANDRES DAMIAN TOAZA CAIZA
RONALD ALEXIS GONZÁLEZ GONZÁLEZ

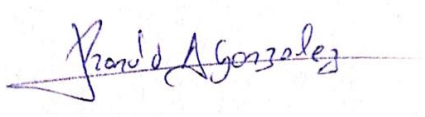
TUTOR:
MANUEL RAFAEL JAYA DUCHE

Quito, febrero 2021

CESIÓN DE DERECHOS DE AUTOR

Nosotros, Andres Damian Toaza Caiza con documento de identificación N.º 1723081400 y Ronald Alexis González González con documento de identificación N.º 1724754914, manifestamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de titulación intitulado: IMPLEMENTACIÓN DE UNA RED CAN CON CONECTIVIDAD A DISPOSITIVOS IOT COMO SOLUCIÓN DE INTERCONEXIÓN DESDE UNA PLATAFORMA ABIERTA, mismo que ha sido desarrollado para optar por el título de: INGENIEROS DE SISTEMAS, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en nuestra condición de autores nos reservamos los derechos morales de la obra antes citada. En concordancia, suscribimos este documento en el momento que hacemos entrega del trabajo final en digital a la Biblioteca de la Universidad Politécnica Salesiana.



…….. …….

TOAZA CAIZA
ANDRES DAMIAN
CI: 1723081400

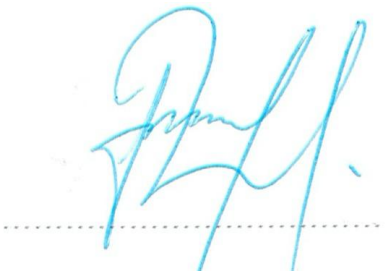
GONZÁLEZ GONZÁLEZ
RONALD ALEXIS
CI: 1724754914

Quito, febrero del 2021

DECLARATORIA DE COAUTORÍA DEL TUTOR

Yo, declaro que bajo mi dirección y asesoría fue desarrollado el Proyecto Técnico, con el tema: IMPLEMENTACIÓN DE UNA RED CAN CON CONECTIVIDAD A DISPOSITIVOS IOT COMO SOLUCIÓN DE INTERCONEXIÓN DESDE UNA PLATAFORMA ABIERTA, realizado por Andres Damian Toaza Caiza y Ronald Alexis González González, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana, para ser considerado como trabajo final de titulación.

Quito, febrero del 2021



MANUEL RAFAEL JAYA DUCHE
CI: 1710631035

DEDICATORIA

Quiero expresar mi gratitud a Dios, quien con su bendición llena siempre mi vida, ser el apoyo y fortaleza en aquellos momentos de dificultad y de debilidad, y a toda mi familia por estar siempre presentes.

Dedico a mis queridos Padres, Rafael Toaza, Martha Caiza por sus consejos, apoyo incondicional, sus valores, su motivación constante que me ayudaron en mi formación como persona y profesional, pero más que nada, por su amor y amistad, a mis hermanos Hamilton y Lewis por sus incentivos en mis estudios y en el deporte, a mis sobrinas Paula y Emilia por lo bellos momentos compartidos, a la memoria de mis abuelos por la fuerza, consejos e inspiración que me brindaron, a mis buenos amigos que siempre me apoyaron. Sería imposible llegar a este punto sin la ayuda de cada uno de ustedes.

Simplemente un millón gracias.

Andres Damian Toaza.

Este trabajo va dedicado principalmente a Dios, por brindarme salud y su bendición.

A mi madre Piedad por su infinito amor y apoyo incondicional, por siempre ser mi pilar quien con su sabiduría ha llegado a guiarme y educarme como una gran persona, por su esfuerzo admirable durante su vida que me ha permitido llegar a ser un excelente profesional.

A mi padre Manuel, por su amor y perseverancia a través de la distancia, quien me apoyo en esta etapa de mi vida para convertirme en profesional.

A mi tío Walter, por ser como un padre para mí en mi vida, con sus consejos y valores que supo impartirme desde muy pequeño.

A mi hermano por ser mi padre, hermano, pero sobre todo mejor amigo, la persona que me supo educar desde pequeño, que siempre ha confiado en mí, quien me apoyado con sus conocimientos durante mis días estudiantiles y por convertirme en tío de mi hermoso sobrino Anthony.

Ronald Alexis González.

AGRADECIMIENTOS

Agradecemos principalmente a Dios por brindarnos la sabiduría necesaria para culminar nuestros estudios y ser nuestra fuerza para seguir luchando día a día.

A nuestra querida Universidad Politécnica Salesiana Campus Sur, por las enseñanzas morales, académicas y también por ser nuestra casa durante estos años de estudio.

A nuestro tutor Msc. Rafael Jaya, por su guía y paciencia brindada en el desarrollo de nuestro proyecto.

A todos y cada uno de los docentes que siempre han sido un pilar fundamental para saber guiarnos correctamente y ayudarnos a corregir nuestros errores e impulsarnos a sobresalir y ser mejores en el ámbito personal y profesional.

A toda nuestra familia quienes siempre han estado ahí para apoyarnos, ayudarnos y aconsejarnos.

A todos nuestros amigos y amigas que a lo largo de la carrera universitaria nos han colaborado de alguna u otra manera para poder lograr este propósito tan anhelado.

Andres Damian Toaza.

Ronald Alexis González.

INDICE

Implementación de una red CAN con conectividad a dispositivos IoT como solución de interconexión desde una plataforma abierta

INDICE DE TABLAS

INDICE DE FIGURAS

INDICE DE CODIGOS

Resumen

El presente proyecto desarrollado, es una solución a la necesidad de conectar dispositivos con tecnología moderna IoT en redes limitadas, el cual, tiene como objetivo la implementación de una red CAN con conectividad a dispositivos IoT como solución de interconexión desde una plataforma abierta, siendo posible manipular sus actuadores y monitorizar los sensores.

Se elaboró un prototipo basado en dispositivos electrónicos que previamente fueron analizados mediante matrices comparativas, que ayudaron a la elección de los componentes que se adaptaron a las necesidades y características de este prototipo.

Se realizó varios análisis para obtener un diseño eficaz en la conectividad CAN, dispositivos IoT y plataforma abierta. La comunicación CAN se llevó a cabo mediante microcontroladores PIC, y el envio de datos hacia la plataforma Thinger.io, se realizó mediante un NodeMCU. Además, se utilizó un osciloscopio, el cual permitió observar el comportamiento de las señales eléctricas, y para el análisis de conectividad, se usó la herramienta de monitoreo PRTG.

En conclusión, para la valoración de resultados, se procedió a verificar la efectividad del sistema y de la aplicación, usando indicadores establecidos, de lo cual, se obtuvo como resultado un 99,09% de efectividad en su funcionamiento. El diseño desarrollado es una solución de bajo costo.

Abstract

The present project developed is a solution to the need to connect devices with modern IoT technology in limited networks. The project aims to implement a CAN network with connectivity to IoT devices as an interconnection solution from an open platform, making it possible to manipulate its actuators and monitor sensors.

A prototype was developed based on electronic devices that were previously analyzed using comparative matrices, and that helped to choose the components that were adapted to the needs and characteristics of this prototype.

Several analyses were carried out to obtain an effective design in CAN connectivity, IoT devices and open platform. CAN communication was carried out using PIC microcontrollers, and data sent to the Thinger.io platform was carried out using a NodeMCU. In addition, an oscilloscope was used, which allowed for observations of the behavior of electrical signals, and for the connectivity analysis, the PRTG monitoring tool was used.

In conclusion, for the evaluation of results, the effectiveness of the system and the application was verified, using established indicators, from which a 99.09% effectiveness in its operation was obtained as a result. The developed design is a low-cost solution.

INTRODUCCION

Antecedentes

El bus CAN surge a los inicios de la década de los ochenta, de la necesidad de encontrar una forma de interconectar los distintos dispositivos de un automóvil. Este adelanto fue posible a un grupo de ingenieros de la compañía alemana Roberth Bosch Gmbh, los mismo que dieron la posibilidad de aplicar sistemas de bus seriales dentro de los automóviles, con la finalidad de satisfacer las demandas de la Sociedad de Ingenieros, y que como resultado se obtuvo la especificación del protocolo CAN (Chikhale, 2018).

A pesar de que el bus CAN fue en un principio desarrollado para aplicaciones en automóviles, las primeras implementaciones se hicieron en el continente europeo, las cuales no se basaron principalmente en el sector automotriz, al contrario, se fabricaron elevadores en Finlandia, sistemas médicos en Holanda y máquinas textiles en Suecia.

En (Gómez, Asesora, Doctora, & Altamirano, 2017), se propuso la necesidad de realizar un dispositivo económico, que permita detectar problemas de comunicación los nodos de una red CAN, por medio de una tarjeta de desarrollo modelo Tiva™ C Series TM4C123G LaunchPad con un ARM® Cortex®-M4F basado en un MCU TM4C123G, el objetivo del software desarrollado en el sistema embebido era monitorear una red CAN, en la cual se pueda ver la comunicación entre nodos y observar el tráfico en el bus de comunicación.

En (Toubes, 2017), se propuso subsanar la falta de herramientas de software libre para la prueba y depuración de aplicaciones para ECUs (Electronic Control Unit), realizando un entorno basado en microcontrolador comercial. Se realizó una aplicación que sea capaz de enviar tramas CAN con características que el usuario desee, y a la par recibir tramas de las ECUs a las que se

encuentre conectada. Se Utilizó una placa basada en el microcontrolador LPC2387 de NXP Semiconductores.

El concepto de IoT se dio a conocer por Kevin Asthon en el Instituto tecnológico de Massachusetts en 1999, por diferentes indagaciones de identificación por radiofrecuencia (RFID) y tecnologías de sensores inteligentes. Por otro lado, entre los años 2008 y 2009 el Grupo de Soluciones Empresariales basadas en internet (IBSG) IoT da a conocer un punto en el tiempo en el que se conectaron a internet más "cosas" que personas (Evans, 2011).

En (Chiang & Zhang, 2016) planteó una arquitectura Fog, es una construcción emergente para informática, almacenamiento, control y redes que comercializa estos servicios más cerca de los usuarios a lo extenso del continuo Cloud-to-Things. Para cubrir escenarios móviles y alámbricos que atraviesa hardware y software. Como arquitectura, admite una variedad creciente de aplicaciones, incluidos los de Internet de las cosas (IoT), sistemas inalámbricos de quinta generación (5G) e inteligencia artificial integrada (IA). Este artículo de la encuesta resume las oportunidades y los desafíos de Fog, centrándose en el contexto de redes de IoT.

Problema de Estudio

En la actualidad las redes CAN se han vinculado con el uso en pequeños campus, permitiendo así la conectividad de los módulos integrados en estos espacios reducidos, de esta manera teniendo una comunicación ambigua con la persona que lo controla y a su vez con el mundo exterior.

En un principio la red CAN fue creada para el uso en sistemas automotrices, no obstante conforme otras investigaciones se puede observar sus ventajas de CAN y sus diversas implementaciones de nuevos módulos en diferentes industrias como la fabricación de aparatos médicos, estudios ferroviarias, estudios en aviación, robótica entre otros(Ng, Ng, Ali, Noordin, & Rokhani, 2010). Asimismo, se crea una variada gama de aplicaciones para otro tipo de actividades corporativas (Chen, Xu, Liu, Hu, & Wang, 2014).

Sin embargo, con el paso de los años la necesidad de conectar terminales inalámbricos que posean tecnología moderna e inteligente como IoT, ha hecho que la red se vea limitada en interconexiones con estos tipos de dispositivos, el cual no permite observar los cambios o funciones cuando se conecten en la misma, dejándola así a la red CAN en completa desventaja con las tecnologías actuales. Por lo tanto, se requiere de una solución que permita la conexión de una red CAN con dispositivos IoT.

Justificación

De la revisión sistemática de la literatura científica se desprende la evolución del IoT como la tendencia tecnológica actual (Tropmann-Frick, 2018), el cual emplea un enfoque para soluciones en la aplicación de sistemas para consumidores en todo el mundo.

El IoT es un descubrimiento científico que permite convertir a todos los aparatos tecnológicos en "Smart-objetos". Esto permite que los aparatos tecnológicos se puedan enlazar entre sí, permitiendo el envío y recepción de información a las personas, volviéndolos más eficientes (Iera, Floerkemeier, Mitsugi, & Morabito, 2010).

A pesar de las utilidades investigadas de IoT, pocos estudios se han enfocado en la aplicación de esta tecnología en redes pequeñas (Lee & Lee, 2015), entrando en este grupo la red CAN, comprendiendo asimismo que no existe lo indicado en el apartado preliminar del proyecto propuesto, el cual permitirá el manejo de una red como CAN y las nuevas tecnologías IoT para una conectividad.

Estas nuevas ideas tecnológicas cada vez son más utilizadas por personas, corporaciones e industrias para proporcionar una forma de vida apegada a la actualidad, en donde el IoT se ajusta al contexto para la aplicación en comunicaciones por medio de redes CAN, donde los bajos costos de conexión ayudan para su desarrollo, y asimismo agilizando su manejo. Además, al crear una red CAN se puede utilizar diferentes nodos IoT que permitan su fácil uso.

Con este proyecto se busca diseñar e implementar un prototipo de red CAN con conectividad a un dispositivo IoT que permita manipular sus módulos por medio de una plataforma abierta que esté conectado a una conexión multipunto, esto quiere decir, que el usuario conectado a la red pueda controlar las distintas funciones que se pueda realizar en los dispositivos.

Donde el objetivo fundamental es solucionar problemas en tiempo real, desarrollando así un sistema tecnológico, el cual permitirá añadir valor a los productos existentes y a los nuevos servicios con una pequeña inversión, mejorando así los medios relacionados con el hogar y facilitando el diario vivir de las personas.

Objetivos

Objetivo general

Diseñar e implementar un prototipo de una red CAN con conectividad a dispositivos IoT como solución de interconexión a una plataforma abierta.

Objetivos específicos

- Revisar de manera sistemática la literatura existente acerca de redes CAN y dispositivos IoT.
- Examinar las diferentes plataformas abiertas en el mercado que permitan realizar una interconexión mediante IoT sobre la red CAN.
- Insertar los módulos sobre una red CAN para el monitoreo y control desde la plataforma abierta.
- Realizar pruebas y documentar los resultados obtenidos sobre la conexión entre la red CAN y los módulos IoT.

Metodología

La metodología aplicada para el diseño del prototipo CAN se denomina pruebas de conceptos (PoC), es decir, esta implementación es parte de una metodología en la que se comprueba la accesibilidad de la idea propuesta (Director, 2017).

Proceso PoC

Figura 1. Ciclo del Proceso PoC
Elaborado por: Los autores

Paso 1: Precisar el problema

Diseñar el problema al que se enfrenta el prototipo CAN marcando los objetivos claros y una organización que precise las diferentes partes del trabajo y el tiempo que se les va a dedicar.

Paso 2: Estudio bibliográfico

Realizar un estudio de los procesos involucrados en el diseño del prototipo CAN, con la finalidad de tener bases suficientes para poder tomar decisiones correctas.

Paso 3: Diseño de la solución

Diseñar una solución que desempeñe con el objetivo principal en el diseño del prototipo CAN. Para ello se examina diferentes métodos y se observa de qué modo cumplen los objetivos propuestos.

Paso 4: Implementación de la solución

Implementar el diseño del prototipo CAN y configurar la solución propuesta.

Paso 5: Diseñar escenarios de prueba

Diseñar el prototipo CAN en escenarios de prueba que ayuden a evidenciar que la solución es funcional y que cumple con los objetivos.

Paso 6: Analizar los resultados y elaborar conclusiones

Después de hacer las pruebas se realiza un análisis de los resultados que han devuelto, basándose en los objetivos que se habían planteado al principio, se valora el éxito del diseño del prototipo CAN. En caso de ser exitoso se proponen trabajos futuros para convertir esa PoC en una implementación real, en caso de considerar que no es válido se concluye que es mejor no seguir esa vía para el desarrollo final, se analizan los errores cometidos y consejos para nuevos prototipos.

CAPÍTULO I

MARCO TEÓRICO

A continuación, se describe los fundamentos teóricos relacionados en el desarrollo del prototipo.

1.1 Fundamentos a las redes de comunicación

La red permite que la distancia no sea un problema que impida a la comunicación crear información en cualquier lugar y utilizarla sin retraso.

1.1.1 Las redes de comunicación

Las redes de comunicación se utilizan para intercambiar información, recursos y servicios entre dispositivos finales, con el objetivo principal de admitir una diversidad de aplicaciones compartidas entre ellas.

Las redes de comunicación como se muestra en la figura 2, están compuestas por nodos, hosts y líneas de comunicación (Yupapin, Mitatha, Ali, & Teeka, 2012).

Comunicación

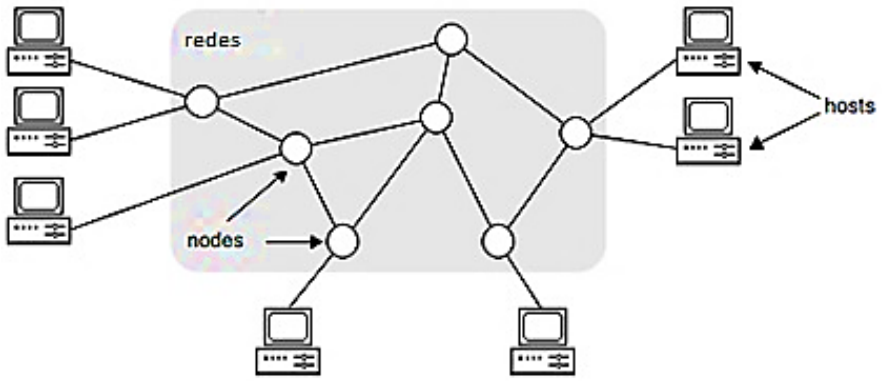


Figura 2. Componentes de red.
Fuente: (Yupapin et al., 2012)

1.1.2 Tipos de Redes

Los tipos de redes se clasifican con frecuencia de acuerdo con los límites geográficos que abarca la red. Cuando la distancia física entre los hosts está dentro de escasos kilómetros, se dice que es

una Red local (LAN) y para distancias grandes, se dice que es una Red metropolitana (MAN) o una Red amplia (WAN) son las más comunes (McMahon, 2003).

También se encuentra la red de área de control (CAN), está compuesta por una interconexión de nodos con área geográfica limitada y la red de área personal (PAN) que es una red informática organizada en torno a una persona individual para comunicarse entre terminales y dispositivos incluidos teléfonos. Como se muestra en la figura 3, los tipos de redes.

Tipos de redes

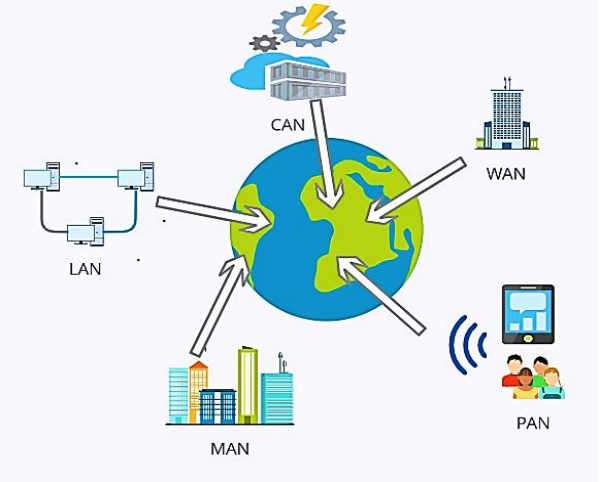


Figura 3. Tipos de Red.
Elaborado por: Los autores.

Las redes LAN, MAN, WAN, PAN y CAN suelen coexistir, por lo tanto los hosts cercanos están conectados por LAN que pueden acceder a hosts en otras LAN, PAN, CAN remotas a través de MAN y WAN (Yupapin et al., 2012).

Estas clasificaciones se basan en los niveles específicos de tecnología que utilizan al pasar de un nivel a otro. A continuación, se destaca el análisis con mayor énfasis de las redes CAN en sus conceptos y en la tecnología que se usa.

1.1.2.1 Controller Area Network (CAN)

La red CAN desde su aplicación por Bosh, destaca en el campo automotriz con el uso de una serie de microcontroladores que conectan dispositivos y sensores en un sistema o subsistema para

aplicaciones de control en tiempo real. No hay un esquema de direccionamiento utilizado en este campo, como en el sentido de las redes convencionales como Ethernet. Por el contrario, los mensajes se transmiten a todos los nodos de la red utilizando un identificador exclusivo de CAN (Corrigan, 2002).

Red Bus CAN

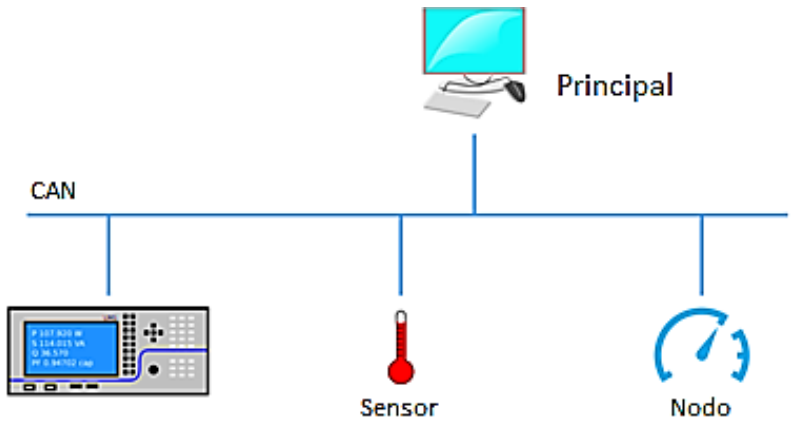


Figura 4: Bus CAN.
Elaborado por: Los autores.

En la figura 4, muestra el uso de CAN en sistemas automotrices, su uso es parecido con la diferencia que se basa en una red de sensores que envían información al ordenado principal, cuya función es monitorear y controlar otros factores incluidos:

- Subsistemas electrónicos
- Sensores de nodos
- Control de nodos

Los beneficios de CAN son los siguientes:

- Económico
- Acceso a la red
- Velocidad única de transferencia de datos compartidos
- Infraestructura

El siguiente es una implementacion destacada de CAN:

- El primer auto que incorporó CAN fue el Mercedes-Benz (Sánchez Vela, 2016)

1.2 Fundamentos a la conectividad

La conectividad de una red se representa mediante una topología estructurada por nodos que se interconectan y se relacionan entre sí por medio de enlaces.

Coexisten diferentes topologías de red, bus, malla, anillo y estrella, cada uno de los cuales se configura a su manera para facilitar la conectividad entre computadoras o dispositivos. Cada uno tiene sus propias ventajas y desventajas en términos de conectividad de red.

La red CAN está basada en una topología tipo bus. El protocolo de la red no limita el número de nodos empleados, pero depende de los chips que se utilizan. Además, permite el aumento de nodos en la red. La transmisión se limita por el tiempo de propagación de la señal a lo largo del bus, también es viable emplear soluciones basadas nuevas tecnologías aplicables (Luis & Andrango, 2007).

1.2.1 Topología Bus

Es una configuración de red que involucra comunicación por parte de sus nodos de red. Cada uno está conectado a un solo cable, que se denomina "backbone". Esto crea una red localizada que puede usarse para una variedad de propósitos. La información emitida por los nodos se propaga por todo el bus en ambas direcciones, logrando alcanzar conectividad a los nodos restantes. En la figura 5, se observa que los nodos se enlazan a un solo por medio de los Transceiver, encargados de controlar el acceso al bus (Rosado Muñoz, 2010).

Topología

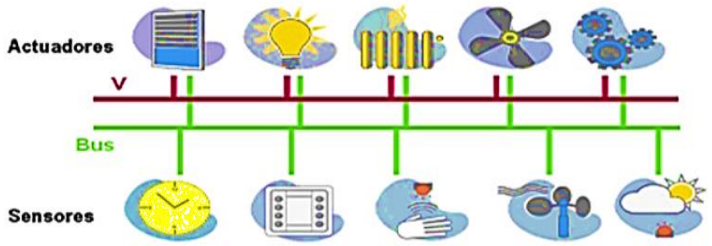


Figura 5. Interconexión en bus.
Fuente: (Rosado Muñoz, 2010)

1.2.1.1 Ventajas:

- Su implementación y crecimiento.
- Sencillez en la arquitectura.
- Es más barato que otras opciones de red.
- El fallo de una estación no afecta al resto de la red
- Es más fácil conectar nuevos nodos a la red.
- Es una red que no ocupa mucho espacio.

1.2.1.2 Desventajas:

- Las opciones de seguridad están limitadas con la topología del bus.
- Hay un límite de equipos dependiendo de la calidad de la señal.
- Los problemas de terminación de bus pueden conducir a problemas de red.
- Altas pérdidas en la transmisión debido a colisiones entre mensajes.
- El desempeño se disminuye a medida que la red crece.

1.3 Fundamentos y aplicaciones de IoT

La creación de tecnologías como Wireless son las que se utilizan en muchas áreas atraviesa muchas áreas de la vida actual. La propagación de estos puntos de conexión en la red de comunicación y trabajo crea el Internet de las cosas (IoT), en el que los actuadores y sensores se

acoplan con el ambiente que nos rodea, y los datos se comparten entre plataformas para un desarrollar común (Gubbi, Buyya, Marusic, & Palaniswami, 2013).

La adaptación de tecnologías inalámbricas como etiquetas, nodos, sensores y actuadores integrados permite que el IoT sea la tecnología revolucionaria para transformar al Internet en un Internet Futuro completamente integrado. Esto da como resultado la generación de datos que se procesan y almacenan para ejecutarse de forma clara y fácil de interpretar a los productos tradicionales.

La conectividad inteligente con las redes existentes que utiliza recursos de red forma parte del IoT. Con la tecnología Wi-Fi y acceso móvil a Internet 4G y 5G, el avance hacia redes de información y comunicación ya es evidente. Asimismo, para que el Internet de las cosas surja con éxito, su entorno deberá contener escenarios habituales como teléfonos inteligentes, portátiles y objetos cotidianos existentes, el cual permitirá una conexión compartida entre los usuarios y los dispositivos.

1.3.1 Definición

El concepto IoT fue creado en 1999 dentro de la comunidad de desarrollo de identificación de radiofrecuencia (RFID), y en la actualidad sobresale recientemente debido al incremento de terminales móviles, comunicaciones integradas, almacenamiento en la nube.

La definición común de IoT se define como internet de las cosas que se basa en una red de objetos físicos. Estos objetos conectados tienen información recopilada, regularmente para iniciar acciones, proporcionando la gestión y toma de decisiones (Patel, Patel, & Scholar, 2016).

IoT se refiere a la idea objetos comunes, que son claros, identificable, localizable y direccionable por medio de terminales por Internet, libremente de la tecnología de comunicación que se use (Wi-Fi, LAN, redes de área amplia). En la figura 6, muestra las capas para la comunicación IoT.

Comunicación IoT

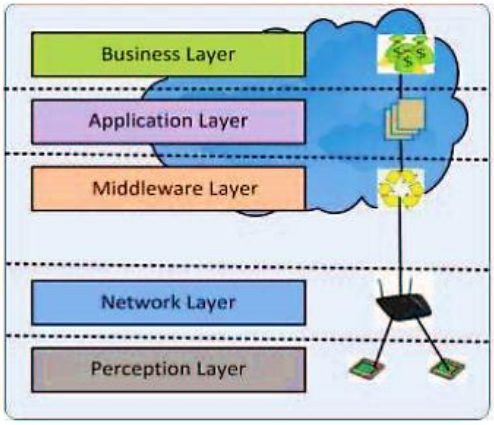


Figura 6. Capas IoT.
Fuente: (Aazam, Khan, Alsaffar, & Huh, 2014)

Las características fundamentales de la IoT son las siguientes:

- Interconectividad
- Servicios relacionados con las cosas
- Cambios dinámicos
- Escala enorme
- Seguridad
- Conectividad

1.3.2 Funcionamiento

En general el Internet de las cosas se refiere a cosas identificables de forma única con sus respectivas interfaces virtuales en una estructura similar a Internet y las soluciones de IoT comprenden un encadenamiento de componentes como:

- Módulo para la interacción con dispositivos IoT locales: Este módulo es responsable de la adquisición de observaciones y su reenvío a servidores remotos para su análisis y almacenamiento permanente.

- Módulo para análisis local y procesamiento de observaciones adquiridas por dispositivos IoT.
- Módulo para aplicación Análisis y procesamiento de datos específicos: Este módulo se ejecuta en un servidor de aplicaciones que sirve a todos los clientes. Este recibe solicitudes de clientes móviles y web y observaciones relevantes de IoT como entrada, ejecuta algoritmos de procesamiento de datos apropiados y genera salida en términos de conocimiento que luego se presenta a los usuarios
- Interfaz de usuario web o móvil: Es la representación visual de mediciones en un contexto determinado e interacción con el usuario, es decir, definición de consultas del usuario.

1.3.3 Aplicación

En la actualidad el Internet no solo es comunicación entre equipos, sino que se ha transformado en una red de terminales de todo tipo, a estos se los puede consideran tecnologías habilitadoras para Internet de las cosas y se puede agrupar en:

- Tecnologías que permiten que objetos adquieran contexto
- Tecnologías que permiten procesar objetos de información contextual
- Tecnologías para mejorar seguridad y privacidad

Las dos primeras pueden ser conjuntamente entendido como bloques de construcción funcionales, que de hecho son las características que diferencia al IoT del Internet habitual. La tercera categoría no es un requisito funcional, sino protección para la información, sin este el uso del IoT se reduciría severamente (Patel et al., 2016).

El IoT tiene aplicaciones que consideran la presencia generalizada en el entorno a través de conexiones inalámbricas y cableadas, estas pueden interactuar entre sí y cooperar con otras cosas de la siguiente manera:

- Persona a persona
- Persona a cosas
- Cosas a cosas

IoT

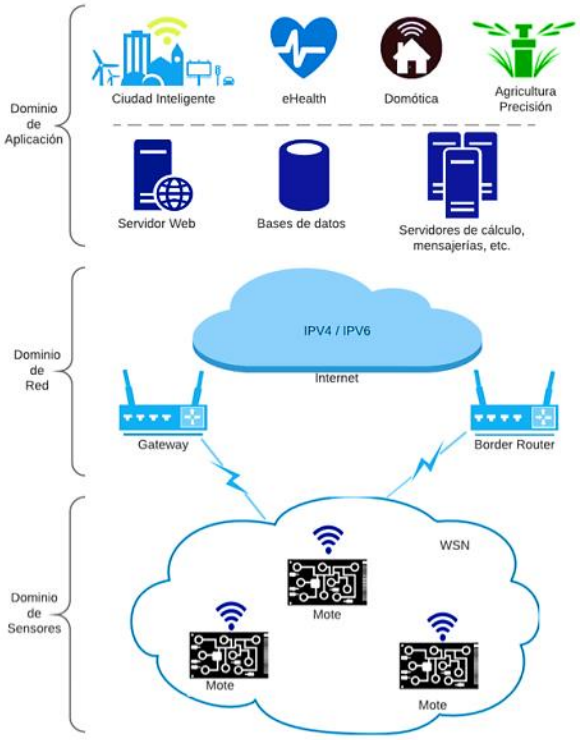


Figura 7. Aplicación IoT.
Fuente: (Ramirez Ivan, 2017)

En la figura 7, muestra los desafíos de investigación y desarrollo para crear un mundo inteligente donde lo real, digital y virtual están convergiendo para crear entornos que hacen que dispositivos de diferente tecnología como autos, smartphones, dispositivos industriales, implementos médicos, usuarios estén conectados y enviando información establecida por los protocolos determinados, para lograr organizaciones inteligentes, también se puede agregar al posicionamiento, rastreo, seguridad, control de procesos, administración, la energía, el transporte, las ciudades e incluso el monitoreo personal en tiempo real y muchas otras áreas puedan ser inteligentes (Evans, 2011).

1.3.4 IoT y su futuro

El desarrollo de tecnologías que permita crear una conexión para una interfaz IoT como electrónica de semiconductores, comunicaciones, sensores, teléfonos inteligentes, sistemas integrados, redes en la nube, virtualización de redes y software será esencial para permitir que los dispositivos físicos funcionen en entornos cambiantes y se conecten todo el tiempo.

Si bien IOT está diseñado para cumplir diferentes funciones, su aplicación y mejoras a futuro se podrían ver reflejadas en:

- Dispositivos de hardware
- Sensores
- Tecnología de la comunicación
- Tecnología de redes
- Software y algoritmos
- Tecnología de procesamiento de datos y señales
- Descubrimiento y tecnologías de motores de búsqueda
- Tecnologías de seguridad y privacidad

El Internet de las cosas no es una tecnología única, sino que es una mezcla de diferentes tecnologías de hardware y software, con esto su desarrollo a futuro ofrece soluciones basadas en la integración de la tecnología en todos los campos existentes, el cual permitirá generalizar y facilitar su implementación, aplicación y uso para los usuarios y máquinas.

1.4 Fundamentos a la interconexión

1.4.1 Modelo de referencia OSI

El modelo OSI es una representación abstracta de la comunicación entre procesos. OSI se ocupa de los estándares para la comunicación entre sistemas. En el modelo de referencia OSI,

la comunicación tiene lugar entre procesos de aplicación que se ejecutan en distintos sistemas.

El modelo OSI posee 7 capas y se divide en dos: capas superiores y capas inferiores.

- La capa superior del modelo OSI se ocupa principalmente de los problemas relacionados con la aplicación, y se implementan solo en el software. La capa de aplicación es la más cercana al usuario final. Tanto el usuario final como la capa de aplicación interactúan con las aplicaciones de software. Una capa superior se refiere a la capa que está justo encima de otra capa.
- La capa inferior del modelo OSI se ocupa de los problemas de transporte de datos. La capa de enlace de datos y la capa física se implementan en hardware y software. La capa física es la capa más baja del modelo OSI y es la más cercana al medio físico. La capa física es la principal responsable de colocar la información en el medio físico.

La Internet moderna no se basa en OSI, sino en el modelo TCP / IP más simple. Sin embargo, el modelo OSI de 7 capas todavía se usa ampliamente, ya que ayuda a visualizar y comunicar cómo operan las redes, y ayuda a aislar y solucionar problemas de redes (Luis & Andrango, 2007).

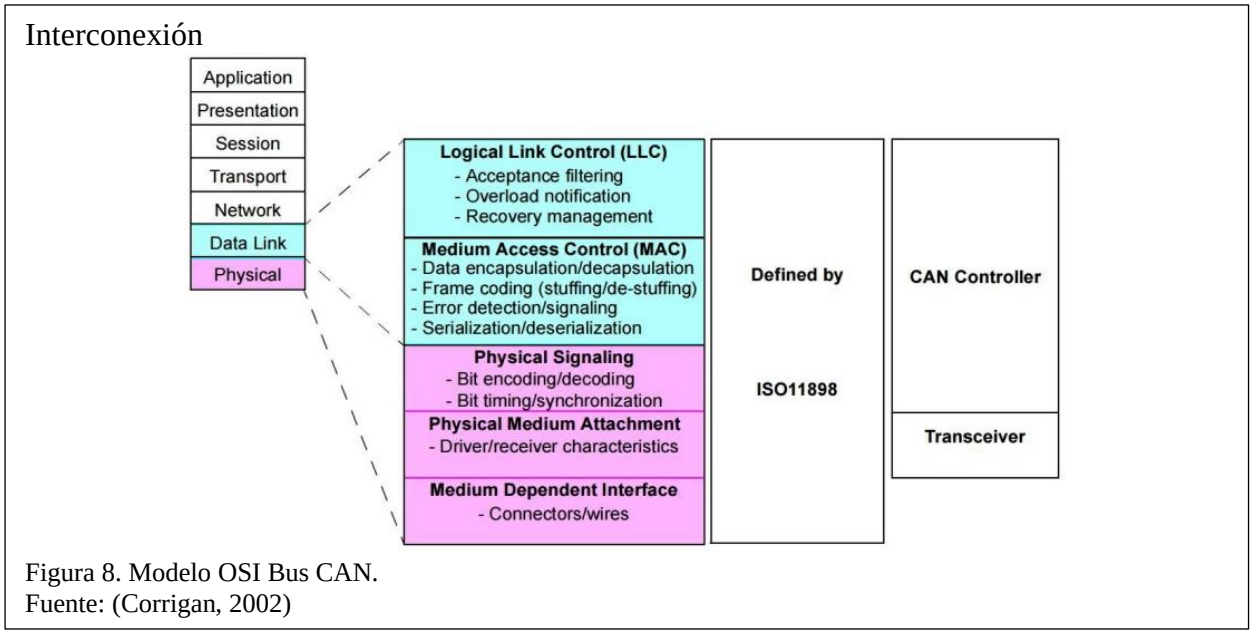


Figura 8. Modelo OSI Bus CAN.
Fuente: (Corrigan, 2002)

La distribución según el modelo OSI para el funcionamiento correcto con el protocolo CAN de la figura 8, determina los intercambios físicos ejecutados por un CAN Transceiver, el cual permite gestionar la transferencia de bits entre los nodos y el bus de datos. El CAN Transceiver se enlaza con el nodo CAN Controller, el cual elige los paquetes para la lectura en los dispositivos de alto nivel. Este se presenta como un circuito integrado que forma parte del host.

1.5 Fundamentos a plataformas abiertas

Con la creciente demanda de productos basados en IoT, el software que puede controlarlos se está abriendo camino en cada industria. Y junto con el software vienen los desarrolladores que traen consigo nuevas formas de unificar y controlar dichos productos. Una de ellas son las plataformas abiertas, un elemento básico en la industria del IoT, cuyas empresas se basan en la creación y control por medio de interfaces, que hacen que el usuario pueda aprovecharlo para su beneficio.

Una plataforma es cualquier cosa que se pueda usar para trabajar o construir sobre ella. En el caso de las plataformas de software, ofrecen un conjunto de servicios que facilitan el desarrollo de aplicaciones cuando se usan.

Exactamente tener una plataforma abierta es no tener restricciones ni limitaciones, sin embargo, la realidad es que podría haber algunas limitaciones o restricciones dependiendo el uso y la ejecución de la plataforma. La razón es que admite un conjunto específico de esquemas abiertos para la integración.

1.5.1 Definición

El concepto de plataformas abiertas puede verse como la capacidad de incorporar soluciones de terceros como parte de una estrategia global de gestión de la información, también puede agregar capacidades de captura a un documento existente o sistema de gestión de contenido, de este modo

mejorando el escenario. Este camino permite un nivel de flexibilidad, liberándolas de estar vinculadas con un único proveedor de origen. En muchos casos, ofrece una solución más ágil, que se adapta dinámicamente a los cambios tecnológicos.

Las plataformas abiertas abarcan el conjunto de componentes y reglas empleados en común en la mayoría de las transacciones de los usuarios. Los componentes incluyen hardware, software y módulos de servicio, junto con una arquitectura que especifica cómo encajan entre sí. Estas reglas se utilizan para coordinar las actividades de los participantes de la red, incluyendo estándares que aseguran la compatibilidad entre los diferentes componentes, protocolos y políticas que rigen el intercambio de información del usuario (Eisenmann, Parker, & Van Alstyne, 2008). En la figura 9, se muestra la estructura de las plataformas abiertas.

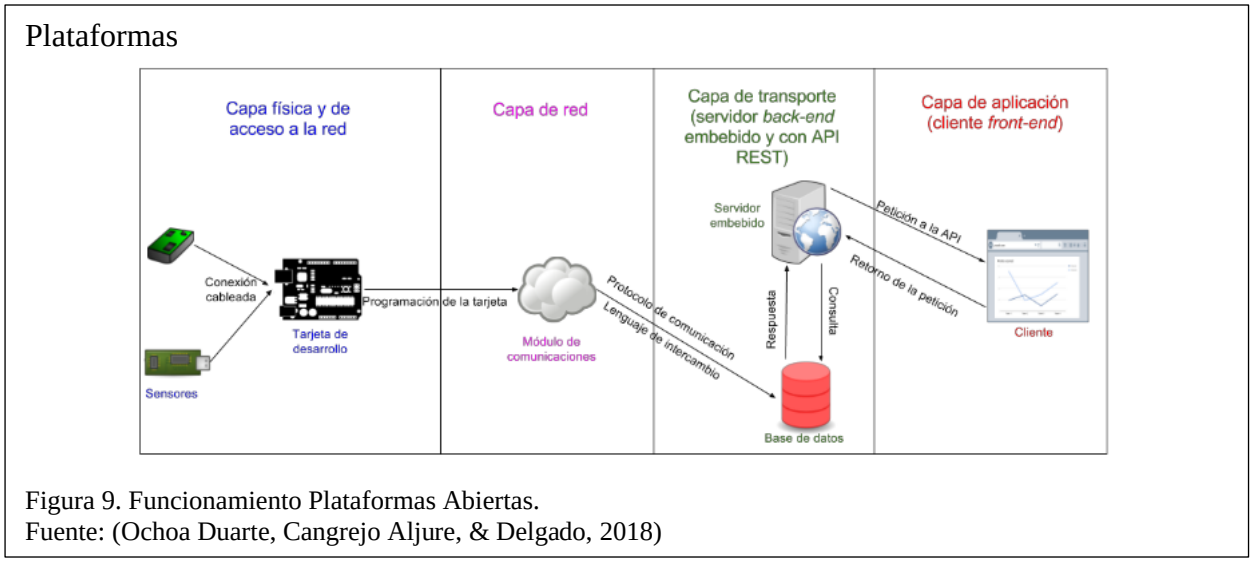


Figura 9. Funcionamiento Plataformas Abiertas.
Fuente: (Ochoa Duarte, Cangrejo Aljure, & Delgado, 2018)

1.5.2 Funcionamiento

Las plataformas son productos formados por múltiples componentes, como computadoras, automóviles y servicios de telecomunicaciones, estos constan de aquellos elementos que se usan en común o se reutilizan en las implementaciones. Una plataforma puede incluir componentes físicos y herramientas para facilitar su uso, sirviendo como base para organizar el desarrollo

técnico de componentes intercambiables y complementarios, así permitiendo interactuar unos con otros (Boudreau, 2010).

Estos comprenden varios tipos de participantes, incluidos usuarios finales, proveedores de plataformas que facilitan el acceso de los usuarios a complementos y patrocinadores que desarrollan tecnologías de plataforma. Cada uno de estos roles puede abrirse, es decir, estructurarse para fomentar la participación.

Estas aplicaciones proporcionan los siguientes servicios:

- Envió de datos.
- Recepción de datos.
- Creación de interfaces de datos.
- Gestión de recursos.
- Gestión de dispositivos.
- Integración del IoT.

1.5.3 Aplicación al IoT

Una plataforma basada en IoT es la unión de dispositivos acoplados que permite la conexión y el intercambio de datos entre módulos y aplicaciones. El IoT no es una sola única tecnología, es un conjunto de dispositivos electrónicos que funcionan en la plataforma, y elegir la plataforma adecuada es importante para el triunfo de sus procedimientos en el presente y el futuro (AT&T, 2016).

La plataforma es el corazón de la tecnología IoT, en ocasiones se denominan "middleware", lo que destaca su función como un intermediario entre las capas de aplicación y el hardware. Al contar con una plataforma robusta, cualquier dispositivo IoT será capaz de integrarse y combinarse con las aplicaciones utilizadas por el dispositivo (Pina, 2017).

1.5.4 Tipos de plataformas

Una plataforma abierta IoT es un software diseñada para la administración, monitorización y control de sensores de terceros o propias, y estos pueden variar dependiendo el uso y tipo de plataforma que se use .

Las plataformas abiertas más destacadas en el mercado son:

- Temboo.
- Ubidots.
- My Devices.
- ThingSpeak.
- Thinger.Io.
- Blynk.
- Thethings.Io.

1.5.5 Gestión de servicios IoT

Las plataformas de IoT deberían proporcionarle acceso a herramientas de software controladas por el usuario para administrar los dispositivos y las conexiones que forman parte de una solución de IoT. La conveniente funcionalidad de administración de servicios permite tomar el control de la red IoT, permitiéndole agregar, mover, eliminar o cambiar las funciones de los dispositivos IoT.

La conectividad celular de banda ancha, 3G, 4G LTE y 5G, juegan un papel importante en la difusión de dispositivos IoT, soportando una gama completa de aplicaciones con bajo y alto ancho de banda.

Las redes de corto alcance para IoT incluyen Bluetooth, ZigBee y tecnologías Wi-Fi, entre otras.

Las redes alámbricas globales pueden trabajar junto con las redes inalámbricas, permitiendo la creación de redes híbridas y proporcionar capacidades adicionales de alto ancho de banda y operación segura.

1.5.6 Soluciones IoT

La mayoría de las soluciones de IoT aprovecha la diversidad de sensores que pueden generar una gran cantidad de datos, como la ubicación, la condición y el estado. Los datos se recopilan y almacenan como flujos de datos. Cada localidad de datos suele ser pequeño, sin embargo, la cantidad de datos recopilados puede acumularse rápidamente, dependiendo de la frecuencia que envíen los dispositivos IoT.

Una plataforma IoT proporciona la capacidad de almacenar y ordenar de forma segura los datos de varios dispositivos IoT, prácticamente cualquier dispositivo y cualquier lectura de sensor puede recibir flujos de datos provenientes de múltiples conectores y procesarlos, de modo que los datos recibidos se pueden usar fácilmente mediante la plataforma (AT&T, 2016).

CAPÍTULO ll

ANÁLISIS DE HARDWARE Y SOFTWARE

2.1 Componentes

A continuación, se muestra un análisis y las características de los materiales que se van a usar para la creación del prototipo, así como también se realizará matrices de comparación de los diferentes componentes electrónicos y plataformas usadas.

Diseño de arquitectura del prototipo

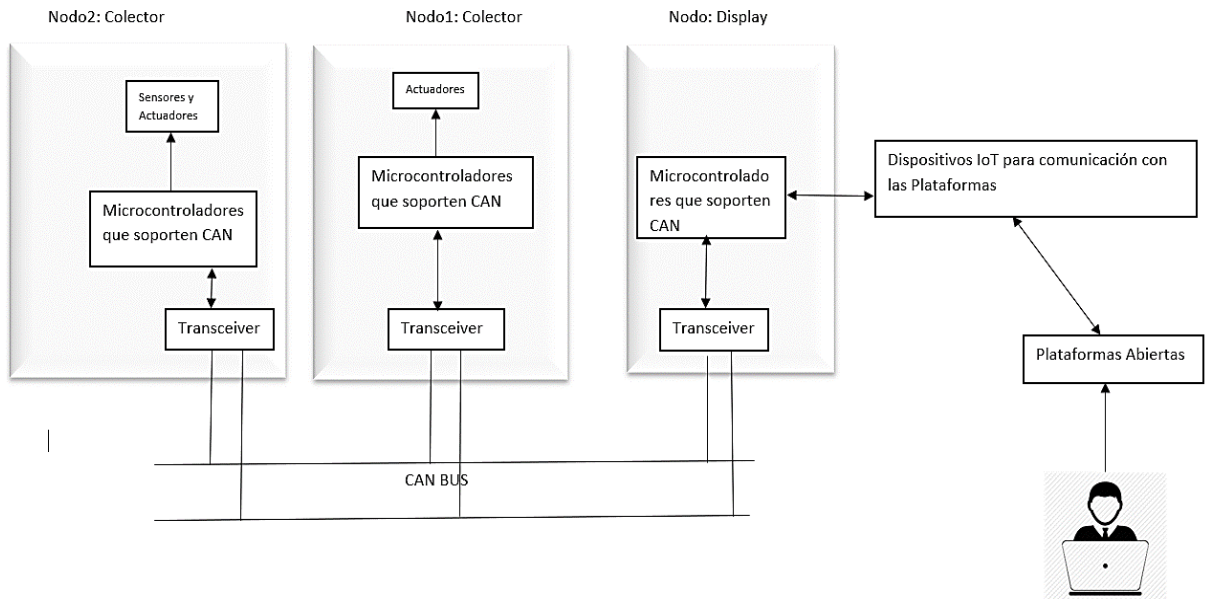


Figura 10. Arquitectura del prototipo.
Elaborado por: Los autores.

Como se observa en la figura 10, en la parte de hardware se va a utilizar los siguientes componentes:

- Sensores y actuadores.
- Microcontroladores.
- Transceivers.
- Dispositivos IoT para comunicación con las Plataformas.

2.1.1 Sensores y actuadores

2.1.1.1 Actuadores

2.1.1.1.1 Chapa Electrónica

En la figura 11, muestra la cerradura eléctrica con selenoide.

Actuador

Figura 11. Chapa Eléctrica.
Elaborado por: (Electronics, 2016)

Características:

Tabla 1. Características de la Chapa electrónica.

Voltaje de funcionamiento	12V
Corriente	0.6A
Potencia	7.5W
Longitud del cable	27cm

Nota A continuación, se muestra las características de la chapa electrónica.
Elaborado por: Los autores.

2.1.1.1.2 Ventilador

En la figura 12, muestra el ventilador pequeño con cables de conexión abiertos.

Actuador

Figura 12. Ventilador.
Elaborado por: (U. Electronics, 2016)

Características:

Tabla 2. Características del ventilador.

Voltaje de funcionamiento	5V
Ruido	21dB
Potencia máxima	1.56W
Tamaño	40x40mm

Nota: A continuación, se muestra las características del ventilador.
Elaborado por: Los autores.

2.1.1.1.3 Potenciómetro de precisión

Sirve para limitar el paso de corriente eléctrica en una alta precisión. Asimismo, se utiliza para controlar parámetros de multimedia. En la figura 13, muestra el potenciómetro.

Actuador

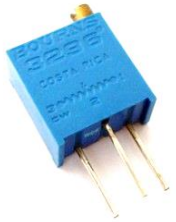


Figura 13. Potenciómetro de precisión.
Elaborado por: (G. Electronics, 2021)

Características:

Tabla 3. Características del potenciómetro de precisión.

Tolerancia	+-10%
Numero de vueltas	25
Potencia máxima	0.5W

Nota: A continuación, se muestra las características del potenciómetro.
Elaborado por: Los autores.

2.1.1.1.4 Potenciómetro de 50 Kohm

Posee una resistencia variable mecánica, limitan el paso de la corriente eléctrica provocando una caída de tensión en ellos. En la figura 14, muestra el potenciometro.

Actuador

Figura 14. Potenciómetro de precisión.
Elaborado por: (G. Electronics, 2021)

Características:

Tabla 4. Características del potenciómetro 50 Kohm.

Potencia	1W
Switch	No
Dimensiones de la base	
Diámetro	1.6cm
Largo	0.7cm
Dimensiones del vástago	
Diámetro	0.6 cm
Largo	1.5cm

Nota: A continuación, se muestra las características del ventilador.
Elaborado por: Los autores.

2.1.1.1.5 Matriz de comparación de actuadores

Tabla 5. Matriz de características.

Características de actuadores	Potenciómetro	Ventilador	Chapa electronica	Potenciometro 50 KOHM
Voltaje (V)	0.5	5	12	0.71
Corriente (MA)	0.5	0.3	0.6	0.5
Tamaño (MM)	39 a 15.5	40 a 40	39 a 20.5	1.6 a 1.5
Peso (G)	0.3	1	5	2.5
Precio ($)	$2.00	$5.00	$15.00	$3.00

Nota: A continuación, se muestra un análisis de las características de los actuadores.
Elaborado por: Los autores.

De acuerdo con la tabla 5, se utilizará los actuadores Potenciómetro, Chapa Electrónica y Ventilador.

Potenciómetro de precisión, se utilizará para envió de datos variables, además de contar con un peso ligero y un bajo costo.

Chapa electrónica, se utilizará para simular un seguro de una puerta o ventana, además de contar con un costo reducido y un tamaño apropiado para el prototipo.

Ventilador, se utilizará como un adaptador básico y fácil de implementar, además de contar con un bajo costo y tamaño apropiado para el prototipo.

2.1.1.2 Sensores

2.1.1.2.1 Sensor de Temperatura

Es un módulo con un sensor de temperatura, cediendo información mediante un bus de datos serie digital, un LED y una resistencia. En la figura 15, muestra el sensor de temperatura.

Sensor

Figura 15. Sensor de temperatura.
Elaborado por: (G. Electronics, 2021)

Características:

Tabla 6. Características del sensor de temperatura.

Voltaje de funcionamiento	3V a 5.5V
Medición de temperatura	-55 ° C a 125 ° C
Precisión de medición	± 0.5 ° C
Tamaño:	18.5mm x 15mm
Interfaz	1-Wire (OneWire)

Nota: A continuación, se muestra las características del sensor de temperatura.

Elaborado por: Los autores.

2.1.1.2.2 Sensor de Temperatura y humedad (DHT11)

Es un sensor de temperatura/humedad digital básico y poco costo. Posee un sensor de humedad capacitiva y un termistor (termómetro de resistencia, o una resistencia cuya resistencia depende de la temperatura), y refleja datos por medio de una señal digital. En la figura 16, muestra el sensor DHT11.

Sensor

Figura 16. Sensor de temperatura DTH11.
Elaborado por: (G. Electronics, 2021)

Características:

Tabla 7. Características del sensor DHT11.

Voltaje de trabajo	3.5V a 5.5V
Corriente de trabajo	0,3 mA (medida) 60uA (en espera)
Salida	Datos serie
Temperatura	0°C a 50°C
Humedad	20% a 90%
Tamaño	16 bits
Exactitud	1°C y 1%

Nota: A continuación, se muestra las características del sensor DHT11.
Elaborado por: Los autores.

2.1.1.2.3 Sensor de vibración módulo ky-002

Este módulo es un sensor de impacto digital. Producirá una señal de alto nivel cuando detecta un evento de descarga. También tenemos que considerar el posicionamiento del interruptor se debe

ubicar físicamente en el área que se requiere monitorear. En la figura 17, muestra el sensor de vibración ky-002.

Sensor

Figura 17. Sensor de vibración ky-002.
Elaborado por: (G. Electronics, 2021)

Características:

Tabla 8. Características del sensor de vibración.

Tensión de trabajo	3.3 V – 5 V
Corriente	5 mA
Dimensiones	18.5 mm x 15 mm
Peso	4 gr
Señal digital de salida Inversa	Alta (HIGH) significa cuando no hay detección de vibración y un nivel bajo (LOW) es porque tenemos vibración.

Nota: A continuación, se muestra las características del sensor de vibración modulo ky-002.
Elaborado por: Los autores.

2.1.1.2.4 Sensor Led RGB

Genera todo tipo de colores con el uso de un microcontrolador de tres colores a través del orificio que puede hacer cualquier color que desee combinando diferentes intensidades de rojo, azul y verde. En la figura 18, muestra el sensor led RGB.

Sensor

Figura 18. Sensor led RGB.
Elaborado por: (G. Electronics, 2021)

Características:

Tabla 9. Características del sensor led RGB.

Tipo	Sensor Led RGB
Voltaje	3.3 V a 5 V
Diámetro del LED RGB	5 mm
Número de modelo	KY-016
Temperatura	-40 a +85
Peso	2 gr
Bajo consumo de energía	Si

Nota: A continuación, se muestra las características del sensor led RGB.
Elaborado por: Los autores.

2.1.1.2.5 Sensor de proximidad

Es un sensor que emite un haz de infrarrojo y recibe el has reflejado, cuenta con un comparador de voltaje de amplio rango y con una sensibilidad ajustable. En la figura 19, muestra el sensor de proximidad.

Sensor

Figura 19. Sensor de proximidad.
Elaborado por: (G. Electronics, 2021)

Características:

Tabla 10. Características del sensor de proximidad.

Potencia	3.3V a 5V
Dimensiones	39mm a 15,5mm (incluye el LED IR)
Tamaño de los orificios de montaje	3 mm
Rango de detección	2cm a 30cm (dependiendo del color del obstáculo)

Nota: A continuación, se muestra las características del sensor de proximidad.
Elaborado por: Los autores.

2.1.1.2.6 Matriz de comparación de sensores

Tabla 11. Matriz de características.

Características de sensores	Temperatura	Proximidad	Led RGB	Vibrador	DTHT11
Voltaje (V)	0.5	3.3 a 5	3.3 a 5	3.3 a 5	3.3 a 5
Corriente (MA)	5	3	1	1	0.3
Tamaño (MM)	18.5 x 15	39 x 15.5	19 x 15	18.5 x 17	39 x 15.5
Peso (G)	0.5	1	1	0.5	0.8
Precio ($)	$2.00	$3.00	$1.00	$2.00	$2.00

Nota: A continuación, se muestra un análisis de las características de los sensores.
Elaborado por: Los autores.

De acuerdo con la matriz comparativa se utilizará los sensores de proximidad y el sensor DHT11. El sensor de proximidad se utilizará para detectar objetos o personas que estén cerca, el mismo que mandará un valor booleano, además de tener un costo bajo, y su tamaño pequeño que se ajusta al prototipo propuesto. El sensor DHT11 medirá la temperatura y la humedad en tiempo real, enlazado a una bocina, donde se alertará si la temperatura supera los 30 °C, además de tener un costo bajo y su tamaño pequeño que se ajusta al prototipo.

2.1.2 Microcontroladores

Estos microcontroladores se desarrollaron con el principal uso en aplicaciones complejas, de alta densidad y de alto número de pines. Los microcontroladores PIC18F ofrecen soluciones rentables

para aplicaciones de propósito general escritas en C que utilizan un sistema operativo en tiempo real (RTOS) y requieren una pila de protocolos de comunicación compleja como TCPF /IP, CAN, USB o ZigBee. Los dispositivos PIC18F proporcionan memoria flash de programa en tamaños de 8 a 128 Kb y memoria de datos en 256 a 4Kb, operando en un rango de 2.0 a 5.0 Voltios, a velocidades de 40 MH.

2.1.2.1 Microcontrolador Microchip Pic18f2480

Microcontroladores PIC

Figura 20. Pic18f2480.
Fuente: (Microchip Technology Inc, 2009)

Características:

Tabla 12. Pic18f2480.

Tipo de memoria	Rápida
Tamaño de la memoria del programa (Kb)	16
Velocidad de CPU (MIPS/DMIPS)	10
SRAM(B)	768
Datos EEPROM/HEF (bytes)	256
Número de módulos can	1
Número de Pines	28
Rango de operación de voltaje (v)	2 a 55

Nota: A continuación, se muestra las características principales del Pic18f2480 .
Elaborado por: Los autores.

2.1.2.2 Microcontrolador Atmel AT90CAN64-16AU

Microcontroladores Atmel

Figura 21. AT90CAN64-16AU.
Fuente: (Futurlec, 2021)

Características:

Tabla 13. Atmel AT90CAN64-16AU.

Tipo de memoria	Rápida
Microcontrolador	64-Pin AVR
EEPROM memoria datos (Kb)	2
I/O puertos	53
Memoria rápida (Kb)	64
SRAM memoria datos (Kb)	4
Oscilador externo (MHz)	Hasta 16

Nota: A continuación, se muestra las características principales del AT90CAN64-16AU.
Elaborado por: Los autores.

2.1.2.3 Microcontrolador STM32F042C4T6

Microcontroladores STM

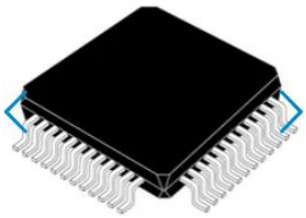


Figura 22. STM32F042C4T6.
Fuente: (Augmented, 2021)

Características:

Tabla 14. STM 32F042C4T6.

Tipo de memoria	Rápida
Microcontrolador	Core: ARM 32-bit Córtex-MO CPU
EEPROM memoria datos (Kb)	3
I/O puertos	32

Memoria rápida (Kb)	16 a 32
SRAM memoria datos (Kb)	6
Oscilador externo (MHz)	Hasta 16

Nota: A continuación, se muestra las características principales del STM 32F042C4T6.
Elaborado por: Los autores.

2.1.2.4 Matriz de comparación de microcontroladores PIC

Tabla 15. Matriz de características.

Características	PIC18F2480	T90CAN64-16AU	STM32F042C4T6
Frecuencia de Operación	DC – 40 MHZ	DC – 40 MHZ	DC – 48 MHZ
Programa de memoria (bytes)	16384	65536	16384
Memoria de Datos (bytes)	768	4	6144
Moria de datos EEPROM (bytes)	256	2048	3072
E/S puertos	5	53	38
Módulos CAN	1	1	1
Voltaje de entrada / salida (V):	2 a 5.5	2.7 a 5.5	1.65 a 3.6
Documentación	Si	Si	Si
Stock	Si	No	No
Costo	$8.38	$7.20	$2.65

Nota: A continuación, se muestra un análisis de las características de los microcontroladores.
Elaborado por: Los autores.

De acuerdo con la tabla 15, se utilizará el PIC18f2480 a razón que este microcontrolador se encuentra en stock en el país, por lo tanto, su costo se mantiene a diferencia de los otros microcontroladores, donde su precio se elevaría por motivos de importación en hasta un 150% más, tomando en cuenta que el modelo a desarrollar es un prototipo donde los costos son base fundamental para el desarrollo de este.

2.1.3 Transceivers

Es un dispositivo que transmite y recibe señales analógicas o digitales. El término se usa con más frecuencia para describir el componente en las redes de área local (LAN), que realmente aplica señales al cable de red y detecta señales que pasan a través del cable. Para muchas LAN, el

transceptor está integrado en la tarjeta de interfaz de red (NIC). Sin embargo, algunos tipos de redes requieren un transceptor externo.

2.1.3.1 Transceiver MCP2551

Transceiver

Figura 23. MCP2551.
Fuente: (Microchip Technology Inc, 2009)

Características:

Tabla 16. MCP2551.

Estilo montaje	SMD/SMT
Voltaje de alimentación operativo (V)	5
Temperatura de trabajo	-40 °C
Velocidad de transmisión de datos	1Mb/s
Número de transceptores	1
Sensible a la humedad	Si
Producto	CAN Transceivers

Nota: A continuación, se muestra las características principales del MCP2551.
Elaborado por: Los autores.

2.1.3.2 Transceiver MAX3051

Transceiver

Figura 24. MAX3051.
Fuente: (Futurlec, 2021)

Características:

Tabla 17. MAX3051.

Estilo montaje	SMD/SMT
Voltaje de alimentación operativo (V)	3.3
Temperatura de trabajo	-40 °C
Corriente de suministro operativa	35Ma
Altura	1.25mm
Longitud	2.9mm
Producto	CAN Transceivers

Nota: A continuación, se muestra las características principales del MAX3051.
Elaborado por: Los autores.

2.1.3.3. Transceiver ATA6561-GBQW-N

Transceiver

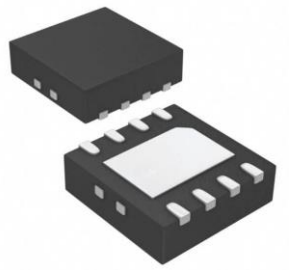


Figura 25. ATA6561-GBQW-N.
Fuente: (Augmented, 2021)

Características:

Tabla 18. ATA6561-GBQW-N.

Estilo montaje	SMD/SMT
Voltaje de alimentación operativo (V)	4.5 a 5
Temperatura de trabajo	-40 °C
Velocidad de transmisión de datos	5Mb/s
Numero de transceptores	1
Número de controladores	1
Producto	CAN Transceivers

Nota: A continuación, se muestra las características principales del ATA6561-GBQW-N.
Elaborado por: Los autores.

2.1.2.4 Matriz de comparación de Transceivers

Tabla 19. Matriz de características.

Características	MCP2551	MAX3051	ATA6561-GBQW-N
Voltaje de alimentación operativo (V)	5	3.3	4.5 A 5
Temperatura de trabajo	-40 °C	-40 °C	-40 °C
Velocidad de transmisión de datos	1	1	5Mb/s
Producto	CAN	CAN	CAN
Stock	Si	No	NO
Costo	$4.00	$2.10	$0.53

Nota: A continuación, se muestra un análisis de las características de los Transceivers.
Elaborado por: Los autores.

De acuerdo con la tabla 19, se utilizará el MCP2551 a razón que este Transceivers se encuentra en stock en el país, por lo tanto, su costo se mantiene a diferencia de los otros Transceivers donde su precio se elevaría por motivos de importación en hasta un 150% más, además que su compatibilidad con el microcontrolador seleccionado anteriormente es de un 100%, así permitiendo su conexión con una total confiabilidad.

2.1.4 Dispositivos IoT para comunicación con las Plataformas

2.1.4.1 Raspberry pi

Es un ordenador muy pequeño, podemos compararla al tamaño de una tarjeta o de un documento personal. Constituye de una lámina base donde se encuentra sus componentes principales (procesador, chip gráfico, memoria RAM). Fue lanzando en 2016 por la Fundación Raspberry Pi con el fin de que más personas puedan aprovechar el poder de la informática y las tecnologías digitales (Raspberrypi, 2021). En la figura 27, muestra la Raspberry pi 3b.

Raspberry

Figura 27. Raspberry 3b.
Fuente: (Raspberrypi, 2021)

Características:

Tabla 20. Raspberry pi 3b.

Procesador	Broadcom BCM2837, Cortex-A53 (ARMv8) 64-bit SoC
Frecuencia	1.4 GHz
RAM	1GB LPDDR2 SDRAM
Conectividad inalámbrica	2.4GHz / 5GHz IEEE 802.11.b/g/n/ac Bluetooth 4.2, BLE
Conectividad de red	Ethernet USB 2.0
Puertos	GPIO 40 pines HDMI 4 x USB 2.0 CSI (cámara Raspberry Pi) Toma auriculares / vídeo compuesto DSI (pantalla táctil) Micro SD Power-over-Ethernet (PoE)

Nota: A continuación, se muestra las características principales de Raspberry pi 3b.
Elaborado por: Los autores.

2.1.4.2. Esp32

Es un único chip compuesto de Wi-Fi y Bluetooth diseñado con ultra baja tecnología. Su rendimiento, robustez y potencia permite ser usada en una amplia gama de aplicaciones y escenarios diferentes. (Espressif, 2019). En la figura 28, muestra el Esp32.

Esp32

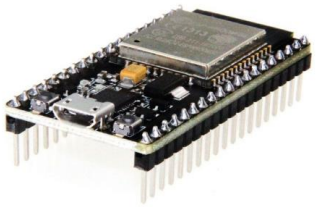


Figura 28. ESP32.
Fuente: (Prometec, 2021)

Características:

Tabla 21. Esp 32.

Microcontrolador	Dual Core Xtensa LX6 de 32 bits
Voltaje de entrada (V)	4
Pines de E/S digitales	36
UART	2
WIFI	Access Point & Station
Bluetooth	4.2 2.4 GHz; BT 2.0 y 4.0 Ble
RAM	520

Nota: A continuación, se muestra las características principales de Esp 32.
Elaborado por: Los autores.

2.1.4.3 Arduino nano 33 IoT

Es una placa diseñada para el desarrollo de tecnología IoT, su tamaño pequeño y sus distintas características hace que sea perfecta para mejorar a diferentes dispositivos con conectividad Wifi y Bluetooth, (Arduino, 2021). En la figura 29, muestra el Arduino Nano.

Arduino

Figura 29. Arduino Nano 33 IoT.
Fuente: (Arduino, 2021).

Características

Tabla 22. Arduino Nano 33 IoT.

Microcontrolador	ARM de baja potencia de 32 Bits
Voltaje de entrada (V)	21
Pines de E/S digitales	14
SPI	1
USB	Nativo en el procesador SAMd21
Pines de entrada analógico	8
Pines de salida analógico	1 (DAC de 10 Bits)

Nota: A continuación, se muestra las características principales de Arduino Nano 33 IoT.
Elaborado por: Los autores.

2.1.4.4 NodeMCU V3

Diseñado para proyectos y aplicaciones IoT, que posee tecnología con conectividad Wifi integrado, que alcanza una gran prestación con la mezcla de varias tecnologías patentadas.

Su funcionamiento se basa en el procesador ESP8266, que permite ser programado con Arduino IDE (ESP8266 Datasheet, 2015). En la figura 30, muestra el NodeMCU.

NodeMCU

Figura 30. Esp8266 wifi.
Fuente: (ESP8266 Datasheet, 2015)

Características

Tabla 23. Esp8266 wifi.

Voltaje de salida en los pines (V)	3.3
Voltaje de entrada USB (V)	5
Voltaje de referencia en el ADC (V)	3.3
Corriente nominal por PIN	12 Ma
Frecuencia de procesado	80MHz (160 MHz Max)
Flash	4Mb

Consumo de corriente en Standby	80MHz
Consumo de corriente	80MHz
Consumo de corriente HTTP	80MHz

Nota: A continuación, se muestra las características principales del NodeMCU Esp8266 wifi.
Elaborado por: Los autores.

3.1.1.4 Matriz de comparación de microcontroladores

Tabla 24. Matriz de características.

Características	ARDUINO NANO 33 IOT	Raspberry pi 3b	Node MCU ESP8266	ESP32
Microcontrolador	SAMD21 Cortex®-M0 + MCU ARM de baja potencia de 32 bits	Broadcom BCM2837, Cortex-A53 (ARMv8) 64-bits SoC	ESP8266	DUAL CORE XTENSA LX6 de 32 bits
E/S digitales	14	22	13	36
Voltaje de operación (V)	21	5	3.3	4
Entrada Analógicas	8	N/A	1	2
Ethernet	N/A	SI	N/A	N/A
HDMI	N/A	SI	N/A	N/A
Wireless	Si	Si	Si	Si
Sistema Operativo	N/A	Linux	N/A	N/A
USB	Nativo en el procesador SAMD21	3 USB 2.0	1 USB 2.0	1 USB 2.0
Costo ($)	$18.40	$90.00	$8.00	$10.00

Nota A continuación, se muestra un análisis de las características de los componentes.
Elaborado por: Los autores.

Tabla 25. Matriz de evaluación.

Evaluación	ARDUINO NANO 33 IOT	Raspberry pi 3b	Node MCU ESP8266	ESP32
Adaptabilidad al diseño	Si	No	Si	Si
Tamaño adaptable	Si	No	Si	Si
Puertos de Conexión	Si	Si	Si	Si

Costo bajo	Si	No	Si	Si

Nota: En la presente tabla se puede evaluar la adaptabilidad al diseño.
Elaborado por: Los autores.

De acuerdo con la tabla 25, se utilizará el Node MCU8266, por adaptabilidad, funcionamiento y reducido costo.

Nota: Se debe tener cuenta que todos los dispositivos deben estar cerca de una red WLAN.

2.2 Plataformas abiertas

Las plataformas abiertas son las que permiten construir o desarrollar sobre su entorno, brindan varios servicios de acuerdo con lo estipulado en el desarrollo de su aplicación.

Tener una plataforma abierta permitirá observar y controlar los diferentes componentes físicos que estén en desarrollo, tomando en cuenta su diferentes restricciones y limitaciones que puedan tener.

2.2.1 Ubidots

Es una plataforma que permite conectar dispositivos inteligentes mejor llamado como internet de las cosas (IOT), su funcionamiento se basa en la visualización y análisis de datos. Convierte los datos obtenidos de cada sensor en información para decisiones comerciales, máquina a máquina o educativa (Ubidots, 2021). En la figura 31, muestra la plataforma Ubidots.

Plataforma

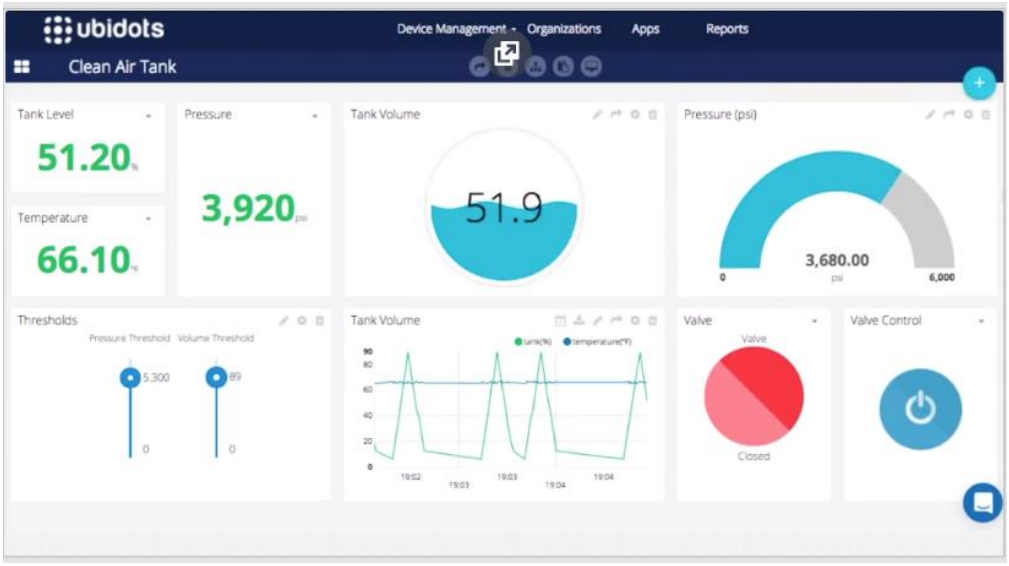


Figura 31. Ubidots.
Fuente: (Ubidots, 2021)

Utiliza protocolos de comunicación como:

- MQTT.
- HTTP.
- TCP.
- UDP.

Además de brindar la oportunidad de configurar alertas (EMAIL, SMS), según la clase de problema que registre el dispositivo conectado por ejemplo (Si alguna persona abre una chapa, o enciende la luz).

2.2.2 ThingSpeak

Es una plataforma de observación de IOT, que consiente en el estudio de datos en la nube en tiempo real. La comunicación proporciona información inmediata de los datos reflejados de los diferentes módulos conectados.

También ofrece diferentes aplicaciones que ayudan a la visualización y el análisis mediante el software MATLAB y actuar al momento sobre los datos (ThingSpeak, 2021). En la figura 32, muestra la plataforma ThingSpeak.

Plataforma

Figura 32. ThingSpeak.
Fuente: (ThingSpeak, 2021)

2.2.3 Thinger.io

Es una plataforma de IOT en la nube, que proporciona todas las herramientas necesarias para crear prototipos, escalar y administrar dispositivos de una manera muy sencilla a comparación con las demás plataformas existentes, además que su infraestructura de la nube es escalable.

Cuenta con una función donde mediante un token detectado por la aplicación del mismo nombre dentro de un celular inteligente, la página web se transforma en una aplicación que posee las mismo Apis en el teléfono que funciona de igual manera, ayudando con esto al mayor control de los dispositivos conectados (Thinger.io, 2020). En la figura 33, muestra la plataforma Thinger.Io.

Plataforma

Figura 33. Thinger.Io.
Fuente: (Thinger.io, 2020)

2.2.4 Blynk

Es una plataforma que permite crear de manera rápida interfaces simples y sencillas para controlar y monitorear proyectos de hardware desde celulares inteligentes (SMARTS).

Propiamente se debe descargar la aplicación, la cual permite crear un proyecto donde se puede crea un panel, botones, deslizantes, gráficos y otros widgets en la pantalla (Blynk, 2020). En la figura 34, muestra la plataforma Blynk.

Plataforma

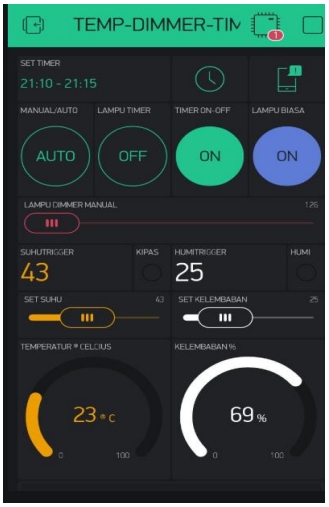


Figura 34. Blynk.
Fuente: (Blynk, 2020)

2.2.5 Matriz de adaptabilidad de plataformas

Tabla 26. Matriz de adaptabilidad.

Características	Ubidots	ThingSpeak	Thinger.io	Blynk
Aplicación Móvil	Si	Si	Si	Si
Monitoreo Web	Si	Si	Si	No
Datos tiempo real	Si	Si	Si	Si
Multiplataforma	No	No	Si	si
Almacenamiento de bases de datos	Si	Si	Si	No
Protocolos TCP/IP	SI	SI	SI	SI
Código abierto	No	Si	Si	Si
Escalabilidad	Si	Si	Si	No
Seguridad	No	No	No	No

Nota: En la presente tabla se puede evaluar la adaptabilidad.
Elaborado por: Los autores.

De acuerdo con la tabla 26, la plataforma que se utilizará es Thinger.Io, que es una plataforma amigable con el usuario, donde para este proyecto su versión gratuita y su gran escalabilidad cumple con el número de dispositivos que se van a controlar, además que nos proporciona un token para smartphones, que mediante la aplicación móvil propia de la plataforma se podrá controlar los actuadores y monitorizar los sensores.

2.3 Software

A continuación, se enunciará algunos programas que sirven para programar PICS y posteriormente se realizará una matriz comparativa .

2.3.1 Mikroc PRO for Pic

Es un compilador ANSI C con todas las funciones para dispositivos PIC de Microchip. Cuenta con un IDE intuitivo, un potente compilador con optimizaciones avanzadas, además de poseer una librería extensa de hardware y software y herramientas adicionales que permiten mejor la experiencia de trabajo (MikroE, 2021). En la figura 35, muestra el software Mikroc pic.

Software

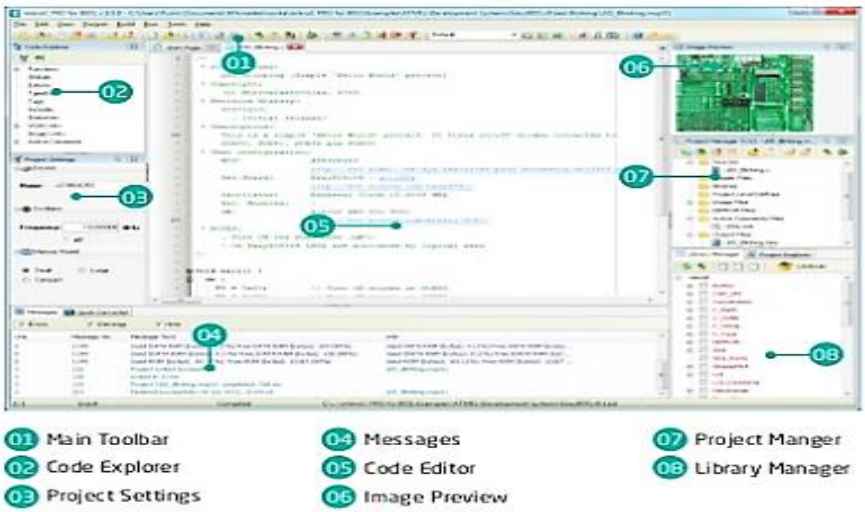


Figura 35. Mikroc pic.
Fuente: (MikroE, 2021)

2.3.2 MPLAB X IDE

Es un ambiente de desarrollo combinado creado por Microchip que nos permitirá llevar a cabo nuestros proyectos basados en microcontroladores PIC. Incluye todos los módulos necesarios para llevar a cabo todas las fases de nuestro proyecto, incluyendo la edición, ensamblaje, simulación y la programación en el microcontrolador (Technology, 2021). En la figura 36, muestra el software MPLAB X.

Software

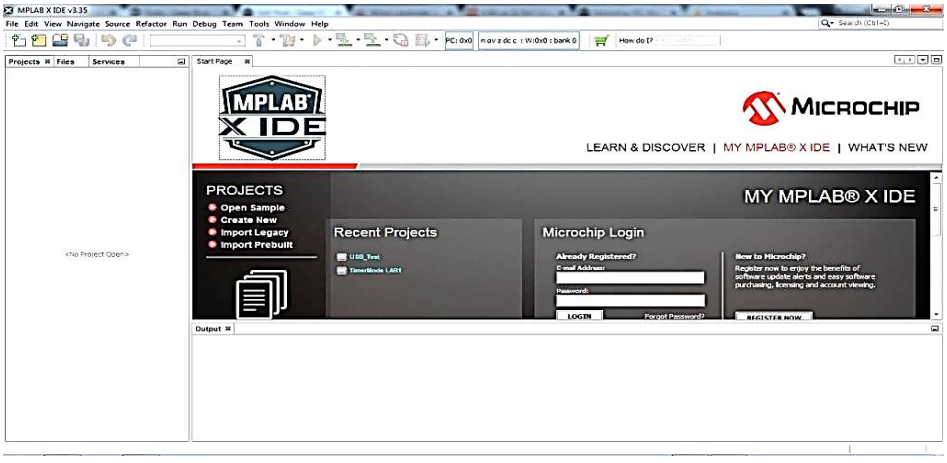


Figura 36. MPLAB X.
Fuente: (Technology, 2021)

2.3.3 Proton IDE

Proton IDE es un entorno de programación de nueva generación, encaminado al diseño de señales. El manejo de instrucciones de Proton admite la programación de microcontroladores PIC para manipular aplicaciones que llevan procesos. Las indicaciones de Basic son fáciles de interpretar que el lenguaje de Microchip (Associates, 2020). En la figura 37, muestra el software Proton IDE.

Software

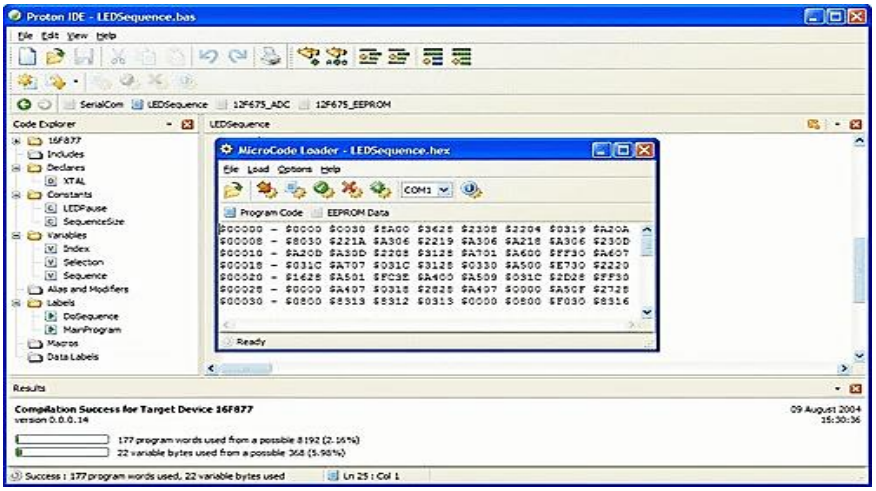


Figura 37. Proton IDE.
Fuente: (Associates, 2020)

2.3.4 Mikro Pascal Pro for Pic

Es un instrumento que trabaja con los chips de 8 bits de Microchip más sonados (PIC10, PIC12, PIC16, PIC18 y desde la versión V1.4, con el PIC16). El autor inicialmente lo creó para su uso personal porque pensó que otros compiladores iban mucho más allá que “un simple compilador de Pascal”, por otra parte, los compiladores comerciales resultaban muy onerosos en término de precios para alguien que quería usarlos a nivel personal (MikroE, 2021). En la figura 38, muestra el software Mikroc Pascal pic.

Software

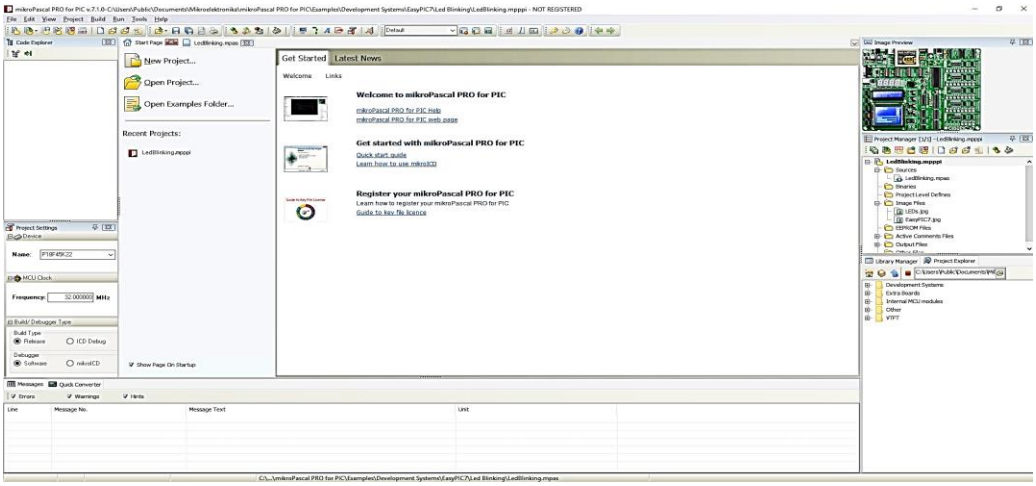


Figura 38. Mikroc Pascal Pic.
Fuente: (MikroE, 2021)

2.3.5 Matriz de adaptabilidad de software

Tabla 27. Matriz de adaptabilidad.

Características	Mikroc PRO for Pic	MPLAB X IDE	Mikro Pascal Pro for Pic	Proton IDE
Licencia libre	No	No	No	No
Lenguaje C	Si	No	Si	No
Amigable con el usuario	Si	No	Si	No
Fácil instalación	Si	No	Si	Si
Multiplataforma	Si	No	Si	No
Librerías CAN	SI	No	No	No

Nota: En la presente tabla se puede evaluar la adaptabilidad.
Elaborado por: Los autores.

De acuerdo con la tabla 27, se utilizará el programa MikroC Pro for pic por conocimiento previo a lenguaje C y su interfaz amigable con el usuario, además de contar con librerías CAN en sus extensas librerías que vienen incluidas dentro de su instalación. Su versión gratuita no se ve limitada para este proyecto.

2.4 Análisis de factibilidad

A continuación, se realizará un análisis de factibilidad al presente proyecto, que permitirá determinar si el desarrollo del prototipo planteado es viable para los desarrolladores de este, nos basaremos en 2 factores

- Factibilidad Técnica.
- Factibilidad Operacional.
- Factibilidad Económica.

2.4.1 Factibilidad técnica

En la tabla 28, se detallará los recursos tecnológicos y características que se han de utilizar en el desarrollo.

Tabla 28. Estudio de factibilidad para el desarrollo.

Recursos	Material	Características
IDE	Arduino IDE	Versión 1.8.9
	Mikro C PRO for pic	Version 7.6.0
Plataforma Abierta	Thinger.IO	Open Source
Módulos	NodeMCU8266	Wifi integrado
	MCP 2551	ISO 11898
	PIC18F2480	Soporta Red CAN
Sensores	Chapa	Trabaja a 12 V
	Ventilador	Ruido 21 db
	Proximidad	Detección de 2 a 30 cm
	Temperatura y Humedad	Salida de datos en serie
	Potenciómetro de precisión	Tolerancia +-10%
Fuente de alimentación	Batería	12V a 1A
Placa	PCB	Circuito
Estación de Trabajo	Computador	Sistemas Actualizados
	Celulares	Android, IOS

Nota: Características para el desarrollo del hardware y software.
Elaborado por: Los autores.

2.4.2 Factibilidad operacional

En este apartado se analizará la rentabilidad del prototipo desarrollado, con el fin que el producto que se ofrece sea funcional y operativo al momento de crearlo y desarrollarlo.

Para ello analizaremos los siguientes puntos:

- La innovación de tecnología que se usa en este prototipo viene orientado a cualquier ente de aplicación, ya sea en automotores, casas inteligentes. Donde el resultado es manejar ciertos dispositivos conectados a la internet, además de poder monitorizar en tiempo real un sensor de temperatura y humedad o simplemente desactivar o activar una chapa o ventilador.
- El usuario final tendrá la facilidad de conectarse a la internet y por medio de la plataforma abierta Thinger.Io observará y manipulará los sensores que tiene conectado al prototipo.

2.4.3 Factibilidad Económica

En la tabla 29, se considera todos los componentes (módulos, sensores, baterías, resistencias, placas) que se utilizaron en el desarrollo del prototipo además de considerar servicios de logística, insumos entre otros.

Tabla 29. Factibilidad económica para el prototipo.

Descripción	Cantidad	Costo Unitario $	Costo Total $
Programas			
Arduino IDE	1	0	0
Thinger.Io	1	0	0
Mikro c Pro for Pic	1	0	0
PICkit2	1	0	0
Módulos, Sensores, Microcontroladores			
MCUESP8266	1	8,00	8,00
MCP2551	1	4,00	4,00
PIC18F2480	3	8,38	25.14
Sensor Chapa	1	15,00	15,00
Sensor Ventilador	1	5,00	5,00
Sensor Temperatura y Humedad	1	2,00	2,00
Sensor Potenciómetro	1	2,00	2,00
Sensor de	1	3,00	3,00

Proximidad			
Fuente de alimentación	1	5,00	5,00
Zumbador alarma	1	2,00	2,00
LEDS	4	0.5	0.20
Materiales Usados			
Placas PCB	4	50,00	200
Caja de policarbonato	1	50,00	50,00
Desarrolladores			
Desarrolladores	2	500	1000
Gastos varios			
Resistencias, transistores, cables, etc.	-	20,00	20,00
Servicios básicos	-	15,00	90,00
		Total	1431.34

Nota: A continuación, se muestra el costo para el desarrollo.
Elaborado por: Los autores.

CAPÍTULO III

DISEÑO Y DESAROLLO

A continuación, se muestra el diseño y desarrollo del prototipo, de igual manera se figurará su funcionamiento con sus respetivos diagramas, de forma que sea intuitivo y se pueda observar los objetivos establecidos.

3.1 Diagrama de flujo

El diagrama de flujo comúnmente se lo aplica para describir el funcionamiento de los procesos o sistemas que se van a ejecutar, en este caso el del prototipo.

3.1.1 Diagrama de flujo de función general del prototipo

Diagrama de flujo

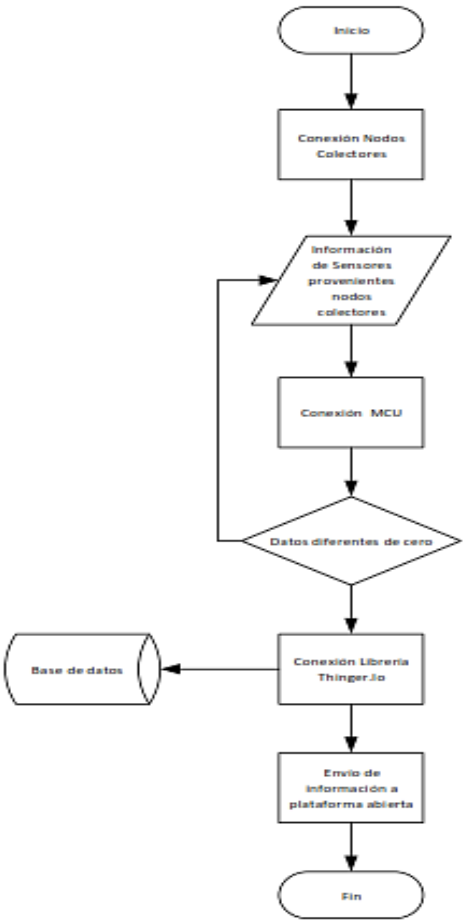


Figura 39. Diagrama de flujos de funcionamiento.
Elaborado por: Los autores.

En la figura 39, se detalla el proceso de la comunicación de los sensores a los nodos colectores y el envio de los datos por medio de la comunicación CAN, de tal forma que se pueda enviar al NodeMCU por transmisión serial, con la finalidad de que puedan llegar hacia la plataforma de Thinger.Io para su posterior visualización y control.

3.1 Arquitectura del prototipo

De acuerdo con el capítulo 3 se estableció el hardware y software que se utilizará en el prototipo a desarrollar, En la figura 40, se muestra el diagrama propuesto de la arquitectura que se va a implementar.

Diseño de arquitectura del prototipo

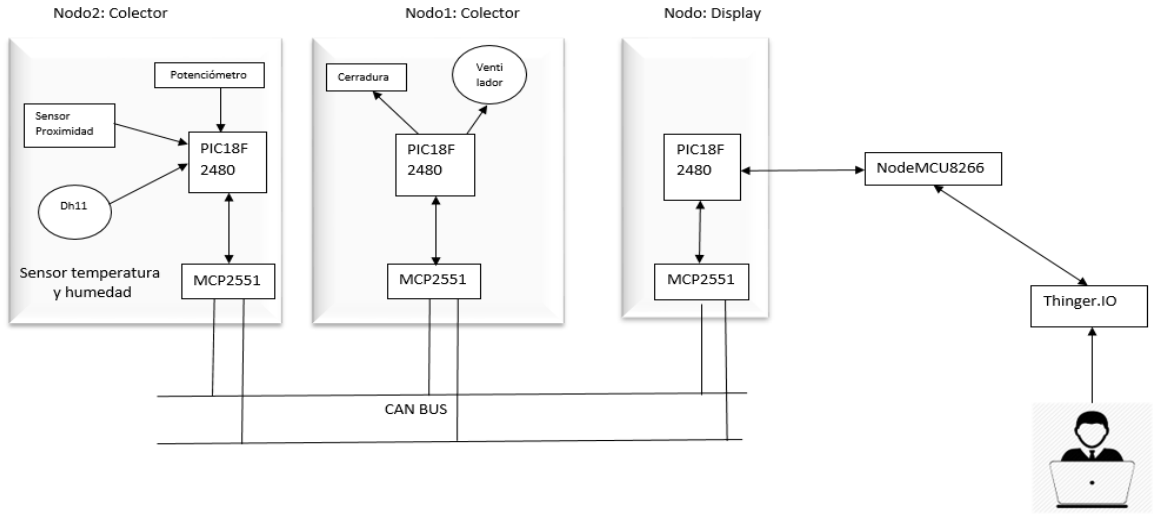


Figura 40. Arquitectura del prototipo.
Elaborado por: Los autores.

El sistema está compuesto por tres nodos CAN:

El nodo (llamado nodo DISPLAY), solicita la lectura de los sensores (DH11, potenciómetro y proximidad) a cada segundo y recibe peticiones para él control de los dispositivos (cerradura y ventilador) el cual trasmite los datos obtenidos al NodeMCU.

El nodo 1 (llamado nodo COLECTOR) recibe peticiones para los dispositivos (cerradura y ventilador).

El nodo 2 (llamado nodo COLECTOR) lee los datos de los sensores (DH11, potenciómetro y proximidad).

El NodeMCU se encarga de enviar los datos obtenidos del nodo DISPLAY por medio de la tecnología Wireless hacia la plataforma abierta.

Thinger.Io es la plataforma abierta donde el usuario final monitorea los sensores y controla los dispositivos ya sea vía web o su vez en su aplicación móvil en tiempo real.

3.2 Construcción del prototipo CAN IoT

En la construcción del prototipo se tomó en cuenta los componentes a utilizar verificando sus dimensiones y compatibilidad en las comunicaciones, de tal forma que se ubiquen en un lugar específico para su correcto funcionamiento.

3.2.1 Nodo COLECTOR 2

Colector

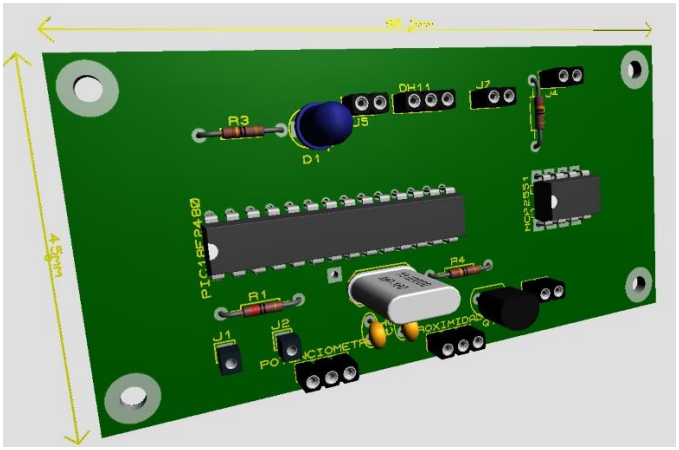


Figura 41. Nodo Colector 2.
Elaborado por: Los autores.

3.2.1.1 PIC18f2480

En la figura 41, se muestra el circuito integrado programable, el cual nos permite recibir datos de los sensores y enviarlos por medio del protocolo CAN.

3.2.2 Nodo COLECTOR 1

Colector

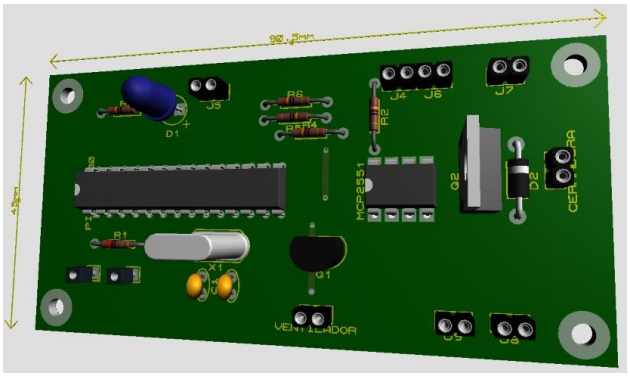


Figura 42. Nodo Colector 1.
Elaborado por: Los autores.

3.2.2.1 PIC18f2480

En la figura 42, se muestra el circuito integrado programable, el cual nos permite controlar los dispositivos IoT por medio del protocolo lo CAN,

3.2.3 Nodo DISPLAY

Nodo Display

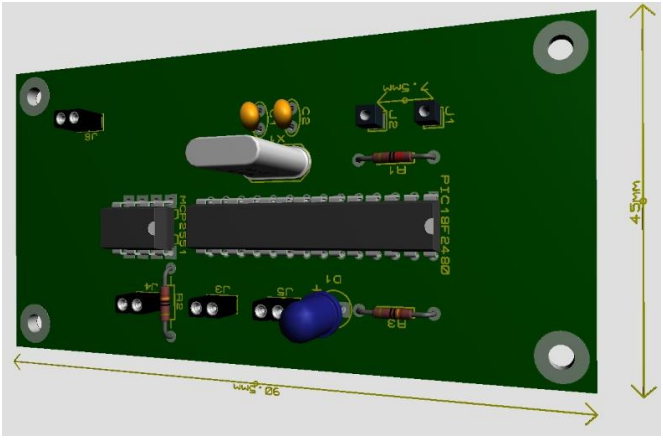


Figura 43. Nodo DISPLAY.
Elaborado por: Los autores.

3.2.3.1 PIC18f2480

En la figura 43, muestra al microcontrolador con la función de solicitar la lectura de los sensores DH11, potenciómetro, proximidad y recibe peticiones para el control de los dispositivos cerradura y ventilador.

3.2.4 NodeMCU

NodeMCU

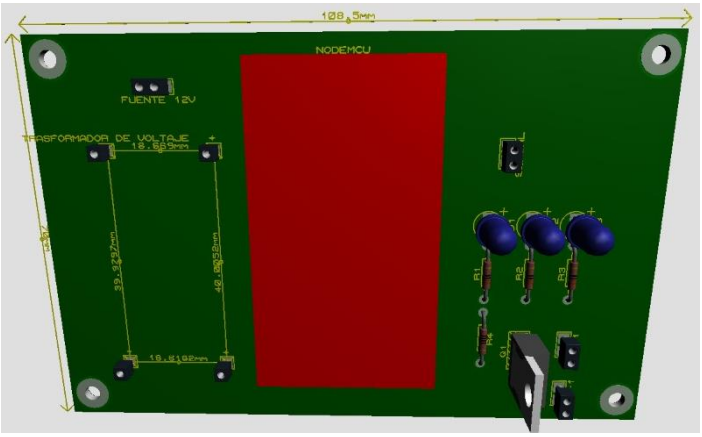


Figura 44. NodeMCU ESP8266.
Elaborado por: Los autores.

3.2.4.1 NodeMCU ESP8266

Es el encargado de recibir los datos obtenidos del Nodo DISPLAY y enviarlos hacia la plataforma abierta para su control y monitoreo de los dispositivos, permite tener comunicación por vía Wireless y serial. En la figura 44, muestra al NodeMCU ESP8266.

3.3 Diagrama de conectividad de los componentes

A continuación, se muestra los diagramas y códigos desarrollados para la elaboración del prototipo.

3.3.1 Nodos COLECTORES

Los nodos colectores son los que permiten recibir las peticiones del Nodo DISPLAY y a su vez solicita los datos de sus nodos, de modo que trabaja como un colector de datos.

3.3.1.1 Diagrama de COLECTOR 2

Nodo COLECTOR

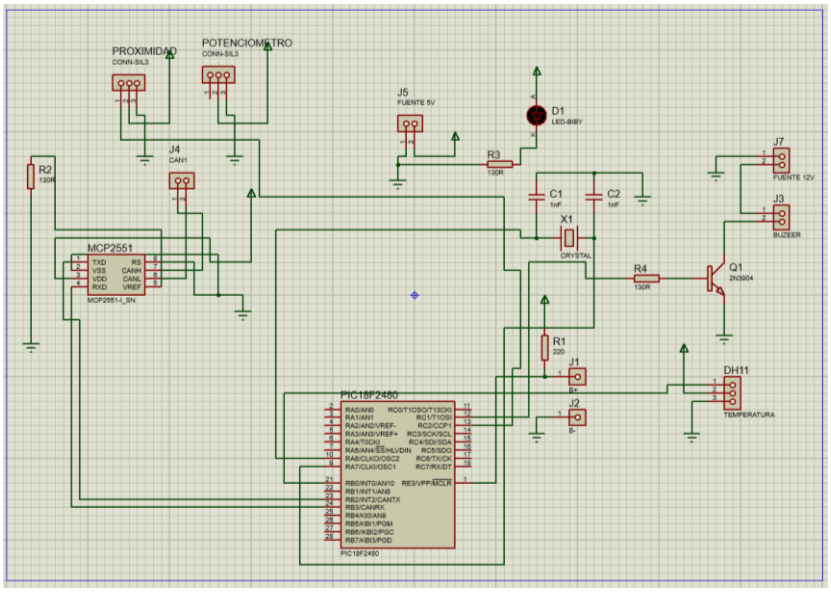


Figura 45. COLECTOR 2.
Elaborado por: Los autores.

En la figura 45, se observa las conexiones del nodo colector dos y su conexión apropiada con los diferentes componentes y sensores que van a enviar datos hacia el nodo colector Display.

Código de Programación COLECTOR 2 y sus sensores

Código de Programación

```
unsigned char Can_Init_Flags, Can_Send_Flags, Can_Rcv_Flags;
unsigned char Rx_Data_Len;
char RxTx_Data[8];
char RxTx_Data_3[8];
char RxTx_Data_4[8];
char RxTx_Data_5[8];
char RxTx_Data_6[8];
char RxTx_Data_7[8];
char RxTx_Data_8[8];
char Msg_Rcvd;
const long ID_1st = 12111, ID_2nd = 3, ID_3th= 4;
long Rx_ID;
unsigned int temperatura = 0;
unsigned int humedad = 0;
unsigned int potenciometro_crduo = 0;
unsigned int potenciometro = 0;
unsigned int digital = 166;
unsigned int i1 = 252;
unsigned int i2 = 253;
unsigned int i3 = 254;
unsigned int i4 = 255;
```

Código 1. Variables y banderas COLECTOR 2.
Elaborado por: Los autores.

En el código 1, se detalla la declaración de variables de los sensores del nodo COLECTOR 2 y de banderas CAN.

Código de Programación

```
Can_Init_Flags = 0;
Can_Send_Flags = 0;
Can_Rcv_Flags  = 0;

Can_Send_Flags = _CAN_TX_PRIORITY_0 &
                 _CAN_TX_XTD_FRAME &
                 _CAN_TX_NO_RTR_FRAME;

Can_Init_Flags = _CAN_CONFIG_SAMPLE_THRICE &
                 _CAN_CONFIG_PHSEG2_PRG_ON &
                 _CAN_CONFIG_XTD_MSG &
                 _CAN_CONFIG_DBL_BUFFER_ON &
                 _CAN_CONFIG_VALID_XTD_MSG &
                 _CAN_CONFIG_LINE_FILTER_OFF;

CANInitialize(1,3,3,3,1,Can_Init_Flags);
CANSetOperationMode(_CAN_MODE_CONFIG,0xFF);
CANSetMask(_CAN_MASK_B1,-1,_CAN_CONFIG_XTD_MSG);
CANSetMask(_CAN_MASK_B2,-1,_CAN_CONFIG_XTD_MSG);
CANSetFilter(_CAN_FILTER_B2_F3,ID_1st,_CAN_CONFIG_XTD_MSG);
CANSetOperationMode(_CAN_MODE_NORMAL,0xFF);
```

Código 2. Librería CAN.
Elaborado por: Los autores.

En el código 2, se detalla la declaración de banderas CAN (iniciación y envio) cada una con sus diferentes parámetros utilizados para la comunicación.

Código de Programación

```
while (1) {
    Dht11_Start();
    DHT11_Read();
   if(DHT11_CHKSM==((DHT11_TMP>>8)+(DHT11_HUM>>8)
    +(DHT11_TMP&0xff)+(DHT11_HUM&0xff))){
    temperatura = DHT11_TMP>>8;
    humedad = DHT11_HUM>>8;
     if(temperatura >= 30){
      portc.f1 = 1;
     }else{
      portc.f1 = 0;
     }
   }
   if(portc.f2 == 1){
    digital = 167;
   }else{
    digital = 166;
   }
   potenciometro_crduo = ADC_Read(0) / 8;
   potenciometro = (potenciometro_crduo * 100) / 127;
   Delay_ms(100);
```

Código 3. Inicio de sensores.
Elaborado por: Los autores.

En el código 3, se detalla dentro del ciclo WHILE la lectura de los sensores (DH11, proximidad, potenciómetro), el primer sensor se divide en una condición IF donde se divide 8bits para temperatura y 8bits para humedad, donde la temperatura al ser mayor de 30° se activa la alerta, el segundo sensor censa y hace una comparación si es igual a 1 se prende caso contrario se apaga y el tercer sensor utiliza la función ADC_Read que es el puerto análogo-digital del PIC y el valor se ajusta en una escala del 0 al 100%.

Código de Programación

```
RxTx_Data_5[0] = i1;
   CANWrite(ID_3th, RxTx_Data_5, 1, Can_Send_Flags);
   Delay_ms(350);
   RxTx_Data[0] = temperatura;
   CANWrite(ID_3th, RxTx_Data, 1, Can_Send_Flags);
   Delay_ms(350);

   RxTx_Data_6[0] = i2;
   CANWrite(ID_3th, RxTx_Data_6, 1, Can_Send_Flags);
   Delay_ms(350);
   RxTx_Data_2[0] = humedad;
   CANWrite(ID_3th, RxTx_Data_2, 1, Can_Send_Flags);
   Delay_ms(350);

   RxTx_Data_7[0] = i3;
   CANWrite(ID_3th, RxTx_Data_7, 1, Can_Send_Flags);
   Delay_ms(350);
   RxTx_Data_3[0] = potenciometro;
   CANWrite(ID_3th, RxTx_Data_3, 1, Can_Send_Flags);
   Delay_ms(350);

   RxTx_Data_8[0] = i4;
   CANWrite(ID_3th, RxTx_Data_8, 1, Can_Send_Flags);
   Delay_ms(350);
   RxTx_Data_4[0] = digital;
   CANWrite(ID_3th, RxTx_Data_4, 1, Can_Send_Flags);
   Delay_ms(350);
```

Código 4. Envío con Banderas.
Elaborado por: Los autores.

En el código 4, se detalla el código de envío con sus respetivos datos y sus banderas del protocolo CAN hacia el nodo DISPLAY.

3.3.1.2 Diagrama de COLECTOR 1

Diagrama de circuitos: Nodo COLECTOR

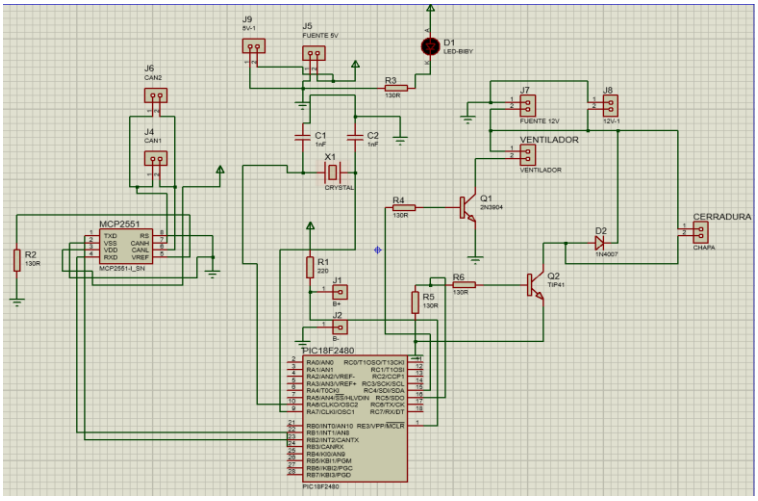


Figura 46. COLECTOR 1.
Elaborado por: Los autores.

En la figura 46, se observa las conexiones del nodo colector1 y su conexión apropiada con los diferentes componentes y actuadores que van a enviar datos hacia el nodo colector Display.

Código de Programación COLECTOR 1 y sus dispositivos IoT

Código de Programación

```
unsigned char Can_Init_Flags, Can_Send_Flags, Can_Rcv_Flags;
unsigned char Rx_Data_Len;
char RxTx_Data[8];
char Msg_Rcvd;
const long ID_1st = 12111, ID_2nd = 3;
long Rx_ID;
void main() {
  PORTC = 0;
  TRISC = 0;
  Can_Init_Flags = 0;
  Can_Send_Flags = 0;
  Can_Rcv_Flags = 0;
  Can_Send_Flags = _CAN_TX_PRIORITY_0 &
                   _CAN_TX_XTD_FRAME &
                   _CAN_TX_NO_RTR_FRAME;
  Can_Init_Flags = _CAN_CONFIG_SAMPLE_THRICE &
                   _CAN_CONFIG_PHSEG2_PRG_ON &
                   _CAN_CONFIG_XTD_MSG &
                   _CAN_CONFIG_DBL_BUFFER_ON &
                   _CAN_CONFIG_VALID_XTD_MSG &
                   _CAN_CONFIG_LINE_FILTER_OFF;
  CANInitialize(1,3,3,3,1,Can_Init_Flags);
  CANSetOperationMode(_CAN_MODE_CONFIG,0xFF);
  CANSetMask(_CAN_MASK_B1,-1,_CAN_CONFIG_XTD_MSG);
```

Código 5. Variables y banderas COLECTOR 1.
Elaborado por: Los autores.

En el código 5, se detalla la declaración de variables de los dispositivos del nodo COLECTOR 2 y de banderas CAN.

Código de Programación

```
while (1) {
    Msg_Rcvd = CANRead(&Rx_ID , RxTx_Data , &Rx_Data_Len, &Can_Rcv_Flags);
     if(RxTx_Data[0] == 6){
      portc.f4 = 1;}
     if(RxTx_Data[0] == 7){
      portc.f4 = 0;}
     if(RxTx_Data[0] == 8){
      portc.f5 = 1;}
```

Código 6. Lecturas nodo COLECTOR 1.
Elaborado por: Los autores.

En el código 6, se detalla dentro del ciclo WHILE la lectura de los dispositivos (cerradura y ventilador), donde existen condiciones que se emplean para que los dispositivos llegan a funcionar de manera eficiente.

3.3.2 Nodos DISPLAY

El nodo DISPLAY es el que está encargado de recibir los datos del nodo COLECTOR 2 donde se conectan los sensores y a su vez enviar las peticiones al COLECTOR 1 donde se encuentran los dispositivos IoT para su control.

3.3.2.2 Diagrama de Nodo DISPLAY

Nodo Display

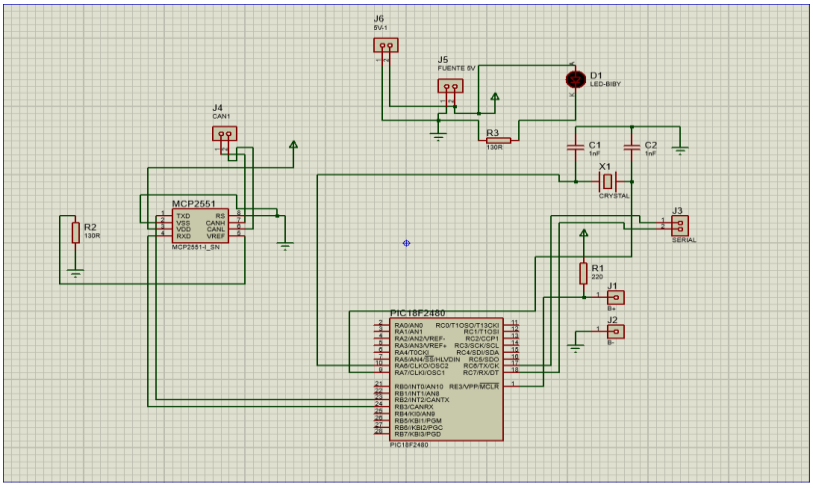


Figura 47. Nodo Display.
Elaborado por: Los autores.

En la figura 47, se observa la conexión del microcontrolador principal a los diferentes nodos (colector1 y colector 2), este nodo Display realiza peticiones a los diferentes nodos para el intercambio de información con el dispositivo de envío.

Código de Programación nodo DISPLAY y sus comunicaciones

Código de Programación

```
unsigned char Can_Init_Flags, Can_Send_Flags, Can_Rcv_Flags;
unsigned char Rx_Data_R_Len;

char RxTx_Data[8];
char RxTx_Data_2[8];
char RxTx_Data_4[8];
char RxTx_Data_6[8];
char RxTx_Data_R[8];

char Msg_Rcvd;
const long ID_1st = 12111, ID_2nd = 3, ID_3th = 4;
long Rx_ID;

unsigned int temperatura = 0;
char temperatura_txt[25];
unsigned int humedad = 0;
char humedad_txt[25];

unsigned int dato_recibido = 0;
char dato_recibido_txt[25];
char recibido_serial;
int contador = 0;
```

Código 7. Variables y dispositivos.
Elaborado por: Los autores.

En el código 7, se detalla las banderas CAN, la longitud de los datos recibidos en bytes, los nodos ID y las variables de los sensores y dispositivos.

Código de Programación

```
void main() {

  trisc.f3 = 1;
  trisc.f4 = 1;
  trisc.f5 = 0;
  portc.f5 = 0;
  UART1_Init(9600);
  Delay_ms(100);
  Can_Init_Flags = 0;
  Can_Send_Flags = 0;
  Can_Rcv_Flags = 0;
```

Código 8. Limpieza y salida de puertos.
Elaborado por: Los autores.

En el código 8, se detalla la limpia de puertos y se configuran como puertos de salida.

Código de Programación

```
RxTx_Data[0] = 6;
  RxTx_Data_2[0] = 7;
  RxTx_Data_4[0] = 8;
  RxTx_Data_6[0] = 9;
  RxTx_Data_R[0] = 0;
  RxTx_Data_R[4] = 0;
  UART1_Write_Text("Start");
  UART1_Write(10);
  UART1_Write(13);
```

Código 9. Valores de dispositivos.
Elaborado por: Los autores.

En el código 9, se detalla los valores asignados a los dispositivos.

Código de Programación

```
while(1) {
    if (UART1_Data_Ready()) {
      recibido_serial = UART1_Read();}
    if(recibido_serial == 'a'){
      CANWrite(ID_1st, RxTx_Data, 1, Can_Send_Flags);}
    if(recibido_serial == 'b'){
      CANWrite(ID_1st, RxTx_Data_2, 1, Can_Send_Flags);}
    if(recibido_serial == 'c'){
      CANWrite(ID_1st, RxTx_Data_4, 1, Can_Send_Flags);}
    if(recibido_serial == 'd'){
      CANWrite(ID_1st, RxTx_Data_6, 1, Can_Send_Flags);}

  Msg_Rcvd = CANRead(&Rx_ID , RxTx_Data_R , &Rx_Data_R_Len, &Can_Rcv_Flags);
   if ((Rx_ID == ID_3th) && Msg_Rcvd) {
    dato_recibido = RxTx_Data_R[0];
    ByteToStr(dato_recibido,dato_recibido_txt);
    UART1_Write_Text(dato_recibido_txt);
    UART1_Write(10);
    UART1_Write(13);
```

Código 10. Lectura y escritura nodo DISPLAY.
Elaborado por: Los autores.

En el código 10, se detalla dentro del ciclo WHILE la lectura y escritura de datos que se reciba tanto por CAN y seriales de los dispositivos y sensores con su respetivo ID, los datos recibidos en bytes se transforman en STRING para ser enviados al NodeMCU.

3.3.3 NodeMCU

El NodeMCU es el encargado de recibir los datos que obtiene el nodo DISPLAY para poderlos procesar, enviar y visualizar en la plataforma abierta de modo intuitivo para el usuario.

3.3.3.1 Diagrama NodeMCU ESP8266

NodeMCU

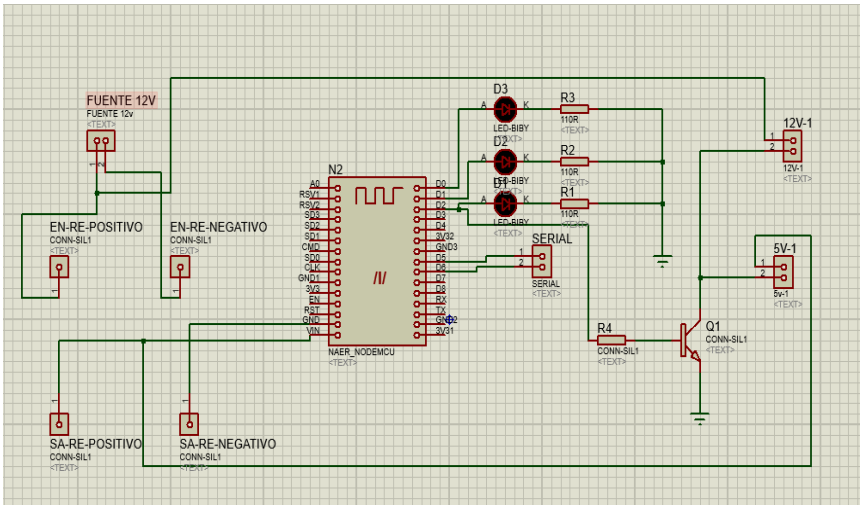


Figura 48. NodeMCU ESP8266.
Elaborado por: Los autores.

En la figura 48, se observa la conexión del dispositivo de envío a una fuente de alimentación, leds y la conexión al nodo colector Display mediante conexión serial para la comunicación de datos.

Código de Programación NodeMCU ESP8266

Código de programación

```
#include <SoftwareSerial.h>
#include <ThingerESP8266.h>
#define USERNAME "andidami"
#define DEVICE_ID "NODO_CAN"
#define DEVICE_CREDENTIAL "Kw5MiC!l5zWP"
#define SSID "**andidami**"
#define SSID_PASSWORD "*********"
#define ventilador 16
#define chapa 5
#define activa_can 4
#define digital 15
SoftwareSerial CAN_Rx_Tx(12, 14); // RX, TX
char cadena_recibida [10];
byte posicion = 0;
int dato_recibido_int = 0;
int temperatura = 0;
int humedad = 0;
int potenciometro = 0;
int estado = 0;
```

Código 11. Definición de librerías.
Elaborado por: Los autores.

En el código 11, se detalla el llamado a las librerías de Thinger.Io, conexión y declaración de las variables.

Código de programación

```
thing.add_wifi(SSID, SSID_PASSWORD);
  thing["Ventilador"] << digitalPin(16);
  thing["Chapa"] << digitalPin(5);
  thing["Digital"] >> digitalPin(digital);
  thing["Temperatura"] >> outputValue(temperatura);
  thing["Humedad"] >> outputValue(humedad);
  thing["Potenciometro"] >> outputValue(potenciometro);
```

Código 12. Designación para Thinger.Io.
Elaborado por: Los autores.

En el código 12, se detalla la designación de pines del NodeMCU para que la plataforma de Thinger.Io los pueda reconocer.

Código de Programación

```
if(digitalRead(ventilador) == HIGH){
    //Serial.println("a");
    CAN_Rx_Tx.println("a");
   }else{
    //Serial.println("b");
    CAN_Rx_Tx.println("b");}
if(digitalRead(chapa) == HIGH){
    //Serial.println("c");
    CAN_Rx_Tx.println("c");
   }else{
    //Serial.println("d");
    CAN_Rx_Tx.println("d");}
if(dato_recibido_int == 252){
        contador = 1;}
if(dato_recibido_int == 253){
        contador = 2;}
if(dato_recibido_int == 254){
        contador = 3;}
if(dato_recibido_int == 255){
        contador = 4;}
```

Código 13. Lectura y escritura NodeMCU.
Elaborado por: Los autores.

En el código 13, se detalla la escritura para los dispositivos el cual nos permite enviarlos por transmisión serial hacia la comunicación CAN y del mismo modo con la lectura de datos recibidos de los sensores con sus diferentes inicializadores a quien corresponde.

Código de programación

```
if(contador == 1 && dato_recibido_int != 252){
        Serial.print("TEMPERATURA: ");
        Serial.println(dato_recibido_int);
        temperatura = dato_recibido_int;
        contador = 0;}
if(contador == 2 && dato_recibido_int != 253){
        Serial.print("HUMEDAD: ");
        Serial.println(dato_recibido_int);
        humedad = dato_recibido_int;
        contador = 0;}
if(contador == 3 && dato_recibido_int != 254){
        Serial.print("POTENCIOMETRO: ");
        Serial.println(dato_recibido_int);
        potenciometro = dato_recibido_int;
        contador = 0;}
if(contador == 4 && dato_recibido_int != 255){
        Serial.print("ESTADO: ");
        Serial.println(dato_recibido_int);
        estado = dato_recibido_int;
        contador = 0;}
if(estado == 166){
        digitalWrite(digital,HIGH);
        Serial.println("ON");}
if(estado == 167){
        digitalWrite(digital,LOW);
        Serial.println("OFF");
```

Código 14. Llegada de datos.
Elaborado por: Los autores.

En el código 14, se detalla el código para garantizar la llegada de los datos de lectura desde la comunicación CAN y la escritura para el control de los dispositivos.

3.3.4 Estructura de Thinger.Io

Estructura

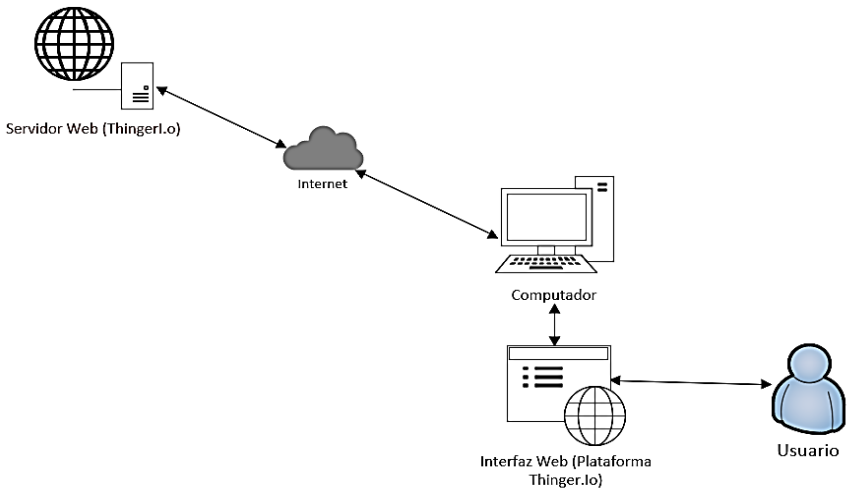


Figura 49. Estructura de Thinger.Io.
Elaborado por: Los autores.

En la figura 49, se detalla el esquema de la estructura de la Plataforma Abierta (Thinger.Io), la misma que se utiliza de la internet para el envío y recepción de datos, estos datos se pueden observar y manipular por medio de un equipo físico que ayuda a la visualización de su interfaz web.

CAPÍTULO IV

PRUEBAS Y RESULTADOS

A continuación, se describe las pruebas correspondientes al prototipo CAN IoT, a nivel de hardware y software para la comprobación de su apropiado funcionamiento.

4.1 Pruebas de funcionamiento

4.1.1 Pruebas de funcionamiento a nivel hardware

Para verificar el encendido y funcionamiento del prototipo, se procede a conectar su respectiva fuente de poder, la misma que activara el NodeMCU, y las comunicaciones entre nodos conectores, sus sensores y dispositivos.

Fuente de poder

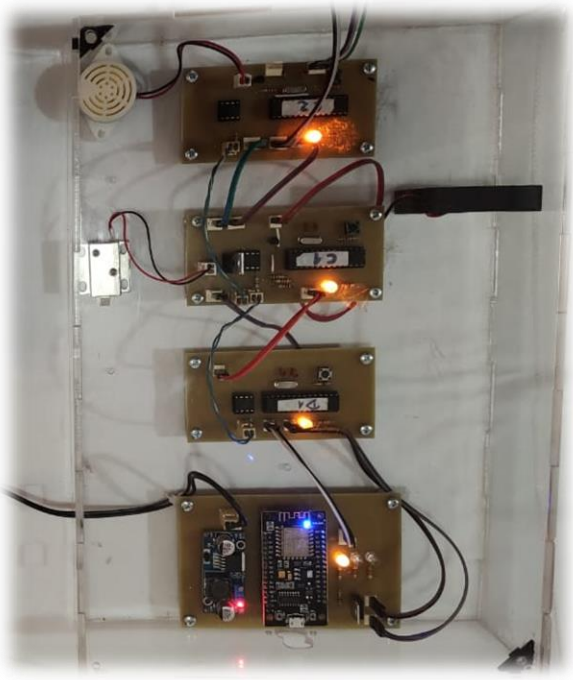


Figura 50. Alimentación con la fuente de poder.
Elaborado por: Los autores.

En la figura 50, se observa la implementacion de los nodos CAN y dispositivo de envío hacia la plataforma, el cual al verificar la comunicación decimos que es efectivo en un 100%.

4.2 Pruebas de ingreso al servidor

Prueba de ingreso

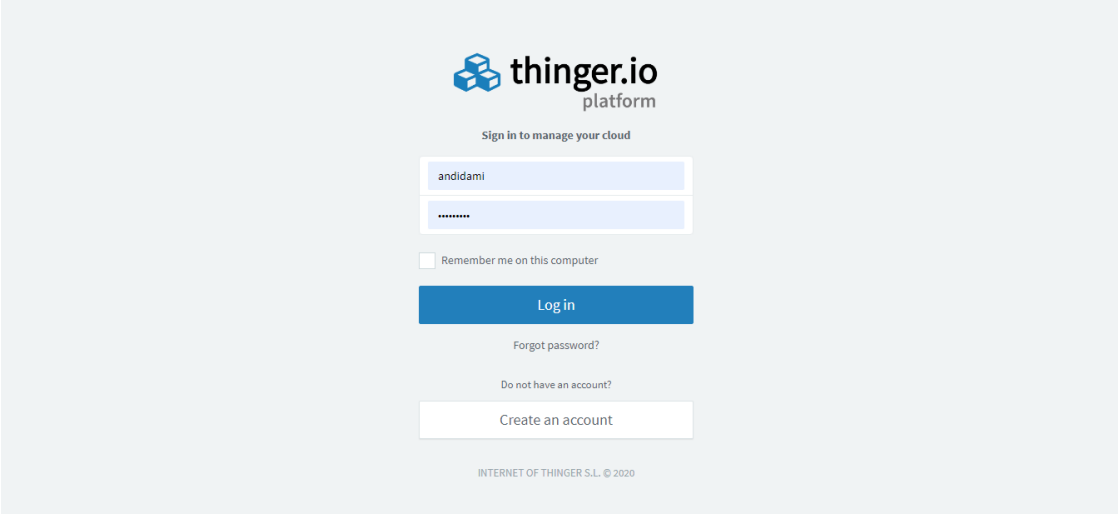


Figura 51. Ingreso de credenciales a la plataforma.
Elaborado por: Los autores.

En la figura 51, se observa que la plataforma Thinger.Io solicita que el usuario ingrese sus credenciales al poseer una cuenta o al contrario registrarse en la plataforma.

Prueba de ingreso

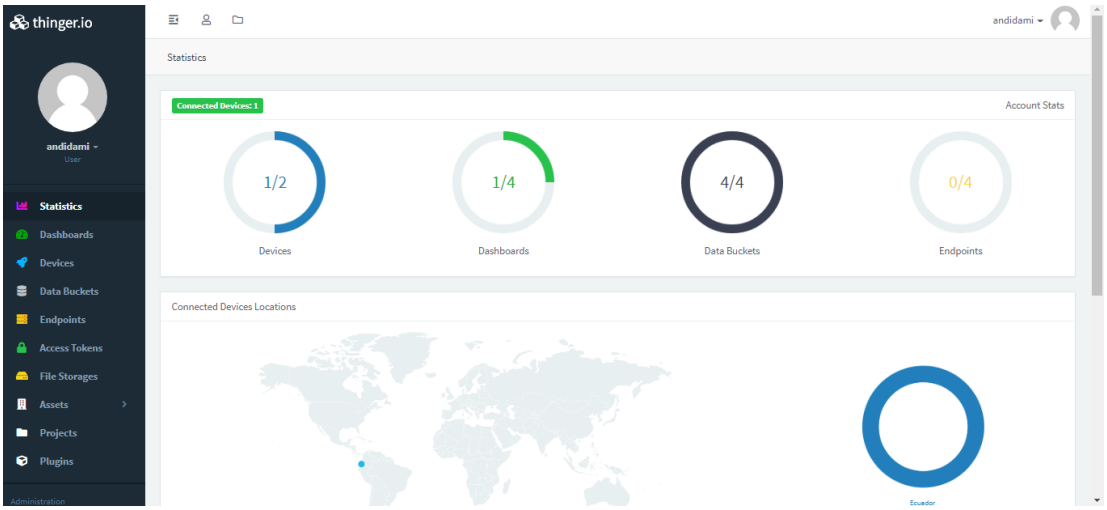


Figura 52. Ingreso principal Thinger.Io.
Elaborado por: Los autores.

En la figura 52, se observa el estado de nuestros dispositivos, en este caso el del prototipo, su interfaz gráfica y el almacenamiento de los datos que están levantados. La conexión que maneja la plataforma hacia nuestro dispositivo de envío se maneja por librerías, donde la plataforma

detecta automáticamente los actuadores o sensores que estén conectados en nuestro prototipo y permite así su monitoreo y su control.

4.3 Pruebas de Conectividad

Plataforma

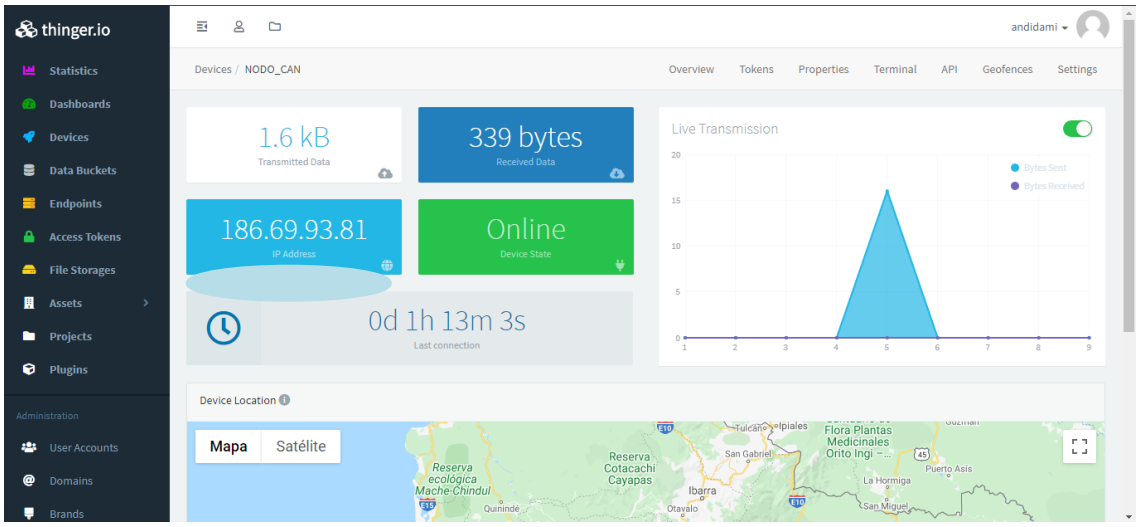


Figura 53. Comprobación de comunicaion.
Elaborado por: Los autores.

En la figura 53, se observa que el prototipo al encenderlo se conecta a la plataforma de Thinger.Io de manera que se puede ratificar su buen funcionamiento, se procede a realizar la comprobación de conectividad entre el prototipo y la plataforma Thinger.Io, para la integración de los nodos del prototipo, sus interfaces y los datos almacenados en el mismo.

Prueba de dispositivos

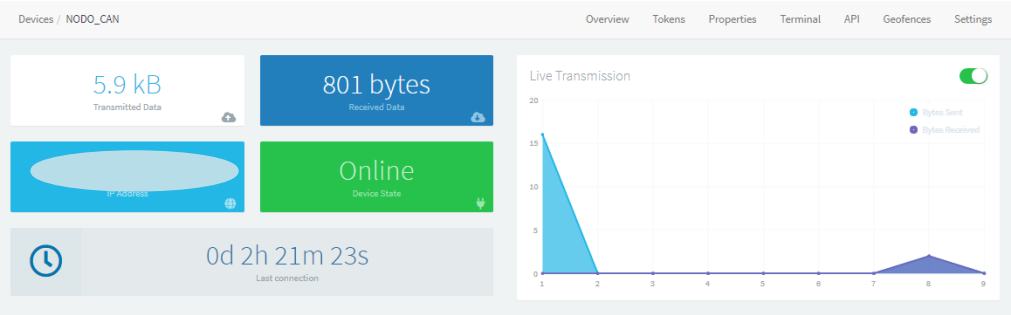


Figura 54. Pruebas de dispositivo.
Elaborado por: Los autores.

En la figura 54, se observa la conectividad de los dispositivos en la plataforma y su emparejamiento en línea, además, sus etiquetas para el control, tanto a nivel web como en la App. En la figura 55, se muestra que la plataforma Thinger.Io mediante un token, puede tener el control del prototipo desde un smartphone, mediante su aplicación móvil.

Aplicación Android

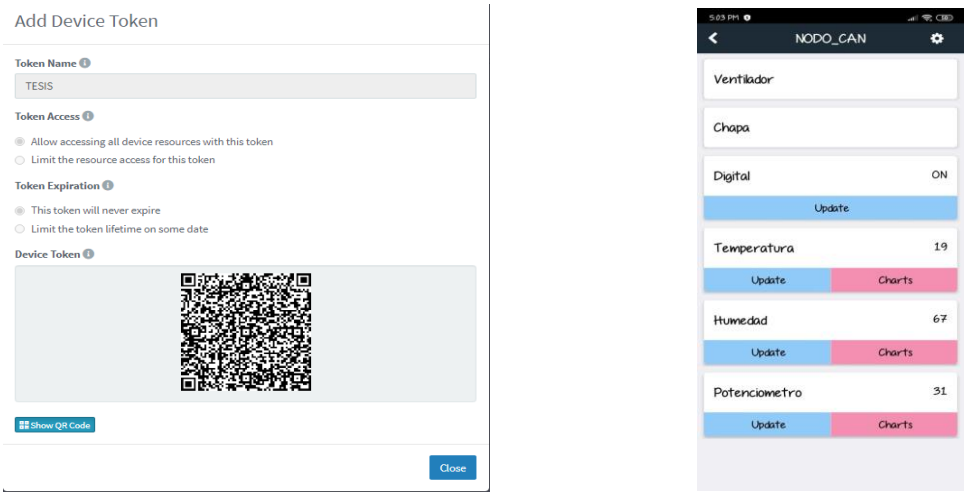


Figura 55. Aplicación Android.
Elaborado por: Los autores.

4.4 Pruebas de Dashboard (Interfaz)

Prueba de Dashboard

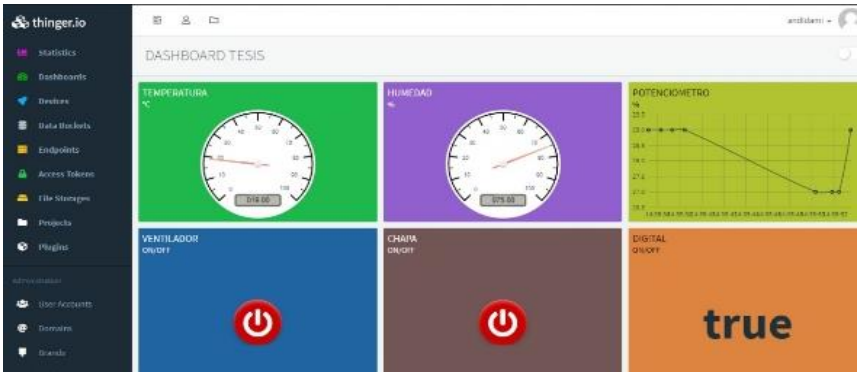


Figura 56. Pruebas de Dashboard.
Elaborado por: Los autores.

En la figura 56, se muestra la interfaz al usuario para el control y monitoreo de los sensores del prototipo, con sus respectivas etiquetas de cada sensor o actuador conectado.

4.5 Pruebas de almacenamiento

Prueba de almacenamiento

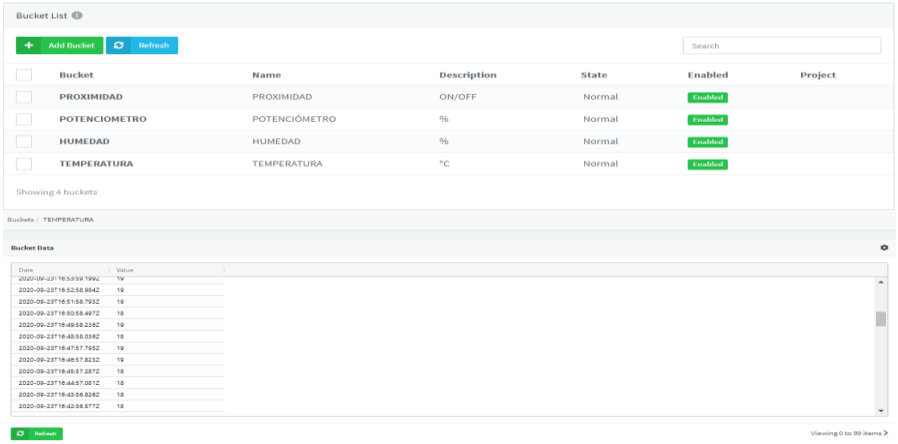


Figura 57. Pruebas de almacenamiento.
Elaborado por: Los autores.

En la figura 57, se muestra los datos obtenidos en el monitoreo de los sensores que se almacenan en la base de datos con su respectiva hora y fecha, asimismo se permite tener un registro exacto de los datos.

4.6 Resultados de plataforma Thinger.Io

Para los siguientes resultados se procede a verificar las pruebas realizadas a la plataforma Thinger.Io en el sistema para el control y monitoreo mediante los indicadores establecidos, asimismo se valorará de acuerdo con la tabla 30, su desempeño en el funcionamiento.

Tabla 30. Escala de valores.

Valores del cumplimento de Thinger.Io		Observaciones
98-100	Se cumple excelentemente	Conservar
75-97	Se cumple normalmente	Posibilidad de mejorar
60-74	Se cumple probablemente	Necesidad de revisar
40-59	Se cumple pobremente	Descartar o profundizar
Menor a 40	No se cumple	Descartar definitivamente

Nota: Valores del cumplimiento de objetivos establecidos.
Elaborado por: Los autores.

4.6.1 Indicadores establecidos

A continuación, se establecen los indicadores:

- Funcionamiento del ingreso al servidor.
- Funcionamiento de conectividad.
- Funcionamiento de dashboard.
- Funcionamiento de App móvil.
- Funcionamiento del almacenamiento.

4.6.2 Valoración Final

Se procede a calificar a la plataforma Thinger.Io con los indicadores establecidos para obtener la efectivad en el prototipo.

Tabla 31. Tabla de indicadores.

Tabla de indicadores		
Indicadores	**%**	Observaciones
Funcionamiento del ingreso al servidor	100	Funciona correctamente.
Funcionamiento de conectividad	100	Funciona según lo planteado.
Funcionamiento de Dashboard	100	Funciona correctamente.
Funcionamiento de App móvil	90	Limitación una su aplicación móvil.
Funcionamiento del almacenamiento	98	Retraso en los datos.
Suma	488	
Total 100%	97.6	

Nota: Muestra la asignación de valores de acuerdo con los indicadores.
Elaborado por: Los autores.

En la tabla 31, se indica la calificación de los indicadores conforme con la plataforma, así también su respectiva observación para cada valoración en las pruebas realizadas.

Tabla 32. Tabla de resultados Thinger.Io.

Valoración	Base	Resultado
Resultado	**100**	**97.6**
Objetivo		**100.0**
% Funcionamiento		**97.6%**

Nota: Tabla de resultados para valorar lo objetivos del 100%.
Elaborado por: Los autores.

En la tabla 32, se detalla los valores obtenidos para evaluar el funcionamiento de la plataforma, basándonos en los indicadores y los objetivos planteados, asimismo dándonos como resultado el

97,6% del cumplimiento de la plataforma en el prototipo, el cual nos da la posibilidad de mejorar en el funcionamiento de la App móvil y en el almacenamiento.

4.7 Obtención de datos desde el prototipo hacia la plataforma

A continuación, se procede examinar los datos obtenidos que arrojan los sensores del prototipo hacía la plataforma, la misma que guarda con detalle los datos obtenidos.

4.7.1 Resultados de datos del funcionamiento de sensores

Datos sensor Temperatura

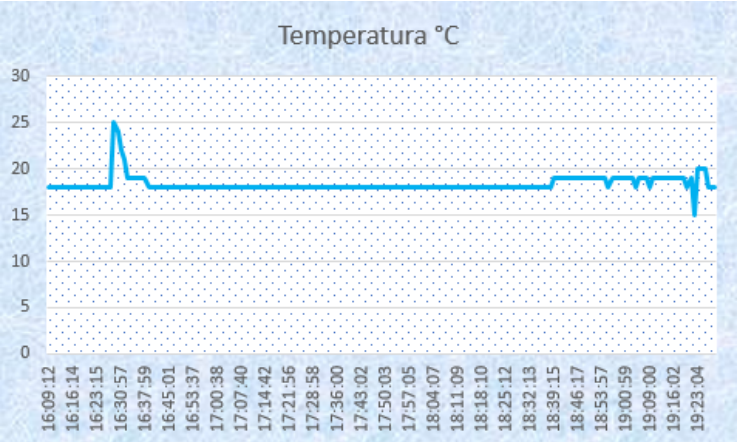


Figura 58. Resultados de Temperatura.
Elaborado por: Los autores.

Tabla 33. Tabla de datos principales en intervalos de temperatura.

Hora	Dato	Nivel de pico
16:12h	18°C	Promedio
16:30h	25°C	Alto
17:21h	18°C	Promedio
18:39h	19°C	Promedio
19:23h	15°C	Bajo

Nota: Tabla de datos para valorar intervalos de temperatura.
Elaborado por: Los autores.

En la figura 58, se observa los datos obtenidos del sensor de temperatura, en un lapso de 3 horas con intervalo de 1 minuto, con una eficiencia de llegada de 98%. En la tabla 33, se describe los datos obtenidos en la gráfica de temperatura, donde, se observa un promedio de 18°C, también se evidencia que a las 16:30h se produjo un evento, el cual nos da un pico alto de 25°C, donde se

refleja la subida de temperatura en el sensor, asimismo, a las 19:23h se evidencia otro evento donde se sometió a una prueba inversa al sensor dándonos un 15°C.

Datos sensor Humedad

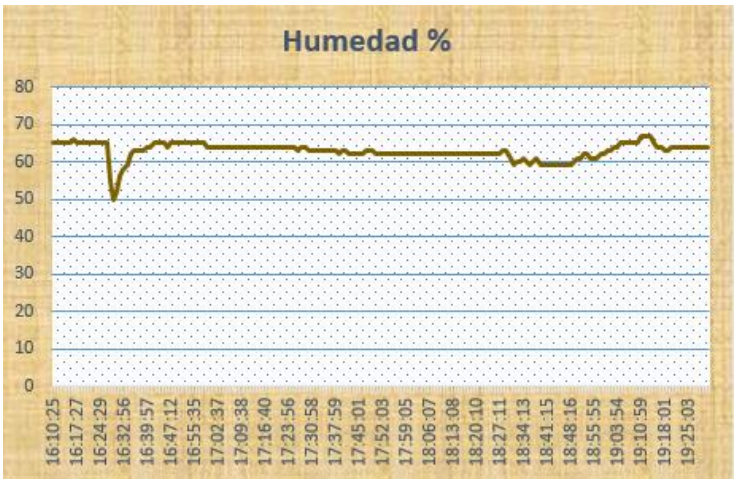


Figura 59. Resultados de Humedad.
Elaborado por: Los autores.

Tabla 34. Tabla de datos principales en intervalos de Humedad.

Hora	Dato	Nivel de pico
16:17h	65%	Promedio
16:30h	50%	Bajo
17:52h	61%	Promedio
18:34h	58%	Bajo
19:10h	68%	Alto

Nota: Tabla de datos para valorar intervalos de humedad.
Elaborado por: Los autores.

En la figura 59, se observa los datos obtenidos del sensor de humedad en un lapso de 3 horas con intervalo de 1 minuto, con una eficiencia de llegada de 98%. En la tabla 34, se describe los datos obtenidos en la gráfica de humedad, donde, se observa un promedio de 65%, también se evidencia que a las 19:10h se produjo un evento, el cual nos da un pico alto de 68%, donde se refleja la subida de humedad en el sensor, asimismo, a las 16:30h se evidencia otro evento donde se sometió a una prueba inversa al sensor dándonos un 50%.

Datos Potenciómetro

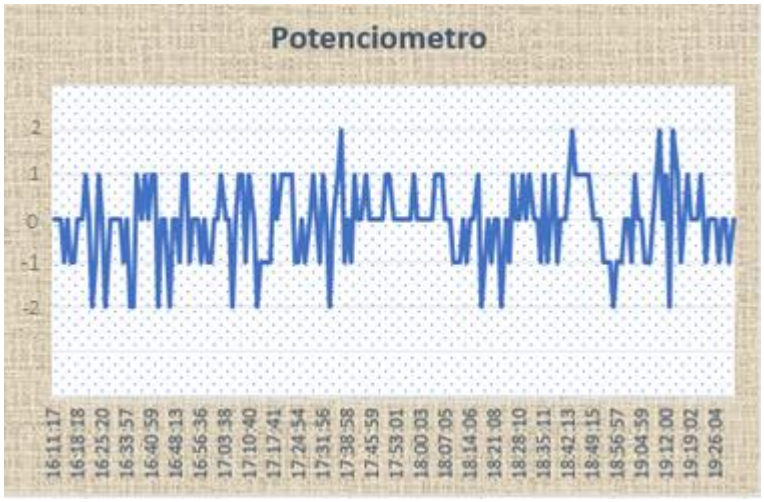


Figura 60. Resultados de Potenciómetro.
Elaborado por: Los autores.

Tabla 35. Tabla de datos principales en intervalos del potenciómetro.

Hora	Dato	Nivel de pico
16:11h	0Ω	Promedio
16:40h	1 Ω	Alto
17:58h	2Ω	Alto
18:42h	-1Ω	Bajo
19:12h	-2Ω	Bajo

Nota: Tabla de datos para valorar intervalos del potenciómetro.
Elaborado por: Los autores.

En la figura 60, se muestra el comportamiento de los datos obtenidos del potenciómetro analógico en un lapso de 3 horas con intervalo de 1 minuto, con una eficiencia de llegada de 98%. En la tabla 35, se describe los datos obtenidos en la gráfica del potenciómetro, donde se observa un promedio de 0Ω en la escala registrada, también se evidencia que a las 17:58h se produjo un evento, el cual nos da un pico alto de 2Ω, donde se refleja el giro del tornillo del sensor, también, a las 19:12h se evidencia otro evento donde se sometió a una prueba inversa en el tornillo del sensor dándonos un -2Ω.

Datos Proximidad

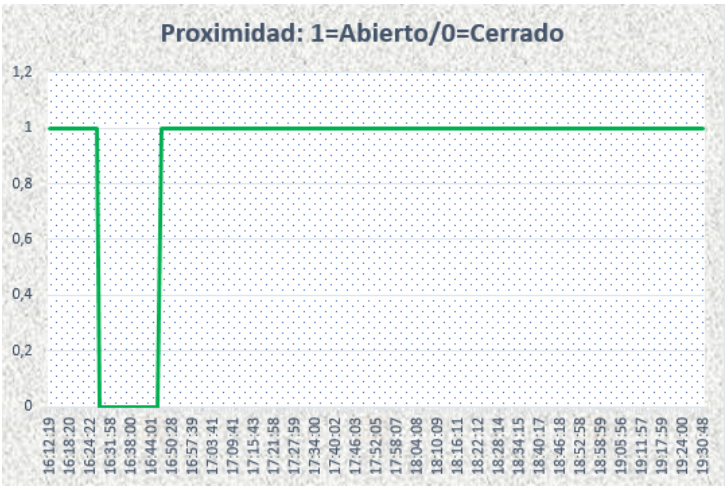


Figura 61. Resultados del sensor de proximidad.
Elaborado por: Los autores.

Tabla 36. Tabla de datos principales en intervalos del sensor de proximidad.

Hora	Dato	Proximidad
16:18h	1	Cerca
16:38h	0	Lejos
17:03h	1	Cerca

Nota: Tabla de datos para valorar intervalos del sensor de proximidad.
Elaborado por: Los autores.

En la figura 61, se observa los datos obtenidos del sensor de proximidad digital en un lapso de 3 horas con intervalo de 1 minuto, con una eficiencia de llegada de 98%. En la tabla 36, se describe los datos obtenidos en la gráfica del sensor de proximidad, donde, se observa dos únicos valores que son 1 y 0. Esto se debe a que el sensor envía datos booleanos y se los puede identificar con los eventos evidenciados, donde el valor "1" se debe a que un objeto está cerca del sensor y el valor "0" cuando no detecta presencia de un objeto.

Tabla 37. Tabla de resultados de pruebas.

Sensor	Datos examinados	Intervalo de tiempo	Eficiencia de datos
Temperatura	5	3 Hrs.	98%
Humedad	5	3 Hrs.	98%
Potenciometro	5	3 Hrs.	98%
Proximidad	3	3 Hrs.	98%

Resultados			
	18	12	98%

Nota: Tabla de datos para valorar la conectividad de red.
Elaborado por: Los autores.

En la tabla 37, se detalla los datos principales de las pruebas realizadas a los sensores en su respetivo funcionamiento.

Tabla 38. Tabla de resultados de sensores.

Valoración	Base	Resultado
Resultado	100	**98**
Objetivo		**100**
% Funcionamiento		**98%**

Nota: Tabla de resultados para valorar lo objetivos del 100%.
Elaborado por: Los autores.

En la tabla 38, se observa los resultados obtenidos de los sensores, dándonos el 98% de funcionamiento general en los datos examinados.

4.8 Análisis y resultados de la conexión entre la red CAN y los módulos IoT.

En el apartado siguiente se analizará los resultados mediante el uso de un instrumento de visualización electrónico (Osciloscopio Gw Instek), el cual permite interpretar señales eléctricas que pueden variar en un tiempo. En la figura 62, muestra el Osciloscopio.

Equipo de análisis

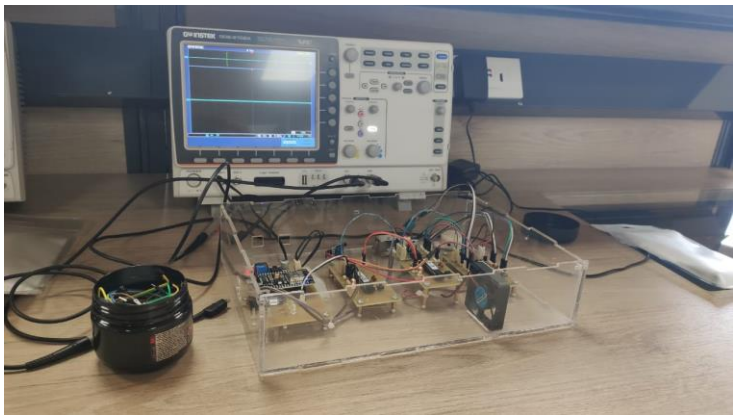


Figura 62. Osciloscopio.
Elaborado por: Los autores.

4.8.1 Comunicación Node Display a dispositivo IoT MCU

Análisis Node Display

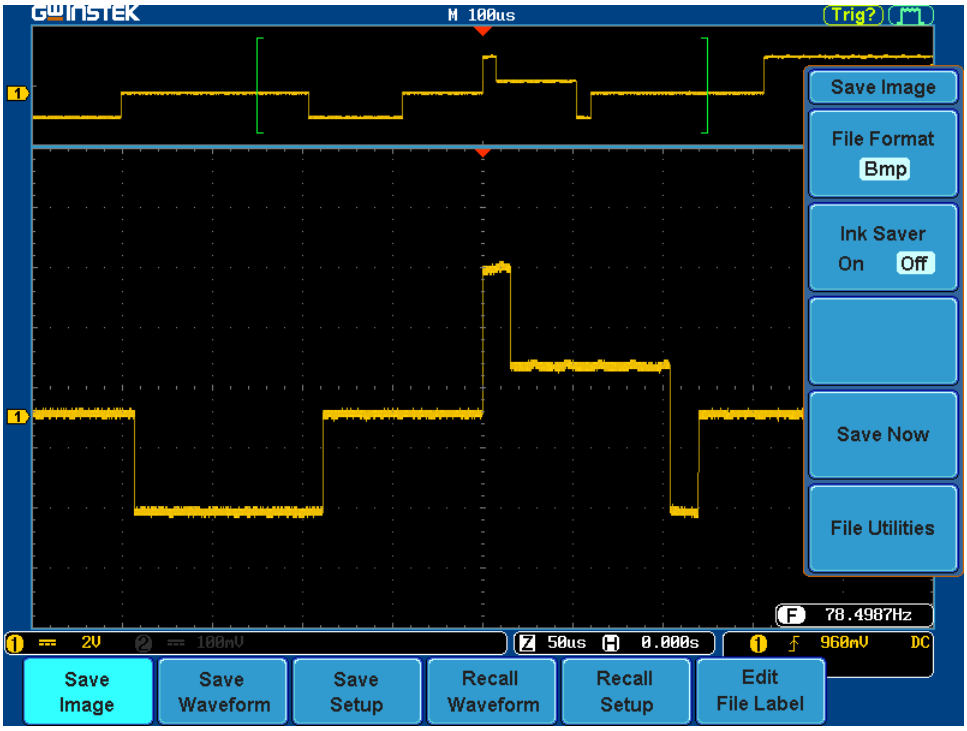


Figura 63. Señal CAN.
Elaborado por: Los autores.

En la figura 63, se observa la gráfica de la señal CAN desde el nodo Display hacia el Node MCU, se tiene en cuenta que la comunicación trabaja a un voltaje igual a 2 y con un tiempo de 50 microsegundos a una frecuencia de 78.4984 Hz. La comunicación es inmediata teniendo como muestra variaciones de las señales eléctricas dependiendo de la información recibida de los respectivos nodes colectores 1 y 2. Las señales varían según el dato que reciba el Node Display, la cresta hace referencia al Node 2 que son los sensores y los valles son los actuadores.

Como se puede observar en la gráfica en la parte superior se tiene una señal ampliada, que se encuentra en Standby es decir no transmite ni recibe la señal, es una onda cuadrada que trabaja a una frecuencia aproximada de 80 kHz. En la parte de abajo de la gráfica se puede observar que el protocolo CAN empieza a trabajar enviando datos a un voltaje de 3,5 Voltios denominado CAN High (dominante), después la señal regresa a su modo recesivo donde el voltaje es de 2,5 Voltios

y finalmente recibe datos a un voltaje de 1.5 Voltios denominado CAN Low, dándonos un 100% en el trabajo con el protocolo CAN. En la figura 64, muestra la señal lógica CAN.

Estado lógico de CAN

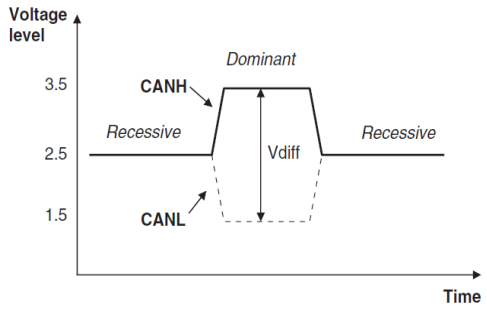


Figura 64. Señal lógica CAN.
Fuente: (Ibrahim, 2008)

4.9 Análisis y Resultados de la red desde el dispositivo de envío hacia la plataforma

4.9.1 PRTG Network Monitor

Es un software que se utiliza para monitorizar sistemas, dispositivos, tráfico y aplicaciones dentro de una infraestructura TI, existe dos tipos de licencias la personal y comercial.

4.9.2 Protocolo ICMP

Es un protocolo que pertenece al modelo TCP/IP que permite comunicar información sobre problemas de conectividad de red al origen de la trasmisión. Utiliza una estructura de paquetes de datos con un encabezado de 8 bytes y una unidad de datos de diferente tamaño.

ICMP define mensajes de consulta que proporcionan información de la red relacionada con el enrutamiento de paquetes, desempeño de la red y direcciones de la subred (Fuentes, 2001)

4.9.2.1 Ping

Es una utilidad que utiliza mensajes ICMP para enviar información sobre la conectividad de la red y la velocidad de trasmisión de datos entre un host y una computadora de destino.

Análisis de red

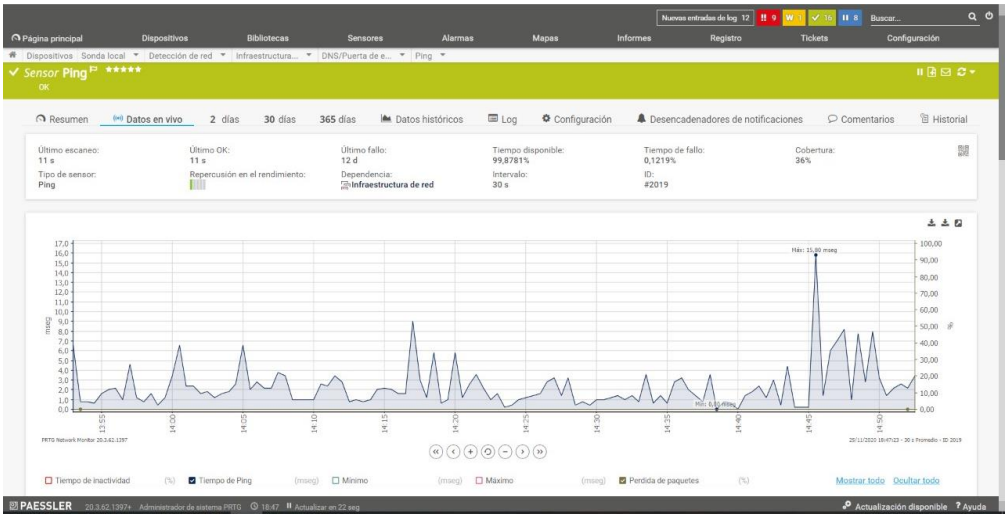


Figura 65. Análisis ICMP.
Elaborado por: Los autores.

Tabla 39. Tabla de resultados de pruebas.

Días	Conectividad disponible	Intervalo de tiempo
1	99.88%	3 Hrs.
2	99.87%	2 Hrs.
3	99.87%	3 Hrs.
4	99.86%	1:30 Hrs.
Resultados		
4 días	99.87%	9:00 Hrs.

Nota: Tabla de datos para valorar la conectividad de red.
Elaborado por: Los autores.

En la tabla 39, se detalla los datos principales de las pruebas realizadas a la conectividad entre la plataforma Thinger.Io y el dispositivo IoT.

Tabla 40. Tabla de resultados de conectividad de red.

Valoración	Base	Resultado
Resultado	**100**	**99.87**
Objetivo		**100**
% Funcionamiento		**99.87%**

Nota: Tabla de resultados para valorar lo objetivos del 100%.
Elaborado por: Los autores.

En la figura 65, se observa la conectividad entre la plataforma Thinger.Io y el dispositivo IoT, donde nos indica la red disponible para la conectividad, asimismo se observa en determinados

intervalos, como en el de las 14:45 a 14:46, un ping máximo de 15,8mseg, debido a las diferentes cargas de datos que envían los sensores. En la tabla 40, se observa los resultados obtenidos, dándonos el 99.87% de funcionamiento en la conectividad.

4.10 Resultados finales de pruebas de efectividad del prototipo

Para la siguiente evaluación, se procede a verificar los resultados de las pruebas realizadas a todo el sistema del prototipo y mediante los indicadores establecidos se calificará la efectividad del funcionamiento. En la tabla 41, muestra la escala de valores con la que se calificará a todo el prototipo.

Tabla 41. Escala de valores.

Valores del cumplimento del objetivo		Observaciones
98-100	Se cumple excelentemente	Conservar
75-97	Se cumple normalmente	Posibilidad de mejorar
60-74	Se cumple probablemente	Necesidad de revisar
40-59	Se cumple pobremente	Descartar o profundizar
Menor a 40	No se cumple	Descartar definitivamente

Nota: Valores del cumplimiento de objetivos establecidos.
Elaborado por: Los autores.

4.10.1 Indicadores establecidos

A continuación, se establecen los indicadores:

- Pruebas de funcionamiento.
- Pruebas de la plataforma.
- Pruebas de datos en sensores y actuadores.
- Pruebas de conectividad CAN.
- Pruebas de conectividad de red.

4.10.2 Valoración Final

Se procede a calificar al prototipo con los indicadores establecidos para obtener la efectivad.

Tabla 42. Tabla de indicadores.

Tabla de indicadores		
Indicadores	%	Observaciones
Pruebas de funcionamiento	100	El prototipo funciona según lo planteado, desde el encendido hasta la aplicación.
Prueba de la plataforma	97.6	Limitación en la plataforma por ser gratuita, con una calificación 97.6%.
Pruebas de datos de en sensores y actuadores	98	Las pruebas realizadas a los sensores y actuadores resultaron efectivas como muestra los datos obtenidos.
Pruebas de conectividad CAN	100	Funcionamiento y lectura correcta del sistema CAN realizado por el osciloscopio.
Pruebas de conectividad de red	99.87	La comunicación funciona correctamente, teniendo una insignificante pérdida de 0,13%
Suma	495.47	
Total 100%	99.09	

Nota: Muestra la asignación de valores de acuerdo con los indicadores.
Elaborado por: Los autores.

En la tabla 42, se indica la calificación de los indicadores de acuerdo con el desarrollo y pruebas realizadas con anterioridad al prototipo con su respectiva observación para cada valoración final.

Tabla 43. Tabla de resultados finales .

Valoración	Base	Resultado
Resultado	**100**	**99.09**
Objetivo		**100**
% Efectividad		**99, 09%**

Nota: Tabla de resultados para valorar lo objetivos del 100%.
Elaborado por: Los autores.

En la tabla 43, se detalla el valor final obtenido en las pruebas de funcionamiento de todo el sistema, dándonos como resultado el 99,09% de efectividad en el funcionamiento del prototipo, además, se concluye que se cumplió todos los objetivos planteados, el cual como valoración final se puede conservar lo realizado y con posibilidad de mejorar en los aspectos negativos.

CONCLUSIONES

El prototipo se sometió a la valoración de pruebas de modo que los resultados obtenidos reflejaron un 99,09% de efectividad y la pérdida hace referencia a las limitaciones de la plataforma y su aplicación móvil.

El voltaje de comunicación inicial entre el Node Display y el dispositivo IoT es de 2V, donde se evidencia que la señal se encuentra en Standby, es decir, no transmite ni recibe señal en su onda cuadrada que trabaja a una frecuencia de 80khz a un tiempo de 50 microsegundos. Se observa que el protocolo CAN empieza a trabajar enviando datos a un voltaje de 3.5v, regresando después a un estado de 2.5v y finalmente recibiendo datos a un voltaje de 1.5v, esto significa que el envio de datos trabaja a un mayor voltaje a diferencia que cuando recibe datos, esta variación de voltaje se produce por causa de que los datos enviados por los sensores son en tiempo real y los datos que reciben los actuadores son por petición del usuario.

Mediante una matriz comparativa de adaptabilidad entre diferentes plataformas abiertas en el mercado se eligió a Thinger.Io, debido a su plataforma amigable con el usuario y su gran escalabilidad con el número de dispositivos que se están controlando y monitoreando. Se realizó pruebas entre la plataforma y el dispositivo IoT dando un resultado de 97.6%, la pérdida se debe a los servicios y aplicaciones que brinda la plataforma de manera gratuita.

Se examinó la conectividad entre la plataforma y el dispositivo IoT mediante la herramienta PRTG en un lapso de 9 hora, donde los resultados reflejan que durante el lapso transcurrido monitorizando el tiempo de disponibilidad de la red fue de un 99,87%, y el tiempo máximo fue de 15.8mseg, esto es debido a que los sensores y actuadores se activaron al mismo tiempo.

RECOMENDACIONES

- Se recomienda que el diseño del prototipo tenga una buena recepción de señal Wifi, ya que se vería afectado al momento del envió de datos a la plataforma abierta.
- Es recomendable usar los dispositivos y sensores utilizados en el diseño de este prototipo, por su facilidad de configuración y conexión, además que se adaptan de manera fácil a cualquier entorno.
- Al utilizar una Plataforma abierta de manera gratuita se puede ver limitado la conexión de los sensores, se recomienda adquirir una suscripción de paga o diseñar un propio entorno web.

LISTA DE REFERENCIAS

Arduino. (2021). Arduino.cc - Home. Retrieved January 13, 2021, from https://www.arduino.cc/

Associates, C. (2020). Proton BASIC Compiler. Retrieved January 13, 2021, from http://www.protonbasic.co.uk/content.php

AT&T. (2016). What you need to know about IoT platforms. (April), 121–130. https://doi.org/10.109/01.orn.0000407780.43380.20

Augmented, S. (2021). ST Microelectronics. Retrieved January 13, 2021, from https://www.st.com/content/st_com/en.html

Blynk. (2020). Blynk IoT platform: for businesses and developers. Retrieved January 13, 2021, from https://blynk.io/

Boudreau, K. (2010). Open platform strategies and innovation: Granting access vs. devolving control. Management Science, 56(10), 1849–1872. https://doi.org/10.1287/mnsc.1100.1215

Chen, S., Xu, H., Liu, D., Hu, B., & Wang, H. (2014). A vision of IoT: Applications, challenges, and opportunities with China Perspective. IEEE Internet of Things Journal, 1(4), 349–359. https://doi.org/10.1109/JIOT.2014.2337336

Chiang, M., & Zhang, T. (2016). Fog and IoT: An Overview of Research Opportunities. IEEE Internet of Things Journal, 3(6), 854–864. https://doi.org/10.1109/JIOT.2016.2584538

Chikhale, S. N. (2018). Automobile Design and Implementation of CAN bus Protocol- A Review. IJRDO - Journal of Electrical And Electronics Engineering (ISSN: 2456-6055), 4(1), 01–05. Retrieved from https://www.ijrdo.org/index.php/eee/article/view/1528

Corrigan, S. (2002). Introduction to the Controller Area Network (CAN) Application Report Introduction to the Controller Area Network (CAN). (May), 1–17. Retrieved from www.ti.com

Director, A. R. D. (2017). Integración de dispositivos IoT en una red comunitaria. (1203), 1–9.

Eisenmann, T. R., Parker, G., & Van Alstyne, M. W. (2008). Opening Platforms: How, When and Why? Working Paper. 09-030. Harvard Business School Working Paper, (August). https://doi.org/10.2139/ssrn.1264012

Electronics, G. (2021). Microcontroladores, Sensores, Componentes Electrónicos. Retrieved January 13, 2021, from http://www.geekbotelectronics.com/

Electronics, U. (2016). Tienda de componentes electronicos en México | CDMX Electrónica. Retrieved January 13, 2021, from https://uelectronics.com/

ESP8266 Datasheet. (2015). ESP8266 Datasheet. Espressif Systems Datasheet, 1–31. Retrieved from https://www.adafruit.com/images/product-files/2471/0A-ESP8266__Datasheet__EN_v4.3.pdf

Espressif. (2019). ESP32 Series Datasheet. Espressif Systems, 1–61. Retrieved from www.espressif.com

Evans, D. (2011). Internet de las cosas Internet de las cosas Cómo la próxima evolución de Internet lo cambia todo. 11.

Futurlec. (2021). The Electronic Components and Semiconductor Superstore. Retrieved January 13, 2021, from https://www.futurlec.com/index.shtml

Gómez, G., Asesora, D., Doctora, :, & Altamirano, A. C. C. (2017). Instituto Tecnológico Y De Estudios Superiores.

Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7),

1645–1660. https://doi.org/10.1016/j.future.2013.01.010
Iera, A., Floerkemeier, C., Mitsugi, J., & Morabito, G. (2010). The Internet of things. IEEE Wireless Communications, 17(6), 8–9. https://doi.org/10.1109/MWC.2010.5675772
Lee, I., & Lee, K. (2015). The Internet of Things (IoT): Applications, investments, and challenges for enterprises. Business Horizons, 58(4), 431–440. https://doi.org/10.1016/j.bushor.2015.03.008
Luis, M., & Andrango, D. (2007). Estudio del protocolo CAN y su aplicacion en redes de control. Escuela Politecnica Nacional.
McMahon, R. A. (2003). Introducing basic network concepts. Introduction to Networking (Mike Meyers' Computer Skills), 1–27. Retrieved from https://www3.nd.edu/~cpoellab/teaching/cse40814_fall14/networks.pdf
Microchip Technology Inc. (2009). Pic18F2480/2580/4480/4580 Data Sheet. 1–390.
MikroE. (2021). mikroC PRO for PIC | Mikroe. Retrieved January 13, 2021, from https://www.mikroe.com/mikroc-pic
Ng, W. L., Ng, C. K., Ali, B. M., Noordin, N. K., & Rokhani, F. Z. (2010). Review of Researches in Controller Area Networks Evolution and Applications. Proceedings of the Asia-Pacific Advanced Network, 30(0), 14. https://doi.org/10.7125/apan.30.3
Patel, K. K., Patel, S. M., & Scholar, P. G. (2016). Internet of Things-IOT: Definition, Characteristics, Architecture, Enabling Technologies, Application & Future Challenges. International Journal of Engineering Science and Computing, 6(5), 1–10. https://doi.org/10.4010/2016.1482
Pina, A. (2017). Diseño de un sistema domótico con comunicaion wifi. Universidad de Zaragoza, 1–98.
Prometec. (2021). Tienda Arduino y Artículos de Electrónica. Retrieved January 13, 2021, from https://store.prometec.net/
Raspberrypi. (2021). Teach, Learn, and Make with Raspberry Pi. Retrieved January 13, 2021, from https://www.raspberrypi.org/
Rosado Muñoz, A. (2010). Sistemas Industriales Distribuidos Tema 2. Redes de comunicación: Topología y enlaces. 21.
Sánchez Vela, Luis Gerardo, Molano Clemente, Martín Jonathan, Fabela Gallegos, Manuel de Jesús, Martínez Madrid, Miguel, Hernández Jiménez, José Ricardo, Vázquez Vega, David, Flores Centeno, O. (2016). Revisión documental del protocolo CAN como herramienta de comunicación y aplicación en vehículos. (474). Retrieved from https://imt.mx/archivos/Publicaciones/PublicacionTecnica/pt474.pdf
Technology, M. (2021). MPLAB® X IDE. Retrieved January 13, 2021, from https://www.microchip.com/en-us/development-tools-tools-and-software/mplab-x-ide
Thinger.io. (2020). Open Source IoT Platform. Retrieved January 13, 2021, from https://thinger.io/
ThingSpeak. (2021). IoT Analytics - ThingSpeak Internet of Things. Retrieved January 13, 2021, from https://thingspeak.com/
Toubes, S. D. (2017). Realización de un analizador de redes can con interfaz serie. Universidad de Vigo, 1–20.
Tropmann-Frick, M. (2018). Internet of things: Trends, challenges and opportunities. Communications in Computer and Information Science, 909, 254–261. https://doi.org/10.1007/978-3-030-00063-9_24
Ubidots. (2021). IoT platform | Internet of Things. Retrieved January 13, 2021, from https://ubidots.com/

Yupapin, P. P., Mitatha, S., Ali, J., & Teeka, C. (2012). Nanocommunication networks. Nanocommunication Networks, 1–193.