

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
SEDE VIÑA DEL MAR – JOSÉ MIGUEL CARRERA

PROYECTO DE SISTEMA DE MONITOREO PARA SILOS DE ALIMENTO DE SALMÓN

Trabajo de Titulación para optar al Título de
Ingeniero de Ejecución en CONTROL E
INSTRUMENTACIÓN INDUSTRIAL

Alumno:

Juan Pablo Berríos Contreras

Profesor Guía:

Martin José García Gutiérrez



CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título ☐ Tesis de Postgrado

Título del trabajo: Proyecto de Sistema de Monitoreo para Silos de Alimento de Salmon

Nombre del candidato(a): Juan Pablo Berríos Contreras

Carrera / Grado: Ingeniería de ejecución en control e instrumentación industrial

Campus: Sede José Miguel Carrera, Viña del mar.

Departamento: Electrotecnia e Informática

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Martín García Gutiérrez , en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución.

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL (marcar una opción)

☒ El trabajo **NO contiene** información que amerite confidencialidad y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (**embargo**) por (**marcar una opción**):

☐ 6 meses ☐ 12 meses ☐ 2 años ☐ 3 años ☐ 5 años ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 12 de AGOSTO 2026. Firma: _____

Estudiante o Candidato(a):

Fecha: 07 de septiembre 2026 Firma: _____

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

RESUMEN

KEYWORDS: monitoreo de nivel, sensor radar, Python, Raspberry Pi.

Este trabajo de título presenta el diseño de un sistema de monitoreo de nivel para silos de alimento para salmón. El proyecto contempla el análisis técnico de distintos dispositivos, entre ellos módulos LoRa, módulo de comunicación 4G para acceso a internet, Raspberry Pi como unidad de procesamiento y sensores de radar para la medición del nivel.

Asimismo, se consideran sus respectivas fuentes de alimentación. y herramientas de software como Python, Raspberry Pi OS (Raspbian), Tailscale y Node-RED para la programación, integración y visualización del sistema.

ÍNDICE

RESUMEN	I
ÍNDICE	II
ÍNDICE DE FIGURAS	V
ÍNDICE DE TABLAS	VI
ÍNDICE DE ECUACIONES	VII
SIGLAS Y SIMBOLOGÍA	VIII
INTRODUCCIÓN	1
CAPÍTULO 1: ANTECEDENTES GENERALES	2
1. ANTECEDENTES GENERALES	3
1.1 Planteamiento del problema	3
1.2 Solución del problema	6
1.3 Sistema actual de medición	6
1.4 Requerimientos	6
1.4.1 Alternativa N°1: Mantener el sistema actual de medición	7
1.4.2 Alternativa N°2: Mejora del sistema actual	7
1.4.3 Alternativa N°3: Cambio total del sistema	8
1.5 Alternativa seleccionada	9
1.5.1 Evaluación de la Alternativa 1: Mantener el sistema actual	9
1.5.2 Evaluación de la Alternativa 2: Mejora del sistema actual	10
1.5.3 Evaluación de la Alternativa 3: Cambio total del sistema	11
1.6 Evaluación de las alternativas y elección final	11
1.7 Objetivos del proyecto	12
1.7.1 Objetivo General	12
1.7.2 Objetivos Específicos	12

CAPÍTULO 2: DESARROLLO DE LA SOLUCIÓN	13
2. Desarrollo de la solución	14
2.1 Diagrama de flujo general del sistema propuesto	14
2.2 Sistema de monitoreo: visión general	16
2.3 Sensor medidor de nivel (Radar de onda electromagnética)	16
2.4 Fuente de alimentación	17
2.5 Comunicación inalámbrica (LoRa)	17
2.6 Unidad de procesamiento central	17
2.7 Visualización y control	18
2.8 Evaluación Técnica y Comparativa de Tecnologías	18
2.9 Evaluación de sensores	19
2.10 Etapa de fuente de alimentación	21
2.11 Etapa de transmisión de datos	22
2.12 Evaluación de Unidad de Procesamiento Central	24
2.13 Etapa de monitoreo o visualización de datos	26
2.14 Configuración de la solución, diagrama y programación de los dispositivos	27
2.15 Ecuaciones	30
2.16 Sensor de medición en silo	31
2.17 RS-485	33
2.18 Cable de comunicación RS-485.....	34
2.19 Conectores	36
2.20 Convertidor de RS-485 a USB	37
2.21 Raspberry Pi	38
2.22 Sistema Operativo (Raspberry Pi OS – Raspbian)	38
2.23 Lenguaje de programación (Python)	39
2.24 Programa desarrollado desde Raspberry Pi en Python	42
2.25 Llamada del sensor por Modbus y petición de la información	42
2.26 Validación de datos y control de errores	41
2.27 Cálculo de variables del sistema	44

2.28 Generación de estados, alertas y avisos	45
2.29 Node-RED y activación del servidor	47
2.30 Implementación	47
2.31 Tailscale	49
2.32 Pasos para instalar Tailscale en Raspberry	50
2.33 Llamado de variables y diseño de dashboard	53
 CAPÍTULO 3: EVALUACIÓN ECONÓMICA DEL PROYECTO	60
3. Evaluación económica del proyecto	61
3.1 Carta Gantt	61
3.2 Costos mano de obra	62
3.3 Costos de materiales a utilizar	63
3.4 Resumen de costo del Proyecto	63
3.5 Estimación del valor referencial del alimento y beneficio económico anual.....	65
3.6 Flujo de caja	67
3.7 Análisis de sensibilidad	68
 CONCLUSIÓN	69
BIBLIOGRAFÍA	70

ÍNDICE DE FIGURAS

Figura 1.1: Sobre producción de salmones y ubicación de centros de cultivos	4
Figura 2.1: Diagrama de flujo	15
Figura 2.2: Esquemático del sistema.....	16
Figura 2.3: Dimensiones de un silo	27
Figura 2.4: Análisis del silo en cubo	28
Figura 2.5: Análisis del silo tronco pirámide invertida.....	28
Figura 2.6: Vista general del silo	29
Figura 2.7: Punto de condiciones	30
Figura 2.8: Sensor de Radar	32
Figura 2.9: Diagrama Interno Sensor	32
Figura 2.10: Comunicación RS485 y tipo de conexión	33
Figura 2.11: Cable de comunicación y alimentación.....	35
Figura 2.12: Conectores macho y hembra	36
Figura 2.13: Modulo conversor Rs485 a USB.....	37
Figura 2.14: Partes de Raspberry pi 4	38
Figura 2.15: Interfaz de sistema operativo	39
Figura 2.16: Verificación de instalación Python.....	40
Figura 2.17: Pasos para creación de archivo Python	41
Figura 2.18: Solicitud de información por ModBus	42
Figura 2.19: Validación de datos	43
Figura 2.20: Calculo de Variables	44
Figura 2.21: Alertas y advertencias	45
Figura 2.22: Código para pontón secundario.....	46
Figura 2.23: Código para pontón Maestro	46
Figura 2.24: Enlace de instalación	47
Figura 2.25: Instalación finalizada.....	48
Figura 2.26: Iniciar servidor.....	48

Figura 2.27: Acceso a servidor.....	48
Figura 2.28: Espacio de trabajo Node Red	49
Figura 2.29: Visualización de Proceso de instalación Tailscale	51
Figura 2.30: Vinculación Raspberry a la Red	51
Figura 2.31: Vista de aplicación de Tailscale	52
Figura 2.32: Acceso a Node Red por Tailscale.....	52
Figura 2.33: Creación de Variable Dato.....	53
Figura 2.34: Librería y Comando Json	54
Figura 2.35: Instalación de librería y comando MQTT	54
Figura 2.36: Configuración nodos dashboard	55
Figura 2.37: Descripción funciones de nodos en Node Red	56
Figura 2.38: Visualización de ejemplo de los silos y porcentaje de llenado	57
Figura 2.39: Arquitectura general de un Pontón	58
Figura 2.40: Vista lateral de Pontón y Diseño final del sistema.....	58
Figura 2.41: Arquitectura de comunicación entre pontones con LoRa y 4G	59
Figura 2.42: P&ID de un pontón.....	59

ÍNDICE DE TABLAS

Tabla 1.1: Requerimientos	7
Tabla 1.2: Tabla de puntuación	9
Tabla 1.3: Clasificación de alternativas	11
Tabla 2.1: Análisis sensor ultrasónico	19
Tabla 2.2: Análisis sensor LIDAR	19
Tabla 2.3: Análisis sensor radar	20
Tabla 2.4: Análisis fuente Mean Well SDR-240-24	21

Tabla 2.5: Análisis fuente Mean Well HDR-60-24	21
Tabla 2.6: Análisis Well HLG 240H 24	24
Tabla 2.7: Análisis LoRa	23
Tabla 2.8: Análisis Wi-Fi	23
Tabla 2.9: Análisis 4G	23
Tabla 2.10: Análisis Raspberry Pi 4	24
Tabla 2.11: Análisis Raspberry Pi 5	25
Tabla 2.12: Análisis Raspberry Pi 3	25
Tabla 2.13: Análisis monitor estándar HDMI	26
Tabla 2.14: Análisis de monitor industrial	26
Tabla 3.1: Carta Gantt del proyecto	61
Tabla 3.2: Costos de mano de obra	62
Tabla 3.3: Costos de materiales a utilizar	63
Tabla 3.4: Resumen del costo del proyecto	63
Tabla 3.5: Flujo de caja	67
Tabla 3.6: Análisis de sensibilidad	68

ÍNDICE DE ECUACIONES

Ecuación 1 : Área en una altura intermedia del trapecio	30
Ecuación 2: Volumen en el tramo trapezoidal ($C \rightarrow B$)	30
Ecuación 3: Volumen total del trapecio (cuando $h=H_2$)	30
Ecuación 4 : Función volumen total por tramos $V(h)$	31

SIGLAS Y SIMBOLOGÍA

A. SIGLAS:

HMI	: Interfaz Hombre-Máquina.
LoRa	: Long Range
E/S	: Entrada/Salida
USB	: Bus Serie Universal.
Json	: JavaScript Object Notation.
MQTT	: Message Queuing Telemetry Transport.

B. SIMBOLOGÍA:

V	: Volts.
%	: Por ciento.
cm	: Centímetros
s	: Segundos.
b	: Bit
m ²	: metros cuadrados
m ³	: metros cúbicos

INTRODUCCIÓN

Chile es uno de los líderes mundiales en la producción de salmón, con una industria acuícola que tiene gran presencia en las regiones de Los Lagos, Aysén y Magallanes. Esta actividad representa una fuente clave de empleo y desarrollo económico, pero también enfrenta desafíos importantes, especialmente en el manejo eficiente del alimento para los peces.

La alimentación es uno de los costos operacionales más altos, y su control impacta directamente en la productividad, el medio ambiente y la sostenibilidad del sector. Actualmente, muchas embarcaciones como Pontones almacenan el alimento en silos, pero el monitoreo de estos niveles se realiza de forma manual o con sistemas industriales costosos, lo que limita su adopción en flotas más pequeñas o antiguas.

Este proyecto propone el desarrollo de un sistema de medición de alimento accesible, adaptable y de fabricación local, utilizando tecnologías abiertas como Raspberry Pi, junto con comunicación inalámbrica LoRa y una plataforma central de monitoreo. La solución busca optimizar la dosificación, reducir desperdicios, mejorar la planificación logística y abrir oportunidades para el desarrollo tecnológico nacional, alineándose con las tendencias globales de eficiencia y sustentabilidad en la industria salmonera.

CAPÍTULO 1: ANTECEDENTES GENERALES

1. ANTECEDENTES GENERALES

Este capítulo presenta la problemática a resolver, el método mediante al cual se da solución y se definen los objetivos a cumplir.

1.1 PLANTEAMIENTO DEL PROBLEMA

El control actual del alimento almacenado en los silos utilizados por la industria salmonera presenta limitaciones operacionales importantes debido a la alta dependencia de estimaciones visuales y controles manuales, lo que puede generar errores en la cuantificación del inventario disponible, desperdicio de alimento, dificultades en la planificación del reabastecimiento e ineficiencias operativas. Considerando que el alimento representa uno de los costos más relevantes dentro del proceso productivo, una gestión inexacta de este recurso puede traducirse no solo en pérdidas económicas, sino también en impactos ambientales asociados a una administración deficiente del proceso de alimentación.

En Chile, la actividad salmonicultura se encuentra estrictamente regulada por la Ley General de Pesca y Acuicultura (LGPA) y el Reglamento de Concesiones y Autorizaciones de Acuicultura (Decreto N°290), los cuales establecen que las empresas deben presentar un proyecto técnico que defina parámetros como la cantidad de peces a cultivar, densidad de siembra y condiciones operacionales asociadas, requiriendo aprobación previa por parte de la Subsecretaría de Pesca y Acuicultura (SUBPESCA). Asimismo, cualquier modificación en la producción autorizada puede requerir actualizaciones del proyecto técnico y, dependiendo del caso, procesos adicionales de evaluación ambiental.

Dentro de este contexto regulatorio, la correcta gestión del alimento adquiere especial relevancia, dado que constituye un recurso crítico tanto desde el punto de vista económico como operacional.

Un control deficiente del inventario puede afectar la trazabilidad del proceso de alimentación, dificultar la planificación operativa y reducir la capacidad de supervisión sobre uno de los insumos más relevantes dentro del ciclo productivo.

La importancia de fortalecer los mecanismos de control en la industria salmonera también se refleja en el marco de fiscalización ambiental vigente. Como se observa en la Figura 1.1, la sobreproducción de salmones, entendida como la superación de los límites autorizados en las respectivas resoluciones de calificación ambiental (RCA), constituye uno de los incumplimientos ambientales relevantes dentro del sector. En 2023, la industria salmonera chilena exportó más de un millón de toneladas, alcanzando un valor superior a USD 6.448 millones, lo que evidencia la magnitud económica y operativa de esta actividad.

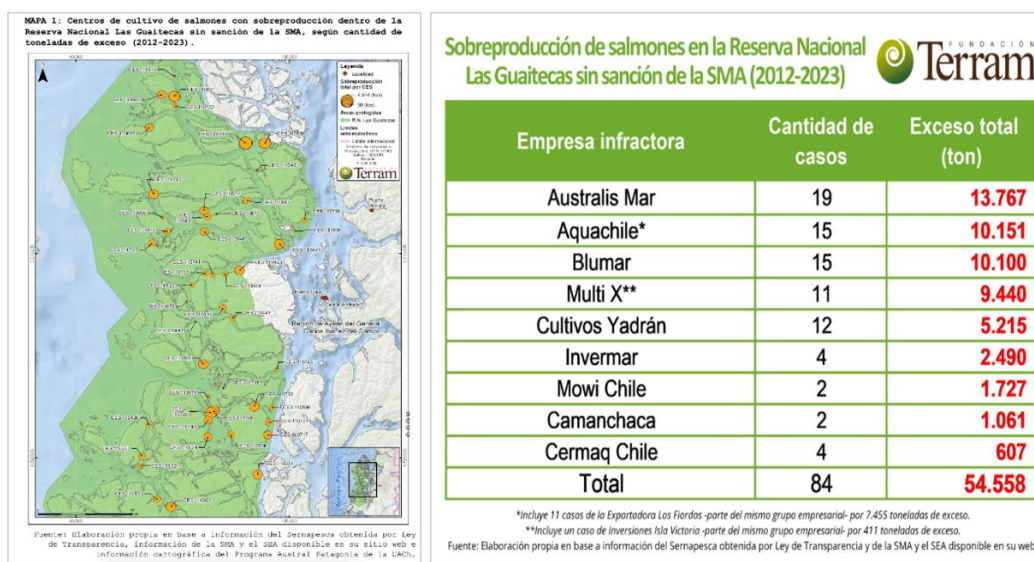


Figura 1.1: Sobreproducción de salmones y ubicación de centros de cultivo
Fuente: <https://www.terram.cl>

Con el objetivo de fiscalizar el cumplimiento de estas obligaciones, el 28 de diciembre de 2012 entró en vigor la Superintendencia del Medio Ambiente (SMA), organismo encargado de supervisar y eventualmente sancionar incumplimientos ambientales dentro del sector acuícola. En este contexto, la incorporación de tecnologías que permitan mejorar la precisión, trazabilidad y supervisión de procesos críticos, como la gestión del alimento almacenado representa una oportunidad concreta para fortalecer el control operacional dentro de la industria.

La sobreproducción de salmones, al producir más de lo autorizado en el permiso ambiental, se traduce en una sobrecarga de nutrientes que se depositan en el fondo marino producto del aumento de las fecas y el alimento no ingerido por los peces, lo cual genera una disminución del oxígeno disponible, poniendo en riesgo la vida de las especies que albergan estos ecosistemas.

Producto de la sobreproducción en el centro Matilde 2 (110778) de Australis Mar, durante el ciclo 2020-2021 la cantidad de alimento utilizado fue de 5.015 toneladas, un 27% por encima de las 3.950 establecidas en la RCA [ver ANEXO 3], por lo cual el alimento no ingerido fue mayor al estimado, como también las fecas de los peces acumulada en el fondo marino del área protegida que con seguridad superó las 231 toneladas declaradas en la RCA. Este aumento de nutrientes hizo disminuir el oxígeno disponible a tal punto que, solo en el transcurso de dos semanas, ocasionó la muerte por asfixia de 31.381 salmones adultos, entre el 14 y 25 de mayo de 2021, según informó la propia empresa a la SMA (en el mismo documento anteriormente citado).

La alimentación y manejo de peces en la industria salmonera chilena impacta directamente en la eficiencia productiva, el medio ambiente y la competitividad del sector. Implementar soluciones tecnológicas para medir y controlar el alimento suministrado permite reducir pérdidas económicas, mejorar el bienestar animal y minimizar la contaminación marina.

Optimizar este proceso fortalece la sustentabilidad, facilita el cumplimiento normativo y mejora la imagen internacional del salmón chileno como producto responsable. Además, impulsa la innovación, promueve una producción eficiente y ayuda a mantener el liderazgo exportador de Chile en el mercado global.

Los principales involucrados en este problema son las empresas salmoneras, que deben garantizar una producción eficiente y sustentable; los trabajadores del sector acuícola, cuya seguridad y condiciones laborales pueden verse afectadas por el uso de tecnologías obsoletas; y las comunidades que habitan cerca de los centros de cultivo, que se ven impactadas por la contaminación y el deterioro del entorno marino. También está involucrado el medio ambiente, debido a la acumulación de residuos orgánicos y el desequilibrio ecológico que puede generarse por una mala gestión del alimento.

Finalmente, la economía del país se ve directamente influenciada, ya que la salmonicultura representa uno de los principales sectores exportadores de Chile, por lo que cualquier mejora tecnológica o productiva tiene efectos positivos en la competitividad nacional y en la generación de empleo.

1.2 SOLUCIÓN DEL PROBLEMA

Se propone un sistema de monitoreo de silos basado en medición de nivel de alimento. Esto permite reducir costos asociados al control manual y al desperdicio de alimento. Mejora la planificación del reabastecimiento y la gestión de inventarios. Ofrece un control aceptable y práctico del alimento en los silos. Su electrónica es actualizable, lo que permite escalar y mejorar el sistema en el futuro.

1.3 SISTEMA ACTUAL DE MEDICIÓN

Actualmente, el control de la cantidad de alimento almacenado en los silos de los pontones se realiza principalmente mediante métodos manuales y estimaciones visuales. El personal verifica el nivel de alimento disponible utilizando las referencias existentes en los silos y la experiencia adquirida durante la operación, sin contar con un sistema automatizado que permita obtener una medición continua y en tiempo real. Este procedimiento puede presentar diferencias entre la cantidad estimada y la cantidad real de alimento disponible.

1.4 REQUERIMIENTOS

Se busca desarrollar un sistema automatizado para monitorear el nivel de alimento en silos instalados en pontones. Este sistema debe entregar información en tiempo real, generar alertas de reposición y registrar datos históricos para mejorar la planificación logística, reducir pérdidas y optimizar la alimentación. A continuación, se presentan los requerimientos en la tabla 1.1 para el desarrollo del proyecto.

Tabla 1.1: Requerimientos

Requerimientos Funcionales
Medir el nivel de alimento en cada silo
Visualizar el inventario en tiempo real
Generar alertas automáticas de nivel
Transmitir datos de forma inalámbrica (LoRa). 4G
Adaptarse a distintos tamaños de silos y embarcaciones
Ofrecer una interfaz clara y fácil de usar para los operadores
Requerimientos No Funcionales
Resistir condiciones marinas (humedad, salinidad)
Contar con protecciones IP67 en los componentes electrónicos
Mantener un costo accesible frente a soluciones comerciales

Fuente: Elaboración propia.

1.4.1 ALTERNATIVA N°1: MANTENER EL SISTEMA ACTUAL DE MEDICIÓN.

La primera alternativa frente a la problemática planteada es mantener el sistema actual, utilizando métodos manuales o rudimentarios para estimar el nivel de alimento en los silos, sin incorporar sensores ni automatización. Esta opción no requiere inversión inicial ni cambios operativos, y tampoco demanda capacitación del personal. Sin embargo, presenta importantes desventajas: baja precisión en la medición, mayor riesgo de sobrealimentación o subalimentación, pérdida económica por desperdicio de alimento, y baja competitividad frente a sistemas automatizados. En consecuencia, aunque no implique costos inmediatos, esta alternativa desaprovecha la oportunidad de mejorar la eficiencia y el control del proceso, limitando el potencial de optimización futura.

1.4.2 ALTERNATIVA N°2: MEJORA DEL SISTEMA ACTUAL

Esta alternativa propone implementar un sistema de medición para los silos basado en componentes económicamente accesibles y de alta eficiencia, incorporando tecnologías ampliamente utilizadas en aplicaciones de monitoreo industrial. El sistema utiliza una Raspberry Pi como unidad de procesamiento principal, junto con sensores de radar para la medición del

nivel de alimento en cada silo. La comunicación entre los sensores y la unidad central se realizará mediante el protocolo RS-485, mientras que la interconexión entre pontones cercanos se llevará a cabo mediante tecnología LoRa para la transmisión inalámbrica de datos a larga distancia. La información recopilada será procesada, organizada y visualizada a través de un servidor local implementado en Node-RED, permitiendo el monitoreo centralizado del sistema. Entre las ventajas destacan el bajo costo frente a sistemas industriales, alta precisión si se calibra correctamente, escalabilidad a múltiples silos y centros de cultivo, mejora de la trazabilidad y posibilidad de análisis de eficiencia, así como integración con alertas y reportes automáticos.

Entre las desventajas se incluyen la necesidad de conocimientos técnicos para instalación y mantenimiento, posibles problemas de conectividad en zonas remotas y calibración periódica.

1.4.3 ALTERNATIVA N°3: CAMBIO TOTAL DEL SISTEMA

Esta alternativa propone una transformación digital completa del proceso de alimentación, adoptando una solución industrial avanzada que integre medición, control y automatización. Se implementarán sensores de nivel industriales con calibración automática, sistemas PLC y SCADA para control centralizado, integración con alimentación automática y monitoreo ambiental, y el uso de inteligencia artificial para ajustar la dosis de alimento en tiempo real. Entre las ventajas destacan la máxima precisión y automatización, reducción de errores humanos, optimización continua basada en datos, y mejoras en la salud de los peces y la eficiencia productiva.

Las desventajas incluyen el alto costo de inversión inicial, la necesidad de personal capacitado en sistemas industriales, mayor complejidad en mantenimiento y operación, y menor flexibilidad para instalaciones pequeñas o con presupuestos limitados. En consecuencia, esta alternativa ofrece un control de nivel industrial y optimización completa del proceso, aunque requiere una inversión significativa y capacidades técnicas avanzadas para su operación.

1.5 ALTERNATIVA SELECCIONADA.

Para determinar la alternativa más adecuada entre las propuestas anteriormente descritas, se utiliza la tabla 1.2 de criterios de evaluación, basada en tres dimensiones clave: innovación, beneficio y factibilidad.

Tabla 1.2: Tabla de puntuación

Tabla de Puntuación de Alternativas de Solución		
Muy deficiente	Aceptable	Óptimo
1	2	3

Fuente: Elaboración propia en base a alternativas

1.5.1 EVALUACIÓN DE LA ALTERNATIVA 1: MANTENER EL SISTEMA ACTUAL

Factibilidad:

Se considera aceptable (2 puntos), ya que mantener el sistema actual es factible al no requerir inversión adicional, modificaciones en la infraestructura ni cambios significativos en la operación. Sin embargo, presenta limitaciones asociadas a la precisión de las mediciones y al control del alimento almacenado.

Innovación:

Se califica como muy deficiente (1 punto), dado que no incorpora nuevas tecnologías ni mejora los procesos de automatización o control.

Beneficio:

Se evalúa como muy deficiente (1 punto), ya que la dosificación manual del alimento puede generar sobrealimentación, pérdidas económicas y contaminación ambiental por exceso de alimento no consumido.

1.5.2 EVALUACIÓN DE LA ALTERNATIVA 2: MEJORA DEL SISTEMA ACTUAL

Factibilidad:

Esta alternativa propone el uso de tecnología abierta y componentes de bajo costo, Raspberry Pi y sensores de radar, junto con módulos de comunicación como LoRa. Gracias a la disponibilidad comercial y la facilidad de implementación, se considera una solución altamente factible, con una puntuación de 3 (óptimo).

Innovación:

Aunque no se trata de una solución industrial de alta gama, esta alternativa incorpora automatización, conectividad remota y visualización de datos, lo que representa un salto significativo respecto al sistema actual. Se califica como aceptable (2 puntos), ya que permite modernizar el proceso sin requerir grandes inversiones.

Beneficio:

La implementación de este sistema permite mejorar la precisión en la dosificación de alimento, reducir el desperdicio, aumentar la trazabilidad y facilitar la toma de decisiones operativas. Se considera un beneficio óptimo (3 puntos), especialmente en contextos donde se busca eficiencia sin comprometer el presupuesto.

1.5.3 EVALUACIÓN DE LA ALTERNATIVA 3: CAMBIO TOTAL DEL SISTEMA

Esta alternativa propone una transformación completa del proceso de alimentación mediante tecnologías industriales avanzadas, como sensores de nivel con calibración automática, sistemas PLC y SCADA, integración con alimentación automática y monitoreo ambiental.

Factibilidad:

Aunque técnicamente viable, su implementación requiere una inversión considerable y personal especializado, lo que limita su aplicabilidad en embarcaciones pequeñas o flotas con bajo presupuesto. Se califica como aceptable (2 puntos).

Innovación:

Representa un alto nivel de automatización y modernización, por lo que se considera óptima (3 puntos) en este aspecto.

Beneficio:

Ofrece mejoras importantes en precisión y eficiencia, pero su alto costo y complejidad pueden dificultar su adopción generalizada. Se califica como bueno (2 puntos).

1.6 EVALUACIÓN DE LAS ALTERNATIVAS Y ELECCIÓN FINAL.

Tabla 1.3: Clasificación de alternativas

Característica/Alternativa	Alternativa 1	Alternativa 2	Alternativa 3
Innovación Tecnológica	1	3	2
Beneficio Económico	1	2	3
Factibilidad de Instalación	2	3	2
Puntuación total	4	8	7

Fuente: Elaboración propia

1.7 OBJETIVOS DEL PROYECTO

Para el correcto desarrollo del proyecto, se definen el objetivo general y objetivos específicos.

1.7.1 OBJETIVO GENERAL

Diseñar un sistema automatizado de monitoreo de silos en pontones

1.7.2 OBJETIVOS ESPECÍFICOS

- a. Describir el sistema de alimentación actual.
- b. Desarrollar una solución de ingeniería a la problemática.
- c. Evaluar económicamente la solución proyectada al problema.

CAPÍTULO 2: DESARROLLO DE LA SOLUCIÓN

2 DESARROLLO DE LA SOLUCIÓN

El siguiente capítulo da a conocer el proceso mediante el cual se da cumplimiento a los objetivos definidos anteriormente.

2.1 DIAGRAMA DE FLUJO GENERAL DEL SISTEMA PROPUESTO

Como se muestra en la Figura 2.1: Diagrama de flujo y en la figura 2.2: Esquemático del sistema, la filosofía de funcionamiento del sistema se basa en la adquisición, transmisión, procesamiento y visualización de la información correspondiente al nivel de alimento almacenado en los silos. En cada silo, un sensor de radar realiza la medición del nivel de pellet y transmite los datos hacia la unidad central mediante comunicación RS-485. La Raspberry Pi recibe y procesa esta información, permitiendo calcular el nivel y volumen de alimento disponible, además de identificar condiciones de operación y generar alertas cuando se alcanzan niveles críticos. En instalaciones que cuenten con más de un pontón, la información puede ser transmitida mediante comunicación LoRa hacia un pontón maestro, desde donde los datos son concentrados y puestos a disposición del operador mediante una interfaz de monitoreo local o remoto. De esta manera, el sistema sigue una estructura secuencial desde la adquisición de la variable de proceso hasta la entrega de información útil para la supervisión y toma de decisiones.

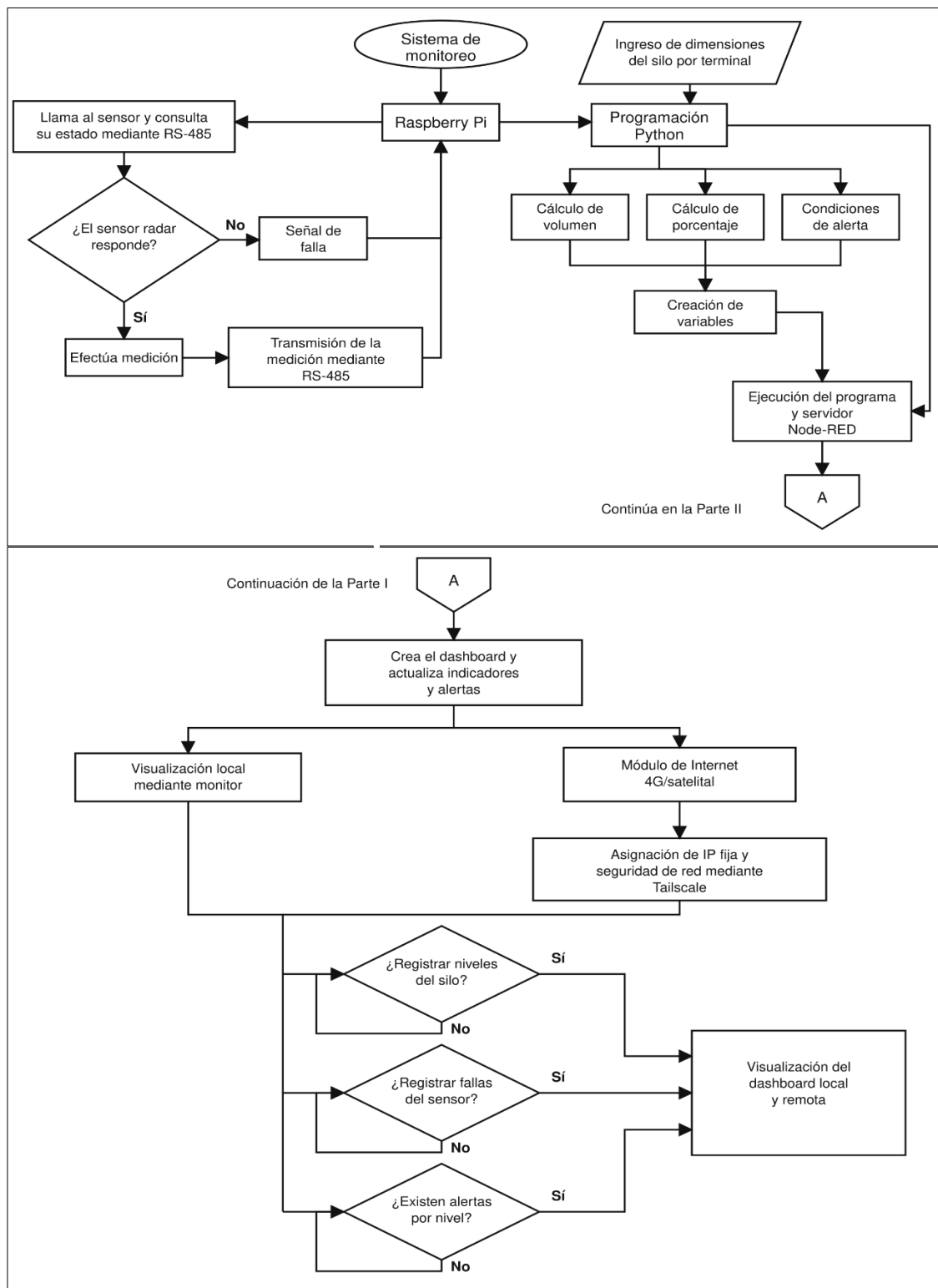


Figura 0: Diagrama de flujo
 Fuente: Elaboración propia a partir de los requerimientos del sistema.

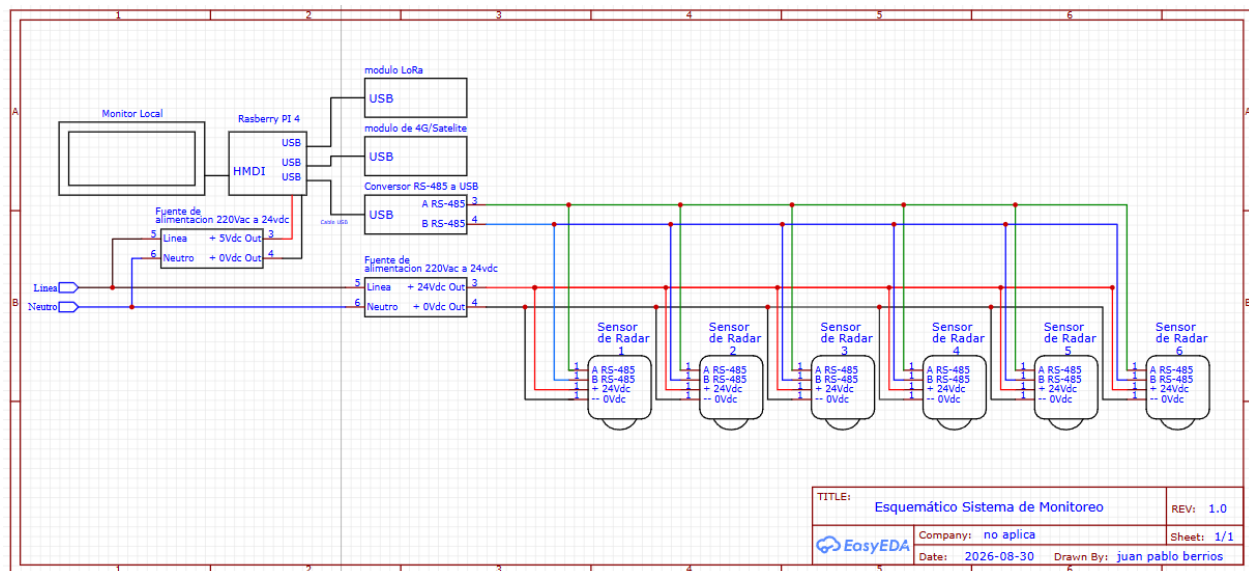


Figura 2.2: Esquemático del sistema
Fuente: Elaboración propia hecha con el software EASYEDA.

2.2 SISTEMA DE MONITOREO VISIÓN GENERAL

El sistema consiste en una red de sensores medidores (uno por silo) que envían información de nivel hacia un sistema central. El objetivo es obtener lecturas periódicas de nivel, y presentarlas en una interfaz para control/decisión.

2.3 SENSOR MEDIDOR DE NIVEL (RADAR DE ONDA ELECTROMAGNÉTICA)

El sensor de nivel funciona mediante la emisión de ondas electromagnéticas hacia el interior del silo. Estas ondas se reflejan en la superficie del pellet y regresan al sensor, permitiendo calcular con precisión la distancia recorrida.

El dispositivo realiza este proceso de forma continua, generando una medición estable del nivel de alimento presente. Su principio de funcionamiento se basa en el tiempo de vuelo de la señal, lo que permite obtener lecturas confiables incluso en condiciones variables dentro del silo.

2.4 FUENTE DE ALIMENTACIÓN

La fuente de alimentación proporciona la energía necesaria para el funcionamiento de los sensores. Este equipo cumple con certificaciones y estándares de protección IP67/IP68, lo que garantiza un adecuado nivel de resistencia frente a humedad, polvo y condiciones ambientales exigentes, características propias del entorno de trabajo donde será instalado.

2.5 COMUNICACIÓN INALÁMBRICA (LORA)

Se dispone de un módulo LoRa en cada pontón, el cual permite transmitir de forma inalámbrica la información correspondiente a los niveles de los silos hacia un pontón maestro. Este pontón central cuenta con acceso a internet, permitiendo posteriormente enviar la información a la red y dejarla disponible para su monitoreo remoto.

2.6 UNIDAD DE PROCESAMIENTO CENTRAL

El sistema central se encarga de recibir las señales enviadas por los sensores a través del estándar de comunicación RS-485 y procesarlas para su posterior almacenamiento y visualización en pantalla y servidor.

En caso de existir pontones cercanos, se designa uno de ellos como sistema central maestro, el cual recibirá la información proveniente de los otros pontones mediante comunicación inalámbrica a través de módulos LoRa. Además, este sistema central cuenta con un módulo 4G que permite el acceso a internet, posibilitando el envío de la información hacia la red para su almacenamiento y monitoreo remoto.

2.7 VISUALIZACIÓN Y CONTROL

La interfaz de usuario permite visualizar de manera clara el nivel de alimento presente en cada silo, utilizando indicadores gráficos simples y directos. La información se muestra en un monitor conectado a la Raspberry Pi, y además se ejecuta un servidor en el mismo equipo, el cual permite acceder a los datos de forma remota desde distintos dispositivos mediante un dashboard desarrollado en Node-RED.

El sistema presenta el estado general de los silos y destaca aquellos que se encuentren en niveles críticos, con el fin de facilitar la toma de decisiones por parte del operador.

Para el envío seguro de la información hacia la red, se utiliza el servicio Tailscale, el cual proporciona una red privada segura entre los dispositivos, permitiendo el acceso remoto mediante una dirección IP virtual fija y garantizando la protección de los datos transmitidos.

Adicionalmente, se utiliza Python para el desarrollo de las subrutinas encargadas de procesar la información recibida, realizar los cálculos aritméticos necesarios, generar alertas de nivel, y detectar posibles fallas en el sistema o en la comunicación de los sensores.

2.8 EVALUACION TÉCNICA Y COMPARATIVA DE TECNOLOGÍAS

A continuación, se realiza una comparación de distintas alternativas de dispositivos disponibles en el mercado. Debido a la amplia variedad de equipos existentes, se seleccionan aquellos que cumplen con los requerimientos funcionales del sistema propuesto. Posteriormente, entre estas alternativas se analiza cuál representa la mejor opción en términos de desempeño, compatibilidad y adecuación al sistema en general.

2.9 EVALUACIÓN DE SENSORES

En las tablas 2.1, 2.2 y 2.3 se comparan sensores de medición de nivel basados en distintas tecnologías, tales como ultrasonido, LiDAR y radar.

Tabla 2.1: Análisis sensor ultrasónico

Sensor Ultrasónico		
Principio de funcionamiento	Ventajas	Desventajas
Emite ondas sonoras de alta frecuencia que rebotan en la superficie del material, midiendo el tiempo de retorno.	- Bajo costo	- Sensible a humedad, espuma y temperatura - Precisión reducida ante viento o movimiento - Puede requerir recalibración frecuente

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Tabla 2.2: Análisis sensor LIDAR

Sensor óptico o LIDAR		
Principio de funcionamiento	Ventajas	Desventajas
Mide la distancia mediante un haz de luz reflejado en la superficie del material.	- Alta precisión - Respuesta muy rápida - Ideal para sólidos secos	- Afectado por polvo, suciedad o niebla - Alto costo de mantenimiento - Menor confiabilidad en ambientes húmedos

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Tabla 2.3: Análisis sensor radar

Sensor radar (onda electromagnética)		
Principio de funcionamiento	Ventajas	Desventajas
Emite ondas electromagnéticas que se reflejan en la superficie del material; mide la distancia a partir del tiempo de retorno.	-Inmune a: humedad temperatura niebla - Alta precisión y estabilidad - Mantención casi nulo - Funciona con movimiento u oscilación del material	Costo inicial más alto

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Selección final: Sensor radar de nivel, tipo FMCW (Frequency Modulated Continuous Wave).

La justificación técnica del sensor radar se fundamenta en que este ofrece la mejor combinación entre precisión, durabilidad y confiabilidad en condiciones ambientales adversas. En centros de cultivo marinos, donde existe humedad, salinidad, movimiento y espuma, los sensores ultrasónicos y ópticos presentan lecturas erráticas. El radar mantiene una medición constante, incluso con variaciones de temperatura y turbulencia en la superficie del alimento.

Además, al no requerir mantenimiento frecuente, reduce costos operativos y garantiza continuidad en el monitoreo del sistema de alimentación.

2.10 ETAPA DE FUENTE DE ALIMENTACION

En las tablas 2.3, 2.5 y 2.6 se comparan diferentes fuentes de alimentación disponibles en el mercado, seleccionando aquellas que cumplen con los requisitos de tensión, potencia y protección ambiental necesarios para la correcta operación del sistema.

Tabla 2.4: Análisis fuente Mean Well SDR-240-24

Mean Well SDR-240-24	
Ventajas	Desventaja
Entrega alta corriente (10 A) Alta eficiencia energética (94 %), rango de entrada 88–264 V AC protecciones internas contra cortocircuito, sobrecarga, sobretensión y sobre temperatura	Mayor costo instalación dentro de gabinete

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Tabla 2.5: Análisis fuente mean Well HDR-60-24

Mean Well HDR-60-24	
Ventajas	Desventaja
Bajo costo Amplio rango de entrada (85–264 V AC) Incluye protecciones internas contra cortocircuito, sobrecarga y sobretensión	Corriente limitada (2.5 A) si se desea agregar más dispositivos

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Tabla 2.6: Análisis Well HLG 240H 24

Mean Well HLG-240H-24	
Ventajas	Desventaja
Alta protección ambiental (IP65/IP67), resistente a polvo, humedad y chorros de agua Amplio rango de entrada 90–305 V AC, compatible con distintas redes eléctricas Enfriamiento pasivo, sin ventiladores ni mantenimiento	No está diseñada para montaje en riel DIN, requiere fijación mecánica Instalación eléctrica menos sencilla (cables sellados en lugar de bornes)

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Selección final: Fuente conmutada industrial 24 V DC – 10 A (Mean Well SDR-240-24)

La justificación técnica de la fuente seleccionada se fundamenta en su alta capacidad de corriente, permitiendo alimentar de forma segura todos los sensores y dispositivos del sistema desde un único punto de suministro. Además, cuenta con protecciones integradas y alta eficiencia, garantizando una alimentación estable y un margen adecuado para futuras ampliaciones del sistema.

2.11 ETAPA DE TRANSMISIÓN DE DATOS

En las tablas 2.7, 2.8 y 2.9 se comparan diferentes tecnologías de transmisión de datos, considerando aspectos como el alcance, consumo energético, estabilidad de la comunicación y capacidad de transmisión, con el propósito de seleccionar la alternativa más adecuada para la comunicación del sistema.

Tabla 2.7: Análisis LoRa

LoRa (punto a punto)	
Ventajas	Desventaja
- Alcances 1–10 km - Muy bajo consumo - Estable en zonas rurales/marinas	- Velocidad baja - No apta para grandes volúmenes de datos

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Tabla 2.8: Análisis Wifi

Wi-Fi	
Ventajas	Desventaja
- Alta velocidad - Fácil integración con Reuters	- Alcance corto - Muy afectado por metal y agua

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Tabla 2.9: Análisis 4G/LTE

4G / LTE	
Ventajas	Desventaja
- Cobertura amplia - Permite conexión a Internet	- Alto consumo - Costo mensual - Señal variable en fiordos

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Selección final: LoRa (punto a punto).

La justificación técnica de la selección de LoRa se basa en su gran alcance, bajo consumo y alta estabilidad en ambientes marinos. Su ancho de banda es suficiente para transmitir los datos del nivel de alimento desde cada silo hacia un nodo central, sin depender de infraestructura externa ni generar costos mensuales. Por estas características, representa la alternativa más confiable y eficiente para el prototipo.

2.12 EVALUACIÓN DE UNIDAD DE PROCESAMIENTO CENTRAL

En las tablas 2.10, 2.11 y 2.12 se comparan diferentes unidades de procesamiento central, considerando aspectos como la capacidad de procesamiento, consumo energético, costo y rendimiento, con el propósito de seleccionar la alternativa más adecuada para el procesamiento y gestión de los datos del sistema.

Tabla 2.10: Análisis Raspberry pi 4

Raspberry Pi 4 Model B (2GB/4GB/8GB)	
Ventajas	Desventaja
- Excelente relación precio/rendimiento - Estable y probada - Económica comparada con PCs industriales	Superado en capacidad por la Raspberry pi 5

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Tabla 2.11: Análisis Raspberry pi 5

Raspberry Pi 5	
Ventajas	Desventaja
- Mayor potencia de procesamiento - Mejor manejo de gráficos y carga pesada	- Más cara - Requiere buena ventilación - Se calienta más que otros modelos

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Tabla 2.12: Análisis Raspberry pi 3

Raspberry Pi 3 Model B+	
Ventajas	Desventaja
- Más económica - Bajo consumo energético	- RAM muy limitada. - Procesamiento más lento

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Selección final: Raspberry Pi 4 Model B (8 GB RAM).

Justificación técnica resumida: La Raspberry Pi 4 ofrece el equilibrio ideal entre costo, potencia y disponibilidad. Tiene suficiente rendimiento para recibir datos LoRa, ejecutar dashboards en tiempo real y manejar gráficos en pantalla HDMI. El modelo 5 es más potente pero innecesario para este prototipo, mientras que la Raspberry Pi 3 puede quedar limitada.

2.13 ETAPA DE MONITOREO O VISUALIZACIÓN DE DATOS

En las tablas 2.13 y 2.14 se comparan diferentes alternativas de monitores para la visualización de datos, considerando aspectos como la compatibilidad, consumo energético, costo, calidad de visualización y resistencia a las condiciones ambientales, con el propósito de seleccionar la alternativa más adecuada para la visualización y monitoreo de los datos del sistema.

Tabla 2.13: Análisis monitor estándar HDMI

Monitor HDMI estándar 21.5"–22" (Samsung / LG / AOC)	
Ventajas	Desventaja
- Económico y fácil de encontrar - Bajo consumo energético - Totalmente compatible con Raspberry Pi - Buena calidad visual para dashboards	- No es industrial - Requiere sala o gabinete techado - No apto para intemperie

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Tabla 2.14: Análisis de monitor industrial

Monitor industrial 21.5" IP65 (táctil o no táctil)	
Ventajas	Desventaja
- Resistente a humedad y salpicaduras - Mayor durabilidad - Construcción robusta	- Más costosa - Mayor peso y complejidad de montaje

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

Selección final: Monitor HDMI estándar de 21.5"-22".

La selección del monitor HDMI estándar de 21.5"-22" se fundamenta en su buena relación costo-beneficio, compatibilidad con la Raspberry Pi y capacidad para proporcionar una visualización clara y amplia de los dashboards de monitoreo. Además, su disponibilidad y bajo costo lo convierten en una alternativa adecuada para su instalación en la sala de control del pontón, sin requerir el uso de un monitor industrial de mayor costo.

2.14 CONFIGURACION DE LA SOLUCION, DIAGRAMA Y PROGRAMACION DE LOS DISPOSITIVOS

En la Figura 2.3 se presentan las dimensiones físicas de un silo contenedor de alimento en pellet. Esta información fue proporcionada por la empresa CERMAQ.

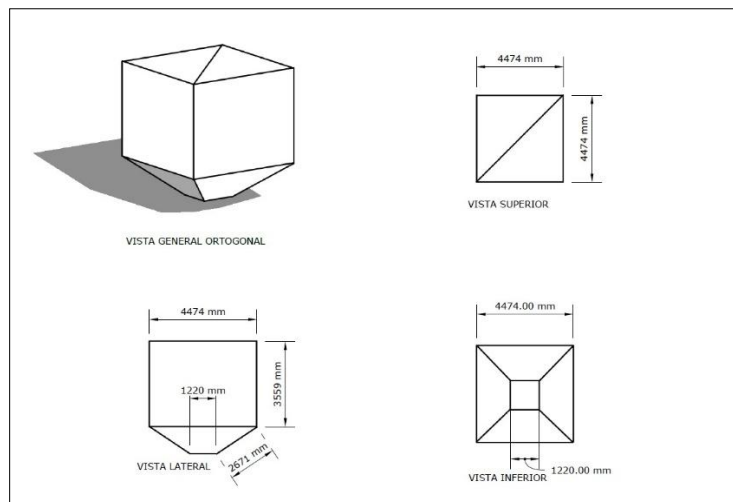


Figura 2.3: Dimensiones de un silo

Fuente: Elaboración propia a partir de información proporcionada por empresa CERMAQ

Para poder calcular su volumen total, se descompone la imagen en dos figuras geométricas conocidas para facilitar el cálculo, tal como se muestran en la figura 2.4 y 2.5. A continuación, se detalla los cálculos volumétricos.

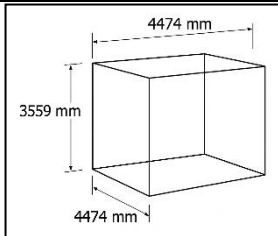


Figura 2.4: Análisis del silo en Cubo

Fuente: Elaboración propia

- Dimensiones del cubo (parte superior del silo)
- Largo = 4,474 m
- Ancho = 4,474 m
- Altura = 3,559 m
- Fórmula del volumen del cubo: $V = L \times A \times H$
- Medidas reales: $V = 4,474\text{m} \times 4,474\text{m} \times 3,559\text{m}$
- Volumen final del cubo: $71,27\text{m}^3$
- En litros $71,27\text{m}^3 \times 1000 = 71.270\text{litros}$

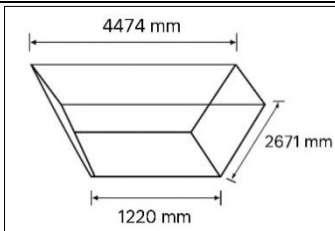


Figura 2.5: Análisis del silo tronco pirámide invertida

Fuente: Elaboración propia

- Dimensiones de la figura (parte inferior del silo)
- Base mayor: 4,474 m
- Base menor: 1,220 m
- Lado inclinado: 2,671 m
- Fondo (profundidad): 4,474 m
- Altura real del trapecio Usando Pitágoras:

$$h = \sqrt{2,671^2 - 1,627^2} \approx 2,118 \text{ m.}$$
- Área del trapecio: $A = ((4,474 + 1,220) \times 2) \times 2,118$
 $A = 6,03 \text{ m}^2$
- Volumen (extrusión con fondo 4,474 m),
 $V = 6,03 \text{ m}^2 \times 4,474 \text{ m}, V = 26,98 \text{ m}^3$

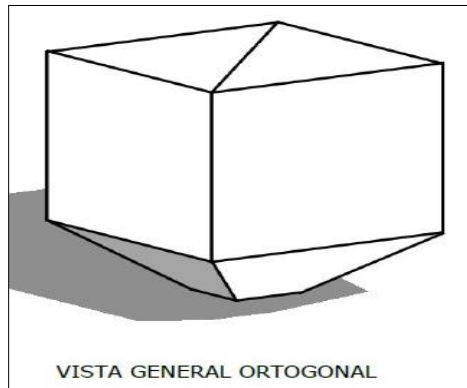


Figura 2.6: Vista General del Silo
Fuente: Elaboración propia

con los cálculos previos se obtiene el volumen total de un solo silo:

- Volumen final del cubo: $71,27 \text{ m}^3$
- Prisma Trapezoidal Irregular $26,98 \text{ m}^3$
- Volumen total = $98,25 \text{ m}^3$

Con el cálculo geométrico del silo se determina su volumen total. Para medir en tiempo real la cantidad de pellet almacenado, se utiliza un sensor de radar que registra continuamente la altura del material dentro del silo. Sin embargo, debido a que el silo no corresponde a una figura geométrica simple, no es posible aplicar una única fórmula directa para el cálculo del volumen parcial. Por este motivo, será necesario definir una función por tramos, donde cada tramo utilice la ecuación que corresponda según la geometría.

Como se observa en la figura 2.7, en el primer tramo, desde la altura A hasta la altura B, la sección interna del silo mantiene forma cuadrada, por lo que se puede utilizar la fórmula básica del volumen de un paralelepípedo. A partir de la altura B hacia abajo, la geometría cambia a una sección trapezoidal irregular, por lo que se debe aplicar una segunda ecuación que permita calcular el volumen considerando las dimensiones variables. El sistema activará automáticamente esta condición según el nivel registrado por el sensor.

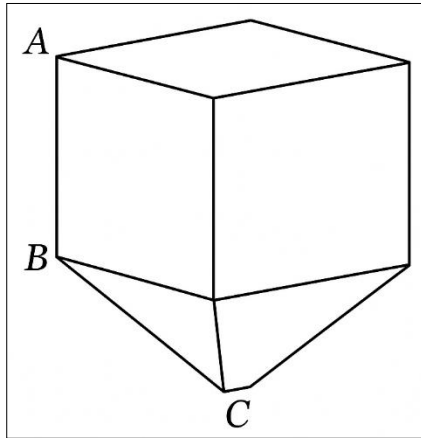


Figura 2.7: Puntos de Condiciones
Fuente: Elaboración propia

Definición de variables

- h : altura del pellet medida desde el punto C hacia arriba.
- H_2 : altura total del tramo trapezoidal ($C \rightarrow B$)
- H_1 : altura del tramo cúbico ($B \rightarrow A$)
- A_C : área de la base (en C)
- A_B : área de la base (en B)
- A_{cuadro} : área de la sección del tramo cúbico

2.15 ECUACIONES

Ecuación 1 : Área en una altura intermedia del trapecio

$$A(h) = A_C + (A_B - A_C) \left(\frac{h}{H_2} \right)^2 \quad (2.1)$$

Ecuación 2: Volumen en el tramo trapezoidal ($C \rightarrow B$)

$$V_{\text{trap}}(h) = \frac{h}{3} [A_C + A(h) + \sqrt{A_C A(h)}] \quad (2.2)$$

Ecuación 3: Volumen total del trapecio (cuando $h=H_2$)

$$V_{\text{trap}} = V_{\text{trap}}(H_2) = \frac{H_2}{3} (A_C + A_B + \sqrt{A_C A_B}) \quad (2.3)$$

Ecuación 4 : Función volumen total por tramos V(h)

$$V(h) \begin{cases} \frac{h}{3} \left[Ac + (Ab - Ac) \left(\frac{h}{H_2} \right)^2 + \sqrt{Ac(Ac + (Ab - Ac) \left(\frac{h}{H_2} \right)^2)} \right], 0 \leq h \leq H_2 \\ \frac{H_2}{3} (Ac + Ab + \sqrt{AcAb}) + A_{cuadrado}(h - H_2), H_2 < h \leq H_2 + H_1 \end{cases} \quad (2.4)$$

2.16 SENSOR DE MEDICIÓN EN SILO

El sensor seleccionado para el sistema corresponde al transmisor radar de nivel SUP-WSR550, como se muestra en la figura 2.8, diseñado para medición de nivel sin contacto en sólidos a granel. El equipo opera en la banda de 76–81 GHz, utilizando tecnología FMCW (Frequency Modulated Continuous Wave) para determinar la distancia entre el sensor y la superficie del material. Entre sus principales características técnicas se encuentran un rango de medición de hasta 15 m para sólidos, precisión aproximada de ± 1 mm, zona ciega de 0,08 m, ángulo de haz cercano a 3° , alimentación 15–28 VDC, consumo máximo de 2 W y grado de protección IP67/IP68. La señal emitida por el radar posee una longitud de onda aproximada de 3,75 mm, obtenida a partir de la relación $\lambda = c/f$, considerando una frecuencia de operación cercana a 80 GHz. Esta longitud de onda reducida permite generar un haz de medición estrecho, disminuyendo la interferencia producida por las paredes del silo o estructuras internas. Para un ángulo de haz de 3° , el diámetro aproximado del área de medición puede estimarse mediante:

$$D \approx 2d \tan\left(\frac{\theta}{2}\right)$$

Donde:

- D = diámetro aproximado del área de medición del radar.
- d = distancia entre el sensor y la superficie del pellet.
- θ = ángulo de apertura del haz del radar.
- tan = función tangente.

Lo que da como resultado aproximadamente 0,26 m a 5 m de distancia y 0,52 m a 10 m, concentrando la medición en una zona reducida de la superficie del pellet.

El sensor entrega la variable medida mediante salida analógica 4–20 mA y comunicación digital RS-485 con protocolo Modbus RTU, permitiendo transmitir directamente la distancia medida en metros hacia el sistema central. A partir de esta variable, el software del sistema calcula el nivel y el volumen de alimento almacenado en el silo utilizando el modelo geométrico previamente definido.



Figura 2.8: Sensor de Radar

Fuente: Elaboración propia en base a información técnica del sensor

En la figura 2.9 se muestra el diagrama de conexión interna del sensor, donde se aplica una tensión de 24 Vcc para su alimentación y se obtiene la señal de medición mediante el estándar de comunicación RS-485 ModBus / TCP/IP / Profibus.

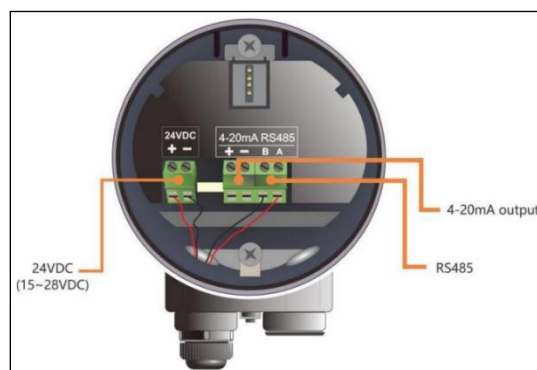


Figura 2.9: Diagrama Interno Sensor

Fuente: Fuente: Sinomeasure, WSR550 Radar Level Transmitter Datasheet.

2.17 RS-485

Para la comunicación entre los sensores y el sistema central se utiliza RS-485 como se muestra en la figura 2.10, un estándar industrial confiable y resistente al ruido eléctrico gracias a su transmisión diferencial. Permite la conexión de múltiples dispositivos en un mismo bus y la transmisión de datos a largas distancias, alcanzando aproximadamente 1200 metros según la velocidad de comunicación.

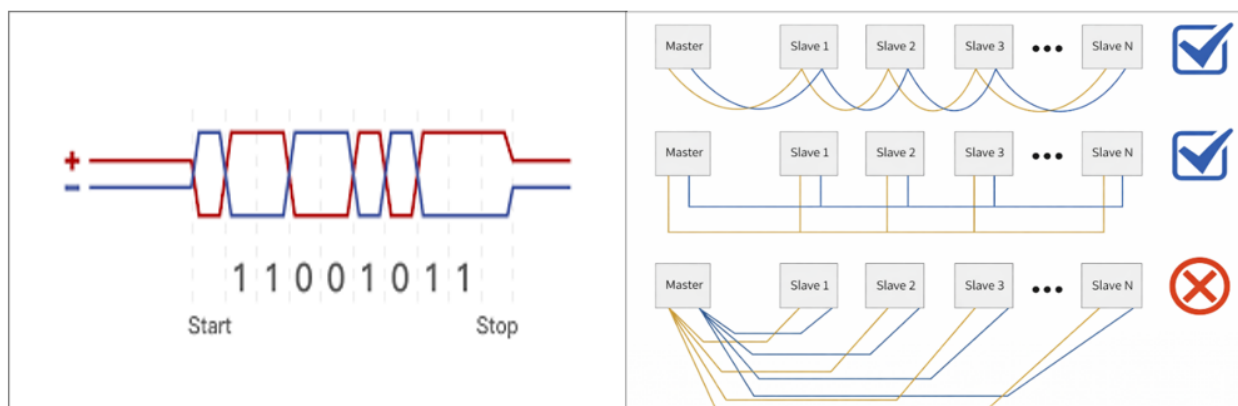


Figura 2.10: Comunicación RS485 y tipo de conexión

Fuente: Elaboración propia a partir de información de protocolos de comunicación

La central actúa como maestro, iniciando la comunicación y solicitando la medición al sensor a través del protocolo Modbus.

Modbus es un protocolo de comunicación industrial basado en el intercambio de mensajes estructurados, el cual define cómo se construyen, transmiten y validan los datos. La estructura de una trama Modbus RTU se compone de los siguientes campos:

[ID] [FUNCIÓN] [DATOS] [CRC]

Donde:

ID (Identificador del esclavo): Corresponde a la dirección del dispositivo dentro de la red, permitiendo identificar a qué sensor se dirige la solicitud.

FUNCIÓN (Código de operación): Define la acción que se desea realizar, como por ejemplo la lectura de registros.

DATOS (Información útil): Contiene los parámetros de la solicitud o la información enviada por el sensor, tales como direcciones de memoria o valores medidos.

CRC (Cyclic Redundancy Check): Es un código de verificación que permite detectar errores en la transmisión, asegurando la integridad de los datos.

El estándar RS-485 utiliza transmisión diferencial entre dos conductores (A y B), donde la información se representa mediante la diferencia de voltaje entre ambas líneas, reconociendo estados lógicos a partir de diferencias superiores a ± 200 mV y operando típicamente en rangos de $\pm 1,5$ a ± 3 V. Estos estados corresponden a valores digitales binarios, donde una diferencia positiva (B mayor que A) representa un '1' y una diferencia negativa (A mayor que B) representa un '0'. A partir de esta representación binaria es donde se construye y opera el protocolo Modbus, el cual utiliza estas secuencias de bits para estructurar la comunicación entre el sistema central y los sensores.

2.18 CABLE DE COMUNICACIÓN RS-485

El medio físico seleccionado para la transmisión de datos mediante el estándar RS-485 corresponde a un cable de instrumentación naval modelo XLPE/SHF1 – 150/250 (300 V) – 90 °C, conforme a la norma IEC 60092-376, tal como se muestra en la figura 2.11, diseñado para aplicaciones en embarcaciones y entornos marinos. Este cable posee conductores de cobre, aislamiento en XLPE y cubierta exterior libre de halógenos, lo que le otorga resistencia a

condiciones ambientales adversas, alta confiabilidad operativa y cumplimiento de normativas para uso en instalaciones mar dentro. Para la implementación del sistema se emplean dos cables independientes, uno destinado a la alimentación eléctrica y otro a la comunicación de datos, debido a las diferencias en los requerimientos eléctricos de cada señal. El cable de alimentación, con una sección de $2,5 \text{ mm}^2$, se utiliza para suministrar energía a los sensores del sistema, operando típicamente a un nivel de 24 V DC , estándar en instrumentación industrial. La mayor sección del conductor permite reducir la caída de tensión a lo largo del cable y soportar adecuadamente la corriente demandada por los dispositivos, evitando sobrecalentamientos y pérdidas de eficiencia.

Por otro lado, el cable de comunicación RS-485, con una sección de $0,5 \text{ mm}^2$, está diseñado para la transmisión de señales diferenciales de baja potencia. Según las características eléctricas del cable, este presenta una impedancia característica aproximada de 80Ω y una capacitancia del orden de 57 nF/km , valores adecuados para la transmisión de datos en redes RS-485, ya que permiten mantener la integridad de la señal y reducir la distorsión durante la propagación. En esta línea, la señal RS-485 opera típicamente con niveles de tensión diferencial en el rango de $\pm 1,5 \text{ V}$ a $\pm 5 \text{ V}$, por lo que no requiere conductores de gran sección. La utilización de un cable de menor sección optimiza la instalación sin comprometer el desempeño eléctrico. Finalmente, la separación física entre el cable de alimentación y el de comunicación permite minimizar el acoplamiento de interferencias electromagnéticas generadas por la corriente de potencia, mejorando así la confiabilidad del sistema de transmisión de datos.



Figura 2.11: Cable de comunicación y alimentación

Fuente: Elaboración propia a partir de la ficha técnica del fabricante, cable RS-485.

2.19 CONECTORES

Para la unión de los cables del sistema se opta por el uso de conectores desmontables, como se muestra en la figura 2.12, ya que estos permiten realizar de forma segura las labores de conexión y desconexión durante los períodos de mantenimiento de los sensores, así como en las operaciones de carga y descarga de material hacia los silos. De esta manera, se evita someter al sensor y al cableado a esfuerzos mecánicos innecesarios, reduciendo el riesgo de daño en los equipos. Debido a que el sistema utiliza dos tipos de cable con distinta sección transversal y distinta función, se define un conector específico para cada uno. Para la línea de alimentación, implementada con cable de 2,5 mm², se seleccionó un conector circular M16 de 2 pines, con capacidad de 10 A y grado de protección IP67. En cambio, para la línea de comunicación RS-485, implementada con cable de 0,5 mm², se escogió un conector M12, apto para señales de comunicación, con capacidad de 4 A a 250 V y protección IP67/IP68. La selección de estos conectores responde a criterios de compatibilidad eléctrica, facilidad de mantenimiento y resistencia ambiental. Además, ambos modelos cuentan con características de hermeticidad y protección adecuadas para su operación en ambiente marítimo, donde existe presencia de humedad, salinidad y condiciones exigentes de trabajo.



Figura 2.12: Conectores macho y hembra

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

2.20 CONVERTIDOR DE RS-485 A USB

La Raspberry Pi no dispone de una interfaz nativa para la lectura directa del estándar de comunicación RS-485, ya que este define únicamente la capa física de transmisión diferencial.

Por lo tanto, se requiere la incorporación de un módulo conversor RS-485 a USB, tal como se muestra en la figura 2.13, el cual permite adaptar las señales provenientes del bus a un formato compatible con los puertos de la Raspberry Pi. Este dispositivo realiza la conversión de niveles eléctricos y del tipo de señal, transformando la comunicación diferencial RS-485 en una interfaz USB serial (generalmente reconocida como un puerto `/dev/ttyUSB` en el sistema operativo). De esta manera, la Raspberry Pi puede acceder a los datos transmitidos por los sensores mediante software, utilizando protocolos de comunicación como Modbus RTU u otros definidos en el sistema.

Una vez recibida la información, el sistema central ejecuta el procesamiento de los datos, aplicando los tratamientos necesarios, tales como la interpretación de tramas, conversión de variables y cálculos aritméticos para obtener parámetros como nivel, distancia o volumen del material almacenado en los silos.



Figura 2.13: Módulo conversor Rs485 a USB

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

2.21 RASBERRY PI

La Raspberry Pi 4 (8GB) es un microcomputador de placa única (SBC) que integra un procesador Broadcom BCM2711 de cuatro núcleos ARM Cortex-A72 a 1.5 GHz, junto con 8 GB de memoria RAM LPDDR4, tal como se muestra en la figura 2.14, lo que le permite ejecutar aplicaciones de procesamiento y control en tiempo real. Su almacenamiento se realiza mediante tarjeta microSD, y dispone de múltiples interfaces de comunicación, incluyendo puertos USB 3.0 y 2.0, conexión Ethernet Gigabit, WiFi y Bluetooth.

Además, incorpora dos salidas micro-HDMI para video en alta resolución y un conector GPIO de 40 pines que permite la conexión directa con sensores, actuadores y módulos de comunicación como SPI, I2C y UART, siendo alimentada a través de un puerto USB-C de 5 V.

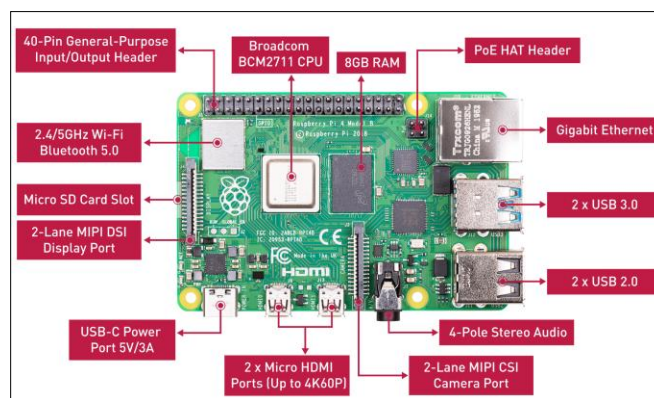


Figura 2.14:

Fuente: Elaboración propia a partir de la ficha técnica del fabricante.

2.22 SISTEMA OPERATIVO (RASBERRY PI OS – RASBIAN)

Raspberry Pi OS (anteriormente conocido como Raspbian) es el sistema operativo utilizado en la Raspberry Pi, el cual proporciona la interfaz mediante la cual el usuario puede interactuar con el equipo y gestionar sus recursos de hardware y software. Este sistema operativo está basado en Linux (Debian), y se caracteriza por ser liviano y optimizado para dispositivos embebidos, permitiendo un uso eficiente de los recursos disponibles.

En el contexto del sistema propuesto, Raspberry Pi OS permite la gestión de los puertos de comunicación, incluyendo la interfaz USB a la cual se conecta el conversor RS-485. A través de este, el sistema operativo reconoce el dispositivo como un puerto serial, posibilitando la recepción de datos provenientes de los sensores de nivel. Posteriormente, estos datos son procesados mediante un programa desarrollado en Python, el cual se encarga de interpretar las tramas de comunicación recibidas, aplicar los cálculos aritméticos definidos en el modelo del sistema como el cálculo de volumen, distancia y porcentaje de llenado y generar información útil para la supervisión del proceso. A continuación, se presenta la interfaz del sistema operativo en la figura 2.15.

Figura 2.15: Interfaz de sistema operativo
Fuente: *Elaboración propia a partir de sistema operativo Raspbian.*

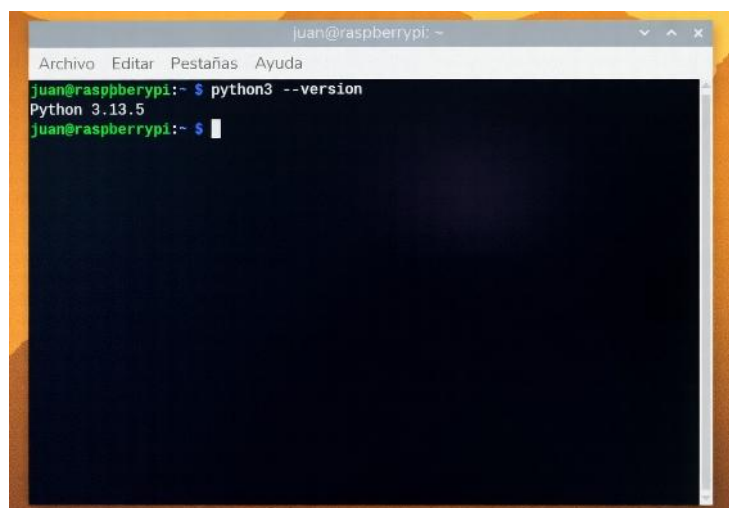
2.23 LENGUAJE DE PROGRAMACIÓN (PYTHON)

Python es el lenguaje de programación utilizado para el desarrollo del software del sistema. Corresponde a un lenguaje de alto nivel, ampliamente utilizado en aplicaciones de automatización, procesamiento de datos y sistemas embebidos, debido a su facilidad de implementación y amplia disponibilidad de librerías.

En este proyecto, Python se emplea para desarrollar el programa encargado de gestionar la comunicación con los sensores a través del puerto serial (proveniente del conversor RS-485 a USB). En este entorno se implementan las rutinas necesarias para la lectura e interpretación de las tramas de datos recibidas.

Además, el lenguaje permite ejecutar los cálculos aritméticos definidos en el modelo del sistema, tales como la determinación de distancia, volumen y porcentaje de llenado del silo. También se incorporan estructuras de control para la detección de errores, validación de datos y gestión de fallas. Adicionalmente, Python facilita la integración con librerías externas y herramientas como Node-RED, permitiendo la actualización del sistema, el manejo de datos y la comunicación con la interfaz de visualización.

Por defecto, este programa se encuentra previamente instalado en el sistema operativo. Para verificar su instalación, se debe acceder a la terminal e ingresar el siguiente comando: `python3 --version`, tal como se muestra en la figura 2.16.



```
juan@raspberrypi: ~  
Archivo Editar Pestañas Ayuda  
juan@raspberrypi:~ $ python3 --version  
Python 3.13.5  
juan@raspberrypi:~ $
```

Figura 2.16: Verificación de instalación Python

Fuente: Elaboración propia a partir del lenguaje de programación Python.

Posteriormente, es necesario crear una carpeta de trabajo y los archivos con la extensión correspondiente (.py) para iniciar el desarrollo del programa, lo cual se detalla a continuación en la figura 2.17.

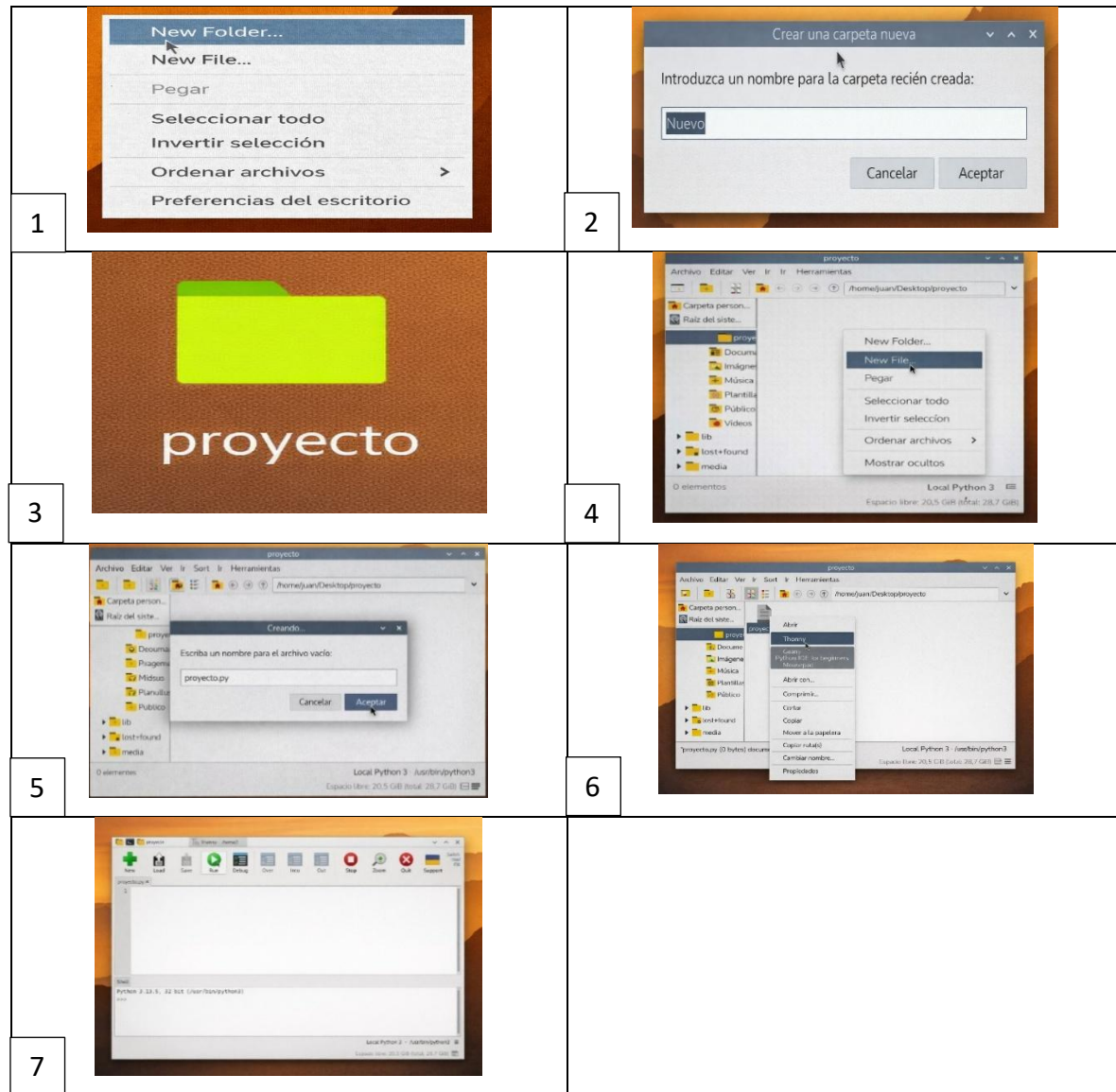


Figura 2.17: Pasos para creación de archivo Python
Fuente: Elaboración propia a partir de sistema operativo Raspbian.

El recuadro siete de la figura 2.17. muestras el programa Thonny, un entorno de desarrollo que viene incluido en el sistema operativo el cual permite escribir y probar código en Python de forma sencilla antes de su ejecución definitiva desde la terminal.

2.24 PROGRAMA DESARROLLADO DESDE RASBERRY PI EN PYTHON

A continuación, se describen las principales etapas de programación implementadas en Python para el funcionamiento de los distintos módulos que componen el sistema:

2.25 LLAMADA DEL SENSOR POR MODBUS Y PETICIÓN DE LA INFORMACION

Mediante un código de programación en Python, se realiza la comunicación con cada sensor a través del protocolo Modbus. El sistema central consulta secuencialmente a cada dispositivo, solicitando el valor de la variable medida. Una vez recibida la respuesta, el dato es procesado por el programa y posteriormente se continúa con la consulta del siguiente sensor. A continuación, se presenta el código en la figura 2.18.

```
Python
from pymodbus.client import ModbusSerialClient
import time

# Configuración del puerto serial RS-485 / Modbus RTU
client = ModbusSerialClient(
    port='/dev/ttyUSB0', # Puerto del conversor RS-485
    baudrate=9600, # Velocidad de comunicación
    parity='N', # Paridad: N = None
    stopbits=1, # Bits de parada
    bytesize=8, # Bits de datos
    timeout=1 # Tiempo de espera en segundos
)

# Lista de direcciones Modbus de los sensores
sensores = [1, 2, 3, 4]

# Registro donde el sensor entrega la variable medida
registro_medicion = 0
cantidad_registros = 1

if client.connect():
    print("Comunicación Modbus iniciada correctamente\n")

    for sensor_id in sensores:
        print(f"Consultando sensor {sensor_id}...")

        print(f"Consultando sensor {sensor_id}...")

        respuesta = client.read_holding_registers(
            address=registro_medicion,
            count=cantidad_registros,
            slave=sensor_id
        )

        if not respuesta.isError():
            valor_medido = respuesta.registers[0]
            print(f"Sensor {sensor_id}: valor medido = {valor_medido}")
        else:
            print(f"Error al leer el sensor {sensor_id}")

        time.sleep(0.5) # Pausa breve antes de consultar el siguiente sensor

    client.close()

else:
    print("No se pudo establecer la comunicación con la red Modbus")
```

Figura 2.18: Solicitud de información por ModBus
Fuente: Elaboración propia a partir del lenguaje de programación Python.

2.26 VALIDACIÓN DE DATOS Y CONTROL DE ERRORES

El software incorpora condiciones de validación, tal como se muestra en la Figura 2.19, para verificar la integridad de los datos recibidos, detectando posibles errores de comunicación, ausencia de señal o lecturas fuera de rango. Esto permite aumentar la confiabilidad del sistema y mejorar la detección de fallas operativas.



```
Python

# Rango válido de medición
valor_minimo = 0
valor_maximo = 10000

def validar_datos(respuesta, sensor_id):
    # Verifica si hubo error de comunicación Modbus
    if respuesta.isError():
        print(f"Sensor {sensor_id}: error de comunicación Modbus")
        return None

    # Verifica si la respuesta contiene registros
    if not hasattr(respuesta, 'registers') or len(respuesta.registers) == 0:
        print(f"Sensor {sensor_id}: ausencia de datos o señal")
        return None

    valor = respuesta.registers[0]

    # Verifica si el valor está fuera de rango
    if valor < valor_minimo or valor > valor_maximo:
        print(f"Sensor {sensor_id}: lectura fuera de rango -> {valor}")
        return None

    return valor

1
```

```
Python

if client.connect():
    print("Comunicación Modbus iniciada correctamente\n")

    for sensor_id in sensores:
        print(f"consultando sensor {sensor_id}...")

        try:
            respuesta = client.read_holding_registers(
                address=registro_medicion,
                count=cantidad_registros,
                slave=sensor_id
            )

            valor_medido = validar_datos(respuesta, sensor_id)

            if valor_medido is not None:
                print(f"Sensor {sensor_id}: lectura válida = {valor_medido}")
            else:
                print(f"Sensor {sensor_id}: dato descartado por error de validación")

        except Exception as e:
            print(f"Sensor {sensor_id}: fallo de comunicación o excepción -> {e}")

    time.sleep(0.5)

2
```

```
3

    time.sleep(0.5)

    client.close()

else:
    print("No se pudo establecer la comunicación con la red Modbus")
```

Figura 2.19: Validación de datos

Fuente: Elaboración propia a partir del lenguaje de programación Python.

2.27 CÁLCULO DE VARIABLES DEL SISTEMA

Dentro del programa se implementan las instrucciones necesarias para procesar los datos recibidos y ejecutar los cálculos aritméticos definidos en el modelo matemático del proyecto, tal como se muestra en la figura 2.20. Entre estas variables se consideran, por ejemplo, la distancia medida, el volumen estimado, el porcentaje de llenado y otras magnitudes requeridas para la supervisión del nivel de alimento almacenado en los silos.

```
Python 1
def calcular_volumen(h, Ac, Ab, H1, H2, Acuadrado):
    # Tramo 1
    if 0 <= h <= H2:
        volumen = (h/3) * (Ac + (Ab - Ac)*(h**2 / H2**2) +
                           math.sqrt(Ac * (Ac + (Ab - Ac)*(h/H2)**2)))

    # Tramo 2
    elif H2 < h <= (H2 + H1):
        volumen = (H2/3) * (Ac + Ab + math.sqrt(Ac * Ab)) + \
                  Acuadrado * (h - H2)

    else:
        volumen = None

    return volumen

def calcular_porcentaje_lleno(volumen_actual, volumen_total):
    if volumen_actual is None or volumen_total <= 0:
        return None
    porcentaje = (volumen_actual / volumen_total) * 100
    return porcentaje

Python 2
# DATOS DEL SISTEMA
# -----
distancia_medida = 1.80 # distancia entregada por el sensor
altura_total = 5.00 # altura total interna del silo

Ac = 1.20
Ab = 2.80
H1 = 3.00
H2 = 2.00
Acuadrado = 2.80

# Altura útil de alimento
h = altura_total - distancia_medida

# Cálculo del volumen actual
volumen_actual = calcular_volumen(h, Ac, Ab, H1, H2, Acuadrado)

# Cálculo del volumen máximo del silo
volumen_total = calcular_volumen(H1 + H2, Ac, Ab, H1, H2, Acuadrado)

# Cálculo del porcentaje de llenado
porcentaje_lleno = calcular_porcentaje_lleno(volumen_actual, volumen_total)

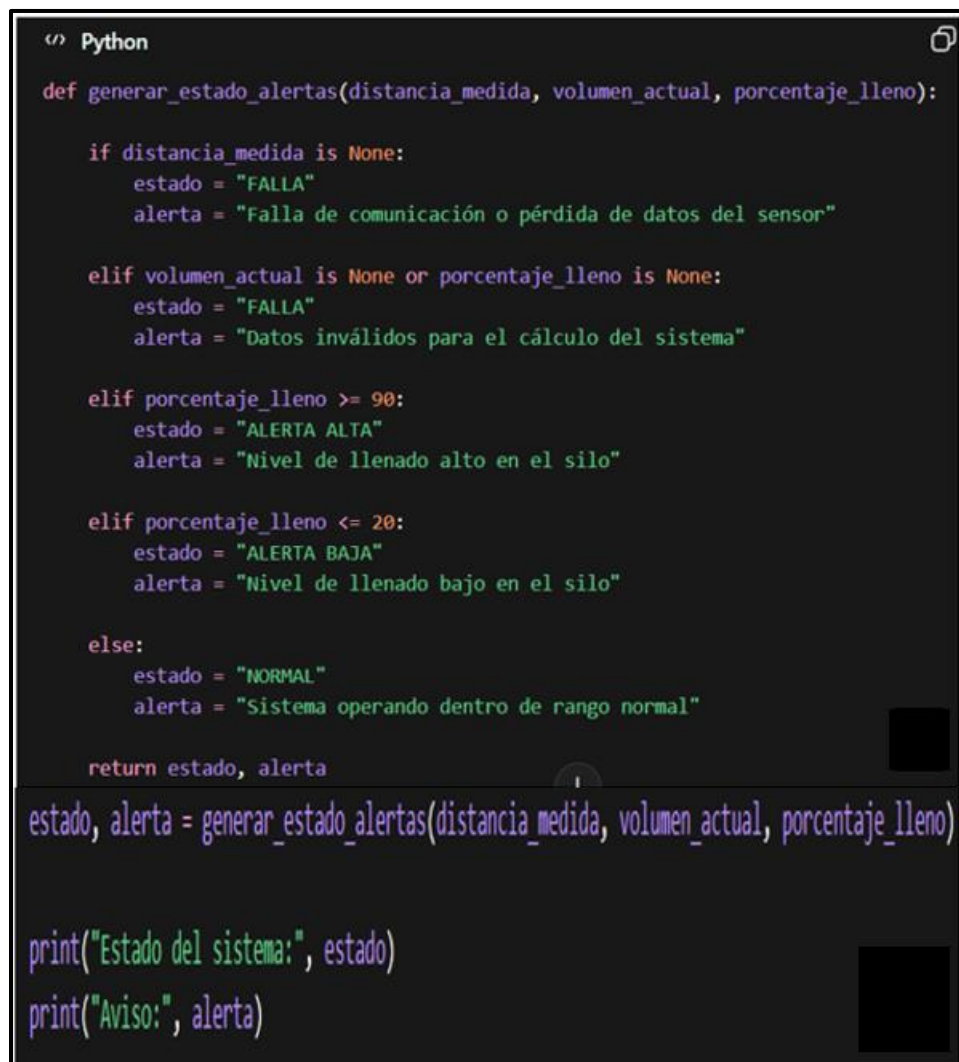
Python 3
# Mostrar resultados
print("Distancia medida por el sensor:", distancia_medida, "m")
print("Altura útil de alimento:", h, "m")
print("Volumen estimado:", volumen_actual, "m3")
print("Porcentaje de llenado:", porcentaje_lleno, "%")
```

Figura 2.20: Cálculo de Variables

Fuente: Elaboración propia a partir del lenguaje de programación Python.

2.28 GENERACIÓN DE ESTADOS, ALERTAS Y AVISOS

Se implementan condiciones lógicas destinadas a evaluar el estado operativo del sistema a partir de las variables calculadas y de la disponibilidad de datos, tal como se muestra en la figura 2.21. De esta forma, el programa genera automáticamente estados, alertas y avisos asociados a niveles altos o bajos de llenado, así como también a fallas de comunicación.



```
Python
def generar_estado_alertas(distancia_medida, volumen_actual, porcentaje_lleno):

    if distancia_medida is None:
        estado = "FALLA"
        alerta = "Falla de comunicación o pérdida de datos del sensor"

    elif volumen_actual is None or porcentaje_lleno is None:
        estado = "FALLA"
        alerta = "Datos inválidos para el cálculo del sistema"

    elif porcentaje_lleno >= 90:
        estado = "ALERTA ALTA"
        alerta = "Nivel de llenado alto en el silo"

    elif porcentaje_lleno <= 20:
        estado = "ALERTA BAJA"
        alerta = "Nivel de llenado bajo en el silo"

    else:
        estado = "NORMAL"
        alerta = "Sistema operando dentro de rango normal"

    return estado, alerta

estado, alerta = generar_estado_alertas(distancia_medida, volumen_actual, porcentaje_lleno)

print("Estado del sistema:", estado)
print("Aviso:", alerta)
```

Figura 2.21: Alertas y advertencias

Fuente: Elaboración propia a partir del lenguaje de programación Python.

El sistema incorpora rutinas de comunicación LoRa orientadas a establecer un enlace inalámbrico entre pontones, como se muestra en la figura 2.22 y 2.23. En esta arquitectura, los pontones secundarios transmiten sus datos al pontón maestro, donde una Raspberry Pi centraliza la información recibida y la reenvía a la red mediante conexión a internet, permitiendo así la supervisión remota del sistema completo.

```
Python
import serial
import time

# Configuración del módulo LoRa conectado por USB
lora = serial.Serial(
    port="/dev/ttyUSB0",
    baudrate=9600,
    timeout=1
)

def enviar_datos_lora(id_ponton, id_silo, distancia, volumen, porcentaje):
    mensaje = f"{id_ponton},{id_silo},{distancia:.2f},{volumen:.2f},{porcentaje:.2f}\n"
    lora.write(mensaje.encode('utf-8'))
    print("Datos enviados:", mensaje.strip())

# Ejemplo de transmisión periódica
while True:
    id_ponton = "PONTON_2"
    id_silo = "SILO_1"
    distancia = 1.85
    volumen = 12.40
    porcentaje = 68.30

    enviar_datos_lora(id_ponton, id_silo, distancia, volumen, porcentaje)
    time.sleep(5)
```

Figura 2.22: Código para pontón secundario

Fuente: Elaboración propia a partir del lenguaje de programación Python.

```
import serial

# Configuración del módulo LoRa conectado por USB en el pontón maestro
lora = serial.Serial(
    port="/dev/ttyUSB0",
    baudrate=9600,
    timeout=1
)

def recibir_datos_lora():
    if lora.in_waiting > 0:
        mensaje = lora.readline().decode('utf-8').strip()
        return mensaje
    return None

while True:
    dato_recibido = recibir_datos_lora()

    if dato_recibido:
        print("Datos recibidos:", dato_recibido)
        partes = dato_recibido.split(',')

        id_ponton = partes[0]
        id_silo = partes[1]
        distancia = float(partes[2])
        volumen = float(partes[3])
        porcentaje = float(partes[4])

        print("Pontón:", id_ponton)
        print("Silo:", id_silo)
        print("Distancia:", distancia, "m")
        print("Volumen:", volumen, "m3")
        print("Porcentaje:", porcentaje, "%")
        print("-----")
```

Figura 2.23: Código para pontón Maestro

Fuente: Elaboración propia a partir del lenguaje de programación Python.

2.29 NODE-RED Y ACTIVACIÓN DEL SERVIDOR

Node-RED es una herramienta de software de código abierto, utilizada para la integración, procesamiento y visualización de datos mediante una interfaz de programación basada en flujos. En el sistema propuesto, se implementa como un servicio ejecutado en la Raspberry Pi, funcionando como un servidor local.

Una vez instalado, Node-RED se ejecuta en la Raspberry Pi y levanta un servidor accesible a través de la red, permitiendo su visualización tanto de forma local como remota mediante un navegador web. Para ello, la Raspberry Pi debe contar con conexión a red (LAN o WiFi), lo que posibilita el acceso a la interfaz desde otros dispositivos dentro de la misma red o mediante acceso remoto configurado. Node-RED se encarga de recibir las variables previamente procesadas por el programa desarrollado en Python, tales como nivel, volumen y porcentaje de llenado de los silos. Estos datos son utilizados para actualizar en tiempo real los elementos gráficos del sistema.

A través de esta plataforma, se desarrolla una interfaz de usuario amigable y adaptable, permitiendo la visualización del estado de los silos mediante indicadores gráficos, paneles de control y menús interactivos. Entre las funcionalidades implementadas se incluyen la visualización de niveles en porcentaje y volumen, generación de alertas visuales y estado del sistema.

2.30 IMPLEMENTACIÓN

Se debe instalar Node-RED en la Raspberry Pi. Para ello, se debe abrir una terminal de comandos e ingresar el comando correspondiente de instalación según en la figura 2.24, el cual permite descargar e instalar automáticamente el entorno necesario para su funcionamiento.

```
bash <(curl -sL https://github.com/node-red/linux-installers/releases/latest/download/update-nodejs-and-nodered-deb)
```

Figura 2.24: Enlace de instalación

Fuente: Elaboración propia a partir del lenguaje de programación Python.

Una vez finalizada la instalación del programa, tal como se muestra en la figura 2.25, se debe ingresar el siguiente comando para iniciar el servidor de Node-RED: *node-red-start*, como se muestra en la figura 2.26.

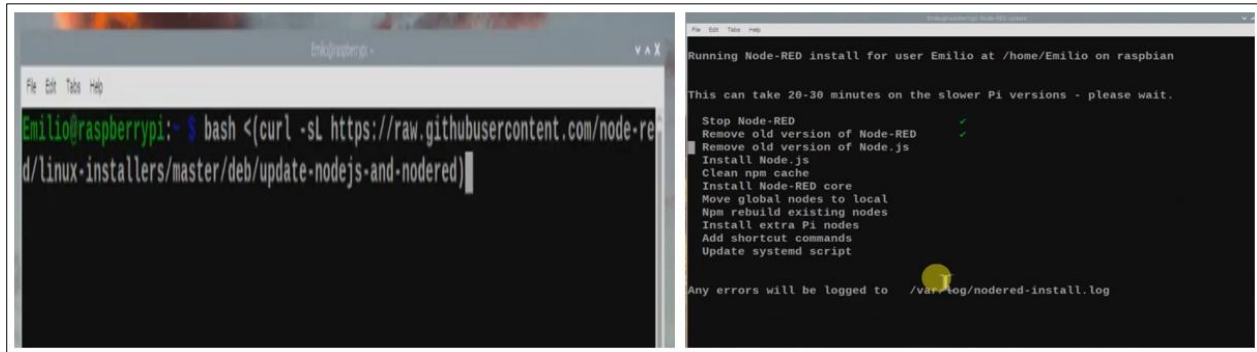


Figura 2.25: instalación finalizada

Fuente: Elaboración propia a partir del lenguaje de programación Python.

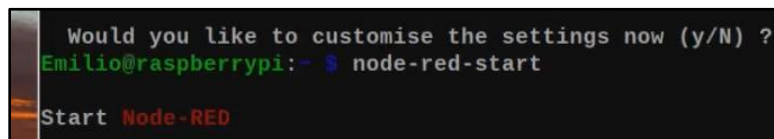


Figura 2.26: Iniciar servidor

Fuente: Elaboración propia a partir del lenguaje de programación Python.

El siguiente paso corresponde al acceso al servidor, el cual se realiza utilizando la dirección proporcionada en la terminal una vez iniciado Node-RED, como se muestra en la figura 2.27.

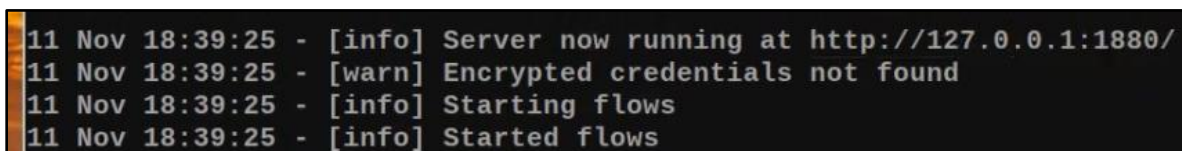


Figura 2.27: Acceso a servidor

Fuente: Elaboración propia a partir del lenguaje de programación Python.

Como se aprecia en la figura 2.27, la dirección `http://127.0.0.1:1880` permite acceder a la interfaz de desarrollo de Node-RED, la cual debe ser ingresada en el navegador web de la Raspberry Pi. Una vez ingresada, el sistema redirige automáticamente a la plataforma donde se realiza la creación del dashboard y la integración de las variables mediante el uso de nodos. Como se aprecia en la figura 2.28.



Figura 2.28: Espacio de trabajo Node Red

Fuente: Elaboración propia a partir de información técnica de Node-RED

2.31 TAILSCALE

Sin embargo, la dirección IP pública puede cambiar debido a que es asignada dinámicamente por el proveedor de red (ISP), especialmente en conexiones mediante routers o módulos 4G. Esta variación impide mantener un acceso remoto estable al servidor Node-RED, ya que la dirección IP no es persistente y, en algunos casos, el dispositivo puede estar detrás de CG-NAT, lo que dificulta o incluso impide el acceso directo desde internet.

Tailscale es una solución de red privada virtual (VPN) basada en el protocolo WireGuard, que permite interconectar dispositivos a través de internet como si estuvieran en una misma red local. En el contexto del sistema propuesto, Tailscale se utiliza como solución al problema de direcciones IP dinámicas y restricciones de acceso derivadas del uso de redes 4G y CG-NAT, ya que asigna a cada dispositivo una dirección IP privada fija dentro de la red virtual, permitiendo así un acceso remoto estable al servidor Node-RED. Adicionalmente, Tailscale proporciona seguridad a nivel de red mediante el cifrado de extremo a extremo y el control de acceso entre dispositivos, evitando la exposición directa de los servicios a internet. En el plan gratuito, la plataforma permite hasta 100 dispositivos y 3 usuarios, lo cual resulta suficiente para la interconexión de los pontones y la unidad central del sistema.

En este esquema, Tailscale se emplea principalmente para la comunicación segura entre los dispositivos, mientras que la visualización del monitoreo por parte de múltiples usuarios se gestiona a través del servidor Node-RED, permitiendo el acceso concurrente mediante navegador web sin requerir múltiples cuentas dentro de la red VPN. No obstante, esta solución debe complementarse con mecanismos de seguridad a nivel de aplicación, como la autenticación en Node-RED, para garantizar una protección integral del sistema.

2.32 PASOS PARA INSTALAR TAILSCALE EN RASBERRY

Se accede a la página oficial de descargas de Tailscale mediante el siguiente enlace: <https://tailscale.com/download/linux>. En dicha página se debe crear una cuenta de usuario utilizando un correo de la institución y seleccionar la opción correspondiente a sistemas Linux, donde se proporciona el comando de instalación. Este comando debe ser copiado y ejecutado en la terminal de la Raspberry Pi para llevar a cabo la instalación del software, tal como se muestra en la figura 2.29.

Comando de instalación en Raspberry pi, copiar y pegar en la terminal:

`curl -fsSL https://tailscale.com/install.sh | sh`

```
File Edit Tabs Help
Setting up iptables (1.8.9-2) ...
update-alternatives: using /usr/sbin/iptables-legacy to provide /usr/sbin/iptables (iptables) in auto mode
update-alternatives: using /usr/sbin/ip6tables-legacy to provide /usr/sbin/ip6tables (ip6tables) in auto mode
update-alternatives: using /usr/sbin/iptables-nft to provide /usr/sbin/iptables (iptables) in auto mode
update-alternatives: using /usr/sbin/ip6tables-nft to provide /usr/sbin/ip6tables (ip6tables) in auto mode
update-alternatives: using /usr/sbin/arptables-nft to provide /usr/sbin/arptables (arptables) in auto mode
update-alternatives: using /usr/sbin/ebtables-nft to provide /usr/sbin/ebtables (ebtables) in auto mode
Setting up tailscale (1.66.4) ...
Created symlink /etc/systemd/system/multi-user.target.wants/tailscaled.service → /lib/systemd/system/tailscaled.service.
Processing triggers for man-db (2.11.2-2) ...
Processing triggers for libc-bin (2.36-9+rpt2+deb12u4) ...
+ [ false = true ]
+ set +x
Installation complete! Log in to start using Tailscale by running:
```

Figura 2.29: Visualización de Proceso de instalación Tailscale
Fuente: Elaboración propia a partir del lenguaje de programación Python.

Una vez finalizada la instalación de Tailscale, se debe ejecutar el siguiente comando en la terminal para iniciar el servicio y vincular el dispositivo a la red: `sudo tailscale up`, como se muestra en la figura 2.20. Al ejecutar este comando, el sistema genera una URL de autenticación. Esta dirección debe ser abierta en un navegador web, donde el usuario deberá iniciar sesión con su cuenta de Tailscale. Al completar este proceso, la Raspberry Pi queda registrada y asociada a la red privada virtual, apareciendo en la lista de dispositivos dentro de la cuenta del usuario.



Figura 2.30: Vinculación Raspberry a la Red
Fuente: Elaboración propia a partir del lenguaje de programación Python.

Una vez finalizado el proceso de instalación y vinculación, es posible visualizar dentro de la plataforma de Tailscale las direcciones IP asignadas a cada dispositivo conectado, como se muestra en la figura 2.31. Estas direcciones corresponden a IP privadas virtuales (rango 100.x.x.x), las cuales permanecen fijas y no dependen de la red local. Esta característica permite acceder de forma remota y segura al sistema de monitoreo, ya que cada dispositivo mantiene una dirección constante, facilitando la conexión directa.

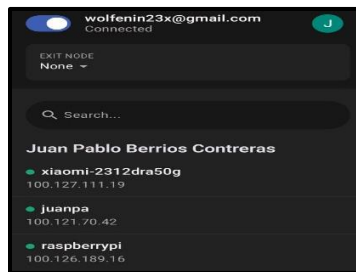


Figura 2.31: Vista de aplicación de Tailscale

Fuente: Elaboración propia a partir de la aplicación móvil de Tailscale en Android.

Finalmente, para acceder al servidor de Node-RED de forma remota, se debe ingresar en el navegador web de cualquier dispositivo conectado a la red de Tailscale la dirección IP virtual asignada a la Raspberry Pi, por ejemplo 100.126.189.16. A dicha dirección se le debe agregar el puerto 1880, que corresponde al servicio de Node-RED. Por lo tanto, el acceso se realiza mediante la siguiente estructura como se muestra en la figura 2.32:



Figura 2.32: Acceso a Node Red por Tailscale

Fuente: Elaboración propia a partir de información técnica de Node-RED

2.33 LLAMADO DE VARIABLES Y DISEÑO DE DASHBOARD

Como se analizó anteriormente, Python es el encargado de realizar el cálculo de las variables del sistema, mientras que Node-RED cumple la función de recibir estos datos y permitir su visualización en la interfaz. Para que este proceso sea posible, es necesario establecer un mecanismo de comunicación entre ambos. Para ello, se crea una variable denominada “dato”, encargada de almacenar la información y darle el formato necesario para posteriormente estructurarla en JSON. Este proceso sigue una secuencia definida que se describe a continuación en la figura 2.33.



Figura 2.33: Creación de Variable Dato

Fuente: Elaboración propia a partir del lenguaje de programación Python.

JSON (JavaScript Object Notation) es un formato de texto ligero que permite organizar datos en pares clave–valor. Se utiliza para agrupar y enviar múltiples variables de forma ordenada entre sistemas, facilitando su interpretación y uso en aplicaciones como Node-RED. Para realizar la conversión, es necesario instalar una librería en Python. Luego, se utiliza el comando “`datos_json = json.dumps(datos)`” para dejar la información lista para su envío, como se aprecia en la figura 2.34.



Figura 2.34: Librería y Comando Json

Fuente: Elaboración propia a partir del lenguaje de programación Python.

En cuanto a la mecánica de comunicación, se debe instalar la librería “import paho.mqtt.client as mqtt”, y utilizar los siguientes comandos de la figura 2.35 para enviar la información.

```
</> Python

import paho.mqtt.client as mqtt

client = mqtt.Client()
client.connect("localhost", 1883, 60)
client.publish("silo/datos", datos_json)
client.disconnect()
```

Figura 2.35: Instalación de librería y comando MQTT

Fuente: Elaboración propia a partir del lenguaje de programación Python.

El envío de los datos hacia Node-RED se realiza mediante el protocolo MQTT, el cual opera bajo un modelo de publicación y suscripción, donde Python envía los datos a un tópico y Node-RED los recibe al estar suscrito a este.

Y finalmente, se presenta la ilustración de la configuración del dashboard en Node-RED en la figura 2.36, donde se muestran los nodos encargados de la recepción de la información y de la visualización de los datos.

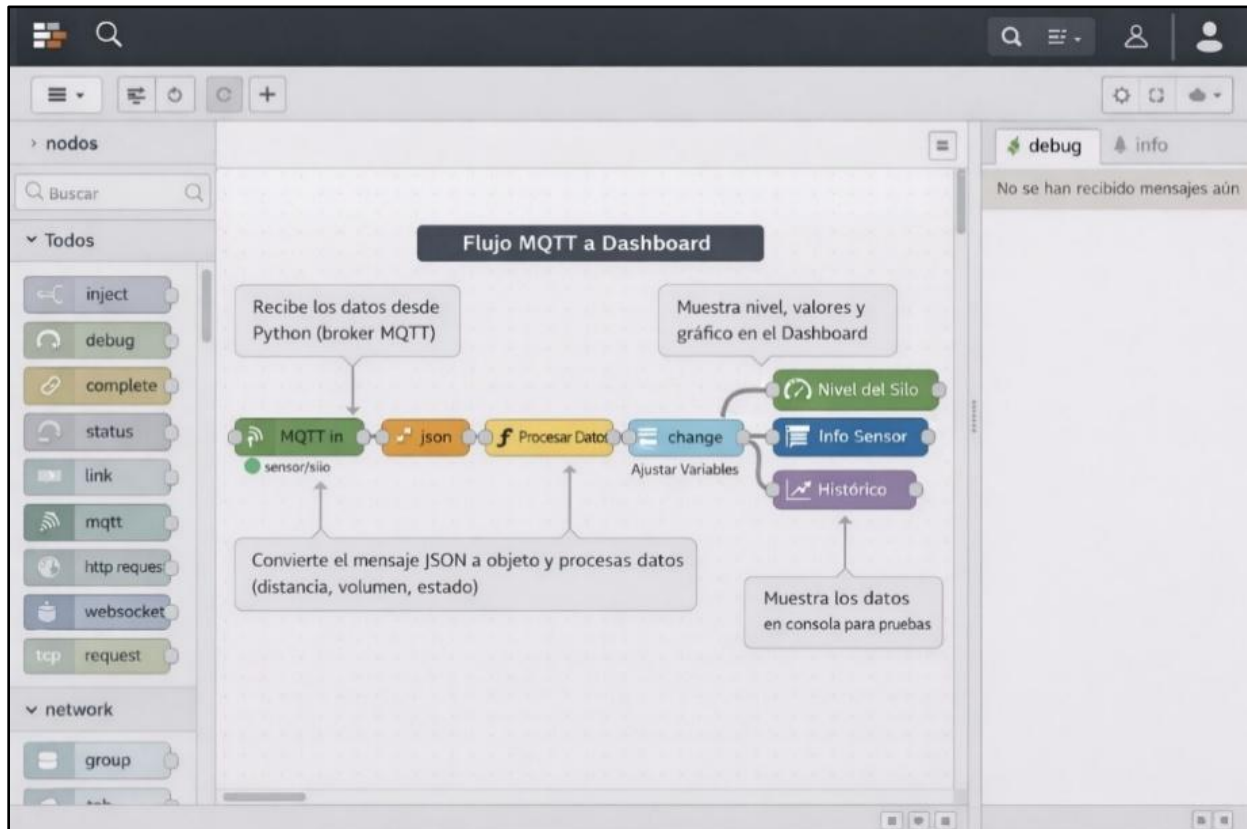


Figura 2.36: Configuración nodos dashboard
Fuente: Elaboración propia a partir de información técnica de Node-RED

Los principales nodos utilizados en Node-RED y sus respectivas funciones se presentan en la figura 2.37, los cuales permiten recibir, procesar y visualizar la información proveniente de los sensores dentro del sistema de monitoreo.

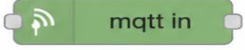
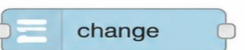
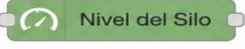
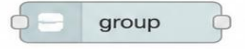
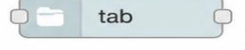
Nodo (Node-RED)	Descripción
 mqtt in	Recibe los mensajes publicados desde un broker MQTT en el tópico configurado. Actúa como la entrada de datos proveniente de los sensores o del dispositivo de adquisición.
 json	Convierte el mensaje en formato JSON (cadena de texto) en un objeto JavaScript. Permite trabajar los datos estructurados de manera más sencilla dentro del flujo.
 Procesar Datos	Ejecuta código JavaScript para procesar la información recibida. En este nodo se realizan cálculos como conversión de distancia a volumen, porcentaje de llenado y determinación del estado del silo.
 change	Modifica propiedades del mensaje, asigna nuevas variables o ajusta el formato de los datos. En este flujo se utiliza para preparar la información que será enviada a los distintos elementos del dashboard.
 debug	Muestra en la consola de depuración los mensajes que pasan por el flujo. Es utilizado para pruebas, validación y seguimiento de los datos en tiempo real.
 Nivel del Silo	Elemento del dashboard (ui_gauge) que muestra de forma visual el nivel del silo en porcentaje mediante un medidor o indicador circular.
 Info Sensor	Elemento del dashboard (ui_text) que presenta información textual del sensor, como distancia medida, volumen calculado o estado actual del sistema.
 Histórico	Elemento del dashboard (ui_chart) que genera un gráfico histórico con la evolución del nivel del silo a lo largo del tiempo.
 group	Nodo del dashboard que permite agrupar elementos (widgets) dentro de una misma sección o panel para organizar la interfaz.
 tab	Nodo del dashboard que crea pestañas dentro de la interfaz web, permitiendo organizar diferentes vistas o secciones del sistema.

Figura 2.37: Descripción funciones de nodos en Node Red
Fuente: Elaboración propia a partir de información técnica de Node-RED

A continuación, se presenta un ejemplo del diseño utilizado para la visualización de los niveles de alimento (pellet) en los silos del pontón, mediante una interfaz gráfica de monitoreo. Como se muestra en la figura 2.38.

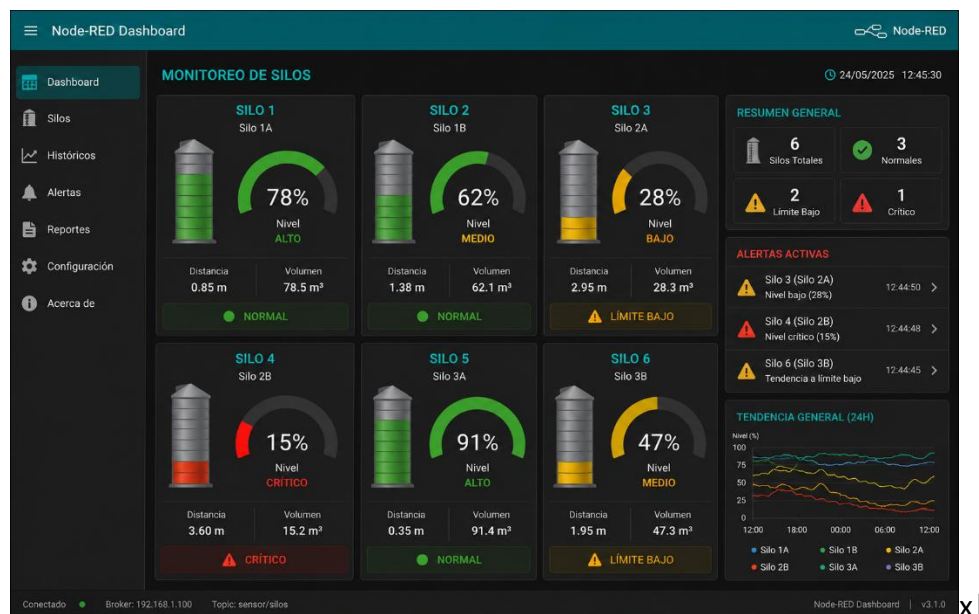


Figura 2.38: visualización de ejemplo de los silos y porcentaje de llenado
 Fuente: Elaboración propia a partir de información técnica de Node-RED

Se presenta una visión general de la arquitectura del sistema de medición por silo. Cada silo incorpora un sensor dedicado a la adquisición y transmisión de datos, mientras que el sistema maestro se ubica en la cabina de control y centraliza el procesamiento y la visualización de la información. Como referencia, se emplea un plano arquitectónico del pontón, como se muestra en la figura 2.39 para indicar la disposición física de los equipos y la ubicación exacta de cada dispositivo en terreno y en la figura 2.40 se muestra el diseño final del sistema.



Figura 2.39: Arquitectura general de un Ponton
Fuente: Elaboración propia a partir de Phillips y Santa María (2008).

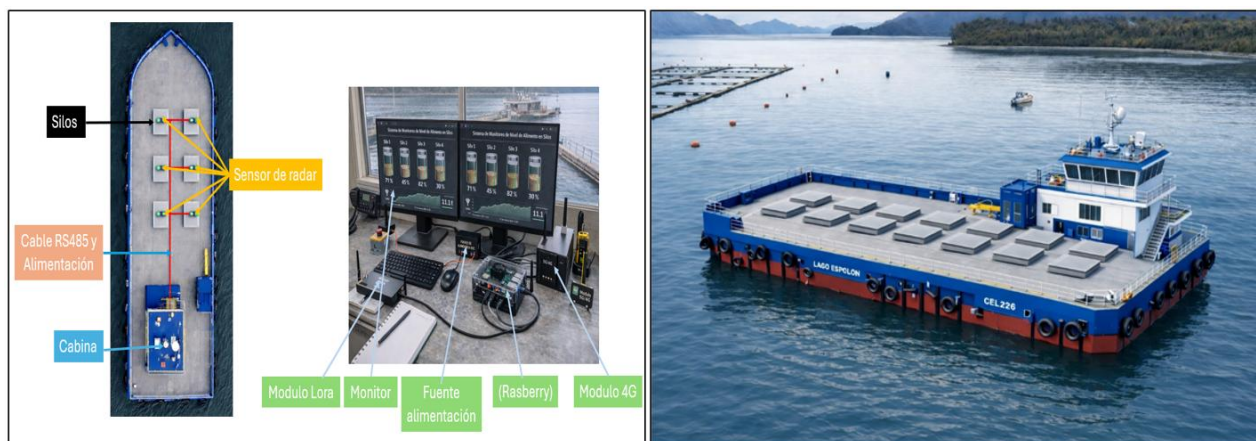


Figura 2.40: Vista lateral de Ponton y Diseño final del sistema
Fuente: Elaboración propia mediante inteligencia artificial generativa.

La arquitectura de comunicación propuesta se basa en una configuración maestro-esclavo, tal como se muestra en la figura 2.41, donde un pontón maestro centraliza la comunicación con los distintos pontones esclavos del sistema. Por otra parte, en la figura 2.42 se presenta el plano P&ID correspondiente a un pontón.

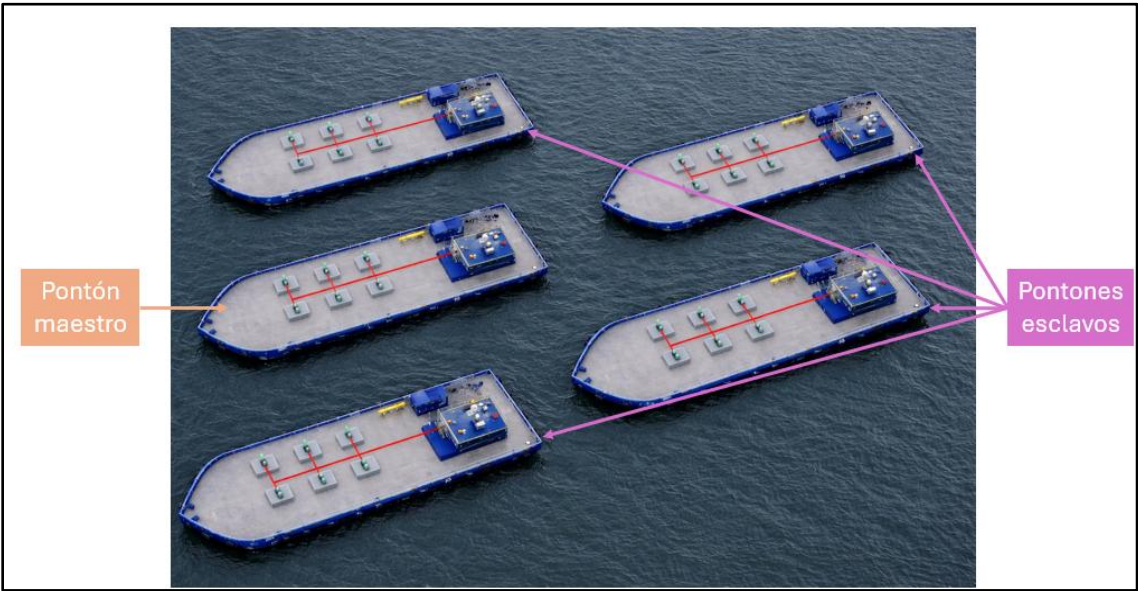


Figura 2.41: Arquitectura de comunicación entre pontones con LoRa y 4G
Fuente: Elaboración propia mediante inteligencia artificial generativa.

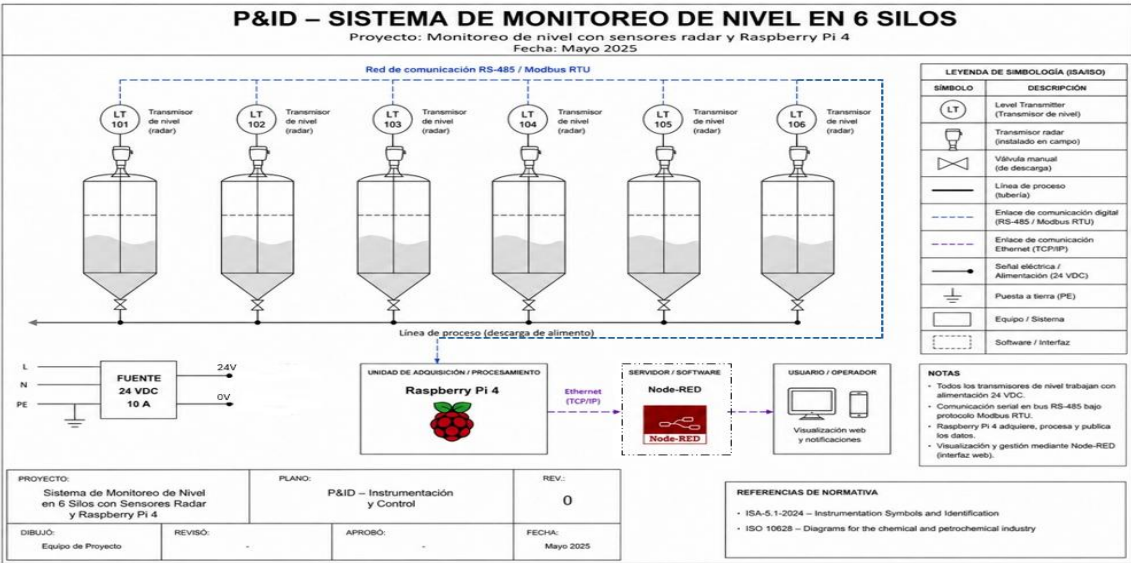


Figura 2.42: P&ID de un pontón
Fuente: Elaboración propia en base al desarrollo del proyecto.

CAPÍTULO 3: EVALUACIÓN ECONÓMICA DEL PROYECTO

3. Evaluación económica del proyecto

En este capítulo se darán a conocer la organización de este proyecto y el tiempo necesario para poder llevarlo a cabo, además se dan a conocer las posibilidades de ganancia o perdidas al implementarlo, también la explicación de los distintos tipos de costos que se verán afectados. Por otra parte, se dará a conocer el costo monetario de cada una de las herramientas, trabajadores y materiales necesarios para poder llevar a cabo este proyecto.

Todos los valores en este proyecto son calculados bajo el valor de (pesos chilenos) y (UF: 40.200,12 al día 06 de mayo 2026).

3.1 CARTA GANTT

La carta Gantt es una herramienta de planificación del trabajo, la cual hace posible llevar un orden respecto a las fechas de comienzo y término de las distintas actividades a realizar en un proyecto. En la tabla 3.1 se aprecia la Carta Gantt realizada para la planificación y desarrollo del proyecto previsto en esta tesis.

Tabla 3.1: Gantt del proyecto

actividad/semana	semana 1	semana 2	semana 3	semana 4	semana 5	semana 6	semana 7	semana 8	semana 9
visita a terreno	20 horas								
cotizacion, contacto proveedores y adquisicion materiales	20 horas								
montaje de sensores y cables		40 horas	40 horas						
montaje de monitores y CPU			16 horas						
montaje de fuente de alimentacion principal			8 horas	8 horas					
instalacion de sistema operativo CPU			8 horas						
prueba de comunicación 1 cpu-sensores				40 horas					
programacion e instalacion de software					40 horas	40 horas			
diseño de dashboard							40 horas		
prueba de funcionamiento final								40 horas	
entrega del proyecto al encargado de pontones									20 horas
Total hora ingeniero	332 horas								
Total hora tecnico	48 horas								

Fuente: Elaboración propia en Excel.

Como se puede ver en la tabla 3.1 de la carta Gantt, el color amarillo muestra las funciones del ingeniero en control de este proyecto y el color celeste muestra el técnico en control encargado de la implementación en terreno del proyecto y apoyo para el ingeniero, con esto se sacan las siguientes conclusiones.

Horas Trabajadas por el ingeniero: 332 horas.

Horas trabajadas por los técnicos: 48 horas.

3.2 COSTOS DE MANO DE OBRA

La mano de obra corresponde al esfuerzo físico y mental requerido para ejecutar las actividades del proyecto, representando el costo asociado al personal encargado de su desarrollo. En este caso, las tareas se distribuyen entre distintos trabajadores para labores de diseño, montaje, integración y puesta en marcha del sistema. La evaluación se realiza en función de las horas hombre (HH) requeridas para cada actividad, asignando posteriormente el costo monetario correspondiente a cada recurso, según se detalla en la Tabla 3.2.

Tabla 3.2: Costos de mano de Obra

Hora Hom bre	Pesos Chilenos	UF	UF Actual		
Tecnico	\$ 7.000	0,17			
Ingeniero	\$ 14.000	0,35	\$ 40.027		

Actividad	Responsable	Horas hombre	Costos en peso chileno neto	Precio con IVA	Costo en UF
Visita a terreno					
Cotizacion, contacto y adquisicion	Ingeniero	40	\$ 560.000	\$ 666.400	16,6
Montaje de equipos e instalacion de sistema operativo	Ingeniero	72	\$ 1.008.000	\$ 1.199.520	30,0
	Tecnico	48	\$ 336.000	\$ 399.840	10,0
Prueba de comunicaci3n 1	Ingeniero	40	\$ 560.000	\$ 666.400	16,6
Programacion e instalacion de software	Ingeniero	80	\$ 1.120.000	\$ 1.332.800	33,3
Diseño de dashboard	Ingeniero	40	\$ 560.000	\$ 666.400	16,6
Prueba de funcionamiento final	Ingeniero	40	\$ 560.000	\$ 666.400	16,6
Entrega del proyecto	Ingeniero	20	\$ 280.000	\$ 333.200	8,3
Total			\$ 5.930.960		148,2

Fuente: Elaboraci3n propia en Excel.

3.3 COSTOS DE MATERIALES A UTILIZAR

A continuación, en la tabla 3.3 se presentan los principales materiales y equipos requeridos para la implementación del proyecto, indicando sus respectivas cantidades, costos unitarios y costos totales, con el propósito de determinar la inversión asociada a los componentes del sistema.

Tabla 3.3: Costos de Materiales a utilizar.

						Valor UF	40.027,00
Nombre	Cantidad	Unidad Medida	Precio por unidad	Precio total neto	Precio total mas IVA	Costo en UF	
Raspberry	1	unidad	\$ 172.000	\$ 172.000	\$ 204.680	4,30	
Sensor de radar	12	unidad	\$ 624.500	\$ 7.494.000	\$ 8.917.860	187,22	
Cable alimentación XLPE SHF1 2x2,5 mm ²	120	metros	\$ 456.000	\$ 456.000	\$ 542.640	11,39	
Cable comunicación RS485 XLPE SHF1	120	metros	\$ 384.000	\$ 384.000	\$ 456.960	9,59	
Fuente alimentacion Mean Well 24V 10A	1	unidad	\$ 140.000	\$ 140.000	\$ 166.600	3,50	
Monitor LED 22" Full HD HDMI	1	unidad	\$ 89.000	\$ 89.000	\$ 105.910	2,22	
Módulo LoRa USB para Raspberry Pi	1	unidad	\$ 58.000	\$ 58.000	\$ 69.020	1,45	
Módem USB 4G LTE liberado	1	unidad	\$ 69.000	\$ 69.000	\$ 82.110	1,72	
Convertidor USB a RS485 industrial	1	unidad	\$ 24.000	\$ 24.000	\$ 28.560	0,60	
				total	\$ 10.574.340	222,00	

Fuente: Elaboración propia en Excel.

3.4 RESUMEN DE COSTOS DEL PROYECTO

En el resumen de costos del proyecto se incorpora el costo asociado al diseño, el cual se calcula en función del tiempo destinado al desarrollo de la investigación y a la ejecución del proyecto. Además, se consideran los costos correspondientes a materiales, mano de obra e imprevistos.

Tabla 2.4: Resumen del costo del proyecto

Detalles del costo	Total pesos chilenos	Total en UF
Materiales	\$ 10.574.340	\$ 264
Mano de obra y Diseño	\$ 5.930.960	\$ 148
Reumen total	\$ 16.505.300	\$ 412
Imprevisto 20% del total	\$ 3.301.060	\$ 82
Total general	\$ 19.806.360	\$ 495

Fuente: Elaboración propia en Excel.

La tabla 3.4 presenta el resumen de costos asociados a la implementación del proyecto, expresados tanto en pesos chilenos (CLP) como en Unidades de Fomento (UF), considerando un valor de referencia de 40,200.12 CLP por UF.

Esta doble representación permite facilitar la comprensión del costo total y mantener una referencia estable frente a posibles variaciones inflacionarias.

En primer lugar, se observa que el costo correspondiente a materiales asciende a 10.574.340 CLP, equivalente a 262,35 UF. Este ítem incluye los insumos, equipos o componentes necesarios para la ejecución del proyecto. Por otra parte, el costo asociado a mano de obra corresponde a 5.930.960 CLP, equivalente a 147,14 UF. representando la mayor proporción del presupuesto total. Esto indica que el desarrollo del proyecto depende principalmente del trabajo técnico y de instalación requerido para su implementación.

La suma de ambos conceptos (materiales y mano de obra) alcanza un subtotal de 16.505.300, equivalente a 409,49 UF. Sobre este monto se incorpora un 20% adicional por concepto de imprevistos, equivalente a 3.301.060 CLP equivalente a 81,9 UF.

con el objetivo de cubrir posibles variaciones en los costos, ajustes durante la ejecución o gastos no considerados inicialmente.

Finalmente, el costo total del proyecto alcanza un valor de 19.806.360 CLP, equivalente a 491,40 UF. Este monto representa la inversión total necesaria para llevar a cabo la implementación propuesta.

3.5 ESTIMACIÓN DEL VALRT REFERENCIAL DEL ALIMENTO Y BENEFICIO ECONÓMICO ANUAL

Para desarrollar la evaluación económica del proyecto, fue necesario establecer un valor referencial del alimento para salmón, debido a que este insumo representa uno de los principales costos operacionales dentro del proceso productivo acuícola. La determinación de este valor resulta fundamental, ya que permite cuantificar económicamente las pérdidas asociadas a errores en la estimación del alimento almacenado, así como también estimar los beneficios potenciales derivados de una mayor precisión en el monitoreo del inventario.

Dado que no se contó con acceso directo a valores comerciales específicos asociados a una empresa en particular, se opta por realizar una estimación indirecta basada en antecedentes económicos reportados por la industria salmonera. Para ello, se considera que el alimento representa aproximadamente el 50 % del costo total de producción del salmón, junto con un índice de conversión alimenticia referencial de 1,2, valor ampliamente utilizado como indicador de eficiencia alimentaria en sistemas de cultivo.

A partir de estos antecedentes, se determinó un valor referencial de 1.640 CLP/kg, cifra que fue utilizada como base para el desarrollo de la evaluación económica y el análisis de sensibilidad del proyecto.

Para estimar el beneficio económico asociado a la implementación del sistema propuesto, se adoptó una metodología conservadora con el fin de evitar la sobreestimación de los ahorros potenciales derivados del proyecto. Si bien el sistema desarrollado contempla el monitoreo del conjunto de silos del pontón, para efectos de la evaluación económica se toma como referencia únicamente un silo representativo con una capacidad operacional equivalente a 30 toneladas de alimento, considerando un período de consumo de una semana. Esto permite establecer una estimación prudente del impacto económico asociado a errores en la medición del inventario bajo el método actual de control manual.

Considerando el valor referencial previamente estimado, el valor económico asociado a dicha capacidad corresponde a:

$$30.000 \text{ kg} \times 1.640 \text{ CLP/kg} = 49.200.000 \text{ CLP}$$

Debido a que el método actual se basa en una estimación visual del nivel de alimento mediante las marcas de referencia del silo, pueden existir diferencias entre la cantidad estimada y la cantidad real almacenada, especialmente cuando el nivel se encuentra por encima o por debajo de una de estas marcas. Para efectos de la evaluación económica, se considera de manera conservadora una desviación operacional referencial del 0,5 %, atribuible a errores de estimación en el control del alimento almacenado, obteniéndose un valor económico semanal asociado a esta desviación de:

$$49.200.000 \times 0,005 = 246.000 \text{ CLP}$$

Finalmente, considerando una operación continua equivalente a 52 semanas (1 año), el ahorro económico potencial asociado a la reducción de esta desviación corresponde a:

$$246.000 \times 52 = 12.792.000 \text{ CLP/año}$$

Este valor fue incorporado como ingreso anual dentro del flujo de caja del proyecto, representando una estimación conservadora del beneficio económico derivado de una mayor precisión en el monitoreo del alimento almacenado.

3.6 FLUJO DE CAJA

A continuación, en la tabla 3.5 se presenta el flujo de caja del proyecto, en el cual se detallan los ingresos, costos y beneficios considerados durante el período de evaluación, permitiendo analizar su comportamiento económico a lo largo del tiempo.

Tabla 3.5: Flujo de caja

Descripción	Año 0	Año 1	Año 2	Año 3	Año 4	Año 5
Ingresos						
Ahorro de comida de pellet por aumento de precisión		\$ 12.792.000	\$ 12.792.000	\$ 12.792.000	\$ 12.792.000	\$ 12.792.000
Costo del proyecto	\$ -19.806.360					
Repuestos		\$ 999.600	\$ 999.600	\$ 999.600	\$ 999.600	\$ 999.600
Mantenimiento		\$ 500.000	\$ 500.000	\$ 500.000	\$ 500.000	\$ 500.000
Pago de plan internet		\$ 330.000	\$ 330.000	\$ 330.000	\$ 330.000	\$ 330.000
Consumo de energía		\$ 225.000	\$ 225.000	\$ 225.000	\$ 225.000	\$ 225.000
Total egresos		\$ 2.054.600	\$ 2.054.600	\$ 2.054.600	\$ 2.054.600	\$ 2.054.600
Flujo de caja neto	\$ -19.806.360	\$ 10.737.400	\$ 10.737.400	\$ 10.737.400	\$ 10.737.400	\$ 10.737.400
Valor presente del flujo	\$ -19.806.360	\$ 9.586.964	\$ 8.559.790	\$ 7.642.669	\$ 6.823.812	\$ 6.092.689
Flujo de caja acumulado	\$ -19.806.360	\$ -10.219.396	\$ -1.659.606	\$ 5.983.063	\$ 12.806.875	\$ 18.899.564

TMAR	12%
VAN	\$ 18.899.564
TIR	46%
PAYBACK	3 años

Fuente: Elaboración propia en Excel.

A partir de los indicadores financieros obtenidos del flujo de caja, es posible concluir que el proyecto presenta viabilidad económica y financiera. La tasa mínima aceptable de retorno (TMAR) se estableció en un 12%, mientras que la tasa interna de retorno (TIR) alcanza un 46%, superando ampliamente el rendimiento mínimo exigido y evidenciando una rentabilidad atractiva para la inversión.

Asimismo, el valor actual neto (VAN) presenta un resultado positivo de \$18.899.564, lo que indica que, al considerar la inversión inicial y descontar los flujos de caja futuros, el proyecto genera valor económico adicional. Este resultado demuestra que la inversión no solo recupera el capital invertido, sino que también produce beneficios económicos.

En cuanto al período de recuperación de la inversión (Payback), este se alcanza aproximadamente en 3 años, permitiendo recuperar el capital inicial dentro del horizonte de

evaluación establecido. Cabe señalar que esta evaluación económica fue desarrollada bajo un enfoque conservador, considerando como referencia operacional únicamente un silo representativo, pese a que el sistema contempla el monitoreo del conjunto completo de silos del pontón. En conjunto, estos indicadores permiten concluir que el proyecto constituye una alternativa técnica y económicamente viable para optimizar la gestión del alimento almacenado y reducir pérdidas derivadas de errores en la estimación manual.

3.7 ANÁLISIS DE SENSIBILIDAD

El análisis de sensibilidad se realiza con el objetivo de evaluar la robustez financiera del proyecto frente a posibles variaciones en las condiciones consideradas en la evaluación económica. Para ello, se definieron distintos escenarios que contemplan una disminución progresiva del porcentaje de pérdidas evitadas, desde un 0,50 % hasta un 0,35 %, junto con incrementos en los principales costos operacionales, correspondientes a mantenimiento, repuestos, energía eléctrica e internet. De esta manera, se analiza el comportamiento de los principales indicadores financieros del proyecto ante escenarios progresivamente más desfavorables, como se muestra en la tabla 3.6.

Tabla 3.6: Análisis de sensibilidad

	Caso 0	Caso 1	Caso 2	Caso 3
Disminución de pérdidas	0,50%	0,45%	0,40%	0,35%
Mantenimiento	–	7%	10%	14%
Repuestos	–	10%	14%	16%
Energía	–	15%	25%	35%
Internet	–	2%	4%	6%
TMAR	12%	12%	12%	12%
VAN	\$18.899.564	\$13.656.381	\$8.742.047	\$3.881.756
TIR	46%	37%	29%	20%
PAYBACK	3 años	3 años	4 años	4 años

Fuente: Elaboración propia en Excel.

CONCLUSIONES

El análisis del proceso actual de almacenamiento, verificación y dosificación de alimento en los pontones permitió identificar limitaciones que afectan directamente la eficiencia productiva y la precisión en la gestión del pellet. Entre los principales hallazgos destacan la ausencia de medición continua, la alta dependencia del control visual y la posibilidad de errores significativos en la estimación del inventario disponible. Estas deficiencias reflejan la necesidad de modernizar el proceso mediante una herramienta tecnológica capaz de proporcionar información confiable en tiempo real, con mínima intervención humana.

El levantamiento de información en terreno, complementado con el estudio de la infraestructura del pontón y del funcionamiento interno de los silos y dosificadores, permitió comprender las condiciones reales de operación y dimensionar los desafíos técnicos del entorno productivo. A partir de esta caracterización, se definieron las bases para una solución orientada a cumplir criterios de robustez, precisión y adaptabilidad, acordes con las exigencias de la industria salmonicultura.

Como respuesta a la problemática detectada, se propone un sistema automatizado de monitoreo de nivel basado en sensores industriales, adquisición de datos y comunicación remota. Esta solución busca mejorar la exactitud en la medición del alimento almacenado, optimizar la toma de decisiones y disminuir pérdidas derivadas de errores en la gestión manual del inventario.

Desde el punto de vista económico, los resultados obtenidos indican que la propuesta presenta condiciones favorables de rentabilidad, generación de valor y recuperación de la inversión, incluso bajo un enfoque conservador de evaluación.

En conclusión, la implementación de esta solución representa una alternativa concreta para avanzar hacia operaciones más eficientes, trazables y confiables en pontones salmoneros. Además de los beneficios técnicos y económicos identificados, su diseño ofrece potencial de replicabilidad en instalaciones con características similares.

BIOGRAFÍA

OpenJS Foundation. (2024). Node-RED: Flow-based programming tool.

<https://nodered.org>

Python Software Foundation. (2024). Python 3 documentation.

<https://www.python.org>

Raspberry Pi Foundation. (2024). Raspberry Pi 4 datasheet.

<https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf>

Raspberry Pi Foundation. (2024). Raspberry Pi OS.

<https://www.raspberrypi.com/software/>

Raspberry Pi Foundation. (s. f.). Raspberry Pi documentation.

<https://www.raspberrypi.com/documentation/>

Semtech Corporation. (s. f.). SX1278 LoRa RF transceiver.

<https://www.semtech.com/products/wireless-rf/lora-connect/sx1278>

Supmea Automation Co., Ltd. (s. f.). WSR550 radar level transmitter datasheet.

<https://www.supmeaauto.com/uploads/2402/sup-wsr550-radar-level-transmitter-datasheet.pdf>

Máster-G. (s. f.). Monitor de PC 22" Full HD 100 Hz MGME2230.

<https://www.masterg.cl/monitor-de-pc-22--full-hd-100hz-1ms-mgme2230/>