UCLV
Universidad Central
"Marta Abreu" de Las Villas

FIE
Facultad de
Ingeniería Eléctrica

Departamento de Automática y Sistemas Computacionales

TRABAJO DE DIPLOMA

Título: Diseño de una aplicación IoT para casas de cultivo.

Autor: Arturo Javier Cárdenas Rivero

Tutores: Dr. C. Iván Santana Ching

Ms. C. Richar Sosa López

Ing. Jorge Armando Portal Díaz

Santa Clara, junio 2019

UCLV
Universidad Central
"Marta Abreu" de Las Villas

FIE
Facultad de
Ingeniería Eléctrica

Department of Automatics and Computational Systems

TRABAJO DE DIPLOMA

Title: IoT application design for monitoring greenhouses.

Author: Arturo Javier Cárdenas Rivero

Thesis Director: Dr. C. Iván Santana Ching

Ms. C. Richar Sosa López

Ing. Jorge Armando Portal Díaz

Santa Clara, june 2019

PENSAMIENTO

Uno de los grandes descubrimientos que un hombre puede hacer, una de sus grandes sorpresas, es encontrar que puede hacer lo que temía que no podía hacer.

Henry Ford

DEDICATORIA

A mi madre, porque este era su gran anhelo y todos mis logros son gracias a ella.

A mi abuela Amelia, por brindarme todo su amor.

A mi padre, por confiar en mí.

AGRADECIMIENTOS

A mi madre, por inculcarme el interés por el estudio, por los inmensos valores transmitidos, por los consejos tan necesarios, por convertirme en un hombre de bien, por estar ahí cuando más lo necesitaba, por indicarme el camino correcto, por ser mi mejor amiga, por todo el amor brindado. A mi madre, que le debo la vida y todos mis logros, más que un agradecimiento, el compromiso de que allá en donde esté, siempre la haré sentir orgullosa.

A mi padre, por el apoyo incondicional y el constante incentivo a la superación, porque sin él, este momento no sería posible.

A mi hermana, porque al seguir mi ejemplo, me impulsa a mejorar cada día.

A mi abuela Amelia, por querer siempre lo mejor para mí y porque su cariño me daba fuerza para seguir adelante.

A mi abuelo Chiqui, mis tíos Rafe, Lourdes, Isa y Osvaldo, porque siempre están ahí cuando más los necesito.

A mis primos Lester y Yasmani, por ser ejemplos a seguir.

A mis hermanos Yoandy y Jorgito, a mi tía Ismarys y a mi abuela Ana.

A Rachel, por entenderme, quererme y apoyarme.

A mis grandes amigos en las buenas y en las malas, Daniel, Rafael y Hermes.

A mis compañeros del aula, Eduardo, Xavier, Javier (El Mono), Carlos, Luis Alberto, Ariel, Luis Rogelio, Ernesto, Leonardo, Aldo, Tony y otros que no menciono, pero los llevo en el corazón.

A los profesores del departamento de Automática por todos los conocimientos transmitidos.

A mis tutores Ching, Richar y Jorge Armando.

A todos muchas gracias.

RESUMEN

Las tecnologías más empleadas para la supervisión de parámetros ambientales relacionados con la agricultura, implementan Redes de Sensores Inalámbricos, plataformas de Internet de las Cosas, protocolos de comunicación ligeros y sistemas relacionales o no relacionales para el almacenamiento de la información. Las casas de cultivo de la UEB de Cultivos Protegidos "Valle del Yabú" carecen de un sistema de supervisión de variables ambientales que contribuya a una correcta toma de decisiones de los expertos. Por ello, se propone una aplicación de Internet de las Cosas para la visualización y almacenamiento de los datos. En la presente investigación se selecciona la plataforma IoT ThingsBoard al soportar implementaciones locales en dispositivos con recursos limitados sin necesidad de conexión a internet. El protocolo de comunicación MQTT es utilizado para el envío de la telemetría debido a su reducida carga computacional, sus niveles de calidad del servicio y al desacople que se produce entre el cliente y el servidor. El despliegue de la aplicación IoT diseñada permitió la visualización de parámetros ambientales como la temperatura, humedad e intensidad luminosa, así como la consulta de los valores históricos almacenados.

Palabras Clave: Internet de las Cosas, Agricultura de Precisión, supervisión local, MQTT, Redes de Sensores Inalámbricos.

ABSTRACT

The most commonly used technologies for monitoring environmental parameters related to agriculture, implement Wireless Sensor Networks, Internet of Things platforms, light communication protocols and relational or non-relational systems for the storage of information. The greenhouses of the UEB Cultivos Protegidos "Valle del Yabú" lack of a supervision system of environmental variables that contributes to a right decision making by the experts. For this reason, an Internet of Things application is proposed for the visualization and storage of the data. In the present investigation, the IoT ThingsBoard platform is selected for supporting local implementations in devices with limited resources without needing an internet connection. The MQTT communication protocol is used to send telemetry due to its reduced computational load, its levels of service quality and the decoupling that occurs between the client and the server. The deployment of the IoT application designed allowed the visualization of environmental parameters such as temperature, humidity and light intensity, as well as the query of stored historical values.

Keywords: Internet of Things, Precision Agriculture, local supervision, MQTT, Wireless Sensor Networks.

TABLA DE CONTENIDOS

LISTA DE ABREVIATURAS

6LoWPAN	IPv6 sobre Redes de Área Personal Inalámbricas de Baja Potencia (*IPv6 over Low power Wireless Personal Area Networks)*
AMQP	Protocolo Avanzado de Cola de Mensajes *(Advanced Message Queuing Protocol)*
AP	Agricultura de Precisión
API	Interfaz de Programación de Aplicaciones (*Application Programming Interface)*
CoAP	Protocolo Aplicación Restringido (*Constrained Application Protocol*)
DTLS	Seguridad de Capa de Transporte de Datagramas (*Datagram Transport Layer Security*)
GPRS	Servicio General de Paquetes vía Radio (*General Packet Radio Service)*
GSM	Sistema Global para las comunicaciones Móviles (*Global System for Mobile communications)*
HTTP	Protocolo de Transferencia de Hipertexto (*Hypertext Transfer Protocol*)
IEEE	Instituto de Ingeniería Eléctrica y Electrónica (*Institute of Electrical and Electronics Engineers*)
IIC	Consorcio de Internet Industrial (*Industrial Internet Consortium*)
IIoT	Internet de las Cosas Industrial (*Industrial Internet of Things*)
IIRA	Arquitectura de Referencia de Internet Industrial (*Industrial Internet Reference Architecture*)

IoT	Internet de las Cosas (*Internet of Things*)
IP	Protocolo de Internet (*Internet Protocol*)
IPv6	Protocolo de Internet versión 6 (*Internet Protocol version 6*)
JSON	Notación de Objetos JavaScript (*JavaScript Object Notation*)
LAN	Red de Área Local (*Local Area Network*)
LiPo	Polímero de Litio
LPWAN	Redes de baja potencia y área amplia (*Low Power Wide Area Network*)
LTE	Evolución a Largo Plazo (*Long Term Evolution*)
M2M	Máquina a Máquina (*Machine to Machine*)
MQTT	*Transporte de Telemetría por Cola de Mensajes (Message Queue Telemetry Transport)*
NoSQL	No solo SQL
OPC-UA	Incrustación y Enlazado de Objetos para Control de Procesos - Arquitectura Unificada (*Object Linking and Embedding (OLE), for Procces Control – Unified Architecture*)
QoS	Calidad de Servicio (Q*uality of Service)*
RAM	Memoria de Acceso Aleatorio (Random Access Memory)
REST	Transferencia de Estado Representacional (R*epresentational State Transfer*)
RPC	Llamada a Procesamiento Remoto (*Remote Procedure Call*)
RPL	Protocolo de Ruteo para Redes de Baja Potencia y con Pérdida (*Routing Protocol for Low-power and Lossy Networks*)
RSMB	Bróker de Mensajes Realmente Pequeño (*Really Small Message Broker*)
SQL	Lenguaje de Consulta Estructurado (*Structured Query Language*).
SSL	Capa de Puertos Seguros (*Secure Sockets Layer*)

TCP	Protocolo de Control de Transmisión (*Transmission Control Protocol***)**
TDMA	Acceso Múltiple por División de Tiempo (*Time Division Multiple Access*)
TLS	Seguridad de la Capa de Transporte (*Transport Layer Security*)
UDP	Protocolo de Datagrama de Usuario (*User Datagram Protocol*)
UEB	Unidad Empresarial Base
UIT	Unión Internacional de Telecomunicaciones
URI	Identificador de Recursos Uniformes (*Universal Resource Identifier*)
WSN	Redes de Sensores Inalámbricos (*Wireless Sensor Network*)
XML	Lenguaje de Marcado Extensible (*eXtensible Markup Language*)

INTRODUCCIÓN

El desarrollo científico y tecnológico, unido a los avances en las esferas de la electrónica y las comunicaciones, ha permitido que el hombre sea capaz de comunicarse remotamente con personas u objetos presentes en ramas de la economía y la sociedad. En consecuencia, el Internet ha evolucionado gradualmente, aumentando la velocidad, capacidad, recursos y servicios disponibles en la nube. Uno de sus conceptos modernos, denominado Internet de las Cosas (IoT, por sus siglas en inglés), hace posible la aplicación real de las características anteriores, permitiendo el monitoreo y control de prácticamente todo lo que tenga capacidad de conectarse a una red (Hernández Rojas, Mazon Olivo y Campoverde Marca, 2015). El IoT tiene el propósito de proveer una infraestructura de tecnologías de la información que facilite el intercambio de bienes y servicios entre las "cosas", de una manera segura y confiable. Su función es superar la brecha entre los objetos en el mundo físico y su representación en los sistemas de información (Weber y Weber, 2010).

La aplicación de nuevas tecnologías en la agricultura convencional, ha introducido el concepto de Agricultura de Precisión, donde el usuario puede acceder a parámetros relacionados con su finca y controlarlos de manera manual o automática (Mat *et al.*, 2017). Con la AP se capturan, almacenan, analizan y visualizan datos vinculados con el comportamiento del suelo, del cultivo, la conducta de animales y el estado de maquinarias agrícolas o tanques de almacenamiento. El mercado de los servicios de AP fue valorado en 4.42 mil millones de dólares en 2017 y se proyecta alcanzar 9.53 mil millones de dólares para 2023, con una tasa de crecimiento anual del 13. 38%. De esta manera, se da respuesta a las necesidades de alimento de una población en constante crecimiento, se logra una producción de cultivos con recursos limitados, se combaten las enfermedades epidémicas

mediante la aplicación adecuada de fertilizantes y plaguicidas, y se potencia el sector agrícola al generar productos más saludables (MarketsandMarkets.com, 2018).

En el contexto de la AP, las Redes de Sensores Inalámbricos (WSN, por sus siglas en inglés), juegan un papel fundamental, que radica en la posibilidad de ser desplegadas en zonas sin una infraestructura establecida. Los nodos sensores que estas redes proponen tienen como fuente de alimentación la energía solar o baterías, permitiendo un rápido despliegue sobre el campo. Además, son de bajo consumo energético, presentan una gran autonomía, son de bajo costo, poseen características de autoconfiguración de red y varios tipos de sensores se pueden integrar en un único nodo. Con las WSN se disminuyen los tiempos de instalación, configuración y mantenimiento, a la vez que aumentan los límites geográficos del área que se desea monitorear (Amondaray *et al.*, 2018).

Las plataformas IoT ofrecen una solución completa en cuanto a los servicios de control de dispositivos y de recolección, procesamiento y visualización de los datos. Generalmente residen en la nube, aunque pueden implementarse en servidores locales, especialmente en áreas remotas de difícil conectividad. Son conocidas como el *middleware*[1] de Internet de las Cosas, actuando como mediador entre la capa de *hardware* y la capa de aplicación. Las plataformas de *middleware* IoT pretenden simplificar la lectura de datos de todo tipo de fuentes (dispositivos físicos, entrada humana, datos en línea, etc.) y proporcionan funcionalidades básicas para filtrar, analizar y crear eventos basados en la información recibida (Prakash Jayaraman *et al.*, 2016).

Actualmente, los invernaderos o casas de cultivo están empezando a convertirse en un método común en los negocios de agricultura, pero es necesario equiparlos con modernas tecnologías para que sean incluidos en la nueva AP. En el caso de Cuba el país tiene la estrategia de implementar tecnologías de automatización innovadoras y de bajo costo, con el objetivo de avanzar hacia la independencia tecnológica y la modernización de las industrias. En consecuencia, el Grupo de Internet de las Cosas, Automatización e Inteligencia Artificial perteneciente a la Universidad Central “Marta Abreu” de Las Villas, a solicitud de la empresa CEDAI de Villa Clara, propone el desarrollo de una aplicación

[1] *Software* que permite a una aplicación interactuar o comunicarse con otras aplicaciones, paquetes de programas, redes, *hardware* y/o sistemas operativos.

IoT para las casas de cultivo de la UEB de Cultivos Protegidos "Valle del Yabú". Con dicha aplicación se dará cumplimiento a la tarea de establecer un sistema de monitoreo de variables ambientales como son la humedad, la temperatura, la incidencia de luz solar, entre otras, garantizando así la calidad de la cosecha y el ahorro de recursos.

Situación del problema:

En investigaciones previas del Grupo de Internet de las Cosas, Automatización e Inteligencia Artificial perteneciente a la Universidad Central "Marta Abreu" de Las Villas se diseñó la arquitectura de *hardware* y *software* de una WSN para la recopilación de parámetros ambientales en las casas de cultivo pertenecientes a la UEB de Cultivos Protegidos "Valle del Yabú". Sin embargo, no se aprovechan todos los servicios que pueden ofrecer las tecnologías actuales en cuanto a visualización, almacenamiento, procesamiento de datos, seguridad y configuración de la red para contribuir a una correcta toma de decisiones de los expertos. El **problema científico** que se presenta radica en la no supervisión de parámetros ambientales de interés en las casas de cultivo de la UEB de Cultivos Protegidos "Valle del Yabú".

Hipótesis:

El empleo de una plataforma IoT permite la implementación local de un sistema de monitoreo, sin necesidad de acceso a internet y con protocolos de comunicación orientados a dispositivos con capacidades de procesamiento y conectividad limitadas. Los valores de telemetría enviados por los nodos sensores se visualizan en una interfaz gráfica idónea para usuarios con poca experiencia en temas de IoT. Los paneles asociados a cada nodo sensor cuentan con indicadores (*widgets*) para cada variable, donde se detallan elementos de la medición como el valor, el tiempo y la fecha en que fue realizada. Además, se establecen gráficas de los valores históricos almacenados en una base de datos con el objetivo de evaluar el desempeño de los parámetros ambientales con el paso del tiempo y mejorar la toma de decisiones.

Con el desarrollo de esta investigación se dará cumplimiento a los siguientes objetivos:

Objetivo General: Diseñar una aplicación IoT para la supervisión de parámetros ambientales de interés en las casas de cultivo de la UEB de Cultivos Protegidos "Valle del Yabú" que permita una correcta toma de decisiones de los expertos.

Objetivos Específicos:

1. Seleccionar la plataforma IoT más adecuada para la aplicación que se propone.
2. Configurar la plataforma IoT seleccionada.
3. Valorar el funcionamiento de la aplicación IoT diseñada.

Organización del informe

El informe del trabajo se estructura en tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos. El capitulario aborda los siguientes temas:

CAPÍTULO I: Se realiza un análisis conceptual del IoT, se exponen diferentes modelos de referencia empleados en el mundo y se describen aplicaciones del IoT en la agricultura.

CAPÍTULO II: Se diseña la arquitectura de *software* de la aplicación IoT. Se describe la selección y configuración de la plataforma IoT, además de otras herramientas de *software* utilizadas.

CAPÍTULO III: Se valoran los resultados de pruebas de funcionamiento realizadas al sistema de supervisión de parámetros ambientales basado en tecnologías de IoT diseñado. Se incluye un análisis económico y medioambiental de la propuesta.

CAPÍTULO 1. APLICACIONES DE INTERNET DE LAS COSAS

Con el IoT los sensores incorporados en los objetos transmiten datos a través de una red hacia una plataforma local o en la nube para su almacenamiento y procesamiento. Esto permite resolver las necesidades de las personas, controlando y modificando el ambiente donde se desempeñan. En el presente capítulo se abordan elementos teóricos sobre el IoT, algunas propuestas en cuanto a sus modelos de referencia y se describen aplicaciones industriales, específicamente en la rama de la agricultura, resaltando las tecnologías y los métodos de implementación del IoT más usados en el mundo.

1.1 Internet de las Cosas

Para establecer una definición de Internet de las Cosas es necesario analizar sus características más relevantes, las cuales se describen en una publicación presentada por la iniciativa IEEE IoT en mayo del 2015 (Minerva, Biru y Domenico, 2015) y que se muestran a continuación:

- Interconexión de las “cosas”: La primera característica del IoT se deriva de su nombre. La palabra “cosas”, se refiere a cualquier objeto físico que sea relevante desde una perspectiva del usuario o aplicación.
- Conexión de las “cosas” a internet: Desde el nombre IoT, también se infiere que las “cosas” están conectadas a internet.
- Cosas únicamente identificables: Un sistema IoT está compuesto de cosas que tienen identificadores únicos.

- Ubiquidad: Es una característica importante de un sistema IoT, indicando una red que esté disponible dondequiera y en cualquier momento. En el contexto del IoT, el "dondequiera" se refiere principalmente al concepto de donde es necesario y el "en cualquier momento" se refiere de manera similar a cuando es necesario.
- Capacidad de sensado y actuación: En un sistema IoT existen sensores y actuadores que proporcionan la inteligencia a las "cosas".
- Inteligencia embebida: Objetos inteligentes y dinámicos, con funciones de inteligencia y conocimiento incorporados.
- Capacidad de comunicación interoperable: El sistema IoT tiene una capacidad de comunicación basada en protocolos de comunicación estándar e interoperables.
- Autoconfiguración: Debido a la heterogeneidad de los dispositivos IoT, estos deben administrarse, tanto en términos de su configuración de *software* y *hardware* como en la utilización de recursos (energía, ancho de banda de comunicación, etc.).
- Programabilidad: Las "cosas" de un sistema IoT tienen una característica de programación. En el nivel más simple, un dispositivo programable puede asumir una variedad de comportamientos ante un comando del usuario sin requerir cambios físicos.

En el artículo (Minerva, Biru y Domenico, 2015) se plantea que el alcance de un sistema IoT varía desde pequeños sistemas que contienen cosas singularmente identificables y pequeños sensores hasta sistemas que interconectan millones de cosas con capacidad para ofrecer servicios complejos. Por tanto, se establecen dos definiciones separadas de IoT para sistemas pequeños y sistemas complejos.

En sistemas pequeños de baja complejidad, el IoT es una red que conecta "cosas" que presentan identificadores únicos a internet. Las "cosas" tienen capacidades de sensado, actuación y potencialidades para la programación. Con el uso de la identificación única y el sensado, la información sobre la "cosa" puede ser obtenida y su estado puede ser modificado desde cualquier lugar, en cualquier momento, por cualquier cosa. En sistemas complejos, se puede establecer el IoT como una red compleja de autoconfiguración y adaptación que interconecta las "cosas" a internet con el uso de protocolos de comunicación estandarizados. Las cosas interconectadas tienen representación física o virtual en el mundo digital, capacidades de sensado o actuación, características de

programabilidad e identificadores únicos. La representación contiene información que incluye la identidad, estado, ubicación o cualquier información de negocios, social o privada que sea relevante. Las cosas ofrecen servicios, con o sin intervención humana, mediante la explotación de la identificación única, la captura de datos, la comunicación, y las capacidades de actuación. El servicio se explota mediante el uso de interfaces inteligentes y está disponible en cualquier lugar, en cualquier momento y para cualquier cosa, siempre que se tenga en consideración la seguridad.

1.1.1 Internet de las Cosas en la Industria

Con el surgimiento de sistemas ciber físicos que permiten altos niveles de automatización, fabricación y producciones personalizadas de bienes y servicios, unido a la flexibilidad que ofrecen las líneas de producción programables, configurables y controlables, surge el término de Industria 4.0, referido a la cuarta revolución de la producción industrial. Este concepto se basa en el despliegue y uso generalizado de recursos computacionales y de comunicación, principalmente en las últimas dos décadas, evidenciado con la aparición de los teléfonos inteligentes, los electrodomésticos inteligentes, ciudades inteligentes, fábricas inteligentes, etc. El concepto de fábrica inteligente se basa en que los sistemas de producción inteligente están interconectados en una jerarquía multinivel, para lograr un alto grado de automatización, flexibilidad, eficiencia, autonomía, seguridad y bajo costo. Los materiales y los recursos serán administrados e introducidos en el proceso eficientemente y la producción se gestionará en tiempo real, minimizando los recursos utilizados para los productos y las operaciones. La reconfiguración y reorganización de los procesos de producción, al igual que la personalización de los productos, será factible en tiempo real y con seguridad para la infraestructura y los operadores, minimizando el impacto ambiental.

El Internet de las Cosas Industrial (IIoT, por sus siglas en inglés), ha surgido como un concepto general de la aplicación del IoT al sector de la industria. La visión del IIoT incluye todos los aspectos de las operaciones industriales, centrándose no sólo en la eficiencia del proceso sino también en la gestión de activos, mantenimiento, etc. Aunque los conceptos básicos son los mismos, los parámetros que identifican el subconjunto de IIoT del IoT, son los fuertes requisitos para la operación continua y la seguridad, así como la tecnología operacional empleada en el sector industrial. Por ejemplo, un *smartwatch* o

reloj inteligente monitorea variables vinculadas con la salud de una persona, mientras que otro sistema de monitoreo puede ser el que involucra la temperatura de una caldera de vapor en determinada industria. En este último caso, la transmisión de datos, generación de eventos y la ejecución de comandos, requieren más exigencias en cuanto a calidad, para evitar accidentes e incluso pérdida de vidas humanas (Serpanos y Wolf, 2018).

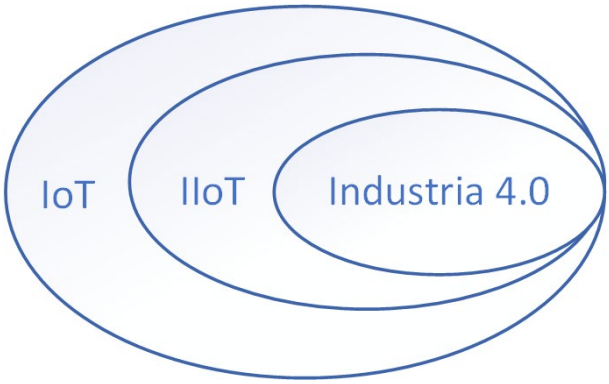


Figura 1.1. Relación entre IoT, IIoT e Industria 4.0 (Serpanos y Wolf, 2018).

1.2 Modelos de referencia de Internet de las Cosas

Como consecuencia del desarrollo de sistemas y servicios de IoT se necesitan establecer modelos de referencia que permitan operaciones eficientes y eficaces, teniendo en cuenta el gran número de dispositivos, sistemas ciber físicos, sistemas de comunicación y redes, y proveedores de servicios involucrados. En los últimos años se han realizado importantes esfuerzos para desarrollar estándares que deberán ser adoptados por los desarrolladores de aplicaciones de IoT.

1.2.1 Modelo de referencia de la Unión Internacional de Telecomunicaciones

La Unión Internacional de Telecomunicaciones introdujo en el 2012 un modelo de referencia para el IoT basado en cuatro capas jerárquicas, específicamente las capas de dispositivo, red, soporte de aplicaciones y servicios, y la capa de aplicación. Además, todas las capas presentan capacidades de gestión y seguridad. La capa de dispositivo es la más baja de la jerarquía e incluye la funcionalidad de dispositivos y *gateway*[2] de comunicación.

[2] Puerta de enlace. Dispositivo que actúa de interfaz de conexión entre aparatos o dispositivos

Contiene tecnologías de comunicación cableadas e inalámbricas como bus CAN, Wifi, Bluetooth, Zigbee, etc. y permite la conversión de protocolos para lograr interoperabilidad.

La capa de red permite el encapsulado de datos enviados por dispositivos y conversión de protocolos relacionados con la capa de red. Las capacidades de red, incluyen la funcionalidad de control para conectividad, movilidad, autenticación, autorización y contabilidad. Las capacidades de transporte suministran conectividad para el tráfico del usuario, así como el transporte de información de control y gestión para el servicio IoT y las aplicaciones.

La capa de soporte de servicios y aplicaciones incluye capacidades genéricas como procesamiento de datos, almacenamiento de información y capacidades especializadas para atender las necesidades de diversas aplicaciones. La capa más alta jerárquicamente es la de aplicación e incluye las aplicaciones y servicios IoT.

Las capacidades de gestión pueden ser genéricas o específicas. Las genéricas se refieren a la configuración de la red, topología, recursos, rendimiento, seguridad y administración de cuentas, mientras que las específicas se refieren a los requisitos de aplicación. Las capacidades de seguridad también pueden ser genéricas o específicas. En este caso, las genéricas se refieren a funciones relacionadas con la autorización, autenticación, integridad, confidencialidad en todas las capas, privacidad en la capa de aplicación, enrutamiento seguro en la capa de red, control de acceso en todas las capas, etc. Las capacidades de seguridad específicas se refieren a los requerimientos individuales de cada aplicación (UIT, 2012; Serpanos y Wolf, 2018).

Figura 1.2. Modelo de referencia del IoT propuesto por la UIT (Serpanos y Wolf, 2018).

1.2.2 Modelo de referencia de Cisco

En 2014 la empresa Cisco propuso un modelo de referencia para el IoT de siete niveles. Los tres niveles inferiores se consideran tecnología operativa. En el nivel más bajo se incluyen los dispositivos físicos y los controladores. Luego, están el nivel de conectividad y el nivel de *edge computing*[3] (procesamiento en el borde) donde se realiza adición de información, eliminación de duplicados y análisis de datos. Los cuatro niveles superiores se relacionan con las tecnologías de la información. El cuarto nivel es de almacenamiento, seguido por los niveles de abstracción de datos y el nivel de aplicaciones. El séptimo y último nivel es el de colaboración y procesos de negocios (Hassan y Rehman, 2018).

En la Figura 1.3 se ilustra el modelo de referencia de IoT propuesto en (Cisco, 2014) y sus niveles. Es importante señalar que en el IoT, los datos fluyen en ambas direcciones. En un patrón de control, la información fluye desde la parte superior del modelo (nivel 7) hasta la parte inferior (nivel 1). En un patrón de monitoreo, el flujo de información es inverso. En la mayoría de los sistemas, el flujo será bidireccional.

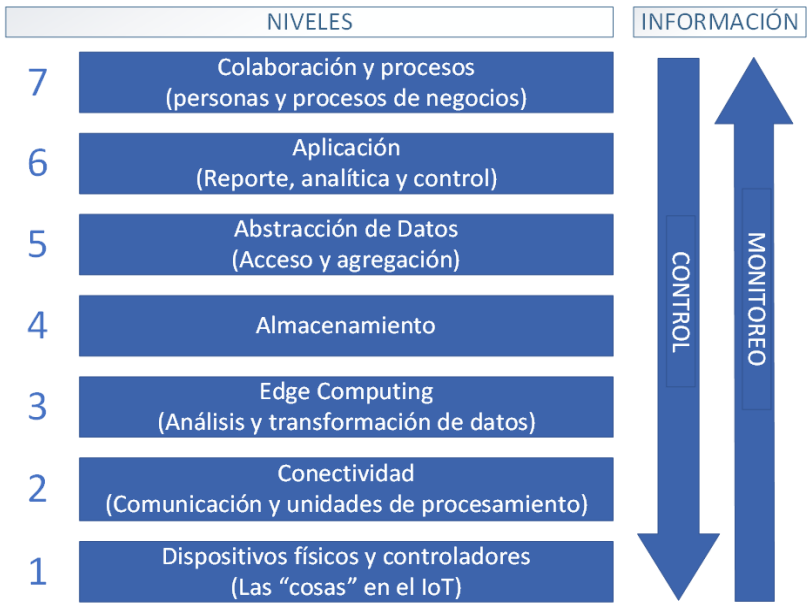


Figura 1.3. Modelo de referencia del IoT propuesto por Cisco (Hassan y Rehman, 2018).

[3] Los datos son procesados en dispositivos como enrutadores o *gateway* sin necesidad de acceder a la nube.

El nivel de dispositivos físicos y controladores está integrado por las "cosas" del IoT, e incluye una amplia gama de dispositivos que envían y reciben información. Los dispositivos son diversos, no hay reglas sobre tamaño, localización, factor de forma, u origen. Las capacidades básicas de un dispositivo son: conversión análoga-digital, generación de datos y que puedan ser consultados o controlados desde la red. El nivel de conectividad tiene como función principal la transmisión de información necesaria y confiable. Incluye la comunicación con dispositivos y entre ellos, entrega segura de los datos, implementación de varios protocolos, conmutación (*switching*) y enrutamiento (*routing*), traducción entre protocolos, seguridad a nivel de redes y análisis de redes (autoconfiguración).

La función del tercer nivel del modelo de referencia, *edge computing*, radica en la necesidad de convertir los flujos de datos de la red en información adecuada para su almacenamiento y procesamiento en los niveles superiores. Un principio básico del modelo de referencia de IoT que se presenta es que el sistema más inteligente inicia el procesamiento de la información tan pronto y tan cerca de la red como sea posible, lo que se refiere a veces como *fog computing* (procesamiento en la niebla). El *edge computing,* incluye filtrado, adición o eliminación de datos, inspección del contenido de paquetes de información, generación de eventos y combinación de análisis entre los niveles de datos y red. El nivel de almacenamiento determina si los datos son de interés para los niveles superiores, si se deben mantener en un disco duro o en memoria para su uso a corto plazo y determina si el sistema de archivos debe ser una base de datos relacional o no. Además, se encarga de organizar correctamente los datos según el sistema de almacenamiento y estos pueden ser combinados o modificados con información previamente guardada. Con este nivel, las aplicaciones acceden a los datos cuando es necesario, o sea, convierte el procesamiento de datos basados en eventos, a procesamiento basado en consultas.

El nivel de abstracción tiene como funciones fundamentales la renderización de datos y su almacenamiento, de forma tal que permita desarrollar aplicaciones simples con un rendimiento mejorado. Esta capa se justifica en el hecho de que, al existir varios dispositivos, puede ocurrir que sean demasiados datos para guardar en un solo lugar y, por tanto, se necesitan múltiples sistemas de almacenamiento. Con este fin se deben realizar un grupo de acciones como la creación de esquemas y vistas de datos según lo necesiten las

aplicaciones, se deben combinar datos de múltiples fuentes y se solucionan las diferencias en cuanto a formato de datos, semántica, protocolo de acceso y seguridad. En este nivel se filtran, seleccionan, proyectan y reformatean los datos para su utilización por las aplicaciones clientes. La interpretación de la información se produce en el nivel de aplicación. El *software* en este nivel interactúa con el nivel inferior y con los datos almacenados, por lo que no tiene que funcionar a velocidades de red. El modelo de referencia de IoT no define estrictamente una aplicación. Algunas aplicaciones se enfocarán en controlar los dispositivos y otras en monitorear o combinar los datos que provienen de los mismos. Los programas de monitoreo y control representan muchos modelos de aplicaciones, patrones de programación y pilas de *software*.

El último nivel del modelo es el de colaboración y procesos, y se destaca por incluir a las personas. Las aplicaciones ejecutan lógica empresarial para empoderar a las personas y estas utilizan los datos asociados para sus necesidades específicas. Sin embargo, en muchas ocasiones la acción necesaria requiere a más de una persona. La gente debe ser capaz de comunicarse y colaborar para que el IoT sea útil. La comunicación y la colaboración a menudo requieren múltiples pasos y por lo general trascienden a múltiples aplicaciones, por ello, el nivel 7 representa un nivel superior al de una sola aplicación.

1.2.3 Arquitectura de referencia del Consorcio de Internet Industrial

El Consorcio de Internet Industrial (IIC, por sus siglas en inglés) es un consorcio internacional abierto sin fines de lucro que establece el marco arquitectónico y la dirección para el internet industrial. Fue fundado por AT&T, Cisco, General Electric, IBM e Intel en marzo de 2014. En (IIC, 2017) se propuso una actualización para la Arquitectura de Referencia del Internet Industrial (IIRA, por sus siglas en inglés). Esta es una arquitectura abierta basada en estándares para sistemas IIoT que maximiza su valor al tener una amplia aplicabilidad en la industria, impulsando la interoperabilidad, tecnologías y desarrollos de estándares. La IIRA se basa en los puntos de vista que se definen analizando los distintos casos de uso del IIoT. Los puntos de vista son: negocios, uso de los sistemas, funcionalidad de los componentes e implementación de los mismos. El punto de vista de implementación describe la arquitectura general de un sistema IIoT, su estructura, la distribución de los componentes, y la topología por la cual están interconectados. Se realiza una descripción

técnica de los componentes, incluyendo interfaces, protocolos, comportamientos y otras propiedades.

Un patrón de arquitectura es una vista simplificada y abstraída de una implementación del sistema de IIoT que es recurrente, pero permite variantes. El patrón de arquitectura de tres niveles comprende los niveles *edge* (borde), plataforma y empresa. El nivel *edge* recopila datos de los nodos, utilizando la red de proximidad. Las características arquitectónicas de este nivel, incluyendo la distribución, ubicación, alcance y la naturaleza de la red de proximidad, varían dependiendo de los casos de uso específicos. El nivel de plataforma recibe, procesa y reenvía los comandos de control desde el nivel empresarial al nivel *edge*. Consolida procesos y analiza los flujos de datos desde el nivel *edge*, proporciona funciones de administración para dispositivos y activos, y ofrece servicios específicos como consulta y análisis de datos. El nivel empresarial implementa aplicaciones específicas, sistemas de soporte para la toma de decisiones y proporciona interfaces a los usuarios finales. El nivel empresarial recibe flujos de datos desde el nivel *edge* y de plataforma, y emite comandos de control hacia ellos.

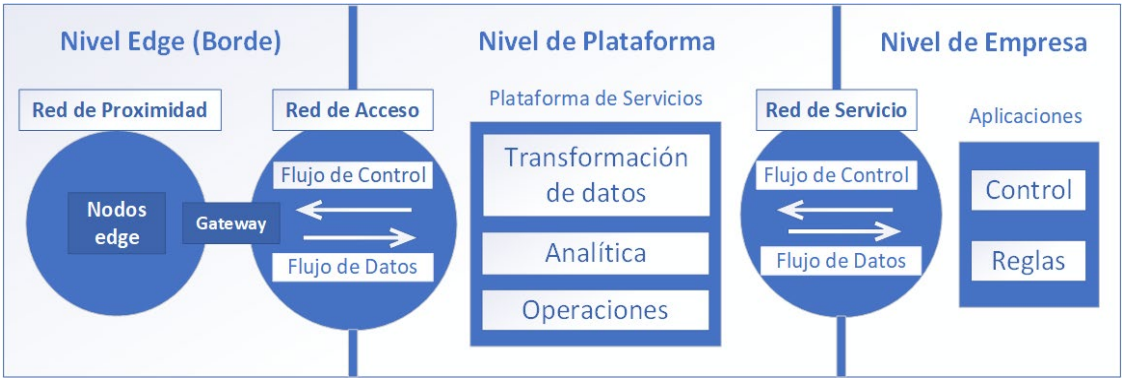


Figura 1.4. Arquitectura de referencia de tres niveles del IIoT propuesto en (IIC, 2017).

Los niveles descritos anteriormente se conectan a través de tres redes fundamentales. La red de proximidad conecta los sensores, actuadores, dispositivos, sistemas de control y activos, llamados colectivamente nodos *edge*. La red de acceso permite la conectividad para los flujos de datos y control entre los niveles *edge* y de plataforma. La red de servicios permite la conectividad entre los servicios en el nivel de plataforma y el nivel de empresa, y los servicios de cada uno de ellos.

1.3 Aplicaciones de Internet de las Cosas en la Agricultura de Precisión

La agricultura moderna se enfrenta a grandes desafíos relacionados con la construcción de un futuro sostenible, dificultado con el aumento poblacional, la urbanización, la contaminación ambiental, los cambios en los gustos alimentarios como resultado del envejecimiento poblacional, la migración, y el cambio climático. Una agricultura moderna debe utilizar las tecnologías y herramientas derivadas de las investigaciones y adelantos científicos. En este sentido, se pueden obtener grandes beneficios con el uso de redes de sensores ambientales, aplicaciones para monitorear el clima, automatización en la aplicación de fertilizantes y pesticidas, empleo de drones y de las plataformas en la nube para procesar los datos recogidos. Hay una amplia gama de aplicaciones posibles: gestión de maquinarias, vigilancia del ganado, piscicultura, cuidado forestal, cultivos urbanos, entre otras. Todas las tecnologías implicadas giran en torno al concepto del IoT y ayudan a los agricultores a través de sistemas de apoyo a la toma de decisiones (Cirani y Ferrari, 2019).

En (Quiñones Cuenca *et al.*, 2017), se diseñó un sistema de monitoreo de variables medioambientales usando una WSN y plataformas IoT. El sistema fue concebido como una alternativa de bajo costo comparado con estaciones meteorológicas convencionales y se basó en componentes de *hardware* y *software* libre. Los nodos se conformaron por módulos de sensores, procesamiento y comunicación. Para el procesamiento se usaron placas Arduino Uno (RedBoard DEV-12757) a las que se acoplan sensores de presión, temperatura, humedad, pluviómetros, anemómetros y fotorresistencias. Para la comunicación entre los nodos sensores y el nodo central, se propone el uso de wifi (2.4 GHz) en ambientes urbanos y el empleo del protocolo de comunicación DigiMesh (900MHz) para el intercambio de información a grandes distancias. El abastecimiento de energía se estableció por medio de paneles solares de 5.2 W, baterías LiPo de 3.7V y 6600mAh y cargadores solares de 450mA. El prototipo diseñado se evaluó en las plataformas IoT Ubidots, Phant y ThingSpeak, que funcionan con distintas estrategias y protocolos de comunicación como HTTP y MQTT.

Otro sistema de monitoreo ambiental para aplicaciones en la agricultura es propuesto en (Cao-hoang y Nguyen Duy, 2017), donde se escogen plataformas Arduino como *hardware* para los nodos sensores. La información recolectada es enviada al *gateway* vía wifi por la

banda de 2.4GHz, que pertenece a la especificación IEEE 802.11 b/g/n. Una Raspberry Pi funciona como *gateway* y es la responsable de almacenar, analizar y transmitir los datos a la nube. El bróker de mensajes[4] de código abierto Eclipse Mosquitto permite la utilización del protocolo de publicación/suscripción MQTT. La herramienta Node-RED provee un editor de flujo para programar el *gateway* de una manera sencilla, y está instalada por defecto en el sistema operativo Raspbian, ejecutado en la Raspberry Pi. La plataforma IBM Watson IoT administra, monitorea y visualiza los datos en tiempo real. Como sistema de alimentación de los nodos sensores se proponen baterías LiPo de 3.7V y 1200mAh acompañadas de paneles solares de 1W.

El sistema de monitoreo animal basado en tecnologías de IoT desarrollado en la Universidad de Aveiro, Portugal, presenta la plataforma SheepIT que recibe datos transmitidos por collares ubicados en los animales. Los datos se transmiten a una red compuesta por balizas fijas instaladas en las áreas de pastoreo. Para mantener el consumo de energía lo más bajo posible, el sistema sigue un mecanismo de TDMA. El *gateway* implementa una baliza (con mayor potencia) que se comunica con las otras, y conecta la red local a internet a través de una conexión de banda ancha (3G). En cuanto a la transmisión de los datos, el bróker RabbitMQ soporta varios protocolos de mensajería como AMQP o MQTT, basados en arquitecturas asincrónicas de publicación/suscripción y permite emplear mecanismos de seguridad, como los certificados SSL/TLS. La herramienta Apache Spark se encarga de la traducción de datos JSON, generación de alarmas y procesamiento de datos, incluyendo técnicas de aprendizaje automático (*machine learning)*. PostgreSQL ha sido elegida como base de datos ya que es adecuada para entornos con una gran cantidad de información, además de los mecanismos de seguridad e integridad que presenta. La plataforma incluye un marco de trabajo (*framework*) REST API para permitir el desarrollo web y la interacción con el usuario u otras plataformas relevantes (Nóbrega *et al.*, 2018).

En la Escuela Colombiana de Ingeniería, se realizó una extensión de la plataforma IoT de código abierto ThingsBoard para la detección de enfermedades que afectan a los cultivos. Se empleó la herramienta de análisis de datos Apache Spark que utiliza una estructura de

[4] Patrón arquitectónico para la validación, la transformación y el ruteo de mensajes.

datos que facilita los cálculos en paralelo y presenta una arquitectura escalable. Para el almacenamiento de la información que maneja la plataforma, se emplea la base de datos NoSQL Apache Cassandra, mientras que para la indexación geoespacial (almacenamiento y consulta de datos geográficos) se utiliza la base de datos NoSQL MongoDB. Los dispositivos creados en ThingsBoard reciben los datos (temperatura, humedad, luminosidad, etc.) de los sensores según el protocolo de comunicación MQTT que presenta un patrón de mensajes publicador/suscriptor. La herramienta Apache Spark revisa los mensajes recibidos, evalúa las reglas definidas por el usuario para la detección de una probabilidad de enfermedad y dispara las acciones asociadas a cada regla (Cadavid *et al.*, 2018; Ramírez Otero, López Pacheco y Mendivelso Sanabria, 2018).

En el artículo (Guerrero Ibañez *et al.*, 2017) se presenta SGreenH-IoT como una plataforma IoT de bajo costo y consumo energético para la monitorización de campos de cultivos e invernaderos. El modelo de topología está basado en clústeres, que son los encargados de recolectar la información en una determinada zona de cultivo. Cada clúster está formado por un conjunto de nodos sensores y un coordinador para regular los tiempos de monitoreo. Para la comunicación de los nodos sensores con los nodos coordinadores se empleó el protocolo ZigBee apoyado en un módulo de radio XBee PRO S2C. La información recogida por los nodos coordinadores se envía a la nube a través de conexiones de red celular 3G o 4G. Los nodos sensores están conformados por un microcontrolador PIC18LF46K22 de Microchip™ y la placa desarrollada tiene capacidad para 4 tipos de sensores, entre los que se encuentran sensores de humedad y temperatura. Como fuente de energía se utilizaron baterías de iones de litio con protección contra descargas y sobrecargas para evitar daños en los circuitos, que suministran 3.7V y 2000mAh de corriente directa. Los nodos coordinadores están formados por Raspberry Pi 3 Modelo B+ con módulos XBee coordinadores. La interfaz gráfica de la plataforma permite consultar los parámetros monitoreados y configurar los sensores de acuerdo las necesidades de cada usuario.

En (Beltrán Martínez y Ortiz Barajas, 2018) los autores realizan la configuración y puesta a punto de un sistema de monitoreo basado en tecnologías de IoT. Una WSN con topología de malla (*mesh*) posibilitó el enrutamiento de los nodos hacia el nodo coordinador en caso

de alguna falla. Los nodos sensores son dispositivos Zolertia RE-Mote que ejecutan el sistema operativo Contiki y soportan el protocolo de comunicación 6LoWPAN para el uso de IPv6 sobre radiofrecuencia. Contiki ofrece la posibilidad de comprobar su funcionamiento a través de tres plataformas de simulación antes de ser ejecutado sobre el *hardware* final, las cuales son Cooja, MSPsim y NetSim. El protocolo de mensajería MQTT es utilizado para la transmisión de los datos entre la WSN y la plataforma IoT. Una Raspberry Pi 3 Modelo B contiene el bróker de mensajes MQTT Mosquitto y cumple con la función de enrutador de frontera (*border router*). Se realizan pruebas en la plataforma IoT Kaa, almacenando la información en la base de datos Apache Cassandra y visualizando los datos con la herramienta Apache Zeppelin.

Los autores del artículo (Prakash Jayaraman *et al.*, 2016) proponen la plataforma SmartFarmNet para aplicaciones de cultivo inteligente que permite la integración de cualquier dispositivo IoT. Soporta analítica de datos escalable para procesar continuamente grandes cantidades de datos de cultivos, y proporciona herramientas para el almacenamiento que pueden hacer frente a la enorme cantidad de información (*Big Data*). Los datos generados a partir de dispositivos se procesan inicialmente en los *gateway* locales, que se comunican tanto con los sensores como con la plataforma SmartFarmNet en la nube. Dichos *gateway* también permiten procesar los datos del sensor más cerca de la fuente, permitiendo un mejor uso del ancho de banda en la comunicación con sitios remotos. Para el almacenamiento de datos emplea una combinación de bases de datos no SQL (NoSQL) y Semantic, ya que optimiza el procesamiento para diferentes escenarios. También proporciona interfaces para consultas de usuario y para la visualización de datos o resultados de análisis.

En la Conferencia de la IEEE sobre Sistemas Abiertos desarrollada en el 2016 en Malasia, se presentó un Sistema de Gestión de Invernaderos basado en tecnología de WSN, que monitorea la humedad del suelo, la humedad del aire y la temperatura. La arquitectura de los nodos presenta una unidad de procesamiento, un módulo de comunicaciones con transceptor inalámbrico, módulo de sensores y un módulo de suministro eléctrico. Los datos se transmiten al *gateway* utilizando tecnología inalámbrica basada en XBee y este envía los datos al sistema central a través de un módulo wifi o GSM, que se encarga de

almacenar los datos en la nube y mostrarlos a los usuarios a partir de una interfaz gráfica. La irrigación con método automático mantiene la condición de humedad del suelo inferior al valor umbral y optimiza el uso de agua y fertilizantes(Mat *et al.*, 2017).

En el artículo (Foughali, Fathallah y Frihida, 2018) se presenta una aplicación IoT para la prevención de enfermedades que afectan los cultivos conformada por cinco nodos sensores Waspmote que se interconectan vía Zigbee utilizando módulos XBee basados en la interfaz de comunicación IEEE 802.15.4. La conexión con el *gateway* (recolecta datos de los nodos) se ejecuta con la tecnología mencionada anteriormente y está constituido por un enrutador (*router)* multiprotocolo Meshlium que soporta 5 interfaces (Wifi, Bluetooth, Zigbee y GPRS). La plataforma IoT que se implementa es Ubidots, que se encarga de analizar los datos obtenidos y de generar alertas a partir de un modelo de predicción programado en lenguaje C.

El sistema de monitoreo para variables agronómicas basado en una red de sensores inalámbricos presentado en (Caicedo Ortiz *et al.*, 2018) emplea Zolertias Z1 como nodos sensores y *gateway.* Estos dispositivos implementan el sistema operativo de código abierto Contiki, diseñado para sistemas de baja capacidad de procesamiento y consumo energético. Su arquitectura se basa en procesos, eventos, y uso de múltiples hilos. Las aplicaciones se desarrollan en lenguaje C. Se emplea el protocolo de ruteo para redes de bajo consumo RPL, que tiene en cuenta la eficiencia energética, se adapta a TCP/IP y facilita la implementación de sensores independientemente la posición de los nodos. El transceptor de radio frecuencia CC2420 empotrado en los *motes*[5] Z1 se encarga de la emisión y recepción de los datos, tiene una capacidad de transmisión de 250 Kb/s y una cobertura de 150m en 2.4GHz. En el diseño de la red inalámbrica, los nodos se colocaron equidistantes unos de otros, para lograr homogeneidad en la comunicación. Los datos se almacenan en un servidor MySQL usando PHP para proveer una interfaz gráfica. La aplicación web tiene control de acceso y visualización de datos de temperatura y humedad del suelo.

[5] Referido a los nodos. Traducción inglesa de “mota de polvo”, indicando el pequeño tamaño y la ubicación.

En (Ferrández Pastor, García Chamizo y Nieto Hidalgo, 2016) se desarrolla una plataforma IoT para el monitoreo de variables ambientales vinculadas al rendimiento de los cultivos. La arquitectura está basada en las capas de dispositivos, borde (*edge)*, de comunicación y de procesamiento en la nube. Las aplicaciones críticas y los procesos básicos de control se instalan en la capa *edge*, mientras que la interfaz de usuario, la analítica y almacenamiento de datos se instalan en la nube (internet o intranet). El protocolo MQTT propicia tiempos de respuesta pequeños e interoperabilidad entre los procesos de control. El bróker asociado a este protocolo permite operaciones de solicitud y respuesta basadas en HTTP o REST para el procesamiento y almacenamiento de datos en la nube. El nivel uno del servicio de calidad (QoS) del bróker MQTT garantiza que le información sea recibida por el destinatario. Se proponen tres tipos de nodos sensores, uno de ellos compuesto por un dispositivo Photon IoT y los otros están formados por Raspberry Pi, a los cuales se les incorporan los sensores correspondientes. La plataforma IoT Ubidots permite la comunicación mediante API REST entre los clientes y el servidor IoT. El lenguaje Python es usado para los algoritmos de control y los datos son almacenados en formato CSV[6].

En el Simposio de la IEEE sobre Informática y Comunicaciones desarrollado en el 2017, se dio a conocer una propuesta ligera y segura para comunicaciones industriales, que se puede aplicar a las redes inalámbricas vinculadas al IIoT. El artículo resalta las potencialidades del protocolo de mensajería MQTT, segundo más usado en aplicaciones de IoT por detrás del HTTP que no se ajusta a dispositivos con recursos limitados y comunicación ligera. MQTT es un protocolo asincrónico y cuenta con un servicio de calidad (QoS) de 3 niveles. Los *motes* están conformados por dispositivos Zolertia Z1, que funcionan con el sistema operativo Contiki, especialmente diseñado para aplicaciones de IoT. Este sistema operativo trae incorporado el ambiente de simulación de red Cooja para comprobar el funcionamiento de aplicaciones diseñadas. Como enrutador de frontera (*border router*) se usa 6LBR, que permite la interconexión entre redes 6LoWPAN/IEEE 802.15.4 con redes locales IPv6. El enrutador de frontera es instalado en una Rasberry Pi a la cual se conecta mediante USB el Zolertia Z1 receptor de los datos. El bróker MQTT Mosquitto es seleccionado en la red WSN descrita (Katsikeas *et al.*, 2017).

[6] Documento en formato abierto sencillo para representar datos en forma de tabla.

Otra aplicación IoT vinculada a la industria agropecuaria es mostrada en (Zamora Izquierdo, Martínez y Skarmeta, 2018), donde se implementa una plataforma IoT para agricultura inteligente basada en procesamiento en la nube y en el borde (*edge computing*). La plataforma FIWARE provee los módulos de *software* y el bróker ORION permite el acceso a los datos almacenados en ella por medio de los protocolos MQTT y CoAP. La arquitectura del sistema presenta tres capas principales, la de sistemas ciber físicos (sensores), *edge computing* y la capa de análisis de datos y administración inteligente en la nube. En los canales cableados se usan conexiones serie industriales (RS485) y en las comunicaciones inalámbricas se utiliza 6LowPAN. Los módulos de control (capa de *edge computing*) están constituidos por controladores IPex16 de Odin Solutions (OdinS), CPU de 32 bits, 4 MB de memoria expandible, y 16 puertos de entrada/salida que pueden ser configurados por *software*. Estos son virtualizados (se crean máquinas virtuales) en una PC Intel Core i5-7400, 16GB de RAM y 1TB de disco duro que se encarga de la instancia local del bróker, almacenamiento de datos y ejecución de agentes IoT. El servidor en la nube (Intel Xeon, 32GB de RAM y 2 discos duros 500GB) alberga el bróker global, ejecuta el procesamiento *Big Data* y se encarga del hospedaje de los servicios web.

En (Guerrero Nicolas y Cano Rigol, 2017) los autores desarrollan una red de sensores de bajo consumo para el monitoreo de variables en diversos escenarios. Los nodos sensores están constituidos por Zolertias Z1, dispositivos con chip TI MSP430F2617 de 16-bit, de ultra bajo consumo, conectividad de radio en el estándar IEEE 802.15.4 y estándar de comunicación 6LowPAN para comunicarse con dispositivos IP. Los *motes* Z1 se alimentan por medio de dos baterías AA o AAA y el consumo es de 365µA en activo, 0.5µA en standby y 0.1µA cuando está apagado. El protocolo de transporte utilizado es UDP. Una Raspberry Pi 3 Modelo B es usada como *router* al instalarse en ella el *software* CETIC 6LBR que permite la interconexión de redes IP y 6LoWPAN. El servidor UDP es programado en lenguaje Python. La Raspberry Pi contiene la base de datos MySQL y un servidor web para mostrar los datos. Los nodos Zolertia trabajan con el sistema operativo Contiki y la programación se realiza en lenguaje C. Contiki permite el uso del protocolo de ruteo RPL, por lo que un nodo puede usar a otros nodos como repetidores para llegar a un determinado destino. Una aplicación Android también es desarrollada en lenguaje Java y XML para facilitar el acceso a los datos generados por cada *mote* Zolertia Z1.

En (Triana Useche y Rodríguez Leguizamo, 2018) los autores desarrollan un prototipo de solución IoT para el monitoreo de cultivos agrícolas implementando la tecnología LPWAN(LoRa), de bajo consumo energético y largo alcance (2 km a 5 km en ambiente urbano y 8 km a10 km en ambiente rural) entre los dispositivos emisor y receptor. Las variables objeto de estudio son temperatura, humedad, radiación solar y Ph. Los nodos sensores están constituidos por placas Arduino Uno con módulos *shield* LoRa acoplados. El *gateway* es un dispositivo Dagino LG-01S y se encarga de recibir la información de los nodos y de reenviarla al servidor en la red a través de una conexión TCP/IP segura. Los nodos se conectan al *gateway* usando tecnología RF LoRaWAN y forman una topología de red en estrella. El servidor web ThingSpeak es utilizado para almacenar la información en una base de datos relacional confeccionada con SQL Server, *software* que también se encarga del procesamiento de los datos. Una aplicación web confeccionada con el programa Visual Studio proporciona la interfaz de usuario.

En la Universidad de Osnabrück, Alemania, se realizó el diseño e implementación de un sistema de monitoreo agrícola con infraestructura basada en IoT. La conectividad a internet se establece a través de una red móvil pública, mediante un módem LTE conectado al *gateway* central. Los nodos sensores están constituidos por dispositivos TelosB (microcontrolador TI MSP430 a 8MHz y 10 kB de RAM) compatibles con el estándar de comunicación IEEE 802.15.4 y de bajo costo. Estos nodos se organizan en forma de clúster con nodos de referencia formados por Raspberry Pi que actúan como *gateway* de la WSN. Otra Raspberry Pi es empleada como *gateway* central con el modem LTE acoplado. El *software* de la plataforma de sensores es TinyOS, un sistema operativo de código abierto para dispositivos con recursos limitados. La alimentación de los sensores se efectúa con baterías AA dado que el consumo de 0.2mA en 100h de trabajo lo permite. El núcleo de la infraestructura de IoT es el bróker MQTT Mosquitto que se ejecuta en el servidor de internet. Un servidor Apache proporciona la interfaz gráfica de usuario, permite consultas a la base de datos y es responsable del análisis y visualización de los mismos (Bauer y Aschenbruck, 2018).

La propuesta de una red de sensores inalámbricos para las casas de cultivos protegidos “San José”, en Santiago de Cuba, presenta una topología de red tipo árbol (*cluster-tree*) que

tiene una tolerancia a fallos intermedia, gran escalabilidad, buena eficiencia energética y es adecuada para nodos alimentados con baterías. Como nodos sensores y estación base se implementan plataformas ēKo que operan en la banda de frecuencia de 2.4GHz, alimentados con baterías AA y paneles solares para alargar la vida de las mismas. La estación base se conecta a una computadora de oficina para recolectar los datos. Se concluye que con el uso de plataformas de *hardware* libre se reducen los costos de inversión significativamente (Amondaray *et al.*, 2018).

En la Universidad Central "Marta Abreu" de Las Villas se diseñó la arquitectura de *hardware* y *software* de una WSN para el monitoreo de variables ambientales en casas de cultivo. El sensor DHT22 es seleccionado para medir la humedad relativa y la temperatura, los módulos YL-69 y YL-38 son utilizados para medir la humedad del suelo, y el módulo de sensor fotosensible LM393 se utiliza en la detección de la intensidad luminosa. En los nodos sensores y en el nodo coordinador se implementan dispositivos Zolertia RE-mote, que cuentan con el chip de radio CC2538 integrado, es compatible con el estándar IEEE 802.15.4, son de bajo consumo energético y funcionan con baja tensión de alimentación. Además, soporta los protocolos IPv6, 6LoWPAN, RPL para redes de bajo consumo y ejecuta el sistema operativo Contiki, que cuenta con el simulador de red Cooja para facilitar tareas de desarrollo y depuración de errores. El *gateway* se implementó en una Raspberry Pi 3 Modelo B conectada vía Ethernet a un enrutador wifi. La computadora de placa reducida empleada (Raspberry Pi) posee un procesador ARM Cortex-A53 de 4 núcleos a 1.2GHz, conectividad Wifi, LAN y Bluethooth, 1GB de memoria RAM y conector para tarjetas SD. La base de datos seleccionada para el almacenamiento de la información fue MariaDB (Estevez Pérez, 2018).

1.4 Consideraciones finales del capítulo

El IoT aplicado al sector industrial puede reducir los costos y los riesgos de operación, así como incrementar la productividad. Estos beneficios no se ignoran en el sector de la agricultura, evidenciado en la amplia gama de aplicaciones de IoT que se han desarrollado en los últimos años, algunas de las cuales se han descrito en este capítulo. Cada empresa u organización que suministra servicios de IoT puede proponer un modelo de referencia para su arquitectura, sin embargo, todos incluyen dispositivos, gestión de datos, seguridad,

comunicación, infraestructura de procesamiento y visualización de los resultados. En cuanto a las tecnologías que se aplican en la AP, las WSN juegan un papel fundamental en la recolección de información desde sitios remotos, por medio de dispositivos con bajo consumo energético y que soportan protocolos de comunicación ligeros. Las aplicaciones que incorporan almacenamiento y procesamiento de datos generalmente emplean plataformas IoT, las cuales proporcionan interfaces de usuario y comunicación con otros servicios hospedados en la nube. En el próximo capítulo se procederá a la selección y configuración de los elementos de *software* que serán implementados en la aplicación IoT propuesta.

CAPÍTULO 2. DISEÑO DE LA APLICACIÓN IOT

Una correcta selección y configuración del *software* que conformará la aplicación IoT propuesta, es crucial para cumplir con estándares de calidad que garanticen la satisfacción de los clientes. Para ello se sigue una metodología, donde el primer paso es la selección de un modelo de referencia que identifique cada uno de los componentes de la aplicación IoT y los agrupe en capas o niveles. Luego se escoge una plataforma IoT entre diversas plataformas de desarrollo disponibles, se determina el protocolo de comunicación más adecuado para dispositivos con limitaciones de recursos energéticos, procesamiento y almacenamiento, y se selecciona el tipo de base de datos para almacenar la información relevante. Posteriormente se definen los dispositivos que actuarán como nodos sensores o *gateway* y se diseña la arquitectura de la aplicación IoT. Por último, se configura cada elemento para garantizar su correcta integración con la plataforma IoT seleccionada.

2.1 Modelo de referencia de IoT empleado

En el capítulo anterior, se abordaron algunos modelos de referencia del IoT que muestran las diferentes capas o niveles que conforman una aplicación de Internet de las Cosas. En el presente trabajo, se ha tomado el modelo de referencia de Cisco descrito en el epígrafe 1.2.2 para desarrollar la aplicación que se propone. Debido al alcance de la investigación, el flujo de información solo será de monitoreo, o sea, que los datos se transmiten desde el nivel inferior al nivel superior. Además, el nivel de abstracción de datos no se encuentra funcional debido a que no se necesitan múltiples sistemas de almacenamiento ni la aplicación tiene un alcance corporativo, sin embargo, en futuros proyectos se podrían incorporar otros escenarios industriales que requieran las capacidades que brinda este nivel. Es importante señalar que este modelo de referencia incluye un nivel destinado a la toma de

decisiones de los expertos, pues las mismas pueden influir en el funcionamiento del sistema.

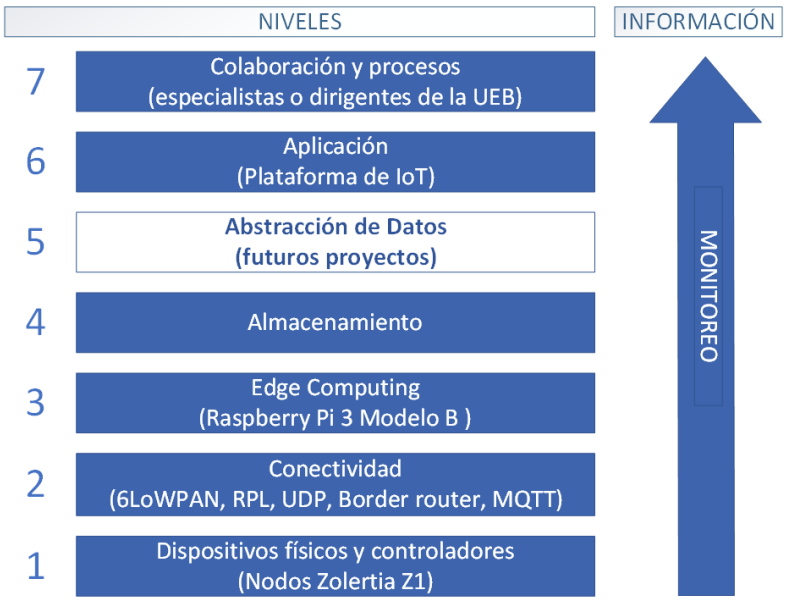


Figura 2.1. Modelo de referencia empleado en la aplicación IoT propuesta.

2.2 Selección de la plataforma IoT

Para seleccionar la plataforma IoT que se va a utilizar en la aplicación, se toma como referencia el artículo (Li, 2018) donde se exponen las características principales que debe cumplir el *middleware* IoT cuando se va a desplegar en un escenario industrial. En la tabla 2.1 se muestra una comparación entre algunas plataformas IoT de código abierto siguiendo los puntos que se describen a continuación:

- Acceso y gestión de dispositivos. La plataforma debe soportar varios tipos de *hardware* (Raspberry Pi, Arduino, BeagleBone, etc.) con el empleo de diferentes protocolos de acceso (ZigBee, CAN, RS-232, Bluetooth, GPIO, LoRA, SIGFOX, etc.) y debe proporcionar la gestión del estado del dispositivo, así como actualizaciones del *firmware*.
- Transmisión de datos en la capa de aplicación. Soporte para los protocolos de capa de aplicación HTTP, MQTT, WebSocket, Modbus, OPC-UA, CoAP y AMPP.

- Almacenamiento de datos y procesamiento inteligente. El almacenamiento puede ser local, en la nube o ambos. Las capacidades de procesamiento de datos incluyen procesamiento de eventos complejos, eventos basados en reglas y capacidades de aprendizaje automático.
- Disponibilidad de interfaces API y capacidades de soporte de aplicación. Las plataformas proporcionan aplicaciones a través de interfaces RESTful.
- Modo de implementación. Capacidad para implementar *gateway*, aplicaciones IoT o ambas.
- Seguridad. La plataforma necesita proporcionar encriptación de datos, transmisión segura, autenticación de identidad y mecanismos de control de acceso para satisfacer las necesidades de las aplicaciones IoT.
- Actividades de desarrollo. Utilización y soporte a nivel mundial de una plataforma IoT basado en los tópicos de foros y sitios web especializados.
- Tipo de *software*. Las plataformas pueden ser total o parcialmente gratuitas (*software* libre), de pago y de código abierto o privado.

Un análisis de la tabla 2.1 permite descartar las plataformas ThingSpeak, Kaa y Alljoyn ya que, a pesar de ser de código abierto, se necesita pagar una licencia para su uso luego de vencido el período de prueba. De las plataformas restantes, solo ThingsBoard, Macchina.io, WSO2IoT y SiteWhere soportan almacenamiento local, requisito indispensable para el despliegue de la aplicación en zonas de difícil acceso y conectividad. La plataforma Macchina.io no es seleccionada producto de sus limitaciones en cuanto al manejo del estado de los dispositivos y procesamiento inteligente. La plataforma SiteWhere no incorpora capacidades de *gateway* y tiene menos soporte por parte de la comunidad de desarrolladores que ThingsBoard y WSO2IoT.

Las plataformas IoT ThingsBoard y WSO2IoT soportan gestión de un gran número de dispositivos, permiten integraciones mediante REST API y soportan la visualización de datos con el empleo de herramientas propias. Sin embargo, las principales diferencias están en los protocolos de recolección de información, en los tipos de bases de datos que soportan y en las herramientas de análisis de datos que utilizan. WSO2IoT soporta los protocolos HTTP y MQTT, mientras que ThingsBoard soporta HTTP, MQTT, CoAP y

Tabla 2.1. Comparación entre diferentes plataformas IoT de código abierto (Li, 2018).

	Características*									
Plataforma IoT	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Iotivity	✓	✓	✓	Nube	✗	✓	✓	✓	Excelente	Gratis
OpenIot	✗	✗	✗	Nube	✓	✗	AP	✓	Buena	Gratis
Thinger.io	✓	✓	/	Nube	/	✓	✓	/	Buena	Gratis
ThingsBoard	✓	✓	✓	✓	✓	RPC	✓	✓	Excelente	Gratis
ThingSpeak	✓	✗	✓	Nube	✓	✓	AP	/	Grande	Pago
Macchina.io	✓	✗	✓	✓	✗	✓	GW	✓	Buena	Gratis
WSO2IoT	✓	✓	✓	✓	✓	✓	✓	✓	Excelente	Gratis
SiteWhere	✓	✓	✓	✓	✓	✓	AP	✓	Buena	Gratis
Kaa	✓	✓	✓	✓	✓	✓	✓	/	Buena	Pago
Alljoyn	✓	✓	✓	✓	✗	✓	✓	✓	Excelente	Pago

*1. Dispositivos heterogéneos; 2. Manejo de Dispositivos; 3. Transferencia en la capa de aplicación; 4. Almacenamiento de datos; 5. Procesamiento inteligente; 6. Interfaz RESTful; 7. Modo de implementación; 8. Seguridad; 9. Actividades de desarrollo; 10. Tipo de *software*. AP: Plataforma de aplicación, GW: *Gateway*. RPC: Llamada a procedimiento remoto ✓: Cumple con los requisitos, ✗: no cumple con los requisitos, /: desconocido.

OPC-UA. Las bases de datos que permite utilizar ThingsBoard son más variadas que las empleadas por WSO2IoT, que son del tipo SQL. ThingsBoard soporta la herramienta Apache Spark y el *plugin* Kafka para el análisis de datos en tiempo real, mientras que WSO2IoT incluye un servidor propio para este fin (Bitencourt, 2018). La plataforma IoT ThingsBoard cumple satisfactoriamente todos los requisitos necesarios para su correcto funcionamiento en aplicaciones industriales, es de *software* libre y tiene un excelente nivel de aceptación a escala mundial. Esta plataforma ha sido seleccionada en la presente investigación y algunas de sus características más destacadas se muestran en el siguiente epígrafe.

2.2.1 Plataforma IoT ThingsBoard

ThingsBoard es una plataforma IoT de código abierto para la recolección, procesamiento, visualización de datos y administración de dispositivos. Permite la conectividad de dispositivos a través de protocolos IoT que se han convertido en estándares de la industria como MQTT, CoAP, HTTP y soporta implementaciones en la nube y en servidores locales. Proporciona al usuario administrador una interfaz web idónea para registrar y administrar dispositivos, donde sus credenciales pueden ser cadenas definidas por el usuario denominadas token de acceso o certificados X.509[7]. ThingsBoard da la posibilidad de crear contenedores llamados activos (*assets*) para reorganizar los datos y se pueden asignar atributos a los dispositivos y activos en forma de pares clave-valor (los atributos pueden ser características de dispositivo como la versión de *firmware*, especificaciones de *hardware*, etc.). Además, permite la definición de reglas que se aplican a los mensajes entrantes y de *plugins* que procesan los mismos. Las reglas consisten en filtros para mensajes recibidos, procesadores que añaden metadatos a los mensajes y acciones que convierten mensajes a nuevos mensajes personalizados reenviados a un *plugin*. Operaciones típicas que podrían aplicarse mediante reglas son las comparaciones entre los valores recibidos y el establecimiento de umbrales (para comprobar, por ejemplo, si una temperatura ha excedido un umbral de seguridad). ThingsBoard también permite definir alarmas para dispositivos y activos, lo cual se puede utilizar para enviar notificaciones a los dispositivos cuando se detectan situaciones problémicas en la fase de preprocesamiento o procesamiento de datos. Además de los protocolos ligeros tales como MQTT y CoAP, la plataforma soporta los servicios tradicionales de REST, o sea, proporciona REST API para administrar entidades, dispositivos y activos para obtener datos de telemetrías y un *plugin* REST para enviar solicitudes HTTP a programas externos. ThingsBoard ofrece la posibilidad de crear paneles actualizados en tiempo real para la visualización de datos, que pueden personalizarse con más de 30 widgets (Tommaso De Paolis, De Luca y Paiano, 2018).

[7] Estándar para infraestructuras de claves públicas

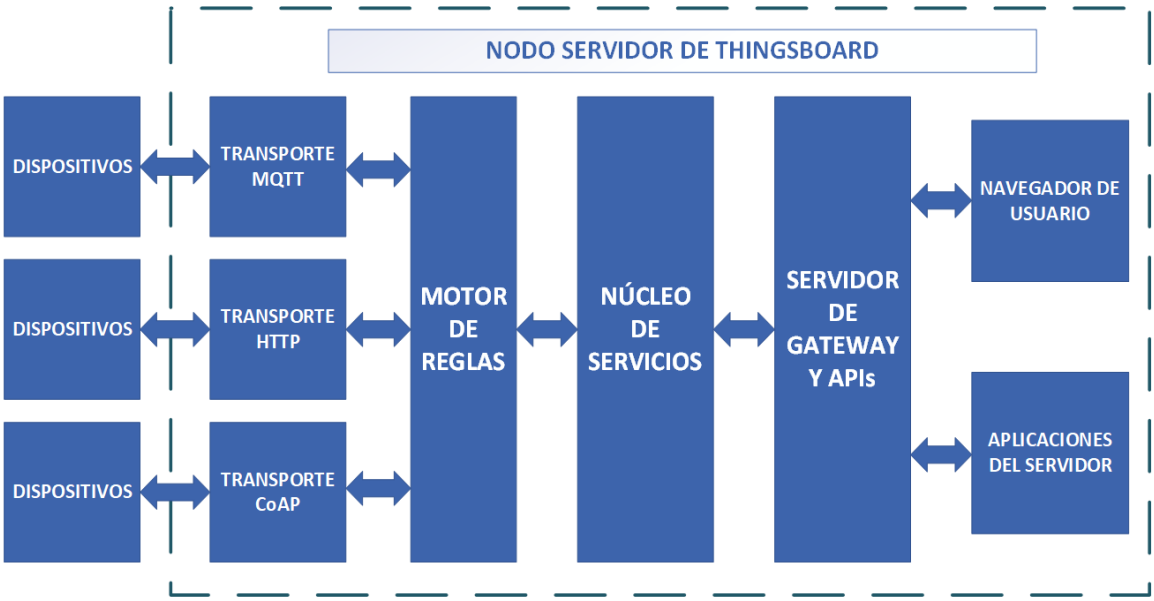


Figura 2.2. Arquitectura de la plataforma IoT ThingsBoard (ThingsBoard, 2019).

Otras características de la plataforma IoT, se exponen en (ThingsBoard, 2019) y se muestran a continuación:

- Proporciona APIs para que las aplicaciones del servidor envíen comandos RPC a dispositivos y viceversa.
- Soporta despliegue monolítico[8] para aplicaciones con recursos limitados.
- Proporciona capacidad para actualizar los microservicios[9].
- Soporta varios inquilinos. Un inquilino puede tener varios administradores y millones de dispositivos y clientes.
- Escalabilidad horizontal. Los dispositivos pueden aumentar de forma lineal a medida que se agregan nuevos servidores de ThingsBoard, sin provocar colapsos, reinicios de servidores o errores de aplicación.
- Tolerancia a fallos.
- Soporta encriptación de transporte tanto para MQTT como para protocolos HTTP.

[8] Minimización del consumo de memoria para ejecutar la plataforma.
[9] Componentes almacenados en servidores independientes a los que se accede mediante APIs.

- Soporta la autenticación de dispositivos y la administración de credenciales de dispositivos.
- ThingsBoard tiene licencia Apache License 2.0, por lo cual se puede utilizar en productos comerciales de forma gratuita.
- Soporta bases de datos SQL, NoSQL e híbridas.

Adicionalmente, ThingsBoard incorpora la funcionalidad de *gateway* IoT, solución de código abierto que permite integrar dispositivos IoT conectados a sistemas heredados y de terceros y permite un mejor aprovechamiento de los recursos de *hardware* existentes. La interconectividad entre los *gateway* es sencilla, debido a que disponen de una amplia comunidad de desarrollo que ha propiciado grandes ventajas en cuanto a su uso. ThingsBoard IoT Gateway ofrece las siguientes características:

- Extensión MQTT para controlar, configurar y recopilar datos de dispositivos IoT que están conectados a intermediarios externos MQTT utilizando protocolos existentes.
- Extensión OPC-UA para recopilar datos de dispositivos IoT que están conectados a servidores OPC-UA.
- Extensión Sigfox para recopilar datos de dispositivos IoT que están conectados a Sigfox Backend.
- Extensión Modbus para recopilar datos de dispositivos IoT que están conectados a través del protocolo Modbus.
- Persistencia de los datos recopilados para garantizar su entrega en caso de fallas de red o *hardware*.
- Reconexión automática al clúster ThingsBoard.
- Mapeo simple pero potente de los datos y mensajes entrantes a un formato unificado.

El *gateway* proporciona una API de integración simple y encapsula tareas comunes relacionadas con ThingsBoard como el aprovisionamiento de dispositivos, persistencia y entrega de datos locales, conversores, adaptadores de mensajes y otros. Para los desarrolladores de aplicaciones está permitido el uso de lenguajes de programación como Python, Go, C, C++ y la conexión puede realizarse a través de un intermediario MQTT externo o un servidor OPC-UA. Los dispositivos IoT que admiten otros protocolos se

pueden conectar a la puerta de enlace implementando extensiones personalizadas (ThingsBoard, 2019).

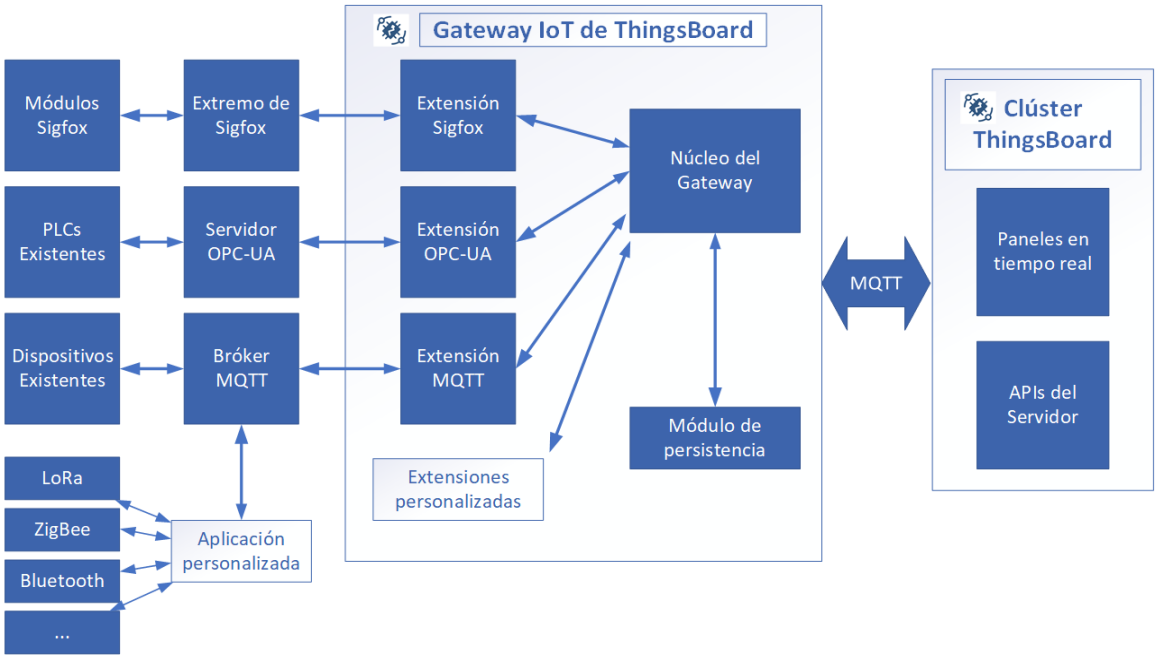


Figura 2.3. Arquitectura del *gateway* IoT de ThingsBoard (ThingsBoard, 2019).

2.3 Selección del protocolo de comunicación de IoT

En la búsqueda de la normalización e interoperabilidad del IoT, se presentan protocolos de comunicación muy prometedores que pueden convertirse en un estándar para aplicaciones de este tipo. Ejemplo de ello, son los protocolos MQTT y CoAP, diseñados especialmente para dispositivos con recursos limitados en cuanto a procesamiento y ancho de banda.

MQTT es un protocolo de mensajería para publicar y suscribirse diseñado para comunicaciones M2M ligeras. Fue desarrollado originalmente por IBM y ahora es un estándar abierto. En cuanto a su arquitectura, MQTT tiene un modelo de cliente/servidor, donde cada sensor es un cliente y se conecta a un servidor, conocido como bróker, sobre TCP. MQTT está orientado a mensajes, donde cada mensaje es un segmento discreto de datos. Cada mensaje se publica en una dirección, conocida como tema (*topic*). Los clientes pueden suscribirse a varios temas. Cada cliente suscrito a un tema recibe todos los mensajes publicados en el tema. Por ejemplo, en una red sencilla con tres clientes y un bróker central (figura 2.4), los tres clientes abren conexiones TCP con el bróker. Dos de los clientes (B y

C) se suscriben al tema “temperatura”. El otro cliente (A) publica un valor de temperatura en el tema y el bróker remite el mensaje a todos los clientes suscritos. El modelo de publicación/suscripción permite a los clientes de MQTT comunicarse uno a uno, uno a muchos y muchos a uno.

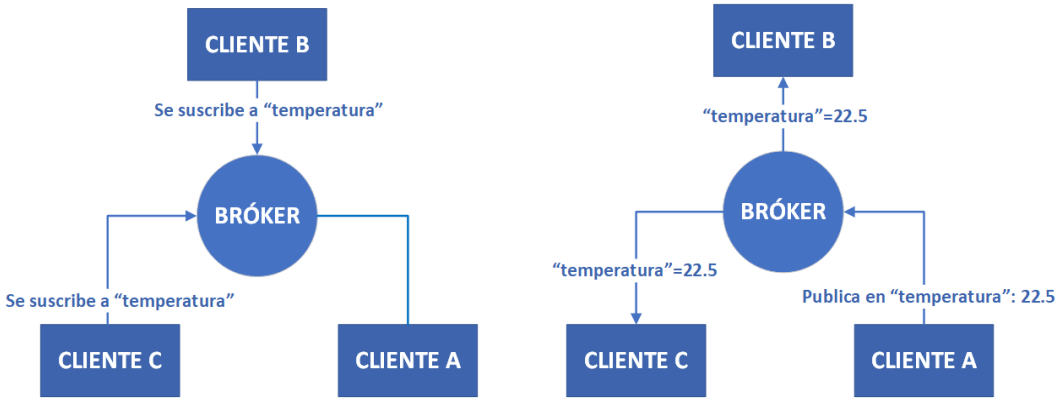


Figura 2.4. Comunicación mediante el protocolo MQTT.

MQTT se enfoca en una mensajería confiable, por lo que incluye búferes de mensajes y niveles de QoS controlados por el bróker. La selección del nivel de QoS depende de la aplicación IoT y de la red que se sustenta la misma. Los tres niveles de QoS que soporta son:

- QoS-0 (como máximo una vez): En este nivel de QoS, el mensaje se envía a lo máximo una vez y no proporciona garantía de entrega de un mensaje.
- QoS-1 (al menos una vez): En este nivel de QoS, los datos se envían al menos una vez y es posible entregar un mensaje más de una vez si el receptor no lo recibe.
- QoS-2 (exactamente una vez): En este nivel de QoS, el mensaje es enviado exactamente una vez usando establecimiento de comunicación (*handshake*) de 4 vías (Soni y Makwana, 2017).

Los clientes MQTT pueden registrar un mensaje personalizado que podrá ser enviado por el bróker si se interrumpe la conexión, indicando a los suscriptores cuando un dispositivo se desconecta. Además, MQTT tiene soporte para la persistencia de mensajes en el bróker (solo el mensaje más reciente). Al publicar mensajes, los clientes pueden solicitar que el bróker lo retenga y cuando un cliente se suscribe a un tema, cualquier mensaje persistido será enviado a dicho cliente. En cuanto a la seguridad, el bróker MQTT puede requerir la

autenticación de usuario y contraseña de los clientes para conectarse. Para garantizar la privacidad, la conexión TCP puede estar cifrada con SSL/TLS. A pesar de que MQTT está diseñado para ser ligero, tiene dos inconvenientes para dispositivos muy limitados. Cada cliente de MQTT debe soportar TCP y tendrá típicamente una conexión abierta al bróker en todo momento. En algunos entornos donde la pérdida de paquetes es alta o los recursos de cómputo son escasos, esto es un problema. El otro problema es que los nombres de temas de MQTT son a menudo cadenas largas que los hacen imprácticos para el estándar IEEE 802.15.4. Ambas deficiencias son resueltas por el protocolo MQTT-SN, que define un mapeo UDP de MQTT y añade soporte del bróker para la indexación de temas (Jaffey, 2014).

CoAP es un protocolo M2M ligero del Grupo de Trabajo IETF CoRE (Constrained RESTful Environments). CoAP soporta tanto la arquitectura de solicitud/respuesta como la de recurso/observación (una variante de publicación/suscripción). CoAP se desarrolla principalmente para interoperar con HTTP y RESTful Web a través de *proxies* simples. A diferencia de MQTT, CoAP utiliza URI[10] en lugar de temas. El publicador publica datos en el URI y el suscriptor se suscribe a un recurso específico indicado por el URI. Cuando un publicador publica nuevos datos en el URI, entonces todos los suscriptores son notificados sobre el nuevo valor. CoAP es un protocolo binario y normalmente requiere cabecera fija de 4 bytes con cargas pequeñas de mensajes, hasta un tamaño máximo que depende del servidor web o de la tecnología de programación. CoAP utiliza UDP como protocolo de transporte y DTLS para la seguridad. Utiliza mensajes “confirmables” o “no confirmables” para proporcionar dos niveles de QoS diferentes. Los mensajes confirmables deben ser reconocidos por el receptor con un paquete ACK[11] y los mensajes no confirmables no. CoAP ofrece más funcionalidad que MQTT, como por ejemplo, el soporte de contenido de negociación para expresar una representación preferida de un recurso, lo que permite que el cliente y el servidor evolucionen de forma independiente (Naik, 2017).

[10] Cadena de caracteres que identifica los recursos de una red de forma unívoca.
[11] Del inglés *Acknowledgement*, que se traduce como acuse de recibo.

Tabla 2.2. Comparación de los protocolos MQTT y CoAP (Naik, 2017).

Criterio	**MQTT**	**CoAP**
Año	1999	2010
Arquitectura.	Cliente/Bróker	Cliente/Servidor o Cliente/Bróker
Abstracción	Publicación/Suscripción	Solicitud/Respuesta o Publicar/Suscribir
Tamaño de encabezado	2 bytes	4 bytes
Tamaño de mensaje	Pequeño (256 MB máximo)	Pequeño para caber en datagrama IP
Calidad del Servicio (QoS) o Confiabilidad	QoS-0: máximo una vez QoS-1: Al menos una vez QoS-2: Exactamente una vez	Mensaje confirmable (similar a máximo una vez) o Mensaje no confirmable (similar al menos una vez)
Protocolo de Transporte	TCP (MQTT-SN usa UDP)	UDP, SCTP
Seguridad	TLS/SSL	DTLS, IPSec
Puerto por defecto	1883/8883(TLS/SSL)	5683 (puerto UDP) / 5684 (DLTS)
Formato de codificación	Binario	Binario
Modelo de Licencia	Código Abierto	Código Abierto
Soporte Organizacional	IBM, Facebook, Eurotech, Cisco, Red Hat, Tibco, ITSO, M2Mi, Amazon, InduSoft, etc	Large Web Community Support, Cisco, Contiki, Erika, IoTivity

Entre las semejanzas de estos protocolos, se encuentran que ambos son de código abierto, se adaptan mejor a entornos restringidos que el protocolo HTTP, corren sobre IP y tienen un amplio rango de implementación. Sin embargo, tienen diferencias fundamentales. MQTT es un protocolo de comunicación para pasar mensajes entre múltiples clientes a través de un bróker central. Desacopla al productor del consumidor dejando que los clientes publiquen y que el bróker decida dónde enrutar y copiar mensajes. A pesar de que MQTT tiene cierto soporte para la persistencia, es mejor como un bus de comunicaciones para los datos en tiempo real. CoAP es, principalmente, un protocolo para transferir información de estado entre el cliente y el servidor. Aunque tiene soporte para los recursos de observación,

CoAP se adapta mejor a un modelo de transferencia de estado, no puramente basado en eventos.

Los clientes MQTT establecen una conexión TCP constante hacia un bróker. Lo clientes y servidores de CoAP envían y reciben paquetes UDP. MQTT no proporciona soporte para etiquetar tipos de mensajes u otros metadatos para ayudar a los clientes a entenderlo, o sea, se puede utilizar para cualquier propósito, pero todos los clientes deben conocer los formatos de mensaje para permitir la comunicación. CoAP, por el contrario, proporciona soporte para la negociación de contenido y descubrimiento, permitiendo que los dispositivos interactúen entre sí para encontrar rutas de intercambio de datos (Jaffey, 2014). Como se ha podido constatar, ambos protocolos tienen sus pros y contras. En el caso específico de esta investigación se utiliza el protocolo MQTT, debido a que el tamaño del encabezado es más pequeño, tiene 3 niveles de QoS, permite los protocolos de transporte TCP y UDP, y tiene mayor soporte por parte de la comunidad de desarrolladores.

2.3.1 Protocolo MQTT-SN

MQTT-SN es un protocolo de publicación/suscripción para WSN considerado una versión de MQTT que se adapta a las particularidades de un ambiente de comunicación inalámbrica y utiliza UDP como protocolo de transporte. Los enlaces de radio inalámbricos tienen en general tasas de fallo más altas que las cableadas debido a su susceptibilidad a la atenuación y a la interferencia, presentan una tasa de transmisión más baja y sus paquetes de datos tienen una longitud muy corta para hacerlos resistentes a errores de transmisión. MQTT-SN está diseñado para asemejarse lo más posible a MQTT, pero se adapta a las peculiaridades de un entorno de comunicación inalámbrica con poco ancho de banda, fallas de enlace, longitud corta de mensajes, etc. Está optimizado para la implementación en dispositivos de bajo costo, alimentados con baterías y con recursos de procesamiento y almacenamiento limitados. En comparación con MQTT, MQTT-SN se caracteriza por las siguientes diferencias:

- El mensaje de conexión se divide en tres mensajes. Los dos adicionales son opcionales y se utilizan para transferir un nuevo tema o mensaje al servidor.
- El nombre del tema en los mensajes de publicación se reemplaza por un identificador de tema de dos bytes.

- Los identificadores de temas predefinidos y los nombres “cortos” de temas se introducen, para los cuales no se requiere ningún registro.
- Los identificadores de tema predefinidos son también un reemplazo de dos bytes del nombre del tema y se conocen con antelación tanto por la aplicación del cliente como por el servidor.
- Un procedimiento de descubrimiento ayuda a los clientes sin una dirección de servidor o *gateway* preconfigurada a encontrar la dirección de la red operativa actual.
- Se define un nuevo procedimiento de mantenimiento (*keep-alive*) *offline* para el soporte de clientes en reposo. Con este procedimiento, los dispositivos alimentados con baterías pueden ir a un estado de reposo durante el cual todos los mensajes destinados a ellos se almacenan en un búfer en el servidor o *gateway* y se entregan cuando se activan (Stanford-Clark y Linh Truong, 2013).

2.3.2 Bróker de mensajería RSMB

RSMB es un bróker MQTT de publicación/suscripción, ligero, de baja sobrecarga y permite el protocolo MQTT-SN. Posibilita la mensajería entre pequeños dispositivos como son los sensores y actuadores a través de redes que están limitadas en términos de ancho de banda, capacidades de procesamiento y confiabilidad. El publicador envía un mensaje al bróker que luego distribuye los mensajes a los usuarios suscritos que han solicitado recibir esos mensajes. El bróker RSMB es similar a los otros brókeres de MQTT (HiveMQ, Mosquitto, Adafruit) pero la diferencia es que proporciona conectividad con otros brókeres en paralelo creando un puente que permite la conexión y el intercambio de mensajes con otros servidores MQTT. También soporta el protocolo MQTT-SN que, cuando se implementa, acepta paquetes UDP y envía esos paquetes al bróker de MQTT local y a los brókeres MQTT externos que están conectados. RSMB ocupa poco espacio de almacenamiento y se puede ejecutar con poca memoria RAM, haciéndolo ideal para la instalación y ejecución en pequeños servidores y dispositivos embebidos(Light y Craggs, 2013; Priya, 2019).

2.4 Selección de la Base de Datos

Teniendo en cuenta la plataforma IoT que se va a implementar en la aplicación que se propone, la selección de la base de datos donde se almacenará la información, se realiza siguiendo las recomendaciones de los desarrolladores de la plataforma ThingsBoard. En (ThingsBoard, 2019) se muestra la base de datos HSQLDB para propósitos de evaluación o desarrollo, la base de datos NoSQL Apache Cassandra y la base de datos relacional PostgreSQL. En el caso de la base de datos Apache Cassandra, esta se descarta debido a que no está recomendada para *hardware* con menos de 4 GB de memoria RAM, y dado que el *software* se instalará en una Raspberry Pi 3 Modelo B (1 GB de RAM), no se garantiza un correcto funcionamiento de la misma. El equipo desarrollador de ThingsBoard propone el empleo de PostgreSQL para dispositivos con 1GB de memoria RAM.

PostgreSQL es un poderoso sistema de base de datos relacional de código abierto que utiliza y extiende el lenguaje SQL combinado con varias características y que permite el almacenamiento seguro de los datos. Los orígenes de PostgreSQL datan de 1986 y tiene más de 30 años de desarrollo activo. PostgreSQL ha ganado una gran reputación por su arquitectura, confiabilidad, integridad de datos, características robustas, extensibilidad, y la dedicación de la comunidad de código abierto detrás del *software* para entregar constantemente mejoras y soluciones innovadoras. PostgreSQL se ejecuta en los principales sistemas operativos y tiene potentes complementos como el popular extensor de base de datos geoespaciales PostGIS.

PostgreSQL viene con muchas características dirigidas a ayudar a los desarrolladores a crear aplicaciones, proteger la integridad de los datos, crear entornos tolerantes a errores, y ayudarle a administrar sus datos sin importar cuán grande o pequeño sea el conjunto de datos. Además de ser libre y de código abierto, PostgreSQL es altamente extensible. Por ejemplo, se pueden definir tipos de datos propios, crear funciones personalizadas, incluso escribir código de diferentes lenguajes de programación sin volver a compilar la base de datos (*PostgreSQL*, 2019).

Tabla 2.3. Características de PostgreSQL (*PostgreSQL*, 2019).

Tipos de datos	Primitivos: Enteros, Numéricos, Cadenas, Booleanos. Estructurados: Fecha/Hora, Arreglos, Rango. Documentos: JSON, XML, clave-valor. Personalización de tipos de datos.
Concurrencia y Rendimiento	Indexación: árbol-B, Multicolumna, Expresiones, Parcial. Sofisticado planificador y optimizador de consultas. Escaneos de un solo índice. Particionamiento de tablas.
Confiabilidad y recuperación	Registro adelantado de escritura. Replicación: Asincrónica, Sincrónica, Lógica.
Seguridad	Diferentes métodos de autenticación. Sistema robusto de control de acceso. Seguridad a nivel de columnas y de filas.
Extensibilidad	Funciones y procedimientos almacenados. Idiomas de Procedimiento: PL/PGSQL, Perl, Python, etc. Datos externos: puede conectarse a otras bases de datos con interfaz SQL.

2.5 Arquitectura de la aplicación IoT diseñada

En investigaciones del Grupo de Internet de las Cosas, Automatización e Inteligencia Artificial perteneciente a la Universidad Central "Marta Abreu" de Las Villas se diseñó la arquitectura de *hardware* que será utilizada como referencia en este trabajo (Estevez Pérez, 2018), sustituyendo los nodos (*motes*) Zolertia RE-mote por dispositivos Zolertia Z1 (producto de la disponibilidad de los mismos) que cuentan con un procesador MSP430F2617 de 16-bit a 16 MHz, 8 kB de memoria RAM, 92 kB de memoria *flash*, transceptor de radio CC2420 empotrado y comunicación IEEE 802.15.4 a 2.4GHz (Zolertia, 2010; Liñan, Bagula y Pietrosemoli, 2016). El enrutador de frontera está formado por un Zolertia Z1 configurado como *border router* conectado por comunicación serie (USB) a una Raspberry Pi 3 Modelo B. La Raspberry Pi 3 Modelo B presenta un procesador ARM Cortex-A53 de 4 núcleos a 1.2GHz, conectividad Wifi, LAN y Bluetooth, 1GB de memoria RAM y conector para tarjetas SD (Raspberry Pi Foundation, 2016).

Los sensores empleados son: sensor de temperatura y humedad relativa DHT22 (alimentación de 3.3V a 6VDC, rango de operación de -40˚C a 80 ˚C y 0 a 100% de

humedad relativa), sensor fotosensible análogo-digital LM393 (alimentación de 3.3 a 5VDC, corriente de operación de 15 mA, salida analógica de 0 a 5V y salida digital de disparo de 0 a 5V) y el sensor de temperatura TMP102 de Texas Instruments integrado en los dispositivos Zolertia Z1 (rango de operación de -25 ˚C a 85 ˚C, exactitud de 0.5 ˚C) para mediciones internas de cada nodo. Los módulos YL69 y YL-38 del sensor de humedad del suelo (alimentación de 3.3V a 5VDC, salida 0a 4.2V) no se encontraban disponibles para la realización de esta investigación.

En cuanto a la topología y la arquitectura de *software* de la WSN para el monitoreo de variables ambientales en casas de cultivos, se toma como referencia la presentada en (Estevez Pérez, 2018). La topología seleccionada es la de una red mallada ad-hoc, pues estas no poseen infraestructura, son flexibles y todos los nodos ofrecen servicios de enrutamiento, o sea, todos los nodos además de realizar la función de nodos finales también son enrutadores. La pila de protocolos utilizada implementa en la capa física el estándar de comunicación IEEE 802.15.4, en la capa de red usa el protocolo 6LoWPAN y el protocolo de ruteo RPL, y el protocolo de la capa de transporte es UDP, permitiendo así la implementación del protocolo MQTT-SN. Como sistema operativo para los *motes* Zolertia Z1 se emplea Contiki, de código abierto y especialmente diseñado para microcontroladores de bajo consumo y baja potencia. Contiki soporta los estándares IPv6 e IPv4 y los estándares inalámbricos de bajo consumo: 6LoWPAN, RPL, CoAP y MQTT. Las aplicaciones de Contiki están escritas en lenguaje C y con el simulador Cooja pueden emularse las aplicaciones antes de grabarlas en el *hardware*.

El sistema operativo Raspbian es instalado en la Raspberry Pi 3 Modelo B, pues contiene un conjunto de programas básicos y utilidades para comenzar a explotar las capacidades de esta computadora de placa reducida. Raspbian es un sistema operativo libre, basado en Debian (Linux), y presenta una amplia comunidad de desarrolladores (Raspbian, 2019). Además, en la Raspberry Pi se instala la plataforma IoT ThingsBoard, que incluye la base de datos PostgreSQL para el almacenamiento de la información, y proporciona el *gateway* IoT. Este último cuenta con una extensión MQTT que permite la obtención de datos de los dispositivos a través del bróker RSMB.

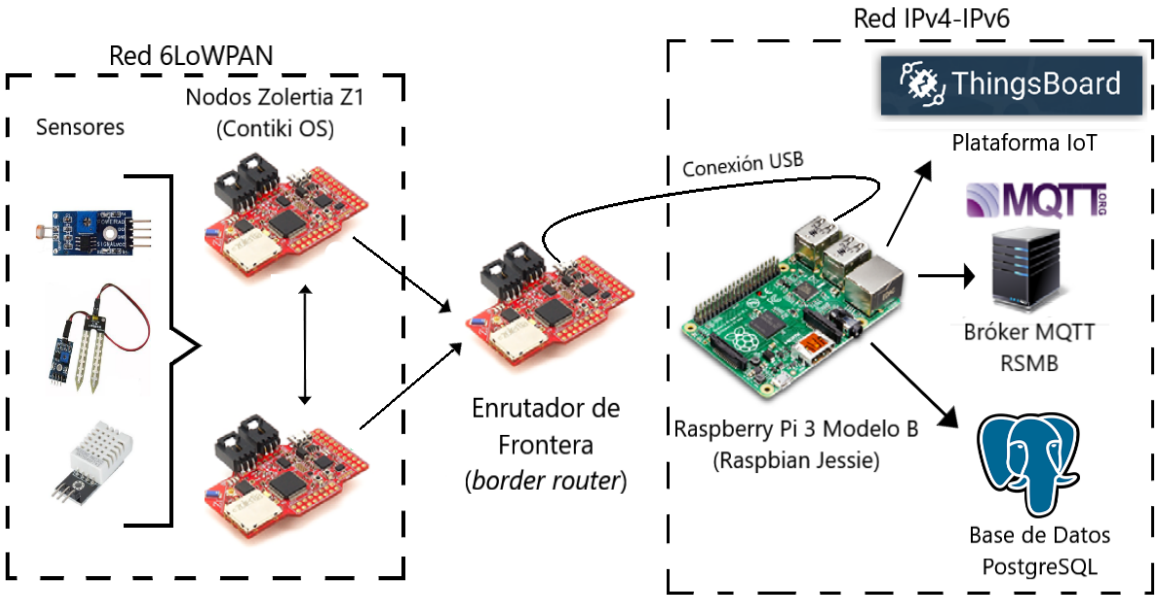


Figura 2.5. Arquitectura de *hardware* y software de la aplicación IoT diseñada.

2.6 Instalación y configuración de la plataforma IoT

Seleccionadas todas las herramientas de *software*, estas se deben integrar para conformar la aplicación IoT. En el presente epígrafe se describen los pasos de instalación y configuración de la plataforma ThingsBoard, así como de los programas informáticos de terceros que complementan la misma. Además, se describe la configuración de la WSN y se destacan aspectos básicos sobre el funcionamiento de la plataforma. Los repositorios de algunos programas utilizados se pueden ver en el anexo II.

El proceso de instalación comienza con el chequeo de la versión de Java instalada en la Raspberry Pi, puesto que los servicios de ThingsBoard utilizan Oracle Java 8 que se encuentra instalado por defecto en Raspbian, por lo cual no deberían presentarse problemas en este paso, y de ser así, cualquier versión de Java superior a la 1.8 puede ser utilizada. Luego, se procede a la instalación de la base de datos PostgreSQL, donde se crea un usuario y su contraseña, así como una base de datos con el nombre *"thingsboard"*, que será utilizada para el almacenamiento de toda la información relacionada con la plataforma. El servicio de ThingsBoard se instala teniendo en cuenta el *hardware* que será utilizado, considerando la cantidad de memoria RAM disponible para su ejecución. En este caso es

necesario limitar el uso máximo de memoria del mismo, estableciéndolo en 256 MB, para evitar el colapso de la Raspberry Pi que cuenta con solo 1GB de memoria RAM.

En el archivo de configuración de la plataforma se modifica la base de datos que será utilizada, inhabilitando la sección de código correspondiente a la empleada por defecto e introduciendo el código relacionado con PostgreSQL, donde se debe introducir el nombre de usuario y la contraseña de la misma. Con el servicio de ThingsBoard instalado y configurado, se ejecuta un *script* de instalación que permitirá cargar los datos relacionados con los usuarios, dispositivos, activos, entidades, reglas y *widgets*, así como la creación de las diferentes tablas dentro de la base de datos. Finalmente, se inicia el servicio de ThingsBoard y la interfaz de usuario web queda habilitada en el *localhost* de la Raspberry Pi con el puerto 8080 (ThingsBoard, 2019).

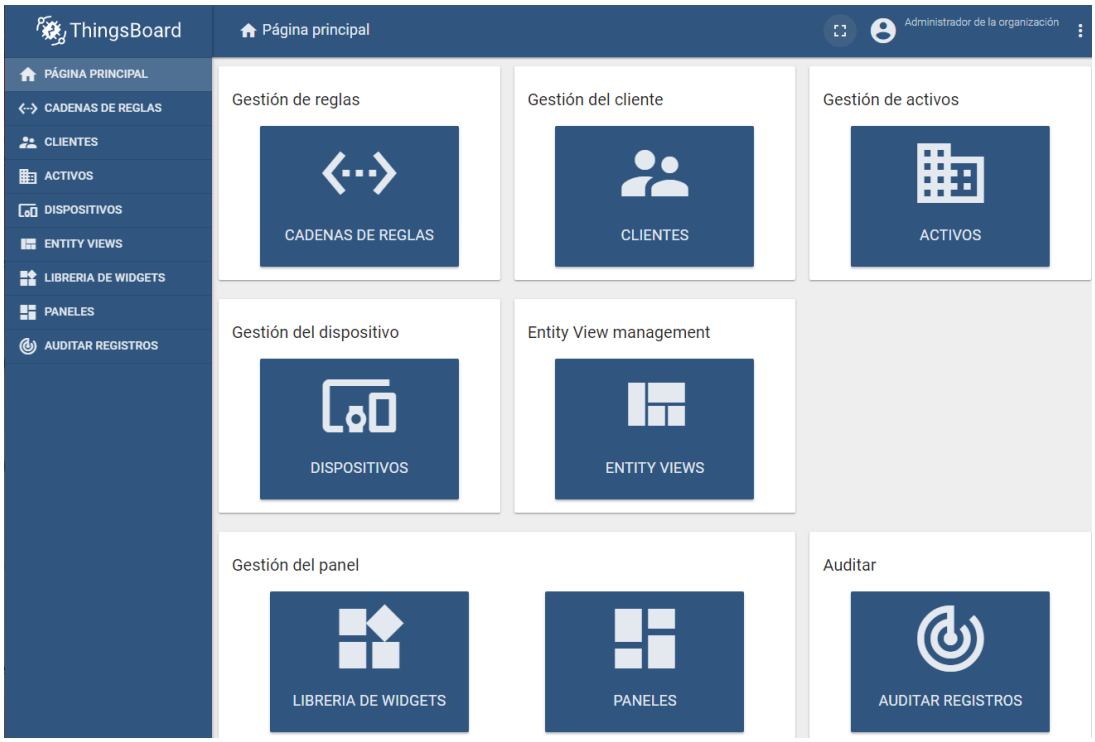


Figura 2.6. Página principal de la interfaz de usuario de la plataforma IoT ThingsBoard.

Otro *software* fundamental que forma parte de la plataforma es el *gateway* IoT de ThingsBoard, que permitirá la integración con dispositivos externos y facilita el intercambio de datos con los mismos. Para su instalación, igualmente se debe tener en cuenta el *hardware* que lo ejecutará, por lo que se necesita limitar el uso de memoria RAM

a 150 MB. Para la activación del servicio de *gateway*, se crea un dispositivo dentro de la plataforma para que cumpla con esta función. En el archivo de configuración del gateway IoT de ThingsBoard, se introducen datos como el *host* y el puerto de la conexión, el código de acceso (token) del dispositivo *gateway* y se especifica si la extensión del protocolo utilizado (MQTT) es local o remota. La extensión del protocolo MQTT contiene la configuración que permite el mapeo de los mensajes en formato JSON provenientes del bróker externo y que contienen la telemetría y los atributos de los dispositivos. En caso de seleccionarse una extensión remota, esta dispone de una interfaz de usuario que facilita la configuración de diferentes brókeres, donde cada una incluye el puerto, el host, el intervalo de reintento de conexión, las credenciales, el mapeo, solicitudes de conexión, desconexión, atributos, actualizaciones de atributos y el RPC correspondiente al servidor (ThingsBoard, 2019).

El bróker RSMB se configura para que reciba datos del puerto 1883 en el caso del protocolo MQTT (comunicación TCP/IP con la plataforma) y del puerto 1884 cuando se utilice el protocolo MQTT-SN (comunicación UDP con los nodos). Esta configuración del bróker RSMB permite crear un puente entre protocolos de la capa de aplicación que emplean diferentes protocolos de transporte, siendo muy útil en el intercambio de información de la aplicación IoT. Es válido aclarar que tanto la plataforma ThingsBoard, como el bróker utilizado, emplean el mismo puerto (1883) para los servicios de MQTT, por lo que si se utilizan estos *software* en el mismo *host* se debe modificar el puerto de uno de ellos, en este caso, el de la plataforma (modificado a 1882).

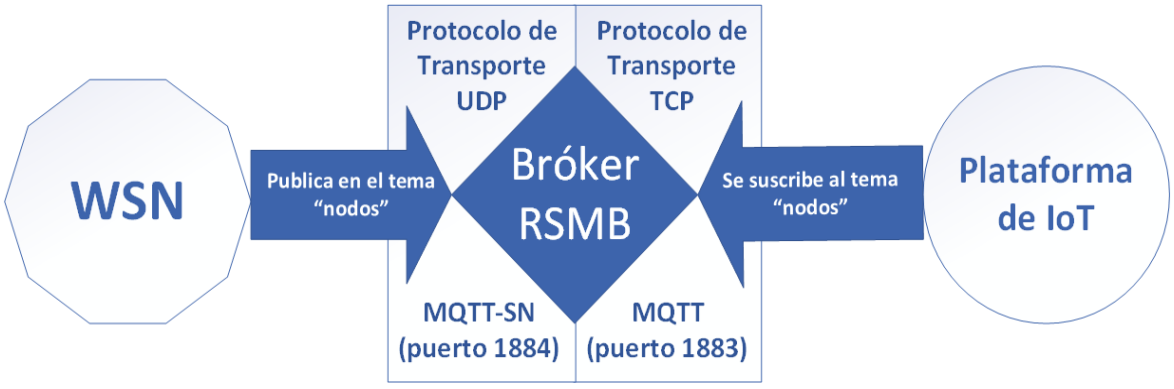


Figura 2.7. Función del bróker en la aplicación IoT que se propone.

2.6.1 Configuración de la WSN

Para la configuración de los nodos, es necesario tener instalada una distribución Ubuntu (mayor de 14.04) del sistema operativo Linux, a la cual se deben incorporar los archivos relacionados con Contiki OS, las herramientas de desarrollo, compiladores y simuladores necesarios. En esta investigación se utilizó Instant Contiki, un entorno de desarrollo de aplicaciones para este sistema operativo, que constituye una máquina virtual de Ubuntu con todos los programas necesarios preinstalados. Algunas de las herramientas fundamentales que deben estar instaladas son el MSP430 (compilador msp430-gcc) y el ARM Cortex-M3 (herramienta para procesadores embebidos con arquitectura ARM). Los programas que se ejecutarán en los nodos Zolertia, se programan en lenguaje C y constan de un archivo de configuración (*project-conf.h*), un archivo para definir reglas de compilación (*Makefile*) y los archivos correspondientes al código de ejecución (archivos con extensión *.c*) con sus respectivas cabeceras (archivos con extensión *.h*). En el caso del nodo enrutador de frontera, el programa utilizado implementa una interfaz basada en comunicación serie llamada *SLIP*, que permite conectar el nodo a un *host* (Raspberry Pi) usando la herramienta "tunslip6" sobre el puerto serie para crear una interfaz de red tunelizada, otorgando prefijos IPv6 para establecer las direcciones IPv6 globales de la red (Liñan, Bagula y Pietrosemoli, 2016). El programa se nombra *RPL Border Router* y se encuentra en los ejemplos que facilitan los desarrolladores de Contiki, por lo que solo se indica el tipo de dispositivo (Z1) antes de su instalación. Los nodos sensores se programan tomando como referencia el código esencial para el funcionamiento de los clientes MQTT-SN (ver anexo II). Este programa es básico, por lo cual se incorporan funcionalidades como las especificaciones de red (dirección del servidor UDP, puerto, tiempo de espera, etc.), activación y desactivación de sensores, envío de datos en el formato requerido, así como otras configuraciones relacionadas con el ahorro de energía y el enrutamiento.

Para comprobar el correcto funcionamiento de la WSN, tal y como proponen los autores en algunas aplicaciones presentadas en el primer capítulo, se emplea la herramienta Cooja, un simulador basado en Java diseñado para la simulación de WSN que ejecutan Contiki. Cooja simula nodos de WSN, incluso, de diferentes tipos de *software* o *hardware*. Es un simulador flexible, pues muchos de sus componentes se pueden sustituir fácilmente o ampliar con funciones adicionales, entre ellas, las simulaciones de *hardware* y de

entradas/salidas. Cooja, además, mezcla la simulación de alto nivel con simulación a bajo nivel del *hardware* de un nodo (Fraga Castro, 2015). Es un simulador gratuito y de código abierto, y su gran utilidad radica en la posibilidad de comprobar el código y el comportamiento del sistema diseñado antes de ejecutarlo en el *hardware*, con la limitación de que no está permitida la simulación de los sensores.

2.7 Consideraciones finales del capítulo

La selección de la plataforma IoT ThingsBoard responde a las necesidades de nuestro país de emplear *software* libres y extensibles, de manera que puedan ser adaptados a numerosos tipos de aplicaciones, no solo en la agricultura. Además, la posibilidad de instalar la misma en una Raspberry Pi le brinda un valor añadido, permitiendo implementaciones locales con dispositivos de bajo costo en el mercado. Respecto a la comunicación, el bróker de mensajería RSMB tiene un papel fundamental, pues soporta los protocolos de transporte UDP y TCP, constituyendo el enlace entre los protocolos de aplicación MQTT-SN y MQTT. De igual forma, en la arquitectura mostrada en el epígrafe 2.5, el nodo enrutador de frontera (*border router*) es un dispositivo esencial, permitiendo la comunicación de la WSN con la red IP local. En el próximo capítulo se valorará el funcionamiento de la aplicación IoT propuesta, mediante la simulación de la WSN y la realización de pruebas reales con el despliegue en las casas de cultivo.

CAPÍTULO 3. PRUEBAS Y RESULTADOS

Con el fin de comprobar el correcto funcionamiento de la aplicación IoT diseñada, se realizan una serie de simulaciones y pruebas reales para garantizar la calidad del producto final que se entrega al cliente. En el presente capítulo, se simulan varias topologías de WSN para valorar el enrutamiento de los nodos y se verifica si existe pérdida de datos al utilizar la topología que cumple con los requisitos mencionados en el epígrafe 2.5 del capítulo anterior. Además, se monitorea el comportamiento de la Raspberry Pi durante la ejecución de la plataforma IoT y los *software* que la complementan. Con el despliegue de la aplicación IoT en las casas de cultivo de la UEB de Cultivos Protegidos "Valle del Yabú" se valorará el comportamiento del sistema ante condiciones reales. Por último, se incluye un análisis económico y medioambiental de la propuesta de aplicación IoT.

3.1 Simulación de la WSN utilizando Cooja

Utilizando la herramienta Cooja se establece una WSN compuesta por un nodo enrutador de frontera y nueve nodos sensores. Se crean tres distribuciones de los nodos, de manera que se compruebe el funcionamiento de la red ante variaciones en la topología de la misma. Las tres topologías se pueden observar en la figura 3.1. En la figura 3.1 a), todos los nodos sensores se encuentran dentro del radio de alcance del nodo enrutador de frontera, por lo cual no es necesario su enrutamiento por otros nodos. En la figura 3.1 b), los nodos 2, 3 y 4 se encuentran dentro del radio de alcance del enrutador de frontera y los nodos restantes deben enrutarse a través de estos para acceder al *border router*. En el caso de la tercera topología, solo el nodo 2 puede acceder de manera directa al enrutador de frontera y los otros nodos deben encontrar la ruta hacia el mismo.

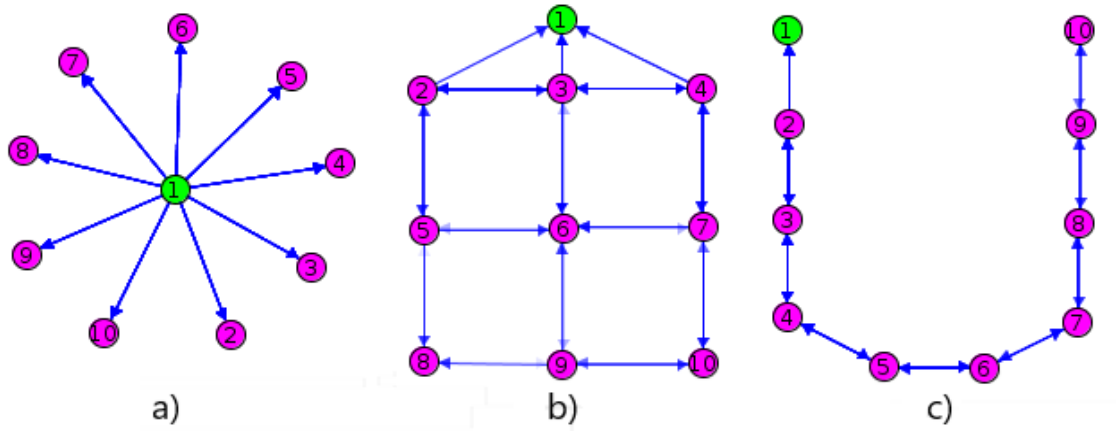


Figura 3.1. Diferentes topologías de WSN simuladas en Cooja.

3.1.1 Resultados obtenidos sobre el enrutamiento de los nodos

En Contiki, el protocolo de enrutamiento predeterminado es RPL, que facilita que los paquetes de datos lleguen al destino correcto utilizando las rutas más eficientes. Durante el proceso de programación de cada nodo sensor, se puede indicar el número máximo de rutas (nodos fuera de rango) o vecindades (nodos en rango) soportadas. El enrutador de frontera establece un servidor web donde se pueden visualizar los nodos vecinos y enrutados, así como el nodo que lo enruta. Esto resulta de suma importancia para comprobar la conexión de los nodos desde cualquier dispositivo conectado a la misma red que el *border router*. En la tabla 3.1 se exponen las direcciones IPv6 locales y globales de cada nodo y en la tabla 3.2 se exponen los datos obtenidos del servidor web antes mencionado.

Tabla 3.1. Direcciones IPv6 de los nodos.

Nodo	Dirección IPv6 Local	Dirección IPv6 Global
Enrutador de Frontera (*Border Router*)	fe80::c30c:0:0:1	fd00::c30c:0:0:1
2	fe80::c30c:0:0:2	fd00::c30c:0:0:2
3	fe80::c30c:0:0:3	fd00::c30c:0:0:3
4	fe80::c30c:0:0:4	fd00::c30c:0:0:4
5	fe80::c30c:0:0:5	fd00::c30c:0:0:5
6	fe80::c30c:0:0:6	fd00::c30c:0:0:6
7	fe80::c30c:0:0:7	fd00::c30c:0:0:7
8	fe80::c30c:0:0:8	fd00::c30c:0:0:8
9	fe80::c30c:0:0:9	fd00::c30c:0:0:9
10	fe80::c30c:0:0:a	fd00::c30c:0:0:a

Tabla 3.2. Estado de los nodos y ruta de los mismos.

Topología de WSN	Nodos Vecinos	Nodos Enrutados	Ruta
a)	Todos	Ninguno	Ninguna
b)	fe80::c30c:0:0:2 fe80::c30c:0:0:3 fe80::c30c:0:0:4	fd00::c30c:0:0:5	fe80::c30c:0:0:2
		fd00::c30c:0:0:6	fe80::c30c:0:0:3
		fd00::c30c:0:0:7	fe80::c30c:0:0:4
		fd00::c30c:0:0:8	fe80::c30c:0:0:2
		fd00::c30c:0:0:9	fe80::c30c:0:0:3
		fd00::c30c:0:0:a	fe80::c30c:0:0:4
c)	fe80::c30c:0:0:2	fd00::c30c:0:0:3	fe80::c30c:0:0:2
		fd00::c30c:0:0:4	fe80::c30c:0:0:2
		fd00::c30c:0:0:5	fe80::c30c:0:0:2
		fd00::c30c:0:0:6	fe80::c30c:0:0:2
		fd00::c30c:0:0:7	fe80::c30c:0:0:2
		fd00::c30c:0:0:8	fe80::c30c:0:0:2
		fd00::c30c:0:0:9	fe80::c30c:0:0:2
		fd00::c30c:0:0:a	fe80::c30c:0:0:2

Las direcciones locales son las establecidas para cada nodo dentro de la WSN, mientras que las direcciones globales se refieren a cada nodo dentro de la aplicación IoT en su conjunto. Un análisis de la tabla 3.2 muestra que en la topología a), todos los nodos son vecinos, o sea, que no se enrutan por otros nodos, sino que acceden directamente a la red IPv6 global. En la topología b), los nodos vecinos son el 2, 3 y 4, por lo tanto, los nodos 5 y 8 se enrutan a través del nodo 2, los nodos 6 y 9 se enrutan por el nodo 3 y los nodos 7 y 10 lo hacen mediante el nodo 4. En la topología c), todos los nodos se enrutan mediante el nodo 2, puesto que es el único nodo vecino del enrutador de frontera. Se debe destacar que el enrutador de frontera solo indica el nodo enrutador más próximo a él, algo que no supone un problema en la investigación actual.

3.1.2 Resultados obtenidos sobre el envío, visualización y almacenamiento de datos

Una de las ventajas de la aplicación IoT propuesta, es la posibilidad de evaluar el desempeño antes de realizar su implementación real. Esto es posible debido a que todo el *software* se ejecuta en el sistema operativo Linux, aunque en distribuciones diferentes, puesto que la Raspberry Pi 3 Modelo B utiliza Raspbian y los resultados mostrados en este

epígrafe son obtenidos a partir de la instalación de la plataforma en Ubuntu 18.04. La principal diferencia al ejecutar la aplicación IoT en ambas distribuciones radica en las limitaciones de recursos computacionales que presenta la Raspberry Pi en comparación con una computadora convencional. Esto no es un obstáculo, pues la plataforma ThingsBoard y su *gateway* IoT permiten la instalación en estos dispositivos sin comprometer la estabilidad.

Para efectuar una comprobación de la aplicación IoT propuesta, se selecciona la topología de WSN mostrada en la figura 3.1 b), ya que posee características similares a la implementación real que se desea realizar. Durante 65 minutos, se simulan los nueve nodos en Cooja, se envían los datos cada 5 minutos al bróker RSMB y este los reenvía a la plataforma IoT. Por tanto, si todo funciona como es debido, el bróker RSMB debe recibir y enviar 117 mensajes (cada nodo publica 13 veces durante los 65 minutos), la plataforma debe visualizar los valores de telemetría (temperatura, humedad, intensidad luminosa, etc.) y estos deben quedar correctamente almacenados en la base de datos. En la figura 3.2 se muestra la información proporcionada por el bróker RSMB al finalizar la prueba, donde se destacan los 117 mensajes recibidos (*messages received*) y enviados (*messages sent*), el tiempo de actividad del bróker (*uptime*) y la desconexión de los nodos al detener RSMB. El tiempo de actividad del bróker es menor que los 65 minutos de la prueba, puesto que en los 5 minutos iniciales no se publica ningún mensaje.

```
20190605 145530.105 CWNAN0301I MQTT-S protocol stopping
20190605 145530.105 CWNAN0017I Closing client C10C00000000000A
20190605 145530.105 4 fd00::c30c:0:0:a:1884 C10C00000000000A -> MQTT-S DISCONNECT duration: 0 (0)
20190605 145530.105 CWNAN0017I Closing client C10C000000000009
20190605 145530.105 4 fd00::c30c:0:0:9:1884 C10C000000000009 -> MQTT-S DISCONNECT duration: 0 (0)
20190605 145530.105 CWNAN0017I Closing client C10C000000000008
20190605 145530.105 4 fd00::c30c:0:0:8:1884 C10C000000000008 -> MQTT-S DISCONNECT duration: 0 (0)
20190605 145530.105 CWNAN0017I Closing client C10C000000000007
20190605 145530.105 4 fd00::c30c:0:0:7:1884 C10C000000000007 -> MQTT-S DISCONNECT duration: 0 (0)
20190605 145530.105 CWNAN0017I Closing client C10C000000000006
20190605 145530.105 4 fd00::c30c:0:0:6:1884 C10C000000000006 -> MQTT-S DISCONNECT duration: 0 (0)
20190605 145530.105 CWNAN0017I Closing client C10C000000000005
20190605 145530.105 4 fd00::c30c:0:0:5:1884 C10C000000000005 -> MQTT-S DISCONNECT duration: 0 (0)
20190605 145530.105 CWNAN0017I Closing client C10C000000000004
20190605 145530.105 4 fd00::c30c:0:0:4:1884 C10C000000000004 -> MQTT-S DISCONNECT duration: 0 (0)
20190605 145530.105 CWNAN0017I Closing client C10C000000000003
20190605 145530.105 4 fd00::c30c:0:0:3:1884 C10C000000000003 -> MQTT-S DISCONNECT duration: 0 (0)
20190605 145530.106 CWNAN0017I Closing client C10C000000000002
20190605 145530.106 4 fd00::c30c:0:0:2:1884 C10C000000000002 -> MQTT-S DISCONNECT duration: 0 (0)
20190605 145530.106 CWNAN0044I Messages sent: 117
20190605 145530.106 CWNAN0043I Messages received: 117
20190605 145530.106 CWNAN0042I Uptime: 3728 seconds
20190605 145530.106 CWNAN0055I Maximum heap use: 149152 bytes
20190605 145530.106 CWNAN0047I Broker stopped
```

Figura 3.2. Información suministrada por el bróker RSMB.

La visualización de los datos enviados por el bróker hacia la plataforma ThingsBoard se muestra en las figuras 3.3 y 3.4. La figura 3.3 es el panel correspondiente al nodo 2 donde se exponen los valores de la temperatura del aire, la temperatura del dispositivo Zolertia, la humedad relativa del aire y la intensidad luminosa. En la figura 3.4 se ilustran los datos históricos de variables ambientales de interés, seleccionando ciertos nodos para mejorar la apariencia visual del panel. Algunos de los datos obtenidos de la base de datos PostgreSQL se muestran en la figura 3.5. Se ha seleccionado el nodo 2 y la temperatura del aire enviada por el mismo para comprobar el almacenamiento de la información y el intervalo de monitoreo de 5 minutos durante la prueba realizada.

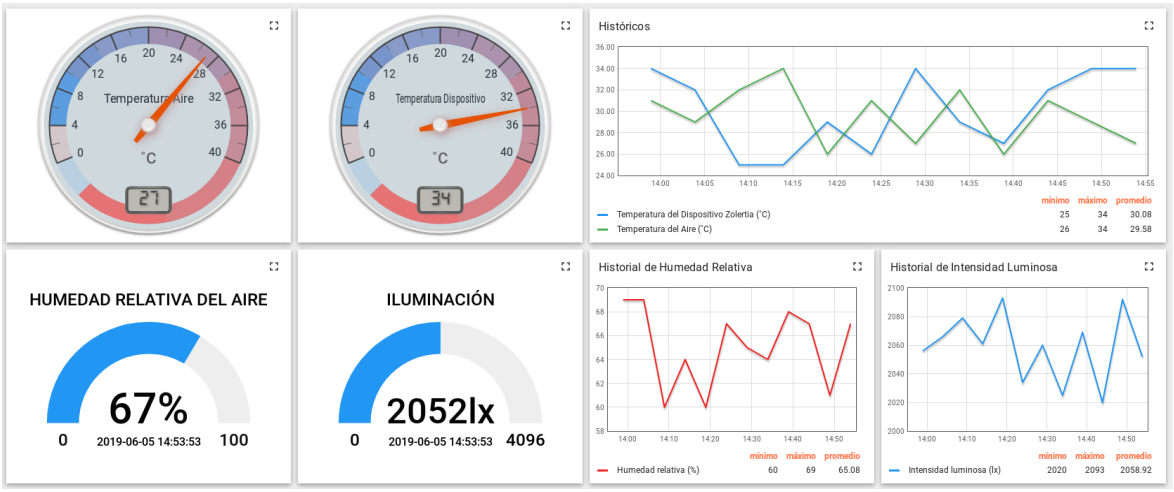


Figura 3.3. Panel del Nodo 2 en la plataforma IoT ThingsBoard.

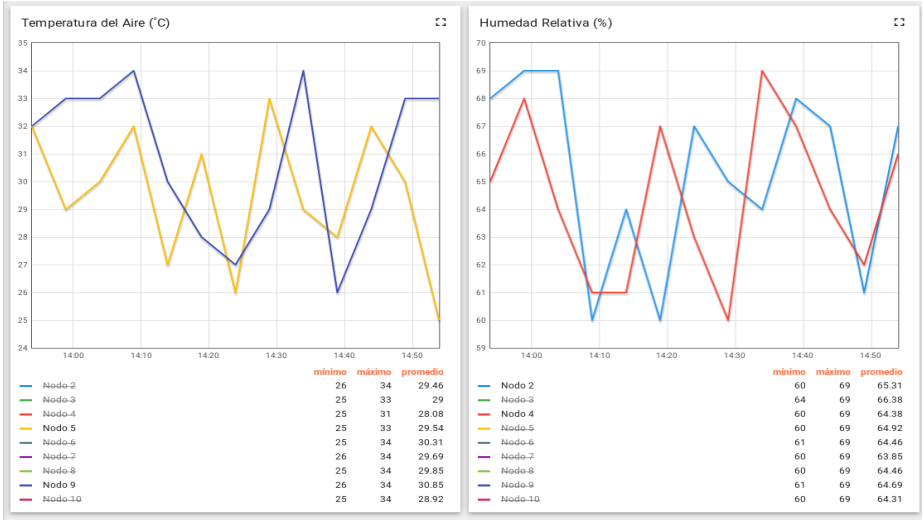


Figura 3.4. Fragmento del panel de históricos en la plataforma IoT ThingsBoard.

Nodo	Variable	Fecha	Valor
Nodo 2	Temperatura del Aire	06/05/19 01:53 PM	28
Nodo 2	Temperatura del Aire	06/05/19 01:58 PM	31
Nodo 2	Temperatura del Aire	06/05/19 02:03 PM	29
Nodo 2	Temperatura del Aire	06/05/19 02:08 PM	32
Nodo 2	Temperatura del Aire	06/05/19 02:13 PM	34
Nodo 2	Temperatura del Aire	06/05/19 02:18 PM	26
Nodo 2	Temperatura del Aire	06/05/19 02:23 PM	31
Nodo 2	Temperatura del Aire	06/05/19 02:28 PM	27
Nodo 2	Temperatura del Aire	06/05/19 02:33 PM	32
Nodo 2	Temperatura del Aire	06/05/19 02:38 PM	26
Nodo 2	Temperatura del Aire	06/05/19 02:43 PM	31
Nodo 2	Temperatura del Aire	06/05/19 02:48 PM	29
Nodo 2	Temperatura del Aire	06/05/19 02:53 PM	27

Figura 3.5. Datos obtenidos de la base de datos PostgreSQL.

3.2 Pruebas de rendimiento computacional de la Raspberry Pi 3 Modelo B

La plataforma IoT ThingsBoard y el *gateway* asociado a la misma se ejecutan en la Raspberry Pi 3 Modelo B, siendo los programas que más recursos computacionales consumen dentro de la aplicación IoT. En este contexto, se realiza un monitoreo durante una hora de los principales componentes relacionados con el procesamiento de los datos, como son la memoria RAM y la CPU, además de chequear la temperatura de esta última para determinar si existe sobrecalentamiento.

En la figura 3.6 se observa el comportamiento de la memoria RAM al ejecutar la plataforma y el *gateway* IoT. La línea amarilla representa el total de memoria (862 MB) que se puede utilizar en la Raspberry Pi, la línea verde representa la memoria disponible para ser usada y la línea amarilla figura la memoria libre del sistema. La memoria libre es la que no contiene ningún dato de relevancia para el sistema y puede ser usada en cualquier momento, mientras que la memoria disponible incluye a la memoria libre y a la memoria en caché, pues posibilita que el sistema operativo cargue archivos que podría necesitar en el futuro. Por tanto, el parámetro que nos interesa en este caso es la memoria RAM disponible. Al iniciar la prueba la memoria disponible es cercana a los 700 MB y al ejecutar ThingsBoard se observa una marcada disminución de la misma hasta los 400 MB. Posteriormente se ejecuta el *gateway* IoT y se produce otra disminución hasta cerca de los 200 MB (262MB). Una vez funcionando estos *software* y la aplicación IoT en su conjunto,

la memoria disponible se mantiene prácticamente invariable, confirmando la estabilidad del sistema en cuanto al uso de memoria RAM.

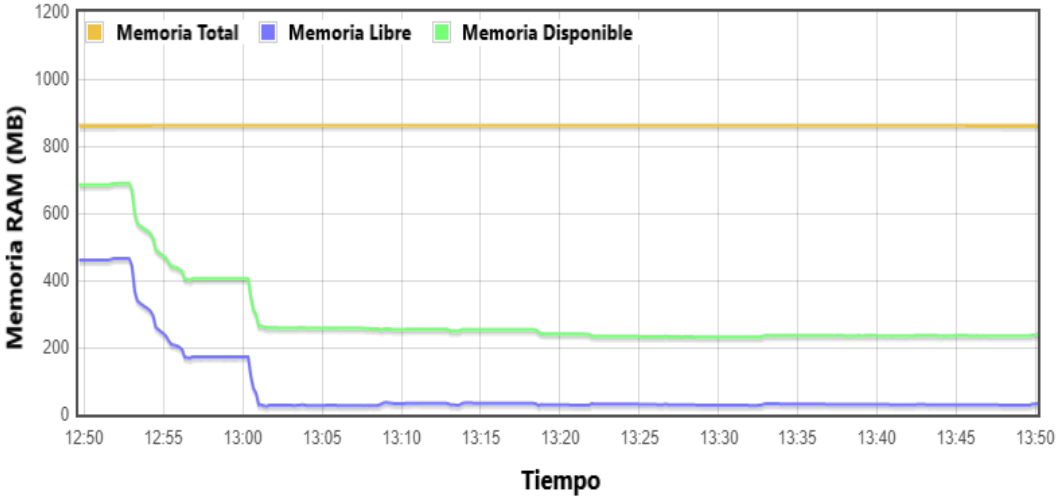


Figura 3.6. Consumo de RAM durante a la ejecución de la aplicación IoT.

La carga de la CPU se visualiza en la figura 3.7, a partir de valores históricos durante uno, cinco y quince minutos. Los datos obtenidos representan la carga media de CPU, en la cual se tienen en cuenta los procesos que se están ejecutando y los que se encuentran a la espera de que se les asigne tiempo de CPU. Esto significa que, para dispositivos con un procesador de un solo núcleo, una carga de CPU de 1.0 se traduce en un 100% de uso del mismo. En este caso, la Raspberry Pi cuenta con un procesador de 4 núcleos, por lo que, si se usaran todos, el valor de la carga de CPU sería de 4.0. Un análisis de la figura muestra que, durante la hora de prueba, las mayores cargas en un minuto se presentaron durante el proceso de ejecución de la plataforma y del *gateway* IoT, mientras que los otros picos corresponden al acceso a la interfaz gráfica de la plataforma. Además, con el paso del tiempo, la carga de la CPU disminuye a pesar de realizar las mismas acciones, debido a que los datos procesados previamente se mantienen en la memoria caché y son reutilizados cuando es necesario. En contraste con lo anteriormente descrito, las cargas durante cinco y quince minutos, evidencian el poco uso de CPU de la aplicación IoT durante su funcionamiento, principalmente porque no se ejecutan tareas de procesamiento inteligente de la telemetría recibida de los nodos sensores.

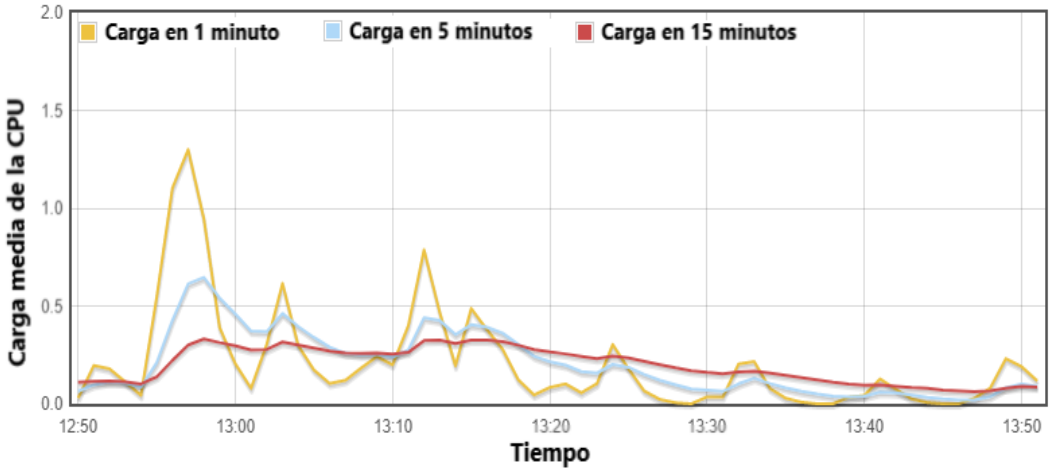


Figura 3.7. Carga de la CPU asociada a la ejecución de la aplicación IoT.

Otro parámetro de suma importancia es la temperatura de trabajo de la CPU, pues no es recomendable que la misma exceda los 80 ℃. En la figura 3.8 se observa el aumento que se produce al iniciar la plataforma y el *gateway* IoT, llegando a superar los 55 ℃, que, si bien es permisible, en un ambiente con condiciones desfavorables esta temperatura pudiera ser mayor. Con los *software* principales ejecutándose correctamente la temperatura se estabiliza alrededor de los 55 ℃, mientras que la disminución que se produce se debe a la incorporación de un ventilador. Con la adición de un fan que facilita la circulación del aire, la temperatura desciende hasta los 41 ℃, evitando posibles daños a la Raspberry Pi por motivos de sobrecalentamiento.

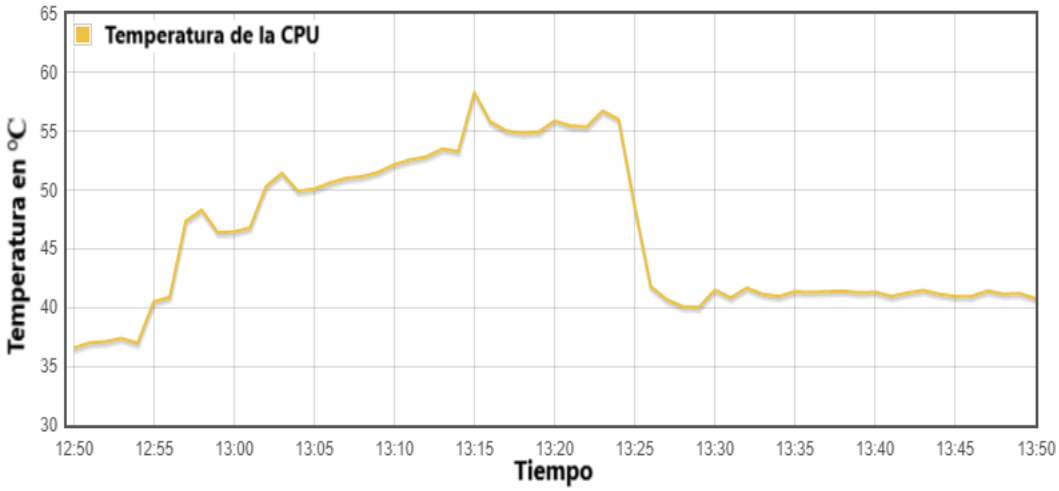


Figura 3.8. Temperatura de la CPU durante la ejecución de la aplicación IoT.

3.3 Despliegue de la aplicación IoT diseñada

Para valorar el funcionamiento de la aplicación IoT bajo condiciones reales, en cuanto a calidad de la conexión, enrutamiento de los nodos y al envío de datos de los mismos hacia la plataforma ThingsBoard, se realizó un despliegue del sistema en la UEB de Cultivos Protegidos "Valle del Yabú" de Santa Clara, Villa Clara. Se estableció una WSN de 6 nodos, 4 de ellos nodos sensores (figura 3.5a), uno destinado al enrutamiento solamente (sin sensores) y el nodo enrutador de frontera. En la caseta de fertirriego se instaló el nodo enrutador de frontera junto con la Raspberry Pi 3 Modelo B (figura 3.5b) que ejecuta la plataforma IoT y el bróker RSMB.

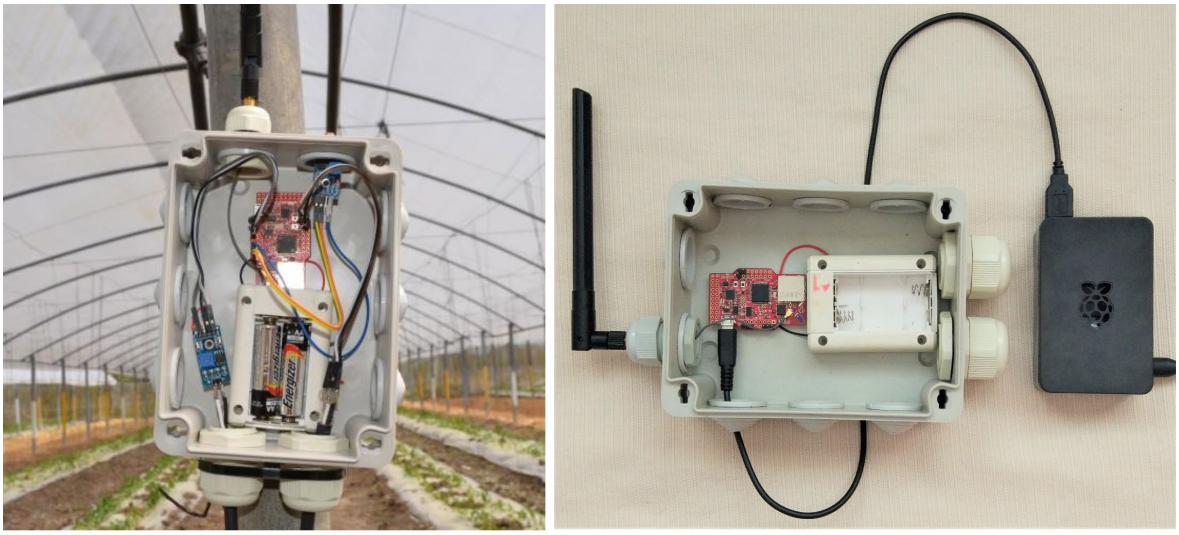


Figura 3.9. Nodo sensor (a) y nodo enrutador de frontera conectado a la Raspberry Pi (b).

3.3.1 Resultados obtenidos en relación al enrutamiento de los nodos

Inicialmente, se desplegaron 4 nodos sensores en la casa de cultivo 5 para determinar si se establecía conexión con el *border router*, pero debido a la lejanía del mismo (220 m aproximadamente) y a que el alcance de radio de los Zolertias Z1 es de 150 m, no se estableció un enlace directo. Por tanto, se decidió mover uno de los nodos hasta el invernadero 11 y colocar otro exclusivamente para enrutamiento en la casa de cultivo 12. De esta forma, los nodos enrutaron correctamente como se muestra en la tabla 3.3, cuyos datos han sido obtenidos de la interfaz web creada por el enrutador de frontera. En la figura 3.6 se puede observar la distribución geográfica de cada componente de *hardware* utilizado.

Tabla 3.3. Estado de los nodos desplegados y su enrutamiento.

Nodos Vecinos	Nodos Enrutados	Ruta
Nodo 7 (fe80::c30c:0:0:7) Nodo 8 (fe80::c30c:0:0:8)	Nodo 2 (fd00::c30c:0:0:2) Nodo 5 (fd00::c30c:0:0:5) Nodo 6 (fd00::c30c:0:0:6)	Nodo 7 (fe80::c30c:0:0:7) Nodo 8 (fe80::c30c:0:0:8) Nodo 7 (fe80::c30c:0:0:7)

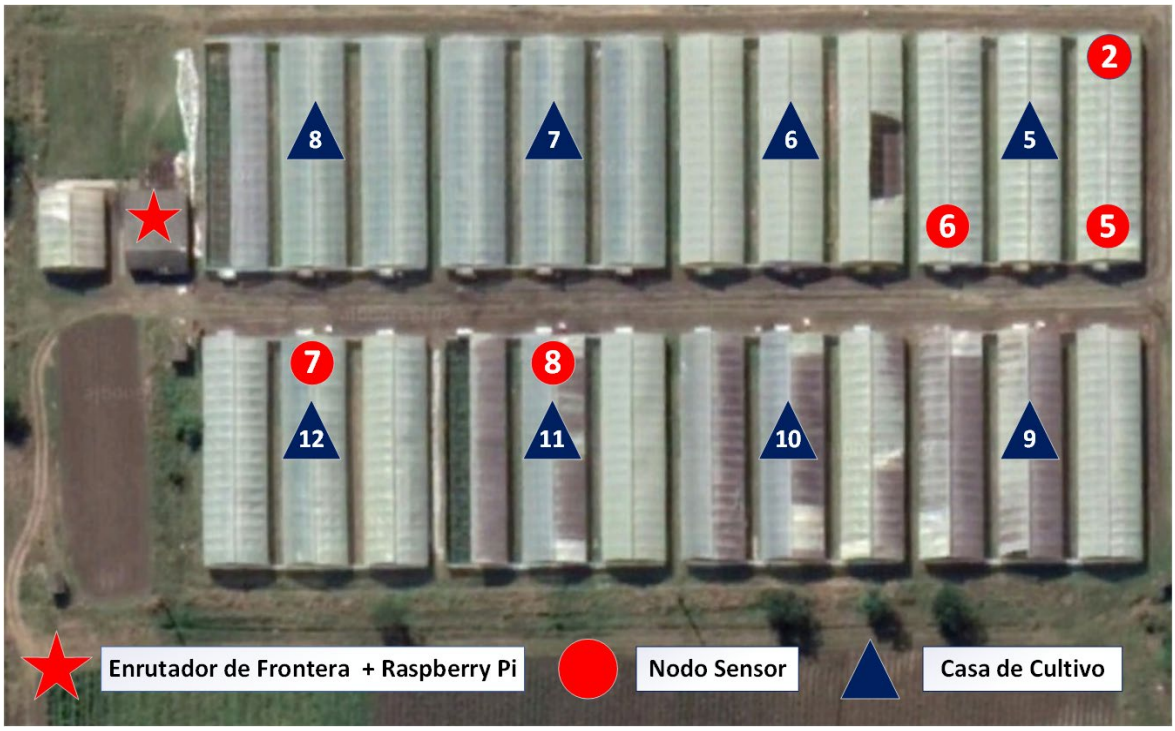


Figura 3.10. Ubicación de los dispositivos de *hardware* en las casas de cultivo.

3.3.2 Funcionamiento de la aplicación IoT diseñada

La obtención de los datos se llevó a cabo durante una 1 hora y 45 minutos aproximadamente, en intervalos de 15 minutos entre cada medición de parámetros. Las variables monitoreadas fueron la temperatura del aire, la humedad relativa, la temperatura del dispositivo Zolertia Z1 y la intensidad luminosa[12]. La figura 3.7 muestra el panel de la plataforma ThingsBoard correspondiente al nodo 8 y en la figura 3.8 se visualiza la

[12] La conversión de voltaje (salida del sensor) a unidades de intensidad luminosidad (lux) no fue posible debido a que, al realizar las pruebas, no se contaba con la ecuación correspondiente al sensor utilizado.

temperatura del aire recopilada por cada uno de los nodos. En los paneles que proporciona la plataforma IoT ThingsBoard, los valores históricos pueden ser consultados en tiempo real o en un intervalo temporal específico. Estos datos pueden ir acompañados de los valores máximos, mínimos y el promedio de cada variable.

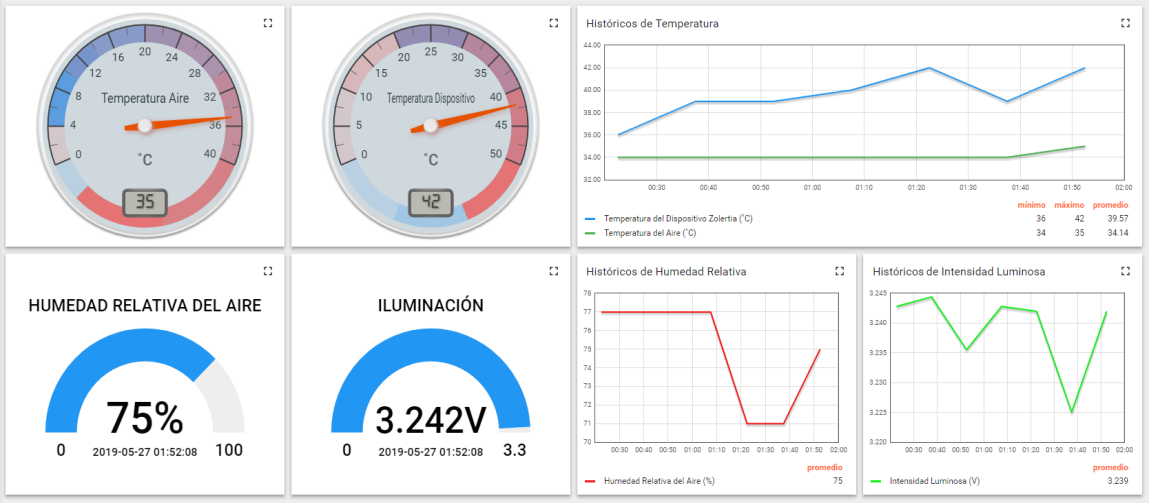


Figura 3.11. Panel del nodo 8 con valores reales obtenidos durante las pruebas realizadas.

Un análisis de la telemetría del nodo 8, confirma la validez de la misma, pues el despliegue de la aplicación IoT se desarrolló al mediodía, donde las temperaturas y la humedad relativa son altas debido a la fuerte incidencia de la luz solar. Además, la temperatura del dispositivo Zolertia es más alta que la temperatura del aire, puesto que este se encuentra funcionando dentro de una caja protectora susceptible al calentamiento por los rayos solares. Las variaciones que experimenta la salida del sensor de intensidad luminosa son del orden de los mV (debido a las nubes fundamentalmente), y a pesar de no contar con la medida de la intensidad luminosa, se puede inferir que es un día soleado, pues la salida de voltaje de dicho sensor se aproxima a su mínimo valor[13]. Los nodos 2, 5 y 6, que se colocaron en la casa de cultivo 5 envían valores de temperatura del aire diferentes, en correspondencia con una mayor o menor exposición de los sensores a la radiación solar o a una incorrecta calibración de los mismos. Los paneles relacionados a cada uno de los nodos anteriores se exponen en los anexos VI, VII y VIII.

[13] La salida de voltaje de estos sensores es inversa a la iluminación. Los paneles de visualización muestran una respuesta directa por decisión del autor.

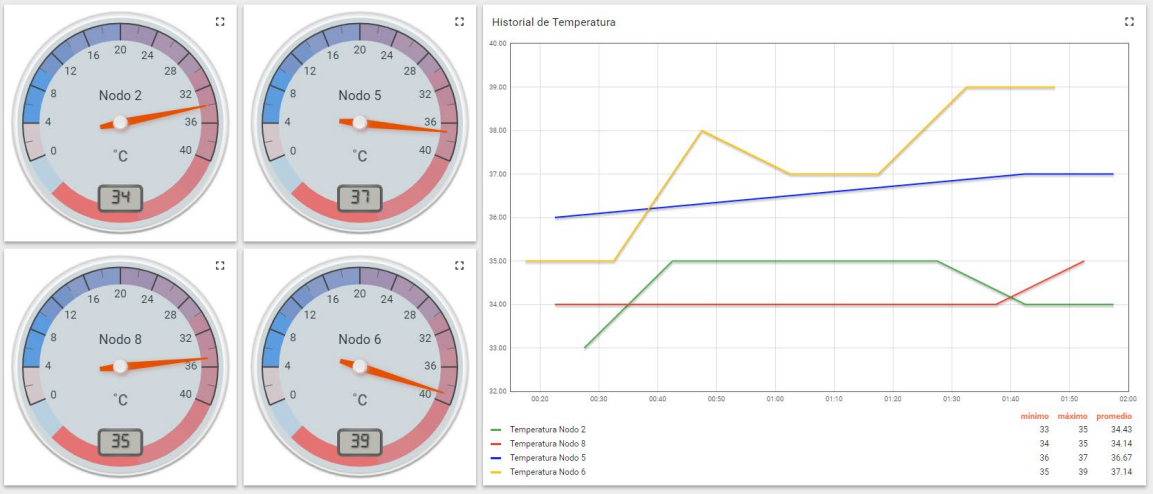


Figura 3.12. Panel con valores de la temperatura del aire obtenidos por cada nodo.

En cuanto al envío de los datos, todos los nodos conectados enviaron las mediciones realizadas, exceptuando el nodo 5 que sufrió una desconexión al inicio de la prueba y ante varios intentos fallidos de reconexión automática tuvo que ser reiniciado manualmente. Después de ese inconveniente, no se produjeron otros eventos de esa naturaleza. Lo anterior demuestra la necesidad de experimentos reales, pudiendo ser detectados problemas que en ambientes simulados no es posible predecir. Se debe destacar, como otro resultado positivo de las pruebas realizadas, la confirmación de que se puede cubrir toda el área de las casas de cultivo, siempre y cuando se coloquen nodos que permitan el enrutamiento hacia el nodo final.

3.4 Análisis económico y medioambiental de la propuesta de aplicación IoT

La aplicación IoT para casas de cultivo diseñada, tiene entre sus ventajas fundamentales, que todo el *software* utilizado es libre, por lo que no se necesita pagar ninguna licencia para el uso de los mismos, incluso, si se utilizan en aplicaciones industriales. Por ejemplo, la plataforma IoT ThingsBoard está licenciada bajo Apache License 2.0, que permite al usuario del *software* la libertad de emplearlo para cualquier propósito, distribuirlo, modificarlo, y distribuir versiones modificadas del mismo (*The Apache Software Foundation*, 2019).

El presente trabajo es la continuación de proyectos previos realizados por el Grupo de Internet de las Cosas, Automatización e Inteligencia Artificial perteneciente a la

Universidad Central "Marta Abreu" de Las Villas, por lo que el *hardware* empleado en la aplicación IoT diseñada, fue seleccionado en (Estevez Pérez, 2018), donde se establece que para nueve nodos de medición, el costo total de la inversión es de 1314.70 USD. Además, se compara la arquitectura de *hardware* diseñada con otras plataformas disponibles en el mercado, cuyo valor asciende hasta los 3000 USD cuando se incluyen los sensores y el *gateway.* Por tanto, la propuesta de aplicación IoT realizada presenta ventajas económicas reduciendo el costo sin afectar el funcionamiento.

En cuanto al aspecto medioambiental, la AP y las modernas tecnologías de WSN, proporcionan grandes beneficios a la sociedad, relacionados con la conservación del agua, la optimización de recursos energéticos, la prevención de la contaminación, la aplicación exacta de productos, el diagnóstico de enfermedades en los cultivos, entre otras. Con la implementación de la aplicación IoT diseñada en las casas de cultivo de la UEB de Cultivos Protegidos "Valle del Yabú", uno de los parámetros que se podrán monitorear es la humedad del suelo, importante a la hora de establecer un sistema de riego eficiente. Esto contribuiría al ahorro de agua, fundamental para afrontar períodos de intensa sequía y no derrochar un recurso natural tan necesario para el desarrollo de la sociedad. Otras variables ambientales monitoreadas (temperatura, humedad del aire, intensidad luminosa), facilitan la aplicación correcta de fertilizantes atendiendo a las necesidades de un determinado cultivo, previniendo la contaminación debido a nitratos, los efectos ligados a los abonos inutilizados o la proliferación de parásitos en los cultivos. Además, con la reducción del uso de pesticidas, se evita la contaminación de las aguas y el impacto negativo que pueden tener en otras especies, tanto animales como vegetales. Por otra parte, la aplicación IoT objetivo de la presente investigación constituye una propuesta sostenible y "amigable" con el medio ambiente, al no contener ningún producto contaminante ni emitir radiofrecuencias que dañen el ecosistema circundante.

3.5 Consideraciones finales del capítulo

Como resultado de las pruebas realizadas en este capítulo se puede afirmar que la herramienta Cooja ofrece un entorno de simulación para WSN confiable y útil para comprobar aplicaciones IoT antes de implementarlas en el *hardware*. También, que la Raspberry Pi 3 Modelo B presenta grandes potencialidades relacionadas con su reducido

tamaño, potencia de procesamiento, almacenamiento y comunicación mediante diversas interfaces. Respecto a la aplicación IoT para casas de cultivo diseñada, se concluye que los *software* que la componen se integran perfectamente, logrando un funcionamiento que cumple con las expectativas de establecer un sistema de supervisión de parámetros ambientales en un servidor local, con tecnologías innovadoras y con un impacto económico y ambiental favorable para la sociedad.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

Con el diseño de la aplicación IoT para su empleo en las casas de cultivo pertenecientes a la UEB de Cultivos Protegidos “Valle del Yabú”, se arriban a las siguientes conclusiones:

1. La capacidad de implementación en un servidor local alojado en la Raspberry Pi, sin necesidad de conexión a internet y el soporte de los protocolos de comunicación más utilizados en el IoT, hacen la selección de la plataforma IoT ThingsBoard adecuada para la aplicación de monitoreo propuesta.
2. El empleo del protocolo de comunicación MQTT, del bróker de mensajería RSMB y del *gateway* IoT de ThingsBoard permite una conexión ligera, con bajo consumo de recursos computacionales y energéticos, y garantiza el envío de los datos en la aplicación IoT diseñada.
3. Con los resultados obtenidos en la simulación de la aplicación IoT y las pruebas reales efectuadas en las casas de cultivo de la UEB de Cultivos Protegidos “Valle del Yabú”, se valora positivamente la supervisión de los parámetros ambientales de interés, lográndose el enrutamiento de los nodos y una adecuada visualización y almacenamiento de los datos, que permitirá una correcta toma de decisiones de los expertos.

Recomendaciones

1. Implementar la aplicación IoT diseñada en las casas de cultivo de la UEB de Cultivos Protegidos “Valle del Yabú”.

REFERENCIAS BIBLIOGRÁFICAS

Amondaray, L. R. *et al.* (2018) «Red de sensores inalámbricos para las casas de cultivos protegidos " San José "», *Revista de Ingeniería Electrónica, Automatica y Comunicaciones (RIELAC)*. La Habana, Cuba: Scielo, 39, pp. 16-26. Disponible en: http://scielo.sld.cu/ (Accedido: 8 de marzo de 2019).

Bauer, J. y Aschenbruck, N. (2018) «IoT Vertical and Topical Summit on Agriculture - Tuscany (IOT Tuscany)», en *Design and Implementation of an Agricultural Monitoring System for Smart Farming*. IEEE.

Beltrán Martínez, D. A. y Ortiz Barajas, E. D. (2018) *Configuración y puesta a punto de un sistema de monitoreo basado en tecnologias de IoT.* Pontificia Universidad Javeriana de Cali.

Bitencourt, E. N. (2018) «IoT Centralization and Management Applying ThingsBoard Platform», *Smart Services Research Unit*. Brazil.

Cadavid, H. *et al.* (2018) «Towards a Smart Farming Platform: From IoT-Based Crop Sensing to Data Analytics», en *Colombian Conference on Computing*. Springer, pp. 237-251. doi: 10.1007/978-3-319-98998-3_19.

Caicedo Ortiz, J. G. *et al.* (2018) «Monitoring system for agronomic variables based in WSN technology on cassava crops», *Computers and Electronics in Agriculture*. Elsevier, 145(January), pp. 275-281. doi: 10.1016/j.compag.2018.01.004.

Cao-hoang, T. y Nguyen Duy, C. (2017) «Seventh International Conference on Information Science and Technology», en *Environment Monitoring System for Agricultural Application Based on Wireless Sensor Network*. Da Nang, Vietnam: IEEE.

Cirani, S. y Ferrari, G. (2019) *Internet of Things. Architectures, Protocols and Standards*, *John Wiley & Sons Ltd*. Hoboken, USA.

Cisco (2014) *The Internet of Things Reference Model*. Disponible en: http://cdn.iotwf.com/resources/71/IoT_Reference_Model_White_Paper_June_4_2014.pdf.

Estevez Pérez, A. A. (2018) *Diseño de Red de Sensores Inalámbricos para aplicación de Internet de las Cosas en casas de cultivos*. Universidad Central «Marta Abreu» de LAs Villas.

Ferrández Pastor, F. J., García Chamizo, J. M. y Nieto Hidalgo, M. (2016) «Developing Ubiquitous Sensor Network Platform Using Internet of Things: Application in Precision Agriculture», *Sensors*. Alicante,, Spain;, pp. 15-20. doi: 10.3390/s16071141.

Foughali, K., Fathallah, K. y Frihida, A. (2018) «9th International Conference on Ambient Systems, Networks and Technologies, ANT-2018 and The 9th International Conference on Ambient Systems, Networks and Technologies The 9th International Conference on Ambient Systems, Networks and Technologies the 8th», en *Using Cloud IOT for disease prevention in precision agriculture*. Elsevier B.V., pp. 575-582. doi: 10.1016/j.procs.2018.04.106.

Fraga Castro, A. (2015) *Simulador COOJA para WSN basado en el sistema operativo Contiki*. Editado por C. M. García Algora. Universidad Central “Marta Abreu” de Las Villas.

Guerrero Ibañez, J. A. *et al.* (2017) «SGreenH-IoT : Plataforma IoT para Agricultura de Precisión .», *Sistemas, Cibernética e Informática*. México, 14, pp. 53-58.

Guerrero Nicolas, A. y Cano Rigol, C. (2017) *Sistema distribuido para medida y procesado en tiempo real de sensores IoT*. Universitat Politècnica de Catalunya.

Hassan, Q. y Rehman, A. (2018) *Internet of things: Challenges, advances, and applications*, *CRC Press*. Chapman and Hall/CRC (Chapman & Hall/CRC Computer and Information Science).

Hernández Rojas, L., Mazon Olivo, E. y Campoverde Marca, M. (2015) «Cloud Computing para el Internet de las Cosas. Caso de estudio orientado a la Agricultura de Precisión.»,

en *Memoria de Artículos del Primer Congreso de Ciencia y Tecnología UTMACH 2015.* Machala, Ecuador: Primer Congreso de Ciencia y Tecnología UTMACH 2015., pp. 48-49.

IIC (2017) «The Industrial Internet of Things Volume G1: Reference Architecture», *Industrial Internet Consortium*. Editado por S.-W. Lin, M. Crawford, y S. Mellor. Disponible en: www.iiconsortium.org.

Jaffey, T. (2014) *MQTT and CoAP, IoT Protocols*, *Eclipse Fundation*. Disponible en: https://www.eclipse.org/community/eclipse_newsletter/archives.php.

Katsikeas, S. *et al.* (2017) «2017 IEEE Symposium on Computers and Communications (ISCC)», en *Lightweight & Secure Industrial IoT Communications via the MQ Telemetry Transport Protocol*. IEEE.

Li, Y. (2018) «An Integrated Platform for the Internet of Things Based on an Open Source Ecosystem», *Future Internet*. Nanjing, China, 10(11), pp. 11-15. doi: 10.3390/fi10110105.

Light, R. y Craggs, I. (2013) *Mosquitto*, *Eclipse Foundation*. Disponible en: www.eclipse.org/proposals/mosquitto/.

Liñan, A., Bagula, A. y Pietrosemoli, E. (2016) *Internet of Things in 5 days*, *Creative Commons*.

MarketsandMarkets.com (2018) *Precision Farming Market by Technology (Guidance System, VRT, Remote Sensing), Application (Crop Scouting, Field Mapping, Irrigation), Offering (Hardware-Sensors, GPS/GNSS, Yield Monitors, Software, Services), and Geography - Global Forecast to 2023*. Disponible en: http://www.marketsandmarkets.com/Market-Reports/precision-farming-market-1243.html.

Mat, I. *et al.* (2017) «IoT in Precision Agriculture applications using Wireless Moisture Sensor Network», *ICOS 2016 - 2016 IEEE Conference on Open Systems*. Kuala Lumpur, MALAYSIA, pp. 24-29. doi: 10.1109/ICOS.2016.7881983.

Minerva, R., Biru, A. y Domenico, R. (2015) «Towards a definition of the Internet of Things (IoT)», *IEEE Internet of Things*, pp. 72-75. Disponible en: iot.ieee.org.

Naik, N. (2017) «Choice of Effective Messaging Protocols for IoT Systems : MQTT , CoAP , AMQP and HTTP», *IEEE*. United Kingdom.

Nóbrega, L. *et al.* (2018) «Animal monitoring based on IoT technologies», *IoT Vertical and Topical Summit on Agriculture*. Aveiro,, Portugal: IEEE.

PostgreSQL (2019). The PostgreSQL Global Development Group. Disponible en: https://www.postgresql.org (Accedido: 10 de abril de 2019).

Prakash Jayaraman, P. *et al.* (2016) «Internet of Things Platform for Smart Farming : Experiences and Lessons Learnt», *Sensors*. Editado por S. X. Yang. Melbourne, Australia: MDPI, pp. 4-5. doi: 10.3390/s16111884.

Priya (2019) *Really Small Message Broker*, *www.engineersgarage.com*. Disponible en: www.engineersgarage.com/Tutorials/RSMB-Broker-for-MQTT-SN.

Quiñones Cuenca, M. *et al.* (2017) «Sistema De Monitoreo de Variables Medioambientales Usando Una Red de Sensores Inalámbricos y Plataformas De Internet De Las Cosas», *Enfoque UTE*. Loja, Ecuador, pp. 329-343. Disponible en: http://ingenieria.ute.edu.ec/enfoqueute/.

Ramírez Otero, C. A., López Pacheco, G. A. y Mendivelso Sanabria, C. F. (2018) «Extensión de la plataforma de fuente abierta thingsboard para el desarrollo de soluciones IOT para el agro Colombiano.» Escuela Colombiana de Ingeniería Julio Garavito, pp. 8-30.

Raspberry Pi Foundation (2016) *Raspberry Pi Hardware - Raspberry Pi Documentation*. Disponible en: https://www.raspberrypi.org/documentation/hardware/raspberrypi.

Raspbian (2019) *Raspbian OS*. Disponible en: www.raspbian.org.

Serpanos, D. y Wolf, M. (2018) *Internet-of-Things (IoT) Systems «Architectures, Algorithms, Methodologies»*. Atlanta, USA: Springer. doi: 10.1007/978-3-319-69715-4.

Soni, D. y Makwana, A. (2017) «International Conference On Telecomunication, Power Analysis And Computing Techniques (ICTPACT-2017)», en *A survey on MQTT: a protocol of Internet of Things (IoT)*.

Stanford-Clark, A. y Linh Truong, H. (2013) *MQTT For Sensor Networks (MQTT-SN) Protocol Specification*. International Business Machines Corporation (IBM).

The Apache Software Foundation (2019) *Apache License*. Disponible en: https://www.apache.org/licenses/LICENSE-2.0.txt.

ThingsBoard (2019) *ThingsBoard IoT Open Source Plataform*. Disponible en: https://thingsboard.io/ (Accedido: 1 de enero de 2019).

Tommaso De Paolis, L., De Luca, V. y Paiano, R. (2018) «Sensor data collection and analytics with ThingsBoard and Spark Streaming», *2018 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS)*. Lecce,, Italy: IEEE, pp. 1-6.

Triana Useche, J. C. y Rodríguez Leguizamo, R. E. (2018) *Prototipo de solución IoT con tecnología «LoRa» en monitoreo de cultivos agrícolas.* Universidad Distrital Francisco José de Caldas.

UIT (2012) «Descripción general de Internet de los objetos», *Recomendación UIT-T Y.2060*. UIT (Serie Y: Infraestructura mundial de la información, aspectos de protocolos de Internet y redes de la próxima generación). Disponible en: https://www.itu.int/ITU-T/recommendations/.

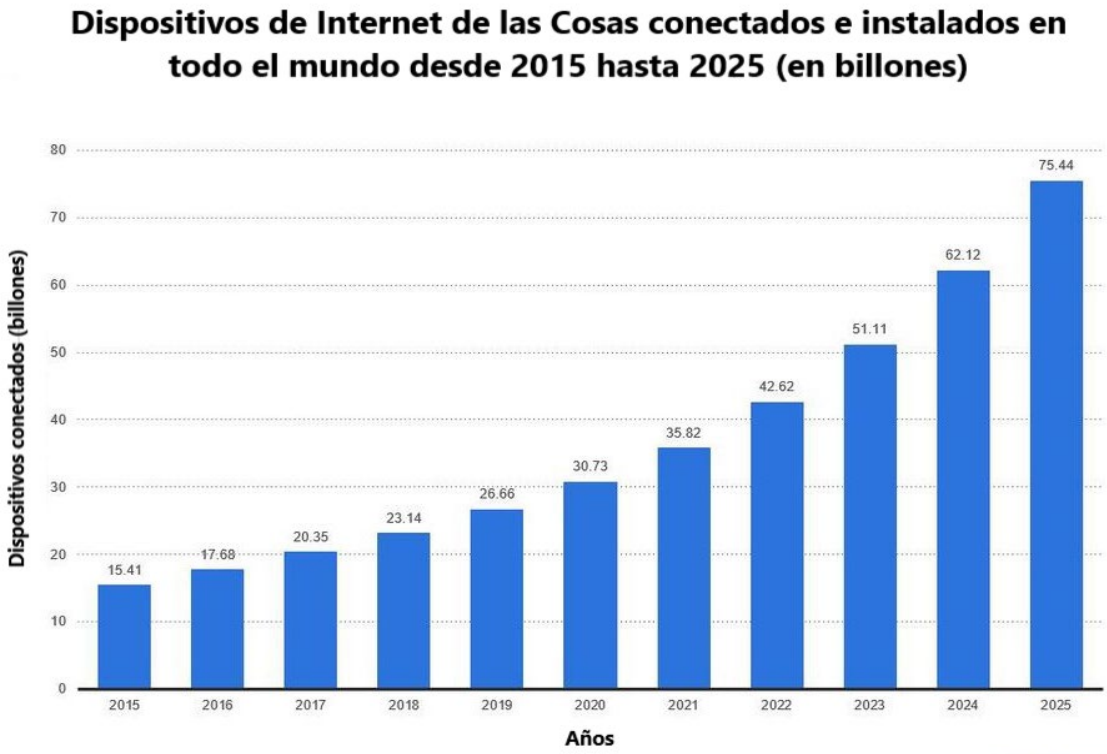
Weber, R. H. y Weber, R. (2010) *Legal Perspectives*, *Internet of Things*. Zurich, Suiza: Springer. doi: 10.1007/978-3-642-11710-7.

Zamora Izquierdo, M. A., Martínez, J. A. y Skarmeta, A. F. (2018) «Smart farming IoT platform based on edge and cloud computing», *Biosystems Engineering*. Spain: Elsevier Ltd, pp. 4-10. doi: https://doi.org/10.1016/j.biosystemseng.2018.10.014.

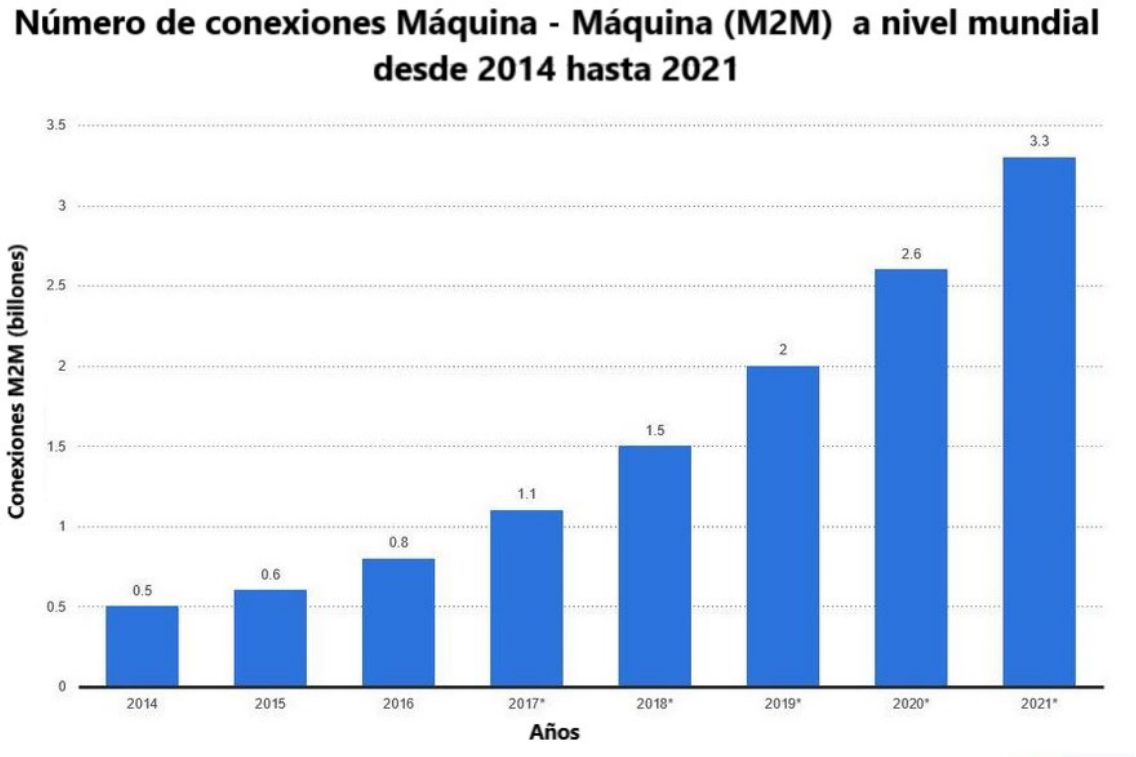
Zolertia (2010) «Z1 Datasheet», pp. 1-20.

ANEXOS

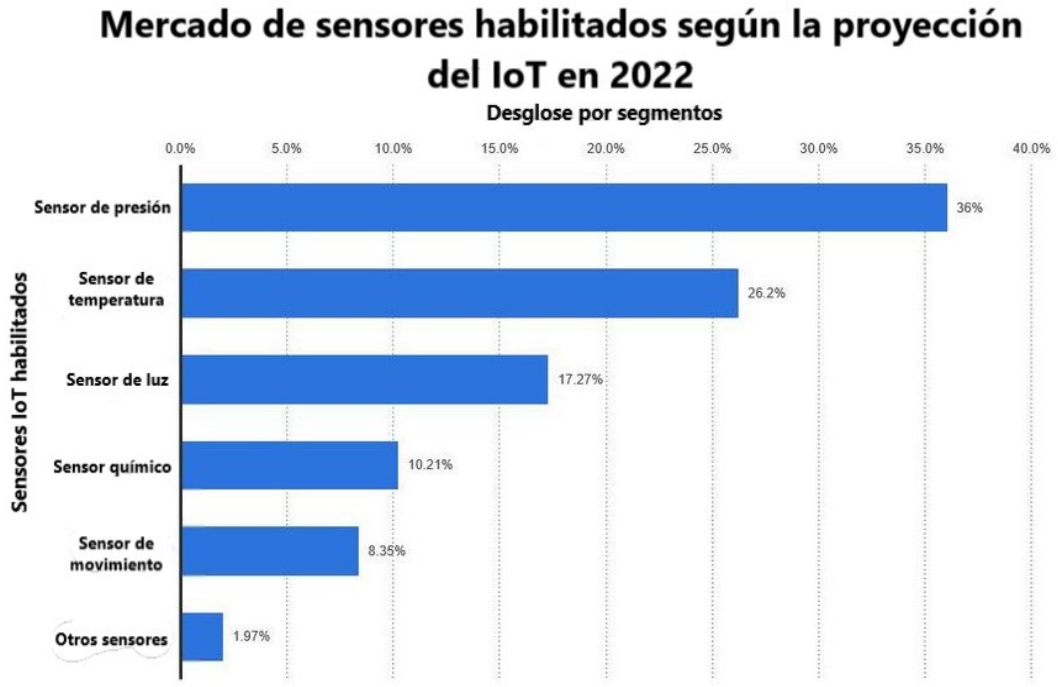
Anexo I Datos relevantes sobre la evolución del Internet de las Cosas.



Fuente: *https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/*



Fuente: *https://www.statista.com/statistics/487280/global-m2m-connections/*

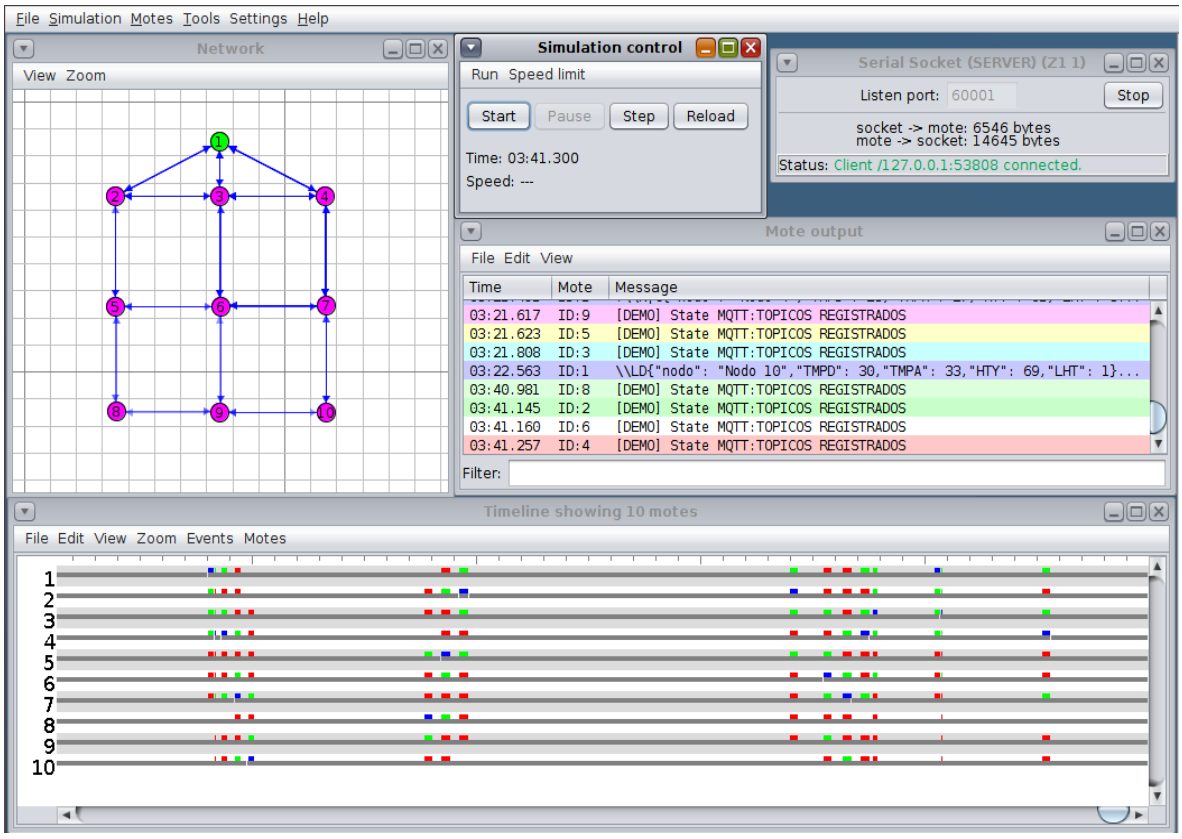


Fuente: *https://www.statista.com/statistics/480114/global-internet-of-things-enabled-sensors-market-size-by-segment/*

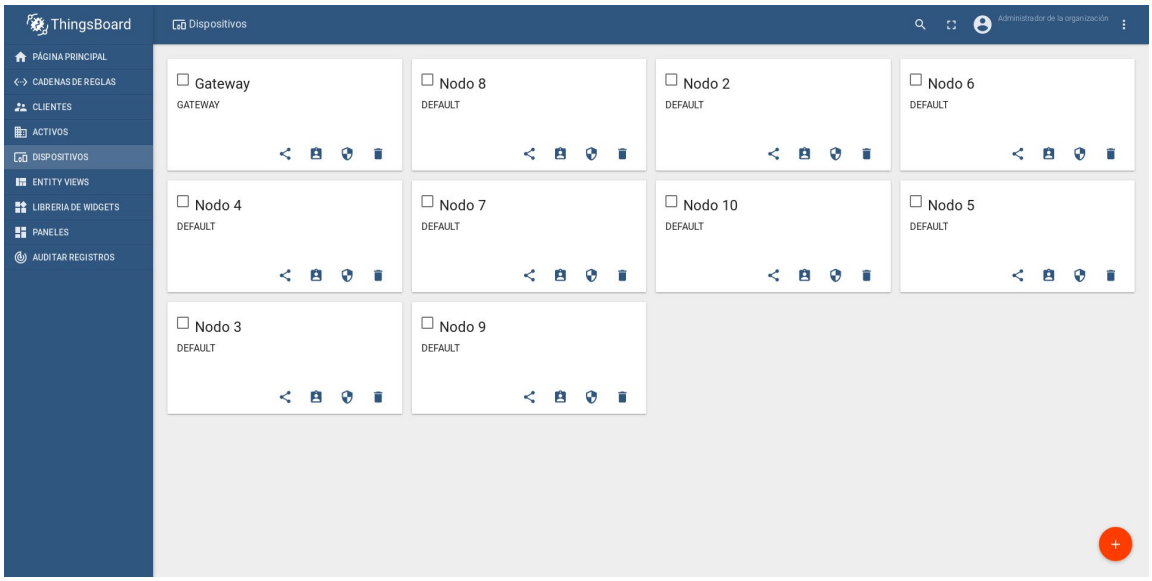
Anexo II Repositorios de algunos *software* utilizados.

Software	Repositorio
Bróker RSMB	https://github.com/eclipse/mosquitto.rsmb
Instant Contiki 3.0	https://sourceforge.net/projects/contiki/files/Instant_Contiki/Instant_20Contiki_203.0/
Contiki OS	https://github.com/contiki-os/contiki
MQTT-SN	https://github.com/aignacio/homestark_mqtt_6lowpan_port

Anexo III Interfaz gráfica del simulador de WSN Cooja.



Anexo IV Panel de dispositivos (simulados en Cooja) de la plataforma IoT ThingsBoard.



Anexo V Información suministrada por el nodo enrutador de frontera durante las pruebas reales.

Neighbors

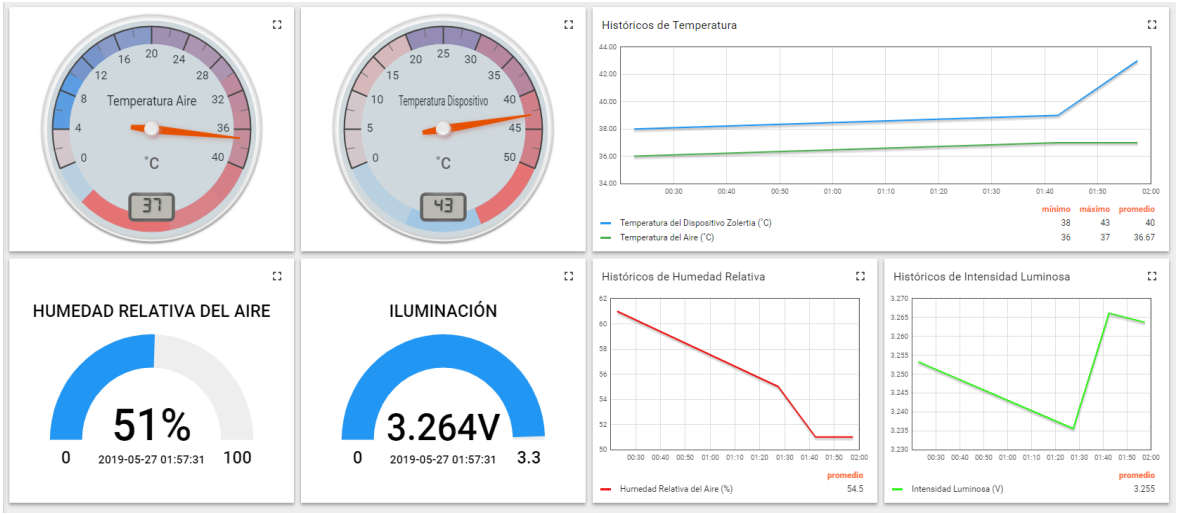
```
fe80::c30c:0:0:8
fe80::c30c:0:0:7
```

Routes

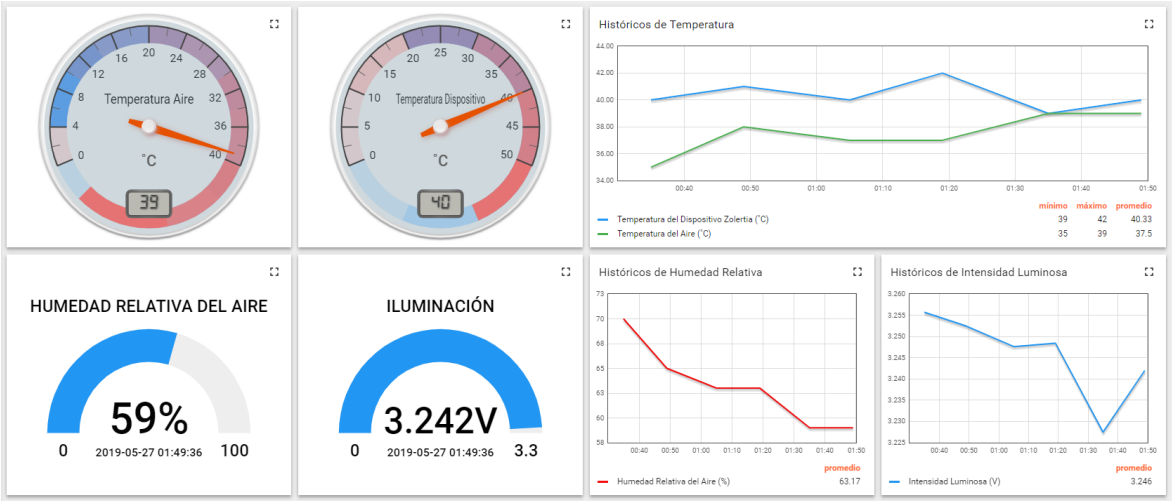
```
fd00::c30c:0:0:7/128 (via fe80::c30c:0:0:7)
fd00::c30c:0:0:5/128 (via fe80::c30c:0:0:8)
fd00::c30c:0:0:6/128 (via fe80::c30c:0:0:7)
fd00::c30c:0:0:8/128 (via fe80::c30c:0:0:8)
fd00::c30c:0:0:2/128 (via fe80::c30c:0:0:7)
```

Anexo VI Panel correspondiente al nodo 2 con valores reales obtenidos durante el despliegue de la aplicación IoT.

Anexo VII Panel correspondiente al nodo 5 con valores reales obtenidos durante el despliegue de la aplicación IoT.



Anexo VIII Panel correspondiente al nodo 6 con valores reales obtenidos durante el despliegue de la aplicación IoT.



Anexo IX Características eléctricas de los dispositivos Zolertia Z1.

Circuito Integrado	Rango de Operación	Consumo de Corriente	Notas
MSP430f2617	1.8V a 3.6V	0.1µA 0.5µA 0.5mA <10mA	Modo apagado Modo de espera Modo activo @1MHz Modo activo @16MHz
CC2420	2.1V a 3.6V	<1µA 20µA 426µA 18.8mA 17.4mA	Modo apagado Bajo consumo Modo inactivo Recepción Transmisión @ 0dBm
ADXL345	1.8V a 3.6V	0.1µA 40µA a 145µA	Espera Modo activo
M25P16	2.7V a 3.6V	1µA 4mA a 15mA	Ultra bajo consumo Modo activo
TMP102	1.4V a 3.6V	1µA 15µA	Modo apagado Modo activo

Anexo X Casas de cultivo de la UEB de Cultivos Protegidos “Valle del Yabú”.