AGRICULTURA DE PRECISIÓN PARA RIEGO INTELIGENTE DE CULTIVOS DE CAFÉ UTILIZANDO TECNOLOGÍAS DE INTERNET DE LAS COSAS

Monografía de Trabajo de Grado

Edinsson Alberto López López

Juan Diego Yasnó Collo

Director: PhD. Miguel Ángel Niño Zambrano

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Sistemas
Grupo I+D en Tecnologías de la Información - GTI
Popayán, julio de 2022

AGRICULTURA DE PRECISIÓN PARA RIEGO INTELIGENTE DE CULTIVOS DE CAFÉ UTILIZANDO TECNOLOGÍAS DE INTERNET DE LAS COSAS

Edinsson Alberto López López

Juan Diego Yasnó Collo

Trabajo de grado presentado a la Facultad de Ingeniería
Electrónica y Telecomunicaciones de la
Universidad del Cauca para obtención del
Título de:
Ingeniero de sistemas

Director: PhD. Miguel Ángel Niño Zambrano

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Sistemas
Grupo I+D en Tecnologías de la Información - GTI
Popayán, julio de 2022

TABLA DE CONTENIDO

1.1 Motivación trabajo de grado 2
1.2 Objetivos 3
1.2.1 Objetivo general 3
1.2.2 Objetivos específicos 3
1.3 Estructura trabajo de grado 4
Capítulo 2 Marco teórico 6
2.1 Variables agroclimáticas en los cultivos de café 6
2.2 Ecosistema IoT 7
2.3 Redes de sensores inalámbricos (WSN, Wireless Sensor Network) 8
2.3.1 Elementos de las WSN 9
2.3.2 Topología en WSN 10
2.3.3 Tecnologías inalámbricas en WSN 11
2.4 Métricas 12
2.5 Modelo 12
2.6 Protocolo MQTT 13
2.7 Agricultura de precisión (AP) 13
2.8 Objeto inteligente 14
2.9 Aprendizaje automático (Machine learning) 15
2.10 Trabajos relacionados 16
2.11 Estado del arte 19
2.11.1 Descripción general de los estudios seleccionados 19
2.11.2 WSN en agricultura de precisión 20
2.11.3 Mitigar o controlar la pérdida de datos en una WSN de objetos inteligentes 21
2.11.4 Objetos inteligentes en AP para el riego inteligente 23
Capítulo 3 Definición modelo de red de objetos inteligentes inalámbricos 27
3.1 Construcción del modelo de red de objetos inteligentes 27
3.2 Etapa 1: Determinar el ámbito del modelo 27
3.3 Etapa 2: Definir el modelo 28
Capa de aplicación 29
Capa de inteligencia y procesamiento de datos 30
Capa de adquisición de datos 31
3.4 Etapa 3: Diseño de la prueba preliminar del modelo 40

3.5 Algoritmos de inteligencia computacional que se pueden utilizar en el modelo 40
3.6 Integración del modelo a la arquitectura de Niño-Zambrano 41
3.7 Restricciones del modelo 43
Capítulo 4 Desarrollo prueba de concepto 45
4.1 Metodología para el desarrollo de la prueba de concepto: 45
4.2 Iteración 1 50
4.3 Iteración 2 57
4.4 Iteración 3 71
Capítulo 5 Conclusiones y trabajo futuro 76
5.1 Conclusiones en cuanto a productos obtenidos: 76
5.2 Conclusiones en cuanto a la investigación: 77
5.3 Trabajo futuro 77
BIBLIOGRAFÍA 78

LISTA DE FIGURAS

Figura 2 Arquitectura Nodo WSN Fuente: Propia....8
Figura 3 Arquitectura de interacción semántica Fuente:[21]....17
Figura 4 Histograma artículos por año Fuente: Propia....19
Figura 5 Calificación de artículos....20
Figura 6 Arquitectura modelo de red Fuente: Propia....29
Figura 7 Relieve central de comunicaciones y el cultivo Fuente: Google Maps....32
Figura 8 Distancia centro de comunicaciones a cultivo Fuente: Google Maps....33
Figura 9 Ubicación cultivo a monitorear Fuente: Google Maps....33
Figura 10 Topología en estrella Fuente: Propia....38
Figura 11 Topología en malla Fuente: Propia....38
Figura 12 Topología hibrida Fuente: Propia....39
Figura 13 Integración con la arquitectura de integración semántica Fuente: Propia....43
Figura 14 Ejecución ejemplo LoRa parámetro "sender" Fuente: Propia....55
Figura 15 Ejecución ejemplo LoRa parámetro "res" Fuente: Propia....55
Figura 16 Pines LoRa/GPS HAT Fuente:....56
Figura 17 Pines Raspberry Py Fuente: Propia....56
Figura 18 Cableado Pines LoRa HAT y Raspberry Pi Fuente: Propia....57
Figura 19 Montaje nodo de la malla Fuente: Propia....60
Figura 20 Pines analógicos habilitados....62
Figura 21 Cadena enviada desde el nodo....63
Figura 22 Suscribirse al bróker MQTT con el tópico indicado Fuente: Propia....63
Figura 23 Cadena capturada del bróker MQTT Fuente: Propia....63
Figura 24 Transformar la cadena capturada en un objeto JSON Fuente: Propia....63
Figura 25 Objeto JSON accedido Fuente: Propia....64
Figura 26 Insertar los datos del JSON en la base de datos Fuente: Propia....64
Figura 27 Datos insertados en la base de datos Fuente: Propia....65
Figura 28 Publicar tópico y obtener la lista de nodos Fuente: Propia....66
Figura 29 Suscribirse al tópico para obtener la lista de nodos Fuente: Propia....66
Figura 30 Lista de nodos actualizada Fuente: Propia....66
Figura 31 Consulta SQL para obtener los nodos registrados Fuente: Propia....67
Figura 32 Consultar la información de un nodo específico Fuente: Propia....68
Figura 33 Función para guardar la información consultada Fuente: Propia....68
Figura 34 Dataset Fuente: Propia....68
Figura 35 Definición de variables independientes y dependientes Fuente: Propia....69
Figura 36 División datos de entrenamiento y de prueba Fuente: Propia....69
Figura 37 Instanciación del modelo de regresión lineal Fuente: Propia....69
Figura 38 Entrenamiento del modelo de regresión lineal Fuente: Propia....70
Figura 39 Interfaz propuesta crear ECA Fuente: Propia....73

LISTA DE TABLAS

Tabla 1 Componentes de un nodo WSN....9
Tabla 2 Topologías más comunes WSN....11
Tabla 3 Estudios WSN en la agricultura....21
Tabla 4 Estudios inteligencia computacional....23
Tabla 5 Objetos inteligentes para el riego inteligente....24
Tabla 6 Características generales riego inteligente....25
Tabla 7 Dispositivos WSN....37

LISTA DE ECUACIONES

Ecuación 1....12
Ecuación 2....12
Ecuación 3....12
Ecuación 4....60
Ecuación 5....60
Ecuación 6....61
Ecuación 7....61

Capítulo 1 Introducción

Aproximadamente hace 10.000 años el hombre descubrió la agricultura, la cual se convirtió rápidamente en parte esencial de la humanidad, pero el acelerado crecimiento de la población, aumenta la demanda de productos agrícolas, en consecuencia los cultivos necesitan cada vez más agua, consumiendo cerca del 70% del agua dulce que se extrae del mundo, la cual casi el 95% se utiliza para el riego [1], en consecuencia es importante realizar investigaciones para hacer uso eficiente de los recursos en este proceso [2].

El consumo mundial de la semilla de cafeto está en ascenso, alcanzando la cifra de 9.537 millones de kilos en el año 2018 [3], posicionando a la producción de café como una de las principales actividades económicas de múltiples países, dentro de los que se destacan Brasil, Vietnam, Colombia e Indonesia. Colombia ocupa el tercer lugar de los mayores exportadores de café a nivel internacional aportando el 8.9%, además los productos agrícolas contribuyen en un 14% al total de exportaciones del país, donde el café y sus derivados representan el 6.9%, siendo así el producto principal del sector agrícola [4]. Normalmente se consumen aproximadamente 140 litros de agua para producir una taza de café de 125 ml [5], por lo cual mejorar la producción en los cultivos de café haciendo uso eficiente del agua impactará positivamente tanto económica como ambientalmente.

Para gestionar adecuadamente los recursos que se consume en los cultivos, se han realizado investigaciones incorporando el concepto de Agricultura de Precisión (AP) [6]–[9], en la cual utilizando tecnologías de redes de sensores inalámbricos (WSN, por sus siglas en inglés, Wireless Sensor Network) [10], se miden variables agroclimáticas de los cultivos para establecer los momentos adecuados para intervenir en ellos.

A nivel mundial países como Brasil, Italia, España o Estados Unidos, han optado por usar tecnologías WSN para acompañar estrategias de riego en sus diferentes cultivos [7][11], en la actualidad más de 330 millones de hectáreas cuentan con instalaciones de riego, lo que representa el 20% del total de la superficie cultivada, aportando el 40% de la producción total de alimentos en todo el mundo [12].

Para el caso de Colombia, en la exploración bibliográfica inicial, no se encontraron estudios que permitan evidenciar el uso extendido de las WSN para la planificación y gestión de los cultivos, los estudios encontrados son aislados, con resultados focalizados en invernaderos y cultivos específicos [13], [14]. Entre los estudios realizados en Colombia se pueden destacar proyectos como la monitorización de variables agroclimáticas mediante WSN en cultivos de

flores durante su crecimiento [15], el diseño e implementación de WSN basado en tecnologías IoT en cultivos de tomate [13], estos proyectos evidencian un factor en común, los cultivos que son objeto de estudio se encuentran en un ambiente cerrado de tipo invernadero, donde se facilita la instalación de sensores y actuadores, permitiendo que la distancia entre nodos sea corta logrando así que su línea de vista esté libre de obstáculos que pueden afectar la transferencia de datos [14]. En contraposición, el despliegue de redes de dispositivos en cultivos a la intemperie incrementa los retos debido a condiciones ambientales adversas y largas distancias entre nodos, esto se pudo evidenciar en proyectos como el sistema de monitoreo de variables agroclimáticas basadas en tecnologías WSN en cultivos de yuca [16], desarrollado en el departamento del Atlántico o el proyecto de un prototipo WSN para el monitoreo de la pudrición de la palma aceitera africana [17].

1.1 Motivación trabajo de grado

Entre los resultados de los diferentes proyectos que aplican tecnologías IoT para la AP, se han evidenciado complicaciones en la transferencia de información, debido a retrasos o pérdida de datos [18] por problemas de conectividad o de hardware en los dispositivos, trayendo consigo toma de decisiones erradas que pueden ocasionar consumo de excesivo de recursos. Con el propósito de optimizar la transferencia de datos y apoyar la toma de decisiones de los dispositivos dentro de una WSN en un ambiente IoT, se están desarrollando soluciones que permitan gestionar de forma inteligente la comunicación de los nodos, introduciendo el concepto de Objeto Inteligente (OI) [19], el cual busca proporcionar un comportamiento colaborativo de los dispositivos para resolver los problemas de conectividad o daño de algún nodo de la red, sin embargo, se requiere mayor capacidad de almacenamiento, procesamiento y comunicación para brindar servicios de mayor calidad a los usuarios pero esto genera un aumento de los costos.

El Grupo de Tecnologías de la Información -GTI de la Universidad del Cauca, ha realizado estudios con el objetivo de proporcionar a los Objetos Inteligentes la capacidad para interactuar entre ellos de manera autónoma, mediante el uso de arquitecturas y mecanismos específicos [20]. Dado que la mayoría de las investigaciones de AP ya han reportado buenos resultados en el riego inteligente, se puede ver una oportunidad de optimizar aún más los resultados de este, mediante la inclusión de redes de objetos inteligentes inalámbricos que puedan ejecutar

adicionalmente procesos de prevención de pérdida de datos o manejo de la integridad de estos recogidos en dichas redes.

De lo anterior, se puede inferir la siguiente pregunta de investigación: ***¿Cómo prevenir o mitigar la pérdida de información en una WSN con el fin de apoyar la precisión en el riego inteligente de los cultivos de café?***

Según la pregunta de investigación, el enfoque de solución para el presente proyecto consiste en utilizar el concepto y arquitectura de objetos inteligentes que maneja el grupo GTI de la Universidad del Cauca, con el fin de proponer un mecanismo de creación de una red de objetos inteligentes inalámbricos, para mejorar el proceso de captura y transmisión de información y así poder prevenir o mitigar la pérdida de esta. Dado que en Colombia y el departamento del Cauca el cultivo de café es base para su economía, la presente investigación tomará como estudio de caso el riego de precisión en cultivos de café.

1.2 Objetivos

1.2.1 Objetivo general

Proponer un modelo de una red de objetos inteligentes inalámbricos, con la finalidad de emplearlo en el riego de precisión de cultivos de café en el departamento del Cauca, mediante la gestión de las eventualidades de pérdida de información que se presenten en la red.

1.2.2 Objetivos específicos

1. Definir un modelo de red de objetos inteligentes inalámbricos, mediante la incorporación de los conceptos de WSN a la arquitectura de Interacción Semántica de Objetos Inteligentes, propuesta por Niño-Zambrano [21].
2. Identificar los posibles algoritmos de inteligencia computacional que se pueden utilizar en el modelo de red de objetos inteligentes inalámbricos definido, con el fin de prevenir o evitar la pérdida de datos, debido a problemas de conectividad o del hardware instalado.
3. Realizar una prueba de concepto en un ambiente simulado, que permita implementar una red de objetos inteligentes inalámbricos bajo el modelo propuesto, en el contexto del riego de precisión en los cultivos de café, mediante el monitoreo de la radiación

solar, humedad y temperatura del suelo, con el fin de evaluar la eficiencia del uso de agua y la tolerancia de la red a la pérdida de información.

1.3 Estructura trabajo de grado

Este trabajo de grado presenta una estructura, la cual fue dividida en capítulos.

- **CAPÍTULO 1. INTRODUCCIÓN**

  En el primer capítulo se describe el problema, como también se contextualiza mediante la motivación del proyecto y también se presenta el objetivo general y objetivos específicos del proyecto.
- **CAPÍTULO 2. MARCO TEÓRICO**

  En el segundo capítulo se busca presentar los conceptos más relevantes y necesarias para el proyecto a desarrollar.
- **CAPÍTULO 3. MODELO DE RED DE OBJETOS INTELIGENTES INALÁMBRICOS**

  En este capítulo se presenta el modelo de red de sensores inteligentes inalámbricos.
- **CAPÍTULO 4. DESARROLLO PRUEBA DE CONCEPTO EN EL CONTEXTO DE RIEGO DE PRECISIÓN DE CULTIVOS DE CAFÉ**

  En este capítulo se presenta las diferentes pruebas que nos permiten validar la funcionalidad del modelo propuesto mediante una metodología basada en iteraciones.
- **CAPÍTULO 5. CONCLUSIONES Y TRABAJO FUTURO**

  Finalmente se muestran las conclusiones que surgieron como resultado de la investigación y se expone el trabajo que hace falta por desarrollar.

Adicionalmente a los capítulos anteriormente descritos, se presenta una colección de Anexos que muestran los procedimientos de implementación técnica y despliegue de la red de objetos inteligentes inalámbricos:

- **Anexo A**

  Se presentan las fichas bibliográficas correspondientes a los estudios obtenidos como resultado de la ejecución del mapeo sistemático.
- **Anexo B**

  Describe el paso a paso de las configuraciones realizadas a los distintos dispositivos del modelo para poder implementar las pruebas realizadas en la prueba de concepto.

- **Anexo C**

  Contiene los códigos fuente utilizados en el desarrollo de la prueba de concepto.

- **Anexo D**

  Contiene el artículo del mapeo sistemático realizado en esta monografía.

Capítulo 2 Marco teórico

En este capítulo se presentan los principales conceptos teóricos sobre los que se desarrolla el presente trabajo, estos conceptos hacen referencia a los núcleos temáticos como IoT, WSN, Agricultura de Precisión, IA y Objeto Inteligente, además, se hace un estado del arte aplicando elementos de la estrategia de mapeo sistemático [22], obteniendo los trabajos de investigación relacionados con el presente.

2.1 Variables agroclimáticas en los cultivos de café

La formación de los botones florales dependen de la duración de la luz solar [23], donde se puede encontrar plantas que logran una mayor floración en días cortos, otras, en días largos u otras son indiferentes a la duración del día, para la floración de los cultivos de café se requieren de días cortos con una duración de luz solar no mayor de 13,5 horas, teniendo a Colombia como ventaja su posición geográfica de las zonas cafeteras dado que presenta durante todo el año días con no más de 13 horas de duración.

La temperatura óptima de aire para los cultivos de café se encuentra entre los 18 a 22°C [23], cuando la temperatura supera los 23°C y es durante temporada seca en un periodo de floración se produce aborto floral y alteraciones en las flores teniendo como consecuencia una disminución considerablemente alta de la producción. En temperaturas menores a 18°C se generan crecimientos exuberantes de las plantas, generando disminución de la floración y por consiguiente baja producción.

El consumo del agua mensual de una planta de café se aproxima a los 125mm [23]. En Colombia se tiene una evaporación en regiones cafeteras entre 3 a 4mm diarios, lo que si ocurre durante un periodo de 30 a 40 días seguidos esto afectaría la producción del grano. Se ha detectado que las zonas cafeteras con Diferencia Hídrica Anual (DHA) inferiores a 150mm, son aptas para el cultivo de café, donde también se puede ver que las zonas con un DHA entre 150 y 200mm son consideradas zonas marginales para el cultivo de café y esta requieren de un riego suplementario para lograr obtener una buena producción. Se encuentran zonas no aptas para el cultivo de café ya que estas superan los 200mm de DHA, las cuales requieren de un riego periódico obligatorio.

El desarrollo del fruto del café pasa por cuatro etapas [23], iniciando en su floración en su primera etapa, seguida del crecimiento de la almendra hasta obtener su tamaño final, luego la etapa de llenado donde obtienen su consistencia sólida y por última la etapa de maduración del

fruto. El aporte hídrico en estas etapas es fundamental, pero en algunas etapas es crítico para su desarrollo, donde al tener un déficit hídrico entre las etapas dos y tres este presenta deshidratación y una posterior caída de los frutos tiernos. Si la deficiencia hídrica ocurre entre la etapa 2 y 3 se presenta el llenado parcial de los frutos o el defecto llamado grano negro.

2.2 Ecosistema IoT

En un ecosistema IoT se pueden encontrar una gran variedad de dispositivos interconectados encargados de recopilar e intercambiar información [24]-[25], teniendo en cuenta que la ubicación y capacidades de los dispositivos, permiten su clasificación en tres niveles de computación (Figura 1):

- **Computación en la nube (CLOUD):** Este nos permite liberación de almacenamiento y procesamiento de información a nivel local, también ofrece una gran variedad de servicios, aplicaciones y plataformas con múltiples tecnologías que posibilitan la computación desde internet.
- **Computación en la niebla (FOG):** Computación en la niebla se puede ver como un complemento de cloud computing, ya que proporciona servicios similares con la diferencia que los nodos se encuentran un nivel más cercano a los usuarios y o de la fuente de datos, este presenta un procesamiento centralizado, mejores tiempos de respuesta, bajo consumo de ancho de banda y alta capacidad de cómputo.
- **Computación en el borde (EDGE):** Computación en el borde surge como solución de los problemas de tiempos de respuesta que son limitaciones de la computación en la nube, ya que permite el procesamiento y adquisición de los datos en dispositivos que se encuentran al bode de la red, sin la necesidad de llevarlos a la nube.

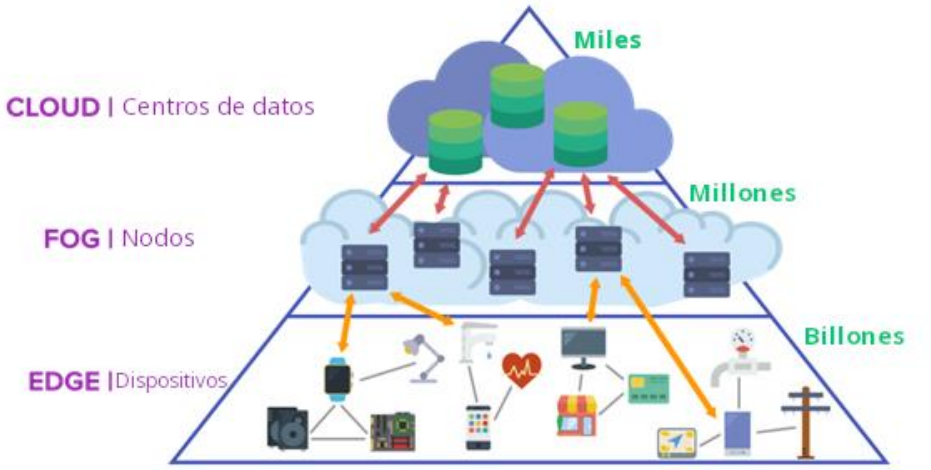


Figura 1 Niveles de procesamiento IoT
Fuente: https://onx.la/488c8

2.3 Redes de sensores inalámbricos (WSN, Wireless Sensor Network)

Según la *International Telecommunication Union (ITU-T)* [26], una red de sensores inalámbricos (WSN) como su nombre lo indica, es una red compuesta por un conjunto de nodos interconectados a través de una interfaz inalámbrica, dichos nodos poseen distintos tipos de sensores con capacidad para medir diversidad de fenómenos del mundo real, por ejemplo, humedad del suelo, temperatura ambiente, radiación solar, etc., en consecuencia las WSN son ampliamente utilizadas en múltiples campos como la salud, agricultura, seguridad, medio ambiente, entre otros. Estas redes de sensores están densamente desplegadas, ya sea dentro del ambiente donde ocurre un fenómeno o muy cerca de él, y se basan en el esfuerzo colaborativo de todos sus nodos para la obtención de los datos.

El componente clave de una WSN es el nodo o también llamado mota, el cual se compone de los elementos que se pueden ver en la Figura 2 y los cuales son descritos en la Tabla 1

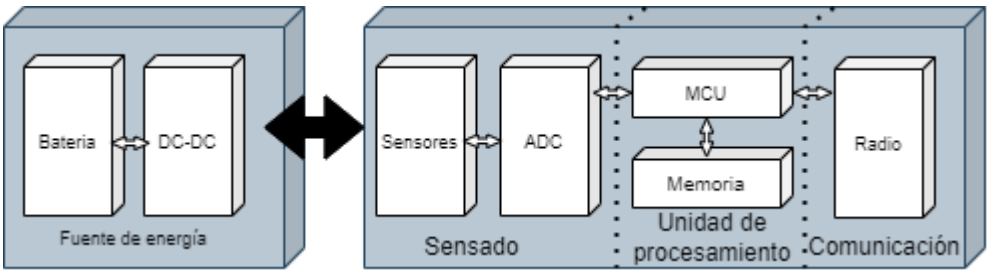


Figura 2 Arquitectura Nodo WSN
Fuente: Propia

Componente		Definición
Unidad de procesamiento	Microcontrolador (MCU)	Circuito integrado con capacidades de procesamiento para implementar un protocolo de comunicación en la WSN
	Memoria	Chip adherido a la tarjeta principal o también removible que permite guardar la información capturada o también puede ser de tipo temporal
Comunicación	Radio	Dispositivo electrónico (Transmisor/Receptor) que permite acceder a un canal de transmisión para enviar y recibir datos para comunicarse con otros dispositivos dentro de la red
Sensado	Sensor	Elemento hardware con la finalidad de capturar señales del medio donde se ubique
	ADC	Conversor de señales analógicas a digitales capturadas por los sensores.
Fuente de energía	Batería	Es la batería que alimenta los circuitos internos del nodo permitiendo el funcionamiento cumplir el objetivo que se le asigne
	DC-DC	Conversor de corriente continua que trasforma de una tensión a otra y tiene como propósito activar el nodo.

Tabla 1 Componentes de un nodo WSN

Una WSN posee características clave como: tiempo de vida, cobertura de red, costo y facilidad de instalación, capacidad de comunicación, consumo de energía [27].

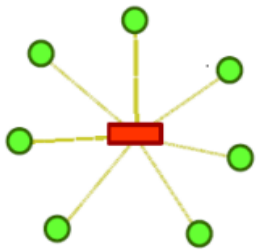
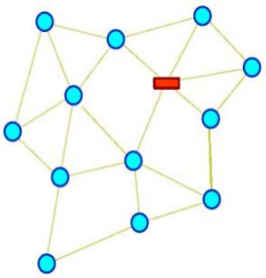
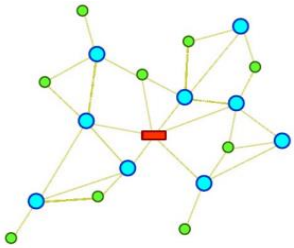
2.3.1 Elementos de las WSN

Para este proyecto las WSN permiten establecer los elementos de la arquitectura que permiten crear o proponer un modelo.

- Sistema de adquisición de datos: Es el medio físico por el cual se transmite la información que acorde al tipo de red que se está trabajando corresponde al espacio físico abierto, es decir se trata de un medio no guiado.
- Motas: Son la parte fundamental de las WSN, son las encargadas de recopilar información del mundo real mediante sensores especializados para a partir de aquí iniciar el monitoreo de lo que se desee
- Gateway (Puerta de enlace): Dispositivo que permite recibir los datos de los nodos sensores y redirigirlo a otra instancia, ya sea para guardarlo o para generar una alerta
- Estación base: Equipo que guarda y administra los datos recopilados por la WSN

2.3.2 Topología en WSN

La topología de una red es la arquitectura que determina la disposición de los enlaces que conectan los nodos, además de la configuración de hardware, la cual define la forma en que se transmite la información a través de los dispositivos. Hay varios tipos de topologías que pueden ser usadas por una aplicación WSN, dentro de las más usadas se encuentra la topología en estrella, malla, e hibrida [28]. Las ventajas y desventajas de cada una de las topologías están presentada en la Tabla 2.

Topología	**Estrella**	**Malla**	**Híbrida**
Diseño			
Ventajas	Gasto de energía óptimo. Si un nodo final falla, este no afecta el resto de la red.	Tolerancia a fallos. Fácilmente escalable. Cubre grandes extensiones.	Bajo consumo de energía. Cubre grandes extensiones. Tolerante a fallos.

	Fácil de agregar o eliminar nodos.		
Desventajas	Limitación de transmisión vía radio entre cada nodo por la distancia. No cuenta con una vía alternativa de comunicación. Se pueden producir cuellos de botella. Si falla el nodo central, falla toda la red.	Largos periodos de tiempo en la transferencia de datos.	En caso de pérdida de nodos intermedios, muchos nodos finales quedan inhabilitados.

Tabla 2 Topologías más comunes WSN

2.3.3 Tecnologías inalámbricas en WSN

Dentro de WSN existen tres tecnologías de comunicación inalámbrica, las cuales son: Bluetooth, ZigBee y GPRS [29], además de estos mecanismos de comunicación se incluye LoRaWAN a raíz de que cumple una de las principales características de las redes de sensores inalámbricos que es el bajo consumo de energía [30]. Otro protocolo incluido es Wifi (IEEE 802.11) debido a que posee amplia documentación al ser altamente utilizado, en contra parte tiene un consumo elevado de energía. Otra motivación importante es que la investigación es flexible en cuanto a la tecnología de comunicación, porque independientemente de cuál se use, lo que se quiere es mejorar la disponibilidad, y la pérdida de información teniendo en cuenta principalmente las topologías. A pesar de que esta investigación está orientada al uso de WSN se incluyeron las tecnologías LoRa y Wifi que a pesar de no estar dentro de los mecanismos de comunicación que usan las redes inalámbricas según [31], permiten generar una aproximación al modelo de red que se necesita para WSN.

2.4 Métricas

Las métricas permiten realizar un análisis característico de datos trasmitidos en la red pudiendo ser utilizadas para medir el rendimiento de la red en general y determinar el estado de cada uno de sus nodos [32]:

- **Packet Delivery Ratio (PDR):** Relación entre la cantidad de paquetes que se envían y los que se reciben.

$$\boldsymbol{PDR = \frac{\sum paquetes\ recibidos}{\sum paquetes\ enviados}} \qquad \textbf{Ecuación 1}$$

- **Throughput:** Determina el rendimiento de la red a partir de los bits trasmitidos en determinado tiempo.

$$Rendimiento = \frac{Numero\ de\ bits}{tiempo\ de\ ejecucion} \qquad \textbf{Ecuación 2}$$

- **End-to-End Delay:** se determina el tiempo que demora un paquete en llegar a su destino a partir de la ruta encontrada y la cola de paquetes en la trasmisión.

$$EED = \frac{\sum(\text{Tiempo de llegada del paquete} - \text{Tiempo de envío})}{\sum \text{Número de conexiones}} \qquad \textbf{Ecuación 3}$$

2.5 Modelo

Un modelo es una representación de una abstracción para cumplir un objetivo, el cual nos permite poner a prueba una hipótesis o algún fenómeno. También permite mostrar las características generales de la estructura de dicho fenómeno o hipótesis, explicar sus elementos, mecanismos y procesos, cómo se interrelacionan y los aspectos teóricos que le dan sustento, para facilitar su comprensión.

Dependiendo del objetivo que se persigue el modelo puede variar, logrando encontrar desde el modelo más básico como puede ser la representación gráfica (maqueta) hasta modelos tan complejos que solo pueden ser representados desde ordenadores con grandes capacidades.

"Los requisitos primordiales para construir cualquier modelo son:

- *Un propósito claramente definido.*
- *Identificar las consideraciones esenciales (incluir en el modelo).*
- *Desechar consideraciones superfluas (estas son fuente de confusión).*
- *El modelo debe representar la realidad en forma simplificada."* [33]

2.6 Protocolo MQTT

El protocolo MQTT mediante un modelo publicador/suscriptor permite la comunicación para enviar mensajes entre dispositivos facilitando su integración [34], minimizando el ancho de banda y consumo de energía. MQTT está compuesto de clientes, servidores o intermediadores, editores, subscriptores y temas (topic) donde los clientes son todos aquellos que editan o se suscriben a la información, mientras que un servidor o intermediario es aquel que controla la entrega de los mensajes recibidos al cliente o clientes determinados que lo solicita, los publicadores son clientes que envían mensajes a través del intermediario a otros clientes suscritos, mientras los suscriptores son aquellos que reciben mensajes a través de un intermediario de acuerdo a un tema solicitado siendo el tema el identificador del mensaje.
Dentro de MQTT se encuentran tres niveles de transferencia basados en la confiabilidad:
Nivel 1 QoS 0: El mensaje es enviado una sola vez lo que indica que si hay un fallo puede no ser entregado.
N**ivel 2 QoS 1:** El mensaje es enviado hasta que se garantice su entrega en caso de fallo puede que se entregue un mensaje duplicado.
N**ivel 3 QoS 2:** Se garantiza que el mensaje llegue al suscriptor y una sola vez.

2.7 Agricultura de precisión (AP)

La Sociedad Internacional de Agricultura de Precisión (ISPA, por sus siglas en inglés, https://www.ispag.org/) define la agricultura de precisión como: *"una estrategia de gestión que recopila, procesa y analiza datos temporales, espaciales e individuales y la combina con otra información para respaldar las decisiones de gestión de acuerdo con la variabilidad estimada para mejorar la eficiencia, productividad, calidad, rentabilidad y sostenibilidad de la producción agrícola en el uso de recursos."* [35].

La AP es una de las aplicaciones del uso de WSN en los cultivos, esta genera una mejor planificación del proceso de cultivo, redundando en la obtención de productos de mejor calidad [36]–[38]. Dentro de los estudios realizados en el campo de la AP se ha hecho especial énfasis

en el mejoramiento del proceso de riego, ya que este proceso beneficia dos aspectos fundamentales para el agricultor, como lo son, el rendimiento del cultivo y el ahorro en el uso del agua, este último beneficiando al medio ambiente. En los estudios encontrados se mencionan dos métodos de riego, riego programado (RP) y riego automático (RA), el primero como su nombre lo indica se programa para ser realizado en momentos específicos del día durante un tiempo determinado mediante activación remota, el segundo método de riego está basado en el uso de una WSN para obtener mediante sensores especializados el nivel de humedad de cada una de las plantas, luego esta información es enviada a un servidor remoto que realiza un procesamiento que determina si se activa la operación de riego dependiendo si el nivel de humedad está en el límite que se ha determinado. Al comparar los dos métodos de riego, se determinó que el RA genera un 60% de eficiencia en el uso del agua sobre el RP [39], esto debido a que el RA proporciona la cantidad de agua exacta que necesita cada planta.

2.8 Objeto inteligente

El grupo GTI de la Universidad del Cauca define a un Objeto Inteligente (OI) [21] o Smart Object (SO) como una entidad física que posee una representación digital dentro de la red, este OI posee ciertas capacidades para denominarse inteligente. Capacidades como ser consciente de su entorno y de los objetos que lo rodean, poder interactuar e intercambiar información de forma cooperativa con otros objetos inteligentes y alto poder de procesamiento para monitorear los eventos del mundo real.

El diseño de los objetos inteligentes se puede definir en tres dimensiones [40]:

- Conciencia: La conciencia es la capacidad de un OI para comprender (es decir, sentir, interpretar y reaccionar) eventos y actividades humanas que ocurren en el mundo físico.
- Representación: Se refiere al modelo de programación y aplicación de un OI, en particular, abstracciones de programación.
- Interacción: Denota la capacidad del objeto para interactuar con el usuario en términos de entrada, salida, control y retroalimentación.

Características del modelo de red de objetos inteligentes inalámbricos

El modelo propuesto debe tener características como lo son:

- **Escalabilidad:** la arquitectura de una red de sensores inalámbricos debe poder agregar nodos sin que se vea afectada la funcionalidad y rendimiento de esta

- **Eficiencia energética:** el tiempo de vida de los nodos de una WSN se ven altamente impactados por la vida útil de la fuente de energía, de la cual se alimentan, ya que estas fuentes son limitadas en la mayoría de los casos siendo imposible la recarga de estas dependiendo de los escenarios de las aplicaciones, teniendo como consecuencia daños de nodos que pueden causar modificación de la topología y por consiguiente retardos de paquetes, de aquí la gran importancia de la administración y conservación de las fuentes de energía de los nodos. [28]
- **Tolerancia a fallos:** dentro de la red de sensores se pueden presentar fallos de hardware o bloqueo de los sensores por falta de energía, daños físicos o puede darse el caso de interferencias en el lugar de despliegue de la red, lo anteriormente mencionado no debe afectar el funcionamiento de la red para así cumplir con el principio de tolerancia a fallos.

2.9 Aprendizaje automático (Machine learning)

El aprendizaje automático [41] (ML), se puede catalogar como una rama de la inteligencia artificial que se centra en el desarrollo de sistemas capaces de aprender o mejorar el rendimiento a partir de un conjunto de datos. Su rendimiento mejora y se ve reflejado a partir de que sus muestras de entrenamiento aumentan. El ML requiere una intervención mínima humana siendo una herramienta potente que facilita la toma de decisiones donde utiliza datos de diferentes fuentes como lo pueden ser redes de sensores u otras fuentes interconectadas, dado que el ML está siendo desarrollado en distintos escenarios o ambientes este se puede ver afectado o presentar una limitación en cuanto a la precisión de sus predicciones debido a la calidad de los datos, representación del modelo y dependencia entre los datos de entrada y salida. Dentro de ML se pueden encontrar dos categorías de aprendizaje: aprendizaje automático no supervisado y aprendizaje automático supervisado siendo este último el más utilizado. Los algoritmos de aprendizaje supervisado utilizan un conjunto de datos previamente conocido y etiquetado para realizar el entrenamiento de un modelo propuesto, el cual permita la predicción de una variable objetivo fuera de la muestra. Por otra parte, los algoritmos de aprendizaje no supervisado descubren patrones sobre un conjunto de datos sin referencia, siendo útil en aplicaciones exploratorias donde no hay un objetivo específico establecido.

Un tipo de algoritmo de aprendizaje no supervisado aplicados a la AP son las redes neuronales artificiales (ANN), que en [42] son utilizadas para evaluar la idoneidad de la tierra agrícola basado en la información recolectada por distintos sensores desplegados en un cultivo, este

procesamiento requiere de poderosas herramientas que puedan trabajar sobre una gran cantidad de información sin identificar, por lo cual se requiere utilizar servicios en la nube, por otra parte los algoritmos de aprendizaje supervisado requieren de menos poder de procesamiento al tener como insumo principal un conjunto de datos de entrada claramente identificados para entrenar dichos algoritmos, esto conlleva que se pueda clasificar o pronosticar los datos de salida con precisión, además al tener conjuntos de datos de entrada y de salida etiquetados se puede medir la precisión y mejorarla con el paso del tiempo.

El aprendizaje supervisado se divide en dos tipos de algoritmos: clasificación y regresión.

- Clasificación: Asigna con precisión datos de prueba en categorías específicas. En el mundo real, se puede usar este algoritmo de aprendizaje supervisado para clasificar el spam en una carpeta separada de la bandeja de entrada de un correo electrónico. Los clasificadores lineales, las máquinas de vectores de soporte, los árboles de decisión y los bosques aleatorios son tipos comunes de algoritmos de clasificación.
- Regresión: muestra la relación entre las variables dependientes e independientes. Los modelos de regresión son útiles para predecir valores numéricos basados en diferentes puntos de datos, como las proyecciones de ingresos por ventas para un negocio determinado. Algunos algoritmos de regresión populares son la regresión lineal, la regresión logística y la regresión polinomial.

2.10 Trabajos relacionados

Como resultado de la investigación realizada se lograron encontrar acercamientos a ejemplos relacionados con este proyecto que implementan tecnologías, diseños y arquitecturas para los modelos de red inalámbrica que involucren WSN.

La arquitectura del modelo de interacción semántica desarrollada por Niño-Zambrano [21] está compuesta por cinco (5) capas, como se puede ver en la siguiente Figura 3

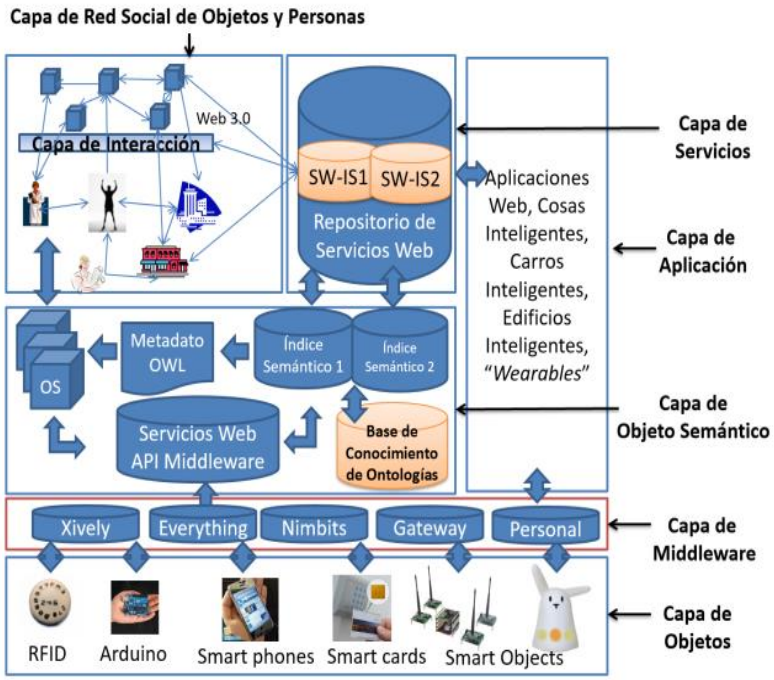


Figura 3 Arquitectura de interacción semántica
Fuente:[21]

Capa de Interacción

La capa más compleja de la arquitectura es la capa de red social de objetos y personas, la cual permite la interacción entre distintos objetos, además de los usuarios a través de aplicaciones, permitiendo intercambiar información acerca del contexto para ajustar los servicios de la IoT teniendo en cuenta las distintas características.

Capa de Middleware

La capa de middleware está compuesta por servidores IoT, los cuales solucionan el problema de la heterogeneidad presente en el hardware actual, donde el middleware captura la información un dispositivo hardware específico y lo pone en un formato estandarizado como JSON para que pueda ser reutilizado por cualquier otro servicio u objeto de la arquitectura mediante el protocolo API REST, solventando las dificultades de conexión que existen con la Capa de Objeto.

Capa de Objeto Semántico

En esta capa se crean representaciones semánticas de los objetos a partir de los metadatos dispuestos en los servidores IoT, las cuales se basan en una ontología llamada ontología de

objeto semántico. También se construye el índice semántico de los objetos para acceder a través de la capa de servicios.

Capa de Servicios

En esta capa se encuentran disponibles los servicios para que las capas: capa de red social de objetos y personas y la capa de aplicaciones puedan consumir los metadatos, datos e información de una forma interoperable y transparente, esta capa también provee los servicios e información necesaria para soportar los distintos escenarios de interacción.

Capa de Objeto

En esta capa se encuentran los dispositivos físicos, los cuales pueden ser conectados a internet, estos dispositivos pueden ser teléfonos móviles, tarjetas y objetos que se conectan con diferentes sensores y/o actuadores, los cuales pueden ser denominados objetos inteligentes.

Li, et al. [43], presenta una revisión de la aplicación de Inteligencia Artificial (IA) en la agricultura para diferentes tareas tales como: análisis de las condiciones climáticas, la temperatura, el uso del agua, el uso de la energía y las condiciones del suelo donde se puede observar que la IA se está enfocando en robótica, monitoreo de suelos y de cultivos y análisis predictivo siendo este ultimo de interés para el desarrollo de este proyecto ya que aportaría en el objetivo de mitigar la perdida de datos de la red.

Codeluppi, et al. [44], propone una plataforma IoT orientada a WSN y aprendizaje automático (ML) llamada LoRaFarM, donde un Gateway inteligente actúa como un recolector de datos sobre una red de sensores instalados en un invernadero y pueda enriquecer su capacidad con un modelo de predicción basado en redes neuronales (NN) que permiten el pronóstico de variables como temperatura del aire en el interior de un invernadero por si se presenta algún faltante de datos del sensor de la WSN. El modelo de predicción es alimentado por datos almacenados en el repositorio DarkSky, este modelo está diseñado para ejecutarse en dispositivos con pocas capacidades. Se observó que esta plataforma está diseñada para trabajar en ambientes internos como invernaderos, mostrando dificultades al enfrentarse a condiciones meteorológicas adversas como la lluvia.

El proyecto desarrollado Trilles, et al. [45], plantea una arquitectura orientada hacia la AP, la cual permite gestionar de manera óptima un cultivo mediante la recolección de información de

distintas variables agroclimáticas tales como: humedad del suelo, radiación solar, temperatura ambiente, entre otros, esta información es proporcionada a distintos modelos, los cuales pueden determinar los momentos y cantidades para la aplicación de agua y fertilizantes, además pueden detectar de enfermedades en los cultivos.

2.11 Estado del arte

Para obtener el estado del arte se realizó un mapeo sistemático basado en el procedimiento de Petersen [22], el cual contiene tres etapas: planificación, ejecución y documentación, las cuales son descritas en el Anexo A. Este mapeo se realizó con el fin de responder las preguntas de investigación descritas en la fase de planificación, además de conocer y contextualizarse con los temas de investigación.

2.11.1 Descripción general de los estudios seleccionados

El interés académico acerca de la AP es reciente, como se puede ver en la Figura 1. En 2015 fueron publicados 2 artículos, y desde ese momento, la cantidad de publicaciones fue aumentando hasta llegar a las 39 publicaciones cinco años después en 2020, cabe aclarar que estos datos se dan a partir del enfoque que se le da al mapeo sistemático.

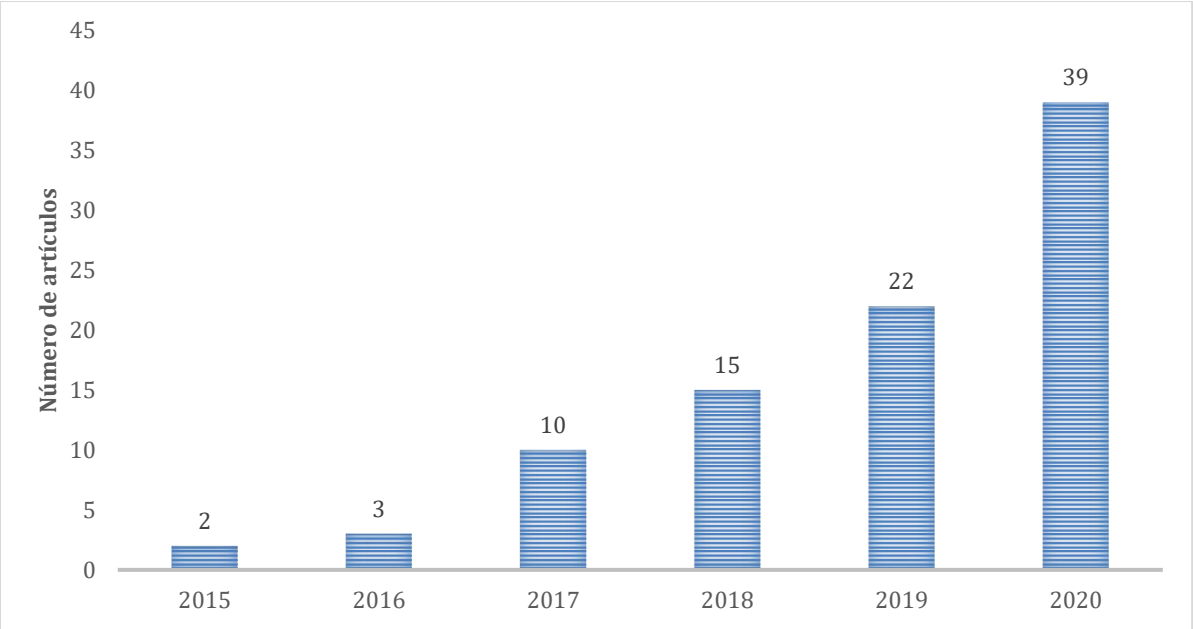


Figura 4 Histograma artículos por año
Fuente: Propia

La Figura 5 muestra la distribución de los artículos entre los lugares de publicación (i.e., revistas, conferencias y libros). La mayoría de los artículos fueron publicados en conferencias (40%), de los cuales 22 (21%) fueron publicados en conferencias indexadas en SCImago. Se pudo encontrar que alrededor del 19% de las publicaciones se encuentran ubicados en los cuartiles Q3 y Q4, además de 9% que no tienen calificación. Es importante decir que aproximadamente el 29% de los artículos fueron publicados en revistas de alto nivel (Q1 – Q2).

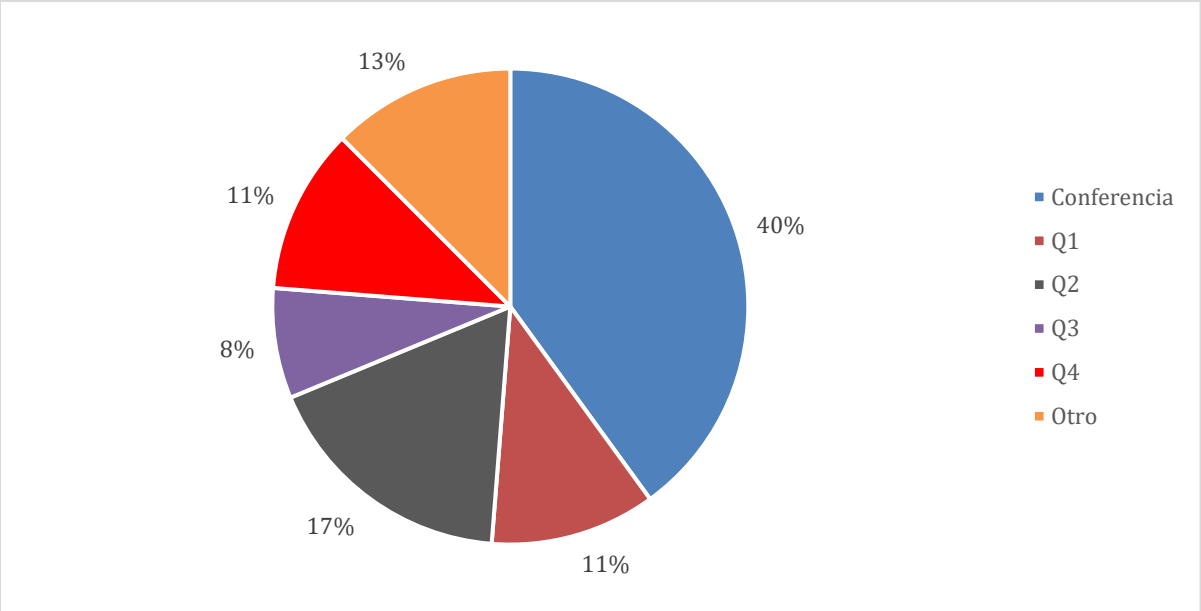


Figura 5 Calificación de artículos
Fuente: Propia

Los resultados que se obtuvieron son presentados a continuación, estos pertenecen a la fase de documentación definidos en la metodología para dar respuesta a cada una de las preguntas de investigación definidas.

2.11.2 WSN en agricultura de precisión

Las investigaciones encontradas (ver Tabla 3) evidencian acercamientos a la implementación de una red de dispositivos con capacidades avanzadas dentro de la AP con el fin optimizar los recursos disponibles sin afectar negativamente el medio ambiente, esto a partir de un procesamiento simple sobre la información recolectada y trabajando en conjunto con servidores en la nube, por lo cual no logran competir con el poder de procesamiento de un OI como ha definido Niño-Zambrano en [21], el cual puede realizar un procesamiento más complejo en el mismo OI ya sea para simular sensores, tener conciencia de los demás dispositivos a su

alrededor o para realizar pronósticos a partir de distintas simulaciones, siendo todo esto necesario para realizar procedimientos que optimicen los procesos agrícolas.

Investigación	Año	Referencia
AI at the Edge: A Smart Gateway for Greenhouse Air Temperature Forecasting	2020	[44]
An architecture model for smart farming	2019	[29]
LoraFarM: A LoRaWAN-based smart farming modular IoT architecture	2020	[46]
Precision agriculture techniques and practices: From considerations to applications	2019	[47]
Smart board for precision farming using wireless sensor network	2019	[48]
Smart Soil Parameters Estimation System Using an Autonomous Wireless Sensor Network with Dynamic Power Management Strategy	2018	[49]
Wireless sensor and actuator system for smart irrigation on the cloud	2015	[50]

Tabla 3 Estudios WSN en la agricultura

Siete (7) estudios presentados en la Tabla 3, se logran acercar a la respuesta de esta pregunta y se resalta que cinco (5) de ellos fueron publicados en los últimos dos años, demostrando una tendencia hacia cada vez dotar de más inteligencia a dispositivos desplegados dentro de los cultivos.

2.11.3 Mitigar o controlar la pérdida de datos en una WSN de objetos inteligentes

Con el estudio realizado en el mapeo sistemático se logra identifica acercamientos que pueden aportar en la identificación de posibles algoritmos de inteligencia computacional (IC), los cuales permitan mitigar o controlar la perdida de datos de la WSN, los estudios encontrados se ven reflejados en la Tabla 4 que se presenta a continuación:

Investigación	Técnicas IC	Referencia
An IoT based smart farming system using machine learning	Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU)	[51]

An Intelligent and Optimal Resource Allocation Approach in Sensor Networks for Smart Agri-IoT	Back Propagation Neural Network (BPNN), Bayesian Neural Network (BNN), LEACH-CS	[52]
AI Crop Predictor and Weed Detector Using Wireless Technologies: A Smart Application for Farmers	Naive Bayes, Convolutional Neural Network (CNN), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Multiple Regression, K-means	[53]
A Hybrid Data Acquisition Model using Artificial Intelligence and IoT Messaging Protocol for Precision Farming	Deep Neural Network based routing algorithm (DNNRA), DN-RPL	[54]
A Multi-collective, IoT-enabled, Adaptive Smart Farming Architecture	Big Data, Machine Learning	[55]
A smart decision system for digital farming	Business Process Management (BPM), Business Rules Management System (BRMS)	[56]
Internet of things based smart agriculture system using predictive analytics	Kalman Filter, Autoregressive Integrate Moving Average Model (ARIMA Model)	[57]
Internet of things platform for smart farming: Experiences and lessons learnt	SSN Ontology	[58]
Review - Machine Learning Techniques in Wireless Sensor Network Based Precision Agriculture	Classification and Regression Trees (CART), Autoregressive Integrate Moving Average Model (ARIMA Model), Deep Neural Network	[41]
Sensors driven ai-based agriculture recommendation model for assessing land suitability	Multi-Layer Perceptron (MLP), Artificial Neural Network (ANN)	[42]

Machine learning Implementation in IoT based Intelligent System for Agriculture	Decision Tree**,** Naive Bayes	[59]
Deep Learning and IoT for Smart Agriculture Using WSN	Long Short-Term Memory (LSTM), Gated Recurrent Unit Network (GRUN)	[60]

Tabla 4 Estudios inteligencia computacional

Dentro del mapeo sistemático se identificó una investigación realizada por Mekonnen [41], en la cual se desarrolla una revisión bibliográfica donde se presentan distintas técnicas de inteligencia computacional utilizadas en la agricultura para realizar múltiples procesos, mostrando una tendencia hacia la utilización de dichas técnicas para obtener el máximo rendimiento de un cultivo y maximizar el uso de recursos. Dentro de las técnicas mencionadas están los algoritmos de aprendizaje automático (machine learning), algunas de ellas son CART para clasificación de cultivos de acuerdo con su rendimiento, modelo ARIMA utilizado para predecir información agroclimática basado en la información recolectada por redes de sensores y redes neuronales profundas para suplir la información de las WSN mediante información de teledetección. Se evidencia que estos estudios no cumplen con la función específica de prevenir o evitar la pérdida de datos, pero pueden ser usados para dicho fin.

2.11.4 Objetos inteligentes en AP para el riego inteligente

Son múltiples estudios los que vienen incursionando en el campo de agricultura de precisión, de los cuales se enfocan en el riego inteligente, estos se presentan en la Tabla 5:

Investigación	Año	Referencia
Smart Irrigation Control System Using Wireless Sensor Network Via Internet-of-Thing	2020	[61]
Smart Irrigation System for Precision Agriculture - The AREThOU5A IoT Platform	2020	[62]
Smart Irrigation and Intrusions Detection in Agricultural Fields Using IoT	2020	[63]
A Cloud-Based Application for Smart Irrigation Management System	2021	[64]
Smart farming using temperature sensor, moisture sensor, flow sensor and ultrasonic sensor leading to water conservation	2019	[65]

Adapting weather conditions based IoT enabled smart irrigation technique in precision agriculture mechanisms	2019	[66]
Monitoring system using web of things in precision agriculture	2017	[6]
An Intelligent Irrigation Scheduling System Using Low-Cost Wireless Sensor Network Toward Sustainable and Precision Agriculture	2020	[67]
Review of Sensor Network-Based Irrigation Systems Using IoT and Remote Sensing	2020	[43]
An interoperable IP based WSN for smart irrigation system	2017	[68]
Precision irrigation using Wireless Sensor Network	2015	[69]
Wireless sensor network and Internet of Things in precision agriculture	2018	[70]

Tabla 5 Objetos inteligentes para el riego inteligente

Posterior a la extracción de información que se realizó sobre los artículos presentados (Tabla 5), se encontraron generalidades en cuanto a las características de las investigaciones, las cuales se muestran a continuación:

	Tecnología		**Procesamiento***				
Referencia	**WSN**	**IoT**	**C**	**F**	**E**	**Comunicación**	**Tipo sensor***
[61]	X	X		X	X	Wifi	HA TA
[62]	X	X	X	X	X	LoRaWAN SigFox MQTT	TA HS
[63]	X	X	X	X	X	Serial	HS
[64]	X	X	X	X	X	Radio Frecuencia	TA HS TS
[65]	X	X	X		X	Wifi, LAN	TA HS
[66]	X	X		X	X	ZigBee Wifi	HA TS

						GSM	HS TA CO_2 LS
[6]	X	X	X			ZigBee GPRS	HS
[67]	X			X	X	Radio Frecuencia GPRS	HS HA TA LS
[43]	Es una revisión						
[68]	X	X	X		X	LoWPAN	HS
[69]	X	X	X	X	X	ZigBee GPRS GSM	HS
[70]	X	X		X	X	Wifi	HA TA HS

Tabla 6 Características generales riego inteligente

Procesamiento:

Cloud (C) Fog (F) Edge (E)

Sensores:

Humedad ambiente (HA)	Temperatura ambiente (TA)	Luz solar (LS)
Humedad del suelo (HS)	Temperatura del suelo (TS)	CO_2

A partir de lo anterior se evidenció que los estudios encontrados realizan el riego de precisión haciendo uso del concepto de WSN combinado con tecnologías IoT para ampliar las capacidades de interacción y procesamiento de información. Otra característica en común de los estudios es la utilización de dispositivos con capacidades de hardware y software avanzadas para realizar procesamientos complejos de información para dejar de depender de una conexión a la web, esto se denomina según el ecosistema IoT mencionado en el marco teórico como procesamiento en la niebla (Fog), por último, es importante mencionar que todos los estudios

necesitan de distintos tipos de sensores para realizar el riego, dentro de los más utilizados se encontraron sensores para monitorear la humedad del suelo, la temperatura ambiente y la radiación solar.

Capítulo 3 Definición modelo de red de objetos inteligentes inalámbricos

En este capítulo se presenta el modelo de red de objetos inteligentes inalámbricos, el cual incorpora conceptos de WSN en la Arquitectura de Interacción Semántica propuesta por **Niño Zambrano** [21] descrita en el capítulo anterior. El modelo cuenta con dos características: la implementación de topologías específicas y el uso de un algoritmo de inteligencia computacional para realizar el pronóstico en caso de que algunos de los nodos de la red dejen de funcionar.

3.1 Construcción del modelo de red de objetos inteligentes

Para la construcción del modelo se hizo uso de la metodología propuesta por Niño, et al. [71], la cual define tres etapas:

- Etapa 1: Definir el ámbito del modelo
- Etapa 2: Definir el modelo
- Etapa 3: Diseño de la prueba preliminar del modelo

3.2 Etapa 1: Determinar el ámbito del modelo

En este apartado se buscó mediante una revisión de la literatura establecer un marco teórico que obtuviera estudios relacionados con WSN, inteligencia computacional, riego inteligente, objetos inteligentes y su uso en la agricultura de precisión, para así lograr tener un marco teórico que permita definir el modelo. Como resultado de este proceso se obtuvieron las fichas bibliográficas correspondientes, las cuales pueden ser visualizadas en el Anexo A.

Dentro de las investigaciones encontradas en el estado del arte se evidenció que los desarrollos de agricultura de precisión desplegados a la intemperie presentan importantes desafíos en cuanto a la contingencia de las condiciones ambientales, debido que generan fallos en los dispositivos instalados dentro de los cultivos, encargados de monitorear las variables agroclimáticas, siendo esta información fundamental para realizar acciones sobre el mismo, como por ejemplo el riego. El aporte del presente proyecto al estado del arte en esta área es proponer un mecanismo para mitigar la pérdida de información agroclimática causada por los daños a los dispositivos generada por condiciones ambientales adversas, este mecanismo consiste en reemplazar la información perdida con información pronosticada por un algoritmo

de inteligencia computacional, y así permitir la consistencia de la información, necesaria para realizar el riego de precisión, además de ello, el presente proyecto propone un modelo para realizar despliegues en agricultura de precisión que no depende de una conexión a internet para realizar distintos tipos de procesamientos sobre la información recolectada del cultivo, siendo esto importante debido que en el contexto del campo colombiano, hay dificultades para acceder al servicio de internet por la infraestructura misma del país.

3.3 Etapa 2: Definir el modelo

Para definir el modelo de red de objetos inteligentes inalámbricos se decidió realizar una revisión bibliográfica con el fin de encontrar una metodología para adaptarla al desarrollo de un modelo relacionada con la agricultura de precisión; si bien se encontraron varios artículos con soluciones cercanas a este proyecto, ninguno presentaba un estándar o metodología específica que se pudiera adaptar, cada trabajo lo desarrolla a su manera, a pesar de esto, los artículos tienen una característica en común y es la definición de la arquitectura basada en capas [43], [45], [51], se identifican tres capas de acuerdo a su responsabilidad, la primera es la capa de adquisición de datos, la segunda es la capa de procesamiento e inteligencia y la tercera capa es la capa de aplicación. Por lo tanto, para construir la arquitectura del modelo de red de objetos inteligentes inalámbricos se establece una arquitectura de tres capas presentada a continuación:

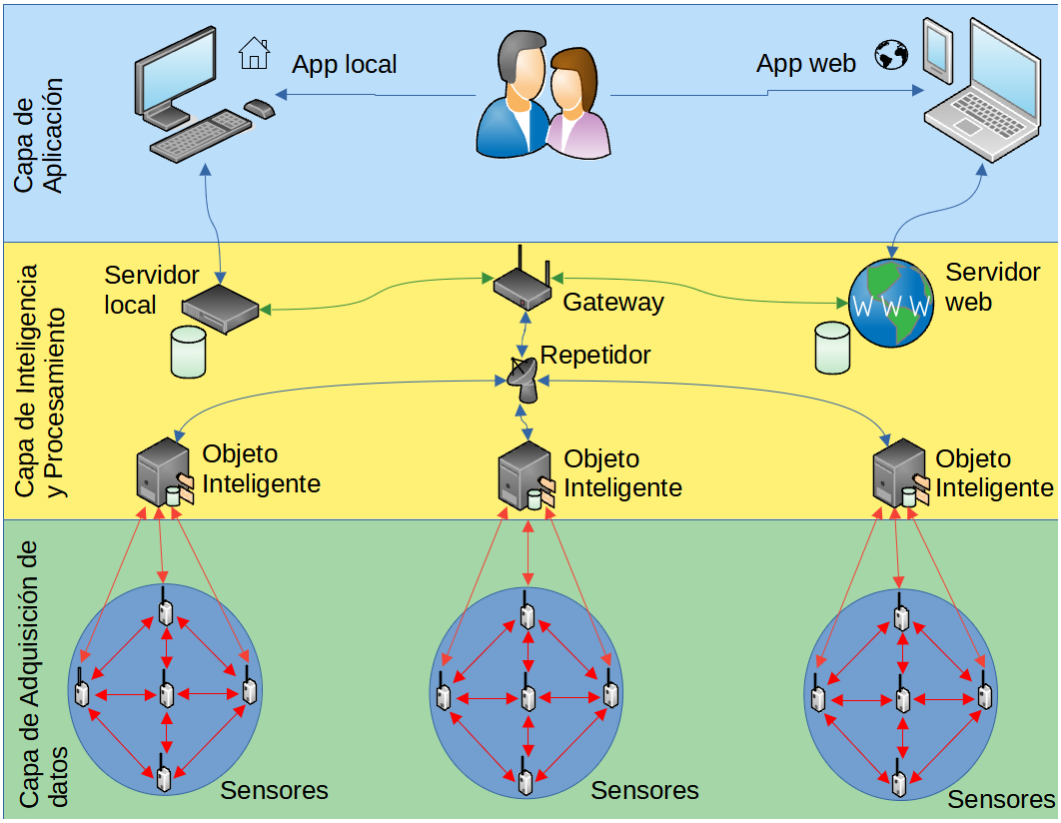


Figura 6 Arquitectura modelo de red
Fuente: Propia

Capa de aplicación

Esta capa es la encargada de ofrecer al agricultor una interfaz para realizar distintas consultas sobre la información agroclimática recolectada del cultivo, esta información es presentada mediante una interfaz gráfica de fácil entendimiento para los usuarios, dicha interfaz contiene gráficos como barras, tortas, histogramas, etc., la cual incluye información tiempo real, así como información histórica de las variables agroclimáticas del cultivo. En esta capa se incluye un sistema para realizar el riego inteligente basado en la información agroclimática además de un sistema de alertas por si ocurre alguna falla dentro de la red desplegada, la interfaz mencionada puede ser consultada de forma local gracias al servidor ubicado en la central de comunicaciones de la finca o a través de internet mediante la disposición de una aplicación web sencilla.

- **Riego inteligente:** el sistema de riego usa un microcontrolador encargado de recibir los datos de los sensores de humedad, temperatura y radiación solar desplegados en el campo, donde se especifica un umbral de temperatura, humedad y radiación solar, el microcontrolador recibe instrucciones para encender o apagar automáticamente las bombas de agua instaladas en el cultivo.

- **Alertas:** el sistema de alertas puede ser de gran ayuda en sistemas de monitoreo ya que pueden advertir de anomalías tales como datos fuera del promedio o fallas en la red además del estado del sistema del riego en el cultivo.

Capa de inteligencia y procesamiento de datos

Esta capa se encarga de la gestión de la información ofreciendo servicios que pueden ser consumidos por la capa de aplicación, además el sistema utiliza técnicas de IA para mitigar la perdida de datos ocasionados por fallas en la red mediante la predicción durante el tiempo que la red esté presentando errores, permitiendo así controlar acciones incorrectas en las tareas de riego. Esta capa se compone por los siguientes elementos:

- **Objeto inteligente:** el funcionamiento constante de la red de sensores es una de las características fundamentales de este modelo, para lograrlo se debe realizar un monitoreo constante de la red con la finalidad de identificar fallas relacionadas con el hardware más específicamente el fallo de un nodo, hacer esto demanda capacidades avanzadas de procesamiento, es por esto que dicha tarea es realizada por el objeto inteligente, el cual puede pronosticar los datos del nodo que falla mediante el uso de técnicas de inteligencia computacional como ML, de esta manera el sistema de riego recibe información completa y constante sobre el cultivo para realizar la distribución de agua en el momento y con la cantidad adecuada.
- **Repetidor:** debido a la irregularidad de los terrenos donde están ubicados los cultivos pueden existir dificultades para la creación de la red, se hace necesario el uso de un elemento que sirva como puente entre el OI y la central de comunicaciones para lograr cubrir grandes distancias y superar diversos obstáculos en la línea de vista, por lo anterior se propone que en este modelo se use un repetidor.
- **Gateway:** el modelo propone dos sistemas de almacenamiento, una de forma local y otro en la nube, se presenta la necesidad de dirigir los datos recolectados a los dos tipos de alojamiento de información, por lo anterior se propone el uso de un Gateway para la realización de esta tarea.
- **Servidor local:** recibe toda la información recolectada por parte de los nodos conectados al objeto inteligente, se plantea un servidor local debido a que muchas fincas donde se ubican los cultivos no cuentan con fácil acceso al servicio de internet además permite que el agricultor consulte de una manera fácil y rápida la información del cultivo.

- **Servidor web:** está en capacidad de almacenar grandes cantidades de información para posteriormente realizar procedimientos complejos, además ofrece la posibilidad de consultar la información almacenada en cualquier parte del mundo.

Capa de adquisición de datos

Es la capa más cercana a la fuente de datos, en este caso es el cultivo, debido a esto es la encargada de gestionar los dispositivos sensores para monitorear las variables agroclimáticas del cultivo, siendo éste el insumo principal para las capas superiores, además de adquirir los datos debe poder transmitirlos gestionando los distintos protocolos de comunicación inalámbrica. Esta capa está conformada por:

- Microcontrolador
- Sensor de humedad
- Sensor de temperatura
- Sensor de radiación solar o luminosidad

Posterior a la definición de la arquitectura del modelo planteado, se hace necesario definir la red de objetos inteligentes inalámbricos así como sus características, esta definición se logra mediante una metodología, para esto se realiza una búsqueda informal que permita identificar una metodología para el diseño de redes, la metodología denominada Metodología Para El Diseño De Una Red De Sensores Inalámbricos encontrada consta de 7 pasos [72], los cuales se definen a continuación:

- Paso 1 Objetivo de la red en el entorno
- Paso 2 Características del entorno
- Paso 3 Conocimiento técnico de los dispositivos WSN a usar
- Paso 4 Topología de red
- Paso 5 Procesamiento de información
- Paso 6 Pruebas dispositivos inalámbricos
- Paso 7 Implementación del modelo de red propuesto

A continuación, se ejecutan cada uno de los pasos planteados por la metodología:

Paso 1: Objetivo de la red en el entorno

Dentro de los estudios realizados y con el enfoque de solución de este proyecto que se centra en la agricultura de precisión y más específicamente en el riego inteligente de cultivos de café, se vio la necesidad de implantar sensores y actuadores en lugares remotos para el monitoreo de diferentes variables agroclimáticas como lo son la temperatura, humedad del suelo y radiación solar, con el fin de realizar múltiples tareas y específicamente riego, a partir de estas variables se encuentra la clave para obtener el máximo rendimiento de los cultivos así como el uso óptimo de los recursos ambientales disponibles, por tal razón se deciden monitorear y gestionar la información incorporando un objeto inteligente a través de las variables anteriormente mencionadas.

Paso 2: Características del entorno

El escenario elegido es un cultivo de café ubicado en la zona rural del municipio de Popayán, más específicamente en la vereda Santana, donde posiblemente existe microclima y existe irregularidad en el relieve entre la central y el área de monitoreo (ver Figura 7). La central de comunicaciones va a estar ubicada en la casa, aproximadamente a 450 metros en línea recta de la zona de despliegue de los dispositivos (ver Figura 8). En la Figura 9 se resalta el área del cultivo a monitorear.

Figura 7 Relieve central de comunicaciones y el cultivo
Fuente: Google Maps

Figura 8 Distancia centro de comunicaciones a cultivo
Fuente: Google Maps

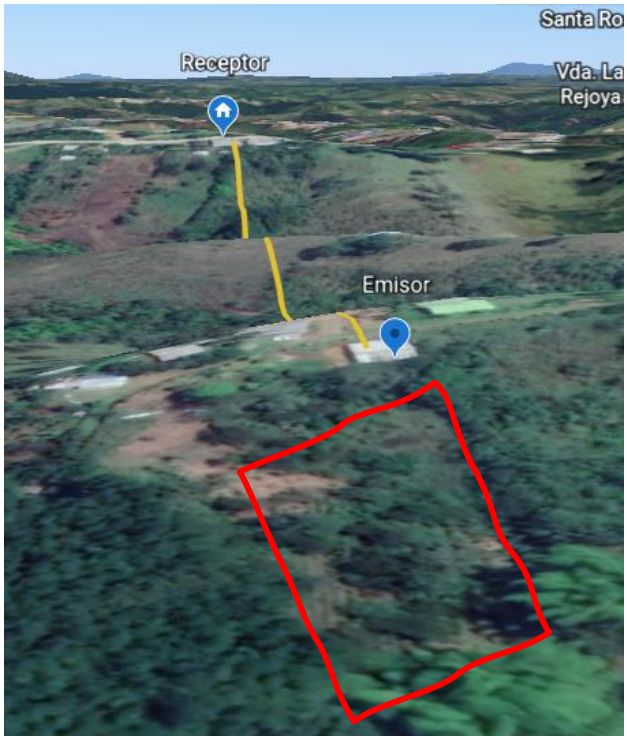


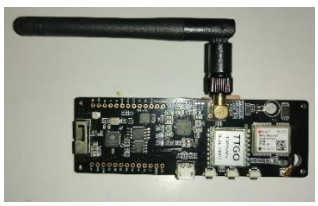
Figura 9 Ubicación cultivo a monitorear
Fuente: Google Maps

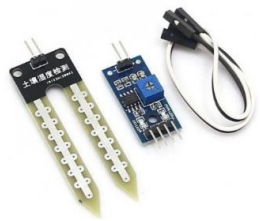
Paso 3: Conocimiento técnico de los dispositivos WSN a usar

Teniendo en cuenta que se necesitan cubrir largas distancias para poder transmitir información entre la zona de despliegue y la central de comunicaciones, se deben usar dispositivos que tengan la capacidad de usar un protocolo de comunicación de larga distancia, uno de los protocolos más conocido es denominado LoRa (Long Range), el cual es ampliamente usado en este tipo de despliegues, a partir de lo anterior y para implementar el modelo propuesto se debe hacer uso de los siguientes dispositivos con capacidad de enviar información a larga distancia: Gateway LoRa, modulo LoRa/GPS HAT el cual es usado como expansión de la Raspberry Pi 3 y un repetidor LoRa, también se usaron los dispositivos TTGO ESP32 y ESP32 DEVKITV1

que cuentan con conectividad inalámbrica Wifi y Bluetooth, utilizados para para cubrir el área de monitoreo (100m x 100m) aproximadamente. Debido a la diversidad de implementaciones de dichos elementos y los buenos resultados obtenidos en diferentes campos de la ciencia especialmente en el campo de agricultura de precisión, en consecuencia, se promueve la exploración de las tecnologías obteniendo una elevada cantidad de guías y tutoriales alojados en repositorios web, además de ello, se cuenta con alta disponibilidad de equipos en el mercado, donde se resaltan los bajos costos siendo viables para el proyecto. Cabe aclarar que el uso los dispositivos no es requerimiento para dicha implementación puesto que en el campo hay muchos más dispositivos con diversas tecnologías que pueden cubrir las necesidades, pero su uso puede inferir una curva de aprendizaje y configuraciones adicionales para el funcionamiento de la red. La descripción de cada uno de los elementos usados es presentada en la Tabla 7.

 **RASPBERRY PI3**	**PROCESADOR**	Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC
	FRECUENCIA DE RELOJ	1,4 GHz
	MEMORIA RAM	1 GB
	CONECTIVIDAD INALÁMBRICA	2.4GHz / 5GHz IEEE 802.11.b/g/n/ac Bluetooth 4.2, BLE
	CONECTIVIDAD DE RED	Gigabit Ethernet over USB 2.0 300Mbps
	PUERTOS	GPIO 40 pines HDMI 4 x USB 2.0 CSI (cámara Raspberry Pi) DSI (pantalla táctil) Toma auriculares / vídeo compuesto Micro SD Micro USB (alimentación) Power-over-Ethernet (PoE)

 **ESP32 DEVKITV1**	**VOLTAJE ALIMENTACION**	5V DC vía USB
	VOLTAJES DE ENTRADAS/SALIDA	3.3V DC
	CONSUMO ENERGIA	5µA en modo de suspensión
	PINES DIGITALES GPIO	24 algunos disponibles solo como entradas
	CONVERSOR ANALÓGICO DIGITAL	Dos ADC de 12bits tipo SAR
	CPU	Tensilica Xtensa 32-bit LX6
	FRECUENCIA DE RELOJ	hasta 240Mhz
	CONECTIVIDAD	Wifi: 802.11 b/g/n/e/i (802.11n @ 2.4 GHz hasta 150 Mbit/s) Bluetooth: 4.2 BR/EDR BLE Modo de control dual
	SEGURIDAD	IEEE 802.11, incluyendo WFA, WPA/WPA2 y WAPI
 **TTGO ESP32**	**ENTRADA**	USB 5V/1ª
	ENTRADA BATERIA	3.7-4.2V
	FRECUENCIA DE FUNCIONAMIENTO RED LORA	915 Mhz
	RAM	8 MB
	FLASH	4 MB
	CONECTIVIDAD	Wifi Bluetooth
	GPS	Módulos GPS NEO-6M

 **GATEWAY DRAGINO LG 308**	**ENTRADA ENERGIA**	12v, 1ª
	PUERTOS	Puertos RJ45 de 10M/100M x 2 1 x Wifi 2.4G (802.11 bgn) 1 puerto de host USB 1 x interfaz Mini-PCIe
	RANURA	Micro SIM
	VELOCIDAD DE DATOS	Enlace descendente de hasta 100 Mbps y de enlace ascendente de 50 Mbps
	BANDA DE FRECUENCIA	US915, AU915, AS923, KR920
 **LORA GPS/HAT**	**Banda de frecuencia**	915 MHZ
	compatibilidad	Raspberry Pi2 modelo B Raspberry Pi 3
	Modem	Lora LX 1279 GPS
	Modulación	FSK, GFSK, MSK, GMSK, LoRa™ y OOK
 **Sensor de humedad del suelo YL-69**	**Pines**	Alimentación: VCC Tierra: GND Salida Digital: D0 Salida analógica: A0
	Voltaje Alimentación	2 voltios a 6 voltios
	LED	Indicador de encendido
	Pines	Alimentación: VCC Tierra: GND Salida analógica: A0
	Precisión	±0.5°C
	Pendiente	10mV/ºC

 **Sensor de temperatura LM-35**	**Voltaje alimentación**	4V – 30V (5V recomendado)
	Rango de temperatura	-10°C hasta +85°C
 **Fotorresistor LDR**	**Voltaje máximo**	6 voltios
	Tiempo de respuesta:	20ms
	variación aproximada	0.5K Ohm (luz día) a 3M Ohm (oscuridad)

Tabla 7 Dispositivos WSN

Paso 4: Topología de red

La red que se propone en este proyecto implementa una topología hibrida, la cual está definida en dos niveles descritos a continuación:

Nivel 1: Es el nivel superior de la red, donde están presentes las conexiones entre los objetos inteligentes y el repetidor o router formando una topología en estrella Figura 10. El repetidor y la central (gateway) presentan una conexión punto a punto para la transferencia de datos a internet y al servidor local, esto con el fin de aumentar la velocidad de transmisión al no tener que recorrer más caminos.

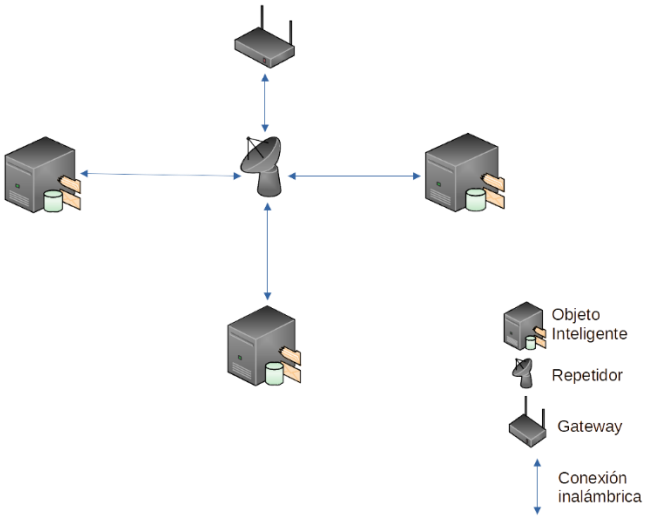


Figura 10 Topología en estrella
Fuente: Propia

Nivel 2: Es el nivel que está en contacto directo con el entorno, por eso está compuesto por los sensores y actuadores conectados a un microcontrolador, y este a su vez está conectado al objeto inteligente definiendo una topología en malla entre ellos (ver Figura 11), estos se comunican entre sí por medio de tecnologías inalámbricas. Esta topología permite redundancia de rutas y cooperación entre los nodos para garantizar la transferencia de los datos incluso si se presentan fallas de hardware en alguno de los dispositivos.

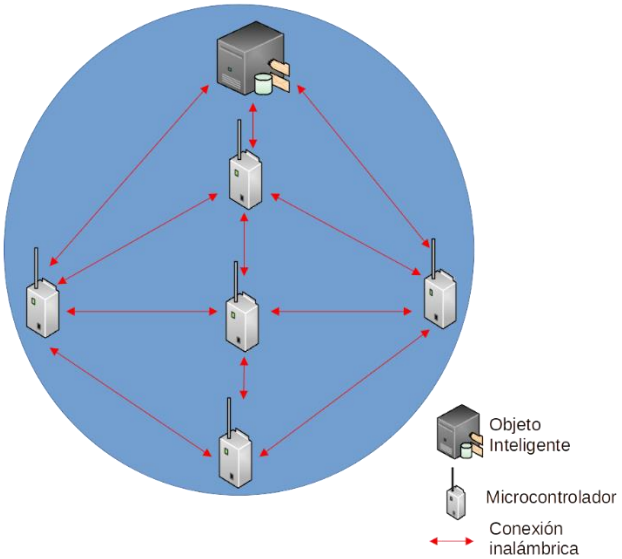


Figura 11 Topología en malla
Fuente: Propia

La topología híbrida resulta de unir las dos topologías anteriores (malla – estrella) obteniendo una topología de dos niveles como se ve en la Figura 12.

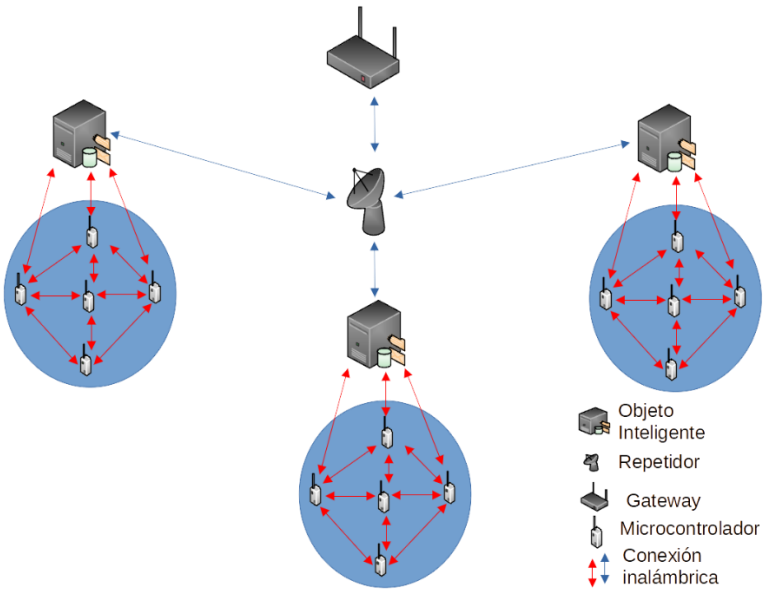


Figura 12 Topología hibrida
Fuente: Propia

Paso 5: Procesamiento de información

En este paso se realizó la configuración dentro del objeto inteligente para añadir la capacidad de realizar procesamientos avanzados sobre la información recolectada por los distintos sensores instalados en el cultivo, además de la capacidad de monitorear los dispositivos vinculados al objeto inteligente. Dichos procesamientos incluyen la recepción de la información recolectada por la malla de sensores al objeto inteligente, la transferencia de información almacenada en el objeto inteligente al servidor local y por último la predicción de la información basado en el histórico de cada sensor que integra la malla en caso de que deje de funcionar por fallas de hardware o relacionadas, dichas fallas deben ser detectadas por el servidor de monitoreo instalado y configurado en el objeto inteligente.

Paso 6: Pruebas dispositivos

Una vez se realicen las configuraciones mencionadas en el paso anterior, en este paso se realizan diferentes pruebas técnicas sobre cada uno de los dispositivos, así como también pruebas a cada una de las configuraciones, dentro de las que se encuentran módulo de comunicación de larga distancia, comunicación de dispositivos de larga distancia, comunicación de la red de sensores con el objeto inteligente además de la recepción y transmisión de mensajes.

Paso 7: Implementación del modelo de red propuesto

Este paso coincide con la Etapa 3 de la metodología propuesta por Niño, et al. [71], el cual será desarrollado en el siguiente capítulo debido que se debe implementar el modelo ya integrado con la arquitectura de interacción semántica para validarlo.

3.4 Etapa 3: Diseño de la prueba preliminar del modelo

Teniendo en cuenta que la fase de evaluación del modelo está ligada a la implementación y posterior validación de la prueba de concepto, esta etapa se presenta en el siguiente capitulo.

3.5 Algoritmos de inteligencia computacional que se pueden utilizar en el modelo

Dentro de la definición de machine learning expuesta en el marco teórico se especifica cada uno de sus enfoques, obteniendo entre ellos el aprendizaje supervisado, este tipo de aprendizaje tiene como enfoque en la agricultura de precisión realizar clasificación o pronósticos de datos dentro de los cultivos, este requiere de un conjunto de datos previamente etiquetado para realizar el entrenamiento del algoritmo y posteriormente la clasificación y/o pronóstico.

Dentro de las técnicas de aprendizaje supervisado enfocadas en pronósticos de datos, se encuentra las técnicas de regresión donde se puede encontrar el algoritmo de regresión lineal, el cual es tenido en cuenta para su uso en el modelo de red de objetos inteligentes inalámbricos propuesto en este proyecto, debido a dos aspectos, primero, el modelo de red propuesto proporciona el conjunto de datos identificados necesarios para la configuración del algoritmo, dado que este tiene como objetivo recopilar y almacenar el historial de tres variables agroclimáticas del cultivo de café, las cuales son: radiación solar, temperatura ambiente y humedad del suelo, estos son monitoreadas para realizar el riego de precisión en el cultivo. Segundo, se busca que si alguno de los nodos que compone la red tiene problemas de hardware o de red, el algoritmo de regresión lineal pronostique el valor de la humedad del suelo a partir del conjunto de datos compuesto por la temperatura ambiente y la radiación solar, con la finalidad de mantener la consistencia de los datos recopilados y lograr realizar el riego de precisión, dado que uno de los objetivos de este proyecto está dirigido a prevenir o evitar la pérdida de datos ocasionados por problemas de hardware o conectividad, aclarando que el algoritmo usado no previene ni evita la perdida de datos, pero si mitiga la perdida de estos mediante la suplantación con los datos pronosticados por el algoritmo, teniendo en cuanta que el algoritmo de regresión lineal proporciona un coeficiente de determinación (R^2), el cual

permite verificar la correlación entre las variables usadas para entrenar el algoritmo, donde se obtienen valores de 0 a 1, indicando que entre más cercano al 1, habrá mayor fiabilidad de los datos pronosticados. Otra de las razones para elegir la técnica de regresión lineal es que su implementación no requiere de altos recursos de hardware y software, permitiendo su implementación en un microcomputador como la Raspberry Pi, además de tener como característica la fácil implementación en el lenguaje de programación Python gracias a las librerías que tiene disponibles.

3.6 Integración del modelo a la arquitectura de Niño-Zambrano

El presente proyecto es un aporte a la Arquitectura de Interacción Semántica realizada por Niño Zambrano [21] en su tesis doctoral, dicho aporte está enfocado en mejorar las capacidades de procesamiento de información al objeto inteligente utilizando técnicas de IA, además de cambiar la conectividad de los sensores y actuadores al objeto inteligente, bajo la modificación de la conexión directa de sensores simples a la conexión de una red de microcontroladores que ya tienen vinculados sensores. De modo general, para realizar la integración del Modelo de Red de Objetos Inteligentes Inalámbricos (ver Figura 6) a la Arquitectura de Interacción Semántica (ver Figura 3), se utilizó el enfoque de niveles de procesamiento de datos, dichos niveles están distribuidos de mayor a menor dependiendo de la capacidad de procesamiento de cada uno y la cercanía de estos al entorno, siendo el más complejo el nivel de procesamiento en la nube (CLOUD), seguido del nivel de procesamiento en la niebla (FOG) y por último el nivel de procesamiento en el borde (EDGE), logrando identificar cada uno de ellos en las capas que componen la arquitectura, con el fin de realizar los aportes correspondientes:

- En la Capa de Objeto se incorpora el concepto WSN de topología de red, la topología incorporada es una malla compuesta por nodos, esto modifica la conexión de sensores y actuadores directamente al objeto inteligente, por la conexión de la malla al objeto inteligente para la gestión de cada uno de los nodos de la red. También se incorpora en esta capa el mecanismo para mitigar la pérdida de datos en cada uno de los nodos de la red que son gestionados por el objeto inteligente y es la implementación del algoritmo de inteligencia para pronosticar datos para posteriormente suplanten los datos que se puedan perder si alguno de los nodos deja de funcionar.
- En la Capa de Objeto Semántico a partir de la configuración del OI se genera una ontología de objeto semántico relacionada con la agricultura de precisión.

- En la Capa de Servicio se agregan dos servidores locales, cada uno trabajando en dos niveles distintos de computación, el primero se encuentra a nivel EDGE donde se configura en el OI para almacenar la información de los sensores y actuadores conectados a los nodos de la malla y realizar distintos procesamientos, el segundo servidor está a nivel de FOG y se configura en un computador ubicado en la central de comunicaciones para almacenar la información de todos los objetos inteligentes desplegados en el cultivo con sus respectivas mallas de nodos de sensores y actuadores, para ofrecer la información en cualquier momento al agricultor sin la necesidad de estar conectado a internet.
- En la capa de aplicación se realiza la configuración donde se da a conocer los recursos (sensores y actuadores) al OI, los cuales en este caso están conectados por medio de una malla de nodos, esta configuración es dada con el fin de inicializar el objeto y poder generar los elementos (ejecutables) que permiten el uso de los recursos por el objeto. También se configura una herramienta de software libre llamada Grafana, la cual provee la capacidad de visualizar gráficos de manera fácil y rápida a partir de información almacenada ya sea en OI (Raspberry Pi) o en el servidor local.
- En la capa de middleware se hace uso del servidor web 'ThingSpeak' con la finalidad de recopilar la información obtenida de los objetos inteligentes mediante sus respectivas mallas de nodos de sensores.

La Figura 13 presenta la integración de forma gráfica al identificar el nivel de procesamiento por cada capa, además se resalta como principal aporte, el realizado en la Capa de Objeto donde se incluye el concepto de WSN de topología de red (topología de malla) a la Arquitectura de Interacción Semántica de objetos inteligentes propuesta por Niño Zambrano [21].

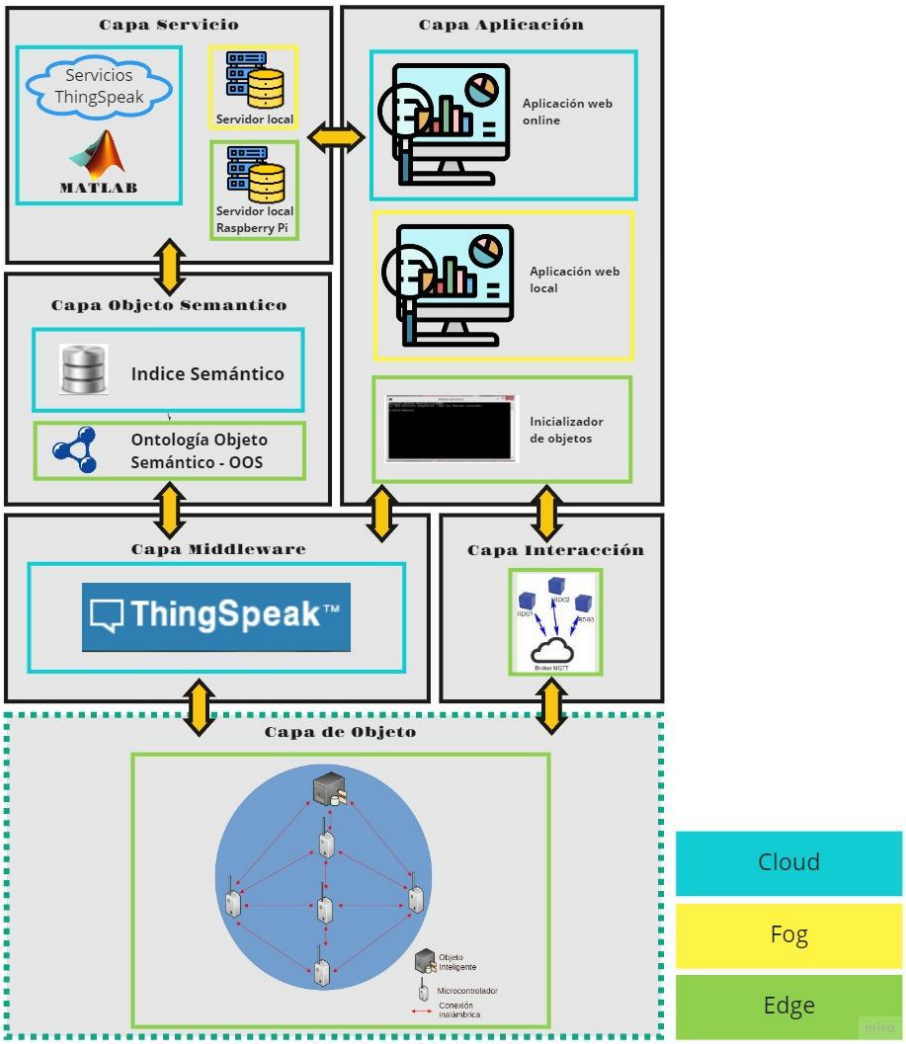


Figura 13 Integración con la arquitectura de integración semántica
Fuente: Propia

3.7 Restricciones del modelo

- La distancia entre nodos de la malla no puede superar los 50m de distancia en línea de vista directa, dado que el protocolo de comunicación puede presentar fallas o desconexiones en distancias más largas, en casos de presentarse obstáculos entre los nodos no debe superar los 30m.
- El modelo presenta un limitante o restricción en cuanto consumo de energía se refiere, dado que, al ser un modelo de red propuesto para riego inteligente, este requiere de una fuente de energía para su funcionamiento, adicional a esto se tiene que la tecnología de comunicación Wifi usada tiene un alto consumo de energía. Como propuesta a esta falencia se propone el uso de paneles solares en cada nodo acompañados de bancos de carga, los cuales puedan proporcionar una alimentación constante durante el

funcionamiento de la red. El uso de estas tecnologías permitiría una movilidad de los diferentes dispositivos sin depender de cableados engorrosos.

Para el caso del consumo energético por los nodos de la malla se propone crear tiempos en los que los nodos de la malla presenten un modo suspendido donde el consumo de energía es mínimo.

- El servidor IoT que se utiliza para este proyecto es 'ThingSpeak' [73], el cual necesita configuraciones extras para permitir escalabilidad a la hora de registro de nuevos datos al agregar un nodo a la malla, puesto la configuración estándar de dicho servidor no permite que se suba la información recolectada por ese nodo a la nube sin antes haber realizado las configuraciones necesarias.

Capítulo 4 Desarrollo prueba de concepto

En este capítulo se presenta el desarrollo de la prueba de concepto para validar el funcionamiento del modelo de red de objetos inteligentes inalámbricos propuesto, donde el objetivo inicial es realizar la prueba de concepto en un escenario simulado, pero la implementación de una simulación requiere inicialmente una revisión de las herramientas que permitan simular los diferentes elementos presentes en el modelo propuesto, además que la curva de aprendizaje de la herramienta de simulación es incierta, lo que podría dificultar el proceso de investigación y desviarse del enfoque del proyecto. Por lo anterior se decidió hacer un despliegue físico dentro del laboratorio que permita una relación más cercana con los dispositivos a usar, permitiendo conocer las ventajas, desventajas y características de estos, siendo un conocimiento útil al momento de hacer un despliegue en un ambiente real, y así descartar o validar su uso en el modelo propuesto.

A partir de lo anterior, la prueba de concepto se implementó siguiendo las fases especificadas en la estrategia metodológica propuesta por Omar Guzmán en su tesis de pregrado [74], dicha estrategia es adaptada al proceso desarrollado en este proyecto y está compuesta de cuatro fases Figura 14.

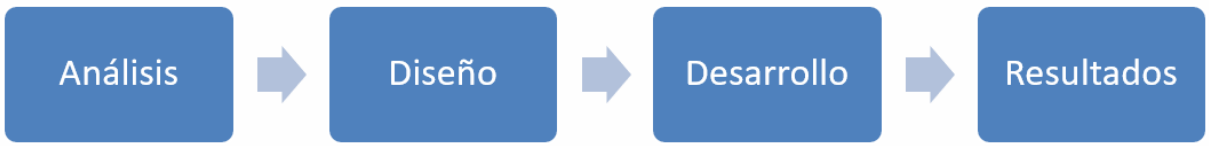


Figura 14 Fases de la estrategia metodológica
Fuente: Propia

La explicación de cada una de las fases es presentada a continuación.

4.1 Metodología para el desarrollo de la prueba de concepto:

Fase de Análisis: es la etapa inicial de la metodología, por lo cual contiene información que indica el contexto sobre el que se realiza las iteraciones, esto hace referencia al escenario donde se va a desplegar la red además de la definición del requerimiento que se espera alcanzar con el desarrollo de la iteración.

Fase de Diseño: se establecen las tecnologías que se proponen en el modelo y que permiten la interacción en el escenario IoT, también se establecen las tecnologías que permitirán las

configuraciones de los diferentes componentes del modelo como, por ejemplo, el Entorno de Desarrollo Integrado (IDE, *Integrated Development Environment*) y librerías. Se deben definir los prototipos a utilizar que mínimo serán 2, donde el prototipo 1 tendrá como objetivo resolver las diferentes funcionalidades del modelo propuesto en sus diferentes niveles y su integración para tener un modelo total, el prototipo 2 se hará una refinación del prototipo 1 a partir de las observaciones obtenidas en este incluyendo la configuración del objeto semántico para tener un prototipo funcional del modelo de red propuesto y poder evaluar su funcionalidad.

Fase de Desarrollo: esta etapa nos permite establecer o tener una perspectiva de las tecnologías y demás componentes usados en la construcción del modelo, permitiendo así obtener unos resultados para validarlos.

Para esta fase se propone la realización de una serie de iteraciones basado en pruebas funcionales sobre los dispositivos, para obtener una rápida realimentación y lograr apropiar conceptos, siendo adecuado para el desarrollo de un producto IoT o incluso de una prueba de concepto, permitiendo tener una visión global de los actores del proyecto y poder implementar el modelo propuesto en el capítulo anterior.

Las iteraciones identificadas para cumplir con la fase de Desarrollo de la prueba de concepto son las siguientes:

Iteración 1: El objetivo de esta iteración es contextualizar las herramientas y tecnologías disponibles para realizar la prueba de concepto, dándole continuación al paso 6 (Pruebas dispositivos) de la metodología propuesta en el capítulo anterior para el diseño del modelo de red.

Iteración 2: Tener un prototipo de red refinado a partir de la iteración anterior que permita recolectar datos a partir de la configuración de los nodos con sus sensores y actuadores y la inclusión de la arquitectura de objeto semántico a lo obtenido en la iteración 1.

Iteración 3: Realizar un análisis de los resultados obtenidos en las iteraciones anteriores para sugerir una realimentación y mejora al modelo.

Resultados: los resultados nos permiten una retroalimentación para la construcción de los prototipos posteriores donde se valida la funcionalidad del modelo con el prototipo final.

Al tener claro las fases de la metodología y haber identificado las iteraciones a realizar para cumplir con la prueba de concepto, se inició el desarrollo de cada una de las fases a continuación:

Análisis:

El modelo de red propuesto en este proyecto hace uso de la arquitectura de objeto semántico propuesta por PhD Miguel Ángel Niño Zambrano [21]. El usuario objetivo de este proyecto son los caficultores del departamento del Cauca, aclarando que la funcionalidad del modelo de red depende de la prueba de concepto actual y no podrá ser usada como producto final.

Esta prueba se relaciona con la implementación de la red incluida en el modelo propuesto en el capítulo anterior, esto con la finalidad de enviar y recibir datos simples entre todos los componentes del modelo, para ello se hace necesario realizar distintas pruebas sobre los dispositivos para conocer las tecnologías y protocolos de comunicación inalámbrica que maneja cada uno, con el fin de realizar las conexiones necesarias teniendo en cuenta la topología de red híbrida propuesta en el modelo, la cual está compuesta en el primer nivel por una topología en estrella y el segundo nivel presenta una topología en malla, para lograr implementar estas conexiones se deben estudiar las tecnologías y protocolos que permitan la interacción de los diferentes componentes del modelo de red propuesto teniendo presente los niveles definidos.

Para la implementación de la red de sensores inalámbricos especificada dentro del modelo se debe dividir los elementos a utilizar en los dos niveles que se especifican en la topología híbrida, el primer nivel está compuesto por cinco microcontroladores ESP32 con módulo de conexión Wifi, los cuales son programados mediante el entorno de desarrollo Arduino instalado en un computador con sistema operativo Windows 10 para formar la malla de sensores, la cual se conecta a una Raspberry Pi 3 con sistema operativo Raspbian, encargada de recibir los datos recolectados por la malla, donde se ejecutan dos tareas: la primera almacenar dichos datos en una base de datos local, la segunda tarea hace parte del segundo nivel donde se implementa una topología en estrella para enviar los datos mediante un protocolo de larga distancia como LoRa a un servidor local para almacenar los datos y ser subidos a un servidor IoT como lo es ThingSpeak.

Lo presentado en el capítulo anterior hace parte de la primera iteración donde se busca tener una conexión entre los niveles del modelo de red propuesto, para la segunda iteración se tiene un escenario donde se busca incluir la arquitectura de objeto semántico en la red obtenida a partir de la iteración 1 donde se tiene como recurso principal el regulador de humedad que consta de una malla de nodos donde cada nodo está relacionado con un sensor de humedad, un sensor de temperatura y un dispositivo de riego emulado por un diodo emisor de luz. En resumen, se utilizan los siguientes elementos:

- Microcontroladores: usados para crear los nodos de la malla, a los cuales se conectan los sensores y actuadores, también usados como mediadores para la conexión de la malla con la Raspberry Pi (OI)
    - 2 ESP32 T-Beam
    - 3 ESP32 Dev Kit V1
- Raspberry Pi 3: dispositivo usado para el despliegue de la arquitectura del objeto semántico y procesamiento y almacenamiento de información.
- Celular: usado para visualizar mediante puerto serial el funcionamiento de un nodo de la malla.
- Adaptador OTP: permite la conexión entre el microcontrolador y el celular vía USB para visualizar datos del puerto serial.
- Computador: dispositivo usado para configuración de microcontroladores a partir de IDE adecuado.
- 5 cables de datos tipo c: permiten la conexión de los dispositivos (Microcontroladores) con el computador o celular para su configuración o visualización de funcionalidad mediante puerto serial.
- Sensores de humedad: dispositivo usado para capturar la humedad del suelo (sensor HL-69)
- Sensores de temperatura: dispositivo usado para capturar la temperatura (sensor LM35)
- Sensor de radiación solara: dispositivo usado para capturar de la luz solar (Foto resistencia LDR)
- Diodo emisor de luz (Emular dispositivo de riego): usado para indicar el estado del riego

A partir de la definición del modelo se obtiene un listado de requerimientos divididos de acuerdo con la iteración que corresponda, dichos requerimientos son presentados a continuación:

Iteración 1

- **Creación de una malla de sensores:** permitir una conexión de dispositivos (placas ESP32) mediante protocolo Wifi, para que interactúen entre ellas mediante mensajes simples.
- **Conexión de la Raspberry Pi a la malla:** permitir tener un dispositivo con mayor capacidad de procesamiento que reciba los datos de la malla.

- **Almacenamiento de datos aleatorios en una base datos relacional instalada en la Raspberry Pi:** permitir la captura los datos de la malla en la Raspberry Pi para posteriormente ser almacenados en una base de datos local.
- **Configuración de módulo 'LoRa HAT' instalado en la Raspberry Pi para intercambio de mensajes:** configurar el módulo LoRa como una extensión de la Raspberry Pi para una conexión de larga distancia para el intercambio de datos.
- **Envió de datos a servidor web IoT:** permitir almacenar datos en la nube para tener acceso a la información de forma remota.

Iteración 2:

- **Configuración del entorno objeto semántico en la Raspberry Pi:** configurar el entorno de objeto semántico para crear el regulador de riego.
- **Instalación de sensores a los nodos de la malla:** se requiere la instalación y configuración del sensor de humedad del suelo, temperatura y radiación solar en cada nodo de la malla.
- **Almacenamiento de datos producidos por los nodos de la malla en una base de datos relacional instalada en la Raspberry Pi:** permitir la captura de los datos obtenidos por los sensores de cada uno de los nodos de la malla en la Raspberry Pi para posteriormente ser almacenados en una base de datos local.
- **Monitoreo de la malla:** permitir monitorear la malla para verificar fallas de los nodos.
- **Implementación algoritmo inteligente:** se requiere implementar un algoritmo inteligente que permita pronosticar las variables agroclimáticas en caso de fallas de nodos de la malla.
- **Almacenamiento de los datos pronosticados:** permitir que los datos que se pronostiquen sean registrados en la base de datos sin caer en conflicto con los demás datos que se están registrando de la malla.
- **Graficación de los datos almacenados en el base de datos:** configuración de la herramienta Grafana en el Raspberry Pi para graficar en series de tiempo los datos sobre cada una de las variables agroclimáticas almacenadas en la base de datos.

Diseño:

Para esta fase se tiene un enfoque incremental del diseño del modelo de red, empezando desde un prototipo inicial que consiste en la creación e interacción de la malla hasta tener un prototipo

final de la red propuesta donde sea posible transmitir la información recolectada por los sensores hasta las capas superiores del modelo, para lograrlo se usan las siguientes tecnologías:

- Protocolos de comunicación inalámbrico: Wifi y LoRa
- Protocolo de mensajería ligero MQTT
- Entorno de Desarrollo: IDE Arduino
- Sistema Operativo: Raspbian, Windows 10
- Librería:
    - rpi-lora-tranceiver-master
    - Painless Mesh
    - Painless Mesh Boost
    - MQTT
    - JSON
- Lenguaje de programación: Python

Desarrollo:

Las configuraciones y procesos pertinentes para el desarrollo de las tres iteraciones serán presentadas en el Anexo B.

4.2 Iteración 1

Para cumplir con el propósito de la iteración 1 se hace necesario el desarrollo de los siguientes requerimientos:

Creación de una malla de sensores:

En este proceso se identificaron las características de las placas, las cuales nos permitieron identificar que comparten características como los módulos Wifi y Bluetooth, donde a partir de nuestra necesidad de crear una topología en malla, se hace una exploración informal de información que nos permita identificar herramientas o métodos para crear una topología en malla entre las placas mediante una conexión Wifi, la cual se pudo observar un alto uso de la librería Painless Mesh, sobre la cual se encuentra gran cantidad de información presente en tutoriales que permiten comprender su instalación y funcionamiento.

El montaje de la malla hace necesario la configuración de un IDE, el cual en nuestro caso basados en nuestro conocimiento dentro de la herramienta y por la amplia compatibilidad con

la placa ESP32 se procede con la descarga e instalación de la última versión del IDE de Arduino (v1.8.16).

Con la información recolectada se hace necesario instalar la librería Painless Mesh en el IDE para la posterior configuración de las placas, teniendo la librería instalada se hace uso del ejemplo StartHere, el cual es proporcionado por la misma librería, este es cargado en cada una de las placas mediante el IDE de Arduino, donde se puede visualizar la interacción de los nodos en la malla creada, la interacción también puede ser visualizada en momentos que se agregue un nuevo nodo, que se desconecte un nodo o en general cuando ocurra algún cambio dentro de las malla.

Resultados:

- Gracias a la amplia documentación acerca de la librería Painless Mesh su uso y configuración resultó un proceso de baja complejidad debido que solo se requiere la carga del código base mediante un IDE a las placas para su funcionamiento.
- Se evidenció una interacción entre los dispositivos donde se envían y reciben mensajes entre los diferentes nodos (placas ESP32), con un mensaje predeterminado por la librería donde se puede resaltar que cada nodo es identificado con un id de tipo entero.
- La librería Painless Mesh cuenta con diferentes funciones, las cuales informan los distintos tipos de eventos en la malla, estos eventos son: la conexión de un nuevo nodo, la desconexión de un nodo, la transmisión y recepción de mensajes entre los nodos.
- La librería presenta una gran escalabilidad en cuanto a nodos se refiere donde solo es necesario cargar el código a una placa nueva con la configuración adecuada y esta se unirá a la malla sin causar conflictos en la misma
- Al momento de fallo de alguno de los nodos la malla garantiza el envío de mensajes de los nodos en funcionamiento, esto debido a su redundancia de conexiones entre nodos permitiendo así mantener el funcionamiento de la malla. Esto se pudo verificar ya que se retira un nodo manualmente y la interacción entre los demás nodos se mantiene.
- Se logró obtener una interacción de 5 nodos que formaron la malla en el prototipo inicial.

Conexión de la Raspberry Pi a la malla:

Para continuar con la construcción del modelo se hace necesario la inclusión de la Raspberry Pi a la malla creada anteriormente, en esta etapa se exploraron dos posibles soluciones, las cuales consistían en:

- Dentro de las exploraciones realizadas se identifica una librería basada en Painless Mesh llamada Painless Mesh Boost, la cual al ser instalada en la Raspberry Pi y con ayuda de un nodo de la malla configurado como intermediario o puente se permite la conexión entre la malla y la Raspberry, la configuración del puente se hace mediante el sketch "Bridge" proporcionado en los ejemplos de la librería Painless Mesh.
- Como segunda exploración se ubicó dentro de los ejemplos proporcionados por la librería un ejemplo llamado puente MQTT, el cual funciona como intermediario entre la malla de sensores y la Raspberry Pi, donde uno de los nodos de la malla fue configurado como puente MQTT, también se hizo necesaria la instalación y configuración de un bróker MQTT en la Raspberry, para este caso se usó Mosquitto. De igual manera se requirió de un acceso a Wifi en común entre la Raspberry Pi y el puente MQTT, del cual se procedió a configurar una placa ESP32 como punto de acceso Wifi que nos permitiera la interacción entre el bróker MQTT y los clientes suscriptores/publicadores.

Resultados:

- Dentro de la primera solución explorada se evidencia una interacción entre la malla y la Raspberry Pi, pero se presenta un inconveniente al intentar modificar elementos del script que genera el ejecutable a instalar en la Rasberry Pi, puesto que se realizan cambios y estos no se ven reflejados en el momento de crear e instalar el nuevo ejecutable. Lo que dificulto la identificación de la obtención de los datos para su visualización en pantalla (Ventana de comandos).
- La librería Painless Mesh Boost es basada en la librería Painless Mesh desarrollada en el lenguaje de programación de alto nivel C++, la cual para su uso fue necesario compilar el código fuente para crear un ejecutable que permitió mediante su ejecución la recepción de mensajes desde la malla pasando por el puente de manera exitosa, pero cuando se realizaban modificaciones sobre el código para entender su funcionamiento los cambios no se vieron reflejados al generar y ejecutar el nuevo ejecutable dificultando la comprensión del código y en consecuencia la búsqueda de una nueva solución.

- En la segunda solución explorada se pudo obtener una interacción entre la malla y la Raspberry Pi mediante el protocolo MQTT, el cual nos proporciona múltiples funcionalidades como lo son obtener datos de toda la malla, obtener datos de un nodo especifico de la malla, enviar un mensaje a un nodo especifico o a todos los nodos de la malla mediante los diferentes tópicos creados en el nodo configurado con el sketch "bridgeMQTT".
- La segunda solución nos permitió manejar una cobertura más amplia entre la Raspberry Pi y el puente MQTT de la malla de aproximadamente 100 metros en línea de vista directa y 60m con obstáculos, dado que entre ellos se encuentra un punto de acceso que nos permitió obtener una distancia más amplia.
- El uso del bróker MQTT como solución presentó un nivel de dificultad intermedio-bajo, dado que, si bien el código del ejemplo del puente MQTT es claro y a la vez el manejo del bróker instalado en la Raspberry Pi fue sencillo, la dificultad se presentó al momento de entender el concepto del manejo de los puertos dentro del sistema operativo de la Raspberry Pi, donde fue necesario añadir algunas configuraciones al archivo de configuración del bróker Mosquitto (mosquitto.conf) para abrir los puertos, una vez esto se solucionó se pudieron recibir y enviar mensajes desde la Raspberry Pi de forma rápida.

Almacenamiento de datos aleatorios en una base datos relacional instalada en la Raspberry Pi:

Para el almacenamiento de datos se realizaron dos métodos a partir de los resultados obtenidos en la prueba anterior, los cuales consistieron en:

- Primero, la escritura en un archivo de texto plano, en el cual se buscaba capturar los datos obtenidos en la Raspberry Pi provenientes de la malla mediante la suscripción a un tópico del bróker MQTT.
- Segundo, el almacenamiento de datos en una base de datos relacional en este caso MySQL. Para la creación de la base de datos y la tabla donde se van a guardar los datos se utiliza la aplicación web phpMyAdmin, la cual proporciona la capacidad para administrar las bases de datos MySQL, esta base datos recibe el nombre de 'agroiot' con una única tabla nombrada 'info_nodo', la cual contiene los campos para el id del nodo, las variables agroclimáticas monitoreadas (temperatura, humedad y radiación solar), el tiempo de monitoreo y el tiempo de llegada a la Raspberry Pi. Para el proceso

de inserción de datos a la BD, se transforma la cadena recibida en el bróker MQTT a un formato tipo JSON para manejar los datos de manera más clara, para esto se crea un código en Python que escucha de forma permanente el bróker al suscribirse al tópico adecuado.

Resultados:

- En el proceso de implementación de la primera solución se entendió el manejo del bróker MQTT mediante el lenguaje de programación Python utilizando librerías externas, esto permitió interactuar con el bróker logrando capturar los mensajes que allí se recibían.
- Al momento de implementar la segunda solución, el conocimiento adquirido en la primera solución fue de gran utilidad debido a que solo se dedicó el esfuerzo a configurar la conexión de la BD en Python.
- La herramienta web phpMyAdmin permite realizar la gestión de la BD mediante una interfaz gráfica que agiliza en gran medida la creación de bases de datos, las tablas que contiene, así como de las columnas allí presentes con los tipos de datos correspondientes.
- Los objetos tipo JSON brindan relativa facilidad para ser tratados dentro de Python debido que la información se puede buscar por palabras claves relacionadas con el valor contenido.

Configuración de módulo 'LoRa/GPS HAT' instalado en la Raspberry Pi para envío de mensajes mediante LoRa:

En esta etapa se realizaron 2 pruebas que nos permitieran la interacción entre dos Raspberry Pi con la expansión del módulo LoRa/GPS HAT y una prueba realizada mediante cableado de pines:

En la primera prueba se explora la librería proporcionada por el fabricante del HAT, llamada Dragino, la cual se puede encontrar como rpi-lora-tranceiver-master, este se encuentra desarrollado en el lenguaje C, el cual debe ser descargado en la Raspberry Pi y posteriormente instalado, el procedimiento puede ser visto en el manual del módulo LoRa ejemplo 3 donde al ejecutarlo es necesario pasar un parámetro para identificar si es emisor Figura 15 o receptor Figura 16.

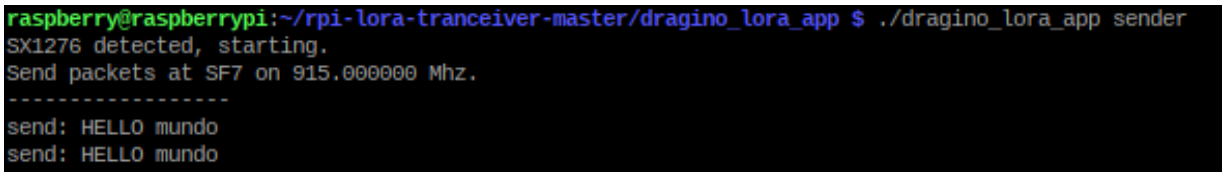


Figura 15 Ejecución ejemplo LoRa parámetro "sender"
Fuente: Propia

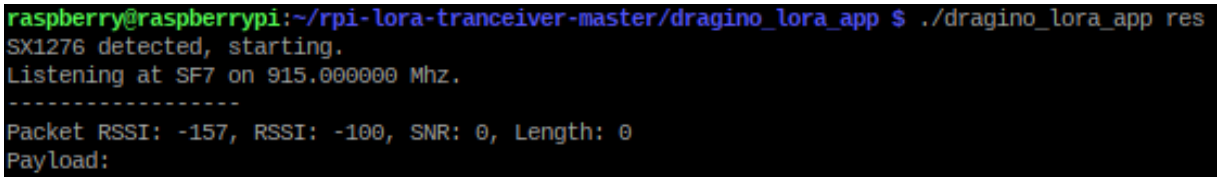


Figura 16 Ejecución ejemplo LoRa parámetro "res"
Fuente: Propia

Para la segunda prueba se buscó trabajar directamente sobre el módulo LoRa que se encuentra en el HAT, mediante la librería de Python dedicada al módulo especifico, en nuestro caso se debió explorar las librerías para el módulo SX1276 y el módulo RFM96. Para el manejo de estas se debe mapear los pines correspondientes del módulo LoRa (ver Figura 17) que se conectan a la Raspberry Pi mediante los puertos o pines GPIO (ver Figura 18) correspondientes, seguido se ejecutaron los códigos ejemplos que proporciona la librería con resultados no muy favorables que serán descritos en la sección de resultados.

Para la prueba realizada mediante el cableado de pines entre el LoRa Hat y la Raspberry Pi (ver Figura 19) se hizo necesario conocer los pines correspondientes del módulo LoRa del Hat (ver Figura 17) y los pines GPIO de la Raspberry Pi (ver Figura 18) que permitiera un acceso especifico o directo al módulo en mención.

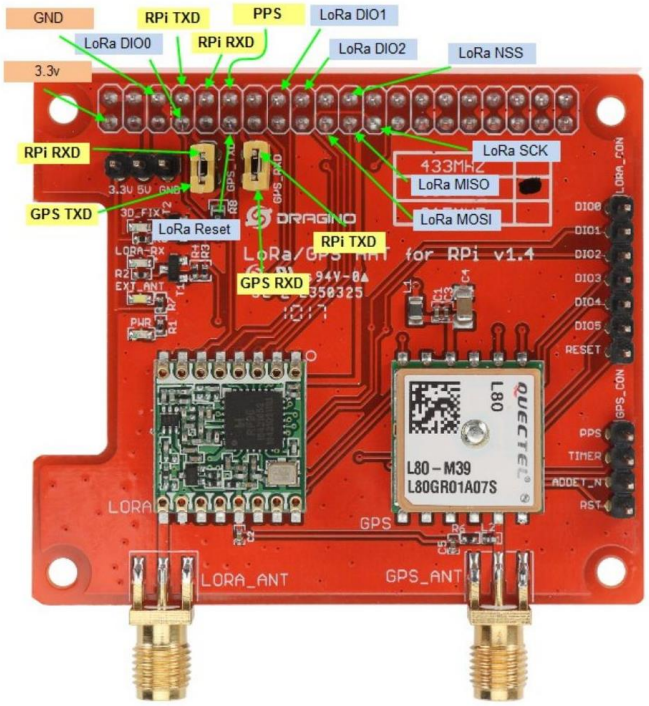


Figura 17 Pines LoRa/GPS HAT
Fuente: [75]

```
raspberry@raspberrypi:~ $ gpio readall
 +-----+-----+---------+------+---+---Pi 3B--+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 |     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
 |   2 |   8 |   SDA.1 |   IN | 1 |  3 || 4  |   |      | 5v      |     |     |
 |   3 |   9 |   SCL.1 |   IN | 1 |  5 || 6  |   |      | 0v      |     |     |
 |   4 |   7 | GPIO. 7 |   IN | 0 |  7 || 8  | 1 | IN   | TxD     | 15  | 14  |
 |     |     |      0v |      |   |  9 || 10 | 1 | IN   | RxD     | 16  | 15  |
 |  17 |   0 | GPIO. 0 |   IN | 1 | 11 || 12 | 0 | IN   | GPIO. 1 | 1   | 18  |
 |  27 |   2 | GPIO. 2 |   IN | 0 | 13 || 14 |   |      | 0v      |     |     |
 |  22 |   3 | GPIO. 3 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO. 4 | 4   | 23  |
 |     |     |    3.3v |      |   | 17 || 18 | 0 | IN   | GPIO. 5 | 5   | 24  |
 |  10 |  12 |    MOSI | ALT0 | 0 | 19 || 20 |   |      | 0v      |     |     |
 |   9 |  13 |    MISO | ALT0 | 0 | 21 || 22 | 0 | IN   | GPIO. 6 | 6   | 25  |
 |  11 |  14 |    SCLK | ALT0 | 0 | 23 || 24 | 1 | OUT  | CE0     | 10  | 8   |
 |     |     |      0v |      |   | 25 || 26 | 1 | OUT  | CE1     | 11  | 7   |
 |   0 |  30 |   SDA.0 |   IN | 1 | 27 || 28 | 1 | IN   | SCL.0   | 31  | 1   |
 |   5 |  21 | GPIO.21 |   IN | 1 | 29 || 30 |   |      | 0v      |     |     |
 |   6 |  22 | GPIO.22 |   IN | 1 | 31 || 32 | 0 | IN   | GPIO.26 | 26  | 12  |
 |  13 |  23 | GPIO.23 |   IN | 0 | 33 || 34 |   |      | 0v      |     |     |
 |  19 |  24 | GPIO.24 |   IN | 0 | 35 || 36 | 0 | IN   | GPIO.27 | 27  | 16  |
 |  26 |  25 | GPIO.25 |   IN | 0 | 37 || 38 | 0 | IN   | GPIO.28 | 28  | 20  |
 |     |     |      0v |      |   | 39 || 40 | 0 | IN   | GPIO.29 | 29  | 21  |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+---Pi 3B--+---+------+---------+-----+-----+
```

Figura 18 Pines Raspberry Py
Fuente: Propia

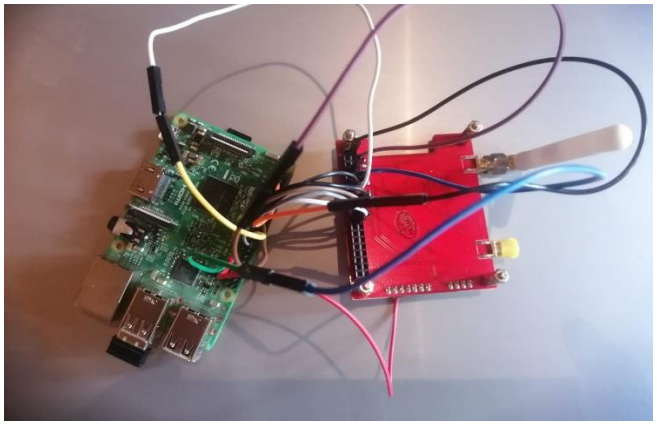


Figura 19 Cableado Pines LoRa HAT y Raspberry Pi
Fuente: Propia

Resultados:

- En la primera prueba se logra enviar un mensaje desde un dispositivo emisor y ser recibido en el receptor, pero debido a la complejidad del código no fue posible adaptarlo a las necesidades del proyecto.
- En la segunda prueba se presentan inconvenientes con la librería ya que arroja errores que no permiten cumplir con el objetivo de envío y recepción de mensajes.
- Se siguen diferentes tutoriales y guías que permitan la configuración de la librería teniendo como resultado errores durante la compilación, estos errores se evidencian como propios de la librería.
- Se pudo identificar que los errores son generados por el mapeo de los pines dado que el LoRa/GPS HAT se encuentra superpuesto sobre la Raspberry Pi y no coinciden con lo especificado en las librerías.

4.3 Iteración 2

Para cumplir con el propósito de la iteración 2 se hace necesario la implementación siguientes requerimientos:

Configuración del entorno objeto semántico en la Raspberry Pi:

Dado que este proyecto se basa en la arquitectura de objeto semántico se hace necesario la configuración de esta en un dispositivo con capacidades de procesamiento como una Raspberry Pi.

Dentro de las configuraciones realizadas se tiene el mapeo del objeto semántico con sus diferentes recursos (sensores y actuadores) donde se tiene la transmisión de datos mediante el

protocolo MQTT de la malla a la Raspberry Pi como un sensor, la transmisión datos de la Raspberry Pi a la malla mediante el protocolo MQTT como actuador, un monitoreo mediante protocolo MQTT de la malla como sensor y un pronóstico de las variables agroclimáticas mapeado como actuador, a continuación se describe la funcionalidad de cada uno de los recursos del OI.

- **Sensor MQTT transmisión datos malla a Raspberry Pi:** para este modelo de red planteado se propone una malla de nodos, los cuales presentan sensores conectaos al nodo encargados de proporcionar datos al OI mediante el protocolo de comunicación MQTT
- **Actuador MQTT transmisión datos Raspberry Pi a malla:** dado que en cada nodo de la malla se cuenta con un actuador que es el encargado de realizar el riego, se hace necesario la transmisión de datos desde el objeto semántico a un nodo especifico de la malla para ejecutar o detener el riego con la orden especificade
- **Sensor MQTT monitoreo de la malla:** es el encargado de verificar el estado de la malla mediate el protocolo de comunicación MQTT donde se suscribe al tópico especifico que retorna un mensaje con los nodos de la malla en funcionamiento.
- **Pronóstico:** el pronóstico es ejecutado a partir del sensor de monitoreo de la malla, el cual establece si hay un fallo de uno o varios nodos de la red, donde este hace uso de los datos recopilados en la BD del nodo especifico que falla y ejecutar el algoritmo de IA para pronosticar las variables agroclimáticas.

Los anteriores recursos fueron mapeados con el fin de inicializar el OI y crear los ejecutables para ser programados de acuerdo con su funcionalidad.

Resultados:

- Se logra inicializar el OI con sus ejecutables correspondientes sin ningún inconveniente.
- Se evidencia que durante las configuraciones de los ejecutables del OI estos no podrán ser usados por los Evento Condición Acción (ECA) que son los que permiten el funcionamiento e interacción entre los recursos del OI, dado que este al incluir un concepto de WSN como lo es la malla de sensores no se encuentra en la capacidad de administrar una cantidad N recursos (sensores y actuadores) debido que este presenta

un diseño para recibir datos o enviar datos de un sensor o actuador de cada tipo mapeado en el OI.

- Se pudo observar durante la configuración del objeto semántico, que este requiere de una ampliación dentro de su definición para el manejo de una estructura de N sensores, ya que estaría conectados a él una malla de sensores, lo que dificulta realizar procesos como es el caso del riego y predicción, puesto que está diseñado para la conexión directa de los sensores y actuadores.
- Se pudo evidenciar que la configuración inicial para inicializar el objeto pierde conocimiento semántico al no tener una caracterización de los recursos como los son los sensores de temperatura, humedad, luz solar y actuador.

Instalación de sensores a los nodos de la malla:

El modelo propuesto depende de la captura de variables agroclimáticas como temperatura (sensor LM35), radiación solar (sensor fotorresistencia LDR) y humedad del suelo (sensor YL69), con lo cual se tiene la configuración de cada uno de estos recursos en cada nodo de la malla, este proceso de configuración se realiza mediante el IDE Arduino, el cual es ejecutado desde un computador para la creación del código que permita la lectura tomada de los sensores, para la lectura de cada sensor se debe crear un circuito entre los pines del nodo y los pines del sensor teniendo en cuanta pines de alimentación (voltaje), pines entrada analógica del nodo, pines de salida analógica del sensor y pines a masa-tierra (GND) respectivamente lo que permita tener un nodo completo (ver Figura 20).

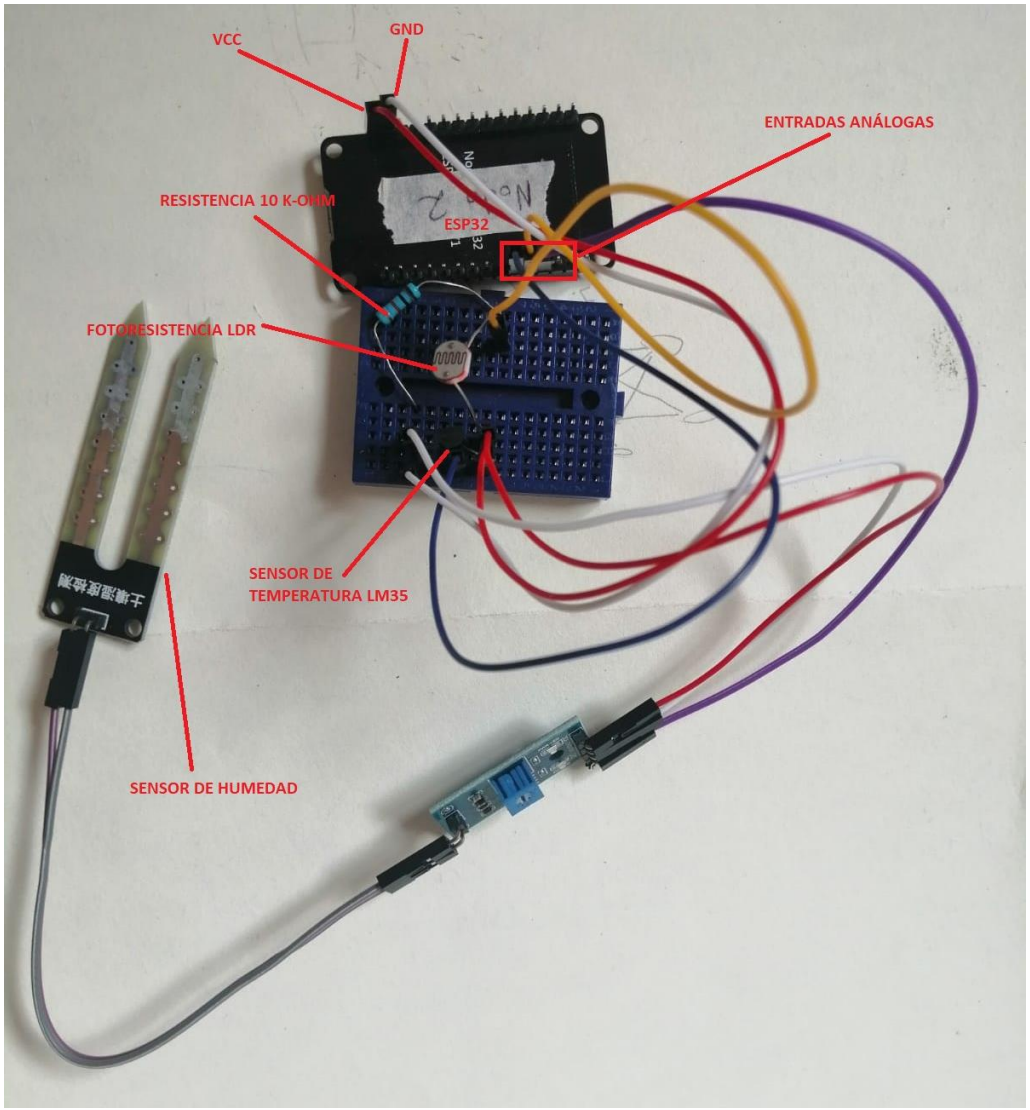


Figura 20 Montaje nodo de la malla
Fuente: Propia

Las conversiones de los datos leídos en los pines analógicos de la placa se representan en porcentaje de humedad del suelo para el sensor de humedad, la intensidad de luz (LUX) para el sensor de radiación y la temperatura en °C par el sensor de temperatura. Dicha conversión se obtiene a partir de las siguientes ecuaciones:

- **Sensor de temperatura**

$$milliVolt = adcVal * \frac{ADC\_VREF\_mV}{\text{ADC\_RESOLUTION}} \qquad \textbf{Ecuación 4}$$

$$tempC = \frac{milliVolt}{10} \qquad \textbf{Ecuación 5}$$

La ecuación Ecuación 4 representa la conversión a milivoltios, está dada por *adcVal* que representa el valor analógico capturado del pin, al cual se encuentra conectado el sensor de temperatura, donde puede variar de 0 – 4095 ya que presentan un conversor con resolución de 12 bits, *ADC_VREF_mV* representa la alimentación de 3300mV (3.3V), el *ADC_RESOLUTION* está dado por el valor máximo de resolución de la entrada analógica 4096.

La ecuación Ecuación 5 representa la temperatura en °C, la cual es obtenida de la división de voltaje (*milliVolt*) sobre 10 mV.

- **Sensor de humedad**

$$valHumedad = map(potValor, LecMinima, LecMaxima, 100,0) \qquad \textbf{Ecuación 6}$$

El cálculo de la humedad del suelo esta dado en porcentaje, el cual es obtenido de la relación entre el valor máximo de lectura de la entrada analógica que representa el 0% de humedad y el mínimo de la lectura que representa el 100% de humedad, se aclara que la resolución de los pines analógicos de la placa esp32 es de 12 bits y su salida varia de 0 – 4095 pero el sensor puede tomar valores mínimos o máximos dependiendo de la calibración. La ecuación Ecuación 6 muestra la función en Arduino que permite la conversión a porcentaje, donde *potValor* es la lectura analógica del pin que recibe la señal del sensor, *LecMinima* representa el valor mínimo de lectura obtenido y *LecMaxima* el valor máximo de lectura donde puede variar de acuerdo con la calibración realizada con el sensor en los ambientes, total mente seco y totalmente húmedo.

- **Sensor de radiación solar (intensidad de luz - LUX)**

$$ilum = (V * A * 10) / (B * Rc * (4096 - V)); \qquad \textbf{Ecuación 7}$$

El cálculo de la intensidad de luz (LUX) está dada por la ecuación Ecuación 7 donde *V* representa la lectura (0-4095) de la entrada analógica donde se encuentra conectado el sensor, al igual que los demás pines presenta una resolución de 12 bits, dentro de la ecuación se cuenta con las contantes *A* que representa la resistencia en oscuridad que es igual a 1000 KΩ, la resistencia a la luz, dada por B igual a 15 KΩ y una resistencia de calibración Rc igual a 10 KΩ.

Resultados:

- Durante la instalación de los sensores en el nodo (ESP32) se hizo necesario identificar los pines analógicos habilitados para recibir datos, de los cuales se pudo observar que los pines ADC1 Figura 21 son los pines habilitados para recibir señales analógicas, también se obtuvo que los pines ADC2 no podrán ser usados mientras el módulo Wifi de la placa esté en funcionamiento, lo que para nuestro caso es de tener en cuenta dado que estos nodos forman la malla mediante conexión Wifi.

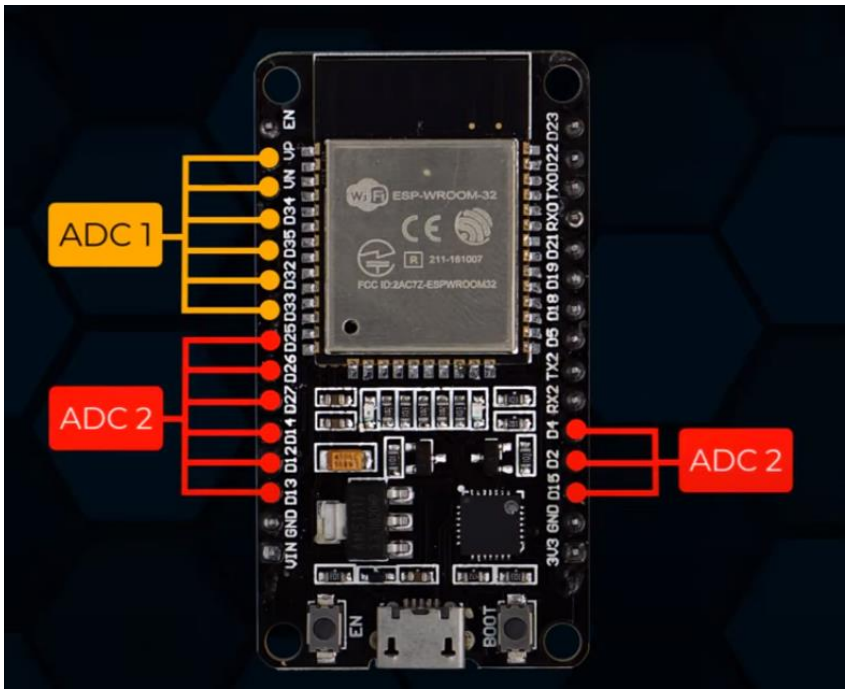


Figura 21 Pines analógicos habilitados
Fuente: http://tinyurl.com/4e89rstr

- Se pudo evidenciar una falla en el sensor de temperatura LM35, donde sus valores presentaban una variación significativamente alta llegando hasta 0°C y subiendo a valores próximos de la temperatura real, el error radica en la alimentación de voltaje del sensor, donde este requiere un voltaje mínimo de 4V y máximo de 20V y el nodo (ESP32) presenta una salida de voltaje de 3.3V lo que provoca las caídas de voltaje en las lecturas del sensor.

Almacenamiento de datos producidos por los nodos de la malla en una base de datos relacional instalada en la Raspberry Pi

El almacenamiento de los datos capturados por los sensores en una base de datos relacional en este caso en el sistema de gestión de bases de datos relacionales MySQL. La base datos (‘agroiot’) utilizada es la misma de la iteración anterior al igual que la tabla (‘info_nodo’), la cual contiene los campos para el identificador del nodo, las tres variables agroclimáticas monitoreadas (temperatura, humedad y radiación solar), y la hora de lectura en formato UTC. Una vez la base de datos está lista para recibir datos se empieza el proceso para lograr registrar la información recopilada por los nodos de la malla mediante los sensores descritos en el requerimiento anterior, el proceso es descrito a continuación:

- Se escribe el código para capturar los datos enviados por los sensores (humedad, temperatura, luz solar) conectados a la placa ESP32, para con ellos armar una cadena de tipo String con una estructura llave-valor igual a la estructura un objeto tipo JSON, el cual es enviado entre todos los nodos de la malla incluyendo el puente MQTT.

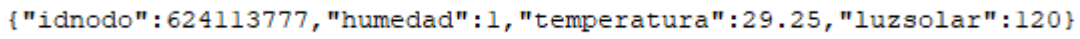


Figura 22 Cadena enviada desde el nodo
Fuente: Propia

- Una vez se recibe la cadena por parte del puente MQTT esta lo publica en el bróker MQTT utilizando el tópico “painlessMesh/from/#” Figura 23, logrando obtener la cadena en la Raspberry Pi al suscribirnos al tópico anterior.

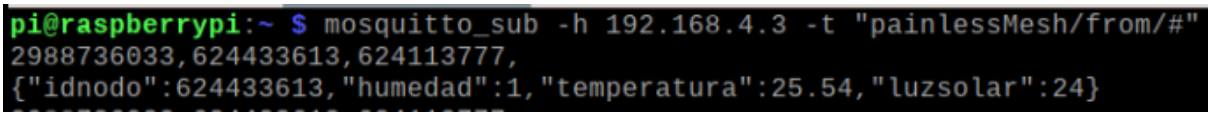


Figura 23 Suscribirse al bróker MQTT con el tópico indicado
Fuente: Propia

- Mediante la librería ‘Paho MQTT’ de Python se puede capturar la cadena recibida en el bróker MQTT Figura 24 para posteriormente transformarla en un objeto de tipo JSON mediante el método ‘json.loads()’ Figura 25 proporcionado por la librería JSON.

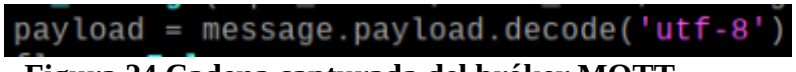


Figura 24 Cadena capturada del bróker MQTT
Fuente: Propia

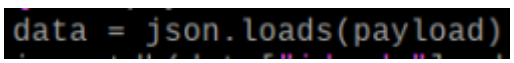


Figura 25 Transformar la cadena capturada en un objeto JSON
Fuente: Propia

- El objeto JSON es nombrado ‘data’, el cual cuenta con las llaves ‘idnodo’, ‘humedad’, ‘temperatura’ y ‘luzsolar’ Figura 26, para referirse a la información que guarda, y se puede acceder a dicha información con el nombre de la llave, facilitando el manejo de los datos para la construcción de la cadena SQL que será ejecutada posteriormente.

```
data["idnodo"], data["humedad"], data["temperatura"], data["luzsolar"]
```

Figura 26 Objeto JSON accedido
Fuente: Propia

- Desde el objeto ‘data’ ya se puede obtener la información para realizar la inserción de los datos en la base de datos, pero primero se debe construir la consulta para insertar los datos Figura 27, obteniendo como dato adicional el tiempo de lectura, refiriéndose al momento que los sensores leen la información, se debe aclarar que las placas ESP32 no pueden guardar información sobre la hora actual es por ello por lo que se utiliza la hora actual de la Raspberry Pi.

```
mycursor = mydb.cursor()
now = datetime.now()
tiempolectura = int(now.timestamp())
sql = "INSERT INTO info_nodo (idnodo, humedad, temperatura, luzsolar, tiempolectura) VALUES (%s, %s, %s, %s, %s)"
val = (idnodo, humedad, temperatura, luzsolar, tiempolectura)
mycursor.execute(sql, val)
mydb.commit()
mydb.close()
```

Figura 27 Insertar los datos del JSON en la base de datos
Fuente: Propia

- Al comprobar que la consulta tiene el resultado esperado, procede a ejecutar un script de Python que va a insertar los datos en la base de datos cada vez que llega la cadena al bróker MQTT, confirmando que los datos de la malla se estaban guardando Figura 28.

idnodo	humedad	temperatura	luzsolar	tiempolectura
624113777	1	0	171	1656715724
624433613	1	23	11	1656715725
624013073	1	31	35	1656715728
624113777	1	27	173	1656715729
624113777	1	24	172	1656715730
624433613	1	26	11	1656715730
624113777	1	29	172	1656715733
624433613	1	21	11	1656715733
624113777	1	0	175	1656715734
624013073	1	0	35	1656715735
624013073	1	11	35	1656715735
624113777	1	0	170	1656715736

Figura 28 Datos insertados en la base de datos
Fuente: Propia

Resultados:

- El tiempo de lectura debió ser insertado como tiempo en formato UTC tipo UNIX, debido que este formato de tiempo es el estándar para múltiples servicios, que luego podrían consumir dichos datos para ser graficados o publicados en alguna aplicación web o móvil.
- Los objetos tipo JSON brindan relativa facilidad para ser tratados dentro de Python debido que la información se puede buscar por palabras claves relacionadas con el valor contenido.
- La realización de este requerimiento no presentó inconvenientes debido que en la Iteración 1 se realizó un procedimiento similar.

Monitoreo de la malla:

El monitoreo de malla se define como la acción de conocer cuando alguno de los nodos ha sido desconectado, teniendo esto en cuenta, para realizar el monitoreo de la malla se hizo uso del puente MQTT contenido de la librería Painless Mesh, el cual contiene un método llamado ‘getNodeList’ con la funcionalidad de obtener la lista de nodos presentes en la malla, este método se activa cuando se realiza una publicación dirigida al tópico "painlessMesh/to/gateway" con el mensaje "getNodes" Figura 29, posterior a esto se obtiene un mensaje al estar suscrito al tópico “painlessMesh/from/gateway” Figura 30, el cual contiene una cadena compuesta por la lista número de identificador de los nodos activos en la malla, incluyendo el mismo puente. Los comandos se pueden ver en las siguientes figuras:

Figura 29 Publicar tópico y obtener la lista de nodos
Fuente: Propia

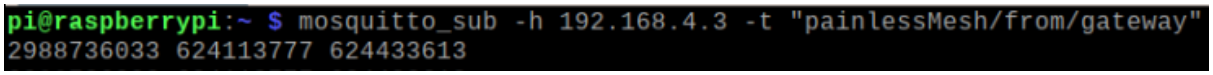


Figura 30 Suscribirse al tópico para obtener la lista de nodos
Fuente: Propia

A partir de esta funcionalidad proporcionada por el puente MQTT se decide probar lo que pasa cuando uno de los nodos es desconectado y volviendo a ejecutar el comando para publicar, encontrando que la lista se actualiza cuando un nodo es desconectado

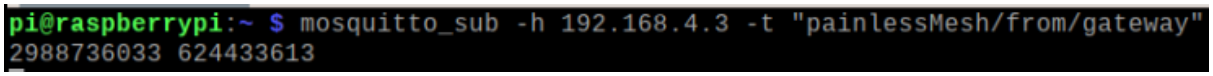


Figura 31 Lista de nodos actualizada
Fuente: Propia

Una vez se conoce esta capacidad del puente MQTT se pretende capturar la cadena con los identificadores de los nodos para convertirla dentro de Python a una lista, para lograr esto utilizaron dos métodos descritos a continuación:

- Como primera solución se procede a utilizar el bróker MQTT mediante el código implementado para la captura de mensajes y posterior inserción en la base de datos, en esta ocasión se pretendía publicar constantemente en el tópico especificado anteriormente con el mensaje “getNodes”, mediante la función de ciclo infinito proporcionado por la librería encargada del manejo de MQTT dentro de Python, para así mismo estar suscrito en el tópico para recibir la cadena, esta solución no fue posible debido que al iniciar la publicación infinita del mensaje el puente se desborda y deja de funcionar.
- Como segunda solución se procede a estudiar el código del puente MQTT para identificar en qué parte se realiza la publicación de la lista de nodos, una vez identificada se procede a utilizar este método para publicar la cadena con la lista de nodos en el mensaje que envía el puente de manera constante al bróker MQTT, para posteriormente utilizar el código para la captura de las cadenas en Python y diferenciar entre la cadena que contiene los datos de la malla y la lista de nodos.

Al obtener la cadena con la lista de nodos en Python, se convierte en una lista para compararla con la lista de nodos registrada en la base de datos, la cual se obtiene mediante la siguiente consulta:

```
"SELECT idnodo FROM info_nodo GROUP BY idnodo"
```

Figura 32 Consulta SQL para obtener los nodos registrados
Fuente: Propia

Una vez se logran comparar las cadenas se obtiene el id del nodo que dejó de funcionar y tomar las acciones necesarias.

Resultados:

- Se logró evidenciar que el puente MQTT no está diseñado para recibir una cantidad elevada de mensajes debido que cuando se intentó, dejó de funcionar perjudicando la funcionalidad para la captura de información desde la malla y en consecuencia la inserción en la base de datos.
- Fue posible reutilizar el tópico para la captura de datos de la malla mediante la modificación del puente MQTT para publicar la lista de nodos en el mismo, reduciendo el trabajo del puente y logrando que no se afectara ninguna otra funcionalidad.

Implementación algoritmo inteligente:

El algoritmo inteligente elegido hace parte del aprendizaje automático supervisado y es la regresión lineal. Para implementar la regresión lineal se usó el lenguaje de programación Python (v3.9) debido que tiene diversas librerías dirigidas a los algoritmos de inteligencia computacional, en este caso se utilizó ‘Scikit-learn’, dicha librería cuenta con los métodos necesarios para la construcción y entrenamiento del modelo de regresión lineal. Para el entrenamiento del modelo mencionado se debe contar con un conjunto de datos considerable donde se etiqueten las variables dependientes e independientes, para esto se utiliza la librería ‘Pandas’, la cual contiene métodos para obtener información de fácil manejo y así realizar el entrenamiento del modelo y la posterior predicción. Una vez se eligen las herramientas, se procede a detallar los pasos realizados para ejecutar la regresión lineal y obtener el pronóstico de datos:

- Para iniciar la ejecución del algoritmo inteligente, primero la función de monitoreo debe detectar que uno de los nodos de la malla ha dejado de funcionar y posteriormente dicha función se encargar de obtener el identificador del nodo.
- Una vez se ha identificado el nodo que ha dejado de funcionar, se construye una consulta SQL (Figura 33) incluyendo el identificador del nodo, para obtener la información recolectada por ese nodo, la consulta es ejecutada en Python utilizando el método 'read_sql' (Figura 34) incluido en la librería 'Pandas', que recibe como parámetros la consulta mencionada y la información de conexión a la base de datos, el método retorna una estructura de datos bidimensional denominada 'dataframe', este 'dataframe' también es llamado 'dataset' (Figura 35) el cual se considera como el conjunto de datos para el entrenamiento del modelo de regresión lineal, compuesto por filas y columnas, donde las filas contienen cada una de las lecturas de los sensores enviadas por el nodo en cuestión y las columnas son las variables agroclimáticas: humedad, temperatura y luz solar monitoreadas por el nodo que ha dejado de funcionar.

```
"SELECT humedad, temperatura, luzsolar FROM info_nodo WHERE idnodo = %s"%str(idnodo)
```

Figura 33 Consultar la información de un nodo específico
Fuente: Propia

```
dataframe = pd.read_sql(sql, mydb)
```

Figura 34 Función para guardar la información consultada
Fuente: Propia

idnodo	humedad	temperatura	luzsolar	tiempolectura
624113777	48	5	174	1656716026
624113777	47	24	173	1656716028
624113777	47	10	173	1656716032
624113777	47	13	167	1656716037
624113777	47	6	170	1656716039
624113777	46	26	171	1656716044
624113777	48	4	171	1656716049
624113777	47	19	170	1656716049
624113777	47	24	170	1656716051
624113777	47	6	168	1656716055
624113777	47	24	169	1656716057
624113777	47	5	170	1656716059
624113777	47	7	170	1656716061
624113777	46	26	169	1656716064
624113777	47	24	172	1656716069

Figura 35 Dataset
Fuente: Propia

- Al tener listo el ‘dataset’ para realizar el entrenamiento del modelo de regresión lineal, se definen tanto las variables independientes como las dependientes, para este caso las variables independientes son la radiación solar (‘luzsolar’) y la temperatura ambiente (‘temperatura’) identificadas con la letra ‘X’, por otra parte, la variable dependiente es la humedad del suelo (‘humedad’) identificada por la letra ‘y’, esta última es la variable que se va a predecir mediante al algoritmo.

```
X = dataframe['luzsolar, temperatura'].values.reshape(-1,1)
y = dataframe['humedad'].values.reshape(-1,1)
```

Figura 36 Definición de variables independientes y dependientes
Fuente: Propia

- Discriminadas las variables independientes y dependientes, se procede a dividir estos datos en dos conjuntos, el primero para datos de entrenamiento y el segundo conjunto para datos de prueba, utilizando el método ‘train_test_split’ Figura 37, incluido en la librería ‘Scikit-learn’, el método recibe los siguientes parámetros, las variables independientes y la variable dependiente definidas en el paso anterior y el tamaño del conjunto de los datos de prueba (‘test_size’) que para este caso es del 10% del total de los datos teniendo en cuenta que el dataset usado cuenta con aproximadamente 1050 datos teniendo como datos de prueba aproximadamente 105.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1)
```

Figura 37 División datos de entrenamiento y de prueba
Fuente: Propia

Los datos de prueba son comparados con los datos obtenidos mediante la regresión lineal, los cuales son evaluados mediante el coeficiente de determinación (R^2).

- Definido el conjunto de datos de entrenamiento se procede a entrenar el modelo de regresión lineal, para ello se instancia el modelo mediante el método ‘LinearRegression’ (Figura 38) incluido en la librería ‘Scikit-learn’ para posteriormente utilizar el método ‘fit’ (Figura 39) que recibe dos variables como parámetros de entrada, las primeras las variables independientes y la segunda la variable dependiente.

```
lm = linear_model.LinearRegression()
```

Figura 38 Instanciación del modelo de regresión lineal
Fuente: Propia

```
lm.fit(X_train, y_train)
```

Figura 39 Entrenamiento del modelo de regresión lineal
Fuente: Propia

- El modelo de regresión lineal está preparado para realizar el pronóstico de la humedad del suelo (variable dependiente) a partir de las variables de humedad del suelo y temperatura, para realizar el pronóstico se utiliza el método 'predict' que recibe como parámetro de entrada el conjunto de datos de prueba de la variable dependiente.

```
y_pred = lm.predict(X_test)
```

Figura 40 Predicción de los datos mediante regresión lineal
Fuente: Propia

- Posterior a la ejecución del código y teniendo en cuenta que el tamaño del conjunto de datos de prueba es de 105 datos, se pronostican los siguientes datos

```
 89 35.65681864554652
 90 38.91575886751064
 91 44.64674646560308
 92 27.909006993128557
 93 43.13108218507172
 94 29.67890664830166
 95 34.0854699457047
 96 33.526188760770296
 97 30.913711840450933
 98 32.34463099610171
 99 47.6780750266658
100 32.00808748840916
101 45.404578605868764
102 34.02734853456447
103 35.267027710373284
104 30.296309244376296
105 32.20420145191067
```

Figura 41 Datos pronósticas mediante el modelo de regresión lineal
Fuente: Propia

- Al obtener los datos pronosticados se calcula el coeficiente de determinación (R^2) con el que comprobamos la eficiencia del modelo de regresión lineal, esto se logró mediante la utilización del método 'r2_score' incluido en la librería 'Scikit-learn', el método recibe como parámetros los datos pronósticados ('y_pred') y el conjunto de datos de prueba ('y_test').

```
rcuad = r2_score(y_test, y_pred)
```

Figura 42 Función para obtener el R^2
Fuente: Propia

Resultado:

- El lenguaje de programación Python ofrece la posibilidad de acceder a librerías externas que ofrecen diversidad de soluciones tanto para la implementación de técnicas de inteligencia artificial como para al manejo de grandes conjuntos de datos, empezando por la facilidad para obtener datos de distintas fuentes como bases de datos SQL como de archivos planos con extensión CSV.
- El coeficiente de determinación (R^2) fue de 0.01%, lo cual indica que no hay una correlación entre la variable dependiente y las variables independientes, este resultado se entiende debido que los sensores no están totalmente calibrados y los datos capturados se obtuvieron en un ambiente de laboratorio, además, el sensor de temperatura presenta lecturas desfazadas, donde se logra evidenciar que son causadas por la fuente de alimentación, la cual es de 3.3 voltios y este requiere como mínimo una de 4 voltios.
- El resultado de la consulta SQL a la base de datos contiene las etiquetas de cada una de las variables (humedad, temperatura, luz solar) que ahí se guardan, y en el DataFrame se puede acceder con el nombre de la variable, facilitando la asignación de una variable independiente como independiente.

4.4 Iteración 3

La finalidad de la iteración 3 es presentar una realimentación sobre los resultados obtenidos posterior al desarrollo de cada uno de los requerimientos, para con este conocimiento proponer alternativas o soluciones que otorguen mejoras al modelo propuesto. donde a partir de las iteraciones anteriores al tener un desarrollo con dispositivos físicos se lograron evidenciar problemas técnicos relacionados con dispositivos propuestos para usar el modelo, también inconvenientes con el uso de la arquitectura de interacción semántica de objetos inteligentes propuesta por Niño Zambrano [21], a lo cual se permite tener un aporte sobre está a partir del modelo de red propuesto.

Configuración del módulo LoRa/GPS HAT instalado en la Raspberry Pi para el intercambio de mensajes

La falta de conocimiento técnico sobre el módulo de expansión para Raspberry Pi llamado LoRa/GPS HAT del fabricante Dragino, trajo diferentes inconvenientes técnicos, los cuales fueron mencionados dentro de los resultados obtenidos en el desarrollo de este requerimiento. El principal inconveniente surgió durante el proceso de configuración de las librerías, las cuales

generaban errores por el mapeo incorrecto de los pines del LoRa/GPS HAT con los pines GPIO de la Raspberry Pi, esto a raíz de que el primero está superpuesto sobre la Raspberry Pi, impidiendo identificar el mapeo correcto de los pines correspondientes al chip LoRa con los pines GPIO, para solucionar esto se propone utilizar el chip LoRa como el que viene preinstalado en el LoRa/GPS HAT, dicho chip puede ser de las referencias RMF96 o SX1276, donde al tener el chip por separado este permitirá un manejo directo de sus pines para realizar las conexiones con los pines GPIO de la Raspberry Pi, tal como lo sugieren las librerías dedicadas a estos chips y así realizar pruebas que permitan el envío y recepción de datos entre dos Raspberry Pi mediante el protocolo LoRa.

Configuración del entorno objeto semántico en la Raspberry Pi:

Para la configuración del escenario de objeto semántico se requiere de una serie de pasos, donde inicialmente se debe configurar el entorno de tal manera que el OI tenga pleno conocimiento de sus recursos (sensores y actuadores) para inicializar el OI, el conocimiento de sus recursos por parte del OI se hace necesarios para que este pueda interactuar con ellos y ejecutar acciones que se tengan como objetivo.

Debido a lo evidenciado en el proceso de configuración del entorno de objeto semántico a la Raspberry pi con el escenario propuesto de tener una malla con N nodos de sensores y actuadores, se hace necesario y se presenta como propuesta la reestructuración del mecanismo inicializador del objeto, perteneciente a la capa de aplicación, de tal manera que el OI tenga la capacidad de reconocer los recursos provenientes de cada nodo de la malla, en este caso los sensores de temperatura, humedad, radiación solar y actuador de riego.

Adicional a esto se requiere la reestructuración del mecanismo de interacción denominado ECA (Evento Condición Acción), los cuales permiten la interacción entre el OI y sus recursos o la interacción con los recursos de otros objetos inteligentes identificados en su entorno, con lo anterior, se propone una interfaz de usuario que permita crear la interacción del objeto con sus recursos. La interfaz propuesta es presentada y explicada a continuación:

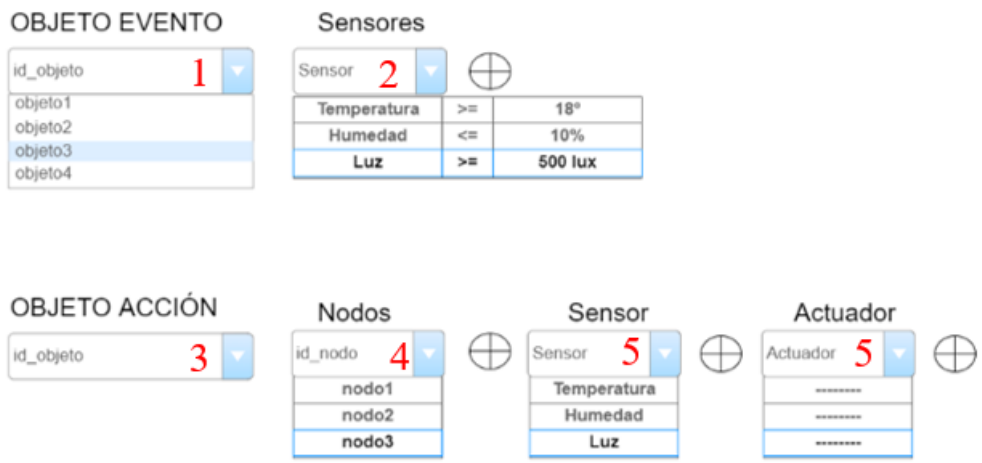


Figura 43 Interfaz propuesta crear ECA Fuente: Propia

1. Seleccionar el objeto sobre el que se da el evento
2. Agregar los recursos (sensor de temperatura, humedad y radiación solar), condicionales (<, =, >, <=,> …) y umbrales para cada recurso sobre los que se genera la condición
3. Seleccionar el objeto sobre el que se da la acción, inicialmente corresponde al mismo objeto sobre el que se da el evento y condición.
4. Agregar el nodo o los nodos sobre los que se genera la acción, entre estos nodos debe estar el nodo correspondiente a los recursos que se dio la condición.
5. Agregar los sensores y/o actuador sobre los cuales se desea que se genera la acción.

La creación de esta interfaz requiere para su funcionamiento la modificación de las entidades creadoras del ECA identificadas en la capa de interacción de la arquitectura de interacción semántica de objetos inteligentes, las cuales son listadas a continuación:

- ECA.py
- EventoECA.py
- AccionECA.py
- PoblarECA.py
- EditarECA.py

También se deben tener en cuanta otros componentes importantes involucrados en la reestructuración del mecanismo de creación ECA como lo son:

- ServidorWebIndependiente.py
- Ontologia.py

Razones de no hacer un despliegue en campo

El realizar la prueba de concepto nos permitió observar las diferentes dificultades para un despliegue en campo donde se pueden encontrar las siguientes.

- Dado que, durante las diferentes pruebas realizadas en laboratorio de los dispositivos, estos se conectaban directamente mediante USB a un computador portátil, el cual funcionaba como fuente de alimentación para los diferentes dispositivos, esto impide tener un despliegue en campo dado que se requeriría de una fuente de alimentación para su funcionamiento en cada dispositivo.
- Otra limitante para el despliegue en campo está relacionada con la protección de los dispositivos, dado que estos no están preparados para estar a la intemperie, siendo necesario diseñar cajas resistentes en las que se puedan encapsular los diferentes dispositivos como lo son las Rasberry Pi y los nodos de sensores que forman la malla.
- El realizar un despliegue en el campo requerirá una serie de pruebas extra, las cuales pueden consistir en buscar la ubicación ideal de los nodos de la malla, distancias adecuadas entre nodos, instalación de fuentes de alimentación, entre otros aspectos que pueden presentar una alta demanda de tiempo para su realización y no estar dentro del enfoque del proyecto.

Conclusiones del desarrollo

- Al realizar un despliegue físico se logró evidenciar ciertos inconvenientes técnicos con los dispositivos a usar en el modelo, debido a esto no se lograron cumplir los siguientes requerimientos:
    - Envió de datos a servidor web IoT.
    - Graficado de los datos almacenados en el base de datos.
- Al tener una topología de malla de nodos de sensores esta permite una escalabilidad alta donde se podrán agregar nodos logrando tener áreas de monitoreo más extensas si las necesidades lo requieren.
- En la prueba de concepto se logró evidenciar que la arquitectura de interacción semántica de objetos inteligentes requiere una restructuración, donde se presenta una propuesta como aporte de este proyecto, la cual identifica los elementos para tener en cuenta para su restructuración.
- El uso de la librería Painless Mesh facilita la agregación de nodos a la malla sin mayor esfuerzo, donde solo se necesita cargar el código fuente de la malla en el nodo a desplegar y este será agregado de forma automática mientras esté al alcance de alguno de los nodos de la malla ya existente.

- El consumo energético es una de las principales falencias del modelo propuesto más específicamente en la capa de adquisición de datos, puesto que para la creación de la malla la librería propuesta se basa en protocolo Wifi, el cual realiza un alto consumo de energía.
- Dentro del desarrollo realizado se pudo evidenciar que la cobertura o alcance entre nodos de la malla se puede ver influenciada por la interferencia u obstrucción de otros elementos entre ellos.
- El desarrollo de las funcionalidades dentro del lenguaje de programación Python fue eficiente debido que el lenguaje tiene una sintaxis simple y de fácil entendimiento, además de ofrecer una amplia variedad de librerías para el manejo de distintos tipos de tecnologías como implementación de algoritmos de inteligencia artificial o la comunicación entre dispositivos mediante el protocolo MQTT.
- Se pudo observar a raíz de los diferentes procedimientos realizados en el desarrollo de la prueba de concepto que la Raspberry Pi tiene altas capacidades para ofrecer distintos tipos de servicios como motor de bases de datos, servidor web local, entre otros, siendo parte fundamental de los desarrollos orientados a IoT.

Capítulo 5 Conclusiones y trabajo futuro

5.1 Conclusiones en cuanto a productos obtenidos:

- Se propone un Modelo de red de objetos inteligentes inalámbricos, siguiendo la metodología para la implementación de redes de sensores inalámbricos basado en los estándares internacionales. El modelo establece mecanismos para el monitoreo constante de la red, permitiendo realizar un pronóstico de los datos al nodo que deja de funcionar, mitigando la pérdida de información de la red.
- El modelo propuesto está en la capacidad de ejecutar un algoritmo de aprendizaje automático que permita mitigar la perdida de datos por fallos en la red.
- Se ha creado un marco de referencia para la implementación de redes de sensores inalámbricos en agricultura de precisión, identificando los aportes y brechas que existen.
- Se realizaron distintos aportes en la arquitectura Interacción Semántico de la WoT principalmente en la Capa de Objeto donde ahora se propone una topología determinada en vez de sensores conectados directamente al OI, obligando la adición de mecanismos para la obtención de información de la topología.
- Se implementó una red de objetos inteligentes inalámbricos con dispositivos reales en un entorno simulado aplicando las capas del modelo propuesto, iniciando por la configuración de una malla de ESP32 que leen datos de tres sensores diferentes que luego es enviada a la Raspberry Pi para su almacenamiento y procesamiento.
- El realizar una prueba de concepto mediante el despliegue de objetos físicos nos permitió determinar limitaciones y falencias o restricciones en cuanto funcionalidad de los dispositivos a usar en el modelo.
- El objetivo específico número tres se cumplió parcialmente, dado que mediante la prueba de concepto se logró implementar un algoritmo de inteligencia (regresión lineal), que puede mitigar la perdida de datos mediante el pronóstico y posterior suplantación de los datos perdidos, pero por dificultades técnicas obtenidas a partir del despliegue físico de los dispositivos, no se logró la implementación total del modelo propuesto, en consecuencia no fue posible evidenciar el uso optimo del agua dentro del cultivo, dado que se requeriría un despliegue en un ambiente real del modelo totalmente desarrollado.

5.2 Conclusiones en cuanto a la investigación:

- Con respecto a la posibilidad de incluir el concepto de WSN como lo es la topología en malla de nodo de sensores a la arquitectura de OI se requiere un trabajo de restructuración del mecanismo de creación de ECA, con el fin de que acepte datos de una malla de N nodos de sensores y no solo los datos de un sensor o actuador. Este trabajo sobrepasa las restricciones de tiempo de la presente investigación y se deja como trabajo futuro, aunque se deja en el modelo información de cuáles serían los elementos para tener en cuenta para su implementación.
- En la actual investigación, aunque fue posible reutilizar la arquitectura de interacción semántica propuesta por el grupo GTI para comunicar y ejecutar acciones en el mismo objeto, no fue posible adecuarla a la gestión de interacción entre objetos de la misma malla por la razón establecida en la conclusión anterior. Sin embargo, se entiende que el problema principal es técnico y no de posibilidad, por lo que se puede inferir que el enfoque podría mejorar los resultados de gestión de eventos de pérdida de datos en los despliegues de agricultura de precisión que utilicen el modelo propuesto.

5.3 Trabajo futuro

- La arquitectura de interacción semántica requiere una ampliación de los mecanismos inicializadores de objeto y mecanismos de creación de ECA, que permitan el manejo datos provenientes de una malla de N nodos de sensores, dado que su arquitectura está diseñada para manejar sensores y actuadores conectados directamente al OI y no una malla de nodos de sensores como se planteó en el modelo propuesto.
- Después de las pruebas realizadas se encontró que el R^2 (coeficiente de determinación) de la regresión lineal es muy bajo, mostrando la necesidad de realizar una configuración y recolección de datos adecuada de los sensores y del algoritmo mismo para mejorar la precisión de la predicción y que sea más confiable.
- Realizar un despliegue real donde se permitan tener indicadores de rendimiento, retardos y relación de paquetes de la red propuesta.

BIBLIOGRAFÍA

[1] C. A. Madramootoo and H. Fyles, “Irrigation in the context of today’s global food crisis,” *Irrig. Drain.*, vol. 59, no. 1, pp. 40–52, 2010, doi: 10.1002/ird.555.

[2] A. y M. A. de la E. E. de A. D. (CSIC) y C. de I. y T. A. del G. de A. Grupo de Riegos, “El uso del agua en la agricultura de regadío y la investigación pública,” 2013.

[3] Óscar Granados, “Un mundo loco por el café,” Madrid, 2018.

[4] Observatorio de Complejidad Económica, “Café, incluso tostado o descafeinado; cáscara y cascarilla de café; sucedáneos del café que contengan café en cualquier proporción (HS92: 0901) Comercio producto, exportadores y importadores,” *Oec.World*, 2017. https://oec.world/en/profile/country/col#trade-products

[5] A. K. Chapagain and A. Y. Hoekstra, “The water needed to have the Dutch drink coffee Value of Water,” 2003.

[6] F. Karim, F. Karim, and A. Frihida, “Monitoring system using web of things in precision agriculture,” in *Procedia Computer Science*, 2017, vol. 110, pp. 402–409. doi: 10.1016/j.procs.2017.06.083.

[7] C. Kamienski *et al.*, “Smart water management platform: IoT-based precision irrigation for agriculture,” *Sensors (Switzerland)*, vol. 19, no. 2, 2019, doi: 10.3390/s19020276.

[8] B. Keswani, “Adapting weather conditions based IoT enabled smart irrigation technique in precision agriculture mechanisms,” *Neural Comput. Appl.*, vol. 31, pp. 277–292, 2019, doi: 10.1007/s00521-018-3737-1.

[9] P. Sanjeevi, “Precision agriculture and farming using Internet of Things based on wireless sensor network,” *Trans. Emerg. Telecommun. Technol.*, 2020, doi: 10.1002/ett.3978.

[10] D. Thakur, Y. Kumar, A. Kumar, and P. K. Singh, “Applicability of Wireless Sensor Networks in Precision Agriculture: A Review,” *Wirel. Pers. Commun.*, vol. 107, no. 1, pp. 471–512, Jul. 2019, doi: 10.1007/s11277-019-06285-2.

[11] C. M. Angelopoulos, G. Filios, S. Nikoletseas, and T. P. Raptis, “Keeping data at the edge of smart irrigation networks: A case study in strawberry greenhouses,” *Comput. Networks*, vol. 167, 2020, doi: 10.1016/j.comnet.2019.107039.

[12] A. Ruiz de Velasco, *El agua en la agricultura /*. México : Secretaria de Fomento, 1912. doi: 10.5962/bhl.title.138862.

[13] V. M. Juan Núñez, R. Faruk Fonthal, and L. M. Yasmín Quezada, “Design and Implementation of WSN and IoT for Precision Agriculture in Tomato Crops,” *2018 IEEE ANDESCON, ANDESCON 2018 - Conf. Proc.*, 2018, doi: 10.1109/ANDESCON.2018.8564674.

[14] A. Cama-Pinto, F. Gil-Montoya, J. Gómez-López, A. García-Cruz, and F. Manzano-Agugliaro, “Sistema inalámbrico de monitorización para cultivos en invernadero,” *DYNA*, vol. 81, no. 184, pp. 164–170, 2014, doi: 10.15446/dyna.v81n184.37034.

[15] C. A. L. Garzón and O. J. R. Riveros, “Temperature, humidity and luminescence monitoring system using Wireless Sensor Networks (WSN) in flowers growing,” 2010. doi: 10.1109/ANDESCON.2010.5633140.

[16] J. G. Caicedo-Ortiz *et al.*, “Monitoring system for agronomic variables based in WSN technology on cassava crops,” *Comput. Electron. Agric.*, vol. 145, pp. 275–281, 2018, doi: https://doi.org/10.1016/j.compag.2018.01.004.

[17] M. Piamonte, M. Huerta, R. Clotet, J. Padilla, T. Vargas, and D. Rivas, *WSN prototype for African oil palm bud rot monitoring*, vol. 687. 2018. doi: 10.1007/978-3-319-

70187-5_13.
[18] K. N. Qureshi, M. U. Bashir, J. Lloret, and A. Leon, “Optimized Cluster-Based Dynamic Energy-Aware Routing Protocol for Wireless Sensor Networks in Agriculture Precision,” *J. Sensors*, vol. 2020, 2020, doi: 10.1155/2020/9040395.
[19] G. Kortuem, F. Kawsar, V. Sundramoorthy, and D. Fitton, “Smart objects as building blocks for the internet of things,” *IEEE Internet Comput.*, vol. 14, no. 1, pp. 44–51, 2010, doi: 10.1109/MIC.2009.143.
[20] S. Guerrero-Narváez, M. ángel Niño-Zambrano, D. J. Riobamba-Calvache, and G. A. Ramírez-González, “Test bed of semantic interaction of smart objects in the web of things,” *Futur. Internet*, vol. 10, no. 5, 2018, doi: 10.3390/fi10050042.
[21] M. A. Niño-Zambrano, “Interacción Semántica de Objetos en la Web de las Cosas,” *XI Coloquio Doctoral de CLADEA*. pp. 1–15, 2013. [Online]. Available: http://www.cladea.org/home/index.php?option=com_phocadownload&view=category&id=157&Itemid=570
[22] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, “Systematic mapping studies in software engineering,” *12th Int. Conf. Eval. Assess. Softw. Eng. EASE 2008*, no. February 2015, 2008, doi: 10.14236/ewic/ease2008.8.
[23] R. A. Jaramillo, “La agroclimatología del cafeto,” *Clima andino y café en Colombia*. pp. 149–157, 2005.
[24] Coremain, “Cloud Computing, Edge Computing, Fog Computing ¿En qué se diferencian?,” 2018. https://www.coremain.com/diferencias-cloud-computing-edge-computing-fog-computing/ (accessed Jul. 05, 2022).
[25] Q. Qi and F. Tao, “A Smart Manufacturing Service System Based on Edge Computing, Fog Computing, and Cloud Computing,” *IEEE Access*, vol. 7, pp. 86769–86777, 2019, doi: 10.1109/ACCESS.2019.2923610.
[26] “Y.2221 : Requirements for support of ubiquitous sensor network (USN) applications and services in the NGN environment.” https://www.itu.int/rec/T-REC-Y.2221/en (accessed Mar. 03, 2022).
[27] J. A. Stankovic, “Wireless Sensor Networks,” in *Ultra-Low Power Wireless Technologies for Sensor Networks*, Boston, MA: Springer US, 2007, pp. 1–12. doi: 10.1007/978-0-387-49313-8_1.
[28] S. Román, M. Cantillo, J. V. Capella, and H. Valencia, “PFC ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA,” 2010.
[29] A. Triantafyllou, D. C. Tsouros, P. Sarigiannidis, and S. Bibi, “An architecture model for smart farming,” *Proc. - 15th Annu. Int. Conf. Distrib. Comput. Sens. Syst. DCOSS 2019*, pp. 385–392, May 2019, doi: 10.1109/DCOSS.2019.00081.
[30] J. De *et al.*, “LoRaWAN-A Low Power WAN Protocol for Internet of Things: a Review and Opportunities Transient stability enhancement View project WSNlife: prolonging the lifetime of mobile wireless sensor networks View project LoRaWAN-A Low Power WAN Protocol for Internet of Things: a Review and Opportunities,” 2017. [Online]. Available: https://www.researchgate.net/publication/318866065
[31] J. S. Rueda and J. M. Talavera Portocarrero, “Similitudes y diferencias entre Redes de Sensores Inalámbricas e Internet de las Cosas: Hacia una postura clarificadora,” *Rev. Colomb. Comput.*, vol. 18, no. 2, pp. 58–74, 2017, doi: 10.29375/25392115.3218.
[32] S. Saigua Carvajal, M. Villafuerte Haro, D. Ávila Pesantez, and A. Arellano, “Evaluación de las topologías físicas de redes WSN para la medición de variables ambientales,” *Rev. Científica y Tecnológica UPSE*, vol. 3, no. 1, pp. 159–165, Dec. 2015, doi: 10.26423/RCTU.V3I1.84.
[33] FAO - Food and Agriculture Organization of the United Nations, “Capitulo 1 - Modelos Y Su Uso,” *Análisis de sistemas de producción animal. v. 2: las herramientas*

básicas. (Estudio FAO Producción y Sanidad Animal 140/2). 1997. Accessed: Feb. 17, 2022. [Online]. Available: https://www.fao.org/3/W7452S/w7452s01.htm#1.3 tipos de modelos

[34] Y. Syafarinda, F. Akhadin, Z. E. Fitri, Yogiswara, B. Widiawanl, and E. Rosdiana, "The Precision Agriculture Based on Wireless Sensor Network with MQTT Protocol," *IOP Conference Series: Earth and Environmental Science*, vol. 207, no. 1. 2018. doi: 10.1088/1755-1315/207/1/012059.

[35] K. Thorp, "Precision Agriculture," in *Encyclopedia of Earth Sciences Series*, vol. 10, no. 5, 2014, pp. 515–517. doi: 10.1007/978-0-387-36699-9_132.

[36] D. Thakur, "Applicability of Wireless Sensor Networks in Precision Agriculture: A Review," *Wireless Personal Communications*. 2019. doi: 10.1007/s11277-019-06285-2.

[37] M. S. Mekala and P. Viswanathan, "A Survey: Smart agriculture IoT with cloud computing," in *2017 International Conference on Microelectronic Devices, Circuits and Systems, ICMDCS 2017*, 2017, vol. 2017-Janua, pp. 1–7. doi: 10.1109/ICMDCS.2017.8211551.

[38] O. Elijah, T. A. Rahman, I. Orikumhi, C. Y. Leow, and M. N. Hindia, "An Overview of Internet of Things (IoT) and Data Analytics in Agriculture: Benefits and Challenges," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3758–3773, 2018, doi: 10.1109/JIOT.2018.2844296.

[39] A. N. Harun, M. R. M. Kassim, I. Mat, and S. S. Ramli, "Precision irrigation using Wireless Sensor Network," *2015 International Conference on Smart Sensors and Application, ICSSA 2015*. pp. 71–75, 2015. doi: 10.1109/ICSSA.2015.7322513.

[40] G. Kortuem, "Smart objects as building blocks for the internet of things," *IEEE Internet Comput.*, vol. 14, no. 1, pp. 44–51, 2010, doi: 10.1109/MIC.2009.143.

[41] Y. Mekonnen, S. Namuduri, L. Burton, A. Sarwat, and S. Bhansali, "Review—Machine Learning Techniques in Wireless Sensor Network Based Precision Agriculture," *J. Electrochem. Soc.*, vol. 167, no. 3, p. 037522, Jan. 2020, doi: 10.1149/2.0222003jes.

[42] D. R. Vincent, N. Deepa, D. Elavarasan, K. Srinivasan, S. H. Chauhdary, and C. Iwendi, "Sensors driven ai-based agriculture recommendation model for assessing land suitability," *Sensors (Switzerland)*, vol. 19, no. 17, Sep. 2019, doi: 10.3390/s19173667.

[43] W. Li *et al.*, "Review of Sensor Network-Based Irrigation Systems Using IoT and Remote Sensing," *Adv. Meteorol.*, vol. 2020, pp. 1–14, Sep. 2020, doi: 10.1155/2020/8396164.

[44] G. Codeluppi, A. Cilfone, L. Davoli, and G. Ferrari, "AI at the Edge: A Smart Gateway for Greenhouse Air Temperature Forecasting," *2020 IEEE Int. Work. Metrol. Agric. For. MetroAgriFor 2020 - Proc.*, pp. 348–353, 2020, doi: 10.1109/MetroAgriFor50201.2020.9277553.

[45] S. Trilles, J. Torres-Sospedra, Ó. Belmonte, F. J. Zarazaga-Soria, A. González-Pérez, and J. Huerta, "Development of an open sensorized platform in a smart agriculture context: A vineyard support system for monitoring mildew disease," *Sustain. Comput. Informatics Syst.*, vol. 28, p. 100309, Dec. 2020, doi: 10.1016/j.suscom.2019.01.011.

[46] G. Codeluppi, A. Cilfone, L. Davoli, and G. Ferrari, "LoraFarM: A LoRaWAN-based smart farming modular IoT architecture," *Sensors (Switzerland)*, vol. 20, no. 7, Apr. 2020, doi: 10.3390/s20072028.

[47] U. Shafi, R. Mumtaz, J. García-Nieto, S. A. Hassan, S. A. R. Zaidi, and N. Iqbal, "Precision agriculture techniques and practices: From considerations to applications," *Sensors (Switzerland)*, vol. 19, no. 17. MDPI AG, Sep. 01, 2019. doi: 10.3390/s19173796.

[48] M. Mizan Maha, S. Bhuiyan, and M. Masuduzzaman, “Smart Board for Precision Farming Using Wireless Sensor Network.”
[49] J. J. Estrada-Lopez, A. A. Castillo-Atoche, J. Vazquez-Castillo, and E. Sanchez-Sinencio, “Smart Soil Parameters Estimation System Using an Autonomous Wireless Sensor Network with Dynamic Power Management Strategy,” *IEEE Sens. J.*, vol. 18, no. 21, pp. 8913–8923, Nov. 2018, doi: 10.1109/JSEN.2018.2867432.
[50] A. A. Nelson Sales, Orlando Remédios, *Wireless sensor and actuator system for smart irrigation on the cloud*.
[51] A. Dahane, R. Benameur, B. Kechar, and A. Benyamina, “An IoT based smart farming system using machine learning,” *2020 Int. Symp. Networks, Comput. Commun. ISNCC 2020*, pp. 1–6, 2020, doi: 10.1109/ISNCC49221.2020.9297341.
[52] S. K. Sah Tyagi, A. Mukherjee, S. R. Pokhrel, and K. K. Hiran, “An Intelligent and Optimal Resource Allocation Approach in Sensor Networks for Smart Agri-IoT,” *IEEE Sens. J.*, vol. 21, no. 16, pp. 17439–17446, 2021, doi: 10.1109/JSEN.2020.3020889.
[53] I. Dasgupta, J. Saha, P. Venkatasubbu, and P. Ramasubramanian, “AI Crop Predictor and Weed Detector Using Wireless Technologies: A Smart Application for Farmers,” *Arab. J. Sci. Eng.*, vol. 45, no. 12, pp. 11115–11127, 2020, doi: 10.1007/s13369-020-04928-2.
[54] J. Alejandrino, R. Concepcion, V. J. Almero, M. G. Palconit, A. Bandala, and E. Dadios, “A Hybrid Data Acquisition Model using Artificial Intelligence and IoT Messaging Protocol for Precision Farming,” Dec. 2020. doi: 10.1109/HNICEM51456.2020.9400152.
[55] G. Kakamoukas *et al.*, “A Multi-collective, IoT-enabled, Adaptive Smart Farming Architecture,” *IST 2019 - IEEE Int. Conf. Imaging Syst. Tech. Proc.*, 2019, doi: 10.1109/IST48021.2019.9010236.
[56] C. C. Baseca, S. Sendra, J. Lloret, and J. Tomas, “A smart decision system for digital farming,” *Agronomy*, vol. 9, no. 5, May 2019, doi: 10.3390/agronomy9050216.
[57] S. M. Patil and R. Sakkaravarthi, “Internet of things based smart agriculture system using predictive analytics,” *Asian J. Pharm. Clin. Res.*, vol. 10, pp. 148–152, 2017, doi: 10.22159/ajpcr.2017.v10s1.19601.
[58] P. P. Jayaraman, A. Yavari, D. Georgakopoulos, A. Morshed, and A. Zaslavsky, “Internet of things platform for smart farming: Experiences and lessons learnt,” *Sensors (Switzerland)*, vol. 16, no. 11, Nov. 2016, doi: 10.3390/s16111884.
[59] K. N. Bhanu, H. J. Jasmine, and H. S. Mahadevaswamy, “Machine learning Implementation in IoT based Intelligent System for Agriculture,” 2020. doi: 10.1109/INCET49848.2020.9153978.
[60] S. Aruul Mozhi Varman, A. R. Baskaran, S. Aravindh, and E. Prabhu, “Deep Learning and IoT for Smart Agriculture Using WSN,” *2017 IEEE Int. Conf. Comput. Intell. Comput. Res. ICCIC 2017*, pp. 1–6, 2018, doi: 10.1109/ICCIC.2017.8524140.
[61] C. J. H. Pornillos *et al.*, “Smart Irrigation Control System Using Wireless Sensor Network Via Internet-of-Things,” Dec. 2020. doi: 10.1109/HNICEM51456.2020.9399995.
[62] A. D. Boursianis *et al.*, “Smart Irrigation System for Precision Agriculture - The AREThOU5A IoT Platform,” *IEEE Sens. J.*, vol. 21, no. 16, pp. 17539–17547, Aug. 2021, doi: 10.1109/JSEN.2020.3033526.
[63] D. Thakur, Y. Kumar, and S. Vijendra, “Smart Irrigation and Intrusions Detection in Agricultural Fields Using I.o.T.,” in *Procedia Computer Science*, 2020, vol. 167, pp. 154–162. doi: 10.1016/j.procs.2020.03.193.
[64] M. Younes and A. Salman, “A Cloud-Based Application for Smart Irrigation

Management System," *2021 8th Int. Conf. Electr. Electron. Eng. ICEEE 2021*, pp. 85–92, 2021, doi: 10.1109/ICEEE52452.2021.9415913.
[65] R. R. Garud and P. B. Mane, "Smart Farming using Temperature Sensor, Moisture Sensor, Flow Sensor and Ultrasonic Sensor Leading to Water Conservation," *Int. J. Recent Technol. Eng.*, vol. 8, no. 3, pp. 4426–4431, Sep. 2019, doi: 10.35940/ijrte.C5541.098319.
[66] B. Keswani *et al.*, "Adapting weather conditions based IoT enabled smart irrigation technique in precision agriculture mechanisms," *Neural Comput. Appl.*, vol. 31, pp. 277–292, 2019, doi: 10.1007/s00521-018-3737-1.
[67] C. Jamroen, P. Komkum, C. Fongkerd, and W. Krongpha, "An intelligent irrigation scheduling system using low-cost wireless sensor network toward sustainable and precision agriculture," *IEEE Access*, vol. 8, pp. 172756–172769, 2020, doi: 10.1109/ACCESS.2020.3025590.
[68] IEEE Communications Society and Institute of Electrical and Electronics Engineers, *An Interoperable IP based WSN for Smart Irrigation Systems*.
[69] International Conference on Smart Sensors and Application 1. 2015 Kuala Lumpur, Institute of Electrical and Electronics Engineers, International Conference on Smart Sensors and Application 1 2015.05.26-28 Kuala Lumpur, IEEE International Conference on Smart Sensors and Application 1 2015.05.26-28 Kuala Lumpur, and ICSSA 1 2015.05.26-28 Kuala Lumpur, *Precision Irrigation using Wireless Sensor Network*.
[70] F. Kiani and A. Seyyedabbasi, "Wireless sensor network and Internet of Things in precision agriculture," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 6, pp. 99–103, 2018, doi: 10.14569/IJACSA.2018.090614.
[71] M. A. Niño-Zambrano and C. A. Cobos, "Unicauca Virtual: Metamodelos de Universidad Virtual y Herramientas de Soporte," *VII Congr. Iberoam. Informática Educ.*, no. January, 2004.
[72] Á. C. Juan Pérez, Elizabeth Urdaneta, "Metodología Para El Diseño De Una Red De Sensores Inalámbricos," *Universidad, Cienc. y Tecnol.*, vol. 18, no. 70, pp. 12–22, 2014.
[73] "IoT Analytics - ThingSpeak Internet of Things." https://thingspeak.com/ (accessed Sep. 13, 2022).
[74] O. Guzman, "Interfaz de Realidad Aumentada para la Internet de las Cosas," Universidad del Cauca, 2016.
[75] "LoRa GPS HAT Single Channel LoRa & GPS module User Manual LoRa GPS HAT Single Channel LoRa & GPS module User Manual Document Version: 1.0 LoRa GPS HAT Single Channel LoRa & GPS module User Manual," 2019, Accessed: Jul. 07, 2022. [Online]. Available: www.dragino.com