UTMACH

UNIDAD ACADÉMICA DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

DESARROLLO E IMPLEMENTACIÓN DE UN GATEWAY PARA UNA RED DE SENSORES INALÁMBRICOS BLE INTEGRADO AL SISTEMA IOTMACH

PRADO VILLACRES JORGE DAVID
INGENIERO DE SISTEMAS

MACHALA
2016

UTMACH

UNIDAD ACADÉMICA DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

DESARROLLO E IMPLEMENTACIÓN DE UN GATEWAY PARA UNA RED DE SENSORES INALÁMBRICOS BLE INTEGRADO AL SISTEMA IOTMACH

PRADO VILLACRES JORGE DAVID

MACHALA
2016

UTMACH

UNIDAD ACADÉMICA DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

TRABAJO DE TITULACIÓN
PROPUESTAS TECNOLÓGICAS

DESARROLLO E IMPLEMENTACIÓN DE UN GATEWAY PARA UNA RED DE SENSORES INALÁMBRICOS BLE INTEGRADO AL SISTEMA IOTMACH

PRADO VILLACRES JORGE DAVID
INGENIERO DE SISTEMAS

HERNANDEZ ROJAS DIXYS LEONARDO

Machala, 18 de octubre de 2016

MACHALA
2016

Nota de aceptación:

Quienes suscriben HERNANDEZ ROJAS DIXYS LEONARDO, LOJA MORA NANCY MAGALY, MOROCHO ROMAN RODRIGO FERNANDO y HONORES TAPIA JOOFRE ANTONIO, en nuestra condición de evaluadores del trabajo de titulación denominado DESARROLLO E IMPLEMENTACIÓN DE UN GATEWAY PARA UNA RED DE SENSORES INALÁMBRICOS BLE INTEGRADO AL SISTEMA IOTMACH, hacemos constar que luego de haber revisado el manuscrito del precitado trabajo, consideramos que reúne las condiciones académicas para continuar con la fase de evaluación correspondiente.

HERNANDEZ ROJAS DIXYS LEONARDO
0923026298
TUTOR

LOJA MORA NANCY MAGALY
0703410027
ESPECIALISTA 1

MOROCHO ROMAN RODRIGO FERNANDO
0703820464
ESPECIALISTA 2

HONORES TAPIA JOOFRE ANTONIO
0704811751
ESPECIALISTA 3

Machala, 18 de octubre de 2016

Urkund Analysis Result

Analysed Document: Prado Villacres Jorge David.docx (D21636977)
Submitted: 2016-09-07 03:45:00
Submitted By: jorgito.sly@gmail.com
Significance: 1 %

Sources included in the report:

Informe Trabajo Titlacion Priscila Valencia.docx (D16312101)

Instances where selected sources appear:

3

CLÁUSULA DE CESIÓN DE DERECHO DE PUBLICACIÓN EN EL REPOSITORIO DIGITAL INSTITUCIONAL

El que suscribe, PRADO VILLACRES JORGE DAVID, en calidad de autor del siguiente trabajo escrito titulado DESARROLLO E IMPLEMENTACIÓN DE UN GATEWAY PARA UNA RED DE SENSORES INALÁMBRICOS BLE INTEGRADO AL SISTEMA IOTMACH, otorga a la Universidad Técnica de Machala, de forma gratuita y no exclusiva, los derechos de reproducción, distribución y comunicación pública de la obra, que constituye un trabajo de autoría propia, sobre la cual tiene potestad para otorgar los derechos contenidos en esta licencia.

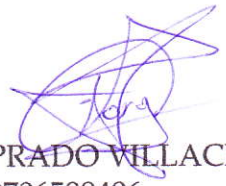
El autor declara que el contenido que se publicará es de carácter académico y se enmarca en las dispociones definidas por la Universidad Técnica de Machala.

Se autoriza a transformar la obra, únicamente cuando sea necesario, y a realizar las adaptaciones pertinentes para permitir su preservación, distribución y publicación en el Repositorio Digital Institucional de la Universidad Técnica de Machala.

El autor como garante de la autoría de la obra y en relación a la misma, declara que la universidad se encuentra libre de todo tipo de responsabilidad sobre el contenido de la obra y que él asume la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.

Aceptando esta licencia, se cede a la Universidad Técnica de Machala el derecho exclusivo de archivar, reproducir, convertir, comunicar y/o distribuir la obra mundialmente en formato electrónico y digital a través de su Repositorio Digital Institucional, siempre y cuando no se lo haga para obtener beneficio económico.

Machala, 18 de octubre de 2016



PRADO VILLACRES JORGE DAVID
0706589496

Dedicatoria

El presente trabajo se lo dedico principalmente a Dios por bendecir cada uno de los pasos que doy en mi vida; a mi padre Jorge Prado Sarmiento y mi madre María Villacres Sedamanos por el apoyo que me dan en cada una de mis metas; a mis hermanos por sus consejos; a mi esposa Marjorie Noblecilla Condoy e hija Massiel Prado Noblecilla que día a día me motivaban, a cada uno de mis maestros por las semillas de conociendo que depositaron en mí.

Agradecimiento

Este trabajo quiero agradecer de manera especial a mi tutor Ing. Dixys Hernández y docente Ing. Bertha Mazón ya que supieron transmitirme sus conocimientos y brindarme su paciencia.

A la Universidad Técnica de Machala, Unidad Académica de Ingeniería Civil, Carrera de Ingeniería de Sistemas por abrirme las puertas del saber y de esa forma brindarme la oportunidad de seguir creciendo en mi vida estudiantil.

Y por último pero de una manera muy especial, a mi madre, que sin importar nada, siempre me apoyo económica y moralmente, por ser mi ejemplo a seguir en la vida.

RESUMEN

El internet de las cosas, es un campo de estudio amplio, donde abarca el uso de dispositivos inteligentes (motes) conectados a una Cloud Computing a través del internet, para intercambiar información y de esta manera tener control sobre cada dispositivo desde cualquier lugar. El conjunto de dispositivos inalámbricos inteligentes forman una WSN o WSAN, estos utilizan la configuración de red IPv6 y la tecnología inalámbrica Bluetooth, mientras que actualmente las direcciones públicas utilizadas en el internet trabajan sobre el protocolo IPv4, por lo que se requiere un Gateway que enlace una WSN o WSAN con una Cloud Computing, utilizando protocolos, estándares y tecnologías inalámbricas del internet de las cosas. Para el análisis/diseño del Gateway IOT, se aplicó la metodología de ingeniería de software RAD, de esta manera se pudieron determinar: los requisitos a través del análisis de los requerimientos y el bosquejo del diseño del Gateway IOT. En el desarrollo del Gateway IOT se utilizó diferentes herramientas informáticas en Hardware y Software. En Hardware se utiliza una Raspberry Pi 3 model B, para utilizar la tarjeta Bluetooth 4.1 que tiene incorporada y así activar Bluetooth Low Energy. En Software se utilizó herramientas Open Source como; el sistema operativo Raspbian, la librería BlueZ con 6LoWPAN, un Broker MQTT, y para la programación del sitio WEB un stack personalizado de MEAN (MongoDb, Express JS, Angular JS, Node JS). El Gateway IOT se comunica con la WSN o WSAN usando Bluetooth Low Energy con: MQTT o Eddystone a través de BlueZ. Con los motes que utilizan Bluetooth Low Energy MQTT, se realiza una conexión lógica mediante 6LoWPAN, de esta manera se puede activar IPv6 en la tarjeta Bluetooth, mientras que con Eddystone se lee los Beacons que publica el mote al realizar un Advertising. El Gateway IOT cuenta con un sitio WEB de administración que permite; la gestión de las interfaces o tarjetas físicas (Wi-Fi, Ethernet, Bluetooth) de la Raspberry Pi 3 model B y el monitoreo/control de los sensores o actuadores de los motes conectados o aceptados, así el usuario puede controlar y gestionar lo que ocurre en el Gateway IOT. Para enlazar la WSN o WSAN con la Cloud Computing, se utiliza un puente MQTT; entre el Broker MQTT del Gateway con el Broker MQTT de la Cloud Computing, de esta manera se puede resolver la comunicación entre las redes que utilizan Protocolos de Internet diferentes, se puede configurar varios puentes MQTT desde la página WEB del Gateway IOT. Se realizó una evaluación de varias pruebas, para determinar: la integración de todos los módulos desarrollado funcionando al mismo tiempo, el rendimiento del Broker MQTT con algunos usuarios conectados y publicando en el Gateway IOT, la usabilidad que tiene el sitio WEB cuando es gestionado por usuarios, el alcance de la tarjeta

Bluetooth utilizando Bluetooth Low Energy MQTT o Eddystone y la integración del Gateway IOT con la Cloud Computing del Sistema IOTMACH. Se le puede adicionar al Gateway IOT otras funcionalidades como: protocolos de comunicación (COAP, XAMP) para enlazar los datos de la WSN o WSAN con la Cloud Computing y aumentar las tecnologías inalámbricas, por ejemplo: Lora, Zigbee. Por lo tanto mediante una investigación bibliográfica que permitió definir los estándares, protocolos y tecnología inalámbrica del internet de las cosas a utilizarse, se logró implementar un Gateway IOT que puede ser gestionado por usuarios a través de un sitio WEB.

Palabras Claves: Gateway IOT, Internet de las Cosas, MQTT, Bluetooth Low Energy, Eddystone.

ABSTRACT

The Internet of Things, is a broad field of study, which includes the use of smart devices connected to a Cloud Computing through the Internet, to exchange information and thus have control over each device from anywhere. The set of smart wireless devices form a WSN or WSAN, these use the IPv6 network configuration and Bluetooth wireless technology, while currently the public addresses used on the internet work on the IPv4 protocol, so a Gateway is required to link A WSN or WSAN with a Cloud Computing, using protocols, standards and wireless internet technologies of things. For the analysis / design of the IOT Gateway, the software engineering methodology RAD was applied, in this way it was possible to determine: the requirements through the analysis of the requirements and the outline of the IOT Gateway design. In the development of the Gateway IOT different hardware tools were used in Hardware and Software. In Hardware a Raspberry Pi 3 model B is used, to use the Bluetooth card 4.1 that has built-in and to activate Bluetooth Low Energy. In Software we used Open Source tools like; The Raspbian operating system, the BlueZ library with 6LoWPAN, a MQTT Broker, and for the WEB site programming a customized MEAN stack (MongoDb, Express JS, Angular JS, Node JS). The IOT Gateway communicates with the WSN or WSAN using Bluetooth Low Energy with: MQTT or Eddystone via BlueZ. With the motes that use Bluetooth Low Energy MQTT, a logical connection is made using 6LoWPAN, this way you can activate IPv6 on the Bluetooth card, while with Eddystone you can read the Beacons that publish the motto when performing a Advertising. The IOT Gateway has a management WEB site that allows; The management of the interfaces or physical cards (Wi-Fi, Ethernet, Bluetooth) of the Raspberry Pi 3 model B and the monitoring / control of the sensors or actuators of connected or accepted motes, so that the user can control and manage what Occurs on the IOT Gateway. To link the WSN or WSAN with Cloud Computing, an MQTT bridge is used; Between the Gateway MQTT Broker and the Cloud Computing MQTT Broker, this way you can resolve the communication between networks using different Internet Protocols, you can configure several MQTT bridges from the IOT Gateway WEB page. An evaluation of several tests was carried out to determine: the integration of all modules developed at the same time, the performance of the MQTT Broker with some users connected and publishing in the IOT Gateway, the usability of the WEB site when it is managed by Users, the range of the Bluetooth card using Bluetooth Low Energy MQTT or Eddystone and the integration of the IOT Gateway with the Cloud Computing of the IOTMACH System. Other features such as: communication protocols (COAP, XAMP) can be added to the IOT Gateway to link the WSN or WSAN data with Cloud

Computing and increase wireless technologies, for example: Lora, Zigbee. Therefore, through a bibliographic investigation that allowed defining the standards, protocols and wireless technology of the internet of the things to be used, it was possible to implement an IOT Gateway that can be managed by users through a WEB site.

Keywords: Gateway IOT, Internet of Things, MQTT, Bluetooth Low Energy, Eddystone.

CONTENIDO

pág.

LISTA DE TABLAS

pág.

LISTA DE FIGURAS

pág.

LISTA DE ANEXOS

pág.

INTRODUCCIÓN

Hoy en día casi todo lo que buscamos está en internet, desde una receta de comida hasta como aprender a programar en algún lenguaje específico, la cual se ha convertido en una herramienta esencial en todos los aspectos, y es que el internet une al mundo. El futuro del internet está envuelto en la comunicación de objetos digitales, por ejemplo: (teléfonos inteligentes, computadoras, autos), los cuales pueden ser conectados por medio de diferentes tecnologías de la información [1], esto es conocido como IOT (*Internet Of Things*).

La IOT, es un nuevo paradigma de conectar objetos o cosas a través de internet [2], el concepto de cosas conectadas al internet es conocido desde principios de 1990, que hasta la actualidad no se ha implementado completamente en la sociedad [3]. El término IOT se utiliza cuando existe comunicación inteligente entre objetos, usando algunos materiales dieléctricos, como menciona [4]. La IOT tiene como objetivo conectar dispositivos por ejemplo: sensores, para intercambiar datos con el internet (Cloud Computing) aplicando diferentes tecnologías [5].

Cuando se menciona *Cloud Computing* hace referencia a un modelo para el acceso a petición de un conjunto compartido de recursos informáticos configurables; por ejemplo: redes, servidores, almacenamiento, aplicaciones, servicios y software, que pueden ser suministrados fácilmente cómo y cuando sea necesario [6].

Una WSAN (*Wireless Sensor and Actuator Network*) en la IOT, es un conjunto de tarjetas electrónicas (nodos sensores, dispositivos o motes) conectadas entre sí por medio de una red inalámbrica, que obtienen información de su alrededor a través de sensores (temperatura, humedad, etc.) o ejecutan órdenes a los actuadores (focos, bombas de agua, etc.) [7]. También se la puede llamar WSN *(Wireless Sensor Network*), cuando los motes no tienen conectados actuadores. Las tecnologías inalámbricas a utilizarse en una WSN o WSAN pueden ser varias, entre ellas están: *Wi-Fi, Bluetooth, ZigBee*. La referencia [8] profundiza sobre las tecnologías inalámbricas mencionadas.

Para los dispositivos B*luetooth* en el año 2015, Google desarrolló *Eddystone, que* es un estándar de código abierto para utilizar B*eacons*, en la tecnología inalámbrica, es el concepto de difusión de pequeños fragmentos de información [9].

Internet hoy en día usa IPv4, aunque actualmente existen dos protocolos para las direcciones IP que son: IPv4 (*Internet Protocol versión* 4) e IPv6 (*Internet Protocol versión* 6). IPv6 está diseñado como el sucesor de IPv4, una de las razones es porque permite la ampliación del espacio de direcciones IP de 32 a 128 bits [10]. El cambio de

IPv4 a IPv6 en el internet será un proceso muy largo, ya que estos protocolos son incompatibles [11].

En una WSN o WSAN puede existir un nodo llamado *Gateway* IOT, el cual cuenta con una comunicación bidireccional entre los nodos sensores y la *Cloud Computing*, por ejemplo: envía la información monitorizada o recibe mensajes de control de los nodos sensores desde la *Cloud Computing* o viceversa [12].

La información puede ser intercambiada en varios protocolos de comunicación, por ejemplo: HTTP (*Hypertext Transfer Protocol*), es el protocolo de la WWW (*World Wide Web)*, se accede a los recursos disponibles a través de una URLs. COAP (*Constrained Application Protocol*), usa paquetes UDP (*User Datagram Protocol*) para transmitir los mensajes [13]. MQTT (*Message Queue Telemetry Transport*), la describen como ideal para un crecimiento inteligente con recursos limitados, ya que usa poco ancho de banda, y se basa en la pila TCP/IP [14].

Se plantea el desarrollo e implementación de un Gateway IOT inalámbrico, que enlace bidireccionalmente la comunicación de una WSN o WSAN con IPv6 y una C*loud Computing* con IPv4, para la transición de datos, basándose en: el concepto de la IOT y uso de herramientas de bajo consumo de datos y energía.

La estructura de este trabajo se organiza de la siguiente manera. El Capítulo 1 presenta la contextualización, requerimientos e importancia de un *Gateway*. Capítulo 2 define los requisitos para la elaboración y el funcionamiento. Capítulo 3 presenta los resultados obtenidos, conclusiones y recomendaciones.

1. DIAGNÓSTICO DE NECESIDADES Y REQUERIMIENTOS

1.1 Ámbito de aplicación: descripción del contexto y hechos de interés

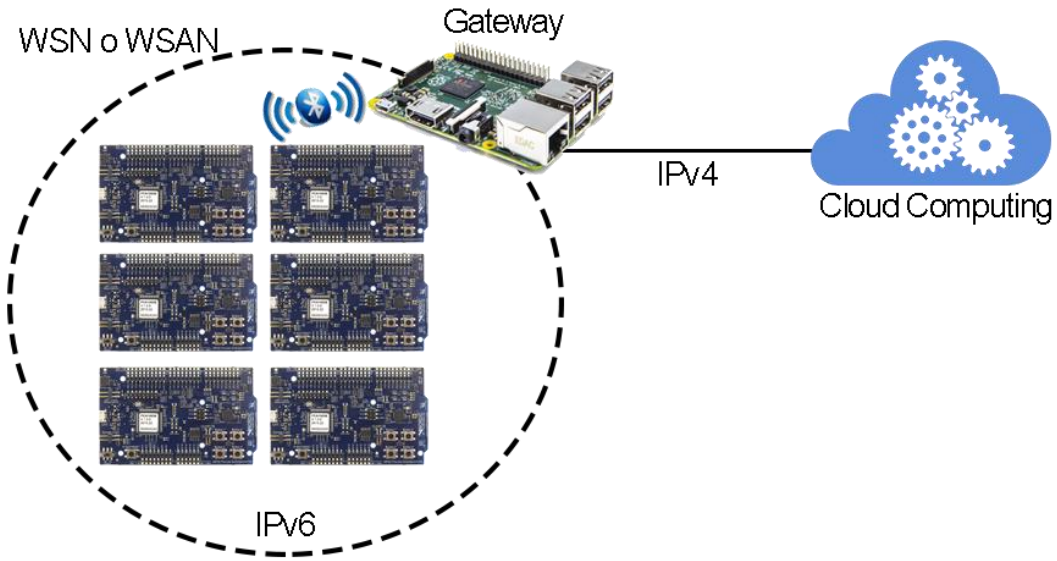
La Figura 1 muestra un Gateway o puerta de enlace que debe comunicarse e intercambiar la información entre dos redes y tecnologías diferentes, por ello este desempeña un rol importante en una WSN o WSAN, donde no debe tener anomalías para aumentar la fiabilidad en la red y estabilidad en los datos [15].

En una WSN o WSAN, los motes no necesitan conexiones físicas con el Gateway y pueden: recoger información a través de sensores (para medir alguna variable de su alrededor), o activar algún actuador (para realizar una acción específica).

La tecnología inalámbrica *Bluetooth* desde su versión 4.0.0 puede trabajar con BLE (*Bluetooth Low Energy*) o también conocida como *Bluetooth* inteligente, la cual entre sus principales ventajas son: corto alcance y bajo costo de recursos [16]; BLE usa el estándar IEEE 802.15.4.

Los nodos sensores o la Cloud Computing pueden intercambiar información con el Gateway usando diferentes protocolos o estándares, por ejemplo: el protocolo MQTT o el estándar *Eddystone*.

Figura 1. Esquema general de la IOT



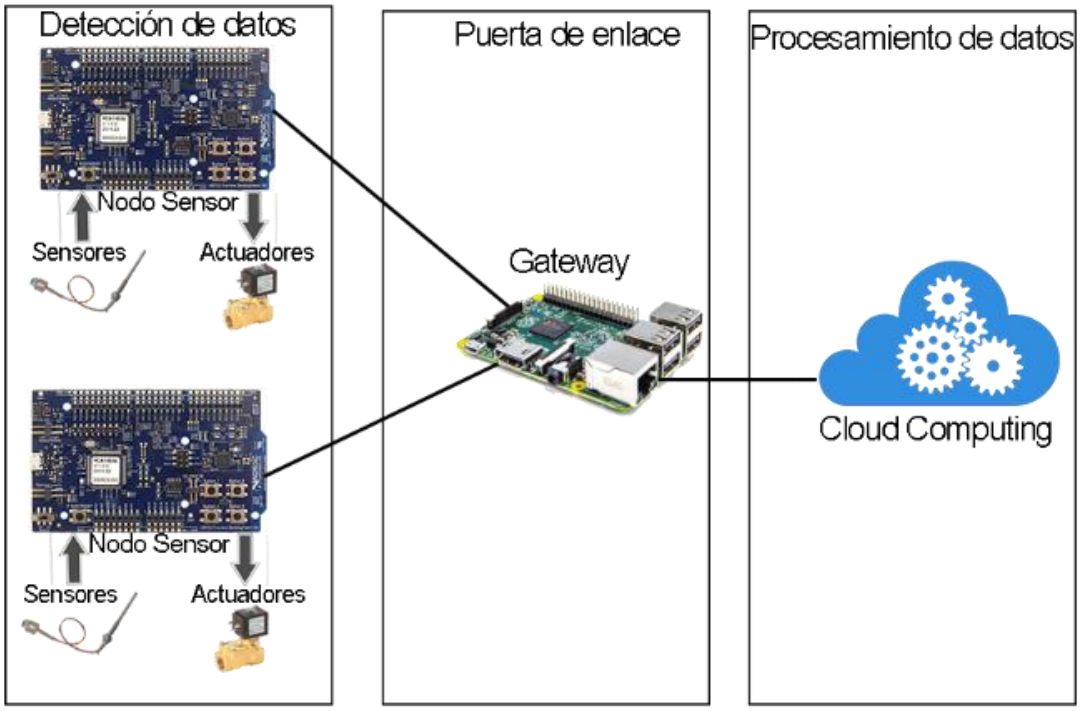
Fuente: Elaboración propia

1.2 Establecimiento de Requerimientos

La Figura 2 presenta un modelo de arquitectura de la IOT basado en la Referencia [17], la cual se divide en tres dominios que son: detección de datos, puerta de enlace y procesamiento de datos [17].

La detección de datos recolecta información de los sensores o ejecuta órdenes a los actuadores, una puerta de enlace que es la encargada de pasar la información entre los dominios de detección y procesamiento de datos o viceversa, el procesamiento de datos se encarga de realizar un trabajo sofisticado de ingeniería, donde se almacena y trata la información para las aplicaciones que lo requieran.

Figura 2. Arquitectura IOT



Fuente: Elaboración propia

Este trabajo se enfoca en buscar soluciones a los diferentes problemas en el dominio de la puerta de enlace o Gateway, estos son:

- El dominio de detección de datos que hace referencia a la WSN o WSAN, que está diseñada con IPv6, mientras el dominio de procesamiento de datos que abarca la *Cloud Computing*, está diseñado con el protocolo IPv4.
- En una WSN con tecnología inalámbrica BLE, existen nodos sensores que se comunican con el Gateway usando el estándar *Eddystone*, mientras que el envío de

datos del Gateway hacia la *Cloud Computing,* se hace con el protocolo MQTT. Se necesita obtener la información de los paquetes enviados por el estándar *Eddystone* para empaquetarlos bajo el protocolo MQTT.

- La WSN o WSAN usa la tecnología inalámbrica BLE, la cual no cuenta con un alcance de señal alto, la *Cloud Computing* se encuentra a Kilómetros de distancia.

1.3 Justificación del Requerimiento a satisfacer

Hoy en día existen muchos objetos inteligentes capaces de interactuar entre sí por medio del internet, pero para logar ese propósito se necesita comunicación con varios servicios al mismo tiempo, por lo cual consumen una gran cantidad de energía y datos de la red. A su vez estos servicios necesitan estar instalados en un hardware que permita la conexión de miles de dispositivos conectados e intercambiando información. Estos servicios actualmente pueden estar configurados o trabajando mediante un modelo tecnológico llamado Cloud Computing, que permite gestionar varios recursos informáticos de una manera eficiente.

Estos dispositivos inteligentes se pueden agrupar en una misma red llamada WSN o WSAN y de esta manera se puede tener varios dispositivos que no necesitan intercambiar datos directamente al internet o *Cloud Computing*, permitiendo así utilizar en cada uno de ellos alguna tecnología de bajo consumo de energía y datos. Por ello se necesita de un Gateway que pueda comunicarse directamente con la Cloud Computing y una WSN o WSAN.

En la actualidad los Gateway IOT inalámbricos que existen, tienen un precio elevado o su funcionabilidad no satisface completamente los requerimientos de los usuarios, por ejemplo: El Gateway IOT Meshlium tiene un costo de 930,00 Euros y funciona solo para los dispositivos Waspmote de Libelium [18]. Por ello se propone el desarrollo de un Gateway IOT inalámbrico (BLE), usando estándares, tecnología o protocolos de la IOT *open source*.

Para realizar el Gateway IOT se utiliza la metodología RAD (*Rapid Application Development*), fue introducida por primera vez por James Martin en 1980 [19], se recomienda para los proyectos en los cuales los requisitos se entienden bien, ayudando a entregar un software escalable y en breve plazo [20]. La referencia [19] menciona que el desarrollo rápido de aplicaciones tiene cuatro etapas, que son:

Planificación de necesidades: Es la etapa más importante dentro del desarrollo rápido de aplicaciones, ya que es aquí donde se debe reunir con un cliente especializado en

los procesos que se desea automatizar o desarrollar, el cual brinda los requisitos del sistema.

El diseño: Se realiza un estudio de los requisitos brindados por el cliente en la etapa anterior y se va bosquejando las primeras versiones del prototipo, las cuales son validadas por el cliente.

La construcción: Es la parte más importante de esta metodología, es aquí donde se desarrolla el diseño realizado en la etapa anterior. Diseño y construcción se deben trabajar en paralelo [19].

Corte y cambio: En esta etapa, se realizan pruebas y mantenimiento del prototipo.

Con este trabajo se puede introducir la IOT en los sectores productivos, por ejemplo: la agricultura, avicultura, ganadería, etc, monitoreando información o realizando acciones en tiempo real y desde cualquier lugar, con protocolos y tecnología de bajo consumo de energía y datos.

2. DESARROLLO DEL PROTOTIPO

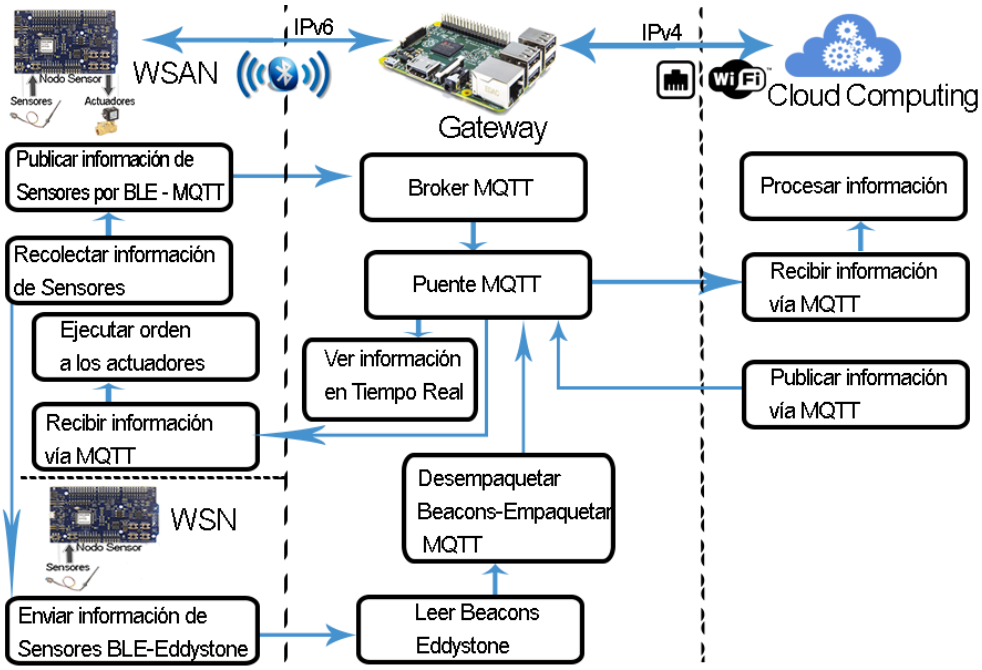
2.1 Definición del prototipo tecnológico

El grupo de investigación IOTMACH de la Universidad Técnica de Machala, se encarga de realizar investigaciones y desarrollar soluciones sobre la IOT en los sectores productivos, por ejemplo: la agricultura y precisión.

El prototipo tecnológico se basa en unir por medio de un Gateway IOT, la Cloud Computing y la WSN o WSAN del sistema IOTMACH. El proceso se realiza cuando un nodo sensor envía/recibe información al Gateway, mediante la tecnología inalámbrica BLE, transmitiendo paquetes en el protocolo MQTT o el estándar *Eddystone* con IPv6 a la Cloud Computing, como se aprecia en la Figura 4.

Cuando el Gateway IOT, recibe información de un paquete con el protocolo MQTT, éste es retransmitido mediante una interfaz que tenga comunicación con la *Cloud Computing*, puede ser: Wi-Fi o Ethernet, las cuales tienen una configuración IPv4, se utiliza un puente (*bridge*) MQTT para convertir el paquete de IPv6 a IPv4 o viceversa, y luego es enviado por la interfaz correspondiente. Si se recibe un paquete con el estándar *Eddystone*, se desempaqueta los *Beacons* y se empaqueta con el protocolo MQTT, para ser enviado al puente del *Broker* MQTT, y realizar el proceso descrito anteriormente.

Figura 3. Gestión del Prototipo

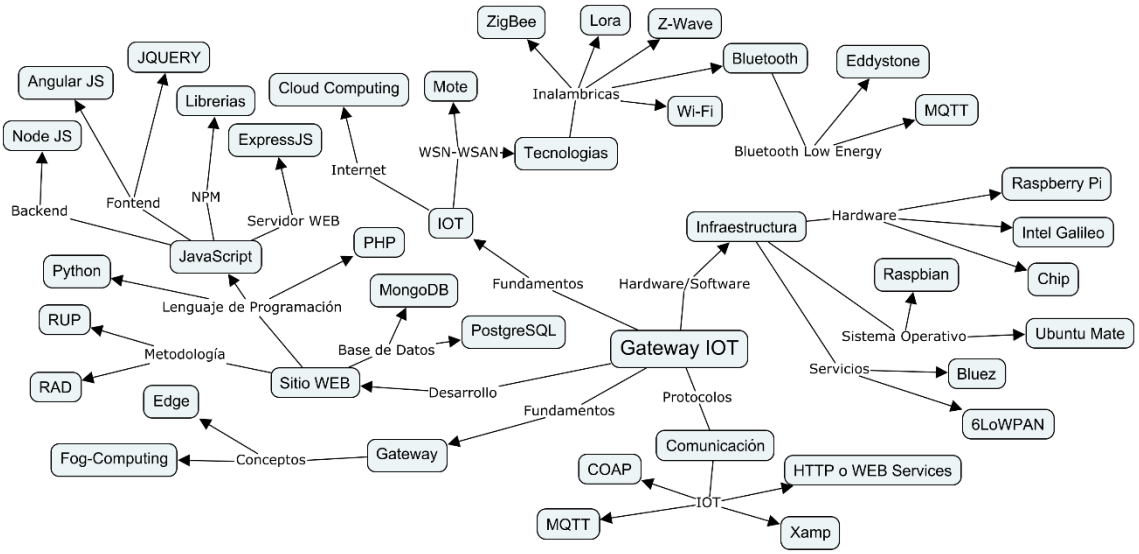


Fuente: Elaboración propia

2.2 Fundamentación teórica del prototipo

En la Figura 4 se puede observar el estado del arte, relacionada a las tecnologías, fundamentos, protocolos, estándares, hardware, software y herramientas consultadas para el desarrollo del Gateway IOT, del sistema IOTMACH.

Figura 4. Mapa conceptual de la fundamentación teórica



Fuente: Elaboración propia

2.2.1 Fundamentos de la IOT. Las diferentes definiciones para la IOT han ido evolucionando a medida que la tecnología y las ideas avanzan, recientemente se ha convertido en un campo de estudio en Tecnologías de la Información, incluso muchos países han invertido una gran cantidad de dinero para el desarrollo de la IOT [21]. La IOT, permite monitorear o controlar objetos inteligentes desde el internet. En Internet actualmente es común utilizar una *Cloud Computing* para varios propósitos, por ejemplo: brindar servicios WEB.

Si un conjunto de motes en una misma red tienen conectados sensores, estos se denominan WSN, si adicionalmente también poseen actuadores, se hace referencia a una WSAN. Una WSN o una WSAN, usan tecnología inalámbrica, por ejemplo: *ZigBee, Wi-Fi, Lora, Z-wave, Bluetooth*.

En la tecnología inalámbrica *Bluetooth* desde la versión 4.0, se puede activar y usar BLE, adicionalmente se puede agregar estándares o protocolos de bajo consumo de datos y energía, por ejemplo: *Eddystone*.

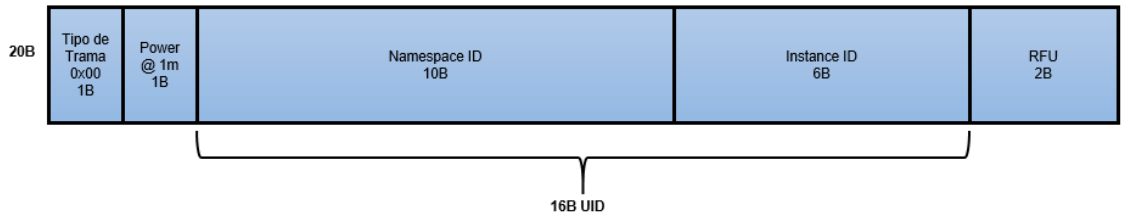
2.2.1.1 *Tecnología Inalámbrica: Bluetooth Low Energy o Bluetooth Smart*. Es una tecnología inalámbrica de bajo consumo, desarrollado para el control en corto alcance y aplicaciones de monitoreo de datos, se espera que sea incorporado en miles de millones de dispositivos en los próximos años [22], la referencia [23] lo considera como el principal candidato para conectar el IOT, funciona desde la versión de *Bluetooth* 4.0.

BLE opera en una banda de 2,4 GHz, sus principales objetivos son el bajo consumo de energía, latencia, y rendimiento, que dependen de algunos parámetros. La vida útil de un dispositivo de BLE alimentado por una batería de tipo botón oscila entre 2 días y 14 años. Aunque BLE trabaje sobre un módulo de *Bluetooth* clásico, actualmente ambos son incompatibles. Un dispositivo que sólo implementa BLE no puede comunicarse con un dispositivo que sólo implementa *Bluetooth* clásico. Hay dos tipos de canales de RF BLE que son: publicidad y datos. El canal de publicidad se usa para la detección, conexión y transmisión de difusión de dispositivos. Mientras el canal de datos se utiliza para la comunicación bidireccional entre dispositivos conectados [22].

2.2.1.2 *Tecnología Inalámbrica: Estándar Eddystone*. Transmite mensajes de baja energía BLE utilizando *Beacons* (pequeños fragmentos de información), que pueden ser leídos cada vez que un dispositivo realiza un anuncio (*Advertised*) [24]. Los *Beacons* son transmitidos en tramas con un límite de 20Bytes, el estándar Eddystone usa tres tipos de tramas que son:

- *Eddystone-UID*: Usa 16Bytes para transmitir el UID, que está compuesto por un identificador de instancia (6Bytes) y el nombre de espacio (10Bytes) del dispositivo, como lo muestra la Figura 5.

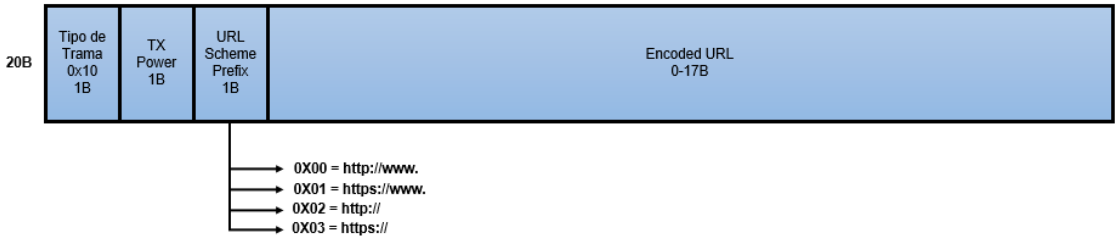
Figura 5. Eddystone-UID



Fuente: Elaboración propia

- *Eddystone-URL:* En esta trama viaja una URL que no puede utilizar más de 17 bytes, el prefijo de la URL ocupa un byte aparte como lo muestra la Figura 6.

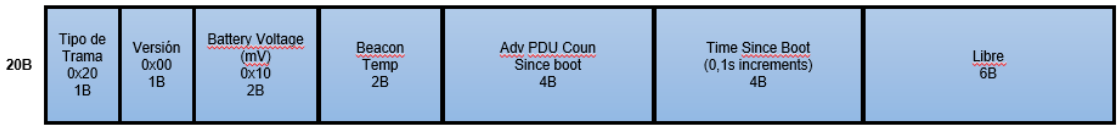
Figura 6. Eddystone-URL



Fuente: Elaboración propia

- *Eddystone TLM*. Esta trama se usa para enviar información como el estado de voltaje de la batería, temperatura, pero solo se ocupan 14Bytes, dejando 6Bytes libres como lo muestra la Figura 7.

Figura 7. Eddystone-TLM



Fuente: Elaboración propia

2.2.1.3 *Cloud Computing.* Consiste en compartir efectivamente cada uno de los recursos computacionales y donde todo se consume como un servicio [25]. Gartner en un informe reciente, proyecta que el mercado de servicios en la Cloud Computing incremente un 16,5% en el 2016, esto da un total de $204 millones de dólares, frente a $175 millones de dólares del año 2015 [26].

La capacidad de almacenamiento de los nodos sensores o Gateway de una WSN o WSAN es limitada, sin embargo estos datos son importantes, porque permiten determinar ciertas acciones, una Cloud Computing puede garantizar la seguridad, fiabilidad de los datos de la WSN o WSAN, para recolectar la información se utiliza técnicas de minería de datos [27].

2.2.2 Concepto de Gateway IOT. Es un enlace entre la WSN o WSAN y la comunicación por internet (Cloud Computing), el Gateway IOT puede acceder directamente a un recurso de la Cloud Computing o comunicarse directamente con una WSN o WSAN siempre y cuando este en la misma red [27].

Actualmente las investigaciones sobre la IOT, se enfocan en minimizar los gastos de recursos informáticos, para de esta manera reducir los tiempos de respuesta. En el Gateway se propone un modelo de procesamiento conocido como Fog Computing, es la extensión de la Cloud Computing, pero en el borde de la red y se caracteriza por ser de baja latencia y tiempo homogéneo [28].

2.2.3 Protocolos de comunicación para la IOT. En la capa de aplicación, existen muchos protocolos de comunicación de red, para dispositivos de la IOT, por ejemplo: HTTP, MQTT, COAP [29].

Actualmente los protocolos de comunicación para la IOT más conocidos son: MQTT y COAP, la diferencia entre estos dos protocolos se e encuentra en el trabajo realizado por la referencia [30].

2.2.3.1 *Protocolo COAP*. Es un protocolo desarrollado para dispositivos con recursos limitados, el funcionamiento de COAP es muy parecido al de cliente/servidor de HTTP, la principal diferencia radica en que COAP realiza las peticiones de forma asíncrona, enviando mensajes UDP.[31]. Por ejemplo cuando se realiza una solicitud GET con HTTP, el cliente no pude realizar otra acción sobre el mismo hilo de solicitud, mientras que con COAP si se realiza una petición GET sobre el mismo hilo de peticiones, el cliente puede realizar cualquier otra solicitud.

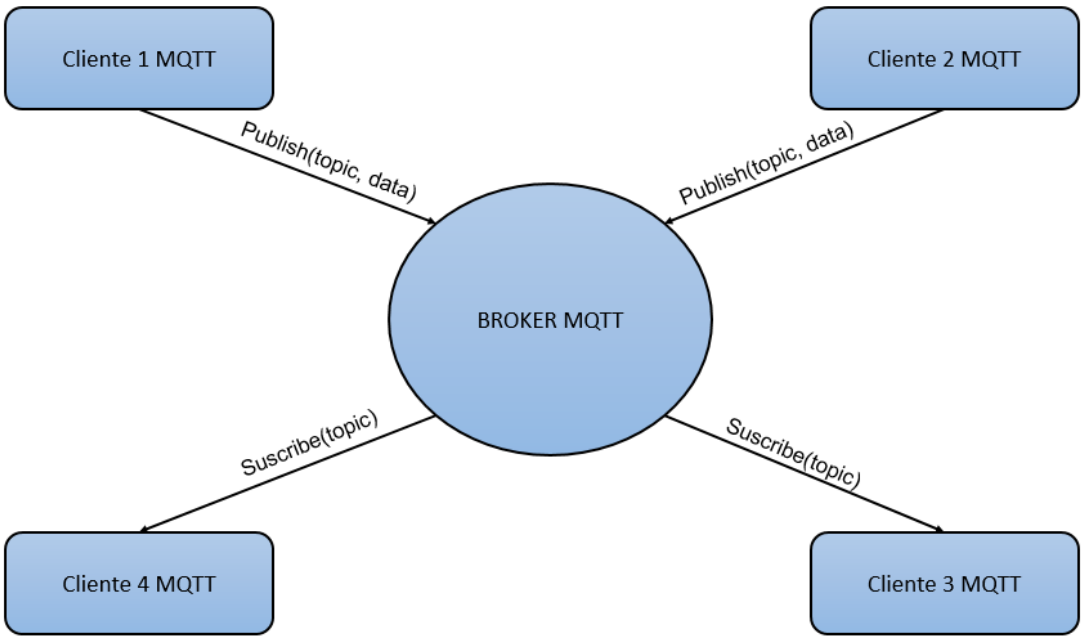
COAP también es compatible con el modelo de arquitectura suscripción/publicación, para ello usa un GET extendido hacia una URL en lugar de usar los tópicos como los que trabaja MQTT. La principal diferencia es que COAP utiliza paquetes UDP mientras MQTT paquetes TCP/IP [32].

2.2.3.2 *Protocolo MQTT*. Es un protocolo que trabaja en la capa de aplicación de la pila TCP/IP, está basado para que funcione en dispositivos con recursos limitados, la principal características es que ocupa poco ancho de banda para transmitir mensajes, la arquitectura que trabaja es publicación/suscripción, enviando los mensajes de M2M (Máquina a Máquina) [32].

En la Figura 8 se puede observar el funcionamiento del protocolo MQTT, donde existe un Broker MQTT, cada uno de los clientes pueden suscribirse o publicar a uno o más tópicos, por ejemplo: El cliente A y B están suscritos al tópico /saludo/ y el cliente C publica el mensaje hola al tópico /saludo/, A y B podrán escuchar el mensaje publicado,

porque al estar suscritos el Broker hace una especie de *Broadcast* hacia el tópico que se publica.

Figura 8. Arquitectura MQTT



Fuente: Elaboración propia

En MQTT también se puede pasar mensajes de uno o más tópicos hacia otros Broker, por ejemplo: El cliente A publica en el tópico /saludo/ del bróker A1 el mensaje hola, el bróker A1 está haciendo un puente hacia el Broker B1, el cual tiene al cliente B suscrito al tópico /saludo/, el mensaje hola lo escucha el cliente B gracias al puente que logra retransmitir el mensaje entre Broker.

2.2.4 Infraestructura del Gateway IOT. Lo fundamental de un Gateway IOT es: El Hardware que es la parte física, Sistema Operativo que permite controlar o activar funcionalidades y los Servicios que permiten gestionar los módulos físicos, por ejemplo: El *Bluetooth*.

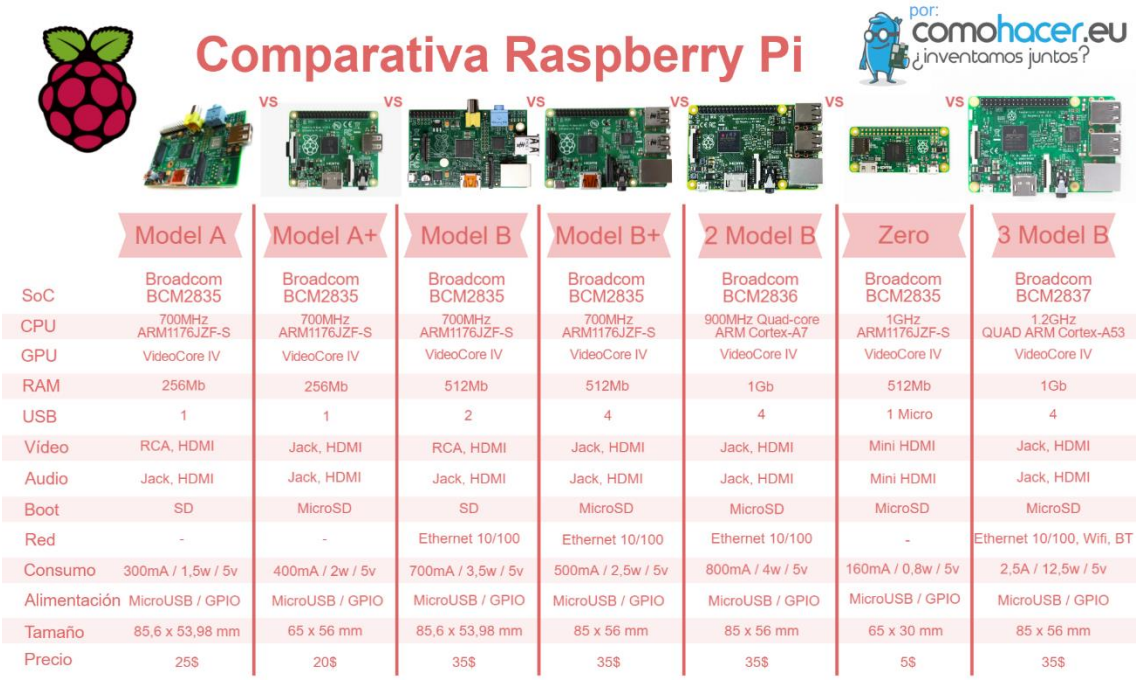
Para el Hardware de un Gateway IOT existen diferentes tecnologías, por ejemplo: Raspberry Pi, Chip, Intel Galileo. Son tarjetas o computadoras pequeñas y de placa única.

Existen diferentes sistemas operativos, por lo general la mayoría son basados en Linux, por ejemplo: Raspbian, Ubuntu Mate.

Los servicios que se pueden instalar o configurar en los sistemas operativos son varios y permiten gestionar de una manera fácil el hardware, por ejemplo: Bluez.

2.2.4.1 *Hardware: Raspberry Pi 3 Model B*. Es una computadora de placa única, muy pequeño, pesa aproximadamente 50g y se puede instalar un sistema operativo mediante una tarjeta micro SD [33]. Actualmente existen varias versiones y con claras diferencias, la Figura 9 muestra una comparativa entre algunas placas de Raspberry Pi.

Figura 9. Comparativa Raspberry PI



Fuente: Obtenida de la Referencia [34]

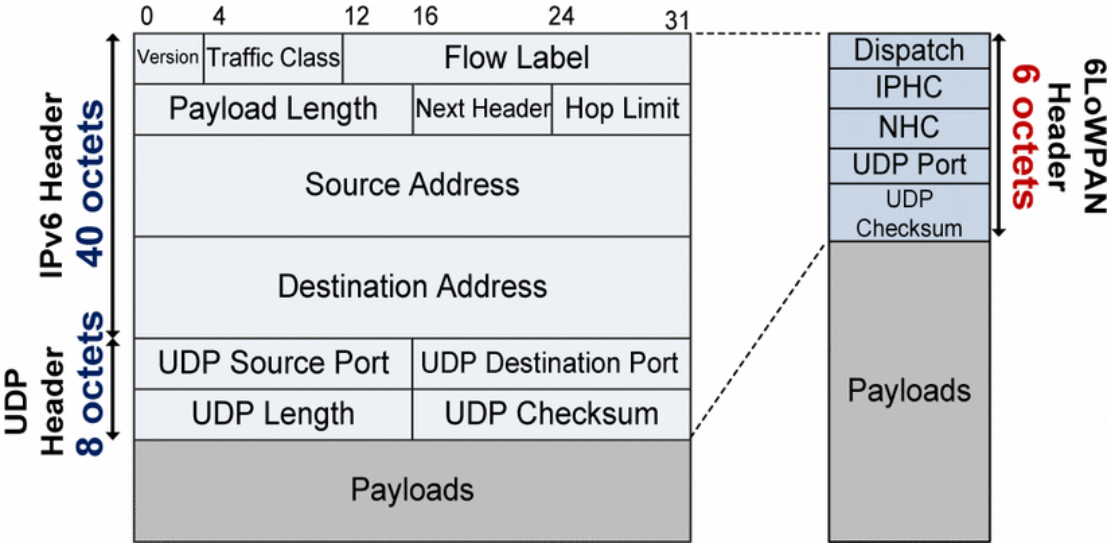
Entre las principales características de la Raspberry Pi 3 modelo B, se puede mencionar que: cuenta con un módulo de *bluetooth* 4.1 y Wi-Fi ya incorporado, la memoria RAM es de 1 GB, y la fuente de alimentación de energía eléctrica es de 5v.

2.2.4.2 *Sistema Operativo: Raspbian*. Los equipos electrónicos con un procesador pueden ejecutar un sistema operativo, para gestionar; los sistemas de archivos, dispositivos periféricos, proporcionar la interfaz de usuario e inicio del dispositivo, *Raspbian* es un sistema operativo libre, derivado de Debian Linux, optimizado para el uso del hardware de Raspberry Pi, está basado en procesadores ARM [35], se instala en una tarjeta SD y cuenta con su propio repositorio [36].

2.2.4.3 *Servicio: 6LoWPAN (Pv6 over Low-power Wireless Personal Area Network)*. Es un estándar definido por el IETF (*Internet Engineering Task Force*), que propone una capa de adaptación entre la capa de red y la capa MAC (*Medium Access Control*), la cual permite la transmisión de IPv6 sobre IEEE 802.15.4, para soportar muchos tipos de servicios. Esta tecnología está ganando rápidamente popularidad por su amplia aplicabilidad, que van desde la salud a la vigilancia del medio ambiente. Con el fin de proporcionar conectividad IPv6 más fiable y eficaz [10].

La figura 10 muestra la estructura de un paquete 6LoWPAN, que tiene poca sobrecarga, por lo que es de peso ligero para ser más eficiente para la transferencia de datos [10]. A través de la dirección física (MAC), se realiza una conversión para asignarle una dirección IPv6.

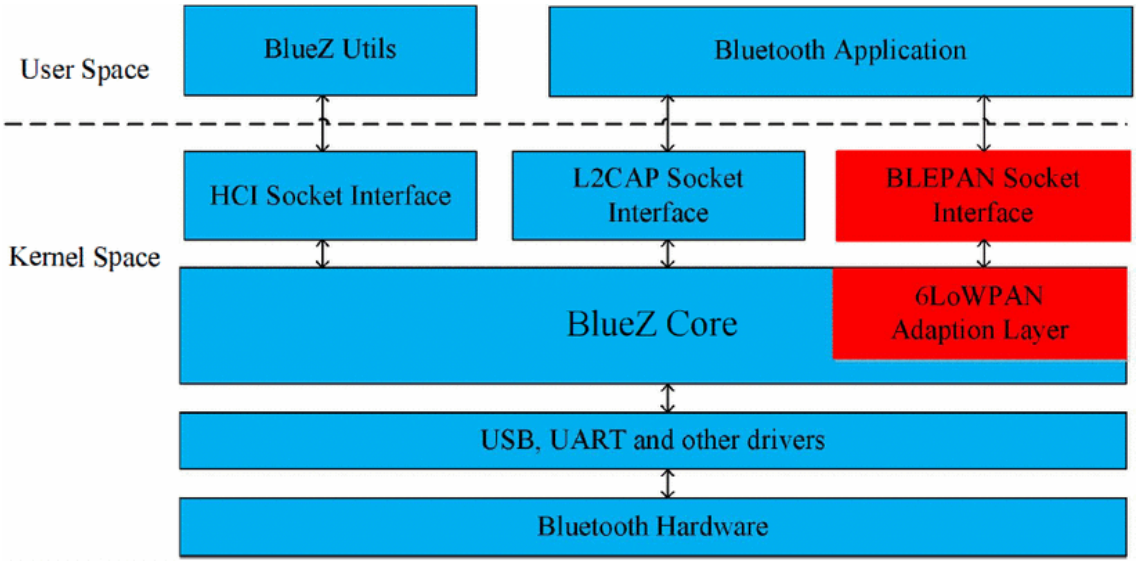
Figura 10. Paquete 6LoWPAN



Fuente: Obtenida de la Referencia [10]

2.2.4.4 *Servicio: BlueZ*. Es la librería que permite instalar y configurar los módulos de *Bluetooth,* en cualquier sistema operativo basado en Linux, además permite trabajar con IPv6 a través de 6LoWPAN, La Figura 11 muestra cómo está conformado *BlueZ* [37]. Aunque la librería es difícil de manipular directamente desde el código fuente, al tener incorporado HCIDUMP, se puede controlar de forma fácil cada conexión y datos [38].

Figura 11. Diagrama general de BlueZ



Fuente: Obtenida de la referencia [37]

2.2.5 Herramientas para el desarrollo WEB. En el desarrollo de sitios WEB se utilizan varios procesos de informática, por ejemplo: Lenguaje de Programación, Base de Datos, metodologías, librerías.

Actualmente los lenguajes de programación del desarrollo de sitios WEB, se dividen en; Backend (Programación en el lado del servidor) y Frontend (Programacion en el lado del Cliente). Para el Backend existen una gran variedad de lenguajes, por ejemplo: Python, PHP, JavaScript. En el Frontend el lenguaje más usado es JavaScript.

Hoy en día las Bases de Datos se dividen en dos tipos que son: Relacionales (SQL) y no Relacionales (NoSQL). La diferencia entre los dos tipos de Bases de Datos radica en que las no Relacionales, procesan con mayor rapidez grandes volúmenes de datos [39]. Un ejemplo de Bases de Datos no Relacionales son: MongoDB, Redis, Cassandra.

Para que el desarrollo de un software o sitio WEB sea; escalable, fiable y cumpla con la solución del problema planteado, se debe aplicar una metodología de ingeniería de software. Una metodología brinda las técnicas para: recopilar información, bosquejar la idea del cliente o usuario y la forma en que se debe desarrollar. Actualmente existen muchas metodologías, por ejemplo: RAD.

2.2.5.1 *Lenguaje de Programación: JavaScript*. Es un lenguaje de programación dominante en el desarrollo de aplicaciones WEB, se lo puede encontrar en el lado del cliente o servidor, es dinámico y fácil de aprender, no existe un estándar de la sintaxis, por ello se recomienda ser muy ordenado al momento de programar, comúnmente es

utilizado para el desarrollo del Frontend de la mayoría de sitios WEB, 13 de cada 15 aplicaciones WEB lo utilizan de una u otra manera [40].

2.2.5.2 *Lenguaje de Programación Backend: Node JS*. Es sinónimo de una nueva tecnología en JavaScript, se pueden construir aplicaciones de una manera rápida y escalable, es un lenguaje de programación ideal para construcción de sistemas en tiempo real, utiliza un modelo orientado a eventos sin bloqueo de E/S que hace que sea ligero y eficiente [41]. Se resume como: el ejecutar JavaScript del lado del Servidor, uno de sus principios es manejar peticiones asíncronas, ya que de esta manera no se realizan bloqueos en el servidor. Node JS maneja un solo hilo de ejecución.

2.2.5.3 *Framework Node JS: Express*. Es un Framework que se basa en Node JS, proporcionando un servidor de aplicaciones Web, alrededor de una interfaz de bajo nivel: dando a los desarrolladores, un medio conveniente para manejar el enrutamiento de URLs a través de peticiones HTTP (GET, DELETE, PUT y POST) [42].

2.2.5.4 *Framework Frontend: Angular JS*. Es un Framework del lado del cliente para aplicaciones WEB, fue desarrollado por *Google*, trabaja con una arquitectura MVC (Modelo, Vista, Controlador). Para aprovechar al máximo este Framework, se lo puede combinar con el uso de *Bootstrap* (Framework de diseño creado por Twitter) [42]. Cuando se usa Angular JS, el desarrollador se ahorra escribir líneas de código al desarrollar el Frontend de un sitio web, porque una de las principales fortalezas de este Framework es reutilizar componentes.

2.2.5.5 *Base de Datos: MongoDB*. Es una base de datos no relacional o también llamada NoSQL, para el almacenamiento de datos usa documentos en formato JSON, se adapta perfectamente a un lado del servidor JavaScript, es ágil y escalable. En Octubre del 2015, MongoDB fue la base de datos más popular orientada a documentos NoSQL y cuarto en sistema de gestión de base de datos en general según lo medidor por “Knowledge Base of Relational and NoSQL Database Management Systems” [42].

2.2.5.6 *Metodología de ingeniería de software: RAD*. Para aplicar esta metodología en el desarrollo de un sistema WEB o software, es necesario tener cada uno de los requisitos bien establecidos. Utiliza herramientas para generar código de una manera rápida y se debe emplear la reutilización de código [20].

2.3 Objetivo del Prototipo

2.3.1 Objetivo General. Implementar un Gateway IOT integrado al sistema IOTMACH, utilizando la tecnología inalámbrica BLE, el estándar Eddystone y el Protocolo MQTT, para que enlace una WSN o WSAN con una Cloud Computing.

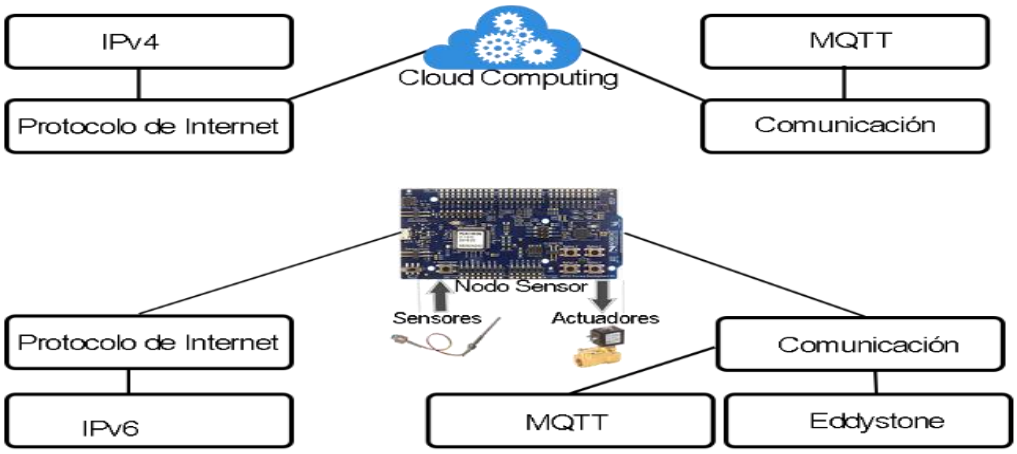
2.3.2 Objetivos Específicos. Para cumplir el objetivo general se planteó los objetivos específicos siguientes:

- Realizar una investigación bibliográfica sobre las tecnologías inalámbricas, estándares, protocolos usados en la IOT.
- Analizar y Diseñar un prototipo de Gateway IOT de acuerdo a los requerimientos planteados.
- Instalar y configurar; el sistema operativo, servicios, protocolos y librerías del Gateway IOT.
- Desarrollar un sitio WEB que permita: la manipulación de las tramas Eddystone o MQTT, la gestión de las interfaces de comunicación (Wi-Fi, Bluetooth, Ethernet) y el monitoreo/control de la WSN o WSAN.
- Evaluar el funcionamiento y eficiencia del Gateway IOT.

2.4 Diseño del Prototipo

En la Figura 12 se observa cada una de las tecnologías, estándares y protocolos que el Gateway IOT necesita utilizar, para enlazar una WSN o WSAN con una Cloud Computing.

Figura 12. Tecnologías, estándares, protocolos de la WSN y Cloud Computing

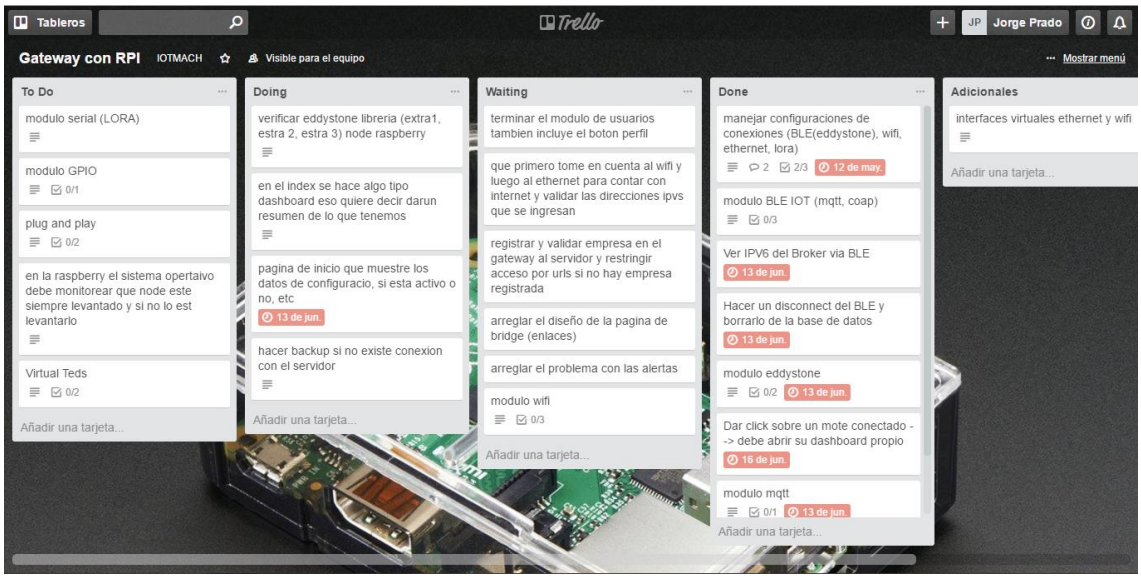


Fuente: Elaboración propia

2.4.1 Planificación. Se realizó una planificación de cada una de las actividades a ejecutarse en el Diseño y Desarrollo del prototipo, esto permite tener una organización, ya que se puede conocer las tareas; desarrolladas, pendientes y en ejecución. .

En la Figura 13 se puede observar la planificación de cada una de las actividades, para esto se utilizó Trello, que es un sistema WEB donde se puede organizar tareas en forma de notas, agregándole una fecha máxima de entrega, cada actividad se encuentra en una columna que representa su estado, que puede ser: por hacer, trabajando, esperando, hecho, o alguna columna adicional que se le desee agregar.

Figura 13.Actividades para el desarrollo del Gateway



Fuente: Elaboración propia

2.4.2 Requisitos del Prototipo. En esta sección se analizó cada uno de los requerimientos a resolver, para cumplir con los objetivos planteados. Los requisitos ayudan a detallar los módulos o funcionalidades a desarrollar en el Gateway IOT.

2.4.2.1 *Requisitos de Hardware*. Para el Hardware se necesita un dispositivo inteligente, que permita: procesar información, almacenamiento mínimo de 40GB y tenga integrada una tarjeta Wi-fi y Bluetooth superior o igual a la versión 4.0.

2.4.2.2 *Requisitos de Sistema Operativo y servicios*. Se necesita un sistema operativo basado en Linux y ligero, para optimizar el almacenamiento. Como servicio se necesita instalar y configurar BlueZ y 6LoWPAN.

2.4.2.3 *Requisitos para el Desarrollo*. Analizando los requerimientos se realizó una lista de requisitos funcionales, la cual se dividió en las siguientes funciones:

- Verificación de Empresa. Esta función permite comprobar que exista en la C*loud Computing* el registro ingresado (solo se realiza una sola vez), en la Tabla 1 se muestra el detalle del requisito.

Tabla 1. RF01 Requisitos de verificación de Empresa

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R1	Ingresar	Insertar el RUC de empresa.	RUC	Obligatorio	Se envía el dato a una URL en la Cloud.	Alta
R2	Obtención	Obtiene los datos de la empresa desde la Cloud	RUC Nombre	Alfanumérico Alfanumérico	Los datos son obtenidos en formato JSON.	Alta
R3	Guardar	Guardar empresa	Datos R2	Según R2	Solo puede existir una empresa.	Alta

Fuente: Elaboración propia

- Usuarios. Se realiza una gestión de ingresar, buscar, modificar, eliminar y autenticar para los usuarios, de esta manera el sitio WEB del Gateway puede ser administrador por varios usuarios, la Tabla 2 especifica los requisitos considerados.

Tabla 2. RF02 Requisitos de Usuarios

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R4	Inserción	Ingreso de datos del Usuario.	Nombre, Apellido	Datos de información del Usuario.	Se debe crear un usuario por defecto con nombre de usuario:Admin y Contraseña: Admin.	Baja
			Nombre de Usuario	Un nombre de usuario único.		
			Contraseña	Mayor a 6 caracteres y menor a 20 caracteres.		
R5	Búsqueda	Consulta los datos de Usuario.	Datos R4	Criterio: Nombre de Usuario.	Busca en la tabla Usuarios.	Media
R6	Modificación	Modificación de los datos de Usuario	Datos R4	Buscar antes según R5	Validar los nuevos datos según las restricciones de R4	Baja
R7	Eliminación	Eliminación de los datos del usuario	Datos R4	Se listan todos los usuarios	Eliminar datos de usuario	Baja
R8	Autenticación	Autenticación de usuario para ingresar al sistema	Nombre de Usuario Contraseña	Datos obligatorios	Se debe verificar los datos en la tabla Usuarios.	Alta

Fuente: Elaboración propia

- Enlace con la WSN o WSAN. En la Tabla 3 se observa la especificación de los requisitos para los enlaces (BLE – MQTT, BLE - Eddystone), que van a intercambiar información con la WSN, de esta manera el usuario puede establecer que enlace está activo.

Tabla 3. RF03 Requisitos de enlace con la WSN o WSAN

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R9	Inserción	Ingreso de los datos de enlace con la WSN.	Nombre	alfanumérico único	Las opciones se crean una sola vez, al iniciar por primera vez el sistema.	Alta
			Identificador	alfanumérico		
			Activo	Valor booleano		
			Icono	alfanumérico		
R10	Búsqueda	Consulta interfaz.	Datos R9	Criterio: Nombre.	Buscar interfaz	Alta
R11	Modificación	Modifica de ajuste	Activo	Buscar según R10	Se modifica únicamente el campo activo	Alta

Fuente: Elaboración propia

- Enlace con la *Cloud Computing.* En la Tabla 4 se observa la especificación de los requisitos, para las opciones de enlace de datos que van a intercambiar información con la *Cloud Computing*.

Tabla 4. RF04 Requisitos de enlace con la *Cloud Computing*

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R12	Inserción	Ingreso de datos enlaces de salida.	Datos R9	Según R9	Las opciones se crean una sola vez, al iniciar por primera vez el sistema.	Alta
R13	Búsqueda	Consulta de enlace.	Datos R12	Criterio: Nombre de enlace.	Buscar enlace	Alta
R14	Modificación	Modifica enlace	Activo	Buscar antes según R10	Se modifica únicamente activo	Alta

Fuente: Elaboración propia

- Módulo *Ethernet*. Con este módulo se puede escribir y leer la configuración de la interfaz física E*thernet* con el fin de cambiar la dirección IP asignada, la cual puede estar con IPv4, IPv6 o ambas, en la Tabla 5 se puede observar los requisitos considerados.

Tabla 5. RF05 Requisitos de Interfaz *Ethernet*

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R15	Escritura	Escribir archivo de red Ethernet.	Dirección IPv4, Máscara de Red, Gateway IPv4 Dirección IPv6, Máscara de Red, Gateway IPv6	Configuraciones para IPv4 Configuraciones para IPv6	Si se escribe una dirección IPv4 obligatoriamente los datos de máscara de red y Gateway son obligatorios caso contrario no, si la dirección es IPv6 la máscara y Gateway se vuelven obligatorios caso contrario no.	Alta
R16	Lectura	Leer archivo de red Ethernet	Datos R15	Según R15	Se lee todos los datos sin excepción	Baja

Fuente: Elaboración propia

- Módulo Wi-Fi. Con los requisitos considerados en la Tabla 6 se puede gestionar la interfaz física Wi-Fi, para de esta forma poder buscar y obtener una lista de redes Wi-Fi cercanas, donde el usuario pueda seleccionar a cual conectarse o desconectarse.

Tabla 6. RF06 Requisitos de Interfaz Wi-Fi

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R17	Búsqueda	Buscar las señales Wi-Fi cercanas	SSID Tipo de Seguridad MAC Alcance	Nombre de la red Wi-Fi En caso no tener N/A Dirección MAC Intensidad	Se listan las señales Wi-Fi cercanas al Gateway.	Media
R18	Mostrar	Muestra lista de señales Wi-Fi cercanas	Datos R17	SSID, Tipos de Seguridad, alcance.	Se muestra la lista de señales Wi-Fi cercanas.	Media
R19	Eliminar	Elimina señales Wi-Fi que no estén al alcance	Datos R17	Señales que no puedan ser alcanzadas	Elimina las señales que son alcanzadas	Baja
R20	Conectar	Conectarse a una red Wi-Fi	Datos R17	Si tiene seguridad pedir contraseña	Si la señal tiene un tipo de seguridad el usuario debe ingresar la contraseña de la red	Media
R21	Desconectar	Desconectarse a una red Wi-Fi previamente conectado	Datos R20	Se puede desconectar sin restricción	Se puede desconectar de cualquier red Wi-Fi, si la red Wi-fi no está al alcance se desconecta automáticamente.	Media

Fuente: Elaboración propia

- Enlace WSN o WSAN con BLE – MQTT. Los requisitos de la Tabla 7 permite al usuario, gestionar que dispositivos de la WSN con BLE - MQTT pueden enviar o recibir información mediante el Gateway, con esto el usuario puede: controlar cada uno de los dispositivos conectados, conectarse con algún dispositivo nuevo, eliminar y desconectar un dispositivo.

Tabla 7. RF07 Requisitos BLE - MQTT

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R22	Búsqueda	Buscar las señales BLE con MQTT cercanas	Id MAC Nombre	Una dirección MAC sin los separadores y es única Una dirección MAC Nombre del dispositivo BLE	Se listan las señales BLE con MQTT cercanas al Gateway.	Alta
R23	Mostrar	Muestra la lista de señales BLE con MQTT	Datos R22	Se muestra nombre y MAC	Se muestra la lista al usuario	Alta
R24	Refrescar	Se elimina dispositivos BLE con MQTT	Datos R22	Se elimina toda la lista de dispositivos según R22	Se refresca cada 30 segundos la lista de dispositivos BLE con MQTT cercanos	Baja
R25	Conectar	Conectarse a un dispositivo BLE con MQTT	Datos R22	Si el dispositivo está cerca puede conectarse	Se conectar y agrega a la lista de dispositivos aceptados, este proceso se realiza automáticamente.	Alta
R26	Desconectar	Desconectarse de un dispositivo BLE con MQTT	Datos R23	Se desconecta y elimina de cualquier dispositivo.	Se muestra la lista de dispositivos aceptados para conectarse y se puede eliminar cualquiera sin importar si está dentro del alcance	Baja

Fuente: Elaboración propia

- Enlace WSN con BLE – Eddystone. Los requisitos de la Tabla 8 permite al usuario, gestionar que dispositivos de la WSN con BLE – Eddystone pueden enviar información al Gateway, con esto el usuario puede: controlar cada uno de los dispositivos aceptados, agregar un dispositivo nuevo, eliminar y desconectar un dispositivo.

Tabla 8. RF08 Requisitos BLE - Eddystone

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R27	Búsqueda	Buscar las señales BLE con Eddystone cercanas	Id	Una dirección MAC sin los separadores y es única	Se listan las señales BLE con Eddystone cercanas al Gateway.	Alta
			MAC	Una dirección MAC		
R28	Mostrar	Muestra la lista de señales BLE con Eddystone	Datos R27	Se muestra MAC	Se muestra la lista al usuario	Alta
R29	Eliminar	Se elimina dispositivos BLE con Eddystone	Datos R27	Se elimina toda la lista de dispositivos según R27	Se elimina el dispositivo BLE Eddystone que en 5 segundos no envía señal	Baja
R30	Conectar	Conexión lógica de dispositivo BLE con Eddystone	Datos R27	Se hace una conexión lógica, según R27	Se conectar de manera lógica (por base de datos).	Alta
R31	Desconectar	Desconectarse de un dispositivo BLE con Eddystone	Datos R28	Se desconecta y se elimina de cualquier dispositivo.	Se muestra la lista de dispositivos aceptados para conectarse y se puede eliminar cualquiera sin importar si está dentro del alcance	Baja

Fuente: Elaboración propia

- Enlace *Cloud Computing* mediante puente MQTT. Los requisitos de la Tabla 9 permiten al usuario, gestionar las conexiones hacia la Cloud Computing mediante un puente MQTT, el sistema se encarga de escribir y leer la configuración MQTT.

Tabla 9. RF09 Requisitos configuración Puente MQTT

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R32	Escritura	Escribir un puente MQTT.	Dirección del Servidor MQTT	Dirección del servidor MQTT, obligatorio	Para que los tópicos creados comiencen a conectarse se necesita reiniciar la Raspberry Pi.	Media
			Nombre de conexión	Alfanumérico, obligatorio		
			Usuario	Alfanumérico		
			Contraseña	Alfanumérico		
			Tópico Entrada	Tópico de entrada		
			Tópico salida	Tópico de salida		
			Direccionamiento del tópico	El tipo de direccionamiento puede ser: both, in, out.		

Tabla 9. (Continuación)

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R33	Lectura	Leer los datos de configuración de Puente MQTT	Datos R32	Según R32	Se lee todos los datos sin excepción	Baja
R34	Eliminar	Eliminar datos de puente MQTT	Datos R32	Según R32	Seleccionar la configuración de puente MQTT que se desea eliminar	

Fuente: Elaboración propia

- Transferencia de información. La Tabla 10 contiene los requisitos para el envío/recepción de datos con la WSN usando BLE - MQTT o BLE - Eddystone.

Tabla 10. FR10 Requisitos envió de datos

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R35	Suscribir	Suscribe al tópico /iotmach/configuracion/	Nombre	Alfanumérico	Los datos son recibidos en JSON, un ejemplo se encuentra en el Anexo F	Alta
			MAC	Dirección MAC		
			Empresa	alfanumérico		
			Localización	Alfanumérico		
			Descripción	Alfanumérico		
			Datos sensores y actuadores (Categoria, señal, ted, interfaz)	Un array de datos		
R36	Suscribir	Suscribe al tópico /wsn/lecturas/	Paquete_id	Alfanumérico	Los datos son recibidos en JSON	Alta
			MAC	Dirección MAC		
			Datos(interfaz, valor)	Un array de datos		
			Batería	Alfanumérico		
R37	Publicar	Publicar en el tópico /iotmach/lecturas/	Datos R36, fecha	Fecha: dd/mm/aaaa	Se publica un JSON en tópico que hace puente MQTT, ejemplo en el Anexo G	Alta
R38	Publicar	Publica en el tópico /iotmach/actuadores/	Paquete_id	String vacio	Se publica un JSON en tópico para él envió de comandos, ejemplo en el Anexo H	Media
			MAC	Dirección MAC		
			Datos(interfaz, valor)	Vector de datos		

Tabla 10. (Continuación)

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R38	Empaquetar	Empaqueta datos Eddystone	Datos R35, R36, R37	Según R35, 36, 37	Empaqueta JSON y publica según R35, 36, 37	Alta
Fuente: Elaboración propia						

- Unidad de Medida. La Tabla 11 indica los requisitos para obtener desde la *Cloud Computing,* la información de la unidad de medida de los sensores por cada mote.

Tabla 11. RF11 Requisitos Unidad de Medida

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R39	Obtener	Obtiene datos	Ted	Alfanumérico	Envía dato a una URL de la Cloud	Media
R40	Guardar	guarda la unidad	Medida	Alfanumérico	Según R39	Media
Fuente: Elaboración propia						

- Nodos Sensores. La Tabla 12 especifica los requisitos para obtener los datos de los sensores y actuadores de cada nodo sensor.

Tabla 12. RF12 Requisitos Nodo Sensores

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R41	Guardar	Guardar motes	Datos R22, 27 y 35	Según en R36	Indistinto si es MQTT o Eddystone	Alta
R42	Buscar	Buscar los datos de un mote	Datos R41	Según R41	Busca información de un mote	Media
Fuente: Elaboración propia						

- Dashboard. La Tabla 13 contiene los requisitos para los datos en tiempo real de un nodo sensor que esté conectado al Gateway mediante BLE MQTT o Eddystone.

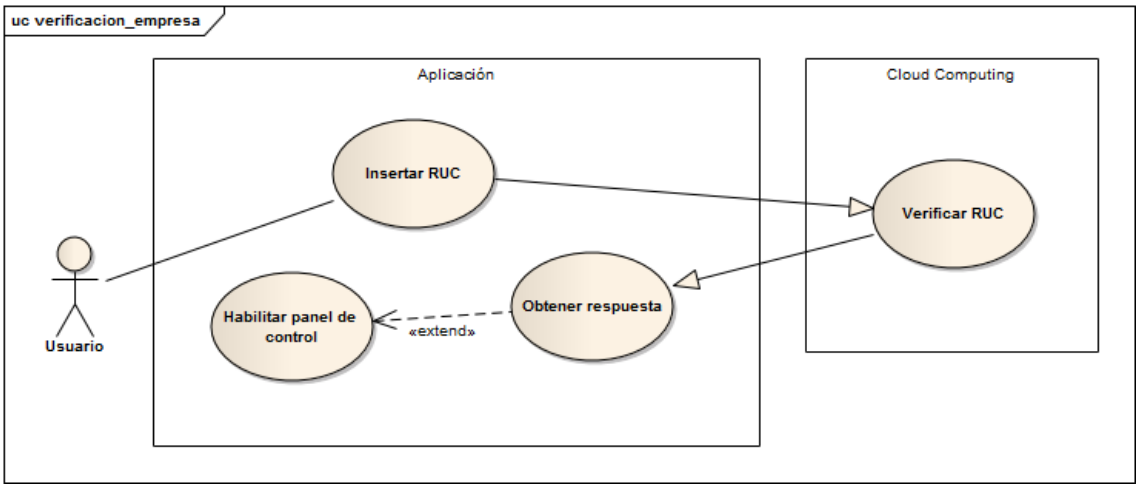
Tabla 13. RF13 Requisitos de Dashboard

Número	Tipo	Descripción	Datos	Restricciones	Observaciones	Prioridad
R43	Mostrar	En tiempo real	Datos R37, y 38	Según R42	Mostrar por cada mote	Alta
R44	Buscar	Busca datos mote	Según R42	Según R42	auto-configuran los componentes.	Media
Fuente: Elaboración propia						

2.4.3 Diagramas de caso de uso. Permiten conocer cómo interactúan los diferentes procesos analizados en los requisitos, para de una manera gráfica explicar la funcionalidad.

2.4.3.1 *Caso de uso de verificación de empresa*. Para comprobar que la empresa está creada en la *Cloud Computing* se envía a una URL el RUC ingresado esperando la respuesta como lo ilustra la Figura 14 y se lo detalla en la Tabla 14.

Figura 14. CU01 Verificación de empresa



Fuente: Elaboración propia

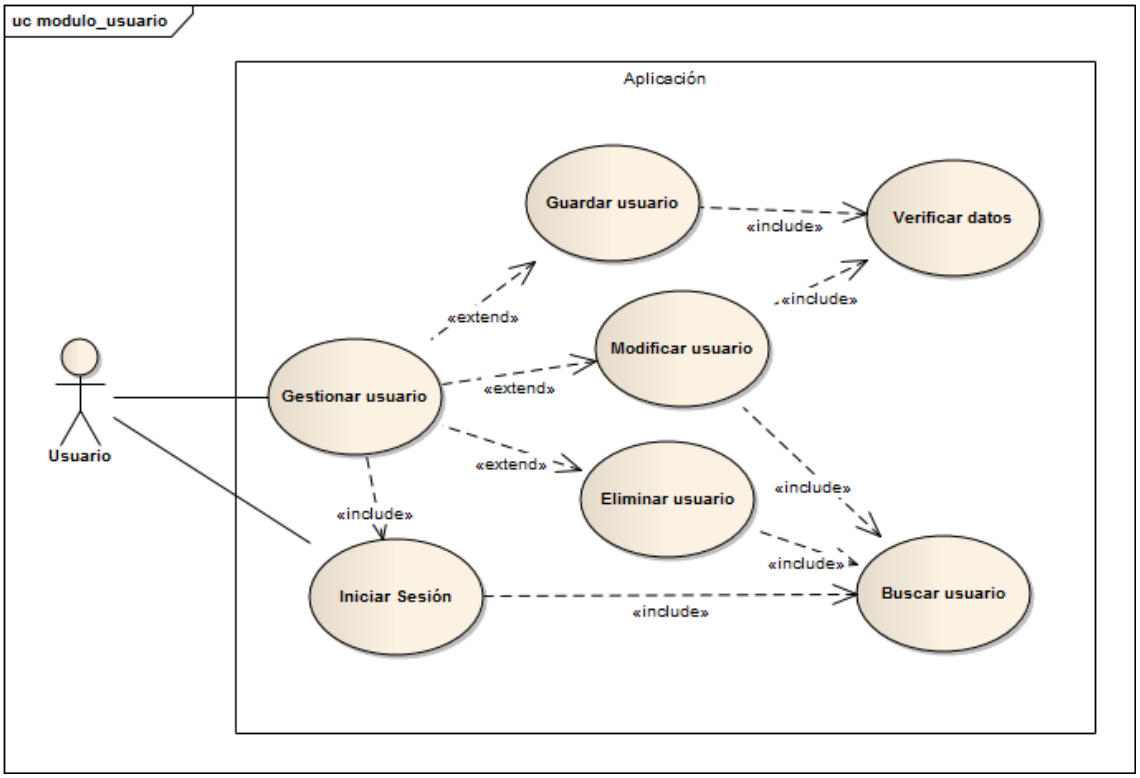
Tabla 14. CU01 Verificación de empresa

Atributo	Descripción
Caso de uso	CU01
Objetivo	Verificar que el Gateway pertenezca a una empresa registrada en la Cloud Computing
Requisito	RF01
Actor	Usuario, Cloud Computing
Pre condición	Iniciar Sesión
Actividades	- El usuario inserta el RUC - El dato ingresado es enviado a la Cloud Computing para su verificación - La Cloud Computing puede responder, vacío si no existe, y devuelve la empresa en caso de existir - Si existe el dato, se extrae el RUC y nombre de empresa y se guarda.
Post condición	Habilitar el Panel de control

Fuente: Elaboración propia

2.4.3.2 *Caso de uso de Gestión de usuarios*. En la Figura 15 se observa la gestión (registrar, buscar, modificar, eliminar) de usuario y la Tabla 15 detalla cada una de las actividades.

Figura 15. CU02 Gestión de usuarios



Fuente: Elaboración propia

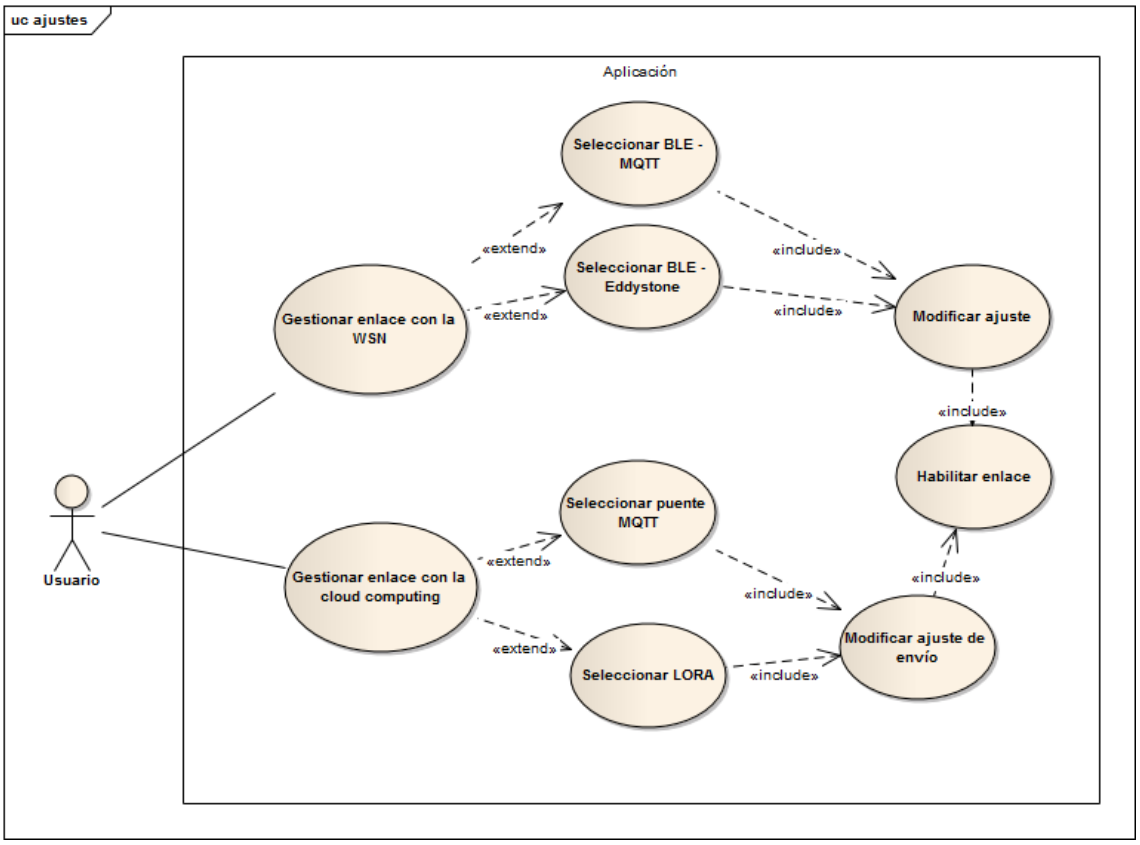
Tabla 15. CU02 Gestión de usuarios

Atributo	Descripción
Caso de uso	CU02
Objetivo	Gestionar el módulo de usuarios, manipulando acciones como: agregar, eliminar, buscar, editar.
Requisito	RF02
Actor	Usuario
Pre condición	Empresa verificada, haber iniciado sesión.
Actividades	- EL inicio de sesión busca en la base de datos las credenciales. - Guardar usuario: ingresa los datos, estos son verificados, si los datos son válidos se procede a guardar. - Modificar usuario: se ingresa el nombre de usuario, el sistema busca y presenta los datos, se modifica y guarda. - Eliminar usuario: el sistema lista todos los usuarios, se selecciona el usuario a borrar, el sistema pregunta si está seguro, una vez que el usuario acepte se elimina.
Post condición	Ninguna.

Fuente: Elaboración propia

2.4.3.3 *Caso de uso de gestión de enlaces*. En la Figura 16 se ilustra el proceso de seleccionar los enlaces activos, esta funcionalidad permite que el usuario pueda controlar por donde desea enviar/recibir información, la Tabla 16 detalla el procedimiento desarrollado.

Figura 16. CU03 gestión de enlaces



Fuente: Elaboración propia

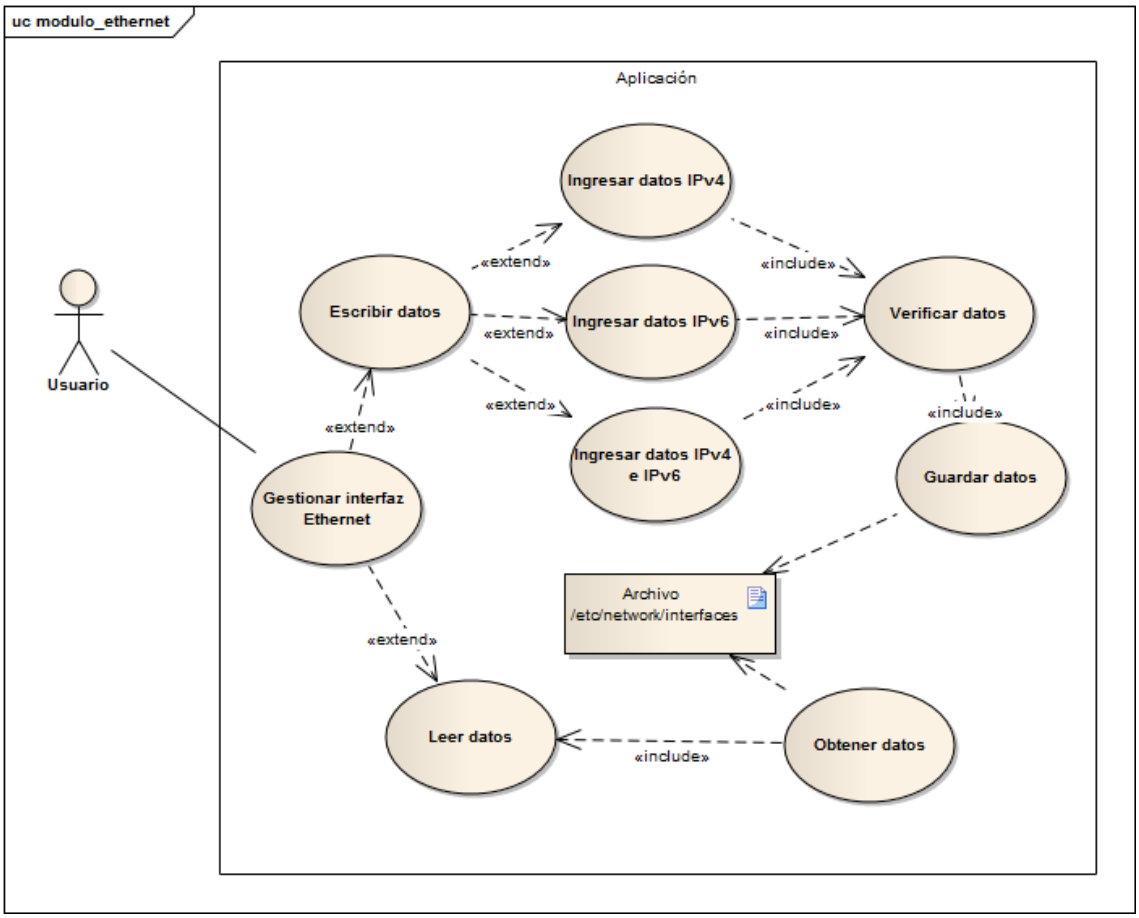
Tabla 16. CU03 gestión de enlaces

Atributo	Descripción
Caso de uso	CU03
Objetivo	Gestionar los módulos de ajustes, donde el usuario podrá seleccionar las interfaces de entrada y salida de datos.
Requisito	RF03, RF04
Actor	Usuario
Pre condición	Iniciado sesión y verificación de empresa (CU01).
Actividades	- Selecciona el enlace este puede ser hacia la WSN o *Cloud Computing*, se puede tener habilitado más de un enlace, se modifica el valor del enlace y se habilita para que comience a funcionar.
Post condición	Enlace Habilitado.

Fuente: Elaboración propia

2.4.3.4 *Caso de uso de interfaz Ethernet*. La Figura 17 se ilustra el proceso para escribir o leer la configuración de red del sistema operativo en la interfaz E*thernet*, la Tabla 17 detalla las actividades desarrolladas.

Figura 17. CU04 interfaz Ethernet



Fuente: Elaboración propia

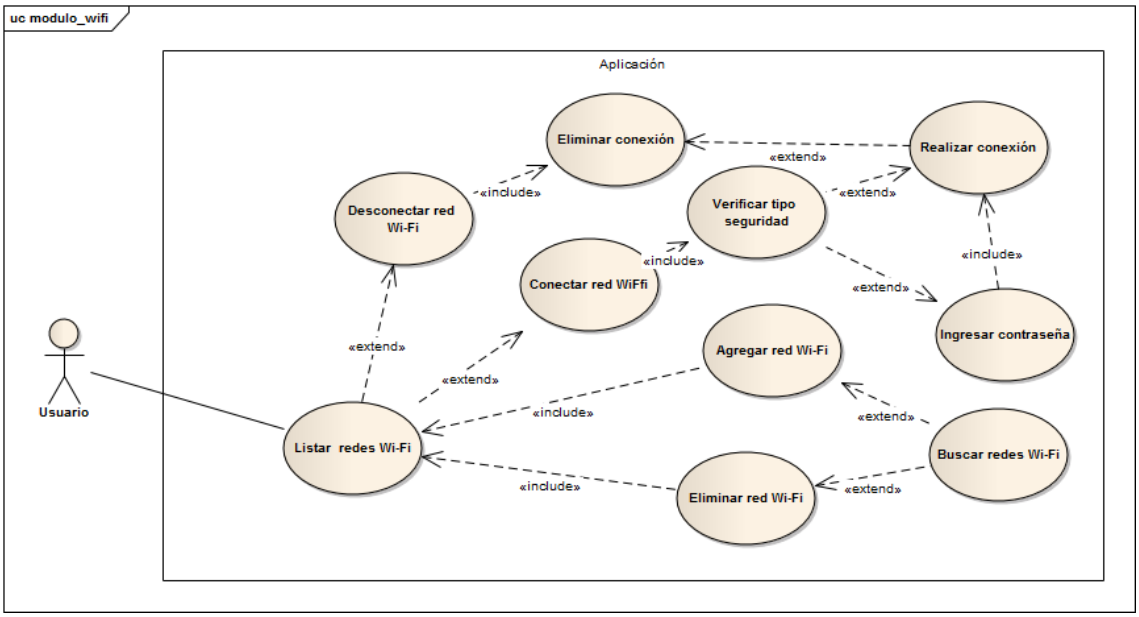
Tabla 17. CU04 interfaz Ethernet

Atributo	Descripción
Caso de uso	CU04
Objetivo	Gestionar la interfaz ethernet, para la escritura o lectura de su configuración con una dirección estática IPv4 o IPv6.
Requisito	RF05
Actor	Usuario
Pre condición	Iniciado sesión.
Actividades	- Escribir: Se puede agregar una dirección IPV4, IPv6 o las dos, estas direcciones son validadas y escritas en el archivo de configuración de red del sistema operativo. - Lectura: Se lee el archivo de red del sistema operativo.
Post condición	Reiniciar el gateway.

Fuente: Elaboración propia

2.4.3.5 *Caso de uso de interfaz Wi-Fi.* La Figura 18 muestra el proceso que debe realizar el Gateway en *background* y el usuario para buscar, listar, conectar y desconectarse de una red Wi-Fi, la Tabla 18 detalla las actividades que se desarrollaron.

Figura 18. CU05 interfaz Wi-Fi



Fuente: Elaboración propia

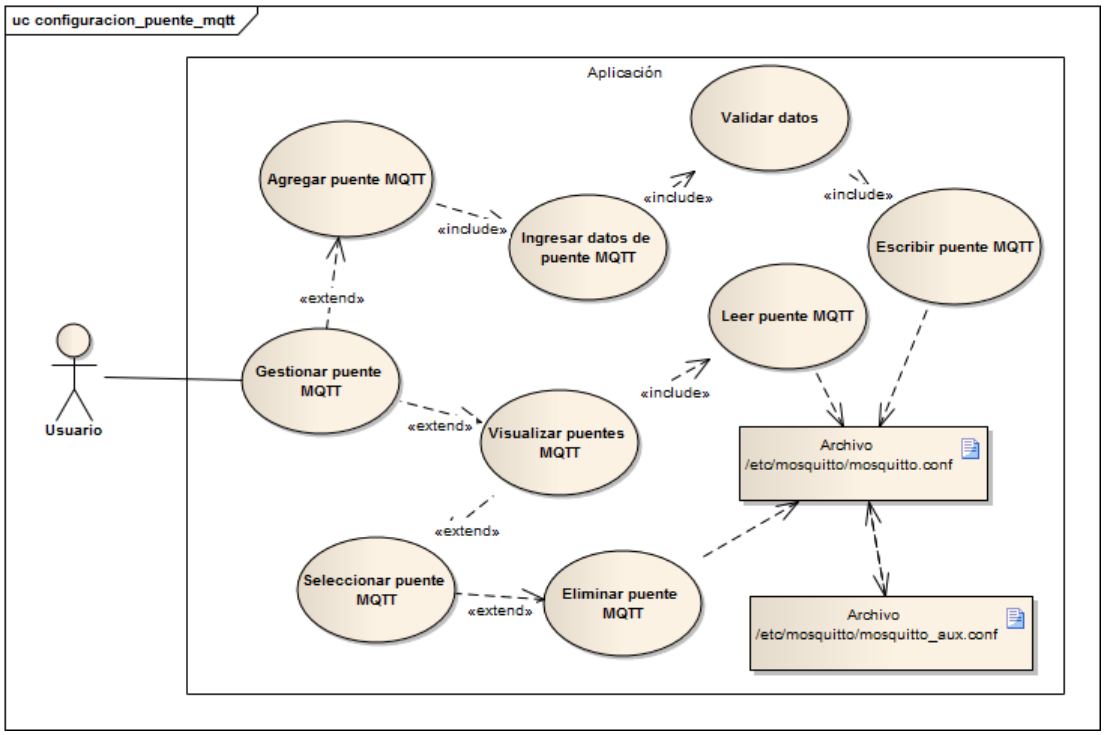
Tabla 18. CU05 interfaz Wi-Fi

Atributo	Descripción
Caso de uso	CU05
Objetivo	Gestionar la interfaz Wi-Fi, para buscar, mostrar, conectarse y desconectarse de una red Wi-Fi.
Requisito	RF06
Actor	Usuario y Aplicación
Pre condición	Iniciado sesión.
Actividades	- El sistema busca las redes Wi-Fi cercanas, estas son agregadas a una lista para que el usuario pueda observarlas. - De la lista de redes cercanas el usuario puede seleccionar una para conectarse - Si la red a conectarse tiene seguridad se pide la contraseña, caso contrario se realiza la conexión pero si existe una conexión a una red, se desconecta y se realiza la conexión a la nueva red seleccionada.
Post condición	Ninguna.

Fuente: Elaboración propia

2.4.3.6 *Caso de uso enlace Puente MQTT*. La Figura 19 ilustra el proceso de escribir y leer el archivo de configuración del puente MQTT, en la Tabla 19 se detalla las actividades que se desarrollaron.

Figura 19. CU06 gestión de enlace puente MQTT



Fuente: Elaboración propia

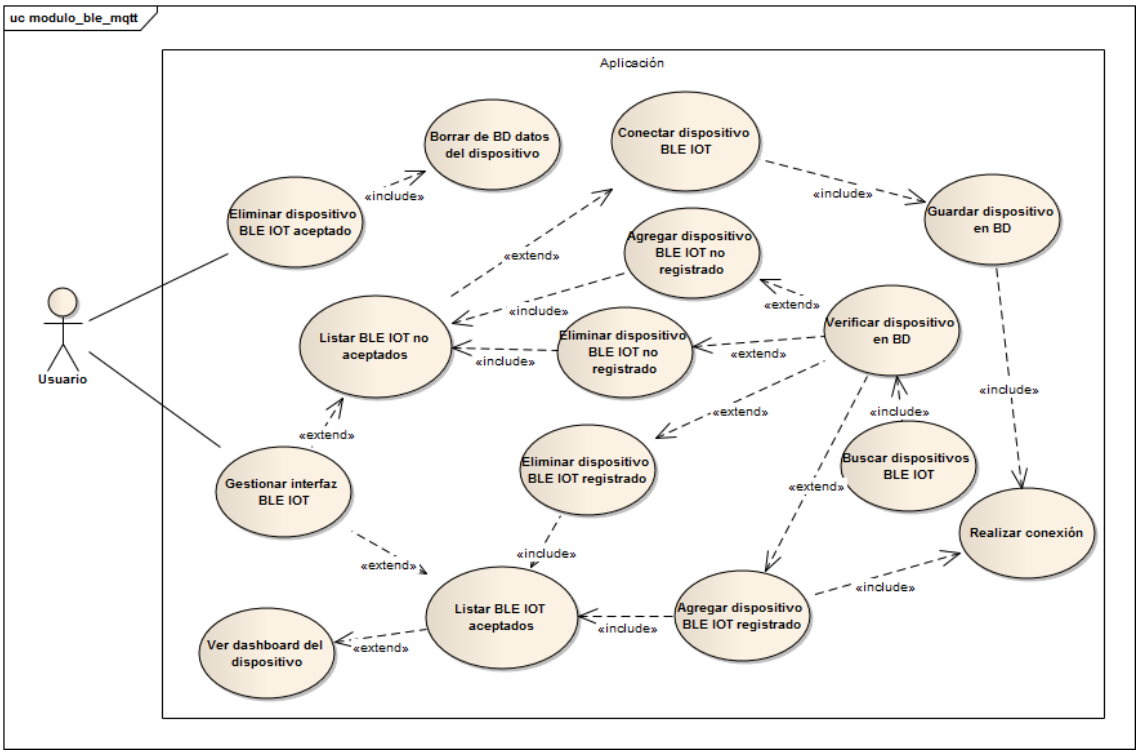
Tabla 19. Gestión de enlace puente MQTT

Atributo	Descripción
Caso de uso	CU06
Objetivo	Gestionar el puente MQTT por donde se envían los datos a la Cloud Computing.
Requisito	RF09
Actor	Usuario
Pre condición	Inicio sesión, enlace habilitado (CU03) y verificar empresa (CU01).
Actividades	- Agregar puente MQTT: Se ingresa los datos, estos son validados para escribirlos en el archivo de puente MQTT. - Para visualizar los puentes MQTT el sistema lista todas las conexiones de puente MQTT. - Para eliminar de la lista de conexiones de puente MQTT se selecciona el puente, el sistema le pregunta si está seguro, si la respuesta es afirmativa se borra la conexión del archivo de configuración.
Post condición	Envió de datos a la Cloud Computing por puente MQTT

Fuente: Elaboración propia

2.4.3.7 *Caso de uso enlace BLE - MQTT*. La Figura 20 muestra el procedimiento para conectar un dispositivo BLE - MQTT con el *Broker* MQTT del Gateway, la Tabla 20 detalla el proceso que realiza el usuario y el sistema en *Background*.

Figura 20. CU07 enlace BLE - MQTT



Fuente: Elaboración propia

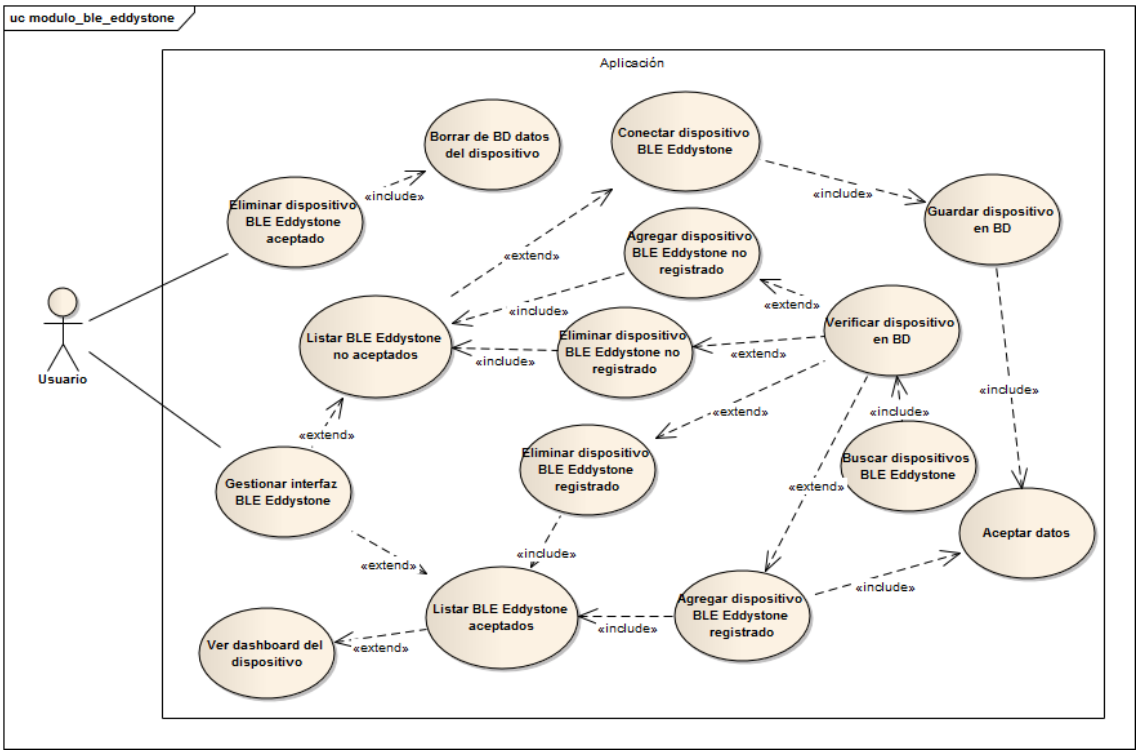
Tabla 20. CU07 enlace BLE - MQTT

Atributo	Descripción
Caso de uso	CU07
Objetivo	Gestionar la interfaz BLE para nodos sensores que usa el protocolo de comunicación MQTT.
Requisito	RF07
Actor	Usuario y aplicación
Pre condición	Inicio sesión, enlace habilitado (CU03) y verificar empresa (CU01).
Actividades	- El sistema busca los dispositivos BLE - MQTT y los agrega a dos listas, una que son para los dispositivos agregados y otra para dispositivos no agregados, son refrescadas eliminando dispositivos fuera del alcance. - De la lista de dispositivos no agregados, el usuario puede seleccionar un dispositivo, el sistema guarda los datos de este dispositivo y realiza la conexión. - Si un dispositivo ya agregado se vuelve a encontrar, automáticamente el sistema realiza la conexión. - Si el usuario desea eliminar un dispositivo el sistema lista todos los dispositivos, el usuario selecciona el que desea.
Post condición	Nodo sensor conectado al Gateway para enviar datos

Fuente: Elaboración propia

2.4.3.8 *Caso de uso enlace BLE – Eddystone*. En la Figura 21 se observa el proceso para aceptar un dispositivo que usa *Eddystone,* la Tabla 23 detalla lo desarrollado.

Figura 21. CU08 interfaz BLE - Eddystone



Fuente: Elaboración propia

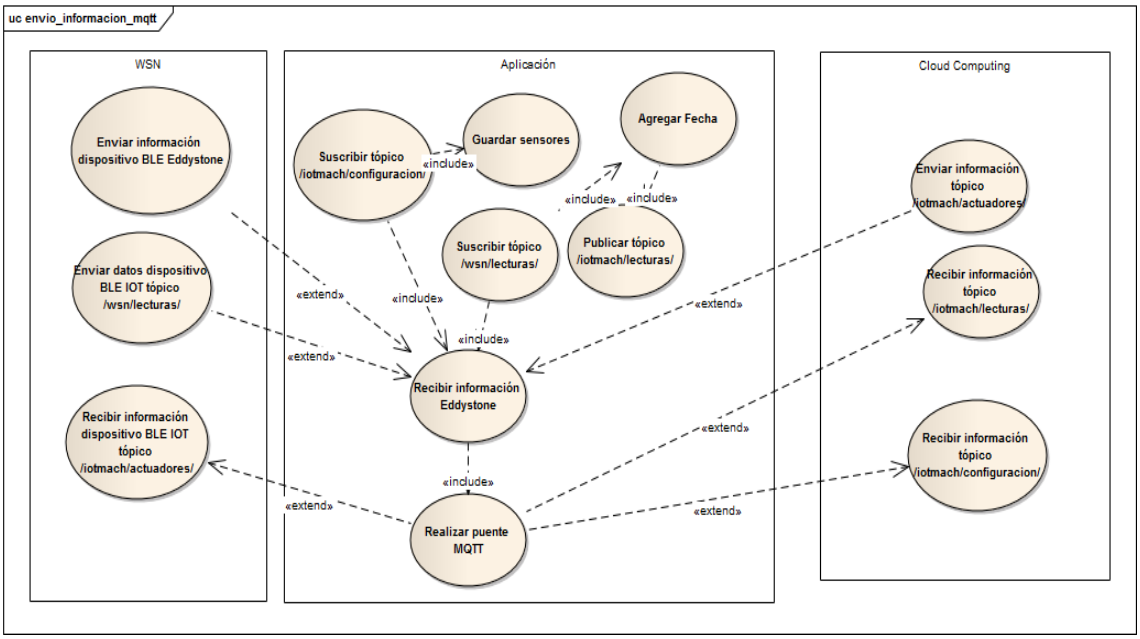
Tabla 21. CU08 interfaz BLE - Eddystone

Atributo	Descripción
Caso de uso	CU08
Objetivo	Gestionar la interfaz BLE para nodos sensores que usa el estándar Eddystone.
Requisito	RF08
Actor	Usuario y aplicación
Pre condición	Inicio sesión, enlace habilitado (CU03) y verificar empresa (CU01).
Actividades	- El sistema busca los dispositivos BLE - Eddystone y los agrega a dos listas, una para los dispositivos agregados y la otra para dispositivos no agregados, cada que un dispositivo desaparezca del alcance se refresca la lista. - Se puede seleccionar un dispositivo no agregado, el sistema guarda los datos y comienza aceptar Beacons. - Si un dispositivo agregado se vuelve a encontrar, automáticamente el sistema acepta los datos. - Si el usuario desea eliminar un dispositivo el sistema lista todos los dispositivos, y selecciona el que desea.
Post condición	Nodo sensor conectado al Gateway para enviar datos

Fuente: Elaboración propia

2.4.3.9 *Caso de uso envió/recepción de datos con MQTT*. La Figura 22 ilustra el proceso que se realiza en *Background* el Gateway, la Tabla 21 detalla las actividades desarrolladas.

Figura 22. CU09 envió de datos con MQTT



Fuente: Elaboración propia

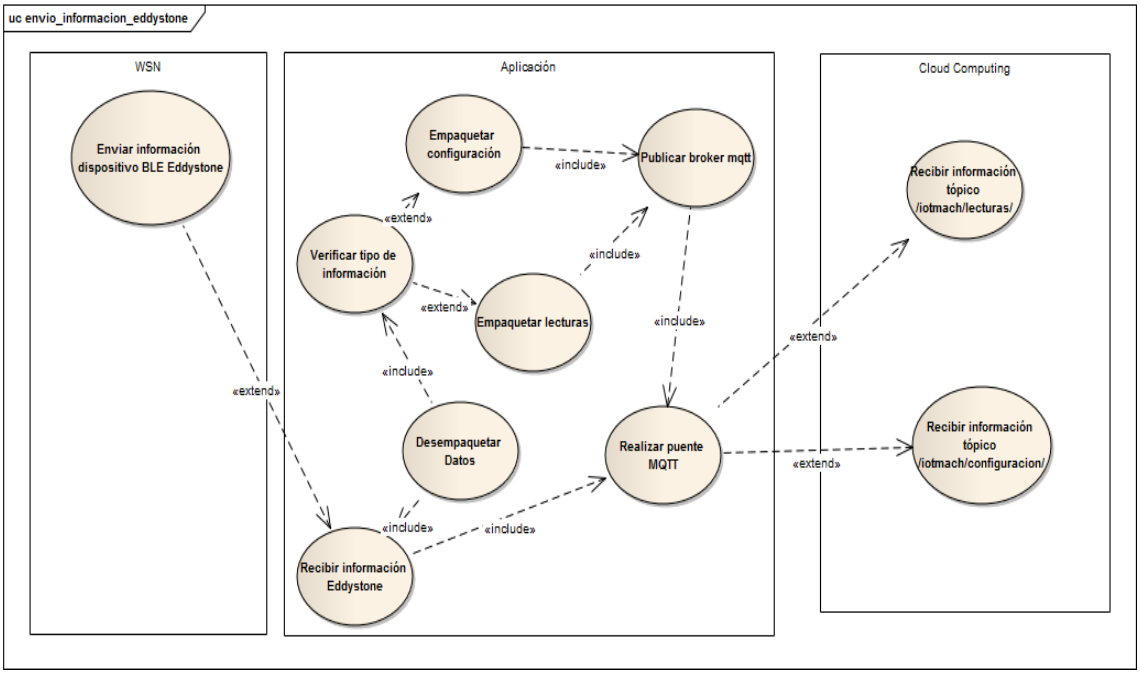
Tabla 22. CU09 envio de datos con MQTT

Atributo	Descripción
Caso de uso	CU09
Objetivo	Gestionar él envió de datos entre los nodos sensores que usan MQTT hacia la cloud computing.
Requisito	RF10, RF12
Actor	WSN, aplicación y cloud computing
Pre condición	Enlace Puente MQTT habilitado (CU03), enlace BLE – MQTT habilitado (CU07) y dispositivo conectado.
Actividades	- Los nodos sensores pueden publicar en diferentes tópicos. - Tópico de configuración: Se usa para recibir la información que detalla cómo trabaja un dispositivo, esta información es guardada en el Gateway y enviada a la cloud computing. - Tópico de lecturas: Este tópico se usa para saber el estado de los actuadores o los datos de sensores, el Gateway cada que reciba datos por este tópico le agrega la fecha y envía a la cloud computing. - Tópico de actuadores: Se usa para enviar desde la cloud computing un comando a los nodos sensores y estos realicen una acción - Se puede tener más tópicos publicando al Gateway pero se necesita configurar un puente para que los datos sean enviados.
Post condición	Ninguna

Fuente: Elaboración propia

2.4.3.10 *Caso de uso de envió de datos con Eddystone*. La Figura 23 ilustra el proceso de envió de información usando *Eddystone* en la WSN y puente MQTT hacia la C*loud Computing*, teniendo que desempaquetar Beacons y empaquetarlos con el protocolo MQTT, la Tabla 23 detalla las actividades desarrolladas.

Figura 23. CU10 envió de datos con Eddystone



Fuente: Elaboración propia

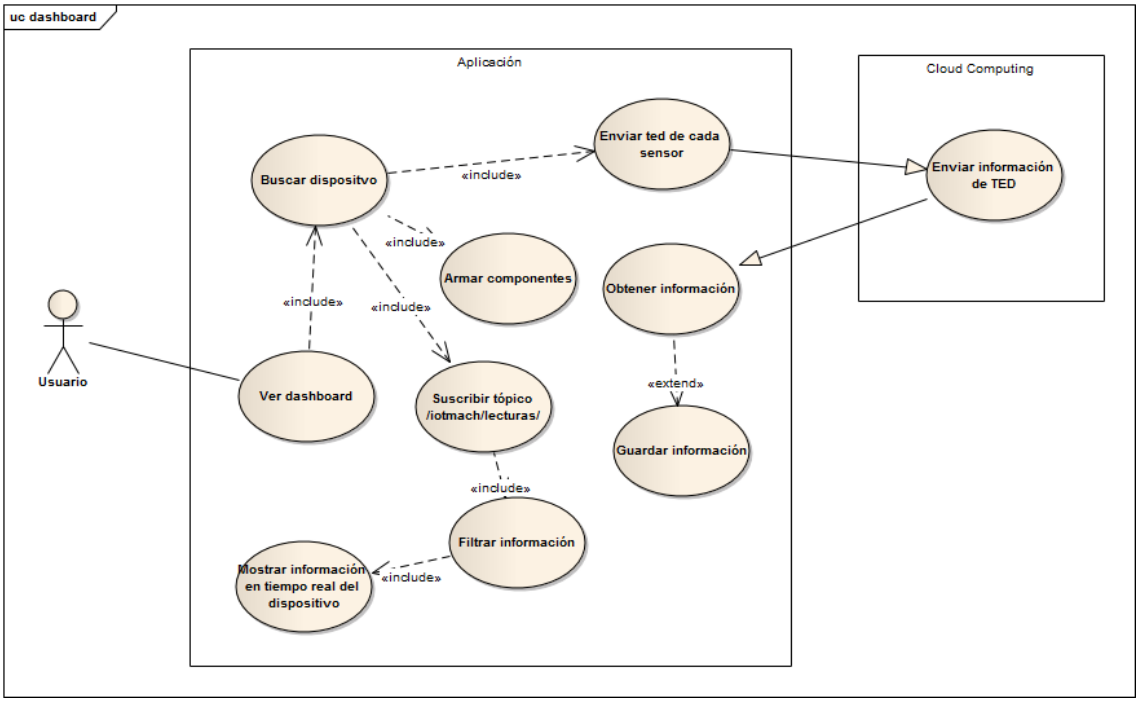
Tabla 23. CU10 envió de datos con Eddystone

Atributo	Descripción
Caso de uso	CU10
Objetivo	Gestionar él envió de datos entre los nodos sensores que usan Eddystone hacia la Cloud Computing.
Requisito	RF10, RF12
Actor	WSN, aplicación y Cloud Computing
Pre condición	Ajustes habilitados
Actividades	- Los nodos sensores envían datos, estos datos son recibidos por el Gateway. - Los datos recibidos se desempaquetan y se realiza un algoritmo para saber por qué tópico del puente MQTT enviarlos. - Una vez enviados por MQTT se usa el estudio de CU09.
Post condición	CU09

Fuente: Elaboración propia

2.4.3.11 *Caso de uso de Dashboard*. La Figura 24 ilustra el proceso de auto-configuración del D*ashboard* basada en la información del nodo sensor, los datos se reciben/envían en tiempo real usando el protocolo MQTT sin importar si el nodo sensor es BLE MQTT o *Eddystone,* la Tabla 24 detalla las actividades desarrolladas.

Figura 24. CU11 Dashboard



Fuente: Elaboración propia

Tabla 24. CU11 Dashboard

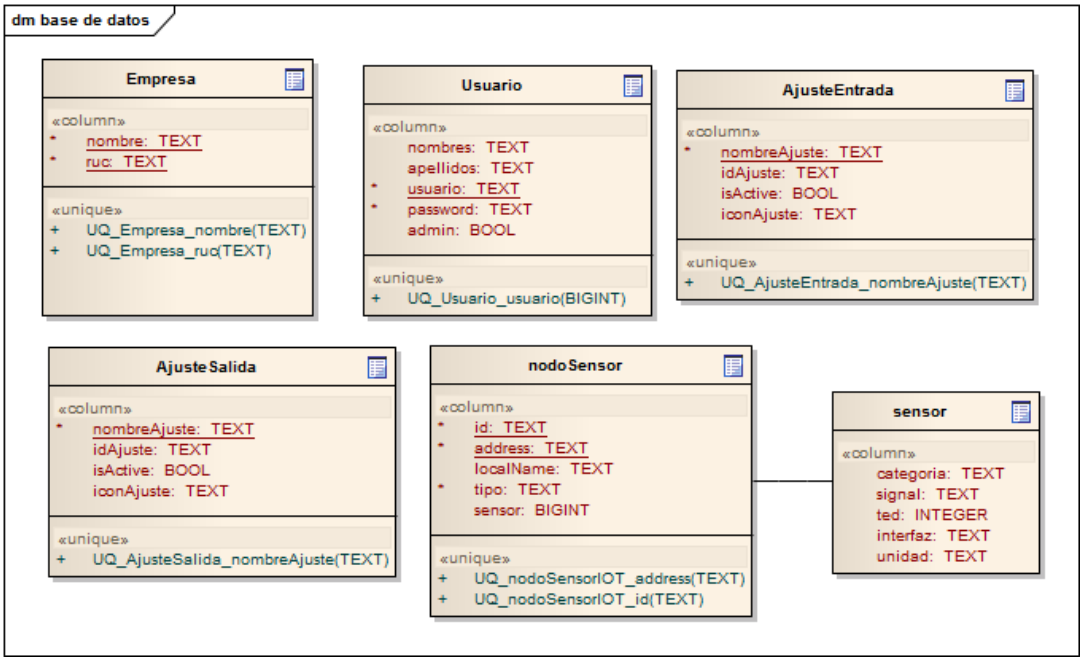
Atributo	Descripción
Caso de uso	CU11
Objetivo	Gestionar la creación y muestra de datos del Dashboard.
Requisito	RF10, RF12, RF13
Actor	Usuario, aplicación y Cloud Computing
Pre condición	Inicio Sesión, enlaces habilitados (CU03) Dispositivos BLE MQTT (CU07) o Eddystone (CU08) aceptados
Actividades	- El usuario selecciona el dispositivo, el sistema busca los datos de este nodo sensor y arma los componentes a usarse, se suscribe al tópico de lecturas y filtra la información para mostrar los datos en tiempo real. - El sistema si tiene conexión en ese momento con la Cloud Computing, busca la información de cada sensor del nodo sensor a través de su ted y actualiza los datos de los dispositivos.
Post condición	Ninguna.

Fuente: Elaboración propia

2.4.4 Diagrama de base de datos. La Figura 25 muestra el modelo de la base de datos no relacional, esto quiere decir que cada dato viene a representar un documento no estructurado. El campo sensor de la tabla nodoSensor es un array de datos, el cual los campos se encuentran representados en la tabla sensor, se separaron las tablas para la demostración de los campos del array.

La tabla empresa recolecta la respuesta de la verificación de empresa que se hace con la cloud computing en el CU01. La tabla Usuario almacena cada uno de los usuarios que pueden acceder al sistema haciendo referencia al CU02. Las tablas AjusteEntrada y AjusteSalida ayudan almacenar la información de los enlace de datos mencionados en el CU03. La tabla nodoSensor almacena cada uno de los dispositivos aceptados y su proceso se refleja en el CU07 y CU08, el array reflejado en la tabla sensor se puntualiza en el CU09, CU10 y CU11.

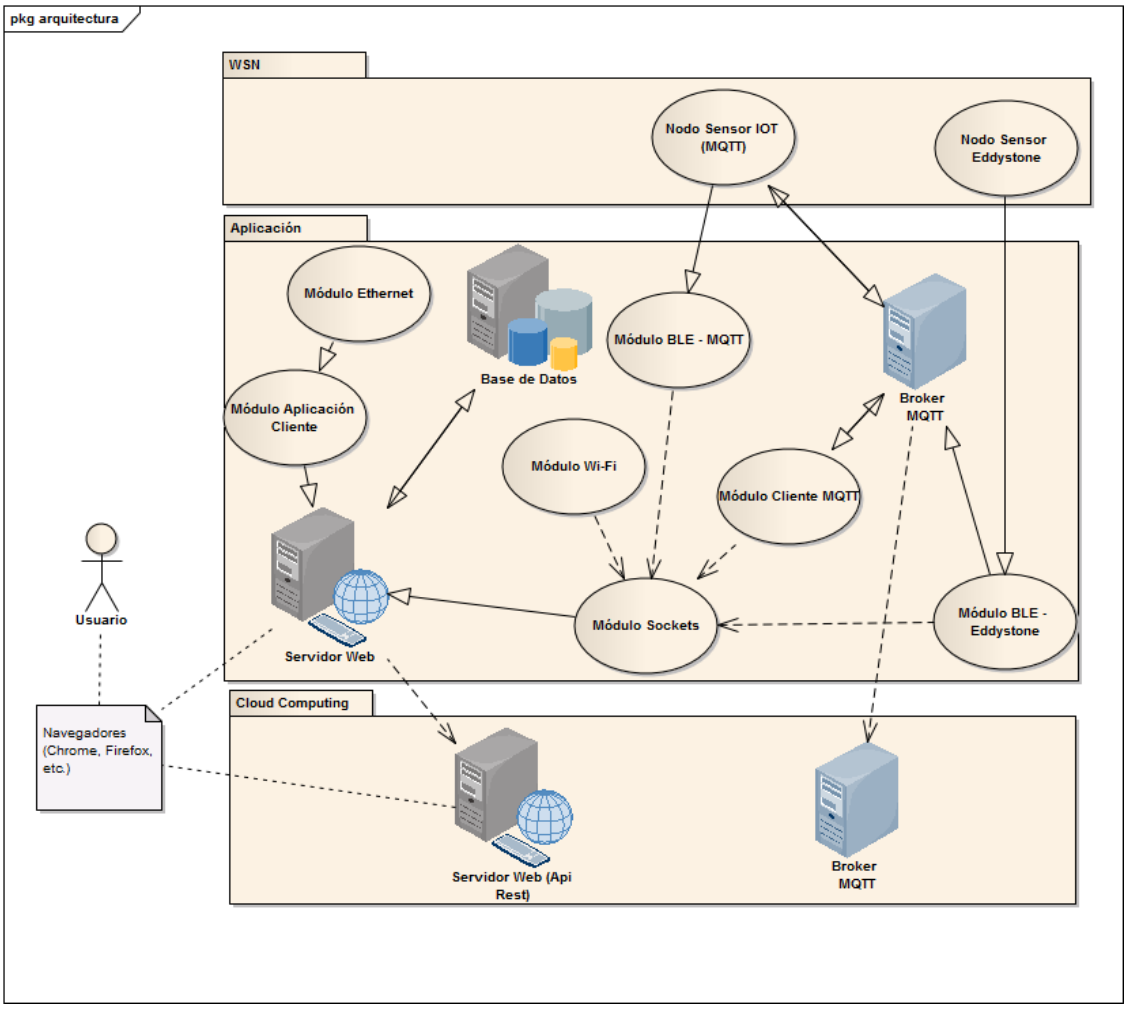
Figura 25. Base de datos



Fuente: Elaboración propia

2.4.5 Diagrama de Arquitectura. En la Figura 26 se observa como está compuesta la arquitectura del Gateway, así mismo se puede observar cómo interactúan La WSN o WSAN y C*loud Computing* con los servicios y módulos desarrollados en el Gateway. El usuario puede administrar el Gateway desde un navegador ingresando al servidor Web que es el encargado de servir cada una de las funcionalidades.

Figura 26. Diagrama de Arquitectura



Fuente: Elaboración propia

2.5 Ejecución y/o ensamblaje del prototipo

Para el desarrollo del prototipo se realizó diferentes tareas, iniciando por el análisis de cada uno de los requisitos, instalación y configuración de: sistema operativo, servicios, Lenguaje de programación y Base de Datos a utilizar.

2.5.1 Preparación del entorno. El Gateway como todo sistema embebido tiene componentes en hardware y software. Para el hardware se seleccionó una Raspberry pi 3 modelo B, ya que tiene incorporado un módulo *Bluetooth* 4.1 con la cual se puede activar BLE, una tarjeta de red con interfaces Wi-Fi y Ethernet, además de contar con una capacidad de procesamiento adecuada para lo que se requiere.

Para el software analizando los requisitos se decidió instalar en una microSD de 64GB el sistema operativo Raspbian, y en base a las herramientas tecnológicas utilizadas en

la referencia [43], que habla sobre el desarrollo de un sistema de IOT con herramientas Open-Source, se utilizó lo siguiente:

- Manejador de Bluetooth: Bluez.
- Broker MQTT: Mosquitto.
- Lenguaje de programación: JavaScript.
- Base de Datos: MongoDB

Para la instalación de Raspbian en la Raspberry Pi se siguió los pasos detallados en la referencia [44], y con dicha instalación se dispuso de blues, que viene integrado en las versiones de raspbian superior o igual a 3.18.11-v7+.

Para usar el protocolo MQTT se instaló un Broker de Mosquitto, siguiendo los pasos del Anexo C, el cual está configurado para usar WebSockets (ver Anexo D). En el Anexo E se encuentra detallado la configuración de un puente hacia el Broker MQTT de la Cloud Computing.

JavaScript es el lenguaje de programación principal utilizado en el sitio web, para ello se instalaron ciertas librerías y frameworks que son:

- Node JS. Para usar JavaScript en el lado del servidor (Back-End), la instalación esta explicada en el Anexo A.
- Npm. Permite gestionar paquetes (librerías) para Node JS y viene incorporado en la instalación de Node JS
- ExpressJS. Como servidor del contenido dinámico de Node JS, la instalación se la realiza de forma global usando el siguiente comando: npm install express –g.
- Angular JS y Jquery. Para la programación del lado del cliente (Front-End), se lo descargó usando los siguientes comandos bower: bower install angular y bower install Jquery.

Para el almacenamiento de datos en el Gateway, se utilizó MongoDB por ser una base de datos que comprime la información, optimizando el espacio de la tarjeta microSD de la Raspberry PI, la instalación y configuración de Mongodb se encuentra detallada en el Anexo B. A continuación se detalla la programación del lado del servidor de los principales módulos del Gateway:

- En la Tabla 25 se encuentra la función que realiza el Gateway para enviar el RUC ingresado por el usuario hacia la Cloud Computing y esperar el mensaje de confirmación, que pueden ser tres: no existe conexión con el servidor, el ruc de la empresa no existe (retorna una lista con los datos vacíos) y empresa registrada (retorna en una lista el RUC y nombre de la empresa).

Tabla 25. Funcionamiento verificación de empresa

Atributo	Descripción
Modulo:	Verificación de Empresa
Caso de Uso:	CU01
URL	/empresa/registro/

Código:	router.post('/registro', function (req, res) {
	request('http://192.168.0.10:8000/empresa/obtener/datos/' + req.body.ruc, function (error, response, body) {
	if (!error && response.statusCode == 200) {
	mensajeJson = JSON.parse(body);
	empresa_ruc = mensajeJson.emp_ruc;
	empresa_nombre = mensajeJson.emp_nombre;
	if (empresa_ruc != "" && empresa_nombre != "") {
	var registrar_empresa=new Empresa({ruc: empresa_ruc, nombre: empresa_nombre});
	registrar_empresa.save(function (error, empres) {
	if (!error) {
	res.redirect('/');
	} else {
	req.flash('mensaje', 'Lastimosamente ocurrio un error');
	res.redirect('/');
	}
	});
	}else {
	req.flash('mensaje', 'La empresa no está registrada');
	res.redirect('/');
	}
	}else{
	req.flash('mensaje', 'Compruebe su conexión a internet');
	res.redirect('/');
	}
	});
	});

Lenguaje de Programación	JavaScript (NodeJS)
Librería:	request

Fuente: Elaboración propia

- En la Tabla 26 se muestra el Inicio de Sesión del sitio WEB, especificado en el CU02, se lo realiza usando la librería Passport y Passport-local. Se guarda las credenciales en las sesiones y cookies del navegador.

Tabla 26. Funcionamiento del Inicio de Sesión

Atributo	Descripción
Modulo:	Inicio de Sesión
Caso de Uso:	CU02
URL	/usuario/login/

Código:	passport.use(new LocalStrategy(User.authenticate()));
	passport.serializeUser(User.serializeUser());
	passport.deserializeUser(User.deserializeUser());
	router.post('/login', passport.authenticate('local',
	{ successRedirect: '/',
	failureFlash: 'Usuario o contraseña incorrectos, vuelva a intentarlo.',
	failureRedirect: '/'
	})
	);

Lenguaje de Programación	JavaScript (NodeJS)
Librería:	passport, passport-local
Fuente: Elaboración propia	

- Para el CU03 se usan dos funciones: La primera verifica si se han creado los enlaces en la base de datos (de no existir generarlos), la otra como muestra la Tabla 27 se encarga de cambiar los estados de los enlaces cuando el usuario realice cierta acción.

Tabla 27. Funcionamiento de modificar Enlace

Atributo	Descripción
Modulo:	Modificar enlace de datos
Caso de Uso:	CU03
URL	/ajustes/

Código:	Ajuste.findOneAndUpdate({ nombreAjuste: req.body.nombreAjuste }, { isActive: req.body.seleccionado }, function(err, ajuste) {
	if (err) throw err;
	res.json({
	data: 'Guardado correctamente'
	});
	});

Lenguaje de Programación	JavaScript (NodeJS)
Librería:	Ninguna
Fuente: Elaboración propia	

- Para controlar la interfaz Ethernet desde la página Web del Gateway se utilizó la librería node-network-interfaces, que permite leer y escribir el archivo /etc/network/interfaces del sistema operativo, la Tabla 28 muestra la lectura que se hace mediante una petición GET.

Tabla 28. Funcionamiento para la lectura de interfaz Ethernet

Atributo	Descripción
Modulo:	Lectura interfaz Ethernet
Caso de Uso:	CU04
URL	/ethernet/

Código:	interfaces.currentConfig('eth0','ipv4').then((interfaceConfig) => {
	interfaz['ipv4']=interfaceConfig;
	interfaces.currentConfig('eth0','ipv6').then((ipv6) => {
	interfaz['ipv6']=ipv6;
	res.json(interfaz);
	}).catch((readError) => {
	res.json({
	error: readError
	});
	});
	}).catch((readError) => {
	res.json({
	error: readError
	});
	});

Lenguaje de Programación	JavaScript (NodeJS)
Librería:	node-network-interfaces
Fuente: Elaboración propia	

- Para la escritura de la interfaz ethernet especificada en el CU04, se utilizó la librería node-network-interfaces, el evento se dispara cuando se realiza una petición post y se envian los datos requeridos. El CU04 indica tres casos que se deben considerar que son: cuando la configuración es IPv4, IPv6 o ambas, pero en la Tabla 29 se indica la funcionalidad de escritura cuando es IPv4 e IPv6, no se consideran las condiciones en esta explicación, aunque estén agregadas en el sitio WEB

Tabla 29. Funcionamiento para la escritura de la interfaz ethernet

Atributo	Descripción
Modulo:	Escritura interfaz Ethernet
Caso de Uso:	CU04
URL	/ethernet/

Código:	interfaces.setConfig('eth0', {
	address: req.body.ipv4,
	netmask: req.body.mascara4,
	gateway: req.body.gateway4
	}).then((newConfig) => {
	interfaces.setConfig('eth0', {
	address: req.body.ipv6,
	netmask: req.body.mascara6,
	gateway: req.body.gateway6
	}).then((newConfig) => {
	res.json({
	data: 'Guardado Correctamente',
	ethernet: newConfig
	});
	}).catch((configUpdateError) => {
	res.json({
	error: configUpdateError
	});
	})

Lenguaje de Programación	JavaScript (NodeJS)
Librería:	node-network-interfaces

Fuente: Elaboración propia

- Para el CU05 se utilizó la librería Wireless, que permite controlar la interfaz Wi-Fi del Gateway, este módulo se comunica con la página principal a través de scokets, cada que una red Wi-Fi es encontrada se publica en el socket un diccionario indicando el ssid (nombre de la red), la dirección MAC del dispositivo, distancia, el tipo de seguridad que usa la red. La página Web escucha el socket y muestra al usuario las redes disponibles, de las cuales el usuario puede seleccionar una, la Tabla 30 contiene la función que realiza el Gateway para poder conectarse a una red sin importar la seguridad.

Tabla 30. Funcionamiento conexión red Wi-Fi

Atributo	Descripción
Modulo:	Conexión a una red con la interfaz Wi-Fi
Caso de Uso:	CU05
URL	Socket(/wifi)

Código:	conn.on('conexion_wifi', function (data) {
	if (network.ssid === data.red.ssid) {
	if(connect==true){
	desactivar_wifi(data.red);
	}
	wireless.join(network, data.clave, function(err) {
	if (err) {
	mensaje={
	estado: "x",
	mensaje: "Ocurrio un problema a conectarse a la red "+data.red.ssid
	}
	conn.emit('mensaje_wifi',mensaje);
	return;
	}
	if(data.red.encryption_type=='WPA&WPA2' || data.red.encryption_type=='WPA'||data.red.encryption_type=='WPA2'){
	bandera=true;
	}
	red_actual=data.red.ssid;
	connect=true;
	conn.emit('actualizar_wifi_conectar',data.red);
	mensaje={
	estado: "ok",
	mensaje: "Conectado a la red "+ data.red.ssid
	}
	conn.emit('mensaje_wifi',mensaje);
	wireless.dhcp(function(ip_address) {
	});
	});
	}
	});

Lenguaje de Programación	JavaScript (NodeJS)
Librería:	wireless

Fuente: Elaboración propia

- Para programar el CU06 se realizó un algoritmo para la lectura y escritura del archivo /etc/mosquitto/mosquitto.conf, que permite realizar puentes MQTT hacia otro bróker, la Tabla 31 muestra la escritura utilizando la librería fs, el proceso se realiza mediante una petición post.

Tabla 31. Funcionamiento para la escritura de puentes MQTT

Atributo	Descripción
Modulo:	Conexión a una red con la interfaz Wi-Fi
Caso de Uso:	CU06
URL	/mqtt/registrar/

Código:	texto += 'connection '+req.body.brige_nombre+' \n';
	texto += 'address '+req.body.mqtt_ip+':'+req.body.mqtt_puerto+' \n'
	texto += 'cleansession true \n';
	texto += 'try_private true \n';
	texto += 'notifications false \n';
	texto += 'start_type automatic \n';
	if(String(req.body.mqtt_username).trim() !== ''){
	texto += 'username '+req.body.mqtt_username+' \n';
	texto += 'password '+req.body.mqtt_password+' \n';
	}
	var contador = 0;
	for(var atributo in req.body){
	if(String(atributo).search('mqtt_direccion'+contador) >= 0){
	texto += 'topic # '+req.body['mqtt_direccion'+contador]+' 0 '+req.body['mqtt_topic_in'+contador]+' '+req.body['mqtt_topic_out'+contador]+' \n';
	contador++; }
	}
	var configuraciones = leerConfiguracion();configuraciones += texto;
	escribirConfiguracion(configuraciones)
	res.render('mqtt/registrar', {
	user: req.user,
	mensaje : 'Guardado correctamente, por favor reinicie el gateway',
	tag: 'success'
	});

Lenguaje de Programación	JavaScript (NodeJS)
Librería:	fs
Fuente: Elaboración propia	

- Para el CU07 se utilizó la librería noble, que permite controlar el modulo Bluetooth del Gateway, encargándose de buscar los dispositivos *Bluetooth* cercanos, los cuales son enviados hacia el sitio WEB mediante sockets, para que el usuario seleccione con que dispositivos el Gateway tiene que intercambiar información. La conexión se la realiza bajo 6LoWPAN, para activar la dirección IPv6 de BLE y permitir la conexión con el Broker Mosquitto, para ello la Tabla 32 muestra el proceso.

Tabla 32. Funcionamiento para la conexión de BLE-MQTT

Atributo	Descripción
Modulo:	Conexión a un nodo sensor BLE - MQTT
Caso de Uso:	CU07
URL	Scoket(/iot)

Código:	BluetoothIoT.findOne({ id: (dispositivoBIOT.id) }, function (err, dispositivo) {
	if (err) { throw err; }
	if(dispositivo == null){
	conn.emit('llenaLista', dispositivoBIOT);
	}else{
	exec('sudo echo "connect '+dispositivo.address+' 1" > /sys/kernel/debug/bluetooth/6lowpan_control');
	conn.emit('vaciaLista', '');
	}
	});

Lenguaje de Programación	JavaScript (NodeJS)
Librería:	noble, child_process
Fuente: Elaboración propia	

- Para programar el CU08 se utilizó la librería eddystone-beacon-scanner, que permite buscar dispositivos con Eddystone, estos al ser encontrados son enviados al servidor de sockets en un diccionario que contiene: id, MAC, nombre, tipo, para mostrarlos al usuario y que pueda seleccionar con quienes el Gateway debe comunicarse. Cada que un dispositivo esta fuera del alcance la lista es refrescada, la Tabla 33 muestra el proceso para la búsqueda de nodos sensores con Eddystone que realiza el Gateway.

Tabla 33. Funcionamiento para la búsqueda de dispositivos BLE-Eddystone

Atributo	Descripción
Modulo:	Conexión a un nodo sensor BLE - Eddystone
Caso de Uso:	CU08
URL	Socket(/eddystone)

Código:	this.eddystone.on('found', function(beacon) {
	var direccion = String(beacon.id);var address = direccion.substring(0,2)+':'+direccion.substring(2,4)+':'+direccion.substring(4,6)+':'+direccion.substring(6,8)+':'+direccion.substring(8,10)+':'+direccion.substring(10,12);
	dispositivoEddystone={
	id: beacon.id, address: address, localName: '', tipo: 'EDDYSTONE'
	};});

Lenguaje de Programación	JavaScript (NodeJS)
Librería:	eddystone-beacon-scanner
Fuente: Elaboración propia	

- Para el envío de datos por MQTT que está especificado en el CU09, se usa la librería mqtt para crear un cliente del bróker mosquitto, con la que el Gateway se suscribe al tópico /iotmach/configuración/ (guardar datos de los nodos sensores) y /WSN/lecturas/ (para completar los datos y enviarlos a la *cloud computing*), como lo muestra la Tabla 34.

Tabla 34. Funcionamiento para él envió de datos BLE - MQTT

Atributo	Descripción
Modulo:	Recepción/envió de datos BLE-MQTT
Caso de Uso:	CU09
URL	Ninguna

Código:	clienteMosquitto.on('message', function (topic, message) {
	if(topic == '/iotmach/configuracion/'){
	var mensajeJson = JSON.parse(message);
	Bluetooth.findOneAndUpdate({address: String(mensajeJson.mac)}, {$set:{datos: mensajeJson.datos }}, {new: true}, function(error, doc){
	if (error) { throw error; }});}
	if(topic == '/WSN/lecturas/'){mensajeJson["fecha"] = new Date();
	publish('/iotmach/lecturas/', JSON.stringify(mensajeJson));}});

Lenguaje de Programación	JavaScript (NodeJS)
Librería:	mqtt
Fuente: Elaboración propia	

- Para desempaquetar los Beacons de Eddystone y empaquetarlos en MQTT como lo indica el CU10, el Gateway realiza el proceso especificado en la Tabla 35, donde publica al bróker Mosquitto los Beacons recibidos por Eddystone.

Tabla 35. Funcionamiento recepción/envió de datos BLE - Eddystone

Atributo	Descripción
Modulo:	Recepción/envió de datos BLE-Eddystone
Caso de Uso:	CU10
URL	Ninguna

Código:	this.eddystone.on('updated', function(beacon) {
	BluetoothEddystone.findOne({ id: (beacon.id) }, function (err, dispositivo) {
	if (err) { throw err; }
	if(dispositivo != null){
	clienteMosquitto.publish('/iotmach/sensor/', beacon.id); }
	});});

Lenguaje de Programación	JavaScript (NodeJS)
Librería:	mqtt, eddystone-beacon-scanner
Fuente: Elaboración propia	

- Para el Dashboard mencionado en el CU11, se busca en la base de datos y se carga el contenido del nodo sensor solicitado como lo muestra la Tabla 36, para las acciones en tiempo real se utiliza un cliente MQTT por WebSockets.

Tabla 36. Funcionamiento del Dashboard

Atributo	Descripción
Modulo:	Recepción/envió de datos BLE-Eddystone
Caso de Uso:	CU10
URL	Ninguna

Código:	var ip = (String(req.headers.host).split(':'))[0];
	Bluetooth.findOne({ address: (req.params.mac) }, function (err, dispositivo) {
	if(err){ throw err} res.render('dashboard', {
	user: req.user, dispositivo: dispositivo, broker: ip,
	topicSuscribe: '/iotmach/lecturas/', topicPublic: '/iotmach/actuadores/'
	});});

Lenguaje de Programación	JavaScript (NodeJS)
Librería:	mqtt, eddystone-beacon-scanner
Fuente: Elaboración propia	

2.5.2 Funcionamiento del prototipo. En esta sección se describe el funcionamiento del sitio WEB del Gateway, el usuario podrá realizar las siguientes acciones descritas a continuación:

- El usuario al encender el Gateway por primera vez necesita realizar ciertas configuraciones, para lo cual debe abrir el navegador e ingresar a la página WEB del Gateway para iniciar sesión, el usuario: admin y contraseña: admin, están configuradas por defecto como lo indica la Figura 27.

Figura 27. Pantalla de Inicio de Sesión

Fuente: Elaboración propia

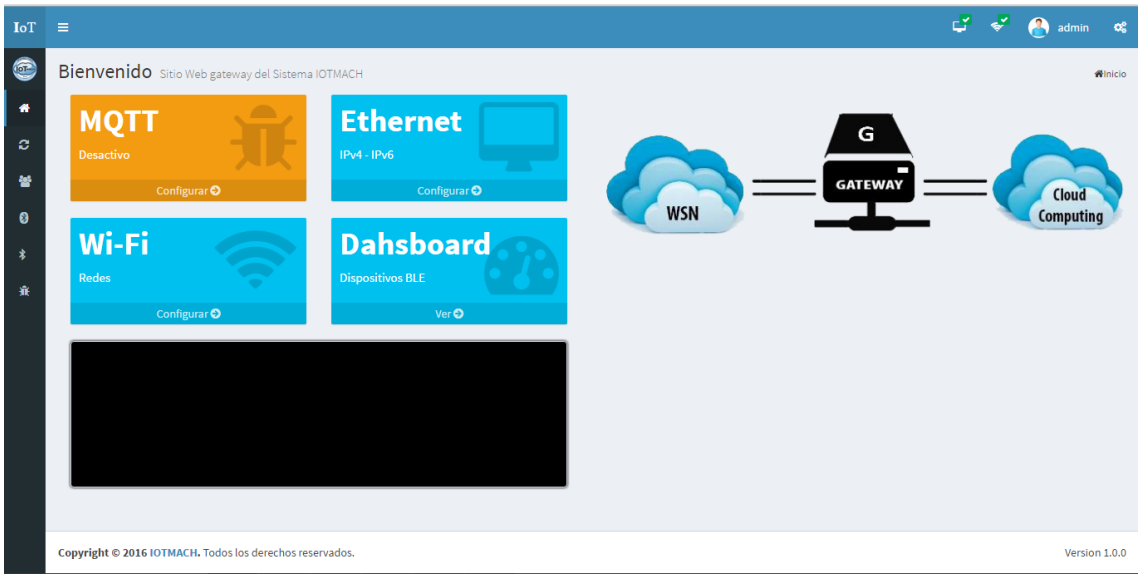
- Si aún no se ha realizado la verificación de la empresa descrita en el CU01, como muestra la Figura 28 el usuario puede: 1. Ingresar el RUC de la empresa registrada en la *Cloud Computing* del sistema IOTMACH, 2. Conectarse a una Red Wi-Fi o cambiar la configuración de la interfaz Ethernet. El Gateway necesita tener conexión hacia la *Cloud Computing*, este proceso se lo realiza solo una vez, por lo que se recomienda realizarlo antes de ponerlo en el sitio de producción. Si el RUC ingresado no está registrado en la Cloud Computing, el usuario no podrá ver el sitio de administración del Gateway-IOTMACH, se recomienda primero crear una cuenta en el sitio WEB de internet del sistema IOTMACH.

Figura 28. Pantalla verificación de empresa

Fuente: Elaboración propia

- Una vez iniciado sesión y verificado la empresa, se muestra el panel de control del sitio WEB. En la Figura 29 se observa varios indicadores los más importantes son: Si el bróker de Mosquitto del Gateway está activo o desactivo, la consola en tiempo real del bróker Mosquitto de esta manera se sabe que clientes MQTT están conectados.

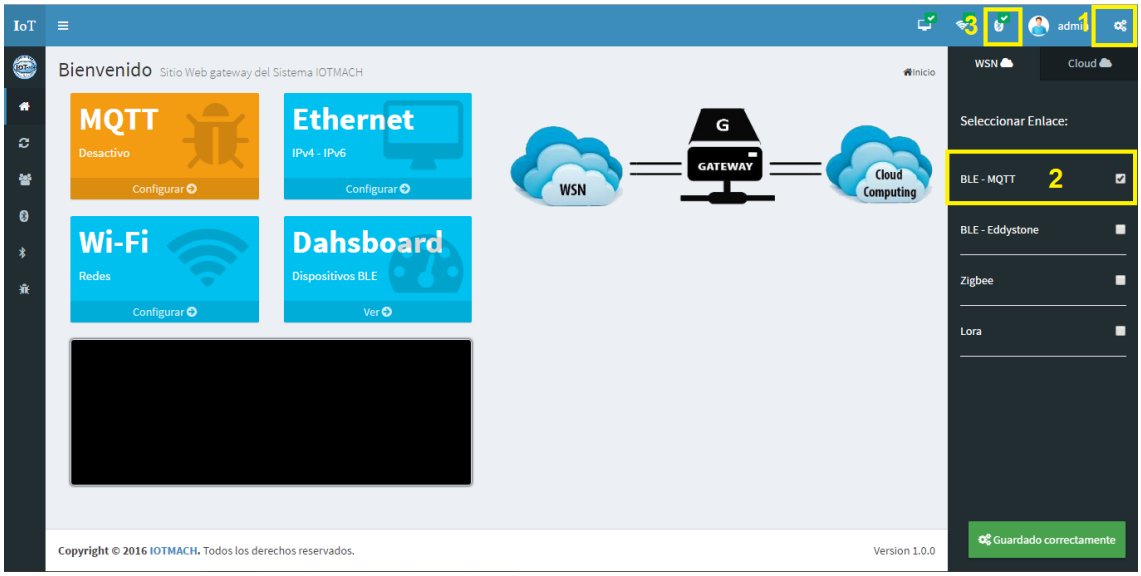
Figura 29. Pantalla del panel de control



Fuente: Elaboración propia

- La Figura 30 muestra lo que debe realizar un usuario para seleccionar un enlace hacia la WSN en donde: 1. Selecciona abrir los enlaces, 2. Dar clic en el enlace, 3. Se activa el icono y servicio. Para programar esta función se consideró el CU03.

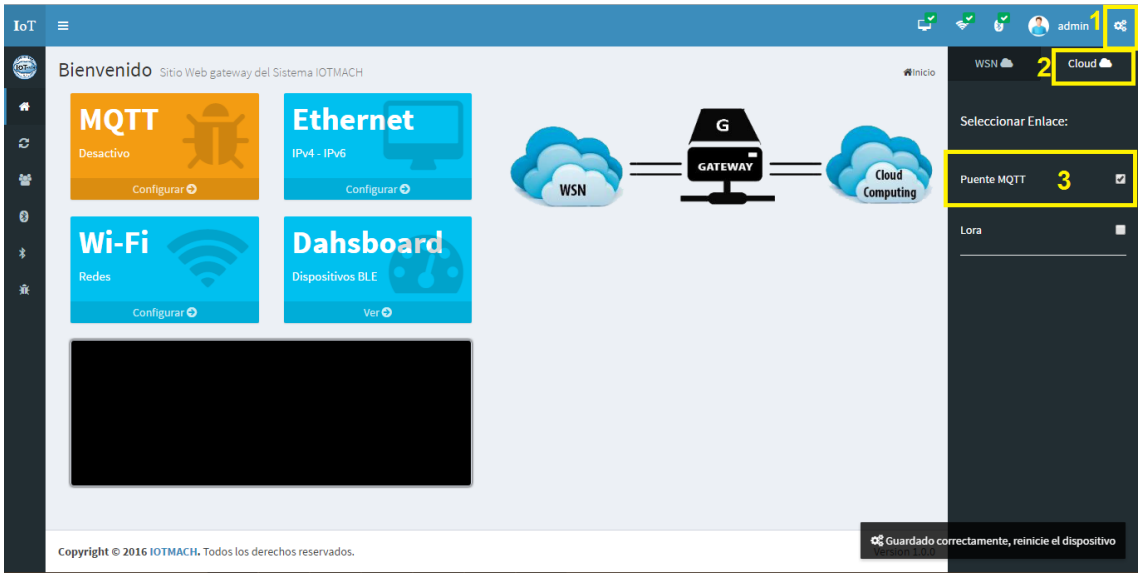
Figura 30. Pantalla seleccionar enlace WSN



Fuente: Elaboración propia

- En la Figura 31 se observa los pasos que el usuario debe realizar para activar un enlace hacia la *Cloud Computing.* Considerando el CU03.

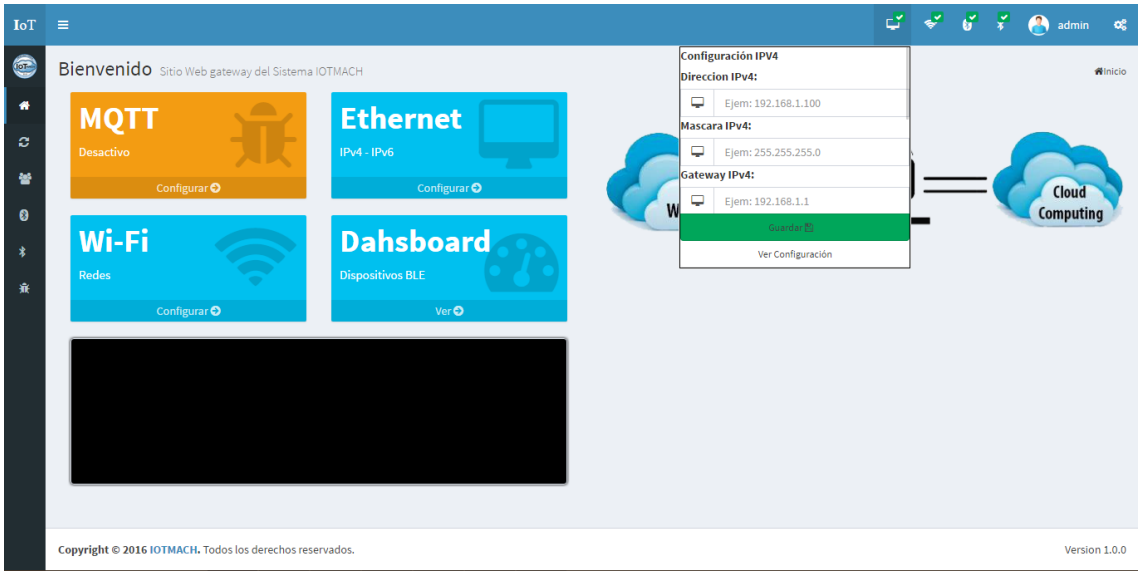
Figura 31. Pantalla seleccionar enlace Cloud Computing



Fuente: Elaboración propia

- Para que el usuario pueda gestionar la interfaz Ethernet, se deben ingresar los datos de la red en IPv4 o IPv6 según como se requiera, dar clic en el botón Guardar como muestra la Figura 32, con esto se cubre el CU04.

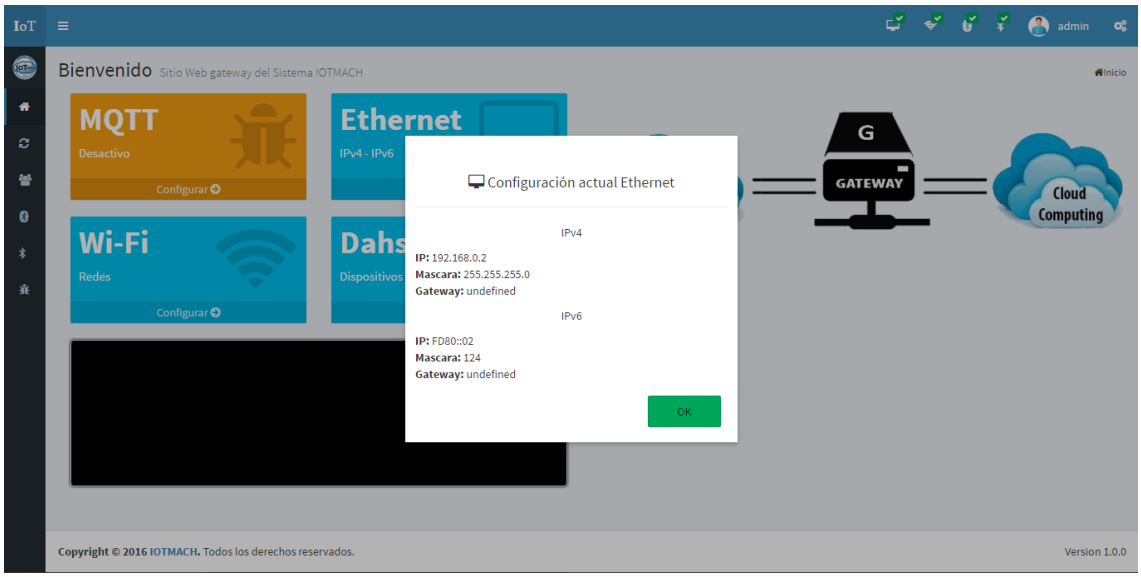
Figura 32. Pantalla configurar interfaz Ethernet



Fuente: Elaboración propia

- En la Figura 32 si el usuario da clic en el botón ver configuración, se muestra la configuración de la interfaz Ethernet, como se puede observar en la Figura 33.

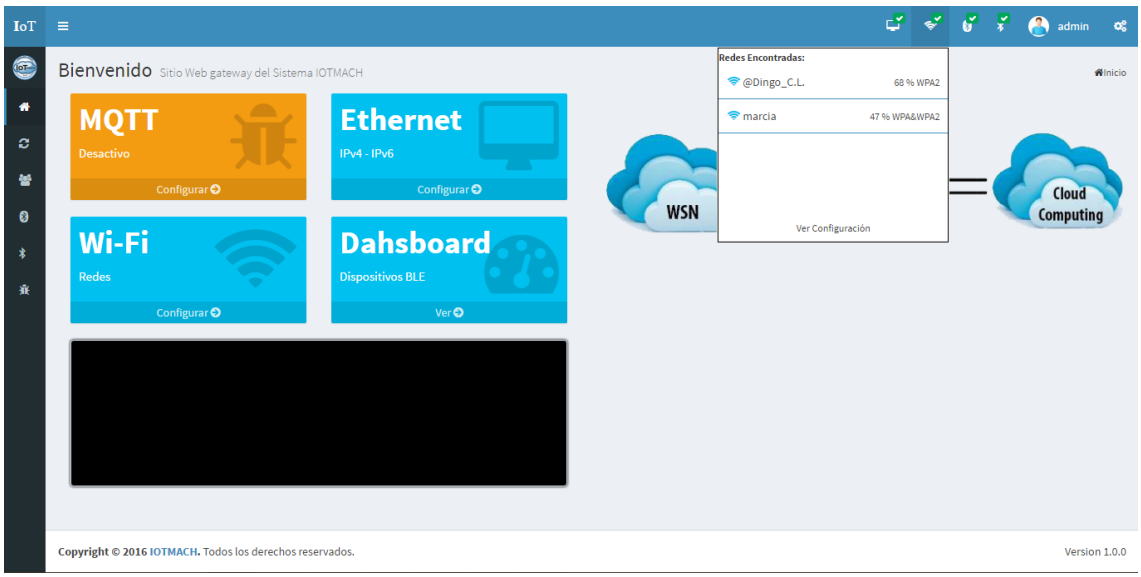
Figura 33. Pantalla muestra configuración Ethernet



Fuente: Elaboración propia

- Para la gestión de la interfaz Wi-Fi especificada en el CU05, primero el sistema se encarga de mostrar al usuario todas las redes Wi-Fi cercanas como lo muestra la Figura 34.

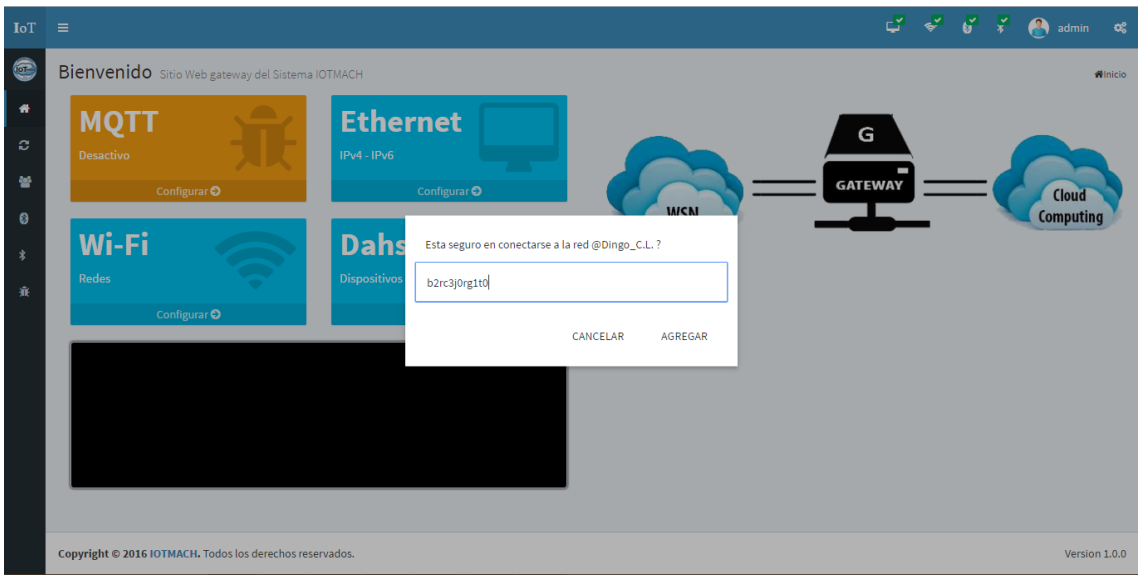
Figura 34. Pantalla mostrar redes Wi-Fi



Fuente: Elaboración propia

- El usuario al seleccionar una red Wi-Fi, si esta posee encriptamiento se le solicita la contraseña, como muestra la Figura 35.

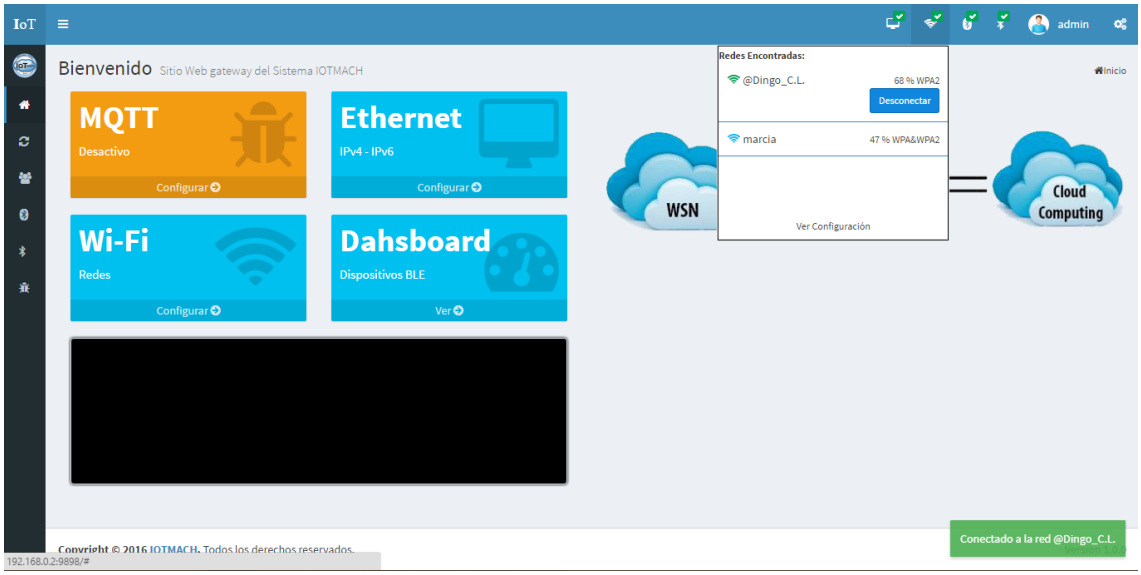
Figura 35. Pantalla ingreso de contraseña red Wi-Fi



Fuente: Elaboración propia

- Una vez ingresada la contraseña de la red Wi-Fi, se le muestra al usuario un mensaje de confirmación y aparece el botón desconectar sobre la red, como lo muestra la Figura 36.

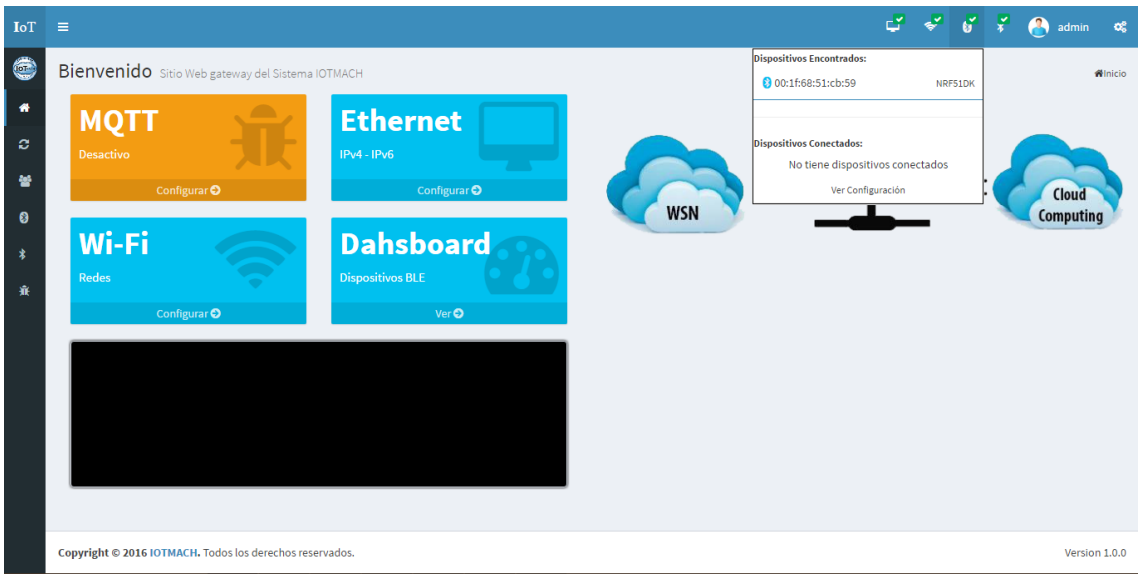
Figura 36. Pantalla red Wi-Fi conectada



Fuente: Elaboración propia

- Para ver un dispositivo BLE-MQTT se da clic sobre el icono del enlace y el sistema muestra los dispositivos cercanos al Gateway, como se observa en la Figura 37.

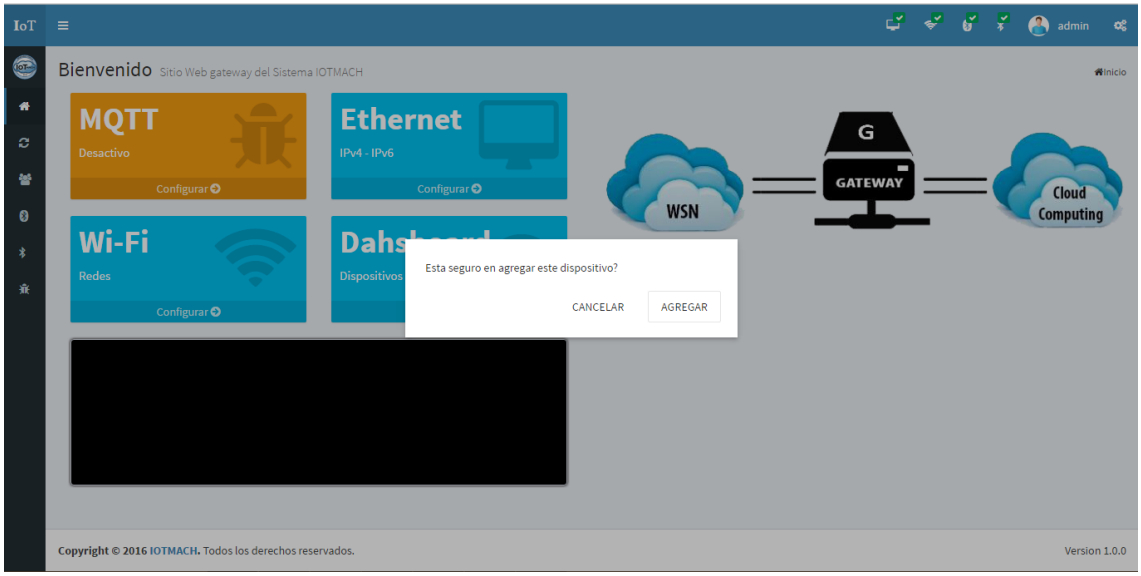
Figura 37. Pantalla mostrar dispositivos BLE-MQTT



Fuente: Elaboración propia

- Si el usuario desea agregar un dispositivo BLE-MQTT, debe dar clic sobre el nombre del mismo y el sistema le pregunta si está seguro como muestra la Figura 38.

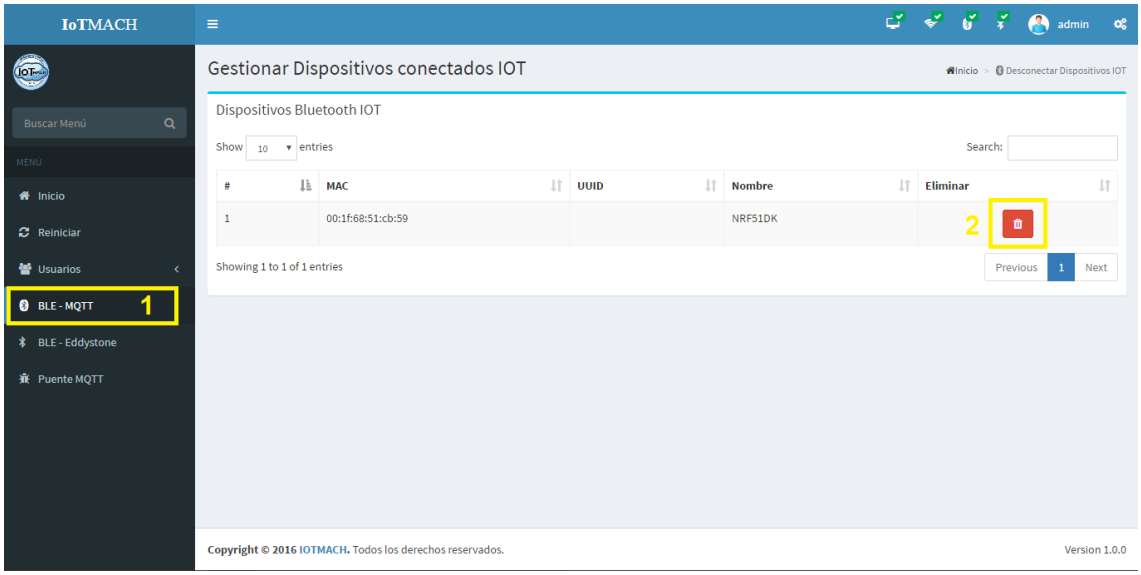
Figura 38. Pantalla conectar un dispositivo BLE-MQTT



Fuente: Elaboración propia

- Si el usuario desea eliminar un dispositivo BLE-MQTT aceptado, debe dar clic en el menú izquierdo como lo indica la Figura 39 y luego seleccionar el dispositivo.

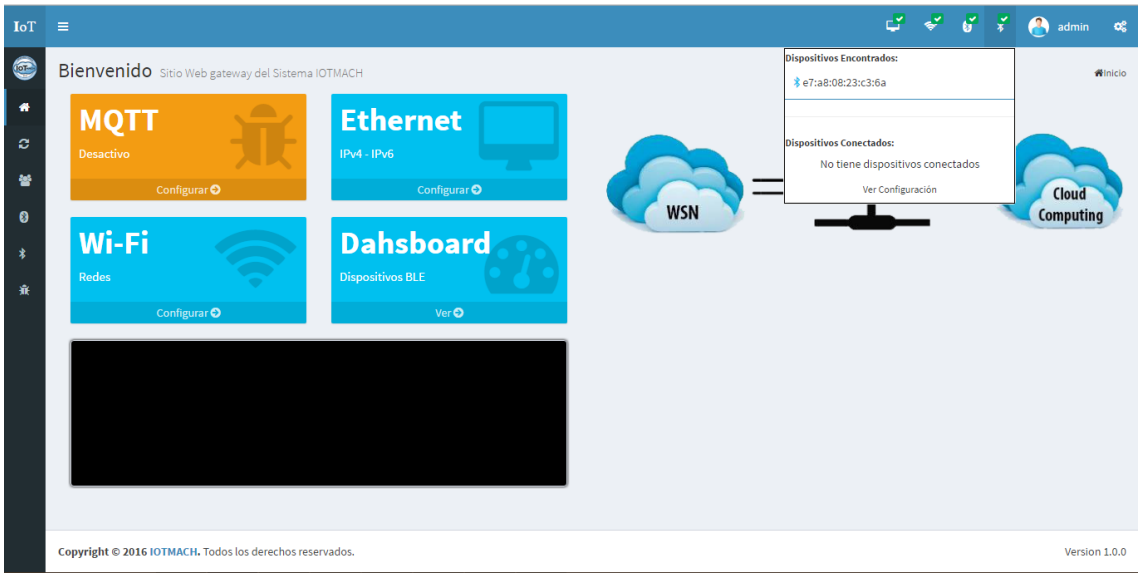
Figura 39. Pantalla eliminar dispositivo BLE-MQTT



Fuente: Elaboración propia

- Para gestionar los dispositivos BLE-Eddytstone especificado en el CU08, se realiza el mismo proceso de BLE-MQTT, en la Figura 40 se puede observar donde se muestran los dispositivos.

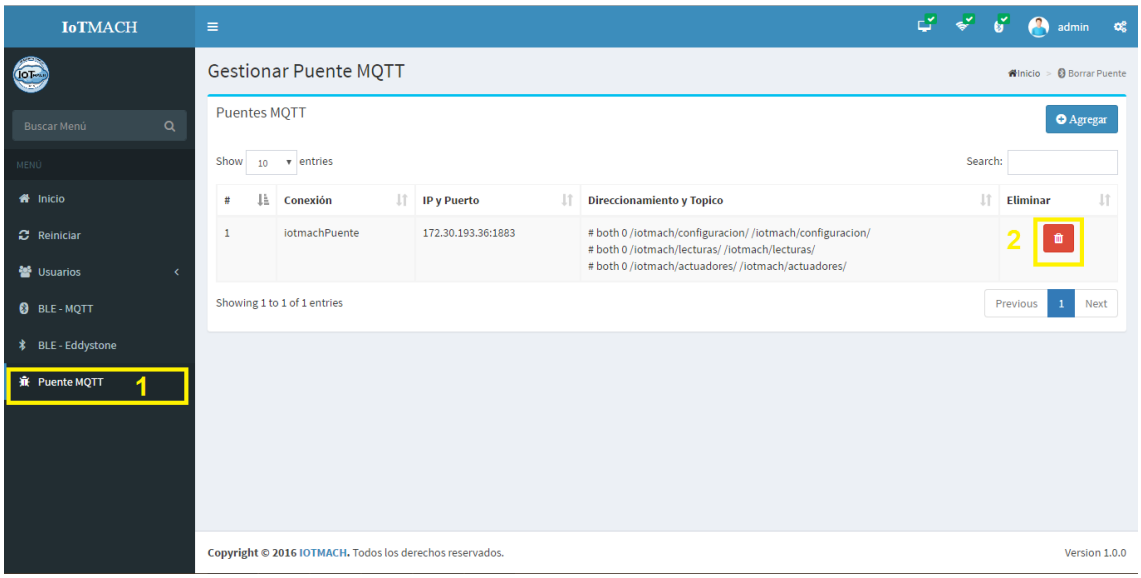
Figura 40. Pantalla mostrar dispositivos BLE-Eddystone



Fuente: Elaboración propia

- El usuario puede gestionar puentes MQTT, para eliminar un puente MQTT especificado en el CU06, se debe realizar los pasos de la Figura 41.

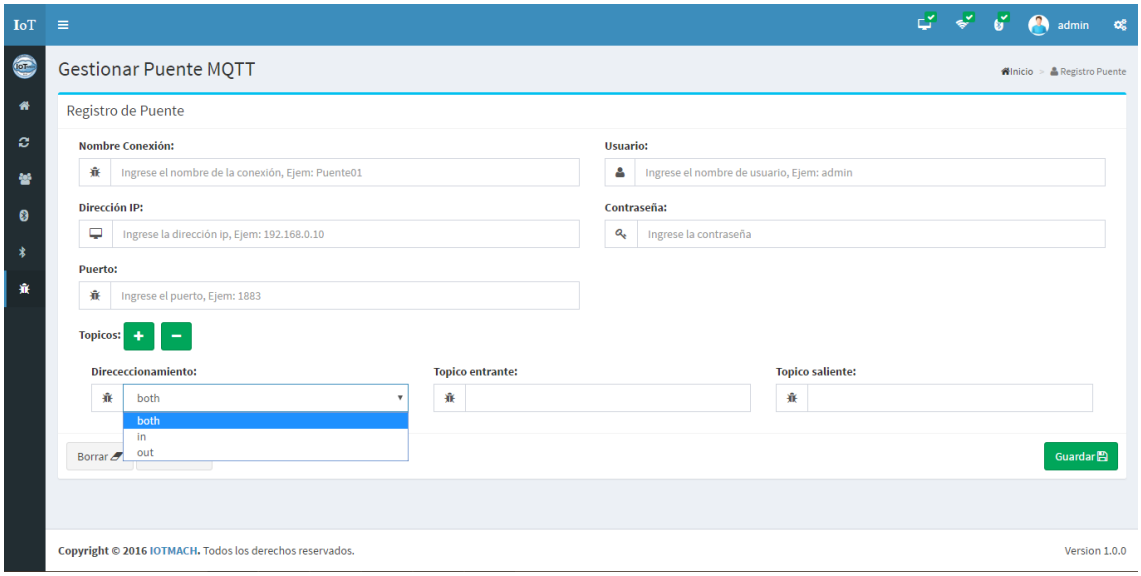
Figura 41. Pantalla para eliminar puente MQTT



Fuente: Elaboración propia

- Si el usuario desea agregar un puente MQTT, debe dar clic al botón agregar que muestra la Figura 41. Se abrirá la pantalla donde se debe ingresar obligatoriamente los siguientes datos: nombre de la conexión, dirección IP del Broker MQTT, puerto del bróker donde se realiza el puente, si el bróker tiene usuario y contraseña se debe ingresar. En la Figura 42 se muestra que además el usuario puede apuntar algunos tópicos de salida y entrada como también elegir el direccionamiento de cada tópico. Si elige direccionamiento booth el tópico está escuchando de entrada y salida, si es in el tópico hace puente solo de entrada y de ser out se realiza el puente solo a las salidas.

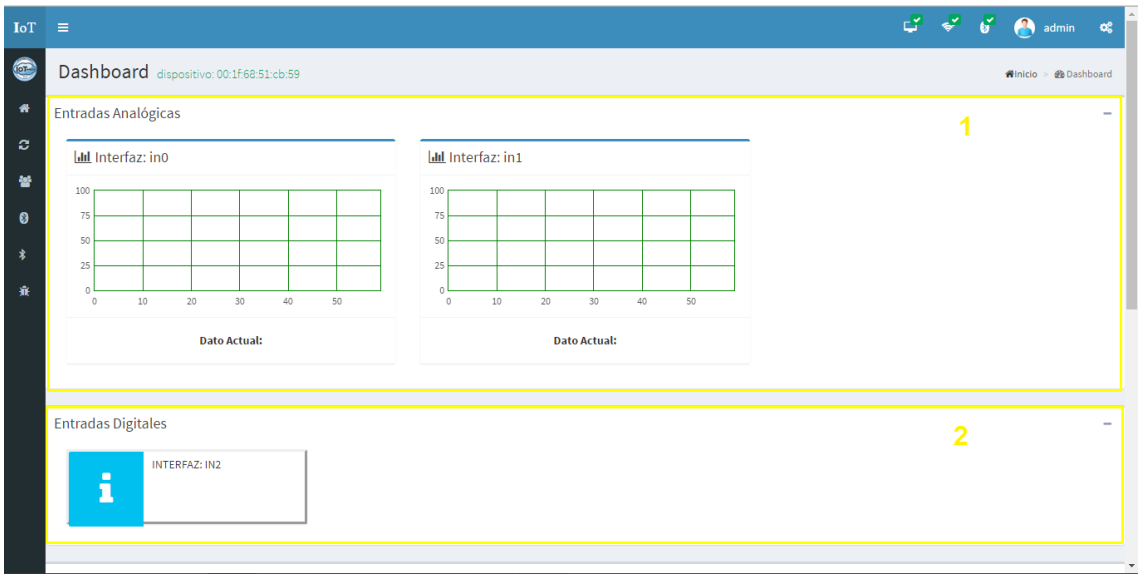
Figura 42. Pantalla para agregar puente MQTT



Fuente: Elaboración propia

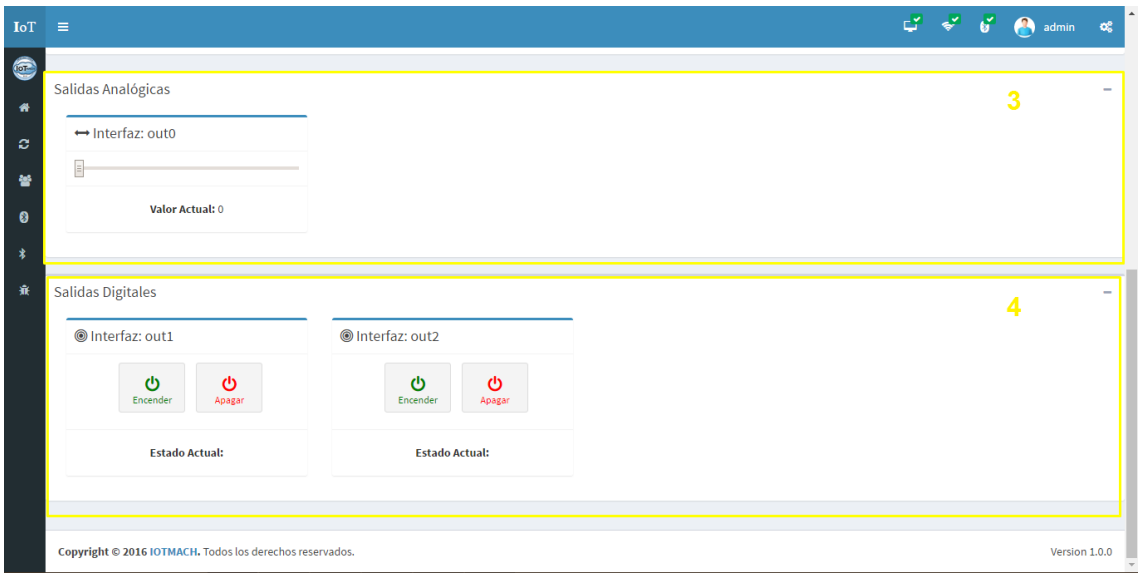
- El Dashboard especificado en el CU11 es auto-configurable, la Figura 43 muestra una parte de las secciones que son: 1. Los datos de las entradas analógicas, donde se usa cuadros estadísticos para que el usuario pueda visualizar las variaciones repentinas de una mejor manera. 2. Los datos de las entradas digitales, que permiten visualizar el dato y estado actual del sensor.
- La Figura 44 detalla la otra sección del Dashboard donde se encuentra: 3. Las salidas analógicas, el usuario puede manipular de una manera fácil el valor que desea. 4. Para que el usuario pueda controlar las salidas digitales, se usan dos botones (on, off) y el dato actual se muestra en tiempo real.

Figura 43. Pantalla Dashboard entradas



Fuente: Elaboración propia

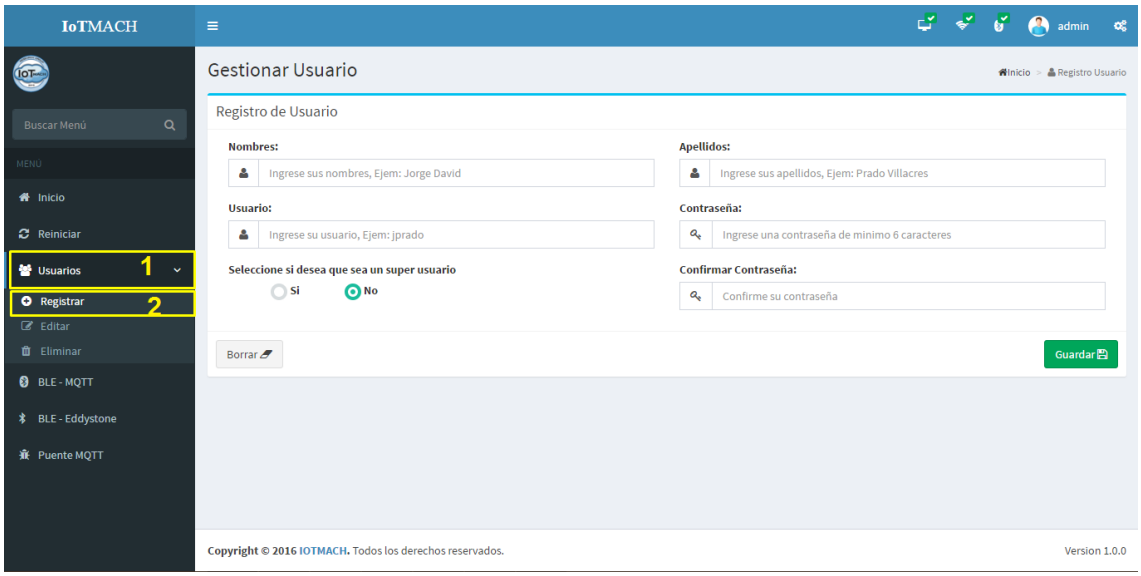
Figura 44. Pantalla Dashboard salidas



Fuente: Elaboración propia

- En la Figura 45 se puede observar el formulario de registro de usuarios, para lo cual se debe realizar: 1. Clic en usuarios. 2. Clic en registrar. Un usuario necesita obligatoriamente: nombre de usuario, contraseña que contenga más de 6 caracteres y confirmar la contraseña, los demás datos los puede llenar el usuario registrado al iniciar sesión.

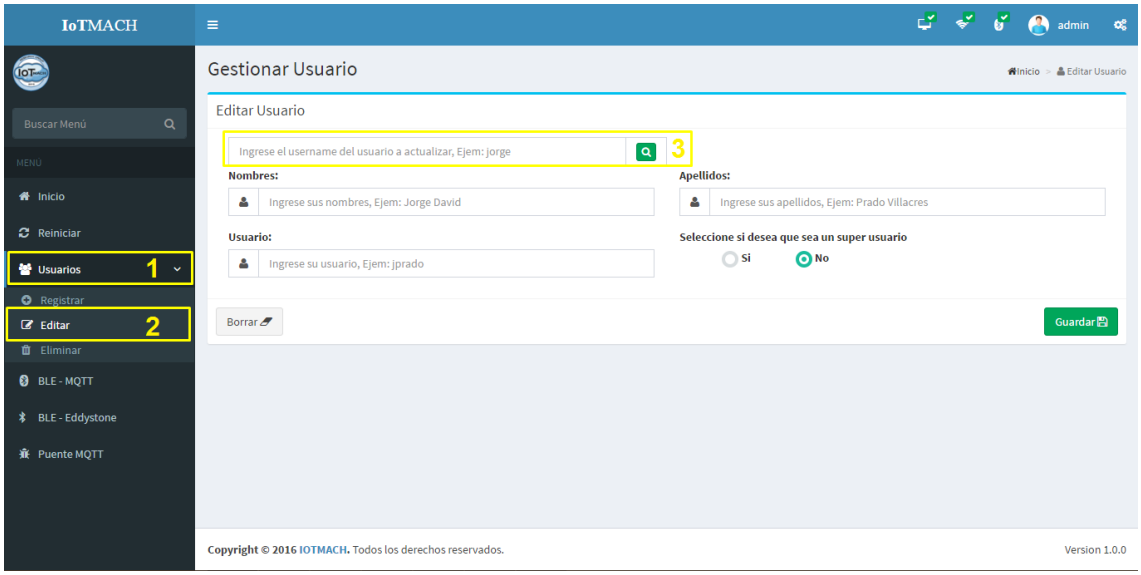
Figura 45. Pantalla registro de usuario



Fuente: Elaboración propia

- En la Figura 46 se observa los pasos que el usuario debe seguir para modificar los datos de un usuario específico: 1. Clic en el botón usuarios. 2. Clic en editar. 3 ingresar el nombre de usuario y dar clic en el botón buscar. Los datos del usuario buscado son presentados para que puedan ser modificados, una vez que se editan se da clic en guardar y se modificar la información registrada en la base de datos.

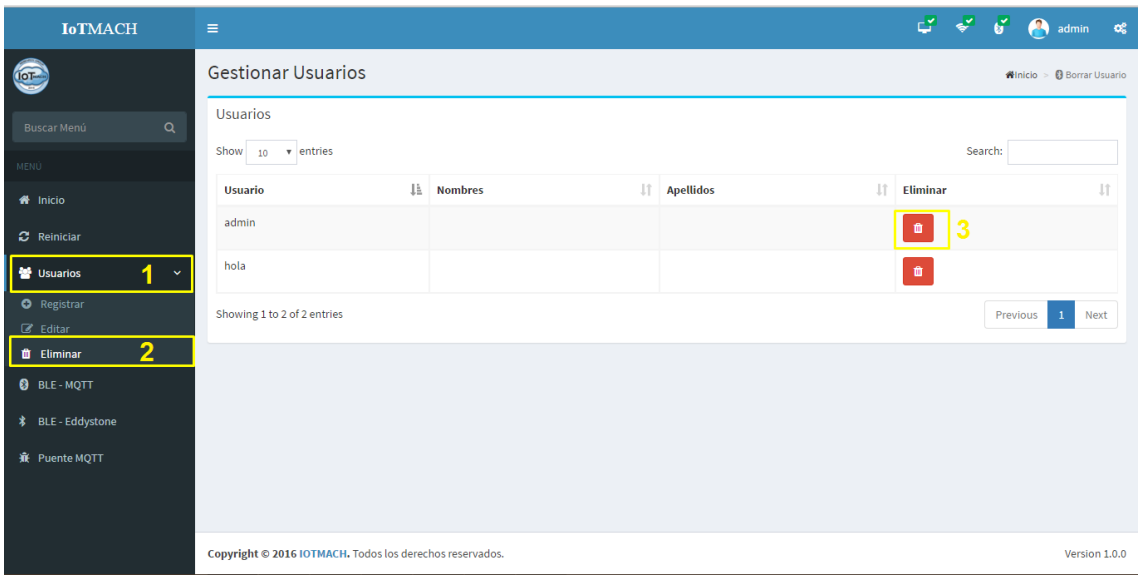
Figura 46. Pantalla editar usuario



Fuente: Elaboración propia

- Para eliminar un usuario, en la Figura 47 se puede observar 3 pasos que son: 1. Clic en el menú usuarios. 2. Clic en el sub-menú eliminar. 3. Clic en el botón eliminar de un usuario específico.

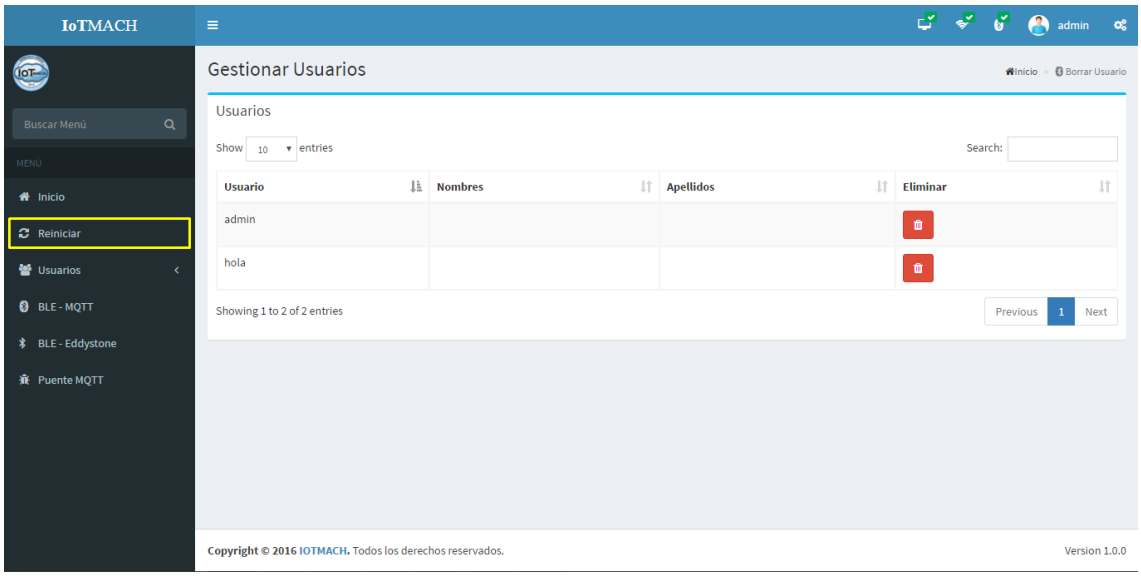
Figura 47. Pantalla para eliminar usuario



Fuente: Elaboración propia

- En algunos procesos se pide al usuario reiniciar el Gateway, en la Figura 48 se puede observar que en el menú existe esta opción.

Figura 48. Pantalla reiniciar Gateway



Fuente: Elaboración propia

3. EVALUACIÓN DEL PROTOTIPO

3.1 Plan de evaluación

Para comprobar ciertas funcionalidades del Gateway IOT se le realizaron varios tipos de pruebas, la Tabla 37 describe cada una de las evaluaciones realizadas. Esto permite medir el rendimiento, factibilidad, y verifica si se cumplió con los objetivos planteados.

Tabla 37. Planes de pruebas a realizar

Prueba	Descripción
Unitaria	Comprobar que todos los servicios y módulos puedan operar juntos de una manera correcta y eficiente.
Rendimiento	Hacer pruebas para medir el rendimiento del Broker Mosquitto.
Alcance	Comprobar la distancia que los dispositivos BLE deben estar del Gateway.
Usabilidad	Conocer si la interfaz de usuario es amigable y comprensible para un usuario común
Integración	Verificar que el Gateway se adapte a toda la estructura del sistema IOTMACH.

Fuente: Elaboración propia

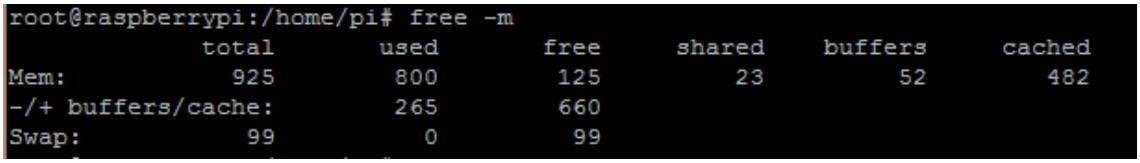
3.1.1 Prueba unitaria. Se realiza una prueba de laboratorio, que permita comprobar si todos los módulos programados pueden funcionar al mismo tiempo, para ello se ejecutan los módulos: Wi-Fi, Ethernet, Sockets, BLE-MQTT, BLE-Eddtystone, Puente MQTT y se verifica si el sitio WEB del Gateway IOT trabaja normalmente. Por lo cual se planteó evaluar cada uno de los módulos, en el Anexo I se detalla la prueba realizada y los parámetros son los siguientes:

- Si el Gateway puede buscar dispositivos inalámbricos (Wi-Fi, BLE-MQTT, BLE-Eddystone).
- Permite conectarse a los dispositivos inalámbricos (Wi-Fi, BLE-MQTT, BLE-Eddystone) encontrados.
- Permite desconectarse o eliminar los dispositivos inalámbricos (Wi-Fi, BLE-MQTT, BLE-Eddystone) aceptados.
- Los dispositivos inalámbricos (Wi-Fi, BLE-MQTT, BLE-Eddystone) que ya no están al alcance del Gateway desparecen de la lista de buscados.
- Se puede gestionar la interfaz Ethernet
- Es posible gestionar los puentes MQTT.
- Los datos enviados a la Cloud Computing y WSAN o WSN llegan correctamente.

- Se puede realizar el monitoreo y control de un dispositivo (BLE-MQTT, BLE-Eddystone), conectado.

Además se planteó conocer cuánto de memoria RAM queda en el GATEWAY IOT, cuando se ejecutan todos los módulos desarrollados, en la Figura 49 se observa que se cuenta con 925MB de las cuales, 800MB están ocupadas y 125MB libres, se destaca que no se ocupa la memoria SWAP.

Figura 49. Consumo de memoria RAM del Gateway



Fuente: Elaboración propia

3.1.2 Prueba de rendimiento. Se realiza una prueba de laboratorio, que permita determinar: cuántos clientes conectados y publicando mensajes al mismo tiempo soporta el Broker Mosquitto del Gateway IOT, el detalle de esta prueba se encuentra en el Anexo I, se publicó durante un minuto diferentes mensajes en los tópicos siguientes: /iotmach/configuracion/, /WSN/lecturas/ y /iotmach/actuadores/.

3.1.3 Prueba de alcance. Se realizó una prueba de campo para medir el alcance de la tarjeta Bluetooth del Gateway IOT, para ello se utilizaron dos motes; uno con BLE-MQTT y el otro con BLE-Eddystone, se busca la distancia ideal en donde el Gateway IOT pueda encontrar y conectarse sin problemas con los motes, se midió la distancia en metros, en el Anexo J se detalla la prueba realizada.

3.1.4 Prueba de usabilidad. Cuando se realiza un software que necesita ser gestionado por uno o más usuarios, se debe conocer que tan amigable es el diseño, lenguaje empleado, etc. Para ello se planteó realizar una encuesta detallada en el Anexo K, en donde se desean evaluar los siguientes parámetros:

- Compatibilidad del sistema con el contexto y lenguaje del usuario
- Consistencia y patrones de componentes
- Aspectos visuales, estética, facilidad de lectura y diseño
- Control y libertado del usuario en el sistema
- Prevención de errores y recuperación del sistema

3.1.5 Prueba de integración. El Gateway debe integrarse al sistema IOTMACH, por lo que se realizó una prueba de campo detallada en el Anexo L, con la persona encargada de gestionar las conexiones de la *Cloud Computing* del sistema WEB de IOTMACH, para evaluar los parámetros siguientes:

- Comunicación del Gateway con la Cloud Computing.
- Integridad de la información enviada/recibida.
- Tiempo de respuesta.
- Información correcta.
- Acceso a la Base de Datos.

3.2 Resultados de la evaluación

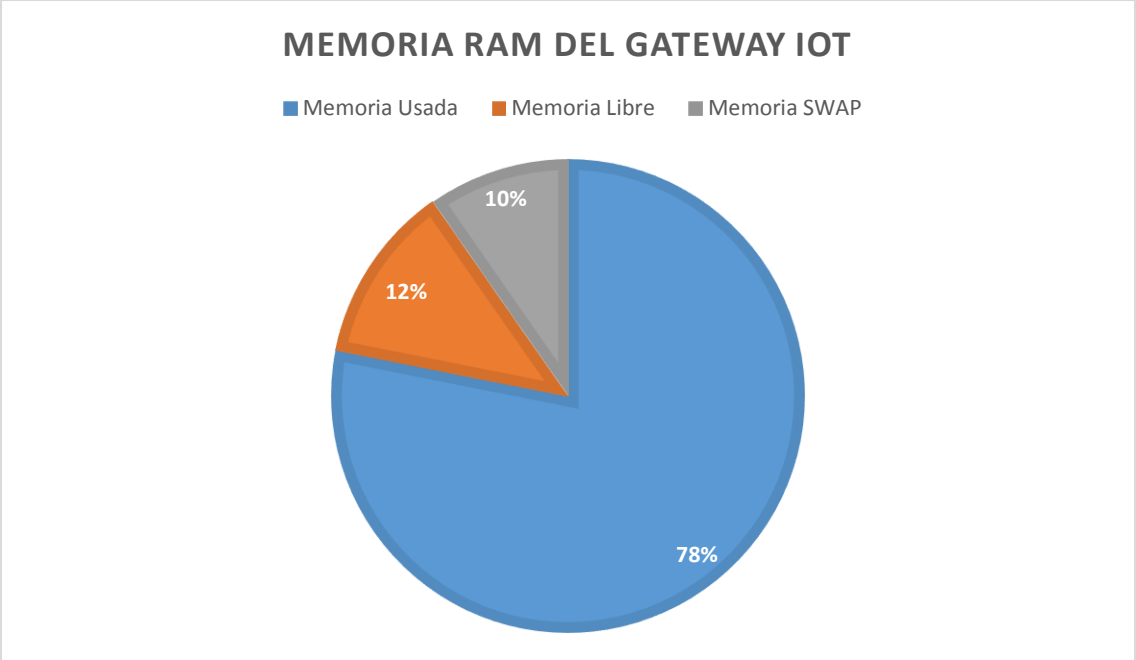
En cada una de las pruebas realizadas se obtuvieron resultados positivos y negativos, esto permite conocer; lo que está bien, los errores o lo que falta por integrarse en el Gateway IOT, en cada una de las pruebas aplicadas se obtuvieron los resultados siguientes.

3.2.1 Resultados de la prueba unitaria. Al momento de ejecutar y habilitar todos los módulos al mismo tiempo, se pudo realizar con normalidad cada una de las tareas del sitio WEB, las funciones que se ejecutan en *Background* se realizan sin ningún problema. Se encontró un problema o bug, cuando un dispositivo BLE MQTT o Eddystone desaparece, este se sigue mostrando durante un tiempo y si vuelve hacer encontrado por el Gateway IOT, este se repite en la lista.

El Gateway IOT usa 800MB de memoria RAM, al momento de trabajar con todos los módulos, quedando libres 125MB de RAM y 99MB de memoria SWAP, que hacen un total de 224MB, en la Figura 50 se puede observar que se está usando el 78% de memoria RAM y que solo el 22% está libre. El 22% de memoria libre es la suma de la memoria RAM libre con la memoria SWAP.

En el 22% de memoria libre se incluye la memoria SWAP, porque si el Gateway IOT se queda sin memoria RAM, entra en funcionamiento la memoria SWAP, que es la memoria reservada en el disco duro o almacenamiento del sistema operativo y sirve para cuando la memoria RAM se llena, entonces la memoria SWAP entra en funcionamiento junto a la memoria RAM.

Figura 50. Memoria RAM en la prueba unitaria



Fuente: Elaboración propia

3.2.2 Resultados de la prueba de rendimiento. El Gateway IOT necesita permitir un promedio de 8 motes conectados y enviando mensajes al mismo tiempo hacia el Broker MQTT, por lo que se tomó una muestra de 13 usuarios, para de esta manera conocer la capacidad del Broker MQTT con varios usuarios trabajando al mismo tiempo.

En la Tabla 38 se puede observar que en cada tópico durante un minuto los 13 usuarios conectados generaron los resultados siguientes: en el tópico /iotmach/configuracion/ con 1540 mensajes enviados se consumieron 5,5 MB de datos, con el tópico /WSN/lecturas/ en 2050 mensajes enviados se consumieron 6,6 MB y en el tópico /iotmach/actuadores/ enviando 3322 mensajes se gastaron 7,5 MB.

En los tres tópicos en un tiempo total de 3 minutos con 13 usuarios conectados al mismo tiempo, se enviaron un total de 6912 mensajes, consumiendo tan solo 19,6 MB del ancho de banda y el Broker MQTT del Gateway IOT respondió cada una de las peticiones satisfactoriamente.

Tabla 38. Resultados prueba de rendimiento

Tópico	Cantidad Mensajes	Total MB Consumidas
/iotmach/configuración/	1540	5,5 MB
/WSN/lecturas/	2050	6,6 MB
/iotmach/actuadores/	3322	7,5 MB
Fuente: Elaboración propia		

3.2.3 Resultados de la prueba de alcance. La aplicación de esta prueba ayudó a determinar el alcance a campo abierto, de la tarjeta Bluetooth 4.1 de la Raspberry Pi, usada como hardware del Gateway IOT, cuando se utiliza: BLE-MQTT y BLE-Eddystone, obteniendo como resultados que con: BLE-MQTT se tiene un mayor alcance que al utilizar BLE-Eddystone, como se puede observar en la Tabla 39.

Tabla 39. Resultados de la prueba de alcance

Tipo de Dispositivo	Alcance en metros
BLE-Eddystone	60m
BLE-MQTT	70m
Fuente: Elaboración propia	

3.2.4 Resultados de la prueba de usabilidad. En esta prueba se puede determinar la usabilidad por parte de los usuarios del sitio WEB del Gateway IOT, por cada parámetro evaluado se obtuvieron diferentes resultados, en la Tabla 40 se pueden observar los resultados de la encuesta aplicada a diferentes usuarios, el Anexo N contiene la lista de las personas encuestadas.

Tabla 40. Resultados prueba de usabilidad

Descripción	Si Cumple	No Cumple
Compatibilidad del sistema con el contexto y lenguaje del usuario	32	12
Consistencia y patrones	41	3
Aspectos visuales, estética, facilidad de lectura y diseño	27	3
Control y libertad del usuario	25	5
Prevención de errores y recuperación	20	10
Fuente: Elaboración propia		

3.2.5 Resultados prueba de integración. En la prueba de integración se obtuvieron resultados positivos, en cada uno de los parámetros medidos, el Gateway se integra de manera correcta con la Cloud Computing del sistema IOTMACH, los mensajes enviados no se pierden y llegan correctamente.

CONCLUSIONES

- Con la investigación bibliográfica desarrollada se puede concluir que ayudó a determinar: las tecnologías inalámbricas, protocolos de comunicación y estándares, actualmente utilizados en la IOT, para que permitan: la reducción del gasto de energía, consumo de datos y las herramientas a utilizar en el desarrollo del Gateway IOT.
- El análisis del Gateway IOT, ayudo a realizar cada uno de los requisitos elaborados para cumplir con los requerimientos planteados en este trabajo, con el diseño se bosquejo: la iteración de los módulos a programar, el modelado de la base de datos y la arquitectura del Gateway IOT. Se puede concluir que se obtuvieron cada una de las necesidades a desarrollar.
- Con la instalación y configuración de los diferentes servicios aplicados en el desarrollo del Gateway IOT, los cuales se seleccionaron en base: a la investigación bibliográfica y requisitos establecidos, se puede concluir que se obtiene un Gateway IOT funcionando en *Background*, sin permitir la gestión de algún usuario.
- Con el desarrollo del sitio WEB, se puede concluir que se obtiene la administración de los servicios, a través de un entorno grafico por parte de los usuarios, donde se permite gestionar cada una de las interfaces físicas (Ethernet, Wi-Fi, Bluetooth) del Gateway IOT, activando y desactivando las que el usuario desee, se puede controlar cada uno de los motes de la WSN o WSAN enlazados con el Gateway IOT. Por lo cual el usuario tiene el total control sobre el Gateway IOT
- Con la aplicación de las diferentes del Gateway IOT, se puede concluir que se implementa una versión estable y escalable, la misma puede ser mejorada añadiéndole otras funcionalidades o haciéndola más eficiente.

RECOMENDACIONES

- Incorporar otros tipos de tecnologías inalámbricas entre una WSN o WSAN y el Gateway, por ejemplo: Lora, Z-Wave, de la misma manera se optimizaría el funcionamiento del Gateway incorporándole Lora entre la comunicación del Gateway y la Cloud Computing.
- Realizar el análisis y diseñó, para mejorar el rendimiento del hardware usado en el Gateway IOT.
- Agregar otros estándares o protocolos de comunicación que se utilicen en la IOT, para que el Gateway cuenta con otro método de intercambio de informacion, tanto con la WSN y la Cloud Computing.
- Optimizar el sitio WEB del Gateway IOT, haciéndolo más amigable para los usuarios que lo utilicen. Implementar diferentes niveles de seguridad en los usuarios que administren el Gateway IOT. Desarrollar un mecanismo de calibración desde el Gateway IOT para cada uno de los sensores o actuadores que se encuentran en los motes. Elaborar un algoritmo que permita a los usuarios gestionar eficientemente los diferentes puentes MQTT. Agregar un módulo que permita controlar el GPIO de la Raspberry Pi 3, para que el Gateway IOT puede tener conectados actuadores o sensores.
- Ejecutar pruebas que permitan determinar el promedio de vida útil del Gateway IOT en diferentes condiciones de uso.

BIBLIOGRAFÍA

[1] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, “Middleware for Internet of Things: A Survey,” *IEEE Internet Things J.*, vol. 3, no. 1, pp. 70–95, Feb. 2016.

[2] D. Niyato, X. Lu, P. Wang, D. I. Kim, and Z. Han, “Economics of Internet of Things: an information market approach,” *IEEE Wirel. Commun.*, vol. 23, no. 4, pp. 136–145, Aug. 2016.

[3] P. Suresh, J. V. Daniel, V. Parthasarathy, and R. H. Aswathy, “A state of the art review on the Internet of Things (IoT) history, technology and fields of deployment,” in *2014 International Conference on Science Engineering and Management Research (ICSEMR)*, 2014, pp. 1–8.

[4] Y. Huang and G. Li, “Descriptive models for Internet of things,” in *Proceedings of 2010 International Conference on Intelligent Control and Information Processing, ICICIP 2010*, 2010, no. PART 2, pp. 483–486.

[5] M. A. Mehaseb, Y. Gadallah, and H. El-Hennawy, “WSN application traffic characterization for integration within the internet of things,” in *Proceedings - IEEE 9th International Conference on Mobile Ad-Hoc and Sensor Networks, MSN 2013*, 2013, pp. 318–323.

[6] B. B. P. Rao, P. Saluia, N. Sharma, A. Mittal, and S. V Sharma, “Cloud computing for Internet of Things & sensing based applications,” in *2012 Sixth International Conference on Sensing Technology (ICST)*, 2012, pp. 374–380.

[7] D. Elmazi, S. Sakamoto, T. Oda, E. Kulla, E. Spaho, and L. Barolli, “Effect of Security Parameter for Selection of Actor Nodes in WSAN: A Comparison Study of Two Fuzzy-Based Systems,” in *2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA)*, 2016, vol. 21, no. 1, pp. 957–964.

[8] J. Lee, Y. Su, and C. Shen, “A Comparative Study of Wireless Protocols :,” in *IECON Proceedings (Industrial Electronics Conference)*, 2007, pp. 46–51.

[9] J. Lindh, “Bluetooth Low Energy Beacons,” *Texas Instruments*, no. January, pp. 1–13, 2015.

[10] D. Han, J.-M. Chung, and R. C. Garcia, “Energy efficient wireless sensor networks based on 6LoWPAN and virtual MIMO technology,” in *2012 IEEE 55th*

International Midwest Symposium on Circuits and Systems (MWSCAS), 2012, pp. 849–852.

[11] J. Hanumanthappa, D. H. Manjaiah, and C. V. Aravinda, “An Innovative Simulation, Comparison Methodology & Framework for Evaluating the Performance Evaluation of a Novel IPv4/IPv6 Transition Mechanisms: BD-SIIT vs DSTM,” in *2010 First International Conference on Integrated Intelligent Computing*, 2010, pp. 258–263.

[12] K. Anusha, “Redundancy based WEP routing technology (IoT-WSN),” in *2015 International Conference on Signal Processing and Communication Engineering Systems*, 2015, pp. 407–410.

[13] C. Bormann, A. P. Castellani, and Z. Shelby, “CoAP: An Application Protocol for Billions of Tiny Internet Nodes,” *IEEE Internet Comput.*, vol. 16, no. 2, pp. 62–67, Mar. 2012.

[14] M. H. Asghar and N. Mohammadzadeh, “Design and simulation of energy efficiency in node based on MQTT protocol in Internet of Things,” in *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, 2015, pp. 1413–1417.

[15] F. Zhen, F. JingQi, and S. Wei, “The gateway anomaly detection and diagnosis in WSNs,” in *2016 Chinese Control and Decision Conference (CCDC)*, 2016, pp. 2401–2406.

[16] Z. Guo, I. G. Harris, C. B. Harris, Y. Jiang, and L. Tsaur, “A residual battery-aware routing algorithm based on DSR for BLE sensor networks,” in *2016 Wireless Telecommunications Symposium (WTS)*, 2016, pp. 1–6.

[17] H. L. Hao Chen, Xueqin Jia, “A Brief Introduction to IoT Gateway,” in *Proceeding of ICCTA*, 2011, vol. 7, pp. 5–8.

[18] Cooking-hacks, “Meshlium 802.15.4-PRO-3G-AP,” *cooking-hacks*, 20016. [Online]. Available: https://www.cooking-hacks.com/meshlium-802-15-4-pro-3g-ap-5906.

[19] S. Hassan, U. Qamar, and M. A. Idris, “Purification of requirement engineering model for rapid application development,” in *2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, 2015, pp. 357–362.

[20] B. Prashanth Kumar and Y. Prashanth, “Improving the Rapid Application

Development process model," in *2014 Conference on IT in Business, Industry and Government (CSIBIG)*, 2014, pp. 1–3.

[21] M. Zhang, F. Sun, and X. Cheng, "Architecture of Internet of Things and Its Key Technology Integration Based-On RFID," in *2012 Fifth International Symposium on Computational Intelligence and Design*, 2012, vol. 1, pp. 294–297.

[22] C. Gomez, J. Oller, and J. Paradells, "Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology," *Sensors*, vol. 12, no. 12, pp. 11734–11753, Aug. 2012.

[23] J. DeCuir, "Introducing Bluetooth Smart: Part 1: A look at both classic and new technologies.," *IEEE Consum. Electron. Mag.*, vol. 3, no. 1, pp. 12–18, Jan. 2014.

[24] A. Akinsiku and D. Jadav, "BeaSmart: A beacon enabled smarter workplace," in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, 2016, pp. 1269–1272.

[25] J. F. H. Barril, J. Ruyter, and Qing Tan, "A view on Internet of Things driving Cloud Federation," in *2016 IEEE International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, 2016, pp. 221–226.

[26] van der M. R. Woods Viveca, "Gartner Says Worldwide Public Cloud Services Market Is Forecast to Reach $204 Billion in 2016," *gartner*, 2016. [Online]. Available: http://www.gartner.com/newsroom/id/3188817.

[27] B. Sun, F. Zhu, and W. Zhang, "Connect wireless sensor network with internet through cloud gateway," in *2014 11th International Computer Conference on Wavelet Actiev Media Technology and Information Processing(ICCWAMTIP)*, 2014, pp. 444–447.

[28] N. Chen, Y. Chen, Y. You, H. Ling, P. Liang, and R. Zimmermann, "Dynamic Urban Surveillance Video Stream Processing Using Fog Computing," in *2016 IEEE Second International Conference on Multimedia Big Data (BigMM)*, 2016, pp. 105–112.

[29] Ngo Manh Khoi, S. Saguna, K. Mitra, and C. Ahlund, "IReHMo: An efficient IoT-based remote health monitoring system for smart regions," in *2015 17th International Conference on E-health Networking, Application & Services (HealthCom)*, 2015, pp. 563–568.

[30] N. De Caro, W. Colitti, K. Steenhaut, G. Mangino, and G. Reali, "Comparison of two lightweight protocols for smartphone-based sensing," in *2013 IEEE 20th*

Symposium on Communications and Vehicular Technology in the Benelux (SCVT), 2013, pp. 1–6.

[31] M. H. Elgazzar, “Perspectives on M2M protocols,” in *2015 IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS)*, 2015, pp. 501–505.

[32] D. Thangavel, X. Ma, A. Valera, H.-X. Tan, and C. K.-Y. Tan, “Performance evaluation of MQTT and CoAP via a common middleware,” in *2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2014, no. April, pp. 1–6.

[33] M. S. D. Gupta, V. Patchava, and V. Menezes, “Healthcare based on IoT using Raspberry Pi,” in *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, 2015, pp. 796–799.

[34] I. PE, “Comparativa y análisis: Raspberry Pi vs competencia,” *comohacer*, 2014. [Online]. Available: http://comohacer.eu/wp-content/uploads/comparativa-raspberry-pi.png.

[35] S. A. Akash, A. Menon, A. Gupta, M. W. Wakeel, M. N. Praveen, and P. Meena, “A novel strategy for controlling the movement of a smart wheelchair using internet of things,” in *2014 IEEE Global Humanitarian Technology Conference - South Asia Satellite (GHTC-SAS)*, 2014, pp. 154–158.

[36] Raspbian, “Repositorio Raspbian,” *raspbian*, 2016. [Online]. Available: http://archive.raspbian.org/.

[37] Haolin Wang, Minjun Xi, Jia Liu, and Canfeng Chen, “Transmitting IPv6 packets over Bluetooth low energy based on BlueZ,” in *Advanced Communication Technology (ICACT)*, 2013, pp. 72–77.

[38] Z. Bradac, P. Fiedler, P. Cach, and R. Vrba, “Wireless communication in automation,” in *10th IEEE International Conference on Electronics, Circuits and Systems, 2003. ICECS 2003. Proceedings of the 2003*, 2003, pp. 659–662.

[39] A. Bansel, H. Gonzalez-Velez, and A. E. Chis, “Cloud-Based NoSQL Data Migration,” in *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, 2016, pp. 224–231.

[40] J. Ye, C. Zhang, L. Ma, H. Yu, and J. Zhao, “Efficient and Precise Dynamic Slicing for Client-Side JavaScript Programs,” in *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, 2016, pp. 449–

459.

[41] K. Lei, Y. Ma, and Z. Tan, “Performance Comparison and Evaluation of Web Development Technologies in PHP, Python, and Node.js,” in *2014 IEEE 17th International Conference on Computational Science and Engineering*, 2014, pp. 661–668.

[42] A. J. Poulter, S. J. Johnston, and S. J. Cox, “Using the MEAN stack to implement a RESTful service for an Internet of Things application,” in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, 2015, pp. 280–285.

[43] A. Campoverde, D. Hernández, and B. Mazón, “Cloud computing con herramientas open-source para Internet de las cosas,” *Maskana*, pp. 173–182, 2015.

[44] TutoElectro, “Cómo Instalar Raspbian en la Raspberry PI 3,” *youtube*, 2016. [Online]. Available: https://www.youtube.com/watch?v=vc1laWfUjyM.

ÍNDICE

ANEXOS

Anexo A. Manual de Instalación de Node JS en raspbian

Se debe descargar la versión de node JS que se desea instalar, para lo cual usamos el siguiente comando.

- wget https://nodejs.org/download/release/v4.4.0/node-v4.4.0-linux-armv7l.tar.gz

Después nos colocamos en la siguiente ruta.

- cd /usr/local

Luego tenemos que descomprimir el archivo descargado, en este caso se realiza de la siguiente manera.

- sudo tar xzvf ~/node-v4.4.0-linux-armv7l.tar.gz --strip=1

Con esto tendremos instalado node JS y npm, para verificar si se instaló correctamente se puede usar los siguientes comandos.

- node –v
- npm –v

Los comandos anteriores deben mostrarnos las versiones instaladas.

Anexo B. Manual de Instalación y configuración de MongoDB

Para descargar e instalar mongodb se hace uso del comando apt-get.

- apt-get install mongodb

Para verificar que la instalación fue correcta en la consola se escribe mongo y debería abrirse la Shell de mongo.

Una vez instalado se configura mongodb para darle seguridad y acceso remoto, para lo cual se necesita editar el fichero /etc/mongo.conf

- nano /etc/mongo.conf

Se debe buscar la línea que tenga bind_ip y auth la cual se la debe dejar de la siguiente manera:

- bind_ip = 0.0.0.0
- la línea auth se la debe descomentar.

Se cierra el archivo y se guarda presionando las teclas Control + x. Ahora se necesita abrir la consola o Shell de mongo, para lo cual basta con escribir mongo. Una vez abierta la Shell procedemos a escribir.

- use admin

Después se necesita crear un súper usuario que en este caso se escribe lo siguiente en la consola de mongo:

db.addUser({

user:"pi",

pwd: "raspberry",

roles:["clusterAdmin","readAnyDatabase","readWriteAnyDatabase","userAdminAnyDatabase","dbAdminAnyDatabase"]

});

Una vez creado el usuario se escribe exit y se reinicia el servicio

- /etc/init.d/mongodb restart

Ahora ya se puede crear las bases de datos o acceder remotamente a mongodb.

Se realiza un update.

- sudo apt-get update

Luego se debe instalar las siguientes librerías, es necesaria la conexión a internet.

- sudo apt-get install libssl-dev
- sudo apt-get install cmake
- sudo apt-get install libc-ares-dev
- sudo apt-get install uuid-dev
- sudo apt-get install daemon

Para que mosquitto pueda trabajar con sockets se necesita descargar e instalar libsockets, para la descarga ejecutamos lo siguiente:

- wget http://git.libwebsockets.org/cgi-bin/cgit/libwebsockets/snapshot/libwebsockets-1.4-chrome43-firefox-36.tar.gz

Luego se tiene que descomprimir el archivo descargado para lo cual se ejecuta lo siguiente:

- tar zxvf libwebsockets*

Accedemos con el comando cd al directorio descargado.

- cd libwebsockets*

Se necesita crear un directorio que lo llamaremos build, este ayuda directorio ayuda a compilar libsokets, para realizar lo mencionado en la consol ejecutamos uno a uno los siguientes comandos:

- mkdir build
- cd build
- cmake ..
- sudo make install
- sudo ldconfig

Ahora retrocedemos de directorio con cd .. y se descarga mosquitto con el siguiente comando.

- wget http://mosquitto.org/files/source/mosquitto-1.4.1.tar.gz

Se necesita descomprimir el archivo descargado para lo cual ejecutamos lo siguiente

- tar zxvf mosquitto-1.4.1.tar.gz

Se accede a la carpeta descomprimida con el siguiente comando

- cd mosquitto-1.4.1

Se necesita editar un archivo para lo cual en este caso se lo realiza con nano

- nano config.mk

Se busca la línea WITH_WEBSOCKETS:=no y se cambia no por yes, quedando de la siguiente manera.

- WITH_WEBSOCKETS:=yes

Ahora ya se puede instalar mosquitto, para lo cual se realiza lo siguiente:

- make
- sudo make install

Anexo D. Configuración básica de mosquitto

Una vez instalado mosquitto con las indicaciones del Anexo C, se necesita configurar para ello accedemos al siguiente directorio.

- cd /etc/mosquitto/

Se necesita crear un archivo donde se guardan las configuraciones de mosquitto.

- nano mosquitto.conf

En el archivo se escribe la siguiente configuración.

user root

listener 1883

listener 9001

protocol websockets

El puerto por donde escucha mosquitto es el 1883, partiendo de ahí se puede crear más puertos que escuche mosquitto pero con websockets, en este caso el puerto 9001.

Ahora para probar que está funcionando correctamente se debe correr el servidor de mosquitto con lo siguiente:

- mosquitto -c mosquitto.conf

Anexo E. Configuración de puente MQTT con mosquitto

Para configurar un puente MQTT usando mosquitto se necesita abrir y editar el archivo de configuraciones de mosquitto, para lo cual en este caso se usará nano.

- Nano /etc/mosquitto/mosquitto.conf

Una vez abierto el archivo anterior, se necesita ubicar en la última línea del archivo para escribir lo siguiente:

connection raspi-PC

address 192.168.10.115:1883

cleansession true

try_private true

notifications false

start_type automatic

username admin

password 123456

topic # both 0 /remote/sensor/

topic # both 0 /remote/actuador/

Donde la primera línea nos indica el nombre con el que se identifica la conexión, luego la segunda línea apunta al servidor y puerto con el que se realiza el puente MQTT. Cleansession, try_private, notifications, start_type indican a mosquitto que la conexión debe ser permanente y en caso de no poderse conectar intentar. Username y password se usa cuando el otro servidor mqtt está configurado con usuario y contraseña, y al final se indica los tópicos que hacen el puente mqtt.

Anexo F. Ejemplo de JSON para el tópico /iotmach/configuracion/

```
var configuracion= {
        nombre: "nombre de dispositivo",
        mac: "0000000ABF0",
        empresa: "070000000001",
        localizacion: "(00,-00)",
        descripcion: "descripcion del dispositivo",    //fabricante, tecnologia, etc
        datos: [
                {
                        categoria: "s", //s=sensor o a=actuador
                        senial: "a",    //a=analogico o d=digital
                        ted_id: "39.1", //ted de plantilla sensor-fabricante
                        interfaz: "in0", //ubicacion del sensor-actuador(in/out) en el mote
                },
                {
                        categoria: "a", //s=sensor o a=actuador
                        senial: "d",    //a=analogico o d=digital
                        ted_id: "35.2", //ted de plantilla sensor-fabricante
                        interfaz: "out0", //ubicacion del sensor-actuador(in/out) en el mote
                },
        ]
}
```

Anexo G. Ejemplo de JSON para el tópico /iotmach/lecturas/

```
var lecturas={
    paquete_id:"1000",
        mac: "00000",
        datos: [
                {interfaz: "in0", valor:"0"},
                {interfaz: "in1", valor:"0"},
                {interfaz: "out0", valor:"0"}, //0=apagado - 1=encendido
        ],
        fecha: "--/--/--- 00:00", //fecha del sistema del gateway o mote
        bateria: "10%";
}
```

Anexo H. Ejemplo de JSON para el topico /iotmach/actuadores/

```
var actuadores={
        paquete_id:"1000",
        mac: "0000000000000",
        datos: [
                {interfaz: "out0", valor:"0"},
                {interfaz: "out1", valor:"1"},
                {interfaz: "out7", valor:"55"}
        ]
        fecha: "--/--/---- 00:00:00"
}
```

Anexo I. Prueba unitaria

PLAN DE PRUEBA UNITARIA

Sistema Aplicación:	Gateway IOTMACH
Ciclo o Proceso:	Todos
Objetivo:	Conocer si todos los módulos del Gateway IOT funcionan al mismo tiempo de manera correcta.
Tipo de Prueba:	Técnica
Fecha:	17/agosto/2016

Roles	**Responsable**	**Firma**
Proveedor-Tester	Jorge Prado	Jorge Prado

- Comprobación del funcionamiento de los módulos

Descripción	**Parámetros**			**Observaciones**
	SI	**No**	**Falla**	
Wi-Fi				
Se muestran las redes Wi-Fi cercanas	X			
Se puede conectar a una Red Wi-Fi	X			
Se puede desconectar de la Red Wi-Fi	X			
Desaparecen la Redes Wi-Fi encontradas que ya no están cerca.	X			
Ethernet				
Se puede observar la configuración de la interfaz Ethernet	X			
Se puede cambiar la configuración de la interfaz Ethernet	X			
BLE- MQTT				
Se muestran los dispositivos Bluetooth cerca.	X			
Se puede conectar a un mote BLE-MQTT	X			

Puede eliminar un dispositivo BLE-MQTT aceptado	X			
Desaparecen los dispositivos Bluetooth encontrados y que ya no están cerca.			X	
BLE - Eddystone				
Se muestran los dispositivos Bluetooth cerca.	X			
Se puede conectar a un mote BLE-Eddystone	X			
Puede eliminar un dispositivo BLE-Eddystone aceptado	X			
Desaparecen los dispositivos Bluetooth encontrados y que ya no están cerca.	X			
Puente MQTT				
Se puede agregar un puente MQTT más hacia otra Cloud Computing	X			
Se puede ver los puentes MQTT agregados	X			
Se puede eliminar un puente MQTT			X	
Transmisión de Datos				
Los datos llegan normalmente a la Cloud Computing	X			
Los datos llegan normalmente a los motes de la WSAN	X			
Monitoreo y Control				
Se pueden observar los datos en tiempo real	X			
Se puede activar actuadores en tiempo real	X			

- Uso de memoria RAM

Total de Memoria	Memoria Usada	Memoria Disponible	Memoria SWAP
925MB	800MB	125MB	99MB

Anexo J. Modelo del plan de prueba del rendimiento del broker MQTT

PLAN DE PRUEBAS DE RENDIMIENTO

Sistema Aplicación:	Gateway IOTMACH		
Ciclo o Proceso:	Bróker MQTT		
Objetivo:	Medir el rendimiento del bróker MQTT con múltiples clientes y consumo de datos en la red de los mensajes MQTT		
Tipo de Prueba:	Técnica		
Lugar:	Universidad Técnica de Machala – Unidad Académica de Ingeniería Civil – Carrera de Ingeniería de Sistemas		
Área:	Laboratorio 6		
Hora Inicio:	10:27 am	**Hora Fin:**	11:00 am
Fecha:	18/agosto/2016		

Roles	**Responsable**	**Firma**
Proveedor	Jorge Prado	Jorge Prado

- **Requerimientos previos a la prueba:**

Conocimientos básicos en computación.

Instalación de MQTT lens

- **Descripción de la Prueba:**

Se piensa enviar varios mensajes al Bróker MQTT para saber si hay pérdidas de datos y determinar el consumo de datos de los mensajes.

- **Plan de Pruebas:**

Detalle	**Valor**
Tópico:	/iotmach/configuración/
Numero de Mensajes:	1540
Tiempo:	1 minuto
Clientes conectados:	13
IP del bróker:	192.168.1.106
Puerto:	1883

Mb inicial:	81,5
Mb final:	87,0

Mensaje enviado:	{
	"nombre":"NRF51DK",
	"mac":"00:1f:68:51:cb:59",
	"empresa":"0704640333001",
	"localizacion":"(00,-00)",
	"descripcion":"descripcion del dispositivo",
	"datos":[
	{
	"categoria":"s",
	"senial":"a",
	"ted_id":39,
	"interfaz":"in0"
	},
	{
	"categoria":"s",
	"senial":"a",
	"ted_id":39,
	"interfaz":"in4"
	}
	]
	}

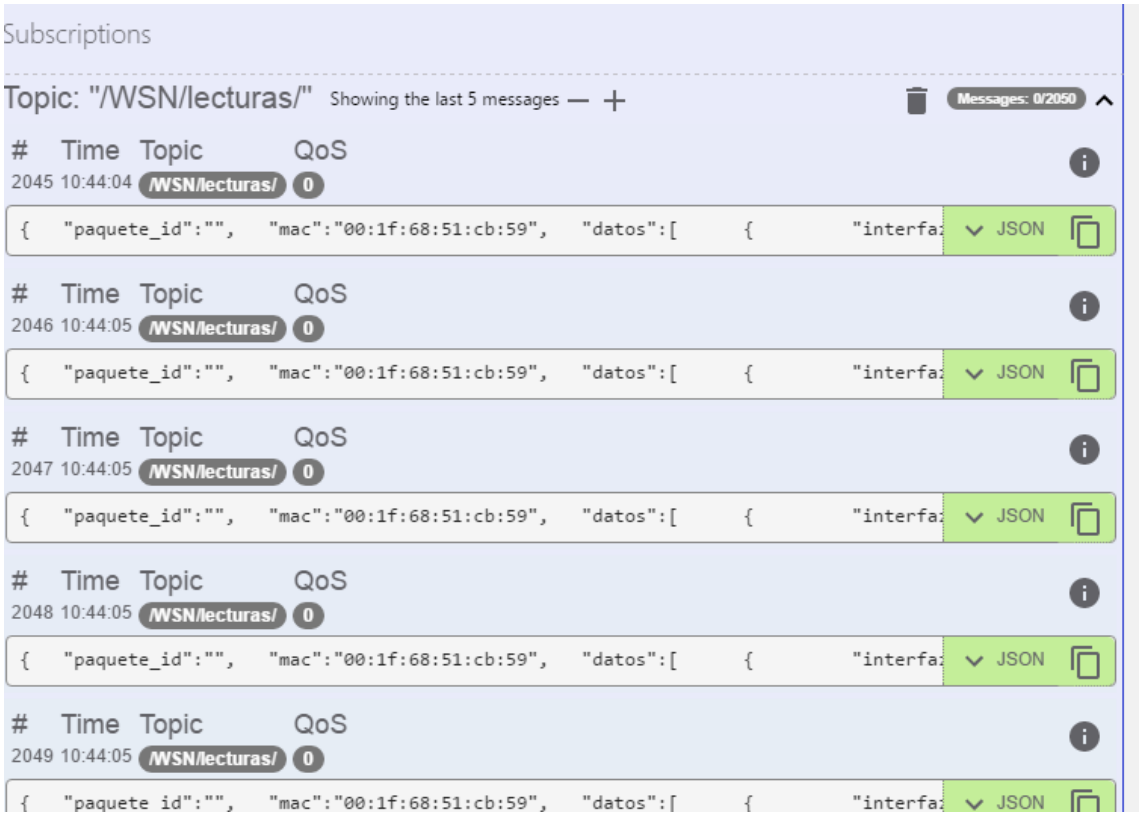
Tópico:	/WSN/lecturas/
Numero de Mensajes:	2050
Tiempo:	1 minuto
Clientes conectados:	13
IP del bróker:	192.168.1.106
Puerto:	1883
Mb inicial:	87,0
Mb final:	93, 6

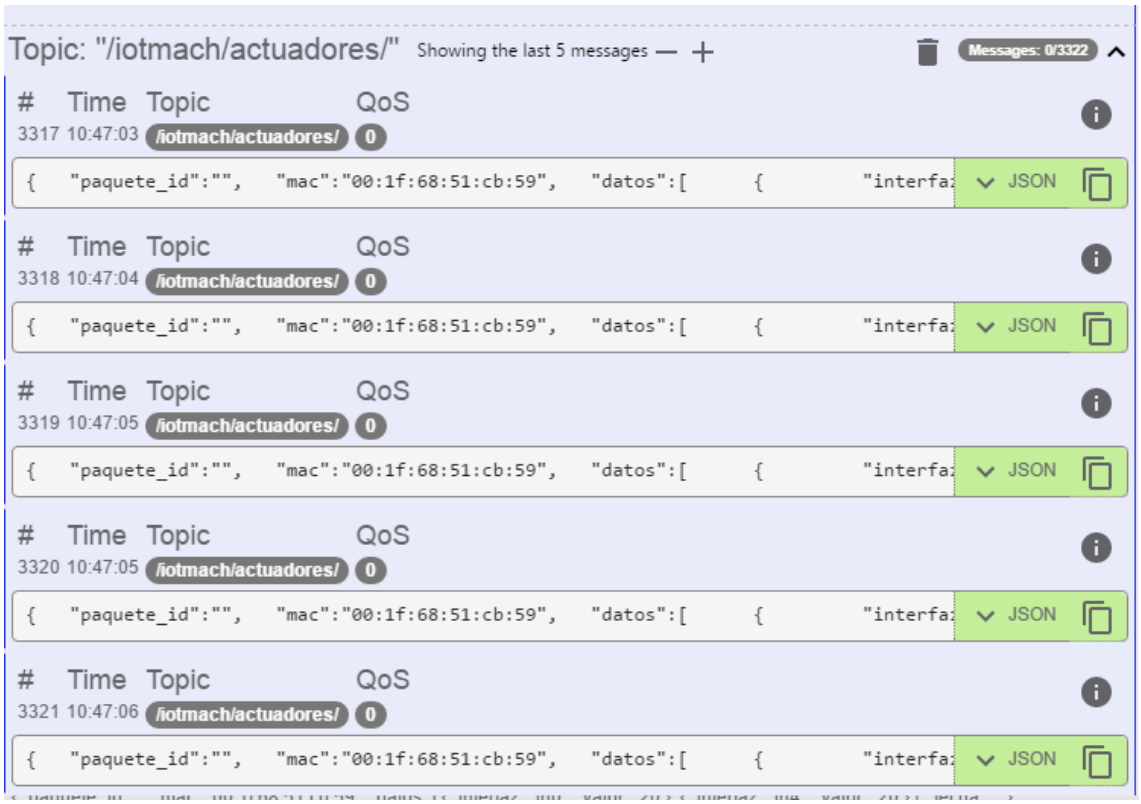
Mensaje enviado:	{
	"paquete_id":"",
	"mac":"00:1f:68:51:cb:59",
	"datos":[
	{
	"interfaz":"in0",
	"valor": 20
	},
	{
	"interfaz":"in4",
	"valor": 20
	}
	],
	"fecha": "",
	"bateria": "10%"
	}

Tópico:	/iotmach/actuadores/

Numero de Mensajes:	3322
Tiempo:	1 minuto
Clientes conectados:	13
IP del bróker:	192.168.1.106
Puerto:	1883
Mb inicial:	93,6
Mb final:	101,1
Mensaje enviado:	{ "paquete_id":"", "mac":"00:1f:68:51:cb:59", "datos":[{ "interfaz":"in0", "valor": 20 }, { "interfaz":"in4", "valor": 20 }], "fecha":"" }

- **Observaciones al plan de Prueba:**


Subscriptions
Topic: "/WSN/lecturas/" Showing the last 5 messages
Messages: 0/2050
Time Topic QoS
2045 10:44:04 /WSN/lecturas/ 0
{ "paquete_id":"", "mac":"00:1f:68:51:cb:59", "datos":[{ "interfa
JSON
Time Topic QoS
2046 10:44:05 /WSN/lecturas/ 0
{ "paquete_id":"", "mac":"00:1f:68:51:cb:59", "datos":[{ "interfa
JSON
Time Topic QoS
2047 10:44:05 /WSN/lecturas/ 0
{ "paquete_id":"", "mac":"00:1f:68:51:cb:59", "datos":[{ "interfa
JSON
Time Topic QoS
2048 10:44:05 /WSN/lecturas/ 0
{ "paquete_id":"", "mac":"00:1f:68:51:cb:59", "datos":[{ "interfa
JSON
Time Topic QoS
2049 10:44:05 /WSN/lecturas/ 0
{ "paquete_id":"", "mac":"00:1f:68:51:cb:59", "datos":[{ "interfa
JSON


Topic: "/iotmach/actuadores/" Showing the last 5 messages
Messages: 0/3322
Time Topic QoS
3317 10:47:03 /iotmach/actuadores/ 0
{ "paquete_id":"", "mac":"00:1f:68:51:cb:59", "datos":[{ "interfa
JSON
Time Topic QoS
3318 10:47:04 /iotmach/actuadores/ 0
{ "paquete_id":"", "mac":"00:1f:68:51:cb:59", "datos":[{ "interfa
JSON
Time Topic QoS
3319 10:47:05 /iotmach/actuadores/ 0
{ "paquete_id":"", "mac":"00:1f:68:51:cb:59", "datos":[{ "interfa
JSON
Time Topic QoS
3320 10:47:05 /iotmach/actuadores/ 0
{ "paquete_id":"", "mac":"00:1f:68:51:cb:59", "datos":[{ "interfa
JSON
Time Topic QoS
3321 10:47:06 /iotmach/actuadores/ 0
{ "paquete_id":"", "mac":"00:1f:68:51:cb:59", "datos":[{ "interfa
JSON

Anexo K. Prueba de alcance

PLAN DE PRUEBAS DE ALCANCE BLUETOOTH

Sistema Aplicación:	Gateway IOTMACH
Ciclo o Proceso:	Dispositivo Bluetooth
Objetivo:	Medir el alcance de la tarjeta Bluetooth integrada en el Gateway IOT.
Tipo de Prueba:	Técnica
Fecha:	17/junio/2016

Roles	**Responsable**	**Firma**
Proveedor-Tester	Jorge Prado	Jorge Prado
Tester	Ing. Dixys Hernández	Ing. Dixys Hernández
Tester	Gustavo Belduma	Gustavo Belduma
Tester	David Aguilera	David Aguilera

- Prueba de alcance a campo abierto.

Tipo de Dispositivo	Alcance en metros	Observaciones
BLE-Eddystone	60m	
BLE-MQTT	70m	

- Observaciones

Anexo L. Plan de prueba para la interfaz de usuario

PLAN DE PRUEBAS DE USABILIDAD

Sistema Aplicación:	Gateway IOTMACH
Ciclo o Proceso:	Interfaz de Usuario.
Objetivo:	Conocer si la interfaz es amigable para los usuarios.
Tipo de Prueba:	Funcional
Tiempo Estimado:	20 minutos
Fecha:	

Roles	**Responsable**	**Firma**
Proveedor	Jorge Prado	
Usuario Final		

- **Requerimientos previos a la prueba:**

Conocimientos básicos en computación.

- **Descripción de la Prueba:**

Conocer si la interfaz gráfica de usuarios es amigable.

- **Plan de Pruebas:**

Nº	**Descripción**	**Observación**	**Cumple**	
			Si	**No**
1	**Compatibilidad del sistema con el contexto y lenguaje del usuario**			
	¿Los términos usados en el sistema para describir funciones o secciones indican de forma clara lo que representan?			
	¿El sistema usa un lenguaje fácil de comprender?			
	¿La información que muestra el sistema aparece ordenada y entendible?			
2	**Consistencia y patrones**			
	¿La localización de los diferentes elementos de la interfaz (encabezamientos, y pie de página) son			

	mantenidos de forman consistente en todo el sistema?			
	¿Los formatos de presentación de información, estilos de fuente, colores, etc. Son usados de forma consistente y estandarizada en todo el sistema?			
	¿La iconografía permite visualizar rápidamente las acciones?			
3	**Aspectos visuales, estética, facilidad de lectura y diseño**			
	¿El aspecto visual del sitio es atractivo?			
	¿Los colores usados en el sitio son armónicos?			
4	**Control y libertad del usuario**			
	¿El usuario puede controlar los procesos de forma conveniente y de acuerdo con sus necesidades e intereses?			
	¿Las funcionalidades o aplicaciones externas son ejecutadas siempre a partir de la iniciativa o el consentimiento del usuario?			
5	**Prevención de errores y recuperación**			
	¿El sistema minimiza la ocurrencia de errores a través de mensajes de confirmación u otra forma?			
	¿Los mensajes de error son significativos identificando el problema ocurrido?			

- **Observaciones al plan de Prueba:**

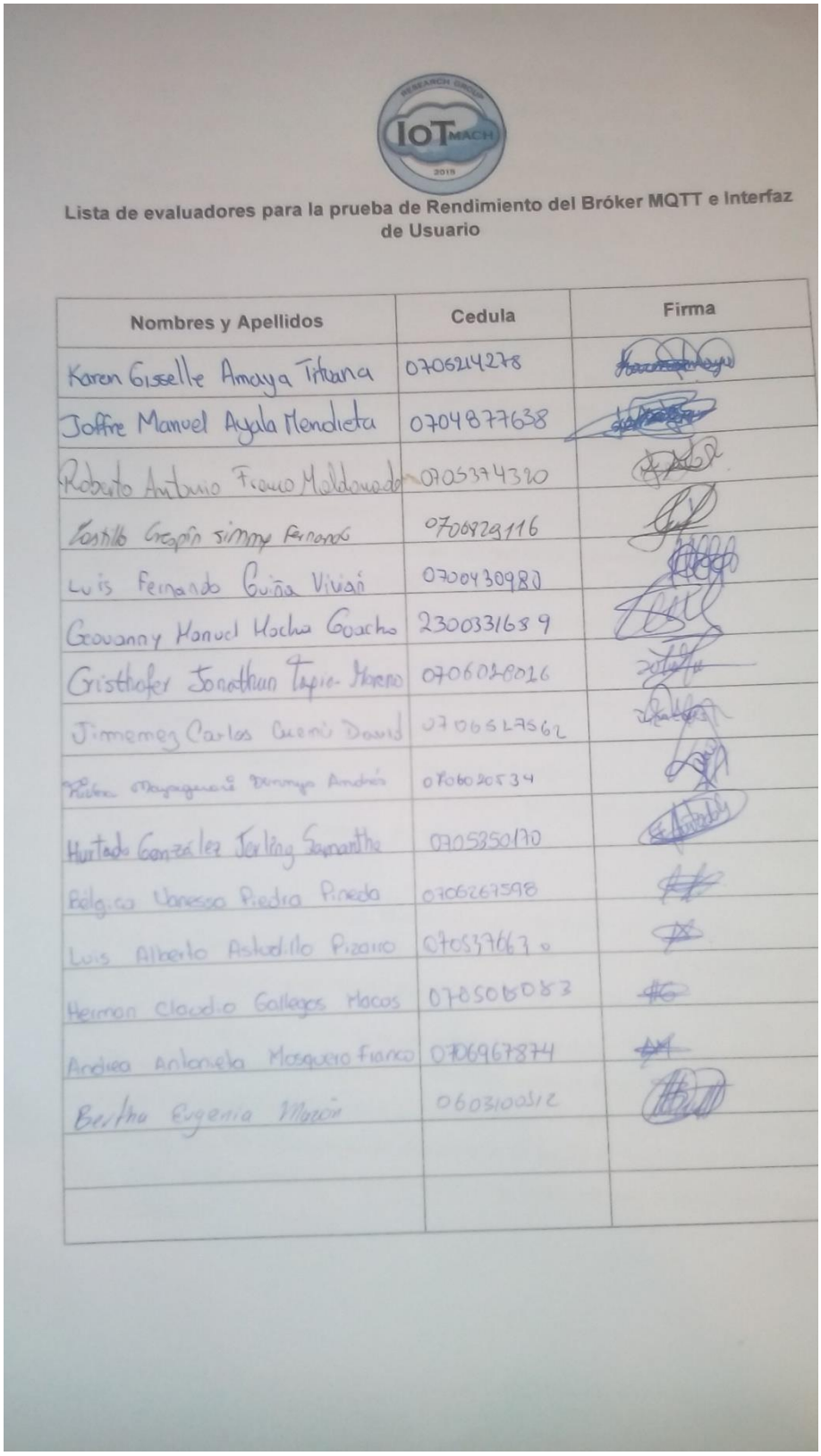
Tester	Aceptación	Fecha
_____ Nombres y Apellidos	_____ Firma	_____ Cedula

Anexo M. Prueba de integración

Responsables	- Jorge Prado - Kevin Valarezo	**Fecha**	29/07/2016
Requisitos Previos	Estar conectados a una red común Manejar tópicos comunes entre las aplicaciones		
Escenario	Conexión entre la aplicación Gateway con el Puente de Protocolos Brdige IOTMACH, enviando los datos emitidos por la WSN hacia el Centro de Procesamiento de Datos. En el cuál el gateway utiliza el protocolo MQTT y Eddystone para la recolección de los datos enviados por los motes y los envía hacia el Bridge IOTMACH y viceversa.		
Caso de Prueba	Conectividad e integracion entre el Gateway y Bridge IOTMACH		

Criterios	**Descripción**	**Componentes**				**Observación**
		Gateway		**IOTMACH**		
		SI	**NO**	**SI**	**NO**	
Comunicación	Existe comunicación entre las dos aplicaciones	X		X		
Integridad de la Información	Los paquetes fueron enviados y recibidos completamente	X		X		
Tiempo de Respuesta	Tiempo de envío/recepción de los datos es aceptable	X		X		
Información Correcta	Los datos contienen la información correcta	X		X		
Base de datos	Se presentaron problemas con la conexión a la base de datos.		X		X	

Anexo N. Lista de personas encuestadas para la prueba de usabilidad



Lista de evaluadores para la prueba de Rendimiento del Bróker MQTT e Interfaz de Usuario

Nombres y Apellidos	Cedula	Firma
Karen Giselle Amaya Tituana	0706214278	
Joffre Manuel Ayala Mendieta	0704877638	
Roberto Antonio Franco Maldonado	0705374320	
Castillo Crespín Jimmy Fernando	0708729116	
Luis Fernando Guiña Viviñ	0700430980	
Geovanny Manuel Mocha Guacho	2300331689	
Gristhofer Jonathan Tapia Moreno	0706028016	
Jimenez Carlos Geovanni David	0706617562	
Molina Mayorga Veronica Andrea	0706020534	
Hurtado González Jerling Samantha	0705350170	
Pólgica Vanessa Piedra Pineda	0706267598	
Luis Alberto Astudillo Pizarro	0705376630	
Hernan Claudio Gallegos Macas	0705065083	
Andrea Antonela Mosquera Franco	0706967874	
Bertha Eugenia Marín	0603100512	