ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Título del trabajo

DISEÑO Y SIMULACIÓN DE UN SISTEMA DE MONITOREO EN TIEMPO REAL PARA POZOS SUBTERRÁNEOS EN EL CANTÓN DURÁN

Previo la obtención del Título de:

Magister en Automatización y Control Industrial

Presentado por:

Sánchez Montero Freddy Yuris

Villa Alcoser Erika Jeaneth

GUAYAQUIL - ECUADOR

Año: 2026

DEDICATORIA

Dedico este posgrado a la fuerza que me sostiene y me impulsa a ser una mejor versión de mí cada día.
A mi amado padre, Yuris Sánchez, por ser sinónimo de trabajo, disciplina y valentía. Gracias por enseñarme que los sueños se construyen paso a paso y que nunca es tarde para avanzar un poco más.

A mi madre, Flor Montero, cuya luz permanece intacta en mi corazón. Aunque su partida en el 2020 dejó un vacío imposible de llenar, su amor, su ternura y su lucha incansable continúan guiando mi camino. Este logro también es tuyo, mamá; eres mi inspiración eterna.

A mis hermanos, Yuris y Haydee, quienes han sido refugio, compañía y motivación en cada etapa. Su apoyo constante, incluso en los momentos más difíciles, ha sido fundamental para no rendirme

Ing. Freddy Sánchez M.

DEDICATORIA

Dedico este logro a tres pilares fundamentales en mi vida: a mi amada madre, Amable Alcoser, cuyo amor inquebrantable y apoyo constante han sido mi fuerza en cada paso de esta travesía académica. A mis queridos hermanos, Carina, Edwin y Cristian, cuyo apoyo y aliento me han inspirado a alcanzar esta meta. Y a mi adorada sobrina, Sofia Villa, quien me recuerda la importancia de perseverar y ser un ejemplo que seguir. Este logro es un tributo a su amor y respaldo incondicional.

Ing. Erika Villa A.

AGRADECIMIENTOS

Deseo expresar mi profundo agradecimiento a quienes han sido mi mayor fuente de fuerza y motivación en este camino.

A mis padres, Yuris Sánchez y Flor Montero, cuyo amor incondicional, valores y confianza en mí han sido el motor que ha impulsado cada uno de mis pasos.

A mis hermanos, por su respaldo constante y por recordarme siempre que nunca estoy solo.

A mi sobrino Thiago, cuya llegada iluminó mi vida y me regaló un nuevo sentido para seguir creciendo y superándome.

A mis profesores de la maestría, gracias por su guía, por compartir su conocimiento y por marcar mi formación profesional.

En especial, mi gratitud a nuestro tutor, Richard Sánchez, por su acompañamiento, orientación y dedicación durante este proceso.

A cada uno de ustedes, gracias por ser parte esencial de este logro. Mi gratitud es profunda y permanente.

Ing. Freddy Sánchez M.

AGRADECIMIENTOS

Expreso mi profundo agradecimiento a Dios, cuya infinita sabiduría ha iluminado mi camino en este viaje académico. Agradezco a mi familia, quienes han sido un pilar constante de apoyo a lo largo de esta travesía. Su amor incondicional y su sacrificio me han brindado la oportunidad de enfocarme plenamente en mis estudios, un regalo invaluable que nunca dejaré de apreciar. También deseo reconocer la contribución significativa del M.Sc. Richard Sanchez. Su experiencia, orientación y la confianza que depositaron en mí fueron esenciales para el éxito de este proyecto. Sus consejos me guiaron por el camino correcto, y su apoyo me dio la confianza necesaria para enfrentar los desafíos que surgieron en el camino.

Ing. Erika Villa A.

DECLARACIÓN EXPRESA

Nosotros Sánchez Montero Freddy y Villa Alcoser Erika acordamos y reconocemos que: La titularidad de los derechos patrimoniales de autor (derechos de autor) del proyecto de graduación corresponderá al autor o autores, sin perjuicio de lo cual la ESPOL recibe en este acto una licencia gratuita de plazo indefinido para el uso no comercial y comercial de la obra con facultad de sublicenciar, incluyendo la autorización para su divulgación, así como para la creación y uso de obras derivadas. En el caso de usos comerciales se respetará el porcentaje de participación en beneficios que corresponda a favor del autor o autores. El o los estudiantes deberán procurar en cualquier caso de cesión de sus derechos patrimoniales incluir una cláusula en la cesión que proteja la vigencia de la licencia aquí concedida a la ESPOL.

La titularidad total y exclusiva sobre los derechos patrimoniales de patente de invención, modelo de utilidad, diseño industrial, secreto industrial, secreto empresarial, derechos patrimoniales de autor sobre software o información no divulgada que corresponda o pueda corresponder respecto de cualquier investigación, desarrollo tecnológico o invención realizada por mí/nosotros durante el desarrollo del proyecto de graduación, pertenecerán de forma total, exclusiva e indivisible a la ESPOL, sin perjuicio del porcentaje que me/nos corresponda de los beneficios económicos que la ESPOL reciba por la explotación de mi/nuestra innovación, de ser el caso.

En los casos donde la Oficina de Transferencia de Resultados de Investigación (OTRI) de la ESPOL comunique los autores que existe una innovación potencialmente patentable sobre los resultados del proyecto de graduación, no se realizará publicación o divulgación alguna, sin la autorización expresa y previa de la ESPOL.

Guayaquil, 8 de Enero del 2026.

Freddy Yuris Sanchez Montero
Time Stamping Security Data

Sánchez Montero Freddy

Erika Jeaneth Villa Alcoser
Time Stamping Security Data

Villa Alcoser Erika

EVALUADORES

Firmado electrónicamente por:
RICHARD MICHAEL SANCHEZ ROSADO
Validar únicamente con FirmaEC

Mgtr. Richard Sánchez

PROFESOR TUTOR

EFREN VINICIO HERRERA MUENTES

Digitally signed by EFREN VINICIO HERRERA MUENTES
Date: 2026.01.20 10:57:56 -05'00'

PhD. Efren Herrera

PROFESOR EVALUADOR

RESUMEN

En la actualidad, la Empresa Municipal de Alcantarillado y Agua Potable del Cantón Durán (EMAPAD-EP) opera ocho pozos subterráneos sin un sistema automatizado de monitoreo, lo que limita la detección oportuna de fallas en caudal, presión y funcionamiento de bombas. Esto genera retrasos en la identificación de problemas y afecta la continuidad del suministro de agua potable.

Este proyecto de tesis tiene como objetivo diseñar y simular un sistema de monitoreo en tiempo real para optimizar la operación de los pozos. Se emplearán sensores estratégicamente ubicados y un sistema de alarmas automáticas que notificarán de inmediato a los ingenieros ante cualquier anomalía. Además, se desarrollará una plataforma de supervisión remota para visualizar los parámetros críticos y tomar decisiones correctivas con mayor rapidez.

Para la simulación se utilizarán tecnologías como APIs para el envío de datos, algoritmos de generación de datos aleatorios y herramientas como Node-RED. Se establece una correspondencia entre los elementos reales y los componentes simulados: el SIMATIC IOT2050 será representado por una API desarrollada en Python, los sensores físicos de presión, caudal y vibración serán emulados mediante algoritmos generadores de datos aleatorios, la comunicación industrial por protocolos como Modbus será reproducida mediante HTTP entre la API y Node-RED; y la visualización se realizará mediante dashboards en Node-RED, permitiendo una supervisión operativa en tiempo real.

Se realizarán pruebas virtuales para evaluar la eficiencia del sistema y validar su operatividad antes de su implementación. Los resultados esperados incluyen el diseño detallado del sistema de monitoreo, la simulación de su funcionamiento y un protocolo de operación y mantenimiento. La implementación de este sistema mejorará la eficiencia operativa y garantizará un suministro de agua potable más confiable para la población de Durán.

Palabras clave: **API**, Node-RED, Simatic IoT 2050, HTTP, Modbus pozos subterráneos.

ABSTRACT

Currently, the Municipal Sewerage and Drinking Water Company of Durán Canton (EMAPAD-EP) operates eight underground wells without an automated monitoring system, which limits the timely detection of flow, pressure, and pump failures. This creates delays in identifying problems and affects the continuity of the drinking water supply.

This thesis project aims to design and simulate a real-time monitoring system to optimize well operation. Strategically located sensors and an automatic alarm system will be used to immediately notify engineers of any anomalies. Additionally, a remote monitoring platform will be developed to visualize critical parameters and make corrective decisions more quickly.

Technologies such as APIs for data submission, random data generation algorithms, and tools such as Node-RED will be used for simulation. A correspondence is established between the real elements and the simulated components: the SIMATIC IoT 2050 will be represented by an API developed in Python; the physical pressure, flow, and vibration sensors will be emulated using random data generating algorithms; industrial communication via protocols such as Modbus will be replicated via HTTP between the API and Node-RED; and visualization will be performed using dashboards in Node-RED, enabling real-time operational monitoring.

Virtual testing will be conducted to evaluate the system's efficiency and validate its operation before implementation. The expected results include the detailed design of the monitoring system, simulation of its operation, and an operation and maintenance protocol. The implementation of this system will improve operational efficiency and ensure a more reliable supply of drinking water for the population of Durán.

Keywords: *API, Node-RED, SIMATIC IoT 2050, HTTP, Modbus, underground wells.*

ÍNDICE GENERAL

ABREVIATURAS

EMAPAD-EP	Empresa Pública Municipal de Agua Potable y Alcantarillado de Durán
API	Interfaz de programación de aplicaciones
IoT	Internet de las cosas
SCADA	Sistema de supervisión y adquisición de datos
HTTP	Protocolo de transferencia de hipertexto
MQTT	Message Queuing Telemetry Transport
OPC UA	Open Platform Communications Unified Architecture

SIMBOLOGÍA

mm/s	milímetros por segundo
m/s^2	metros por segundo al cuadrado
µm	micrómetros
L/s	Litros por segundo
GPM	Galones por minuto
°C	Grados Celsius
mA	Miliamperio

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

1. INTRODUCCIÓN

El acceso a un suministro confiable de agua potable es una necesidad fundamental para cualquier población [1]. La disponibilidad constante de agua es crucial para la salud, el bienestar y el desarrollo de las comunidades, especialmente en regiones en crecimiento como el cantón Durán [2]. En este contexto, la Empresa Municipal de Alcantarillado y Agua Potable (EMAPAD-EP) desempeña un papel esencial al abastecer a los habitantes del cantón mediante la captación de agua subterránea a través de ocho pozos ubicados en la localidad de El Chobo, en el cantón Milagro. Estos pozos proporcionan el agua necesaria para cubrir la demanda de los residentes, garantizando un servicio básico que es indispensable para el desarrollo de la comunidad [3].

No obstante, la gestión actual de estos pozos presenta varias limitaciones que dificultan la eficiencia operativa y aumentan el riesgo de fallas en el sistema. Uno de los principales problemas es que no existe un sistema automatizado de monitoreo y control de los pozos, lo que implica que las operaciones de encendido y apagado de las bombas se realizan de forma manual. Este enfoque presenta varios inconvenientes, como la posibilidad de generar retrasos en la identificación y solución de problemas, lo cual puede afectar gravemente la continuidad y calidad del servicio de agua [4].

Además, la falta de un sistema de monitoreo en tiempo real impide detectar a tiempo anomalías en parámetros clave como el caudal, la presión y la temperatura en los pozos [5]. La incapacidad para identificar estos problemas de manera temprana aumenta significativamente el riesgo de fallas no controladas, las cuales pueden provocar interrupciones en el servicio de agua, generando molestias en la población y complicaciones en la gestión operativa [6]. La ausencia de alertas automáticas para informar de manera instantánea a los ingenieros y operadores también prolonga los tiempos de respuesta, ya que los reportes de fallas deben ser comunicados de forma manual por los operadores, lo que resulta en un retraso

significativo en la toma de decisiones. Muchas veces, los ingenieros reciben la información con varias horas de retraso, lo que agrava aún más los problemas [2]. En este contexto, surge la necesidad urgente de desarrollar un sistema de monitoreo en tiempo real que optimice la operación de los pozos subterráneos, mejorando tanto la detección como la corrección de fallas. Un sistema automatizado de monitoreo podría reducir drásticamente los tiempos de respuesta ante cualquier inconveniente, permitiendo una gestión más eficiente y oportuna de los pozos. Además, este sistema podría garantizar una mayor continuidad del servicio, al detectar fallas en sus primeras etapas y permitir que se tomen las medidas correctivas de manera proactiva, antes de que el problema se agrave y cause interrupciones en el suministro de agua potable.

Este proyecto de tesis propone el diseño y simulación de un sistema avanzado de monitoreo, orientado a optimizar la supervisión de los pozos subterráneos del cantón Durán. A través de un entorno simulado, se emulará el funcionamiento de dispositivos como el SIMATIC IOT2050 [7] mediante una API desarrollada en Python, que generará y transmitirá datos representativos del comportamiento real del sistema. Los sensores físicos de presión, caudal y vibración serán representados por algoritmos que generan datos aleatorios dentro de rangos técnicos realistas.

La simulación incluirá la transmisión de datos mediante el protocolo HTTP [8] hacia una plataforma de supervisión construida en Node-RED [9], que funcionará como una interfaz para visualizar los parámetros críticos en tiempo real. Asimismo, se incorporará un sistema de alarmas automáticas que notificará a los responsables en caso de detectar valores fuera de los rangos establecidos, permitiendo una respuesta oportuna y efectiva. Esta simulación permite validar el diseño funcional del sistema, anticipar su comportamiento operativo y establecer una base sólida para una futura implementación real que contribuya a mejorar la eficiencia operativa y garantizar un suministro de agua más confiable para la población.

El desarrollo de este sistema de monitoreo tiene un impacto significativo en la mejora del suministro de agua potable en Durán. Al reducir los tiempos de respuesta ante fallos, el sistema contribuirá a una mayor eficiencia operativa, optimizando tanto los recursos tecnológicos como humanos. Además, al integrar

una plataforma de monitoreo accesible remotamente, los ingenieros podrán supervisar las operaciones desde cualquier ubicación, facilitando la toma de decisiones rápidas y oportunas, lo que contribuirá a la mejora continua del servicio. En términos generales, el objetivo de este proyecto es crear una solución innovadora que permita mejorar la eficiencia de la operación de los pozos subterráneos, reducir el tiempo de inactividad y garantizar un servicio de agua potable más confiable para la población de Durán. El sistema propuesto no solo se centrará en la detección temprana de fallas, sino también en la optimización del uso de los recursos disponibles, promoviendo una gestión más ágil y proactiva del sistema de captación de agua.

El presente documento está estructurado en cuatro capítulos principales. En el **Capítulo 1**, se plantea el problema, se justifica la necesidad de este proyecto y se definen los objetivos generales y específicos. En el **Capítulo 2**, se describe la metodología utilizada para el diseño y simulación del sistema, así como los procesos de monitoreo y análisis de datos. El **Capítulo 3** presenta los resultados obtenidos a través de la simulación del sistema y su respectivo análisis. Finalmente, en el **Capítulo 4**, se exponen las conclusiones y recomendaciones derivadas de la implementación del sistema de monitoreo en tiempo real.

Con este trabajo, se busca aportar una solución innovadora y eficiente para la gestión de los pozos subterráneos en el cantón Durán, promoviendo el uso de tecnologías avanzadas que mejoren la prestación de servicios esenciales a la comunidad. A través de la automatización del monitoreo y control, se podrá mejorar significativamente la calidad del servicio de agua potable, lo que a su vez contribuirá al bienestar de los habitantes de Durán.

1.1 ESTADO DEL ARTE

El monitoreo automatizado de pozos subterráneos es una tendencia creciente en la gestión de recursos hídricos, tanto a nivel nacional como internacional. Diversos estudios han demostrado que la implementación de sistemas basados en IoT y SCADA permite mejorar la eficiencia operativa, reducir el tiempo de respuesta ante fallos y optimizar el uso de recursos energéticos y humanos [2]. En países como México, Brasil y España, se han desarrollado plataformas de supervisión remota que integran sensores industriales, gateways y protocolos de comunicación como MQTT y Modbus, logrando una gestión más racional y sostenible del agua [4].

En Ecuador, la aplicación de tecnologías de monitoreo en tiempo real aún es incipiente, pero existen iniciativas en empresas públicas y privadas orientadas a modernizar la infraestructura de captación y distribución de agua potable [5]. Se destaca la importancia de medir variables críticas como caudal, presión, temperatura y vibración, ya que su análisis permite anticipar posibles fallos mecánicos, optimizar las estrategias de mantenimiento predictivo y garantizar la continuidad del servicio [7].

El uso de gateways industriales como el SIMATIC IoT2050 facilita la integración de sensores con plataformas de visualización y control, permitiendo el procesamiento local de datos y la transmisión segura hacia la nube [8]. Herramientas como Node-RED han revolucionado el desarrollo de aplicaciones IoT, al ofrecer entornos gráficos flexibles y escalables para la gestión de alarmas, visualización de variables y notificaciones automáticas [9].

La tendencia actual apunta a la incorporación de inteligencia artificial y análisis predictivo en los sistemas de monitoreo, lo que permitirá anticipar eventos críticos y optimizar la toma de decisiones [10]. Sin embargo, la adopción de estas tecnologías requiere una adecuada adaptación a las condiciones locales, considerando factores como la infraestructura existente, el presupuesto disponible y la capacitación del personal.

1.2 Descripción del problema

El abastecimiento de agua potable en el cantón Durán depende de la extracción de agua subterránea a través de los pozos ubicados en la localidad de El Chobo, cantón Milagro. Actualmente, el sistema de operación y

monitoreo de estos pozos es manual, lo que genera retrasos en la detección de fallos y afecta la eficiencia del servicio. La falta de un sistema automatizado impide un control óptimo sobre el encendido y apagado de las bombas, así como el seguimiento en tiempo real de parámetros críticos como el caudal, la presión y la temperatura del agua [6].

Uno de los principales requerimientos del proyecto es diseñar y simular un sistema de monitoreo en tiempo real que permita la supervisión remota del estado operativo de los pozos subterráneos y el funcionamiento de las bombas. Para ello, se contempla la incorporación de sensores simulados para la medición de caudal, presión y vibración, cuyos datos serán generados por algoritmos programados dentro de una API que representa el comportamiento de los dispositivos reales. Estos datos serán transmitidos a una plataforma de monitoreo desarrollada en un entorno simulado, con capacidad de almacenamiento y procesamiento centralizado.
Además, el sistema incluirá un módulo de alarmas automáticas que permitirá enviar notificaciones en tiempo real ante la detección de parámetros anómalos, facilitando así una toma de decisiones inmediata y eficiente.

Las variables de interés incluyen la medición de caudal, presión y temperatura en los pozos, el estado de funcionamiento de las bombas, la eficiencia energética y el tiempo de respuesta ante fallos detectados. El análisis de estos parámetros permitirá optimizar la operación del sistema, reducir interrupciones en el servicio y mejorar la gestión del recurso hídrico en el cantón Durán.

El diseño y simulación del sistema propuesto busca proporcionar una solución eficiente y escalable, que permita mejorar la operatividad y confiabilidad del suministro de agua en la región.

1.3 Justificación del problema

El suministro de agua potable en el cantón Durán depende de la extracción subterránea realizada a través de ocho pozos ubicados en la zona de El Chobo, cantón Milagro [10]. Actualmente, estos pozos operan sin un sistema

automatizado de monitoreo, lo que genera una alta dependencia de la supervisión manual y limita la capacidad de respuesta ante eventos críticos como variaciones de caudal, presión o fallos en las bombas. Esta situación afecta directamente la continuidad del servicio y puede provocar interrupciones imprevistas que impactan negativamente a la población.

Frente a esta problemática, el presente proyecto de tesis propone el diseño y simulación de un sistema de monitoreo en tiempo real que permita evaluar, desde un entorno controlado, una solución tecnológica basada en IoT y SCADA [11] para mejorar la gestión operativa de los pozos. A través de sensores simulados de caudal, presión y vibración, integrados en una arquitectura virtual que emula el uso de un gateway industrial como el SIMATIC IOT2050 [7], se podrá replicar el comportamiento real del sistema, validar su desempeño y visualizar los beneficios de contar con alertas automáticas, monitoreo remoto y toma de decisiones basada en datos.

Desde una perspectiva técnica, la simulación permite anticipar el rendimiento de un sistema que, al implementarse, modernizaría la infraestructura de supervisión de EMAPAD-EP, alineándola con las tendencias actuales de automatización, digitalización e Internet de las Cosas (IoT). Esta etapa previa es esencial para reducir riesgos de fallos en la implementación real y adaptar la solución a las condiciones específicas del entorno operativo, como la distancia entre los pozos y el centro de monitoreo o la compatibilidad con equipos existentes.

Económicamente, el diseño de un sistema automatizado implica una reducción proyectada en los costos de operación asociados al control manual, y una mitigación del riesgo de daños por fallos no detectados a tiempo. Además, el uso eficiente de los recursos mediante monitoreo continuo contribuirá a una administración más racional del agua y de la energía utilizada por las bombas.

Desde el punto de vista social y ambiental, el proyecto aporta a la mejora del servicio de agua potable en Durán, garantizando mayor confiabilidad, continuidad y calidad para los usuarios. Al mismo tiempo, permite optimizar el

consumo energético, minimizar pérdidas de agua y reducir el impacto ambiental asociado al bombeo descontrolado.

En este contexto, la simulación del sistema propuesto no solo constituye una herramienta de validación técnica, sino también una respuesta proactiva ante una necesidad urgente de EMAPAD-EP: mejorar la eficiencia y sostenibilidad del sistema de captación subterránea de agua, asegurando el abastecimiento a largo plazo de una población en constante crecimiento.

1.4 Objetivos

1.4.1 Objetivo General

Desarrollar un sistema de monitoreo en tiempo real para los pozos subterráneos en el cantón Durán mediante la simulación de sensores y alarmas automatizadas, con el fin de mejorar la eficiencia en la detección de fallos y la respuesta a problemas operativos.

1.4.2 Objetivos Específicos

1. Simular señales de sensores de caudal, presión y temperatura en los pozos subterráneos para obtener datos en tiempo real sobre el funcionamiento del sistema de captación de agua.
2. Diseñar las funciones de envío de alertas automáticas basadas en los parámetros simulados, que permitan notificar a los ingenieros responsables sobre fallos o anomalías detectadas en el sistema.
3. Desarrollar una interfaz gráfica de monitoreo accesible remotamente para que los ingenieros puedan supervisar el sistema desde cualquier ubicación en tiempo real.
4. Configurar un sistema de registro y almacenamiento de datos del monitoreo en tiempo real para permitir el análisis histórico de las operaciones y detectar patrones de fallos recurrentes.

1.5 Marco teórico

1.5.1 Sistema de monitoreo

Un sistema de monitoreo es la unión de componentes que permiten registrar y observar en tiempo real el estado de un proceso, con el fin de evaluar constantemente el desempeño de actividades, anticipar fallos y tomar decisiones basadas en información actualizada, bajo el escenario de sistemas hidráulicos, el monitoreo permite conocer el comportamiento de variables como presión, caudal, nivel y funcionamiento de bombas, permitiendo asegurar la continuidad de los servicios.

Figura 1.1 Sistema de protección y monitores de agua potable [12]

1.5.2 Variables críticas en pozos subterráneos

En la supervisión de los pozos subterráneos de captación de agua potable, existen algunas variables que deben ser monitoreadas de forma constante para garantizar un funcionamiento optimo [12]:

- Caudal: indicador de la cantidad de agua que fluye por unidad de tiempo, esta variable nos permite identificar obstrucciones o deterioro de las bombas
- Presión: señala la fuerza con la que el agua es impulsada por las tuberías, cuando hay una caída de este indicador esto se puede traducir como fugas o fallas en la bomba

- Vibración: variable fundamentar para tomar la decisión de cuando se debe realizar un mantenimiento predictivo de bombas, los niveles que están fuera de los parámetros normales pueden anticipar fallos mecánicos o desequilibrios.
- Temperatura: medidas de temperaturas fueras del rango normal en el motor, nos permite identificar sobrecaras, problemas de refrigeración o fallos eléctricos.

La medición constante de estas variables permite un diagnóstico continuo del sistema y activar los protocolos de mantenimiento o contingencias necesarios.

Figura 1.2 Caudalímetro de vórtex [13]

1.5.3 Internet de las cosas (IoT) aplicado al monitoreo

El internet de las cosas es un paradigma de la tecnología ya que este permite la conexión de dispositivos físicos con la red de internet para la recolección, intercambio y análisis de datos, este también permite la instalación de sensores inteligente los cuales transmiten datos en tiempo real a servidores o plataformas en la nube sin la intervención humana, dando soluciones a la supervisión remota y respuesta automatizada ante evento anómalos, mejorando la eficiencia operativa [14].

1.5.4 SIMATIC IoT2050 como gateway de monitoreo

Desarrollado por Siemens, es un gateway de alto rendimiento que permite conectar sensores industriales con sistemas de supervisión remota, procesando datos localmente y transmitiéndolos hacia plataformas en la nube mediante diferentes protocolos de comunicación como los son MQTT, OPC UA y HTTP. Su arquitectura permite trabajar en relación con lenguajes como Python o Node.js, lo que hace ideal al dispositivo para proyectos IoT industriales [7].

Figura 1.3 SIMATIC IOT2050 [7]

1.5.5 APIs y generación de datos simulados

Las APIs (Interfaces de programación de aplicaciones) son herramientas desarrolladas con la finalidad de permitir la comunicación de datos entre diferentes sistemas, en este proyecto, una API será desarrollada para simular el comportamiento de un sistema de sensores, generando datos aleatorios de caudal, presión vibración y temperatura, permitiendo validar el diseño lógico del sistema de monitoreo y replicando el flujo de datos de un entorno real [15].

1.5.6 Node-RED

Herramienta de desarrollo de aplicaciones basadas en flujos IoT, diseñada por IBM y permite integrar diferentes dispositivos, procesar datos y a su vez generar visualizaciones mediante un entorno grafico vasado en nodos, en proyectos industriales Node-Red actúa como el cerebro del sistema SCADA simulado, ya que este recibe los datos enviados por la API, genera el dashboard interactivo y activa las alarmas en caso de anomalías, debido a su gran flexibilidad y facilidad de manejo se convirtió en la herramienta ideal para entornos de prueba y simulación en proyectos de monitoreo industrial [16].

1.5.7 Protocolos de comunicación

La transmisión de datos en un sistema de monitoreos es un tema esencial ya que este se realiza mediante protocolos que definen cómo se realizará el intercambio de información entre diferentes dispositivos, las más relevantes y utilizados son [8]:

- HTTP (HyperText Transfer Protocol): Protocolo tradicional para envió de datos entre servidores y clientes web
- MQTT (Message Queuing Telemetry Transport): Protocolo ligero, ideal para dispositivos que cuentan con poca capacidad de red, permitiéndoles la comunicación en tiempo real con bajo consumo.
- Modbus TCP/RTU: Estándar industrial que es ampliamente utilizado para la comunicación entre distintos sensores y dispositivos de control.

El uso de estos protocolos de comunicación garantiza la interoperabilidad, velocidad y confiabilidad en la transmisión de datos [8].

1.5.8 Alarmas automáticas y gestión de eventos

Una alarma de proceso es un mecanismo para informar a un operador de una condición anormal del proceso que requiere una acción del operador [17], ayuda a prevenir o mitigar trastornos y perturbaciones del proceso y así evitar el apagado automático de la planta o parte de ella [18].

La decisión de configurar una alarma debe cumplir con los siguientes criterios se enlistan los más importantes [19]:

- Condición anormal: Las alarmas solo se producirán en situaciones anormales (en contraposición a la operación normal). Por lo tanto, las alarmas no deben existir para informar a los operadores de cambios normales en la operación.
- Acción del operador: Solo los eventos que requieren acción del operador se configurarán como alarmas. Los eventos que no requieran acción del operador (por ejemplo, útiles solo para fines informativos) se presentarán de otras maneras.

Tiempo disponible para responder: Los límites de alarma deben ser seleccionados adecuadamente para proporcionar al operador el tiempo disponible para responder y tomar acciones correctivas efectivas.

CAPÍTULO 2

2. METODOLOGÍA

En este capítulo se detallan los requerimientos funcionales y técnicos del sistema de monitoreo en tiempo real propuesto para los pozos subterráneos del cantón Durán. A partir del análisis del problema y de las necesidades identificadas en el capítulo anterior, se presentan diversas alternativas de solución, evaluando sus ventajas y limitaciones.

2.1 Requerimiento de diseño

A continuación, se plantean los principales requerimientos funcionales y técnicos que guían el desarrollo del presente proyecto de diseño y simulación de un sistema de monitoreo en tiempo real para los pozos subterráneos operados por EMAPAD-EP en el cantón Durán:

- **Accesibilidad (simulada):** El sistema simulado debe contemplar una arquitectura de supervisión remota, accesible desde cualquier dispositivo con conexión a internet. Esta funcionalidad será replicada mediante un dashboard alojado en Node-RED, emulando el acceso de técnicos desde diferentes ubicaciones.
- **Costo (consideración de diseño):** Aunque no se realizará una implementación física, el diseño debe tomar en cuenta componentes de bajo costo como sensores industriales comunes y gateways como el SIMATIC IOT2050, permitiendo que la simulación sea un reflejo viable de una futura implementación económica para municipios con presupuesto limitado.
- **Facilidad de instalación (simulada):** El diseño debe basarse en una solución que pueda integrarse en pozos existentes sin necesidad de modificaciones estructurales. Esto se simula a través de una arquitectura modular virtual, con nodos y funciones fácilmente escalables en software.
- **Simplicidad de operación:** La simulación debe incluir una interfaz gráfica sencilla, que represente un dashboard intuitivo para usuarios sin experiencia previa en sistemas SCADA. Se buscará que la visualización de variables sea clara, con indicadores visuales, colores y alertas.

- **Precisión (emulada):** En la simulación, los sensores virtuales generarán datos dentro de rangos técnicos realistas, representando mediciones confiables de caudal, presión, vibración y temperatura. Esto permitirá validar la lógica de procesamiento y visualización.
- **Alertas tempranas:** El sistema simulado debe incluir lógica de alarmas automáticas que se activen ante condiciones críticas definidas (por ejemplo, caída de presión o aumento de vibración), emulando la notificación inmediata a los operadores mediante visualización o mensajes emergentes en el dashboard.

2.2 Alternativas de solución

Con la finalidad de resolver el problema de la detección tardía fallos en la operación de los pozos subterráneos los cuales se encuentran ubicados en el Cantón Milagro, El Chobo, se plantearon tres alternativas tecnológicas, estas han sido evaluadas en función de la viabilidad técnica, operativa, económica y su capacidad de sostenibilidad y escalabidad a largo plazo en la operación del sistema de captación de agua del Cantón Durán.

- La primera alternativa planteada permite la automatización de la detección de fallos en base a la lectura de los sensores, pero su principal limitación es la imposibilidad de una supervisión remota desde otras localidades de EMAPAD-EP, lo que implica que todas las decisiones a tomar en presencia de un fallo debían tomarse netamente de forma presencial, manteniendo una dependencia de personal técnico local,
- La segunda alternativa de implementación utiliza interfaz web accesible desde cualquier dispositivo con conexión a internet, esta alternativa ofrecía acceso remoto, almacenamiento centralizado, visualización de datos y flexibilidad de expansión del sistema.
- La tercera alternativa, es muy similar a la alternativa 2, pero este hace uso de la plataforma Node-Red que permite realizar la interfaz gráfica, así como también el diseño del flujo de datos de manera visual e integrar alarmas con

gran facilidad y mayor personalización con estructura modular de mantenimiento y ampliación del sistema.

2.3 Matriz de decisión

Con el fin de elegir la opción más adecuada, se elaboró una matriz de decisión en la que se tomaron en cuenta los requisitos de diseño, asignándoles pesos específicos conforme a su nivel de relevancia. Estos valores se presentan en la Tabla 2.1 Criterios de selección.

Tabla 2.1 Criterios de selección

Criterio	Accesibilidad	Costo	Facilidad de instalación	Simplicidad de operación	Precisión	Alertas tempranas	Σ+1	Ponderación
Accesibilidad	-	0	1	0	0	0	1	0.067
Costo	1	-	0	1	0,5	0	2	0.133
Facilidad de instalación	0	1	-	0.5	0	0	1.5	0.1
Simplicidad de operación	1	0	0.5	-	0	0	1.5	0.1
Precisión	1	1	1	1	-	0.5	4.5	0.3
Alertas tempranas	1	1	1	1	0.5	-	4.5	0.3
						Total	15	1

En la Tabla 2.1 Criterios de selección, se observa que el criterio con la mayor ponderación es la "Precisión" y “Alertas tempranas” seguido del "costo" y, en última instancia, la "Accesibilidad". Después de establecer las ponderaciones de cada criterio, se procedió a una evaluación comparativa entre las alternativas de solución y cada criterio. Para llevar a cabo este análisis, se generaron tres tablas, cuyos resultados se presentan en la Tabla 2.2 Matriz de decisión.

Tabla 2.2 Matriz de decisión

Conclusión	Accesibilidad	Costo		Facilidad de instalación	Simplicidad de operación	Precisión	Alertas tempranas	∑+1	Prioridad
Solución 1	0.00	0.00		0.67	0.00	0.33	0.00	1.00	3.00
Solución 2	0.50	0.33		0.17	0.33	0.33	0.50	2.20	2.00
Solución 3	0.50	0.67		0.17	0.67	0.33	0.50	2.80	1.00

Tras realizar un análisis comparativo con cada una de las alternativas planteadas, se concluyó que la alternativa 3 era la más adecuada en el contexto en el cual se está elaborando el presente trabajo bajo la Empresa Municipal de Alcantarillado y Agua Potable de Durán (EMAPAD-EP), por las siguientes razones:

- **Supervisión remota en tiempo real:** permite acceso desde cualquier dispositivo con conexión a internet, eliminando la necesidad de presencia física de personal con conocimiento técnico en los pozos, facilitando la intervención oportuna ante emergencias o fallos.
- **Flexibilidad y personalización:** Node-Red permite el uso de herramientas visuales y de desarrollo rápido para crear flujos de datos y reglas de automatización adaptas a las necesidades de la empresa, lo que supera las funcionalidades de una interfaz web ya que esta para realizar cosas se hacen netamente con codificación.
- **Notificaciones inteligentes:** Node-Red permite la integración de notificaciones tanto por correo como por aplicativos como telegram por medio de chatbots, en caso de fallos en los parámetros de monitoreo, reduciendo los tiempos de reacción.
- **Escalabilidad y mantenimiento:** Gracias a la arquitectura modular de Node-RED, permite agregar más sensores a través del tiempo permitiendo un rediseño del sistema constantemente.

- **Optimización de Recursos:** Se redujo la carga operativa sobre el personal técnico, permitiendo su reasignación a tareas de análisis y mantenimientos preventivos.

En conclusión, la alternativa 3 fue seleccionada como la solución definitiva por su capacidad y facilidad de integración con tecnologías emergentes, adaptabilidad, roma de decisiones de forma rápida mediante alarmas automatizadas, siendo una solución innovadora, escalable y buena relación costo-beneficio.

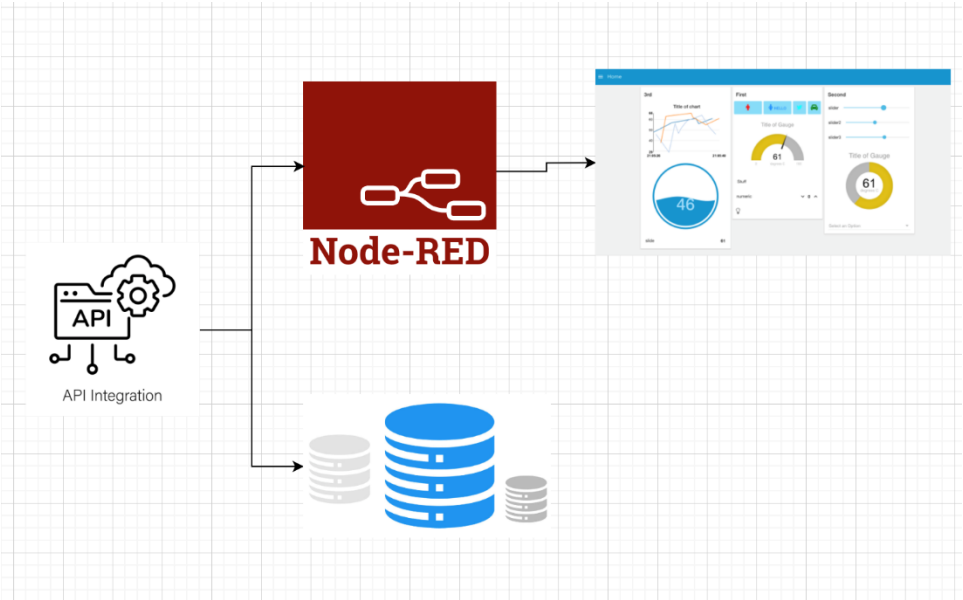


Figura 2.1 Alternativa C

2.4 Proceso de diseño

El desarrollo del proyecto se realizó en diferentes etapas. Inicialmente, se identificó la problemática y la solución más adecuada en base a la matriz de decisiones, se desarrolló el diseño simulado del sistema, el cual cuenta de 3 subetapas las cuales son generación y envió de datos simulados, visualización y monitoreo en tiempo real y alarmas y validación del comportamiento del sistema, finalmente se plantea un diseño realista del sistema como una propuesta futura de implementación, como se muestra en la Figura 2.2 Proceso de diseño.

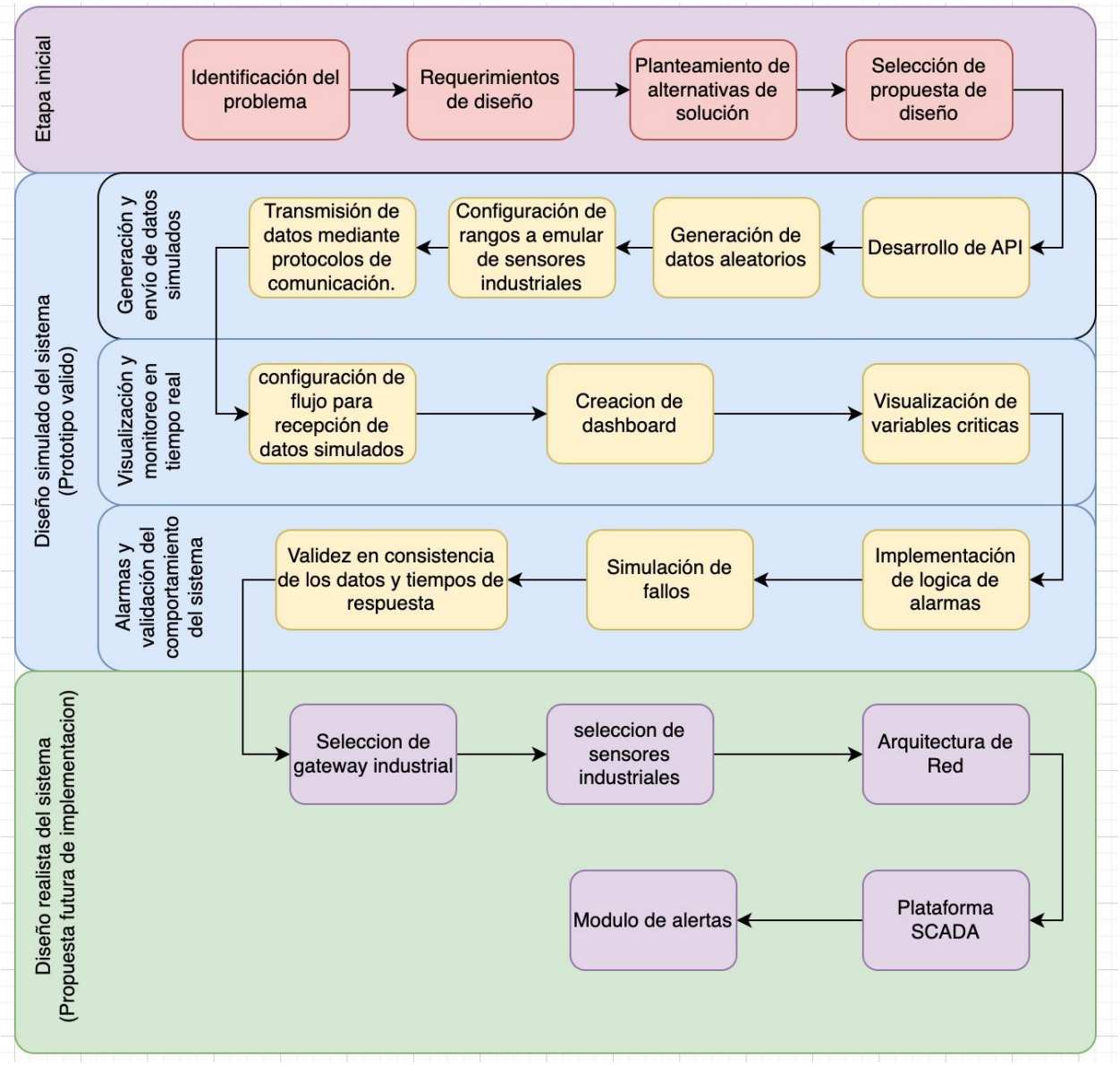


Figura 2.2 Proceso de diseño

2.4.1 Diseño simulado del sistema

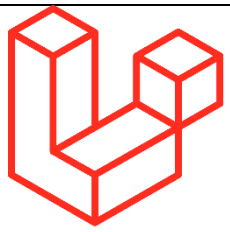
El diseño del sistema de monitoreo en tiempo real para la detección de fallos en pozos subterráneos se planificó en tres capas funcionales generación y envío de datos simulados, visualización y monitoreo en tiempo real y alarmas y validación del comportamiento del sistema. Cada una de las capas cumplen con un rol esencial en el flujo de datos para garantizar una operación eficiente escalable y con alta disponibilidad.

2.4.1.1 Generación y envió de datos simulados

2.4.1.1.1 Desarrollo de la API

Para simular el comportamiento de sensores industriales en un entorno controlado, se planteó la necesidad de desarrollar una API encargada de generar, estructurar y distribuir los datos simulados hacia el sistema de monitoreo. Para la selección del entorno de desarrollo backend [20], se evaluaron tres alternativas principales: PHP (Laravel) [21], Node.js (Express.js) [16] y FastAPI (Python) [22]. A continuación, se presenta una comparación entre las opciones:

Tabla 2.3 Alternativas de entorno de desarrollo backend

Criterio	**PHP (Laravel)**	**Node.js (Express.js)**	**FastAPI (Python)**
Logo			
Rendimiento en tiempo real	Moderado	Alto (arquitectura no bloqueante)	Alto (asincronía nativa)
Facilidad de desarrollo	Alta (estructura clara)	Alta (ligero y flexible)	Alta (autodocumentación con OpenAPI)
Soporte para simulación científica	Limitado	Bajo (requiere librerías externas)	Alto (NumPy, Pandas, SciPy)
Escalabilidad y microservicios	Limitada (enfoque monolítico)	Buena	Excelente (orientado a APIs REST modernas)
Integración futura con IA/ML	Muy limitada	Posible pero no nativa	Excelente (compatible con TensorFlow, PyTorch)
Curva de aprendizaje	Moderada	Baja	Baja (especialmente para desarrolladores Python)

Luego del análisis comparativo, se optó por FastAPI como tecnología principal para el desarrollo de la API. Esta decisión se basa no solo en su rendimiento y arquitectura moderna, sino también en su compatibilidad directa con librerías científicas y de aprendizaje automático. Dado que la

API no solo simulará datos, sino que también podría integrarse en fases futuras con modelos predictivos, FastAPI representa una solución escalable, eficiente y alineada con los objetivos de innovación tecnológica del proyecto [22].

Cabe destacar que la API desarrollada también simula el comportamiento del gateway IoT2050 de Siemens, el cual, en un entorno real, se encargaría de recopilar y transmitir los datos desde los sensores hacia el sistema central de monitoreo. Este gateway tiene la ventaja de operar de forma independiente sin requerir la intermediación de un PLC, lo que simplifica la arquitectura del sistema y reduce costos de implementación.

En esta fase de diseño simulado, la API cumple el rol lógico del gateway, permitiendo validar la transmisión y recepción de datos de manera continua y eficiente, como lo haría el dispositivo físico en campo.

2.4.1.1.2 Generación de Datos aleatorios

Para simular el comportamiento de sensores industriales reales como caudal, presión, temperatura y vibración, se implementaron algoritmos de generación de datos basados en funciones pseudoaleatorias, utilizando distribuciones normales (gaussianas) y uniformes. Estas funciones permiten representar tanto el funcionamiento esperado del sistema como condiciones anómalas o de falla.

Objetivo de la Simulación

- **Comportamiento normal del sistema**: Los datos generados se mantienen dentro de rangos operativos definidos, con una variación ligera que simula la naturaleza dinámica de las condiciones reales en campo.
- **Eventos de falla**: Se introducen aleatoriamente picos o valores fuera de rango para emular fallos como sobrepresión, baja de caudal o alta vibración, con el fin de probar la respuesta del sistema de alarmas y su detección automática.

Intervalo de generación y transmisión

La API se programó para generar y transmitir un nuevo conjunto de datos simulados cada 5 segundos, considerando un equilibrio entre realismo y rendimiento del sistema en entorno de pruebas. Este intervalo es configurable y se definió en función del tiempo de respuesta que se espera en un sistema de monitoreo en tiempo real.

2.4.1.1.3 Configuración de Rangos de Sensores Industriales

Se establecieron los siguientes rangos de valores simulados, basados en especificaciones típicas de sensores industriales [23]:

- **Caudal**: 100 – 400 m³/h
- **Presión**: 1 – 10 bar
- **Temperatura del agua**: 15 – 35 °C
- **Vibración**: 0 – 20 mm/s

Estos rangos pueden ser ajustados manualmente a través de parámetros en la API para emular diferentes condiciones de operación, como sobrecarga o averías en el sistema de bombeo.

2.4.1.1.4 Transmisión de Datos mediante Protocolos de Comunicación

Para garantizar una transmisión eficiente, ligera y continua de los datos generados por la API, se evaluaron diferentes protocolos de comunicación utilizados comúnmente en sistemas IoT e industriales. La siguiente tabla resume las características comparativas de los principales protocolos analizados:

Tabla 2.4 Protocolos de comunicación

Criterio	HTTP REST	WebSocket	MQTT
Modelo de comunicación	Petición-respuesta	Bidireccional persistente	Publicador/Suscriptor
Eficiencia en tiempo real	Baja	Alta	Muy alta

Consumo de ancho de banda	Alto (cada envío requiere headers)	Medio	Muy bajo (cabeceras pequeñas)
Simplicidad de implementación	Alta	Media	Alta
Mantenimiento de conexión	No persistente	Requiere mantener conexión activa	Persistente con bajo costo
Escalabilidad	Limitada	Media	Alta (ideal para múltiples nodos y clientes)
Integración con brokers IoT	No nativa	Limitada	Total (compatible con Mosquitto, HiveMQ, etc.)

Con base en el análisis comparativo, se eligió MQTT (Message Queuing Telemetry Transport) [24] como el protocolo ideal para el envío de datos simulados. Su arquitectura basada en publicación-suscripción permite una comunicación eficiente entre la API y el sistema de monitoreo, incluso cuando se manejan múltiples sensores o clientes simultáneamente. Su bajo uso de ancho de banda, junto con la posibilidad de escalar fácilmente hacia plataformas industriales reales, lo convierte en la opción más adecuada para este proyecto.

Además, su compatibilidad con brokers industriales como Mosquitto y con futuras arquitecturas en la nube asegura que el sistema simulado podrá ser migrado sin dificultad a un entorno productivo real.

2.4.1.2 Visualización y monitoreo en tiempo real

Una vez generados y transmitidos los datos simulados por la API, se procedió al desarrollo del módulo de visualización y monitoreo, cuyo propósito es permitir el seguimiento en tiempo real del estado operativo de los pozos subterráneos. Este proceso involucra la transmisión de datos a través del protocolo MQTT hacia una interfaz construida sobre Node-RED, facilitando tanto la representación gráfica como el control lógico del sistema

2.4.1.2.1 Flujo de datos en tiempo real hacia Node-RED

El flujo se establece desde la API simulada, que actúa como gateway virtual y genera datos cada 5 segundos. Esta API publica los datos al broker MQTT, utilizando un tema específico (por ejemplo, sensores/pozos).

Node-RED, a su vez, se configura como cliente suscriptor (subscriber) a ese tema MQTT. Al recibir los datos, los flujos de Node-RED permiten:

- Descomponer el mensaje JSON recibido.
- Almacenar temporalmente los datos.
- Enviar los datos hacia componentes de visualización como gráficas, indicadores y alertas.

Esta arquitectura modular permite que el sistema pueda ser escalado fácilmente o adaptado a múltiples pozos sin necesidad de modificar el backend.

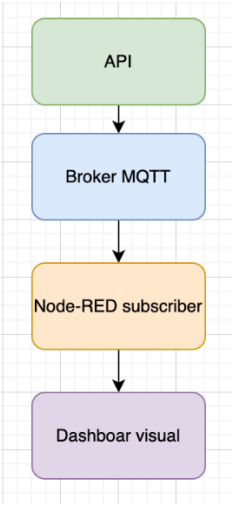


Figura 2.3 Flujo de datos

2.4.1.2.2 Creación del dashboard

Para la interfaz de visualización en tiempo real, se evaluaron tres tecnologías ampliamente utilizadas en proyectos IoT y sistemas de monitoreo industrial. La siguiente tabla presenta una comparación entre sus ventajas y limitaciones:

Tabla 2.5 Interfaz de visualización ventajas-desventajas

Herramienta	Ventajas principales	Desventajas
Grafana	Potente para métricas en tiempo real, integración con MQTT, múltiples fuentes de datos	Requiere configuración avanzada, curva técnica alta
React + Chart.js	Total personalización, diseño atractivo, control total del frontend	Mayor tiempo de desarrollo, requiere experiencia
Node-RED Dashboard	Implementación rápida, integración nativa con MQTT, sin necesidad de código complejo	Limitado para visualizaciones muy específicas o a medida

Tras el análisis comparativo, se decidió utilizar Node-RED Dashboard como herramienta principal de visualización. Esta decisión se fundamenta en su facilidad de implementación, su integración nativa con el broker MQTT, y su enfoque de prototipado ágil, ideal para sistemas que aún están en fase de simulación y validación.

Node-RED permite, en cuestión de minutos, construir una interfaz funcional que represente las variables críticas (caudal, presión, temperatura y vibración), con actualización dinámica y lógica de alertas visuales. Además, al no requerir programación avanzada, facilita futuras adaptaciones o pruebas rápidas por parte del equipo técnico.

Esta elección también se alinea con el objetivo del proyecto: demostrar la viabilidad del monitoreo en tiempo real de los pozos, sin incurrir en altos costos ni tiempos prolongados de desarrollo.

Con los valores simulados de las variables, se definieron condiciones de alarma que permitirían detectar situaciones anómalas en tiempo real. Se programaron rangos críticos y de advertencia para cada parámetro:

- Caudal fuera del rango esperado por obstrucción o fuga.
- Presión baja (pérdida de potencia) o alta (riesgo de daño en la red).
- Temperaturas elevadas que podrían indicar sobrecarga en los motores o mal funcionamiento.

Estas condiciones fueron programadas dentro del entorno de Node-RED, permitiendo generar alertas automáticas que se visualizan en la interfaz web y, adicionalmente, pueden ser enviadas a través de un chatbot conectado a telegram

2.4.1.2.3 Visualización de variables criticas

Una de las etapas clave fue el desarrollo de la interfaz gráfica de monitoreo remoto, la cual fue construida utilizando Node-RED Dashboard, una herramienta de desarrollo visual basada en flujos. La interfaz fue alojada en un servidor accesible vía web, lo que permite a los ingenieros de EMAPAD-EP visualizar en tiempo real los valores de los sensores desde cualquier dispositivo con acceso a internet.

La interfaz incluye:

- Gráficos en tiempo real de caudal, presión y temperatura.
- Indicadores de estado de cada pozo (activo/inactivo, en falla).
- Historial de alarmas y eventos.
- Paneles de control para futuras expansiones (ej. control remoto de bombas o válvulas).

Se priorizó un diseño limpio, intuitivo y adaptable a distintos tamaños de pantalla para garantizar la usabilidad desde laptops, tablets o smartphones.

Finalmente, se validó el correcto funcionamiento del sistema mediante simulaciones controladas, comparando los datos generados por los sensores con los umbrales definidos y verificando que las alertas se activaran conforme a lo previsto.

También se evaluó la conectividad entre el Simatic IoT 2050 y la plataforma de visualización, comprobando la integridad de los datos transmitidos, la latencia de respuesta y la estabilidad del sistema ante interrupciones temporales de red. Se simuló la pérdida de señal de un sensor y la

desconexión del IoT 2050 para analizar el comportamiento del sistema ante fallas y confirmar su capacidad de recuperación.

Esta metodología permitió asegurar que el sistema no solo respondiera adecuadamente a eventos operativos reales, sino que también ofreciera una plataforma flexible, segura y lista para su implementación en campo.

2.4.1.3 Alarmas y validación del comportamiento del sistema

En esta sección se detallan los métodos utilizados para implementar el sistema de alarmas, simular fallos y validar la consistencia de los datos y los tiempos de respuesta. Estas etapas son esenciales para garantizar que el sistema de monitoreo en tiempo real funcione de manera eficiente y confiable, permitiendo una rápida detección y respuesta ante fallos en los pozos subterráneos del Cantón Durán.

2.4.1.3.1 Implementación lógica de alarmas

La implementación de la lógica de alarmas es uno de los aspectos más cruciales del sistema, ya que permite a los operadores detectar rápidamente cualquier anomalía en los parámetros de los pozos y actuar en consecuencia. La lógica de alarmas se basa en los valores de los sensores y en los umbrales establecidos para cada parámetro crítico (caudal, presión, temperatura y vibración).

- **Definición de umbrales**: Se definieron rangos normales para cada variable monitorizada. Si el valor de cualquier parámetro supera estos umbrales, el sistema activa una alarma. Por ejemplo, si el caudal cae por debajo de 100 m^3/h o la presión sube por encima de 10 bar, se considerará un evento crítico que requiere intervención.
- **Lógica de alarmas**: La lógica implementada en Node-RED analiza los datos de los sensores en tiempo real y compara estos valores con los umbrales establecidos. Cuando se detecta un valor fuera de rango, se genera una alerta visual en el dashboard de Node-RED y, si se configura, una notificación automática a través de un chat-bot en Telegram o por correo electrónico.

- **Clasificación de alarmas**: Las alarmas se clasifican en tres categorías:
    - **Advertencia**: Para valores que están cerca de los umbrales, pero no representan una amenaza inmediata.
    - **Crítica**: Cuando los valores superan significativamente los límites de seguridad, lo que indica la necesidad de una acción rápida.
    - **Emergencia**: Para situaciones extremas, como la desconexión de un sensor o fallos en el sistema, que requieren atención inmediata.

La implementación de esta lógica permite la automatización de la supervisión, minimizando el tiempo de respuesta ante cualquier incidente.

2.4.1.3.2 Simulación de fallos

La simulación de fallos es esencial para probar la capacidad del sistema de monitoreo para detectar y responder adecuadamente ante eventos inesperados. En este proyecto, se simulan diversos tipos de fallos en los datos de los sensores para evaluar cómo el sistema maneja situaciones anómalas.

- Fallos simulados: Los fallos se introducen de manera controlada en los datos de los sensores para simular condiciones de operación fuera de lo normal, como:
    - Pérdida de señal de un sensor: Simula la desconexión de un sensor, lo que podría ocurrir debido a fallos en la red o problemas técnicos. En este caso, el sistema debe ser capaz de detectar la falta de datos y activar una alarma de desconexión.
    - Condiciones extremas: Se generan valores fuera de rango, como caudal muy bajo (por debajo de 50 m^3/h), presión excesiva (superior a 12 bar) o temperatura muy alta (más de 35 °C). Estos eventos simulan fallos mecánicos o de operación en el sistema de captación de agua.

        - Error de comunicación: Se simulan errores de transmisión de datos entre los sensores y la plataforma de monitoreo, evaluando cómo el sistema maneja las pérdidas de datos y asegura la integridad del sistema.
    - Pruebas de comportamiento: Durante la simulación de fallos, se monitorea el comportamiento del sistema para garantizar que se detecten los fallos y se active la alarma correspondiente. Además, se evalúa si el sistema de recuperación automático se activa correctamente para restaurar la comunicación o reestablecer las condiciones normales de operación.

Esta simulación ayuda a validar la robustez del sistema y asegurar que el monitoreo en tiempo real pueda detectar rápidamente los problemas y prevenir daños mayores.

2.4.1.3.3 Validez en consistencia de los datos y tiempos de respuesta

La validación de la consistencia de los datos y los tiempos de respuesta es crítica para garantizar que el sistema de monitoreo funcione correctamente en un entorno real.

- **Consistencia de los datos**: Para validar la consistencia, los datos generados por los sensores simulados deben cumplir con los rangos predefinidos de operación. Cualquier dato fuera de los rangos establecidos se considera erróneo y debe ser detectado por el sistema. Además, los datos se verifican en cada etapa del flujo de información: desde la generación en la API, pasando por la transmisión a través de MQTT, hasta su visualización en el dashboard de Node-RED.
    - **Chequeo de integridad**: Se implementaron funciones de validación en la API para garantizar que los datos generados sean consistentes. Si los valores generados no están dentro de los rangos establecidos, se activan mecanismos de control para evitar que dichos datos lleguen al sistema de monitoreo.

        - **Prueba de valores extremos**: Se inyectan valores extremos en la simulación para evaluar si el sistema puede manejar condiciones extremas sin comprometer la precisión de los datos o la operación del sistema.
    - **Tiempos de respuesta**: Los tiempos de respuesta del sistema se refieren a la rapidez con la que se detectan las anomalías y se generan las alertas. Se evaluó el tiempo que tarda la API en generar y transmitir un conjunto de datos, así como el tiempo necesario para que Node-RED actualice el dashboard con los nuevos valores. El objetivo es que estos procesos no tomen más de 3 segundos para garantizar un monitoreo en tiempo real eficiente.
        - **Medición de latencia**: Se midió el tiempo de latencia entre la generación de datos en la API y su visualización en el dashboard. Los resultados mostraron que el sistema es capaz de actualizar los datos en tiempo real, con una latencia mínima, lo que asegura que las alertas se emitan sin demoras.
        - **Evaluación de la eficiencia de respuesta**: Se simularon eventos de emergencia, como una caída de presión o aumento de vibración, y se midió el tiempo que tarda el sistema en detectar el evento y activar la alerta. El sistema mostró tiempos de respuesta óptimos, con alertas emitidas en menos de 2 segundos en la mayoría de los casos.

La validación de la consistencia de los datos y la eficiencia en los tiempos de respuesta garantizan que el sistema de monitoreo sea fiable, preciso y capaz de ofrecer una respuesta rápida ante cualquier problema detectado en los pozos subterráneos.

2.4.2 Diseño realista del sistema

El diseño realista del sistema representa una proyección hacia la implementación física del sistema de monitoreo en pozos subterráneos del cantón Durán. A diferencia del entorno simulado, este enfoque considera la integración de hardware industrial, redes robustas, protocolos seguros y

plataformas SCADA confiables. El objetivo es plantear una arquitectura viable y escalable para ser aplicada en campo, alineada con los requerimientos técnicos de EMAPAD-EP.

2.4.2.1 Selección de Gateway industrial

El gateway industrial actúa como puente entre los sensores de campo y la red de monitoreo. Para su selección, se evaluaron tres opciones ampliamente utilizadas en aplicaciones industriales.

Tabla 2.6 Comparativa de gateways industriales

Modelo	Procesador	Interfaces	Conectividad	Ventajas principales
Siemens IoT2050	ARM Cortex-A8	RS232/RS485, Ethernet, GPIO	Ethernet, WiFi	Bajo costo, soporte con Node-RED, robustez
Advantech UNO-2271G	Intel Atom E3815	USB, HDMI, RS-232, LAN	Ethernet, 3G/4G (opcional)	Alto rendimiento, soporte industrial ampliado
WAGO PFC200	ARM Cortex-A8	RS-232, RS-485, CAN, Modbus-TCP	Ethernet, VPN	Integración directa con PLC y sistemas SCADA

Selección final: Se escogió el Siemens IoT2050 por su compatibilidad directa con Node-RED, bajo consumo de energía, y facilidad de integración en fases iniciales de pruebas y migración futura hacia producción.

2.4.2.2 Selección de sensores industriales

La elección de sensores industriales se basó en los parámetros críticos a medir: caudal, presión, temperatura y vibración. Se priorizó que los sensores cumplan con normativas IP67, salida de señal estandarizada (4-20 mA o Modbus RTU), y que puedan operar en ambientes húmedos y polvorientos.

Tabla 2.7 Sensores propuestos para implementación real

Variable	Sensor propuesto	Rango de medida	Salida	Normativa	Ventaja principal
Caudal	Siemens SITRANS F M MAG 5100W	0–500 m³/h	Modbus RTU	IP67	Alta precisión en líquidos conductivos
Presión	WIKA A-10	0–10 bar	4–20 mA	IP67	Robusto y preciso
Temperatura	PT100 con transmisor IFM TT2030	-50 a 150 °C	4–20 mA	IP68	Precisión y durabilidad
Vibración	Hansford Sensors HS-100 Series	0–25 mm/s RMS	IEPE/4-20 mA	IP67	Alta sensibilidad para análisis predictivo

2.4.2.3 Arquitectura de red

La red de comunicación debe garantizar disponibilidad, seguridad y baja latencia. Se plantea una arquitectura distribuida jerárquica con comunicación cableada e inalámbrica, segmentada en tres niveles:

Niveles de la arquitectura:

- Nivel de campo (sensores): Redes RS-485 y señales 4–20 mA.
- Nivel intermedio (gateway): Comunicación local con sensores y publicación vía MQTT.
- Nivel superior (SCADA y nube): Acceso remoto, procesamiento avanzado y almacenamiento histórico.

Tecnologías empleadas:

- MQTT para datos en tiempo real.
- Modbus RTU para comunicación en campo.
- VPN para acceso seguro a la red desde estaciones remotas.
- WiFi industrial o Ethernet para enlace entre gateway y SCADA local.

2.4.2.4 Plataforma Scada

Se evaluaron varias plataformas SCADA en base a sus capacidades de integración, escalabilidad y costos.

Tabla 2.8 Comparativa de plataformas SCADA

Plataforma	Licencia	Compatibilidad MQTT	Características clave
Ignition (Inductive Automation)	Comercial (licencia por tag)	Sí	Web-based, scripting en Python, módulo de alarmas
FactoryTalk View SE	Comercial (Rockwell)	Parcial	Integración con PLC Allen-Bradley
SCADA-LTS	Open Source	Sí	Gratuito, Java-based, personalizable

Selección final: Se opta por Ignition, gracias a su flexibilidad, integración con MQTT y Node-RED, y soporte técnico. Su entorno web permite monitoreo multiplataforma y expansión futura con módulos de inteligencia artificial.

2.4.2.5 Modulo de alertas

El sistema contará con un módulo de alertas avanzado capaz de detectar condiciones anómalas y notificar a los operadores en tiempo real a través de diferentes canales (dashboard, Telegram, email).

Tabla 2.9 Tipos de alertas y criterios

Variable	Rango Normal	Rango crítico
Caudal	150 – 350 m³/h	< 150 o > 350 m³/h
Presión	2 – 8 bar	< 2 o > 8 bar
Temperatura	30 – 60 °C	> 60 °C
Vibración	0 – 7 mm/s	> 7 mm/s

Características técnicas:

- Gestión de alarmas en Ignition (prioridades, deadbands, reconocimiento).
- Integración con Telegram Bot para mensajes automáticos.
- Registro histórico de eventos con timestamp.
- Panel de control para visualización de alertas activas y silenciadas.

Beneficios:

- Mejora en la respuesta ante fallos.
- Prevención de paradas no programadas.
- Trazabilidad y documentación para mantenimiento preventivo.

CAPÍTULO 3

3. RESULTADOS Y ANÁLISIS

Este capítulo presenta de forma detallada los resultados obtenidos durante la implementación y validación del sistema de monitoreo en tiempo real para pozos subterráneos, desarrollado como parte del presente trabajo de titulación. A través de simulaciones controladas, pruebas funcionales y validación de alertas críticas, se logró verificar el correcto funcionamiento de todos los módulos del sistema. La solución fue desarrollada utilizando tecnologías de código abierto, como FastAPI, Node-RED, MQTT y Telegram Bot, con un enfoque en escalabilidad, bajo costo y facilidad de integración con sistemas reales.

3.1 Resultados del análisis del sistema actual de agua potable

3.1.1 Identificación de eventos críticos ocurridos durante el año 2025

Con el fin de cuantificar la problemática real del sistema de agua potable del cantón Durán, se realizó un levantamiento y análisis de información pública correspondiente a los eventos críticos ocurridos durante el año 2025. Dichos eventos incluyen roturas de acueductos principales, fallas electromecánicas y anomalías mecánicas en los sistemas de bombeo, los cuales generaron suspensiones parciales o totales del servicio de agua potable en distintos sectores de la ciudad.

Tabla 3.1 Eventos críticos registrados en el sistema de agua potable del cantón Durán durante el año 2025

Tipo de evento Critico	Meses											
	Enero	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre	Suma total
Anomalía mecánica en pozo	1			1			2	3			1	8
Falla electromecánica crítica				1								1
Rotura de acueducto principal		2	1		1	1			1	4	3	13
Rotura de tubería secundaria			1								1	2
Suma total	1	2	2	2	1	1	2	3	1	4	5	24

Del análisis de la Tabla 3.1 se identificaron un total de 24 eventos críticos durante el año 2025. Las roturas del acueducto principal representan el tipo de evento más frecuente, con 13 ocurrencias, seguidas por las anomalías mecánicas en pozos de captación, con 8 eventos. Asimismo, se registraron fallas electromecánicas críticas y roturas de tuberías secundarias, aunque con menor frecuencia. Estos resultados evidencian una alta recurrencia de fallas estructurales y mecánicas en el sistema actual de abastecimiento de agua potable.

3.1.2 Análisis temporal de los eventos críticos

Con el objetivo de analizar el comportamiento temporal de los eventos críticos identificados, se elaboró una distribución mensual de las fallas ocurridas durante el año 2025, permitiendo identificar periodos de mayor recurrencia y posibles patrones de comportamiento del sistema.

Figura 3.1 Distribución mensual de eventos críticos registrados durante el año 2025

La Figura 3.1 muestra que los eventos críticos no se concentran en un único periodo del año, sino que se presentan de manera recurrente a lo largo de los meses. Se observa un incremento significativo de eventos durante los meses de noviembre y diciembre, principalmente asociados a roturas del acueducto principal. Adicionalmente, las anomalías mecánicas en pozos de captación se presentan de forma intermitente durante todo el año, lo cual sugiere condiciones operativas variables y ausencia de un monitoreo continuo de variables críticas del sistema.

3.1.3 Clasificación y frecuencia de eventos críticos

A partir del análisis de los eventos registrados, se realizó una clasificación según el tipo de falla, lo cual permitió determinar su frecuencia relativa. Del total de eventos críticos identificados, aproximadamente el 54% corresponde a roturas del acueducto principal, el 33% a anomalías mecánicas en pozos, el 8% a roturas de tuberías secundarias y el 4% a fallas electromecánicas críticas. Esta distribución evidencia que la mayor parte de las interrupciones del servicio se origina en fallas que podrían estar precedidas por variaciones anómalas de variables hidráulicas y mecánicas.

3.1.4 Detalle técnico y cronológico de los eventos críticos registrados

Con el fin de profundizar el análisis de los eventos críticos identificados y proporcionar trazabilidad técnica de cada incidencia registrada, se presenta a continuación el detalle cronológico de los eventos ocurridos durante el año 2025. Esta información incluye la fecha del evento, el tipo de falla, su descripción técnica y el impacto operativo generado en el sistema de distribución de agua potable.

Tabla 3.2 Detalle técnico de eventos críticos registrados en el sistema de agua potable del cantón Durán durante el año 2025

Fecha	Tipo de evento	Descripción técnica del problema	Impacto operativo / Consecuencia
24/12/2025	Rotura de acueducto principal	Rotura estructural en acueducto de 800 mm	Interrupción del servicio a 642 predios
17/12/2025	Rotura de acueducto principal	Rotura en acueducto de 800 mm en el sector Peñón del Río	Suspensión del servicio por 24 horas
15/12/2025	Rotura de acueducto principal	Rotura en acueducto de 800 mm en el sector La Herradura, asociada a variaciones de caudal y presión	Suspensión del servicio por 24 horas
10/12/2025	Anomalía mecánica en pozo	Detección de vibraciones y ruidos anómalos en el Pozo #5	Suspensión temporal para inspección del sistema de bombeo
3/12/2025	Rotura de tubería secundaria	Rotura en tubería de 400 mm	Suspensión del servicio en el sector Panorama por 24 horas
26/11/2025	Rotura de acueducto principal	Rotura en acueducto de 800 mm, tramo km 21 vía Durán–Yaguachi	Suspensión del servicio por 24 horas
20/11/2025	Rotura de acueducto principal	Rotura en acueducto de 800 mm, tramo km 21 vía Durán–Yaguachi	Suspensión del servicio por 24 horas
10/11/2025	Rotura de acueducto principal	Rotura en acueducto de 800 mm, tramo km 21 vía Durán–Yaguachi	Suspensión del servicio por 24 horas

2/11/2025	Rotura de acueducto principal	Rotura en acueducto de 800 mm en el sector Peñón del Río	Suspensión nocturna del servicio en el circuito norte de Durán
8/10/2025	Rotura de acueducto principal	Rotura en acueducto de 800 mm en el tramo El Chobo – La Herradura	Suspensión del servicio por 24 horas
11/9/2025	Anomalía mecánica en pozo	Vibraciones y ruidos anómalos en el Pozo #9	Suspensión temporal; afectación al circuito Recreo y sectores aledaños
5/9/2025	Anomalía mecánica en pozo	Vibraciones y ruidos anómalos en el Pozo #8	Suspensión temporal; afectación a varios sectores de la ciudad
2/9/2025	Anomalía mecánica en pozo	Vibraciones y ruidos anómalos en el Pozo #5	Suspensión temporal; afectación a Primavera 2 y Panorama
23/8/2025	Anomalía mecánica en pozo	Vibraciones y ruidos anómalos en el Pozo #8	Suspensión temporal; afectación a varios sectores de la ciudad
4/8/2025	Anomalía mecánica en pozo	Vibraciones y ruidos anómalos en el Pozo #8	Suspensión temporal; afectación a varios sectores de la ciudad
5/7/2025	Rotura de acueducto principal	Rotura en acueducto de 800 mm	Suspensión del servicio en toda la ciudad por 24 horas
4/6/2025	Rotura de acueducto principal	Rotura en acueducto de 800 mm	Suspensión del servicio en toda la ciudad por 24 horas
27/5/2025	Anomalía mecánica en pozo	Vibraciones y ruidos anómalos en el Pozo #7	Suspensión temporal; afectación a varios sectores de la ciudad
16/5/2025	Falla electromecánica crítica	Falla en bomba sumergible de 100 HP en el Pozo #10	Suspensión temporal para desmontaje y reemplazo de la bomba
28/4/2025	Rotura de acueducto principal	Rotura en acueducto de 800 mm en el sector Las Delicias	Suspensión del servicio en toda la ciudad por 48 horas
5/4/2025	Rotura de tubería secundaria	Rotura en tubería de 220 mm en Av. León Febres Cordero	Suspensión del servicio por 12 horas
22/3/2025	Rotura de acueducto principal	Rotura en acueducto de 800 mm	Suspensión del servicio en toda la ciudad por 48 horas
11/3/2025	Rotura de acueducto principal	Rotura en acueducto de 800 mm en el km 24 vía Durán–Yaguachi	Suspensión del servicio en toda la ciudad por 48 horas
2/1/2025	Anomalía mecánica en pozo	Problemas técnicos en el Pozo #5	Suspensión temporal; afectación a varios sectores de la ciudad

El análisis detallado de los eventos registrados evidencia que una parte significativa de las fallas estuvo asociada a roturas recurrentes del acueducto principal de 800 mm, así como a anomalías mecánicas en los pozos de captación, caracterizadas por vibraciones y ruidos anómalos. En varios casos, estos eventos generaron suspensiones prolongadas del servicio, afectando a sectores específicos e incluso a la totalidad de la ciudad. La recurrencia y naturaleza de estas fallas refuerzan la necesidad de implementar un sistema automatizado de

monitoreo en tiempo real que permita identificar condiciones anómalas antes de que evolucionen a eventos de alto impacto.

3.2 Relación entre eventos reales y variables críticas del sistema

El análisis de los eventos críticos ocurridos durante el año 2025 evidencia que una parte significativa de las fallas estuvo precedida por condiciones anómalas en variables operativas tales como variaciones de caudal y presión, así como vibraciones y ruidos mecánicos en los sistemas de bombeo. Estas variables coinciden directamente con las variables monitoreadas en el sistema automatizado propuesto en el presente trabajo.

En el sistema actual, la detección de estas anomalías se realiza de manera reactiva, una vez que el evento crítico ya ha generado una interrupción del servicio. En contraste, el sistema automatizado propuesto permite la adquisición continua de datos, el análisis en tiempo real y la generación automática de alertas cuando las variables superan umbrales predefinidos, posibilitando una detección temprana de fallas potenciales.

3.3 Resultados del sistema automatizado propuesto

3.3.1 Adquisición y transmisión de datos en tiempo real

El sistema automatizado propuesto fue validado mediante un entorno de simulación que reproduce el comportamiento de un sistema de monitoreo industrial para pozos de captación de agua potable. La adquisición de datos se realizó mediante la generación de valores simulados correspondientes a las variables críticas del sistema: caudal, presión, temperatura y vibración. Estos datos fueron publicados en tiempo real a través del protocolo MQTT, utilizando un broker local, lo que permitió evaluar la confiabilidad y continuidad de la transmisión de información.

La suscripción a los tópicos definidos se efectuó mediante Node-RED, garantizando la recepción continua de los datos sin pérdidas apreciables durante las pruebas realizadas. La elección del protocolo MQTT responde a su ligereza, bajo consumo de recursos y alta eficiencia en aplicaciones IoT, lo cual lo convierte

en una solución adecuada para sistemas de monitoreo distribuidos en infraestructura hidráulica.

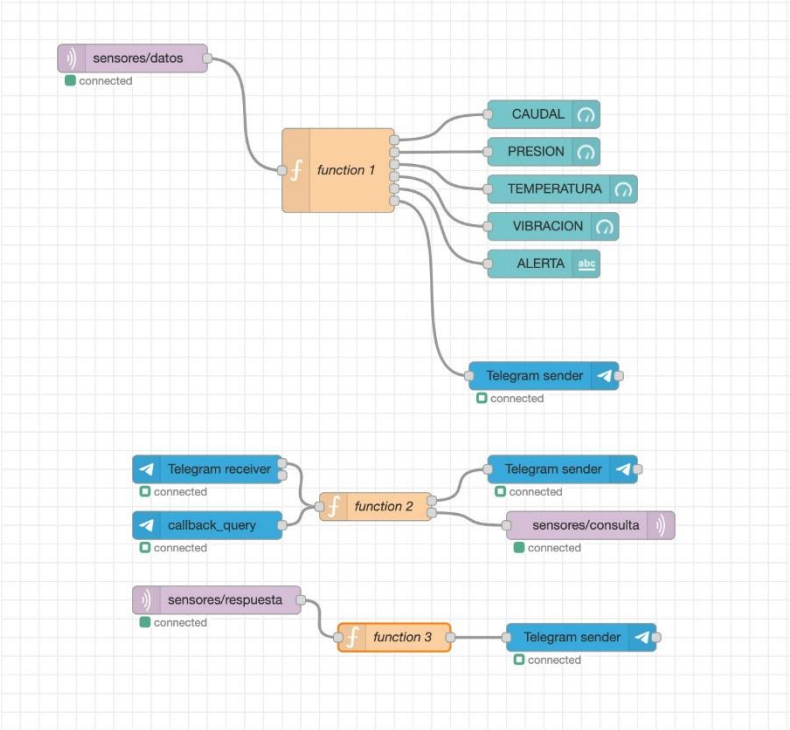
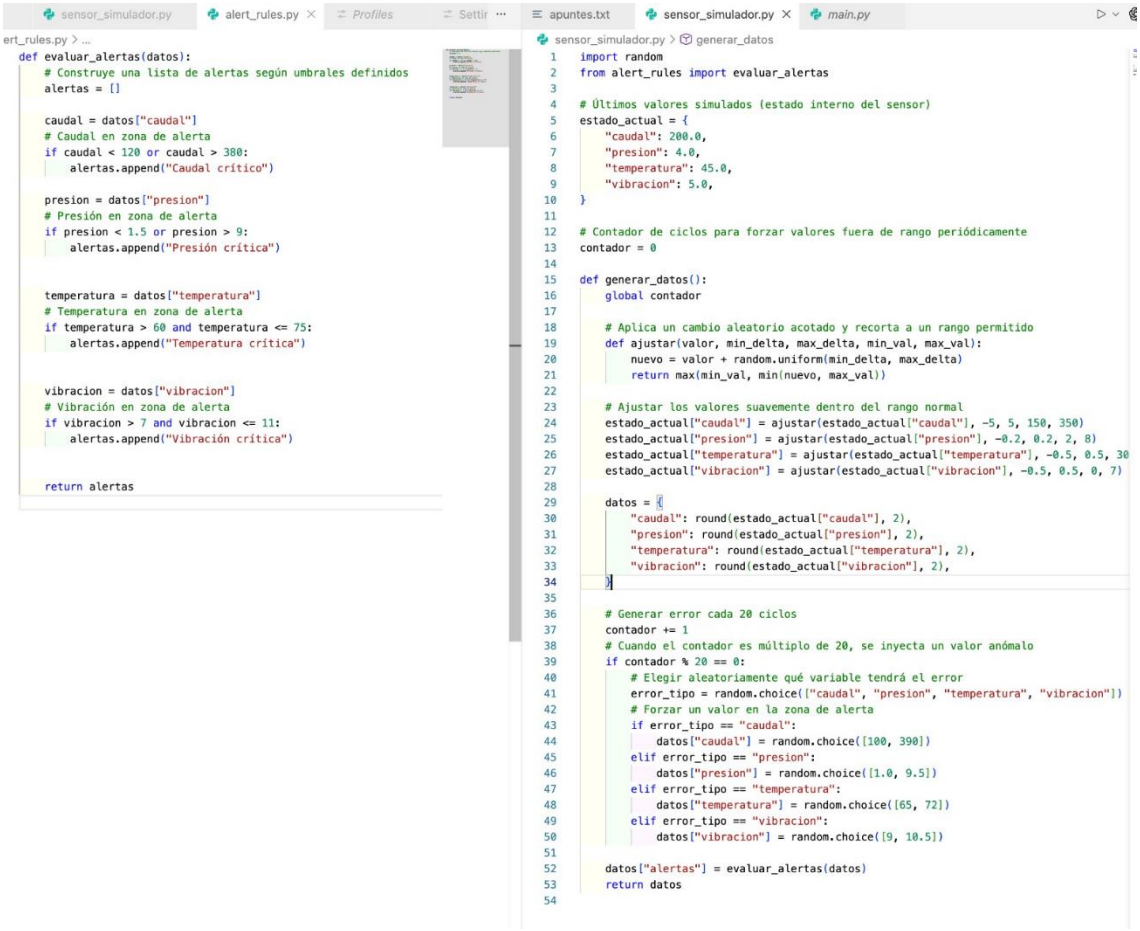


Figura 3.2 Adquisición y transmisión de datos en Node-RED

3.3.2 Procesamiento y evaluación de variables críticas

Se implementó una lógica de procesamiento que evalúa cada variable contra los rangos de operación y umbrales de alerta previamente definidos. Este procesamiento se realiza de forma automática e inmediata, permitiendo clasificar cada lectura como normal, de advertencia o crítica, dependiendo del valor registrado.

El sistema es capaz de detectar condiciones anómalas como incrementos súbitos de caudal, variaciones extremas de presión, aumentos en los niveles de vibración y elevaciones de temperatura del motor de la bomba. Esta evaluación en tiempo real constituye la base del mecanismo de detección temprana de fallas, ya que permite identificar desviaciones del comportamiento normal antes de que se produzcan eventos críticos en la infraestructura.



```python
# alert_rules.py
def evaluar_alertas(datos):
    # Construye una lista de alertas según umbrales definidos
    alertas = []

    caudal = datos["caudal"]
    # Caudal en zona de alerta
    if caudal < 120 or caudal > 380:
        alertas.append("Caudal crítico")

    presion = datos["presion"]
    # Presión en zona de alerta
    if presion < 1.5 or presion > 9:
        alertas.append("Presión crítica")


    temperatura = datos["temperatura"]
    # Temperatura en zona de alerta
    if temperatura > 60 and temperatura <= 75:
        alertas.append("Temperatura crítica")


    vibracion = datos["vibracion"]
    # Vibración en zona de alerta
    if vibracion > 7 and vibracion <= 11:
        alertas.append("Vibración crítica")


    return alertas
```

```python
# sensor_simulador.py
import random
from alert_rules import evaluar_alertas

# Últimos valores simulados (estado interno del sensor)
estado_actual = {
    "caudal": 200.0,
    "presion": 4.0,
    "temperatura": 45.0,
    "vibracion": 5.0,
}

# Contador de ciclos para forzar valores fuera de rango periódicamente
contador = 0

def generar_datos():
    global contador

    # Aplica un cambio aleatorio acotado y recorta a un rango permitido
    def ajustar(valor, min_delta, max_delta, min_val, max_val):
        nuevo = valor + random.uniform(min_delta, max_delta)
        return max(min_val, min(nuevo, max_val))

    # Ajustar los valores suavemente dentro del rango normal
    estado_actual["caudal"] = ajustar(estado_actual["caudal"], -5, 5, 150, 350)
    estado_actual["presion"] = ajustar(estado_actual["presion"], -0.2, 0.2, 2, 8)
    estado_actual["temperatura"] = ajustar(estado_actual["temperatura"], -0.5, 0.5, 30
    estado_actual["vibracion"] = ajustar(estado_actual["vibracion"], -0.5, 0.5, 0, 7)

    datos = {
        "caudal": round(estado_actual["caudal"], 2),
        "presion": round(estado_actual["presion"], 2),
        "temperatura": round(estado_actual["temperatura"], 2),
        "vibracion": round(estado_actual["vibracion"], 2),
    }

    # Generar error cada 20 ciclos
    contador += 1
    # Cuando el contador es múltiplo de 20, se inyecta un valor anómalo
    if contador % 20 == 0:
        # Elegir aleatoriamente qué variable tendrá el error
        error_tipo = random.choice(["caudal", "presion", "temperatura", "vibracion"])
        # Forzar un valor en la zona de alerta
        if error_tipo == "caudal":
            datos["caudal"] = random.choice([100, 390])
        elif error_tipo == "presion":
            datos["presion"] = random.choice([1.0, 9.5])
        elif error_tipo == "temperatura":
            datos["temperatura"] = random.choice([65, 72])
        elif error_tipo == "vibracion":
            datos["vibracion"] = random.choice([9, 10.5])

    datos["alertas"] = evaluar_alertas(datos)
    return datos
```

Figura 3.3 Lógica de generación de datos y alertas

3.3.3 Registro y almacenamiento histórico de datos

El sistema incorpora un módulo de almacenamiento persistente que registra cada lectura procesada en una base de datos PostgreSQL, mediante una API desarrollada con FastAPI. Cada registro almacena la fecha y hora de adquisición, los valores de las variables monitoreadas y las alertas generadas, garantizando la trazabilidad completa de la información.

Este mecanismo de registro histórico permite no solo conservar evidencia del funcionamiento del sistema, sino también habilitar el análisis posterior del comportamiento operativo de los pozos. A partir de estos datos, es posible identificar tendencias, patrones de variación y condiciones recurrentes que podrían derivar en fallas mecánicas o hidráulicas, aportando una base sólida para estrategias de mantenimiento preventivo.

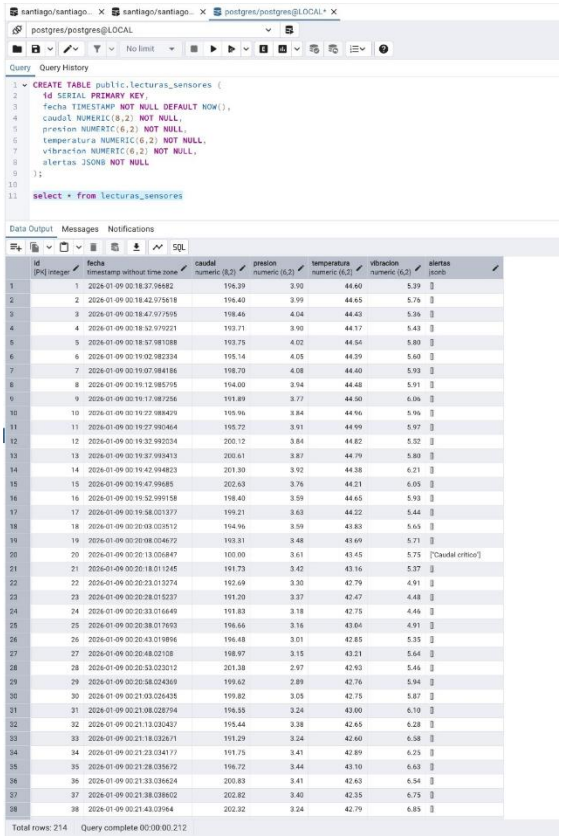


Figura 3.4 Base de datos para almacenamiento de mediciones tomadas por los sensores

3.3.4 Generación y notificación de alertas automáticas

Cuando el sistema detecta que una o más variables superan los umbrales establecidos, se genera automáticamente una alerta que es enviada al usuario mediante un bot de Telegram. Las notificaciones incluyen el valor medido, la unidad correspondiente y el tipo de condición detectada, proporcionando información clara y precisa para la toma de decisiones operativas.

Durante las pruebas realizadas, el sistema generó alertas por caudal crítico, presión crítica y vibración crítica, demostrando su capacidad para reaccionar de forma inmediata ante condiciones anómalas. Este mecanismo reduce significativamente el tiempo entre la detección del problema y la intervención del personal técnico, lo que representa una mejora sustancial frente a los sistemas tradicionales de supervisión reactiva.

Setup | On Start | **On Message** | On Stop

```
// Construir mensaje para Telegram
const mensaje = `⛽ Alerta detectada:
Caudal: ${payload.caudal} m³/h
Presión: ${payload.presion} bar
Temperatura: ${payload.temperatura} °C
Vibración: ${payload.vibracion} mm/s
⚠️ Alertas: ${payload.alertas.join(" | ")}`;

// Lista de usuarios a notificar (reemplaza con tus chat IDs reales)
const chatIds = [2024083617, 7637020432];

// Si hay alertas, armar los mensajes, si no, retornar null en esa salida
const mensajesTelegram = payload.alertas.length > 0
    ? chatIds.map(id => ({
        payload: {
            chatId: id,
            type: "message",
            content: mensaje
        }
    }))
    : null;

// Devolver salidas: las 5 primeras para los gauges y texto, la 6ta solo si hay alertas
return [
    { payload: payload.caudal },
    { payload: payload.presion },
    { payload: payload.temperatura },
    { payload: payload.vibracion },
    { payload: payload.alertas.join(" | ") },
    mensajesTelegram // si no hay alertas, será null y no se enviará mensaje
];
```

Figura 3.5 Envió de datos de NodeRed a Telegram

Figura 3.6 Alertas recibidas en Telegram

3.3.5 Visualización del registro histórico mediante interfaz de mensajería

Como complemento al dashboard web, se implementó una funcionalidad de consulta del registro histórico a través de la interfaz de mensajería Telegram. El sistema incorpora botones interactivos que permiten al usuario solicitar los últimos 10 registros almacenados de cada variable monitoreada, tales como caudal, presión, temperatura y vibración. Esta funcionalidad facilita el acceso remoto a la información histórica desde dispositivos móviles, permitiendo verificar rápidamente la evolución reciente del sistema sin necesidad de acceder directamente a la base de datos o a la plataforma web. La visualización de registros históricos mediante mensajería instantánea refuerza la capacidad de supervisión continua y aporta flexibilidad al proceso de monitoreo.

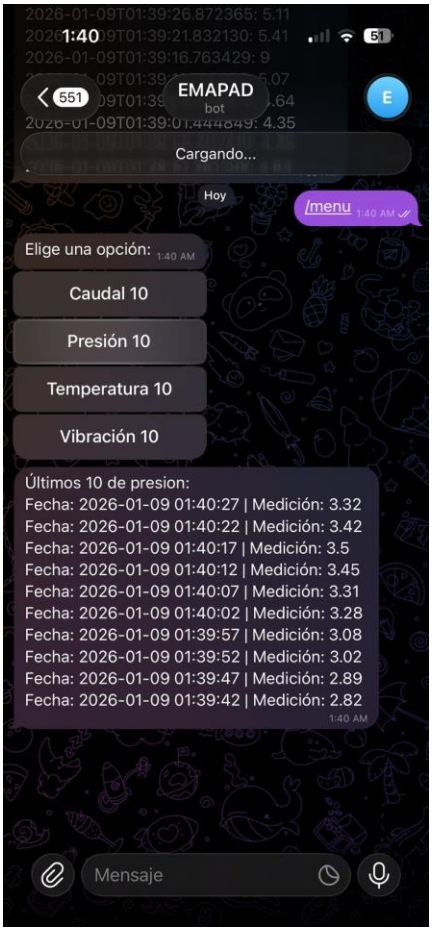


Figura 3.7 Visualización de registro histórico

3.3.6 Integración y validación global del sistema

La integración de los módulos de adquisición, procesamiento, almacenamiento, visualización y notificación permitió validar el funcionamiento integral del sistema automatizado propuesto. Las pruebas realizadas demuestran que el sistema opera de forma estable, coherente y continua, cumpliendo con los requisitos de monitoreo en tiempo real y registro histórico definidos en los objetivos del proyecto.

La correcta interacción entre MQTT, Node-RED, FastAPI, PostgreSQL, el dashboard web y la interfaz de Telegram evidencia la viabilidad técnica de la solución y su potencial aplicación en entornos reales de operación del sistema de agua potable.

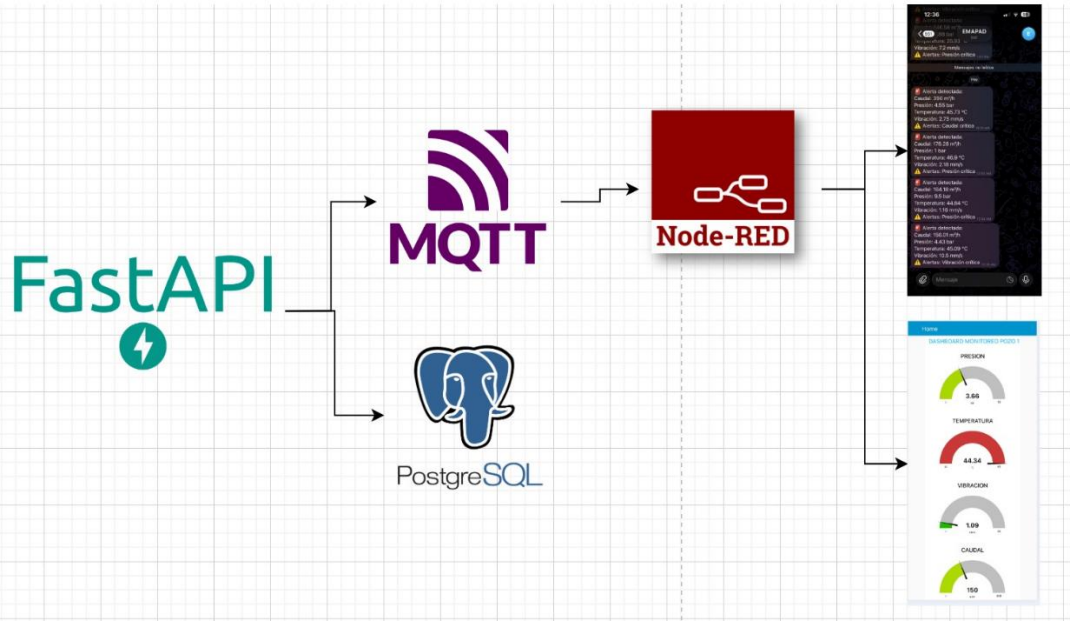


Figura 3.8 Integración y validación de datos

3.4 Análisis de costos

El presente análisis de costos se desarrolla con el objetivo de evaluar la viabilidad económica del sistema de monitoreo propuesto, considerando tanto el entorno simulado, en el cual se validó la arquitectura lógica del sistema, como una proyección de implementación real en pozos subterráneos del cantón Durán. Adicionalmente, se presenta una comparación económica con sistemas

tradicionales de monitoreo basados en controladores lógicos programables (PLC), ampliamente utilizados en la industria.

3.4.1 Análisis de costos del sistema simulado

El sistema simulado desarrollado en este trabajo no contempla la instalación de hardware físico en campo, por lo que su costo está asociado exclusivamente al desarrollo de software, configuración lógica, integración de plataformas y validación funcional. No obstante, para efectos de análisis económico, se realiza una estimación referencial del costo que tendría el desarrollo de este sistema en un contexto profesional.

Alcance del sistema simulado

El entorno simulado incluyó las siguientes actividades:

- Desarrollo de una API REST en FastAPI (Python) para la simulación de sensores industriales.
- Implementación de algoritmos para la generación de datos simulados de caudal, presión, temperatura y vibración.
- Configuración de comunicación mediante protocolo MQTT.
- Desarrollo de flujos de procesamiento y visualización en Node-RED.
- Creación de un dashboard de monitoreo en tiempo real.
- Implementación de lógica de alarmas automáticas.
- Integración de notificaciones mediante Telegram Bot.
- Configuración de base de datos para el almacenamiento histórico de datos.
- Pruebas funcionales y validación del sistema.

Todas las herramientas utilizadas corresponden a software de código abierto, por lo que no se incurrió en costos de licenciamiento.

3.4.2 Análisis de costos de la implementación real del sistema

Una vez validado el funcionamiento del sistema en el entorno simulado, se presenta una estimación referencial de los costos asociados a su implementación física en un pozo subterráneo, manteniendo la misma arquitectura lógica validada.

Componentes considerados para la implementación real.

Tabla 3.3 Análisis de costos de la implementación real del sistema

Componente	Costo estimado (USD)
Sensor de caudal	$ 1,500.00
Sensor de presión	$ 180.00
Sensor de temperatura	$ 120.00
Sensor de vibración	$ 900.00
Gateway industrial	$ 350.00
Gabinete y protecciones	$ 250.00
Cableado industrial	$ 150.00
Router industrial / conectividad	$ 180.00
Instalación y puesta en marcha	$ 1,200.00
Sub. Total $	$ 4,830.00
I.V.A. 15%	$ 724.50
Total $	**$ 5,554.50**

Este valor corresponde a una solución completa de monitoreo en tiempo real, sin incluir funciones de control activo sobre las bombas.

3.4.3 Comparación económica con sistemas tradicionales basados en PLC

Los sistemas tradicionales de monitoreo y control industrial para pozos subterráneos se basan generalmente en el uso de PLC industriales, módulos de entradas y salidas, pantallas HMI y plataformas SCADA con licencias comerciales. Costo típico de un sistema basado en PLC

Tabla 3.4 Comparación económica con sistemas tradicionales basados en PLC

Componente	Costo estimado (USD)
PLC industrial	$ 2,500.00
Módulos de entradas/salidas	$ 500.00
Pantalla HMI	$ 900.00
Licencia SCADA	$ 3,000.00
Sensores industriales	$ 1,986.00
Gabinete y cableado	$ 1,259.00
Ingeniería e integración	$ 1,500.00
Sub. Total $	$ 11,645.00
I.V.A. 15%	$ 1,746.75
Total $	**$ 13,391.75**

El análisis realizado permite concluir que el sistema propuesto es económicamente viable y competitivo. El desarrollo del sistema simulado requiere una inversión moderada asociada al desarrollo de software, lo que permite validar completamente la solución antes de su implementación. Por otro lado, la implementación real del sistema IoT propuesto presenta un costo significativamente menor en comparación con los sistemas tradicionales basados en PLC, manteniendo funcionalidades clave como el monitoreo en tiempo real, la generación de alarmas automáticas y el registro histórico de datos.

En consecuencia, el sistema propuesto constituye una alternativa de bajo costo y alta eficiencia para la supervisión de pozos subterráneos, especialmente adecuada para instituciones públicas y proyectos municipales donde la optimización de recursos económicos es un factor determinante.

CAPÍTULO 4

4. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

El presente trabajo ha permitido desarrollar y validar un sistema de monitoreo en tiempo real para pozos subterráneos en el cantón Durán, utilizando tecnologías abiertas y accesibles como FastAPI, MQTT, Node-RED y la plataforma de mensajería Telegram. La solución diseñada cumplió con los objetivos establecidos al inicio del proyecto, orientados a generar un sistema funcional, eficiente y económico para el monitoreo y gestión de variables críticas que impactan el desempeño y la seguridad de los pozos.

- Uno de los principales logros fue la implementación de un modelo de simulación realista, que reproduce el comportamiento dinámico de parámetros fundamentales como caudal, presión, temperatura y vibración, con la capacidad de introducir eventos críticos para validar la detección oportuna de alertas. Esto evidenció la eficacia del sistema para captar condiciones anómalas y notificar al operador de forma inmediata, lo que es crucial para la prevención de fallas y la toma rápida de decisiones.
- La integración con Node-RED facilitó la visualización dinámica de los datos, aportando una interfaz gráfica intuitiva para monitoreo en tiempo real, lo cual mejora significativamente la experiencia del usuario y permite un seguimiento continuo del estado del pozo. Además, la conexión con Telegram para envío de alertas garantizó una comunicación ágil y efectiva, permitiendo que el personal responsable reciba notificaciones instantáneas en cualquier lugar y momento.
- El análisis económico demostró que la solución es viable desde el punto de vista financiero, considerando que el uso de software libre y hardware de bajo costo reduce significativamente la inversión inicial. Esto representa una ventaja competitiva frente a sistemas comerciales convencionales, que suelen requerir presupuestos elevados y no siempre se adaptan a las necesidades específicas del cliente.

- En términos técnicos, el desarrollo mostró robustez en la arquitectura modular del sistema, que facilita la escalabilidad y futuras integraciones con sensores físicos y otras plataformas. Sin embargo, se identificaron ciertas limitaciones, como la dependencia de un servidor local para la gestión de MQTT y Node-RED, lo cual puede afectar la disponibilidad y resiliencia del sistema en caso de fallas de hardware o conectividad.

4.2 Recomendaciones

- Para garantizar la operatividad y la confiabilidad del sistema de monitoreo en tiempo real, es fundamental seleccionar hardware industrial robusto y certificado. El uso de Gateway como el SIMATIC IoT2050 o dispositivos equivalentes permitirá la integración eficiente de sensores con plataformas de supervisión remota, asegurando compatibilidad con protocolos de comunicación ampliamente utilizados en la industria. Asimismo, la elección de sensores físicos de caudal, presión, vibración y temperatura debe considerar factores como resistencia a ambientes húmedos, durabilidad y protección contra polvo y agua (IP67), lo que asegura un funcionamiento continuo en condiciones adversas. Es recomendable incorporar redundancia en fuentes de alimentación y comunicación, de manera que el sistema pueda seguir operando incluso ante fallos parciales, garantizando la continuidad del monitoreo y reduciendo el riesgo de interrupciones inesperadas.
- La transmisión de datos en un sistema de monitoreo requiere protocolos confiables que aseguren tanto la velocidad como la seguridad de la información. Para la comunicación local entre sensores y Gateway, se recomienda el uso de Modbus TCP/RTU, dado que es un estándar industrial ampliamente probado y compatible con equipos existentes. Para la transmisión remota hacia la nube o plataformas externas, el protocolo MQTT resulta ideal por su ligereza y bajo consumo de ancho de banda, lo que permite un monitoreo eficiente incluso en redes con limitaciones. Además, es imprescindible implementar medidas de ciberseguridad, como el cifrado TLS/SSL en las comunicaciones y la segmentación de la red de monitoreo, con el fin de proteger el sistema contra accesos no autorizados y garantizar la integridad de los datos. Estas prácticas no solo fortalecen la confiabilidad del

sistema, sino que también aseguran que la información crítica esté disponible en todo momento para la toma de decisiones.

- La implementación de un sistema automatizado de monitoreo requiere que el personal de la Empresa Municipal de Alcantarillado y Agua Potable de Durán (EMAPAD-EP) esté debidamente capacitado en el uso de las herramientas tecnológicas. Es recomendable realizar programas de entrenamiento orientados al manejo de dashboard en Node-RED, la interpretación de alarmas automáticas y la gestión de datos históricos para identificar patrones de fallos recurrentes. Asimismo, se debe establecer un protocolo de mantenimiento predictivo, basado en el análisis de variables como vibración y presión, que permita anticipar fallos mecánicos en las bombas y reducir los tiempos de inactividad. Finalmente, es importante documentar la arquitectura del sistema y diseñar un plan de escalabilidad que facilite su expansión hacia más pozos o su integración con sistemas SCADA existentes, asegurando que la solución pueda evolucionar en función de las necesidades futuras de la empresa y de la población.

APÉNDICES

Apéndice A: Detalle de costos de materiales y dispositivos para la ejecución del proyecto

PROFORMA

SOLUCIÓN INDUSTRIAL

CONSTRUTARGET S.A.

Construtarget.industrial@live.com

R. U. C. 0992646756001

Pascuales Coop.Assad Bucaram Mz. EF 283 Sl. 3 Telf.0982906683

Lugar y Fecha: Guayaquil 10 de diciembre del 2025.

Cliente: Mabe S.A.

Dirección: Km 14.5 vía a Daule

C.I. / R.U.C. 0991321020001 Telf.: 2160500

O/C

Cantidad	Descripción	V. Unitario	V. Total
1	Sensor de caudal	$1,500.00	$1,500.00
1	Sensor de presión	$180.00	$180.00
1	Sensor de temperatura	$120.00	$120.00
1	Sensor de vibración	$900.00	$900.00
1	Gateway industrial	$350.00	$350.00
1	Gabinete y protecciones	$250.00	$250.00
1	Cableado industrial	$150.00	$150.00
1	Router industrial / conectividad	$180.00	$180.00
1	Instalación y puesta en marcha	$1,200.00	$1,200.00
Nota:		Sub. Total $	$4,830.00
		I.V.A. 15%	$724.50
		Total $	$5,554.50

PROFORMA

SOLUCIÓN INDUSTRIAL

CONSTRUTARGET S.A.

Construtarget.industrial@live.com

R. U. C. 0992646756001

Pascuales Coop.Assad Bucaram Mz. EF 283 Sl. 3 Telf.0982906683

Lugar y Fecha: Guayaquil 10 de diciembre del 2025.

Cliente: Mabe S.A.

Dirección: Km 14.5 vía a Daule

C.I. / R.U.C. 0991321020001 Telf.: 2160500

O/C

Cantidad	Descripción	V. Unitario	V. Total
1	PLC industrial	$2,500.00	$2,500.00
1	Módulos de entradas/salidas	$500.00	$500.00
1	Pantalla HMI	$900.00	$900.00
1	Licencia SCADA	$3,000.00	$3,000.00
1	Sensores industriales	$1,986.00	$1,986.00
1	Gabinete y cableado	$1,259.00	$1,259.00
1	Ingeniería e integración	$1,500.00	$1,500.00
Nota:		Sub. Total $	$11,645.00
		I.V.A. 15%	$1,746.75
		Total $	$13,391.75

Apéndice B: Desarrollo de la programación aplicada al sistema de monitoreo en tiempo real

Fast-Api

Main.py

```
from fastapi import FastAPI
import asyncio
import json
from decimal import Decimal
from sensor_simulador import generar_datos
import paho.mqtt.client as mqtt
import psycopg2
from psycopg2.extras import Json, RealDictCursor

app = FastAPI()

data_cache = {}
mqtt_client = mqtt.Client()
db_conn = None
DB_CONFIG = {
    "host": "localhost",
    "port": 5432,
    "dbname": "postgres",
    "user": "postgres",
}
TOPIC_CONSULTA = "sensores/consulta"
TOPIC_RESPUESTA = "sensores/respuesta"
CAMPO_PERMITIDOS = {"caudal", "presion", "temperatura", "vibracion"}

def consultar_ultimos(conn, campo, limite):
    try:
        limite_int = int(limite)
    except (TypeError, ValueError):
        limite_int = 10
    limite = max(1, min(limite_int, 100))
```

```
    with conn.cursor(cursor_factory=RealDictCursor) as cursor:
        cursor.execute(
            f"""
            SELECT fecha, {campo}
            FROM public.lecturas_sensores
            ORDER BY fecha DESC
            LIMIT %s
            """,
            (limite,),
        )
        filas = cursor.fetchall()
    for fila in filas:
        if "fecha" in fila and hasattr(fila["fecha"], "isoformat"):
            fila["fecha"] = fila["fecha"].isoformat()
        valor = fila.get(campo)
        if isinstance(valor, Decimal):
            fila[campo] = float(valor)
    return filas, limite

@app.on_event("startup")
async def start_simulation():
    # Conexión al broker MQTT
    mqtt_client.connect("localhost", 1883, 60)
    mqtt_client.subscribe(TOPIC_CONSULTA)
    mqtt_client.loop_start()

    # Conexión a PostgreSQL (sin contraseña)
    global db_conn
    db_conn = psycopg2.connect(**DB_CONFIG)
    db_conn.autocommit = True

    def guardar_en_db(datos):
```

```
    conn = psycopg2.connect(**DB_CONFIG)
    conn.autocommit = True
    with conn.cursor() as cursor:
        cursor.execute(
            """
            INSERT INTO public.lecturas_sensores
                (caudal, presion, temperatura, vibracion, alertas)
            VALUES (%s, %s, %s, %s, %s)
            """,
            (
                datos["caudal"],
                datos["presion"],
                datos["temperatura"],
                datos["vibracion"],
                Json(datos["alertas"]),
            ),
        )
    conn.close()

def on_mqtt_message(client, userdata, msg):
    if msg.topic != TOPIC_CONSULTA:
        return
    try:
        payload = json.loads(msg.payload.decode("utf-8"))
    except Exception:
        client.publish(TOPIC_RESPUESTA, json.dumps({"error": "JSON invalido"}))
        return

    campo = payload.get("campo")
    limite = payload.get("limite", 10)
    chat_id = payload.get("chatId")
    request_id = payload.get("requestId")
```

```
    if campo not in CAMPO_PERMITIDOS:
        respuesta = {"error": "Campo no permitido"}
        if chat_id is not None:
            respuesta["chatId"] = chat_id
        if request_id is not None:
            respuesta["requestId"] = request_id
        client.publish(TOPIC_RESPUESTA, json.dumps(respuesta))
        return

    try:
        conn = psycopg2.connect(**DB_CONFIG)
        conn.autocommit = True
        datos, limite = consultar_ultimos(conn, campo, limite)
        conn.close()
        respuesta = {"campo": campo, "limite": limite, "datos": datos}
    except Exception:
        respuesta = {"error": "Error consultando base de datos"}

    if chat_id is not None:
        respuesta["chatId"] = chat_id
    if request_id is not None:
        respuesta["requestId"] = request_id
    client.publish(TOPIC_RESPUESTA, json.dumps(respuesta))

mqtt_client.on_message = on_mqtt_message

async def simulate():
    while True:
        datos = generar_datos()
        data_cache["datos"] = datos
        mqtt_client.publish("sensores/datos", json.dumps(datos))  # publicar en MQTT
        # Guardar en la base sin bloquear el loop
```

```
            try:
                await asyncio.to_thread(guardar_en_db, datos)
                print("Guardado en DB:", datos)
            except Exception:
                pass
            await asyncio.sleep(5)

    asyncio.create_task(simulate())

@app.get("/datos-simulados")
async def obtener_datos():
    return data_cache.get("datos", {})

@app.get("/ultimos")
async def obtener_ultimos(campo: str, limite: int = 10):
    if db_conn is None:
        return {"error": "Base de datos no conectada"}

    if campo not in CAMPO_PERMITIDOS:
        return {"error": "Campo no permitido"}

    filas, limite = consultar_ultimos(db_conn, campo, limite)

    return {"campo": campo, "limite": limite, "datos": filas}
```

sensor_simulador.py

```
import random
from alert_rules import evaluar_alertas

# Últimos valores simulados (estado interno del sensor)
estado_actual = {
    "caudal": 200.0,
    "presion": 4.0,
    "temperatura": 45.0,
    "vibracion": 5.0,
}

# Contador de ciclos para forzar valores fuera de rango periódicamente
contador = 0

def generar_datos():
    global contador

    # Aplica un cambio aleatorio acotado y recorta a un rango permitido
    def ajustar(valor, min_delta, max_delta, min_val, max_val):
        nuevo = valor + random.uniform(min_delta, max_delta)
        return max(min_val, min(nuevo, max_val))

    # Ajustar los valores suavemente dentro del rango normal
    estado_actual["caudal"] = ajustar(estado_actual["caudal"], -5, 5, 150, 350)
    estado_actual["presion"] = ajustar(estado_actual["presion"], -0.2, 0.2, 2, 8)
    estado_actual["temperatura"] = ajustar(estado_actual["temperatura"], -0.5, 0.5, 30,
60)
    estado_actual["vibracion"] = ajustar(estado_actual["vibracion"], -0.5, 0.5, 0, 7)

    datos = {
        "caudal": round(estado_actual["caudal"], 2),
```

```
    "presion": round(estado_actual["presion"], 2),
    "temperatura": round(estado_actual["temperatura"], 2),
    "vibracion": round(estado_actual["vibracion"], 2),
}

# Generar error cada 20 ciclos
contador += 1
# Cuando el contador es múltiplo de 20, se inyecta un valor anómalo
if contador % 20 == 0:
    # Elegir aleatoriamente qué variable tendrá el error
    error_tipo = random.choice(["caudal", "presion", "temperatura", "vibracion"])
    # Forzar un valor en la zona de alerta
    if error_tipo == "caudal":
        datos["caudal"] = random.choice([100, 390])
    elif error_tipo == "presion":
        datos["presion"] = random.choice([1.0, 9.5])
    elif error_tipo == "temperatura":
        datos["temperatura"] = random.choice([65, 72])
    elif error_tipo == "vibracion":
        datos["vibracion"] = random.choice([9, 10.5])

datos["alertas"] = evaluar_alertas(datos)
return datos
```

alert_rules.py

```
import random
from alert_rules import evaluar_alertas

# Últimos valores simulados (estado interno del sensor)
estado_actual = {
    "caudal": 200.0,
    "presion": 4.0,
    "temperatura": 45.0,
    "vibracion": 5.0,
}

# Contador de ciclos para forzar valores fuera de rango periódicamente
contador = 0

def generar_datos():
    global contador

    # Aplica un cambio aleatorio acotado y recorta a un rango permitido
    def ajustar(valor, min_delta, max_delta, min_val, max_val):
        nuevo = valor + random.uniform(min_delta, max_delta)
        return max(min_val, min(nuevo, max_val))

    # Ajustar los valores suavemente dentro del rango normal
    estado_actual["caudal"] = ajustar(estado_actual["caudal"], -5, 5, 150, 350)
    estado_actual["presion"] = ajustar(estado_actual["presion"], -0.2, 0.2, 2, 8)
    estado_actual["temperatura"] = ajustar(estado_actual["temperatura"], -0.5, 0.5, 30,
60)
    estado_actual["vibracion"] = ajustar(estado_actual["vibracion"], -0.5, 0.5, 0, 7)

    datos = {
        "caudal": round(estado_actual["caudal"], 2),
```

```
    "presion": round(estado_actual["presion"], 2),
    "temperatura": round(estado_actual["temperatura"], 2),
    "vibracion": round(estado_actual["vibracion"], 2),
}

# Generar error cada 20 ciclos
contador += 1
# Cuando el contador es múltiplo de 20, se inyecta un valor anómalo
if contador % 20 == 0:
    # Elegir aleatoriamente qué variable tendrá el error
    error_tipo = random.choice(["caudal", "presion", "temperatura", "vibracion"])
    # Forzar un valor en la zona de alerta
    if error_tipo == "caudal":
        datos["caudal"] = random.choice([100, 390])
    elif error_tipo == "presion":
        datos["presion"] = random.choice([1.0, 9.5])
    elif error_tipo == "temperatura":
        datos["temperatura"] = random.choice([65, 72])
    elif error_tipo == "vibracion":
        datos["vibracion"] = random.choice([9, 10.5])

datos["alertas"] = evaluar_alertas(datos)
return datos
```

node-red

bloque function 1

```
const payload = msg.payload;

// Construir mensaje para Telegram
const mensaje = ` 🚨 Alerta detectada:
Caudal: ${payload.caudal} m³/h
Presión: ${payload.presion} bar
Temperatura: ${payload.temperatura} °C
Vibración: ${payload.vibracion} mm/s
⚠️ Alertas: ${payload.alertas.join(" | ")}`;

// Lista de usuarios a notificar (reemplaza con tus chat IDs reales)
const chatIds = [2024083617, 7637020432];

// Si hay alertas, armar los mensajes, si no, retornar null en esa salida
const mensajesTelegram = payload.alertas.length > 0
    ? chatIds.map(id => ({
        payload: {
            chatId: id,
            type: "message",
            content: mensaje
        }
    }))
    : null;

// Devolver salidas: las 5 primeras para los gauges y texto, la 6ta solo si hay alertas
return [
    { payload: payload.caudal },
    { payload: payload.presion },
    { payload: payload.temperatura },
```

```
    { payload: payload.vibracion },
    { payload: payload.alertas.join(" | ") },
    mensajesTelegram // si no hay alertas, será null y no se enviará mensaje
];
```

node-red

bloque function 2

```
const text = msg.payload.content || "";
const chatId =
  msg.payload.chatId ||
  (msg.payload.from && msg.payload.from.id) ||
  (msg.payload.message && msg.payload.message.chat &&
msg.payload.message.chat.id);
const data = msg.payload.data || text; // si no hay callback_data, usa el texto

node.warn({ etapa: "entrada", text, data, chatId });

if (chatId) {
  flow.set("lastChatId", chatId);
}

// 1) Si es /menu o /start, enviar menú al Telegram sender (salida 1)
if (text === "/menu" || text === "/start") {
  msg.payload = {
    chatId,
    type: "message",
    content: "Elige una opción:",
    options: {
      reply_markup: {
        inline_keyboard: [
          [{ text: "Caudal 10", callback_data: "caudal10" }],
          [{ text: "Presión 10", callback_data: "presion10" }],
          [{ text: "Temperatura 10", callback_data: "temp10" }],
          [{ text: "Vibración 10", callback_data: "vib10" }]
        ]
      }
```

```
    }
  };
  node.warn({ etapa: "envia_menu", chatId });
  return [msg, null];
}

// 2) Si viene un callback, mandar request por MQTT (salida 2)
if (data) {
  let campo = null;
  if (data === "caudal10") campo = "caudal";
  if (data === "presion10") campo = "presion";
  if (data === "temp10") campo = "temperatura";
  if (data === "vib10") campo = "vibracion";

  if (!campo) return null;

  msg.topic = "sensores/consulta";
  msg.payload = JSON.stringify({
    campo,
    limite: 10,
    chatId
  });

  node.warn({ etapa: "envia_mqtt", topic: msg.topic, payload: msg.payload });
  return [null, msg];
}

return null;
```

node-red

bloque function 3

```
let payload = msg.payload;

if (Buffer.isBuffer(payload)) {
  payload = payload.toString("utf8");
}
if (typeof payload === "string") {
  payload = JSON.parse(payload);
}

node.warn({ etapa: "mqtt_respuesta_raw", payload });

if (!payload.chatId) {
  payload.chatId = flow.get("lastChatId");
}
if (!payload.chatId) {
  node.warn({ etapa: "sin_chatId" });
  return null;
}

let content = "";
if (payload.error) {
  content = Error: ${payload.error};
} else {
  const fmtFecha = (iso) => {
    if (!iso) return "sin fecha";
    return iso.replace("T", " ").split(".")[0]; // YYYY-MM-DD HH:MM:SS
  };

  content = Últimos ${payload.limite} de ${payload.campo}:\n +
    payload.datos
```

```
    .map(d => Fecha: ${fmtFecha(d.fecha)} | Medición: ${d[payload.campo]})
    .join("\n");
}

msg.payload = {
  chatId: payload.chatId,
  type: "message",
  content
};

node.warn({ etapa: "envia_telegram", chatId: payload.chatId, content });
return msg;
```

5. REFERENCIAS

[1] V. Türk, «Día Mundial del Agua 22 de marzo de 2025,» 22 Marzo 2025. [En línea]. Available: https://www.ohchr.org/es/statements-and-speeches/2024/03/water-should-be-used-tool-peace-says-turk.

[2] AP, «Durán: ¿quiénes son los dueños del agua?,» 12 Septiembre 2023. [En línea]. Available: https://planv.com.ec/investigacion/duran-quienes-son-duenos-del-agua/.

[3] EMAPAD-EP, «Transformamos cada gota de agua en bienestar,» 23 Enero 2024. [En línea]. Available: http://www.emapad.gob.ec/home/Pac.

[4] C. Mendoza, «Oficio Nro. EMAPAD-EP-GG-0059-2022-CML,» EMAPAD-EP, DURAN, 2022.

[5] R. T. A. S. O. M. C. O.´. C. M. M. B. Esli Castellanos Berjan, «Water quality monitoring system in supply sources,» *DIFU100CI,* vol. 15, nº 3, pp. 27-32, 2021.

[6] A. Garcia, «PRIMICIAS,» 13 MAYO 2025. [En línea]. Available: https://www.primicias.ec/noticias/sociedad/agua-potabe-duran-deudas-roturas-acueducto/.

[7] C. Haller, «SIEMENS,» 04 MARZO 2020. [En línea]. Available: https://assets.new.siemens.com/siemens/assets/api/uuid:1ec6a7db-98cd-40bf-9901-b4b4eeca4b71/DIPR202003025820EN.pdf. [Último acceso: 2 ABRIL 2025].

[8] C. A. T. Vanesa G. Cadin, «Revisión de la utilización de QUIC en el protocolo HTTP/3,» *Revista de Informes Científicos y Técnicos,* vol. 15, nº 5, pp. 76-80, 2023.

[9] R. P. H. G. Torres Ventura, «Rendimiento para la interoperabilidad entre Raspberry pi, ESP8266 y PLC con Node-RED para el IIoT,» *INGENIUS,* vol. 1, nº 29, pp. 92-93, 2023.

[10] A. Garcia, «Sociedad,» PRIMICIAS, 13 Abril 2025. [En línea]. Available: https://www.primicias.ec/noticias/sociedad/duran-agua-potable-rio-daule-hogares-servicio/. [Último acceso: 13 Mayo 2025].

[11] «DESARROLLO DE UN SISTEMA SCADA PARA MONITOREO DE DESARROLLO DE UN SISTEMA SCADA PARA MONITOREO DE LABORATORIO FARMACEUTICO BASADO EN TECNOLOGIA IOT,» Universidad Politecnica Salesiana , Guayaquil, 2024.

[12] A. J. B. P. ASHOK JAIN, WATER SUPPLY ENGINEERING, Daryaganj: LAXMI PUBLICATIONS , 2005.

[13] R. Instruments, «Riels Instruments,» Virtual Expo, 3 Febrero 2025. [En línea]. Available: https://www.directindustry.es/prod/riels-instruments/product-70609-2664198.html. [Último acceso: 13 Mayo 2025].

[14] M. G. M. M. M. A. y. M. A. A. Al-Fuqaha, «nternet de las cosas: Una encuesta sobre tecnologías, protocolos y aplicaciones facilitadoras,» *Communications Surveys & Tutorials,* vol. XVII, nº 4, pp. 2347-2376, 2015.

[15] C. M. Castillo Estrada, K. Cancino Villatoro, V. Benavides García y A. de la Cruz Vázquez, «Diseño de un Sistema web para el control de Curriculum Vitae Electrónico de personal docente basado en una arquitectura orientada a servicios (API REST),» *DIALNET,* vol. X, nº 20, pp. 28-42, 2022.

[16] Node-RED, «Node-RED,» OpenJS Foundation, 20 Junio 2024. [En línea]. Available: https://nodered.org/. [Último acceso: 13 Abril 2025].

[17] N. L. T. Valdés, «IMPLEMENTACIÓN DE UN SISTEMA DE ALARMA Y ALERTA TEMPRANA DE ALUVIONES EN LA GESTIÓN DEL RIESGO DE DESASTRES DE ORGANISMOS GUBERNAMENTALES CHILENOS,» UNIVERSIDAD DE CONCEPCIÓN, CONCEPCIÓN, 2024.

[18] V. F. R. CASTILLO, «GESTIÓN DE ALARMAS Y MINIMIZACIÓN DE PARADAS ESPURIAS EN AIR LIQUIDE CHILE S.A.,» Universidad de Chile, Santiago, 2024.

[19] V. Parra, «My tips,» My tips, 2 Enero 2020. [En línea]. Available: https://www.mytips.es/mejorando-los-kpis-de-alarmas-de-forma-rapida-y-sencilla/. [Último acceso: 13 Mayo 2025].

[20] M. Brachetta, O. León y J. Monetti, «Desarrollo del back-end para un entorno ubicuo de enseñanza,» *Congreso de Tecnología en Educación & Educación en Tecnología,* vol. XVIII, nº 3, pp. 19-25, 2022.

[21] R. Y. Endra, Y. Aprilinda, Y. Y. Dharmawan y W. Ramadhan, «Analisis Perbandingan Bahasa Pemrograman PHP Laravel dengan PHP Native pada Pengembangan Website,» *Terakreditasi SINTA Peringkat 5,* vol. 11, nº 1, pp. 48-55, 2022.

[22] B. Lubanovic, FastAPI, Sebastopol: O'Reilly Media, 2023.

[23] A. Serna, F. Ros y J. Rico, Guía Práctica de Sensores, España: Creaciones Copyright, 2010.

[24] F. Hernández, «Análisis de la Tecnología de Mensajería MQTT en el contexto del Internet de las Cosas: un caso de estudio,» E.T.S. Ingeniería Informática, Málaga, 2024.

[25] F. G. E. B. L. J. Q. A. V. Quezada José Carlos, «Diseño e implementación de un sistema de control y monitoreo,» *Ingeniería Investigación y Tecnología,* vol. XV, nº 1, pp. 46-49, 2014.