Universidad de Cuenca

Facultad de Ingeniería

Carrera de Electrónica y Telecomunicaciones

Implementación de una aplicación de Internet Industrial de las Cosas para la integración de tecnologías de comunicación heterogéneas de un sistema multi-tanque

Trabajo de titulación previo a la obtención del título de Ingeniero en Electrónica y Telecomunicaciones.

Autores :

Christian Marcelo Garzón Vazquez
marcelo.garzon.vazquez@gmail.com C.I. 030202524-2

Erika Paola Serrano Ramírez
erikaserranor30@hotmail.com C.I. 030224064-3

Director :

Ing. Luis Ismael Minchala Ávila, PhD C.I. 030145348-6

Cuenca - Ecuador

25 de octubre de 2021

Resumen

En el entorno actual del sector de producción y manufactura, las industrias se enfrentan al reto que supone el avance tecnológico, la competencia entre los fabricantes existentes en el mercado y la globalización del internet de las cosas (Internet of Things (IoT)). Frente a esto, la industria evoluciona y se reinventa, concibiéndose así la Industria 4.0 y el internet industrial de las cosas (Industrial Internet of Things (IIoT)). Es clave para el IIoT constantemente trabajar en la integración de tecnologías heterogéneas, pues la industria difícilmente implementa soluciones utilizando equipos de un mismo fabricante o proveedor. Esto le permitirá a la industria optimizar sus procesos de fabricación, alcanzar mayor flexibilidad y generar propuestas de valor para los usuarios o clientes.

En este contexto, este trabajo de titulación propone la implementación de una aplicación IIoT que permita la integración de tecnologías de comunicación heterogéneas, utilizando equipamiento de laboratorio de la Universidad de Cuenca para la automatización de un sistema multi-tanque. Esta aplicación integra desde equipos de alta gama como el controlador lógico programable (Programmable Logic Controller (PLC)) Simatic Siemens S7-1500 hasta equipos de bajo costo como el PLC Controllino Maxi Automatization, mediante la implementación de un servidor de plataforma abierta para comunicaciones industriales de arquitectura unificada (Open Platform Communications - Unified Architecture (OPC-UA)) y mayormente *software* libre. El servidor OPC-UA permite la integración de la aplicación IIoT con los sistemas de control y supervisión de la planta. La arquitectura propuesta toma como base la manufactura integrada por computador (Computer-Integrated Manufacturing (CIM)), desde una virtualización de las capas de sistemas de ejecución de manufactura (Manufacturing Execution Systems (MES)) y planificación de recursos empresariales (Enterprise Resource Planning (ERP)) a través de una solución IIoT. La solución IIoT permite la interacción total con la planta, desde sistemas de monitoreo implementados en *dashboard*, almacenamiento de información en una base de datos local y en la nube, sistemas de consulta por *chat* y servicios de detección de fallas utilizando *machine learning*; todos ejecutados de forma paralela y aislada. El propósito de este proyecto es explotar las capacidades de los equipos utilizados y potenciar los resultados, al implementar soluciones novedosas en el campo de las telecomunicaciones.

Palabras clave : Industria 4.0. IIoT. Control. Automatización. OPC-UA. CIM. Virtualización.

Christian Marcelo Garzón Vazquez
marcelo.garzon.vazquez@gmail.com
Erika Paola Serrano Ramírez
erikaserranor30@hotmail.com

Abstract

In the current environment of the production and manufacturing sector, industries face different challenges, such as technological improvement, the competition amongst manufacturers in the market and the Internet of Things (IoT) going worldwide. In order to deal with this, the industry evolves and reinvents itself, thus conceiving the 4.0 Industry and the Industrial Internet of Things (IIoT). It is key for IIoT to constantly work on the integration of heterogeneous technologies, since the industry hardly implements solutions using equipment from the same manufacturer or supplier. This will allow the industry to optimize its manufacturing processes, achieve greater flexibility and generate stronger value proposals for users or customers.

In this context, this work proposes the implementation of an IIoT application that allows the integration of heterogeneous communication technologies, using laboratory equipment from the University of Cuenca for the automation of a multi-tank system. This application integrates a variety of hardware, from high-end equipment such as the Simatic Siemens S7-1500 programmable logic controller (PLC) to low-cost equipment such as the Controllino Maxi Automatization PLC, through the development of an Open Platform Communications with Unified Architecture industrial server (OPC-UA) and mostly free software. The OPC-UA server allows the integration of the IIoT application with the control and supervision systems of the plant. The proposed architecture is based on Computer Integrated Manufacturing (CIM), from a virtualization of the Manufacturing Execution Systems (MES) and Enterprise Resource Planning (ERP) layers by means of an IIoT solution. The IIoT solution enables total interaction with the plant, from monitoring systems implemented in dashboards, data storage in a local database and in the cloud, chat consultation systems and fault detection services using machine learning; all executed in parallel and in isolation. The purpose of this project is to exploit the capabilities of the equipment used and enhance the results by implementing innovative solutions in the field of telecommunications.

Keywords : 4.0 Industry. IIoT. Control. Automatization. OPC-UA. CIM. Virtualization.

Índice general

Resumen I

Abstract III

Índice general V

Índice de figuras IX

Índice de tablas XI

Cláusula de Propiedad Intelectual XIII

Cláusula de Propiedad Intelectual XV

Cláusula de licencia y autorización para publicación en el Repositorio Institucional XVII

Cláusula de licencia y autorización para publicación en el Repositorio Institucional XIX

Dedicatoria XXI

Dedicatoria XXIII

Agradecimientos XXV

Agradecimientos XXVII

Abreviaciones y acrónimos XXIX

1. Introducción 1

1.1. Identificación del problema . . . 1
1.2. Justificación . . . 2
1.3. Alcance . . . 2
1.4. Objetivos . . . 3
1.4.1. Objetivo general . . . 3
1.4.2. Objetivos específicos . . . 3
1.5. Contribuciones del trabajo de titulación . . . 4

2. Fundamentos teóricos **5**
2.1. Industria 4.0 . . . 5
2.1.1. Sistemas ciberfísicos (Cyber-Physical System (CPS)) . . . 5
2.1.2. Internet industrial . . . 6
2.2. Internet industrial de las cosas (IIoT) . . . 6
2.2.1. Arquitectura de referencia de IIoT . . . 6
2.2.2. IIoT y la Industria 4.0 . . . 8
2.2.3. IIoT y la automatización industrial . . . 8
2.3. Redes industriales y IIoT . . . 9
2.3.1. Retos de las redes industriales . . . 9
2.3.2. Protocolos de las redes industriales . . . 10
2.3.3. Integración de la red de comunicaciones industriales . . . 10
2.4. Manufacturada integrada por computador (CIM) . . . 10
2.4.1. Arquitectura CIM . . . 10
2.4.2. Bus de campo . . . 11
2.4.2.1. Modbus TCP/IP . . . 12
2.4.2.2. Profibus . . . 12
2.4.2.3. Profinet . . . 12
2.5. Plataforma de comunicaciones abiertas (Open Platform Communications (OPC)) . . . 13
2.5.1. Arquitectura OPC unificada (OPC-UA) . . . 13
2.6. Algoritmos de control . . . 14
2.6.1. Control proporcional-integral-derivativo (Proportional–Integral–Derivative (PID)) . . . 14
2.6.2. Control On-Off . . . 16
2.7. Docker . . . 16
2.7.1. Cliente y servidor Docker . . . 17
2.7.2. Imágenes Docker . . . 17
2.7.3. Registros Docker . . . 18
2.7.4. Contenedores Docker . . . 18
2.7.5. Comandos clave de Docker . . . 18
2.8. Métodos de detección de fallas . . . 18
2.8.1. Análisis de componente principal (Principal Component Analysis (PCA)) . . . 19
2.8.2. Redes neuronales . . . 20
2.9. Servicios de almacenamiento en la nube . . . 21
2.10. *Chatbot* . . . 21

3. Estado del arte **23**
3.1. Estado del arte de sistemas de control de nivel multi-tanque . . . 23
3.2. Estado de arte de protocolos, arquitectura y virtualización para la Industria 4.0 . . . 24
3.3. Estado del arte de integración de sistemas heterogéneos . . . 26

4. Metodología **27**
4.1. Descripción del sistema . . . 27
4.2. Diagrama de instrumentación y procesos del sistema . . . 30
4.3. Descripción de la arquitectura . . . 30

4.3.1. Primera capa o nivel de campo . . . 31
4.3.2. Segunda capa o nivel de control . . . 31
4.3.3. Tercera capa o nivel de supervisión . . . 33
4.3.4. Cuarta capa o MES . . . 33
4.3.4.1. *Dashboard* . . . 34
4.3.4.2. Base de datos local . . . 35
4.3.4.3. Envío de datos a la nube . . . 35
4.3.4.4. Sistema de detección de fallas con *Machine Learning* . . . 38
4.3.4.5. *Chatbot* . . . 38
4.4. Diseño del *hardware* . . . 40

5. Resultados **43**
5.1. Funcionamiento del sistema multi-tanque . . . 43
5.1.1. Porcentaje de sobreimpulso y tiempo de estabilización del sistema . . . 43
5.1.2. Cambios de referencia en el sistema . . . 46
5.2. Análisis de los sistemas de interacción con la planta . . . 48
5.2.1. Sistema de supervisión y control en planta . . . 48
5.2.2. Sistema de monitoreo y control administrativo . . . 48
5.2.3. Sistema de monitoreo externo a la planta . . . 50
5.2.4. Sistema de consulta externo por *chat* . . . 50
5.2.5. Análisis de longitudes de paquetes enviados por la planta . . . 50
5.3. Análisis del envío de información a la base de datos local y a la nube de almacenamiento de información . . . 54
5.4. Análisis del servicio de detección de fallas . . . 56
5.5. Comparación con la solución IIoT de Siemens . . . 59

6. Conclusiones y trabajos futuros **63**
6.1. Conclusiones . . . 63
6.2. Trabajos futuros . . . 65

A. Diseño de los tanques y armazón de la planta del sistema multi-tanque **67**

B. Diseño del tablero de instrumentación **73**

Bibliografía **77**

Índice de figuras

2.1. Arquitectura conceptual de IIoT . . . 8
2.2. Arquitectura CIM . . . 11
2.3. Entradas y salidas del controlador. . . . 15
2.4. Diagrama de bloques: control proporcional. . . . 15
2.5. Diagrama de bloques: control integral. . . . 15
2.6. Diagrama de bloques: control derivativo. . . . 16
2.7. Arquitectura Docker. . . . 17

3.1. Arquitectura propuesta en [50] . . . 25

4.1. Esquema general del sistema realizado en el trabajo. . . . 28
4.2. Diagrama de proceso e instrumentación del sistema. . . . 30
4.3. Arquitectura planteada del sistema. . . . 31
4.4. Diagrama de red del sistema. . . . 32
4.5. Sistema de control planteado. . . . 33
4.6. Diagrama de flujo *dockerfile*. . . . 36
4.7. Diagrama de flujo código Python. . . . 37
4.8. Diagrama de flujo del entrenamiento del clasificador. . . . 39
4.9. Diagrama de flujo de la evaluación del comportamiento de la planta. . . . 39
4.10. Diagrama de flujo del Chatbot. . . . 40
4.11. Diseño del sistema de tanques. . . . 41
4.12. Diseño del tablero de instrumentación. . . . 41

5.1. Sistema multi-tanque ensamblado. . . . 44
5.2. Tablero de instrumentación ensamblado. . . . 44
5.3. Tablero de instrumentación ensamblado, parte interna. . . . 45
5.4. Comportamiento transitorio del sistema multi-tanque. . . . 45
5.5. Comportamiento transitorio del sistema multi-tanque utilizando flotadores. . . . 46
5.6. Comportamiento promedio del sistema en las pruebas de sobreimpulso y tiempo de estabilización. 47
5.7. Comportamiento promedio del sistema en las pruebas de cambio de referencia. . . . 47
5.8. Diseño del Human-Machine Interface (HMI). . . . 49
5.9. Diseño del *dashboard* en Node-RED, parte superior. . . . 49
5.10. Diseño del *dashboard* en Node-RED, parte inferior. . . . 49
5.11. Diseño del *dashboard* en la nube de almacenamiento de información de Thinger.io. . . . 50
5.12. Consulta de información del sistema al *chatbot*. . . . 51

5.13. Captura de envío de paquete del servidor a Node-RED en Wireshark. 51
5.14. Captura de envío de paquete de Node-RED a la nube Thinger.io en Wireshark. 52
5.15. Verificación de método POST en protocolo HTTP. 52
5.16. Captura de envío de paquete del servidor a Python en Wireshark. 52
5.17. Captura de envío de paquete de Python a MySQL en Wireshark. 53
5.18. Captura de envío de paquete del servidor al *chatbot* en Wireshark. 53
5.19. Secuencia de paquetes capturados entre servidor y *chatbot* en Wireshark. 53
5.20. Paquetes de datos recibidos en el *databucket* de Thinger.io. 55
5.21. Comparación de gráficas de respuesta del sistema con los datos de MySQL y Thinger.io. . . . 56
5.22. Geometría del sistema. 58
5.23. Geometría del sistema luego del clasificador. 59
5.24. Matriz de confusión del clasificador. 59
5.25. Diagrama de araña de la solución IIoT de Siemens vs. la solución IIoT desarrollada 61

Índice de tablas

4.1. Equipos que forman parte del sistema. 29
4.2. Variables implementadas en el servidor OPC-UA. 34

5.1. Resultados de las pruebas de sobreimpulso y tiempo de estabilización. 46
5.2. Resultados de las pruebas de cambio de referencia 47
5.3. Longitudes mínima y máxima de paquetes analizados en el servidor de contenedores. 54
5.4. Resultados de paquetes enviados, recibidos y pérdidas para MySQL y Thinger.io durante 1 hora de almacenamiento de información. 55
5.5. Parámetros utilizados en el clasificador SVM. 58
5.6. Comparación entre la solución IIoT Siemens y la solución desarrollada. 60

Cláusula de Propiedad Intelectual

Christian Marcelo Garzón Vazquez, autor/a del trabajo de titulación "Implementación de una aplicación de Internet Industrial de las Cosas para la integración de tecnologías de comunicación heterogéneas de un sistema multi-tanque", certifico que todas las ideas, opiniones y contenidos expuestos en la presente investigación son de exclusiva responsabilidad de su autor/a.

Cuenca, 25 de octubre de 2021.

Christian Marcelo Garzón Vazquez

C.I: 0302025242

Cláusula de Propiedad Intelectual

Erika Paola Serrano Ramírez, autor/a del trabajo de titulación "Implementación de una aplicación de Internet Industrial de las Cosas para la integración de tecnologías de comunicación heterogéneas de un sistema multi-tanque", certifico que todas las ideas, opiniones y contenidos expuestos en la presente investigación son de exclusiva responsabilidad de su autor/a.

Cuenca, 25 de octubre de 2021.

Erika Serrano R.

Erika Paola Serrano Ramírez

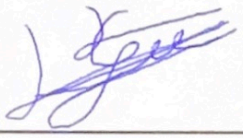
C.I: 0302240643

Cláusula de licencia y autorización para publicación en el Repositorio Institucional

Christian Marcelo Garzón Vazquez en calidad de autor/a y titular de los derechos morales y patrimoniales del trabajo de titulación "Implementación de una aplicación de Internet Industrial de las Cosas para la integración de tecnologías de comunicación heterogéneas de un sistema multi-tanque", de conformidad con el Art. 114 del CÓDIGO ORGÁNICO DE LA ECONOMÍA SOCIAL DE LOS CONOCIMIENTOS, CREATIVIDAD E INNOVACIÓN reconozco a favor de la Universidad de Cuenca una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos.

Asimismo, autorizo a la Universidad de Cuenca para que realice la publicación de este trabajo de titulación en el repositorio institucional, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Cuenca, 25 de octubre de 2021.



Christian Marcelo Garzón Vazquez

C.I: 0302025242

Cláusula de licencia y autorización para publicación en el Repositorio Institucional

Erika Paola Serrano Ramírez en calidad de autor/a y titular de los derechos morales y patrimoniales del trabajo de titulación "Implementación de una aplicación de Internet Industrial de las Cosas para la integración de tecnologías de comunicación heterogéneas de un sistema multi-tanque", de conformidad con el Art. 114 del CÓDIGO ORGÁNICO DE LA ECONOMÍA SOCIAL DE LOS CONOCIMIENTOS, CREATIVIDAD E INNOVACIÓN reconozco a favor de la Universidad de Cuenca una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos.

Asimismo, autorizo a la Universidad de Cuenca para que realice la publicación de este trabajo de titulación en el repositorio institucional, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Cuenca, 25 de octubre de 2021.

Erika Serrano R.

Erika Paola Serrano Ramírez

C.I: 0302240643

Dedicatoria

A mis padres
con mucho amor y cariño
les dedico todo mi esfuerzo
y trabajo puesto para
la realización de esta tesis.

Christian Marcelo Garzón Vazquez

Dedicatoria

A mis padres, Oscar y Diana, por el apoyo, dedicación, confianza y amor infinito depositados en su hija. Gracias por velar siempre por mí.

A mis hermanos, Xavier y Paúl, por brindarme su apoyo en los momentos que más les he necesitado. Gracias por nunca decirme que no.

A mi familia en general, este trabajo se desarrolló durante momentos difíciles en nuestras vidas, sin sus palabras de aliento, la fe en Dios y nuestro angelito, no hubiera encontrado la fuerza para seguir firme en este camino.

Segura estoy de que celebrarán este triunfo conmigo. Los amo a todos.

Erika Paola Serrano Ramírez

Agradecimientos

A mi familia, por el apoyo que me han brindado en estos años de estudio, a mis compañeros, por hacer que este periodo universitario sea lleno de anécdotas y buenos momentos, a mis profesores, por el conocimiento que me han transmitido en las aulas de clase y de manera especial a mi director de tesis Ing. Ismael Minchala, por el tiempo dedicado a este proyecto.

Christian Marcelo Garzón Vazquez

Agradecimientos

A Dios y mi familia, pilares fundamentales durante el proceso de mi formación académica universitaria.

A todos mis profesores, que a lo largo de la carrera universitaria me han brindado sus conocimientos y me han motivado a siempre seguir adelante y dar lo mejor de mí, permitiéndome crecer en el ámbito académico y personal. En especial, al Ing. Ismael Minchala, director de este trabajo, por el apoyo y tiempo dedicado al desarrollo de este proyecto.

A todos mis compañeros de clase, segura estoy de que el apoyo mutuo nos ha llevado a cumplir nuestras metas como estudiantes. En especial a Marcelo G., por su dedicación y esfuerzo para sacar este proyecto adelante.

Erika Paola Serrano Ramírez

Abreviaciones y Acrónimos

AMQP Advanced Message Queuing Protocol. 14
API Application Programming Interface. 7, 34, 38

CBA Component Based Automation. 13
CIM Computer-Integrated Manufacturing. 2, 3, 5, 10, 11, 24–26, 30, 63
CPPS Cyber-Physical Production Systems. 5, 26
CPS Cyber-Physical System. 5, 9, 26

DCS Distributed Control System. 6

EFTA Emerging Technologies and Factory Automation. 14
ERP Enterprise Resource Planning. 3, 10, 11, 13, 21, 25

HMI Human-Machine Interface. 4, 7, 27, 29, 33, 42, 48, 49, 56, 64
HTTP Hypertext Transfer Protocol. 35, 51, 54

ICT Information and Communications Technology. 5
IED Intelligent Electronic Device. 7
IIoT Industrial Internet of Things. 3, 5–9, 22, 23, 26–28, 30, 33, 43, 50, 59–61, 63, 64
IoT Internet of Things. 5, 6, 8, 21, 24, 25, 35, 63
ISE Integral Square Error. 23, 24
IT Information Technology. 1

LAN Local Area Network. 13
LED Light Emitting Diode. 4, 34, 64
LPWAN Low Power Wide Area Network. 10

M2M Machine-to-Machine Commnication. 2, 8, 10, 25
MES Manufacturing Execution Systems. 2, 3, 10, 11, 25, 26, 33, 65
MIMO Multiple-Input Multiple-Output. 24
MQTT Message Queue Telemetry Transport. 14

NTP Network Time Protocol. 35, 54, 55

OPC Open Platform Communications. 3, 5, 13
OPC-UA Open Platform Communications - Unified Architecture. 13, 14, 23, 24, 26, 27, 33–35, 38, 51–54, 63, 64

PCA Principal Component Analysis. 19, 38, 56, 65
PID Proportional–Integral–Derivative. 14, 15, 23, 24, 26, 27
PLC Programmable Logic Controller. 4, 6, 14, 25, 27, 29, 31–33, 42, 43, 64

RTU Remote Terminal Unit. 7

SCADA Supervisory Control And Data Acquisition. 7
SVM Support Vector Machine. 20, 38, 58

TCP Transmission Control Protocol. 52–54
TLBO Teacher-Learner Based Optimization. 24
TSN Time-Sensitive Network. 13

UDP User Datagram Protocol. 13
URL Uniform Resource Locator. 35

WAN Wide Area Network. 13, 14
WLAN wireless Local Area Network. 10
WPAN Wireless Personal Area Network. 10

CAPÍTULO 1

Introducción

Este capítulo presenta la identificación del problema, justificación y los objetivos del presente proyecto.

1.1. Identificación del problema

En el entorno actual de competencia global, desarrollo tecnológico e innovación, las empresas, sobretodo de manufactura, se ven forzadas a reconfigurar sus procesos. La Industria 4.0 y la manufactura inteligente son parte de una transformación, en la que las tecnologías de fabricación y de la información se han integrado para crear innovadores sistemas de manufactura, gestión y formas de hacer negocios. Esto permite optimizar los procesos de fabricación, alcanzar una mayor flexibilidad, eficiencia y generar una propuesta de valor para sus clientes, así como responder de forma oportuna a las necesidades de su mercado [1].

La conceptualización que existe sobre Industria 4.0 es reciente, sin embargo, ha sido definida como una maquinaria física y dispositivos con sensores y *software* que trabajan en red y permiten predecir, controlar y planificar mejor los negocios y los resultados organizacionales [2]. La Industria 4.0 representa un enfoque a la innovación de nuevos productos y procesos, a través de fábricas inteligentes, totalmente integradas en redes de trabajo que propician nuevas formas de colaboración e infraestructuras sociales [3].

Los sistemas de tecnología de la información (Information Technology (IT)) ahora pueden admitir instrumentación, monitoreo y análisis generalizados debido a una caída en los costos de cómputo, ancho de banda, almacenamiento y sensores. Esto significa que es posible monitorear máquinas industriales a mayor escala. Estas tecnologías están madurando y cada vez están más disponibles, y este parece ser un punto clave. Sin embargo, como se menciona en [4], solo recientemente los líderes de negocios industriales han sido testigos de la estabilidad y madurez de las soluciones, herramientas y aplicaciones dentro de estos sectores de IT que alcanzan un nivel de confianza y disminuyen las preocupaciones.

1.2. Justificación

De acuerdo a [5], las tecnologías básicas en que se sustenta la Industria 4.0 son: comunicaciones móviles, la nube (*cloud computing*), análisis de datos, comunicación máquina a máquina (Machine-to-Machine Commnication (M2M)), plataformas sociales, impresión 3D (fabricación aditiva), robótica avanzada y colaborativa, realidad aumentada, y, seguridad.

La integración de estas tecnologías requiere de un adecuado control de proceso. Según [6], el control de procesos consiste en dos funciones claramente diferenciadas: la adquisición de datos y el control. Si se trata de establecer el nexo con el mantenimiento se concluirá que la adquisición de datos contribuye con la información para el mantenimiento y las acciones de control con la implantación de las acciones con fines tanto operativos como de mantenimiento. El control automático de procesos es parte del progreso industrial, se usa fundamentalmente porque reduce el costo de los procesos industriales, lo que compensa con creces la inversión en equipo de control.

Con base en lo expuesto, la integración de tecnologías heterogéneas y la automatización de procesos para un sistema multi-tanque se implementará de acuerdo a una arquitectura de manufactura integrada por computador (CIM). Ésta corresponde a una filosofía de implementación de un sistema informático que integra todos los procesos existentes en un proceso de fabricación, tanto en lo que se refiere a las áreas comerciales, como a las de diseño, fabricación, distribución, *etc*.

1.3. Alcance

El trabajo propuesto tiene como finalidad el desarrollo de una aplicación que permita la integración de distintos equipos y tecnologías heterogéneas de laboratorio correspondientes a un sistema multi-tanque, que corresponde a una representación didáctica de los problemas de control de nivel que suelen ocurrir en la industria.

Este proyecto propone la implementación, desde cero, de un sistema de control multi-tanque en una arquitectura CIM truncada. La arquitectura CIM en su totalidad se divide en 5 niveles jerárquicos [7] que se explican a continuación.

- Nivel 1: Campo
  Es el nivel inferior de la jerarquía CIM. En este nivel se ubican los dispositivos de campo que interactúan con el proceso tales como sensores y actuadores.

- Nivel 2: Control
  En este nivel se efectúa el control de operaciones de los dispositivos de fabricación. Se encuentra en este nivel el controlador de cada recurso individual. Por ejemplo, máquinas-herramienta, robots, sistemas de medición, sistemas de transporte.

- Nivel 3: Supervisión
  Se realizan funciones de coordinación de máquinas y operaciones. En él se sitúa el sistema de control que secuencia y controla una tarea específica. Gestiona los recursos y materiales.

- Nivel 4: Sistemas de ejecución de manufactura (MES)

A este nivel corresponden las funciones de planificación de la producción del conjunto de la factoría. Funciones de elaboración de secuencias de producción, secuenciamiento de tareas y coordinación de recursos en la planta.

- Nivel 5: Planificación de recursos empresariales (ERP)
  Este es el nivel superior y en él se realizan funciones de gestión de la empresa. Se establecen las políticas de producción del conjunto de la empresa en función de los recursos y costes del mercado.

La arquitectura CIM truncada se conforma únicamente de los primeros 3 niveles de la arquitectura total; la interconexión de éstas comprende el uso de redes industriales, los otros 2 niveles se pueden concebir dentro de una virtualización de servicios.

Las capas MES y ERP del modelo CIM truncado se virtualizarán parcialmente a través de una solución de internet industrial de las cosas (IIoT). Esta solución involucra la integración de al menos un *middleware* capaz de gestionar el protocolo abierto de comunicación (OPC), para solucionar la heterogeneidad de comunicación de equipamiento multimarca, además de integrar el flujo de información a la nube. La integración de la información de sensores y actuadores en la nube potencia la creación de gemelos digitales y la integración con agentes autónomos para toma de decisiones, almacenamiento y procesamiento de información, además de visualización remota a través de *dashboards*.

Finalmente, el proyecto también implica la instalación, programación e integración de soluciones de automatización industrial de un sistema multi-tanque del laboratorio de máquinas eléctricas de la Universidad de Cuenca.

1.4. Objetivos

1.4.1. Objetivo general

Desplegar una aplicación de internet industrial de las cosas (IIoT) en equipamiento de laboratorio que implica la integración de tecnologías de comunicación heterogéneas para la automatización de un sistema multi-tanque.

1.4.2. Objetivos específicos

El presente trabajo tiene los siguientes objetivos específicos:

- Revisar el estado del arte relacionado a estructuras, arquitecturas, protocolos y *software* aplicable a Industria 4.0.
- Integrar los dispositivos (equipos) de laboratorio mediante la construcción y despliegue de un tablero de control.
- Programar los equipos de automatización para una correcta operación del sistema multi-tanque.
- Desarrollar la solución de IIoT para procesamiento y despliegue de datos de proceso.
- Virtualizar el servicio de detección de fallas de la capa de sistemas de ejecución de manufactura (MES) en la solución IIoT.

1.5. Contribuciones del trabajo de titulación

Este trabajo presenta las siguientes contribuciones.

- Un sistema multi-tanque conformado por dos tanques principales y un tanque de reserva o bombeo, además de una bomba, tres electroválvulas y dos pares de sensores diferentes.
- Un tablero de control automático o manual donde se integran los equipos PLC SIEMENS, PLC Controllino, panel HMI, placa Up Squared Two, enrutador y fuentes mediante el cableado hacia el sistema multi-tanque y sus dispositivos, botones y diodos emisores de luz (Light Emitting Diode (LED)).
- Un sistema de control para el sistema multi-tanque.
- La virtualización de servicios mediante la implementación de un agente detector de fallas en un servidor de contenedores.
- Visualización de comportamiento del sistema desde el panel HMI y diferentes *dashboards* implementados en herramientas de *software* libre.
- Un sistema de consulta del comportamiento del sistema mediante un *chatbot* implementado en una herramienta de *software* libre.

Capítulo 2

Fundamentos teóricos

Este capítulo, presenta los principales conceptos relacionados con la Industria 4.0, el internet industrial de las cosas (IIoT), las redes industriales, arquitectura CIM, comunicación OPC, algoritmos de control, *software* Docker, métodos de detección de fallas, servicios de almacenamiento en la nube y *chatbot*.

2.1. Industria 4.0

Las tres primeras revoluciones industriales se caracterizan por estar impulsadas por la producción mecánica que depende de la energía del agua y del vapor, el uso de mano de obra masiva y energía eléctrica, y el uso de producción electrónica automatizada, respectivamente. La cuarta revolución industrial, conocida como Industria 4.0, se propuso por primera vez en 2011 con el objetivo de desarrollar la economía alemana. Esta revolución se caracteriza por su dependencia del uso de sistemas ciberfísicos (CPS) capaces de comunicarse entre sí y de tomar decisiones autónomas y descentralizadas, con el fin de incrementar la eficiencia industrial, la productividad, la seguridad y la transparencia [8].

2.1.1. Sistemas ciberfísicos (CPS)

Lo que distingue a CPS de los sistemas de información y comunicaciones más convencionales es el carácter en tiempo real de sus interacciones con el mundo físico. Si bien los sistemas CPS y los sistemas de tecnologías de información y comunicación (Information and Communications Technology (ICT)) procesan datos y/o información, el enfoque de CPS está en el control de los procesos físicos. Éstos utilizan sensores para recibir información, incluidas las mediciones de parámetros físicos y actuadores para participar en el control de los procesos físicos; además, a menudo implican un alto grado de autonomía [8].

Las soluciones técnicas que están conectadas a través del internet de las cosas (IoT) se pueden definir como CPS. En el área de fabricación y manufactura, estas instancias se denominan sistemas de producción ciberfísicos (Cyber-Physical Production Systems (CPPS)). Estos sistemas tienen como objetivo cerrar la brecha entre el dominio físico y el digital [9]. Por lo tanto, no solo se necesita de una infraestructura básica, sino también

soluciones inteligentes para la

- interacción entre los sistemas físicos y los humanos (interacción humano-máquina);
- reflejo de los objetos físicos en el mundo de la información (sombra digital de la producción como modelo de valor en tiempo real de los sistemas de valor agregado);
- transacción en términos de servicios de software;
- interoperación que se habilita mediante la interacción;
- plataformas basadas en la nube;
- prescripción para manejar *big data* con el objetivo de recuperar información, y;
- comunicación nueva e inesperada utilizando los conceptos y soluciones provenientes de la arquitectura de red IoT.

2.1.2. Internet industrial

El Internet industrial proporciona una forma de obtener una mejor visibilidad y conocimiento de las operaciones y los activos de la empresa a través de la integración de sensores de máquinas, *middleware*, *software* y sistemas de almacenamiento y computación en la nube *back-end*. Por lo tanto, proporciona un método para transformar los procesos operativos comerciales utilizando como retroalimentación los resultados obtenidos al interrogar grandes conjuntos de datos a través de análisis avanzados. Las ganancias comerciales se logran a través de ganancias de eficiencia operativa y productividad acelerada, lo que resulta en una reducción del tiempo de inactividad no planificado y una eficiencia optimizada y, por lo tanto, ganancias [4].

2.2. Internet industrial de las cosas (IIoT)

IoT es la proliferación de dispositivos inteligentes como tabletas, teléfonos, electrodomésticos como televisores y otros sensores, *etc*. El beneficio de usar dispositivos inteligentes en el hogar sería reducir las facturas de electricidad y ahorrar tiempo, *etc*. Gestionar el uso de recursos basado en sensores o programar tareas pesadas como hacer funcionar el lavavajillas, la lavadora o la secadora cuando el consumo eléctrico es el más barato. Los dispositivos de IoT son comúnmente utilizados por el aficionado u otro uso del consumidor, e incluso en la industria. Sin embargo, IIoT está diseñado para tareas pesadas como fabricación, monitoreo, *etc*. Por lo tanto, el internet industrial de las cosas utiliza dispositivos, actuadores, sensores, *etc*., más precisos y duraderos (resistentes al calor/frío). Tanto IoT como IIoT tienen los mismos principios básicos, tales como gestión de datos, red, seguridad, nube, *etc*. Las principales diferencias entre estos dos son la escalabilidad y el volumen de datos generados y cómo se manejan estos datos [10].

2.2.1. Arquitectura de referencia de IIoT

De acuerdo a [11], IIoT lleva el concepto de internet de las cosas al nivel empresarial. Cada empresa tiene sus grupos de dispositivos únicos con interfaces limitadas. Ahora, considerando los desafíos, no existe una solución única que resuelva todo el problema. Los componentes clave de una arquitectura IIoT se detallan a continuación y se presentan visualmente en la Figura 2.1:

- Sistema de control industrial: término general utilizado para definir la integración de *software* y *hardware* para controlar la infraestructura crítica. Generalmente se desarrolla utilizando un sistema de control distribuido (Distributed Control System (DCS)), control lógico programable (PLC), sistema de control

de supervisión y adquisición de datos (Supervisory Control And Data Acquisition (SCADA)), unidades terminales remotas (Remote Terminal Unit (RTU)), servidores de control, interfaz hombre-máquina (HMI), dispositivo electrónico inteligente (Intelligent Electronic Device (IED)) y muchos otros sistemas específicos de la industria.

- Dispositivos: sensores, intérpretes, traductores. Estos son algunos dispositivos específicos que se utilizan en la industria para proporcionar datos al usuario final de la aplicación. Proporcionan interacción máquina a máquina, interacción persona a máquina y viceversa.
- Almacenamiento transitorio: componente esclavo de la arquitectura maestra donde la representación transitoria de los objetos de datos se almacena temporalmente, lo que garantiza la durabilidad durante fallas de operación y del sistema, incluidas las redes.
- Procesadores locales: sistema de procesamiento de datos de baja latencia que proporciona un procesamiento rápido de datos. Se puede integrar con el propio dispositivo para el procesamiento de datos. Este procesador se puede clasificar en filtros de datos, administradores de eventos, procesadores de datos, motor basado en reglas, detectores de señales, algoritmos, enrutadores, *etc*.
- Aplicación: proporcionan información sobre las operaciones de campo en tiempo real, ayudan al personal a administrar los dispositivos, interactuar con otros sistemas y manipular los datos. Las notificaciones, alertas y visualización les ayudan a tomar decisiones efectivas y calculadas.
- Canales: medios de intercambio de datos entre el sistema y la aplicación. Incluye protocolo de redes, comunicaciones por satélite, interfaz de programación de aplicaciones Application Programming Interface (API), enrutadores, *etc*.
- Puertas de enlace: las puertas de enlace proporcionan una conexión a través de varias redes y protocolos que permiten la transferencia de datos entre diferentes dispositivos IIoT. Incluye enrutadores de señales inteligentes, protocolos de transferencia de información, *etc*.
- Recopiladores: reúnen datos de las puertas de enlace utilizando protocolos estándar. Puede ser hecho a medida, este tipo de dispositivos varían de una industria a otra dependiendo de las necesidades.
- Procesadores: los procesadores son el corazón de cualquier tipo de solución IIoT. Sus funciones principales incluyen transformaciones de datos, detección de señales, modelos analíticos, procesamiento de eventos complejos, *etc*.
- Almacenamiento de datos permanente: sistema de almacenamiento de datos de larga duración conectado al sistema IIoT. Trabajan como historiadores de los dispositivos junto con datos de diferentes fuentes que alimentan los datos a los procesadores para el procesamiento analítico avanzado y la preparación de modelos.
- Modelos: existen básicamente dos tipos de modelos, uno es el modelo analítico y otro es el modelo de datos. Los modelos de datos proporcionan una estructura a los datos, mientras que los modelos analíticos son construcciones personalizadas para satisfacer las necesidades específicas de la industria.
- Seguridad: aspecto importante del sistema basado en IIoT. Corre a través de las tuberías desde la fuente hasta el consumo. Incluye autorización de datos, cifrado, autenticación, gestión de usuarios, cortafuegos, enmascaramiento, *etc*.
- Entornos informáticos: el entorno mencionado varía de una industria a otra según las necesidades comerciales y su panorama.
- Computación de niebla: acerca los análisis analíticos a la fuente.
- Computación en la nube: análisis de escala global en toda la industria.
- Computación híbrida: combinación de computación en la nube y niebla que optimiza las operaciones

adaptadas a las necesidades de campos específicos.

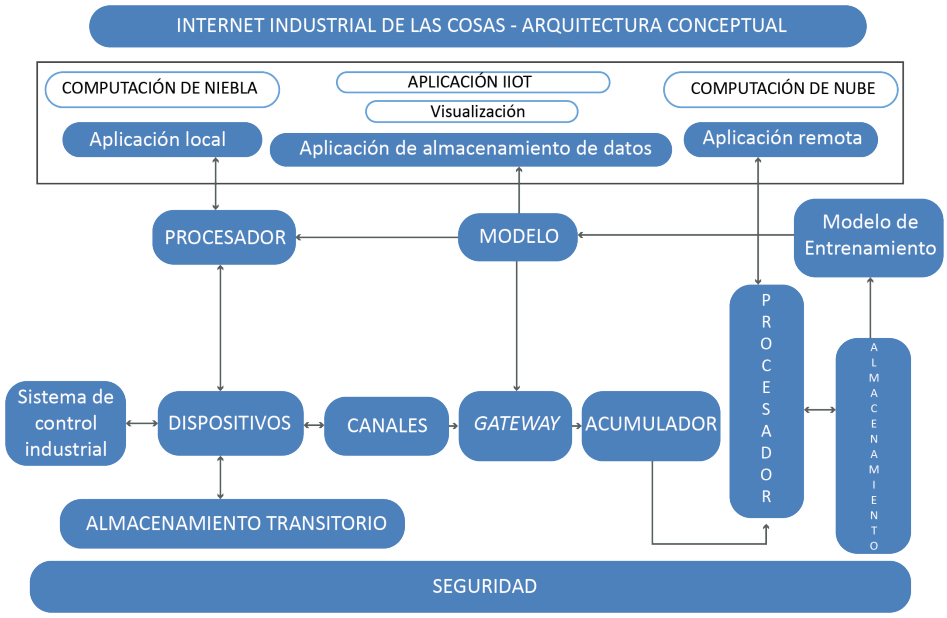


Figura 2.1: Arquitectura conceptual de IIoT

2.2.2. IIoT y la Industria 4.0

La Industria 4.0 se compone de prácticas industriales y de fabricación tradicionales con el aumento del mundo tecnológico que nos rodea. Esto incluye el uso de la participación a gran escala de M2M e IoT que ayuda a los fabricantes y consumidores de la misma manera con el aumento de la automatización, la comunicación y el monitoreo desarrollados, junto con el auto diagnóstico y nuevos niveles de análisis para proporcionar un futuro productivo. La Industria 4.0, toma una amplia tecnología de la información y la comunicación que trae la evolución en la cadena de suministro y producción que se traduce en el desarrollo tanto de la automatización como de la digitalización. Significa auto optimización, auto configuración en máquinas e incluso inteligencia artificial para completar tareas complejas, esto se traduce en eficiencias de costos y bienes o servicios de mejor calidad. Los intereses y requisitos de las industrias antes mencionados llevaron el inicio de la evolución de IoT a la nueva era de la tecnología, conocida como IIoT [11].

2.2.3. IIoT y la automatización industrial

IoT en la automatización industrial, que se transformó del nivel comercial al industrial, trae la tecnología contemporánea y revolucionaria de la Industria 4.0, conocida como IIoT. El internet industrial de las cosas improvisará sobre cómo se diseñan y utilizan las redes de automatización industrial, tanto ahora como en el futuro. También aumentará la productividad de la red de automatización industrial con la gran cantidad de dispositivos finales conectados actualmente, considerando el valor de los nuevos datos que están disponibles en el dispositivo final y la categorización de las prácticas de seguridad cibernética. El IIoT puede proteger el sistema industrial del tiempo de inactividad al detectar la señal más temprana de mal funcionamiento y prevenir que se agrave. La instalación de sensores en un sistema industrial permite a los operadores saber qué está sucediendo con los equipos casi en tiempo real, con alta precisión. Los sensores proporcionan un informe detallado y/o

información sobre el estado, el rendimiento y la seguridad de los equipos. Cuando uno puede predecir problemas del sistema, también puede evitar fallas y funcionamiento inadecuado. En conjunto, IIoT contribuye mucho en las industrias recientes en términos de velocidad de respuesta, rendimiento, precisión y seguridad del sistema, lo que sitúa la automatización industrial en el siguiente nivel [11].

2.3. Redes industriales y IIoT

Dado que los dispositivos de IIoT generan una gran cantidad de datos, las redes industriales requieren transmisión de datos, *big data*, aprendizaje automático o prácticas de inteligencia artificial. En una red doméstica, la pérdida de los datos generados sería trivial, pero en este tipo de redes industriales es vital. Los datos en IIoT deberían ser más precisos, continuos y sensibles [10].

2.3.1. Retos de las redes industriales

Las redes industriales están sujetas a muchos desafíos técnicos que deben considerarse antes, durante o después de cualquier implementación. Según [10], estos desafíos son los siguientes, pero no se limitan a:

- Coexistencia inalámbrica: significa el funcionamiento seguro de los dispositivos que utilizan tecnología inalámbrica para su comunicación. Especialmente, la interferencia de la señal es el principal problema contra la coexistencia de diferentes estándares de comunicación inalámbrica. Por lo tanto, al diseñar IIoT, se debe mantener la coexistencia inalámbrica de los dispositivos, por ejemplo, proporcionando suficiente distancia física entre los dispositivos o dividiendo y compartiendo de manera eficiente el espectro de frecuencia que se está utilizando.
- Latencia: a veces denominado retraso (para las redes industriales) es el tiempo transcurrido entre el tiempo de liberación de un comando específico y el tiempo de inicio de la ejecución de ese comando específico. En algunos casos específicos, también podría denominarse el tiempo transcurrido entre la recopilación de datos y el resultado de la reacción.
- Interoperabilidad: se refiere a la capacidad esencial de varias mercancías o sistemas computarizados para conectarse e intercambiar datos rápidamente entre sí, sin enfrentar ninguna restricción.
- Transmisión de datos del sensor: herramienta novedosa y poderosa para redes industriales, que mejorará las capacidades de los equipos de análisis de datos. Sin embargo, hay varios desafíos que deben considerarse: escalabilidad, durabilidad de los datos, tolerancia a fallas.
- Seguridad: los sensores y dispositivos IIoT desempeñan un papel fundamental en la seguridad industrial. IIoT aprovecha los dispositivos de bajo costo y bajo consumo para implementar protocolos de seguridad eficientes y, al mismo tiempo, mejora la productividad. El enfoque principal de la seguridad está en los problemas internos basados en el riesgo en la serie de producción para prevenir problemas de amplio espectro como el trabajo diario relacionado con pequeños accidentes y grandes desastres como accidentes e incidentes nucleares.
- Protección y privacidad: la interconexión de las fábricas inteligentes con los CPS y el IIoT mejora la inteligencia de las infraestructuras, pero introduce vulnerabilidades de seguridad cibernética que pueden conducir a problemas críticos como fallas del sistema, violaciones de la privacidad y/o violaciones de la integridad de los datos. A medida que la privacidad de los ciudadanos adquiere importancia, la privacidad que contiene información de los usuarios de IIoT, como la información de identificación personal, debe mantenerse confidencial.

2.3.2. Protocolos de las redes industriales

Según [12], los protocolos de comunicación industrial se pueden clasificar en algunos tipos, que son los siguientes: Primero, la red de sensores inalámbricos incluye WirelessHART, IEC 62591 (WirelessHART), ISA 100.11a y Zigbee. En segundo lugar, la comunicación M2M incluye CoAP, OPC-UA, DDS y Modbus. En tercer lugar, la mensajería incluye MQTT, AMQP y XMPP. En cuarto lugar, la red de área amplia de bajo consumo (Low Power Wide Area Network (LPWAN)) incluye NB-IoT, Sigfox, LoRa y LoRaWAN. En quinto lugar, la red celular incluye 5G, 4G, 3G, LTE, WiMax, GPRS y GMS. En sexto lugar, la red de área local inalámbrica (wireless Local Area Network (WLAN)) incluye IEEE 802.11. En séptimo lugar, la red de área personal inalámbrica (Wireless Personal Area Network (WPAN)) incluye IEEE 802.15.4.

2.3.3. Integración de la red de comunicaciones industriales

Se consigue la integración de los diferentes equipos y dispositivos existentes en una industria dividiendo las tareas entre grupos de procesadores con una organización jerárquica. Así, dependiendo de la función y el tipo de conexiones, se suelen distinguir cinco niveles en una red industrial [13], éstos se explican en la siguiente sección.

2.4. Manufacturada integrada por computador (CIM)

La fabricación digital se hizo prominente en la década de 1980, y es entonces cuando los fabricantes de máquinas y herramientas, y otros tipos de fabricantes desarrollaron y promocionaron CIM [14]. CIM se utiliza para describir el proceso de integración de todos los elementos que intervienen en la fabricación, mediante técnicas informáticas. CIM se refiere a un enfoque global, en un entorno industrial, que tiene como objetivo mejorar el rendimiento industrial. Este enfoque se aplica de manera integrada a todas las actividades, desde el diseño hasta la entrega y la post venta, y utiliza varios métodos, medios y técnicas para mejorar simultáneamente la productividad, disminuir los costos, cumplir con las fechas de vencimiento, aumentar la calidad del producto, asegurar la flexibilidad a nivel local o global en un sistema de fabricación, e involucrar a todos los actores. En tal enfoque, los aspectos económicos, sociales y humanos son al menos tan importantes como los aspectos técnicos [15].

2.4.1. Arquitectura CIM

En la Sección 1.3 ya se realizó una introducción a la arquitectura CIM, en esta sección se profundiza más en el tema. La arquitectura CIM en su totalidad se divide en 5 niveles jerárquicos, como se observa en la Figura 2.2, el primero es el nivel de campo o proceso, el segundo es el de control o máquina, el tercero de supervisión, el cuarto de gerencia o sistemas de ejecución de manufactura (MES) y el quinto de empresa o planificación de recursos empresariales (ERP). Los diferentes niveles de integración de la pirámide CIM se describen a continuación [16]:

- Nivel de campo: se obtienen datos desde el proceso o máquina por medio de dispositivos de entrada (sensores) y de acuerdo a las referencias pre-ajustadas se corrigen las variables a controlar, por medio de dispositivos de salida (actuadores) y así mantener el buen funcionamiento de la máquina.
- Nivel de control: en este nivel se organiza la información recabada del nivel anterior para presentar y analizar informes de los valores de las variables controladas, así como reportes de alarmas.

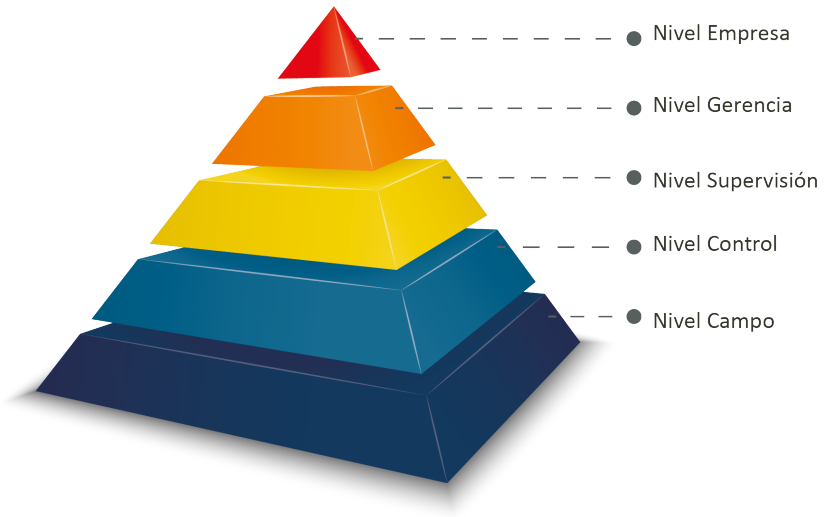


Figura 2.2: Arquitectura CIM

- Nivel de supervisión: se realiza la planificación de los trabajos a realizar y la administración de planta productiva basada en la información obtenida del nivel de control, concentrándose en las tareas de seguridad del personal, continuidad de la producción y mantenimiento.
- Nivel MES: se enfoca a la organización de los trabajos, la obtención y la administración de los recursos. Se planifica la producción, los niveles de aseguramiento de calidad y el mantenimiento de la planta productiva basándose en la información obtenida desde la base de la pirámide CIM.
- Nivel ERP: en este nivel se integra la información obtenida en los niveles inferiores para realizar la logística general de la planta productiva, basado en los objetivos estratégicos planteados en la planificación enfocada al corto, mediano y largo plazo tomando en cuenta las actividades principales de la administración que son las ventas, compra de insumos y comercialización.

Cada uno de los niveles de CIM, además de llevar a cabo tareas específicas, realiza un tratamiento y filtrado de la información que es transmitida en sentido ascendente o descendente a través de la pirámide. De esta forma se limita el flujo de información a los estrictamente necesarios en cada nivel. También existe un tráfico en sentido horizontal dentro de cada nivel, con distinciones en cada uno de ellos.

Esta estructura no es universal, varía con el tamaño del sistema de fabricación y sus características particulares. Además, para cualquiera de los niveles, no hay un estándar universalmente aceptado que cubra todos los aspectos desde el nivel físico al de aplicación [13].

2.4.2. Bus de campo

Los buses de campo son redes digitales, bidireccionales, multipunto, montadas sobre un bus serie que tiene como función la interconexión de los dispositivos actuador/sensor correspondientes al nivel de campo o proceso [17].

Con base en lo expuesto en la Sección 2.3.2 y [17] a continuación se presenta una descripción general sobre ciertos buses de campo.

2.4.2.1. Modbus TCP/IP

En el área de los protocolos de comunicación industrial, el estándar de la industria Modbus puede conectar dispositivos electrónicos industriales [12]. Modbus es un protocolo que principalmente fue diseñado para que los controladores programables de Modicon, marca registrada por Gould Inc., puedan comunicarse con otros controladores además de otros dispositivos de otras redes. Para realizar la transmisión de datos Modbus utiliza los modelos maestro-esclavo y cliente-servidor, este protocolo es ideal para la monitorización remota de los elementos de campo. Modbus es un protocolo de enlace que utiliza el estándar serie RS-232C [17].

Entre variadas características que se pueden mencionar sobre este protocolo, la comunicación es una de las más importantes. La comunicación es de tipo asíncrona. El intercambio de mensajes puede ser de dos tipos:

- Intercambios punto a punto: aquí se mantiene el modelo de intercambio maestro-esclavo, en donde el maestro realiza un requerimiento y el esclavo da una respuesta.
- Mensajes difundidos - *broadcast*: en este tipo de comunicación el maestro envía un mismo mensaje a todos los esclavos sin recibir respuesta de los mismos. Por ejemplo, este tipo de comunicación se utiliza cuando se requiere enviar datos comunes de configuraciones.

Modbus TCP/IP es una extensión del protocolo Modbus que permite utilizarlo sobre la capa de transporte TCP/IP. Este protocolo nació de la idea de lograr la monitorización de una red de manera remota desde cualquier parte del mundo, gracias a que Modbus TCP/IP se puede usar en internet.

La combinación de una red física versátil y escalable como *Ethernet* con el estándar TCP/IP y una representación de datos independiente del fabricante, como Modbus, proporciona una red abierta y accesible para el intercambio de datos de proceso [17].

2.4.2.2. Profibus

Es un protocolo dinámico que brinda solución a tareas de comunicación maestro-esclavo e involucra todos los perfiles de comunicación industrial como automatización de procesos, seguridad y control de datos. Hoy en día, este protocolo abarca todos los niveles de automatización, desde elementos de campo, como sensores y/o actuadores, hasta sistemas complejos que gestionan múltiples equipos con toda su información [17].

Con Profibus llegó el estándar de bus de campo que era libre y transparente a los fabricantes. Los dispositivos de diferentes fabricantes eran adquiridos con su correcta interfaz.

2.4.2.3. Profinet

Profinet es Profibus sobre *Ethernet*. Este protocolo ofrece soluciones de red para fábricas y procesos de automatización, para aplicaciones de seguridad y aplicaciones de control de movimiento sincronizado. La comunicación Profinet se basa en protocolos *Ethernet*, UDP, TCP e IP [18]. Existen dos versiones de redes Profinet:

- Profinet I/O: funciona desde la integración de dispositivos de campo descentralizados simples y aplicaciones de tiempo crítico [18]. Entre las características físicas más destacadas están: alta resistencia a condiciones de humedad, condensaciones, temperaturas extremas, vibraciones e interferencias electromagnéticas [17].

- Profinet de automatización basada en componentes (Component Based Automation (CBA)): se ocupa de la integración de los sistemas de automatización distribuidos basados en componentes [18].

2.5. Plataforma de comunicaciones abiertas (OPC)

De acuerdo con el propio sitio web de la Fundación OPC [19], que es la responsable del desarrollo y mantenimiento de este estándar, OPC es el estándar de interoperabilidad para el intercambio seguro y confiable de datos en el espacio de automatización industrial y en otras industrias. Es independiente de la plataforma y garantiza el flujo continuo de información entre dispositivos de múltiples proveedores.

El estándar OPC es una serie de especificaciones desarrolladas por proveedores de la industria, usuarios finales y desarrolladores de *software*. Estas especificaciones definen la interfaz entre clientes y servidores, así como también entre servidores, incluido el acceso a datos en tiempo real, monitoreo de alarmas y eventos, acceso a datos históricos y otras aplicaciones.

2.5.1. Arquitectura OPC unificada (OPC-UA)

Con la introducción de arquitecturas orientadas a servicios en los sistemas de fabricación, surgieron nuevos desafíos en seguridad y modelado de datos. La Fundación OPC [19] desarrolló las especificaciones OPC de arquitectura unificada, conocido como OPC-UA, para abordar estas necesidades y, al mismo tiempo, proporcionó una arquitectura de plataforma abierta con tecnología rica en funciones, preparada para el futuro, escalable y extensible.

El núcleo de la norma de comunicación de OPC-UA es la interoperabilidad y la estandarización. La tecnología OPC tradicional resuelve el problema de la interoperabilidad entre los dispositivos de *hardware* a nivel de control; por otro lado, también se requiere estandarización de la comunicación a nivel de empresa [12], bajo este contexto OPC-UA permite la integración horizontal y vertical de información de sensores/actuadores/máquinas a ERP [20].

Para el intercambio de información, OPC-UA ofrece dos mecanismos de comunicación [20]: el primero es un modelo cliente-servidor donde el cliente accede a la información proporcionada por el servidor a través de servicios definidos; el segundo método de comunicación es OPC-UA PubSub, que se encuentra en operación, en su mayoría sin redes sensibles al tiempo (Time-Sensitive Network (TSN)).

Hasta hace poco, OPC-UA ha tenido limitaciones en términos de procesos críticos en tiempo real. En este sentido, se está desarrollando y verificando el modelo de suscriptor-editor, conocido como OPC UA PubSub, y las comunicaciones en tiempo real utilizando TSN. Dos métodos están disponibles para diferentes escenarios:

1. OPC UA PubSub para mensajería a través de redes de áreas locales (Local Area Network (LAN)). Los datos serán multidifusión a través del protocolo de datagramas de usuario (User Datagram Protocol (UDP)) por un servidor OPC-UA (editor) para el consumo de cualquier número de clientes autorizados (suscriptores). Esto permitirá una distribución de datos extremadamente eficiente sin intermediación.
2. OPC UA PubSub para mensajería a través de redes de áreas globales (Wide Area Network (WAN)). Este método proporciona un método seguro y altamente escalable para compartir datos de cualquier número de

editores con cualquier número de suscriptores habilitados para OPC-UA. Para el intercambio de datos en redes WAN, la especificación OPC UA PubSub define asignaciones en los protocolos de mensajería más relevantes, como el de transporte de mensajería por colas Message Queue Telemetry Transport (MQTT) y el de mensajería por colas avanzada Advanced Message Queuing Protocol (AMQP).

La forma más sencilla para que un cliente y un servidor intercambien datos es utilizando los servicios de lectura (*READ*) y escritura (*WRITE*) de OPC-UA, que permiten a un cliente leer y escribir uno o más atributos de nodos, mantenidos en el espacio de direcciones del servidor (*AdressSpace*) de OPC-UA. Como la mayoría de los otros servicios, los servicios de lectura y escritura están optimizados para operaciones de lectura/escritura masivas y no para lectura/escritura de valores individuales [21].

Una forma diferente y más sofisticada de acceder a los datos se basa en suscripciones y elementos supervisados; este es el método preferido para los clientes que necesitan actualizaciones cíclicas de los valores de las variables. Las suscripciones y los elementos supervisados se colocan en la parte superior de un nivel de sesión, que es una conexión lógica entre un cliente OPC-UA y un servidor OPC-UA creado en el contexto de un canal seguro.

2.6. Algoritmos de control

Durante la conferencia de tecnologías emergentes y automatización de fábrica (Emerging Technologies and Factory Automation (EFTA)) celebrada en Chipre 2017, el control se presentó como un componente valioso y crítico de la Industria 4.0 en muchas presentaciones y paneles de discusión con especialistas y académicos de la industria. La motivación fue que fusionar la automatización con la tecnología de la información requiere características en las que el control juega un papel clave que es vital para el desempeño del proceso en general [22].

2.6.1. Control proporcional-integral-derivativo (PID)

Un controlador PID es un controlador de tres términos que tiene una larga historia en el campo del control automático, desde principios del siglo pasado. Debido a su intuición y su relativa simplicidad, además de un rendimiento satisfactorio que puede proporcionar con una amplia gama de procesos, se ha convertido en el controlador estándar en entornos industriales [23].

Un controlador PID se puede encontrar en prácticamente todo tipo de equipos de control, ya sea como controlador autónomo o como bloque funcional en controladores lógicos programables (PLC) [23]. De acuerdo con [24], el control PID sigue siendo una herramienta de control importante por tres razones: historial de éxito, amplia disponibilidad y simplicidad de uso. Estas razones se refuerzan entre sí, asegurando que el marco más general de control digital con controladores de orden superior no haya podido realmente desplazar el control PID.

El marco de señales de entradas y salidas para el controlador de tres términos se muestra en la Figura 2.3 y se usa para discutir los tres términos del controlador PID.

- Control proporcional: se indica mediante el terminal P del controlador PID. Se utiliza cuando la acción del controlador debe ser proporcional al tamaño de la señal de error del proceso $e(t) = r(t) - y_m(t)$. Las representaciones de tiempo y dominio de *Laplace* para el control proporcional se dan como:

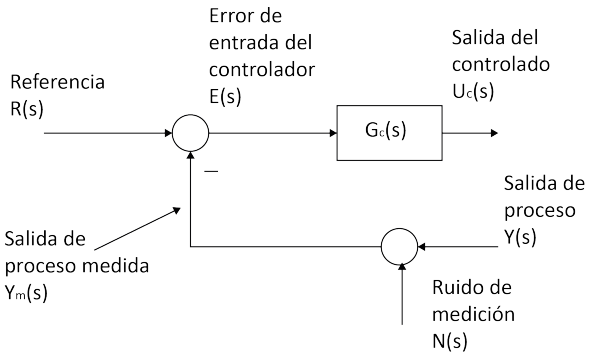


Figura 2.3: Entradas y salidas del controlador.

dominio del tiempo: $u_c(t) = k_P e(t)$,
dominio de *Laplace*: $U_c(s) = k_P E(s)$,

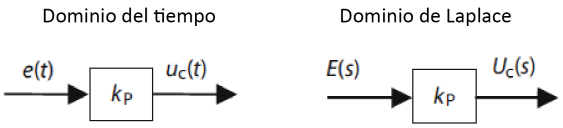


Figura 2.4: Diagrama de bloques: control proporcional.

donde la ganancia proporcional se denota como k_P [24]. La Figura 2.4 muestra los diagramas de bloques para el control proporcional.

- Control integral: se denota por el término I en el controlador PID y se usa cuando se requiere que el controlador corrija cualquier desviación estable de un valor de señal de referencia constante. El control integral supera la deficiencia del control proporcional al eliminar la compensación sin el uso de una ganancia de controlador excesivamente grande. Las representaciones de tiempo y dominio de *Laplace* para el control integral se dan como:
  dominio del tiempo: $u_c(t) = k_I \int^t e(\tau)d\tau$,
  dominio de *Laplace*: $U_c(s) = \left[\frac{k_I}{s}\right] E(s)$,

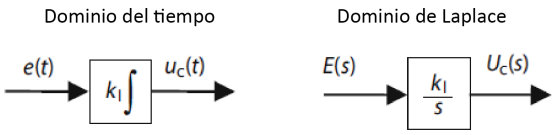


Figura 2.5: Diagrama de bloques: control integral.

donde la ganancia integral del controlador se denota como k_I [24]. Los diagramas de bloques de tiempo y de *Laplace* se muestran en la Figura 2.5.

- Control derivativo: si un controlador puede usar la tasa de cambio de una señal de error como entrada, entonces esto introduce un elemento de predicción en la acción de control. El control derivativo usa la tasa de cambio de una señal de error y es el término D en el controlador PID. Las representaciones de

tiempo y dominio de *Laplace* para el control derivado se dan como:
dominio del tiempo: $u_c(t) = k_D \frac{de}{dt}$,
dominio de *Laplace*: $U_c(s) = [k_D s]E(s)$,

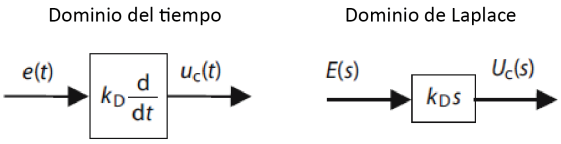


Figura 2.6: Diagrama de bloques: control derivativo.

donde la ganancia del control derivado se denota como k_D [24]. Esta forma particular se denomina control derivado puro, para lo cual las representaciones del diagrama de bloques se muestran en la Figura 2.6.

2.6.2. Control On-Off

Uno de los controladores más adoptados es sin duda el controlador On-Off, donde la variable de control puede asumir solo dos valores, u_{max} y u_{min}, dependiendo del signo de la señal de error de control. Formalmente, la ley de control se define como presenta la Ecuación 2.1:

$$u = \begin{cases} u_{\max} & si \quad e > 0 \\ u_{\min} & si \quad e < 0 \end{cases} \tag{2.1}$$

es decir, la variable de control se establece en su valor máximo cuando el error de control es positivo y en su valor mínimo cuando el error de control es negativo [23].

2.7. Docker

Docker es una plataforma de código abierto que ejecuta aplicaciones y facilita el desarrollo y la distribución del proceso. Las aplicaciones que están integradas en Docker están empaquetadas con todas las dependencias de soporte en un forma estándar llamada contenedor. Estos contenedores se ejecutan de forma aislada sobre el *kernel* del sistema operativo [25]. Antes de que apareciera Docker, la canalización de desarrollo generalmente involucraba combinaciones de varias tecnologías para administrar el movimiento de *software*, como máquinas virtuales, herramientas de administración de configuración, sistemas de administración de paquetes y redes complejas de dependencias de bibliotecas. Todas estas herramientas debían ser administradas y mantenidas por ingenieros especializados, y la mayoría tenían sus propias formas únicas de configurarse [26].

Docker proporciona una función para automatizar las aplicaciones cuando se implementan en contenedores. En un entorno de contenedor donde las aplicaciones se virtualizan y ejecutan, Docker agrega una capa superior adicional de implementación. La forma en que Docker está diseñada es para brindar un entorno rápido y liviano, donde el código se puede ejecutar de manera eficiente y, además, proporciona una facilidad adicional del proceso de trabajo competente para tomar el código de la computadora para probarlo antes de la producción [27], lo que a nivel de industria representa un beneficio.

Hay muchas razones por las cuales utilizar Docker [26], a continuación se listan algunas de ellas:

- reemplaza a las máquinas virtuales,
- es un *software* de creación de prototipos,
- es un *software* de empaque,
- habilita una arquitectura de microservicios,
- permite el modelado de redes,
- facilita la productividad completa (*full-stack*) sin conexión,
- reduce los gastos de depuración,
- permite la documentación de dependencias y puntos de contacto del *software*, y
- permite la entrega continua.

Hay cuatro componentes internos principales de Docker, incluidos cliente y servidor Docker, imágenes Docker, registros Docker y contenedores Docker; éstos se detallan a continuación.

2.7.1. Cliente y servidor Docker

Docker se puede explicar como una aplicación basada en cliente y servidor, como se muestra en la Figura 2.7. El servidor de Docker recibe la solicitud del cliente de Docker y luego la procesa en consecuencia. El demonio/servidor de Docker y el cliente de Docker se pueden ejecutar en la misma máquina o se puede conectar un cliente de Docker local con un servidor o demonio remoto, que se está ejecutando en otra máquina [28].

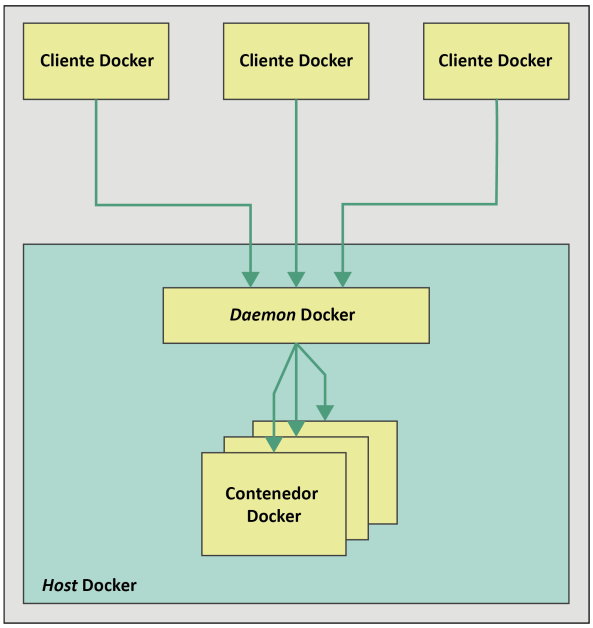


Figura 2.7: Arquitectura Docker.

2.7.2. Imágenes Docker

Existen dos métodos para crear una imagen Docker. El primero es crear una imagen utilizando una plantilla de solo lectura. La base de cada imagen Docker es una imagen base, que también se puede crear desde cero. El segundo método consiste en crear un archivo Docker conocido como *dockerfile*. El *dockerfile* contiene una lista de instrucciones que se ejecutan al usar el comando *"Docker build"* desde el terminal *bash*, y crea una imagen Docker [28].

2.7.3. Registros Docker

Las imágenes Docker se colocan en los registros Docker. Existen dos tipos de registros, públicos y privados. *Docker Hub* se denomina registro público, en el que todos pueden extraer imágenes disponibles y enviar sus propias imágenes sin crear una imagen desde cero. Las imágenes se pueden distribuir a un área en particular, ya sea pública o privada, utilizando la función *Docker Hub* mencionada [28].

2.7.4. Contenedores Docker

La imagen Docker crea un contenedor Docker. Los contenedores contienen todo el kit necesario para una aplicación, por lo que la aplicación se puede ejecutar de forma aislada [28].

2.7.5. Comandos clave de Docker

La función central de Docker es crear, enviar y ejecutar *software* en cualquier ubicación que tenga Docker. Para el usuario final, este es un programa de línea de comandos que se ejecutan. Como cualquier herramienta de control de fuente, este programa tiene subcomandos que realizan diferentes operaciones [26], éstos son:

- *docker build*: estructura o construye la imagen Docker,
- *docker run*: ejecuta la imagen Docker como un contenedor,
- *docker commit*: confirma un contenedor Docker como una imagen, y
- *docker tag*: etiqueta una imagen Docker.

2.8. Métodos de detección de fallas

En la actualidad, los equipos y sistemas industriales se han vuelto más complejos y costosos, con menos tolerancia a la degradación del rendimiento, la disminución de la productividad y los peligros de seguridad causados por fallas inesperadas, que estimulan una creciente demanda de detección y diagnóstico de fallas en tiempo real y técnicas de control tolerantes a fallas [29]. Las fallas son acciones impredecibles e internas que ocurren dentro de un sistema, auto generadas por el mal funcionamiento de uno o más componentes y que pueden llevar al sistema completo a un estado no deseado [30]. El principal objetivo de un agente detector de fallas es preservar la estabilidad del sistema y mantener un rendimiento de control aceptable en caso de fallas del sistema [31].

Las fallas pueden detectarse utilizando distintos métodos y técnicas de detección y diagnóstico, y pueden ser aplicadas dependiendo de las características dinámicas del sistema y del número de señales disponibles por variable [30]. De acuerdo con [30], existen muchos métodos de detección de fallas, a continuación se detallan algunas técnicas:

- Chequeo de límites y detección de cambios: chequeos de tendencias, estimación de media, varianza y pruebas estadísticas.
- Análisis para señales periódicas: filtros pasabanda, análisis de *Fourier*, transformada de *Fourier*, entre otras.
- Análisis de señales estocásticas: análisis de espectro, correlación.
- Análisis para señales periódicas no estacionarias: transformada *Wavelet*.
- Detección de fallas por medio de métodos de identificación de procesos: funciones de correlación de estimación, estimación de parámetros para procesos lineales y no lineales.

- Detección de fallas con ecuaciones de paridad, observadores de estado, estimación de estados, detección de fallas en lazos de control, detección de fallas por medio de análisis de componentes principales.
- Entre los métodos de diagnóstico están: clasificación de *Bayes*, clasificación polinomial, árboles de decisión, redes neuronales. Métodos de inferencia como lógica difusa, sistemas híbridos neuro-difusos, entre otros.

2.8.1. Análisis de componente principal (PCA)

Históricamente, el análisis estadístico multivariante y las técnicas de monitoreo de procesos estadísticos se han aplicado en una amplia gama de campos, incluidos la genómica, el procesamiento de señales y varios procesos industriales. PCA es una de las técnicas multivariantes más utilizadas con diversas aplicaciones [32].

PCA conserva la mayor variabilidad posible del conjunto de datos al encontrar un nuevo conjunto de variables o componentes principales que son combinaciones lineales de los del conjunto de datos original que maximizan sucesivamente la varianza y no están correlacionados entre sí. Estos nuevos conjuntos de componentes principales se obtienen resolviendo un problema de valores propios. PCA captura la varianza en m variables del conjunto de datos original en una dimensión reducida por l componentes principales retenidos.

Matemáticamente, PCA es la descomposición de vectores propios de la covarianza o la matriz de correlación obtenida de la matriz de datos $X \in R^{n\times m}$ que contiene n observaciones igualmente espaciadas (en el mismo intervalo de tiempo) de m variables de proceso y se escala a cero media y varianza unitaria, en un subespacio transformado de dimensión reducida. La matriz de covarianza muestral de X se define como indica la Ecuación 2.2.

$$cov(X) = \Sigma = \frac{X^T X}{n-1} \tag{2.2}$$

Entonces, la descomposición se expresa de acuerdo a la Ecuación 2.3:

$$X = TP^T = \tilde{X} + E \tag{2.3}$$

donde $T \in R^{nm}$ y $P \in R^{mm}$ son la matriz de puntuación y la matriz de carga, respectivamente. Las matrices $\tilde{X}$ y E representan la estimación de X y la parte residual del modelo PCA, respectivamente, éstas se definen de acuerdo a las Ecuaciones 2.4 y 2.5:

$$\tilde{X} = T_l P_l^T \triangleq \sum_{i=1}^{l} T_i P_i^T \tag{2.4}$$

$$E = T_{m-l} P_{m-l}^T \triangleq \sum_{i=l+1}^{m} T_i P_i^T \tag{2.5}$$

La proyección de componentes principales reduce el conjunto original de variables a l componentes principales donde l es igual o menor que la dimensión más pequeña de X. La descomposición asume que las cargas de componentes principales son ortonormales ($P_i^T P_j = 0$ para $i \neq j$, $P_i^T P_j = 1$ para $i = j$) y las puntuaciones de componentes principales no están correlacionadas. Los valores P_i son los autovectores de la matriz de covarianza, $\sum$ o correlación, dados por la Ecuación 2.6:

$$cov(X)P_i = \Lambda_i P_i \tag{2.6}$$

donde Λ es el valor propio asociado con el vector propio P_i. Las cargas contienen información sobre cómo se relacionan las variables entre sí, mientras que los vectores T_i son puntuaciones que contienen información sobre cómo se relacionan las muestras entre sí. Se debe tener en cuenta que para X y cualquier par de T_i, P_i, P_i tiene una relación con T_i como como indica la Ecuación 2.7:

$$XP_i = T_i \tag{2.7}$$

Esto se debe a que el vector de puntuación T_i es la combinación lineal de los datos X originales definidos por P_i. Los pares T_i, P_i están dispuestos en orden descendente de acuerdo con la Λ_i asociada, que es una medida de la cantidad de varianza descrita por los pares T_i, P_i. Además, como estos pares están en orden descendente de, Λ_i, el primer par captura la mayor cantidad de información de cualquier par en la descomposición [32].

2.8.2. Redes neuronales

Las redes neuronales son entrenadas para detectar y diagnosticar fallas usando un conjunto representativo de datos del proceso y el conocimiento de expertos. Este error no es más que la diferencia entre la decisión correcta realizada por el experto y los valores generados por la red neuronal en entrenamiento. El ajuste de los parámetros internos, conocidos como pesos de la red neuronal, permitirá tomar decisiones más reales en el momento de detectar y diagnosticar fallas [30]. Las redes neuronales han sido ampliamente utilizadas en tareas de identificación y diagnóstico de fallas dadas sus capacidades de aproximar cualquier función multivariada lineal o no lineal, a partir de datos [33].

Las redes neuronales se aplican para detectar y diagnosticar fallas especialmente cuando se tiene definidos los patrones (con fallas) de las variables del proceso. El proceso de detección de fallas se basa en la comparación de la respuesta actual y la repuesta anticipada del sistema. La respuesta anticipada del sistema es generada por la red neuronal basada en un modelo de predicción. En el proceso de diagnóstico de fallas se utilizará un clasificador neuronal, que permitirá la ubicación e identificación de la falla ocurrida [30].

Dentro de los clasificadores neuronales basados en estadística se tiene la técnica de máquina de vectores de soporte (Support Vector Machine (SVM)). Ésta es una técnica no lineal representativa y potencialmente eficaz para clasificar todo tipo de conjuntos de datos. La detección de fallas se puede considerar como un problema de clasificación especial involucrado en el método basado en modelos y el método basado en datos, con el propósito de reconocer oportunamente la condición de falla [34].

SVM es un enfoque estadístico multivariante y se ha vuelto popular debido a su efecto preferible de clasificación y regresión. El clasificador basado en SVM tiene una mejor propiedad de generalización porque se basa en el principio de minimización del riesgo estructural [35].

El conjunto de datos de entrenamiento que contiene dos clases corresponde a la matriz X con la forma mxn en la que m representa el número de muestras observadas, mientras que n representa la cantidad de variables observadas. x_i se denota como un vector de columna para representar la i-ésima columna de X. Se supone que cada muestra está en una clase positiva o en una clase negativa. Además, un vector de columna Y sirve como etiqueta de clase, que contiene dos entradas -1 y 1. Se denota que $y_i = 1$ está asociado con una clase y $y_i = -1$ con la otra clase. Si el conjunto de datos de entrenamiento es linealmente separable, SVM intentará separarlo

mediante un hiperplano lineal definido por la Ecuación 2.8:

$$f(x) = \langle w, x \rangle + b = 0 \tag{2.8}$$

donde w es un vector m-dimensional y b es un escalar. Los parámetros w y b deciden la posición y orientación del hiperplano separador. Un hiperplano de separación se considera óptimo si crea una distancia máxima entre los vectores más cercanos y el hiperplano. Los puntos más cercanos de cada clase son los que se conocen como vectores de soporte [34].

2.9. Servicios de almacenamiento en la nube

La transformación digital o digitalización se relaciona con la interacción entre los mundos físico e informático y consiste en la virtualización de elementos estructurales que reflejan la realidad, como productos, pedidos y recursos, gestionados en arquitecturas orientadas a servicios para la integración de las capas técnica y comercial de una empresa [36]. La fabricación digital se puede definir como la digitalización de las operaciones de suministro, producción y entrega de una empresa en red, y utiliza ontologías y modelos digitales [37].

Las empresas de fabricación y manufactura ya han adoptado la computación en la nube en las capas superiores de los procesos comerciales de suministro, *marketing* digital y ERP que, sin embargo, aún no están integrados en tiempo real a las capas de producción y logística [37].

La computación en la nube es el resultado de la computación en red, la computación de servicios públicos y la computación automática. La nube es un sistema informático distribuido y paralelo que consiste en un conjunto de ordenadores virtualizados e interconectados que proporciona uno o más recursos informáticos unificados en función de los requisitos entre los proveedores de servicios y los consumidores de servicios [38].

Según [39], la computación en la nube tiene clientes, centros de datos y servidores distribuidos como componentes. Éstos se detallan a continuación:

- Clientes: usuarios como computadoras, laptops, tabletas o teléfonos móviles.
- Centros de datos (*datacenters*): son una colección de servidores donde se aloja(n) aplicación(es). La virtualización se realiza donde se crean múltiples instancias de servidores virtuales.
- Servidor distribuido: servidores que no residen localmente y que están geográficamente lejos.

2.10. *Chatbot*

La inteligencia artificial continúa creciendo en popularidad en varias plataformas industriales, y se vuelve especialmente prominente en la tecnología *chatbot* [40]. Históricamente, los *chatbots* se han utilizado para realizar conversaciones de charla o tareas basadas en diálogos como soporte de ventas. Según esta muestra, la mayoría de las aplicaciones de *chatbots* están orientadas a tareas [41].

En un escenario de IoT, las interfaces de conversación se pueden integrar con plataformas de IoT. Por ejemplo, la vida asistida por el ambiente, donde el modelo propuesto utiliza un *chatbot* complementario y monitorea las variaciones emocionales en tiempo real en pacientes o ancianos facilitando el monitoreo personalizado de las

emociones [42]; de la misma forma el monitoreo de variables correspondiente a sistemas propios de empresas en el contexto IIoT. Aunque es imperativo integrar la conciencia de las emociones en las aplicaciones de *chatbot* industriales, el énfasis hacia la caracterización autónoma e inteligente de las emociones sigue siendo limitado [42].

Capítulo 3

Estado del arte

Este capítulo presenta un resumen y puntos importantes sobre estudios y artículos relacionados al control de sistemas conformados por varios tanques, sensores, bombas, etc., especialmente enfocados en el control PID; se profundiza también en IIoT y la Industria 4.0, arquitecturas para sistemas industriales, protocolo de comunicación OPC-UA, computación en la nube y virtualización de recursos. Para esto, la información se clasifica en tres secciones, la primera sobre el control, la segunda sobre la Industria 4.0 y la tercera sobre la integración de sistemas heterogéneos.

3.1. Estado del arte de sistemas de control de nivel multi-tanque

El diseño e implementación de un experimento remoto para el control el nivel de un sistema de dos tanques llevado a cabo en [43] fue uno de los primeros estudios analizados. Lo interesante de este artículo es su detallada propuesta de trabajo basada en aprendizaje colaborativo. Respecto a la arquitectura física de su trabajo en sí, el sistema incluye dos tanques, una bomba que permite la circulación del agua del tanque inferior al superior, dos sensores ultrasónicos que miden ambos niveles de los tanques, una electroválvula que detiene el flujo entre el tanque superior e inferior y una válvula manual por motivos de seguridad. El control se enfoca en el tanque superior y el tanque inferior sirve como *buffer*. Los algoritmos de control, On-Off y PID, se implementan en el microcontrolador que está conectado a una computadora personal mediante un enlace RS232. A través del programa *Watch Tank*, desarrollado con fines de monitoreo en la plataforma LabVIEW, el usuario puede elegir/modificar el tipo de controlador y los parámetros característicos, así como el nivel de referencia del tanque superior.

Lo destacable de [43] es que con este sistema, los usuarios del sistema pueden probar físicamente los algoritmos PID, en particular, la influencia de las ganancias proporcionales, integrales y derivativas en el rendimiento del controlador. Sin embargo, este control puede optimizarse y coordinarse con diferentes técnicas cuando la arquitectura física del sistema requiera utilizar más tanques, equipos o dispositivos. Esto, por ejemplo, lo realiza el estudio presentado en [44]. En éste, un controlador PID óptimo es diseñado para el control de nivel de un sistema de tres tanques. Utilizan otras técnicas como un enfoque basado en el error cuadrado integral (Integral Square Error (ISE)) para ajustar el controlador PID. El ISE de respuesta al escalón unitario

se minimiza para obtener una configuración óptima del controlador. La técnica de optimización basada en enseñanza-aprendizaje (Teacher-Learner Based Optimization (TLBO)) es la utilizada para minimizar el ISE. Los resultados obtenidos con esta técnica los comparan con otros métodos conocidos y encuentran que la técnica propuesta funciona mucho mejor en comparación con otras, en términos de resultados cualitativos y cuantitativos.

Ya dentro del IoT y la industria, los autores de [45] consideran métodos de control tradicionales como los ya expuestos en los estudios anteriores mediante otro enfoque. En este trabajo, un sistema de control que utiliza IoT está diseñado para monitorear y controlar un sistema de tanque distribuido. Se implementa un prototipo utilizando principalmente un sensor ultrasónico, bomba de agua y placa Arduino. El sistema se supervisa y controla a través de internet mediante el uso de la plataforma *Blynk*, un conjunto de software de IoT totalmente integrado. Con el sistema IoT, el nivel de agua se puede monitorear de forma remota dentro del tanque en tiempo real, por otro lado el Arduino está programado para usar ese nivel de agua actual e implementar la acción del controlador bajo un esquema PID.

Por otro lado, existen diversos tipos de control fuera del controlador PID. Por ejemplo, los autores de [46] proponen un diseño de control de modo deslizante para un proceso de múltiple entrada múltiple salida (Multiple-Input Multiple-Output (MIMO)) mediante un esquema de compensación de retardo de tiempo para un sistema de tanque cuádruple. El control de modo deslizante es diseñado para la clase de sistema MIMO no lineal. La eficacia del algoritmo de control propuesto para el sistema compensado se comparó con un sistema convencional en presencia de retrasos en el proceso, incertidumbre adaptada y variables definidas por el usuario. Los resultados de la simulación demostraron que el sistema compensado elimina el efecto de los retrasos del proceso con precisión en comparación con el sistema retrasado. Además, el sistema compensado da una respuesta satisfactoria y proporciona una convergencia más rápida para diferentes valores de retardos del proceso y valores más altos de variables definidas por el usuario en comparación con el sistema retardado. Con la revisión de este estudio se encuentra que las soluciones para el control de sistemas multi-tanque dentro de la industria pueden proponerse desde diferentes enfoques.

3.2. Estado de arte de protocolos, arquitectura y virtualización para la Industria 4.0

A nivel de la Industria 4.0 existen diversos protocolos populares basados en *Ethernet* que pueden utilizarse para diferentes aplicaciones. En el artículo presentado en [47] se realiza un estudio comparativo entre algunos de estos protocolos, entre ellos OPC-UA. En éste se comparan las características propias de cada protocolo, el *overhead* de paquetes y el rendimiento. Con base en esto, se determina que OPC-UA tiene su fuerza en el modelado semántico de información y ofrece un alto rendimiento frente a otros protocolos. Este protocolo es ampliamente utilizado en la industria. En [48] se presenta una solución para el sistema de manufactura de la Industria 4.0 basado en OPC-UA y automatización ML. El objetivo de este estudio es demostrar la disponibilidad y aplicabilidad del enfoque en la solución propuesta mediante OPC-UA.

En lo que se refiere a arquitectura para los sistemas industriales, CIM es una tecnología idónea y usualmente muy utilizada como solución para los sistemas de manufactura, tal como es propuesto en [49]. Este artículo resume la evolución de las tecnologías de fabricación que están asociadas con los desarrollos hacia un sistema CIM y también se centra en los últimos desarrollos de investigación en CIM y proporciona una metodología de

justificación paso a paso hacia un sistema CIM para una pequeña o mediana empresa. Este estudio determina que el futuro exitoso de la industria manufacturera está involucrado en la utilización eficiente y eficaz de CIM y sus componentes.

En un estudio más reciente detallado en [50], se propone una arquitectura de referencia para CIM eficiente, ha identificado los diferentes aspectos relacionados con esta tecnología y detectado las limitaciones de los enfoques existentes. Proponen que el sistema de producción de la empresa se divida en dos niveles, como muestra la Figura 3.1:

- Nivel de planta. Consiste en la parte local del sistema de producción que será específico para cada planta. Contiene todos los sistemas y equipos físicos de fabricación, tales como: las máquinas y sus PLC, las impresoras y su servidor, las estaciones de trabajo y finalmente el MES. Este nivel representa así los cuatro primeros niveles de la pirámide CIM.
- Nivel corporativo. Es la parte central de los sistemas de producción que se compartirá entre las plantas de la empresa. Contiene solo el sistema ERP de la empresa.

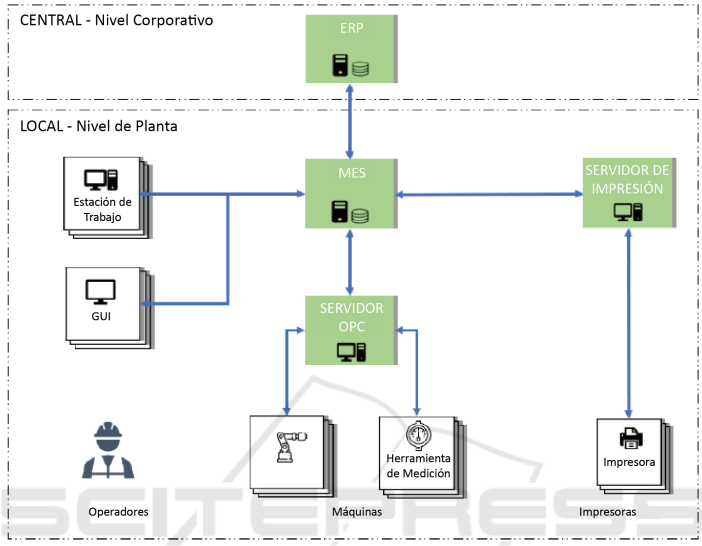


Figura 3.1: Arquitectura propuesta en [50]

Lo destacable de la propuesta de este estudio es que esta arquitectura toma en consideración seis aspectos principales: integración de datos, integración de sistemas, seguridad, monitoreo y análisis de datos, movilidad y finalmente computación en la nube.

El estudio literario realizado en [51] define a la Industria 4.0 como una filosofía de fabricación que incluye sistemas de automatización modernos con una cierta autonomía, intercambios de datos flexibles y efectivos que fomentan la implementación de tecnologías de producción de próxima generación, innovación en el diseño y una producción más ágil, así como productos personalizados. Esta definición indica claramente implementaciones de automatización (M2M, IoT) con capacidad de toma de decisiones de autonomía, intercambio efectivo de datos (ERP, nubes), que respalda la innovación e invención de tecnologías de generación futura, así como una utilización más personal de datos. Respecto al sistema de computación y almacenamiento en la nube, este estudio de revisión lo destaca como el servicio de almacenamiento en línea más simple que brinda conveniencia

operativa con aplicaciones basadas en web que no requieren ninguna instalación; adicionalmente reduce los costos, elimina la complejidad de la infraestructura, amplía el área de trabajo, protege los datos y brinda acceso a la información en cualquier momento. Por estas razones, la industria ha experimentado un cambio importante en el uso de soluciones en la nube y esto seguirá creciendo y representará un gran desafío para otros medios de almacenamiento de datos.

Actualmente, la transformación digital de la fabricación y manufactura se acelera gracias a dos factores importantes: los servicios en la nube y la virtualización de recursos. Esta es la idea en la que se enfoca el estudio realizado en [37]. Este artículo puntualiza que la virtualización de la capa MES significa que los recursos de *hardware* y *software* de la plataforma de computación en la nube se organizan y administran en un modelo de grupo, lo que permite múltiples beneficiarios; esto quiere decir que, la adopción de la computación en la nube considera la virtualización de cargas de trabajo MES. Por otro lado, recomienda que la realidad de la fabricación y manufactura debe reflejarse dentro de estos modelos y arquitecturas a través de gemelos digitales, cerrando la brecha entre el procesamiento de la información y los aspectos de la palabra real.

3.3. Estado del arte de integración de sistemas heterogéneos

Para la integración de sistemas heterogéneos son diversos los enfoques que pueden adoptarse. Por ejemplo, los autores de [52] proponen una plataforma de integración para sistemas de comunicación heterogéneos industriales. Este documento describe dos posibles conceptos de integración del estándar *Ethernet* Profinet IO de tiempo real utilizando Interbus y Profibus como ejemplo. Con este enfoque, es posible configurar y operar los sistemas de bus de campo sin más herramientas de ingeniería específicas. Aunque existen diferencias entre los dos sistemas de bus de campo, el método de modelado utilizado permite asignar todas las propiedades de ambos sistemas de bus de campo a una única red Profinet. Por tanto, Profinet es una plataforma de integración adecuada para sistemas de comunicación industriales heterogéneos.

Por otro lado, en [53] se acoplan sistemas de producción heterogéneos mediante un multi-agente basado en CPPS. Este artículo propone un enfoque que implementa el concepto de CPS en rápida evolución para el caso especial de los sistemas de producción por medio de agentes software. En su solución, para permitir la entrada de pedidos de clientes en el sistema, se desarrolló e implementó un prototipo de un cliente web que especifica requerimientos de recetas personalizadas y lotes solicitados, y es capaz de reenviar los pedidos especificados a los agentes del cliente implementados. Para evaluar la implementación del enfoque, se ingresaron manualmente diferentes órdenes en el sistema utilizando el cliente web proporcionado y se observó manualmente la reacción y el comportamiento de los diferentes sistemas de producción de laboratorio (simulados). Se verificó que el agente coordinador descompone correctamente los pedidos de los clientes y despacha los pedidos para las operaciones de producción a los diferentes CPPS donde se ejecutan.

El análisis de este estado del arte permite ampliar los conocimientos a aplicarse para el desarrollo y diseño propuesto con el fin de cumplir los objetivos de este trabajo de titulación. Con respecto al control del sistema multi-tanque, en este trabajo se opta por sistemas tradicionales como controladores PID y On-Off. Con respecto a protocolos y arquitecturas, OPC-UA y CIM constituyen la base de desarrollo de este trabajo. A la vez, gracias a su diversidad de funcionalidades, éstos facilitan la integración de la variedad de sistemas heterogéneos propuestos y la virtualización de capas superiores, parte fundamental para la constitución del IIoT.

Capítulo 4

Metodología

Este capítulo presenta de manera técnica la metodología empleada para el desarrollo e implementación de los sistemas de instrumentación, comunicación y control del sistema multi-tanque. Se presenta la arquitectura, las características de los dispositivos utilizados, el diagrama de instrumentación de la planta, el sistema de control aplicado, los protocolos de comunicación y el diseño del tablero de control.

4.1. Descripción del sistema

Se desarrolla una aplicación IIoT que controla el nivel de agua de un sistema conformado por dos tanques de reserva y un tanque de bombeo. El sistema presenta tres electroválvulas que regulan el paso del líquido entre los tanques. Se utilizan tres controladores, uno por cada válvula, el primero de tipo PID y dos de tipo ON-OFF. Además, para monitorizar el nivel de los tanques se emplea sensores ultrasónicos.

Se utiliza el PLC Controllino para obtener los datos de los sensores, acondicionar las señales recibidas y enviar la información al PLC Siemens utilizando el protocolo Modbus TCP/IP. A su vez, el PLC Siemens se encarga de realizar el sistema de control de la planta con los datos obtenidos. La información de todo el proceso se muestra en el HMI Siemens que obtiene los datos del PLC a través del protocolo Profinet. La unión entre la planta y las aplicaciones IIoT se realiza utilizando un *middleware* llamado OPC-UA. Este es un protocolo de comunicación máquina a máquina abierto y multiplataforma para la automatización industrial desarrollado por la Fundación OPC. Se basa en la arquitectura cliente-servidor y se enfoca en la recopilación y el control de datos entre equipos y sistemas industriales. Se implementa este servidor en el PLC Siemens.

El servidor de contenedores es la placa de desarrollo Up Squared Two, en su interior lleva el sistema operativo Ubuntu 18.04. Para implementar las aplicaciones IIoT se utiliza la herramienta Docker, que en comparación con los métodos de virtualización de máquinas o servidores posee la ventaja de ser más ligero y portátil, con una carga de recursos significativamente menor. Las aplicaciones IIoT implementadas son:

- ***Dashboard* en Node-red** Su función es proporcionar un panel de visualización de las variables asociadas

al sistema multi-tanque, además de controlar los actuadores y los *setpoints* de la planta.

- **Base de datos local** Esta aplicación recolecta los datos generados en la planta y los envía a una base de datos implementada en MySQL.
- **Envío de datos a la nube** Esta aplicación cumple con una de las corrientes IIoT que es el envío de la información a la nube dando la facilidad de recolectar la información de diversas plantas y sistemas evitando la barrera de encontrarse en una misma red o lugar geográfico.
- **Sistema de detección de fallas con aprendizaje de máquina** Al tener los datos del comportamiento de la planta se utiliza las herramientas de aprendizaje de máquinas para clasificar 4 diferentes fallos dentro del sistema, estos son: fuga en el tanque 1, fuga en el tanque 2, falla en la válvula de entrada y falla en la válvula de salida.
- ***Chatbot*** La aplicación crea un *bot* en la plataforma Telegram capaz de interactuar mediante mensajes y consultar información de la planta.

En la Figura 4.1 se observa un esquema general del sistema.

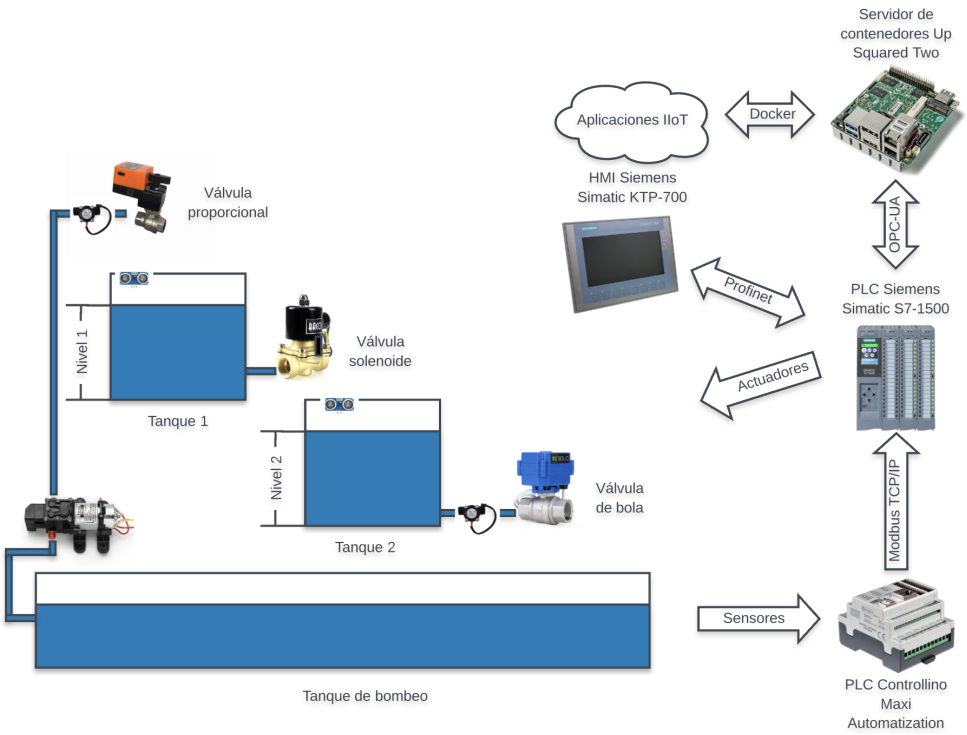


Figura 4.1: Esquema general del sistema realizado en el trabajo.

En la Tabla 4.1 se detallan los componentes que forman parte del sistema y sus características.

Tabla 4.1: Equipos que forman parte del sistema.

Equipo	Características
Bomba Favson F3012	El voltaje de alimentación es de 12 V, proporciona un caudal de 4 L/min, soporta una presión de hasta 100 psi, la entrada y la salida de la bomba se adapta a tuberías de 3/8”.
Electroválvula Winner WVA4-3	El voltaje de alimentación es de 24V y se controla con una señal de voltaje de 0 a 10V, es una válvula proporcional de tipo bola con tubería de 1/2”.
Electroválvula BACOENG 2W-15	El voltaje de alimentación es de 12V, es una electroválvula de tipo solenoide con tubería de 1/2”.
Electroválvula U.S. SOLID USS-MSV00002	El voltaje de alimentación es de 24V y el caudal que libera es proporcional al tiempo de alimentación en las bobinas de apertura o cierre, es una electroválvula de tipo bola con tubería de 3/4”.
Sensor ultrasónico HC-SR04	El voltaje de alimentación es de 5V y el rango de medida va de los 2 cm a 4 m con un margen de error de ±3 mm.
Sensor de flujo de agua DIGITEN	El voltaje de alimentación es de 5V y permite medir flujos entre 1 a 30 L/min con una presión menor a 253.816 psi y un margen de error de ±2 %, se adapta a tuberías de 3/4”.
PLC Controllino Maxi Automatization	El voltaje de alimentación es de 24V y tiene en su interior el microcontrolador Atmega2560, cuenta con 18 entradas digitales, 2 entradas analógicas, 8 salidas digitales, 2 salidas analógicas y un puerto Ethernet con compatibilidad con el protocolo Modbus TCP/IP.
PLC Siemens Simatic S7-1500	El voltaje de alimentación es de 24V, cuenta con 14 entradas digitales, 2 entradas analógicas, 14 salidas digitales, 2 salidas analógicas y una interfaz Profinet con dos puertos Ethernet.
HMI Siemens Simatic Basic Panel KTP 700	El voltaje de alimentación es de 24V, tiene una pantalla LCD a color de 7” y se puede manejar gracias a los botones integrados o su pantalla táctil. La comunicación se realiza mediante la interfaz Profinet.
Servidor de contenedores Up Squared Two	El voltaje de alimentación es de 5V, cuenta con el procesador Intel Pentium N4200 que tiene una arquitectura de 64 bits, una memoria RAM de 4GB y una memoria eMMC de 32GB.

4.2. Diagrama de instrumentación y procesos del sistema

El diagrama de instrumentación y procesos del sistema está diseñado según la norma ISA/ANSI 5.1 y se observa en la Figura 4.2. En el diagrama se muestra a la bomba acompañada de un elemento FV100, este es una válvula antiretorno que cumple la función de proteger a la bomba y tuberías de golpes de ariete y de aumentos de presión provocados por el cierre de la válvula FV108. Se observa además las tres electroválvulas FCV101, FCV103 y FCV106 que controlan el paso del líquido, estas se conectan al nodo controlador FC201. Este nodo recibe la información de nodos auxiliares LY200 y FY200 que son aquellos que preprocesan los datos de los sensores de nivel LT104 y LT105, y los sensores de caudal FT102 y FT107 respectivamente. Finalmente la información se envía del nodo FC201 a los nodos UG301 que es la interfaz hombre-máquina y UR300 que es un registro de retención de datos encargado de intercambiar la información con el nodo UG302 que simboliza las aplicaciones IIoT.

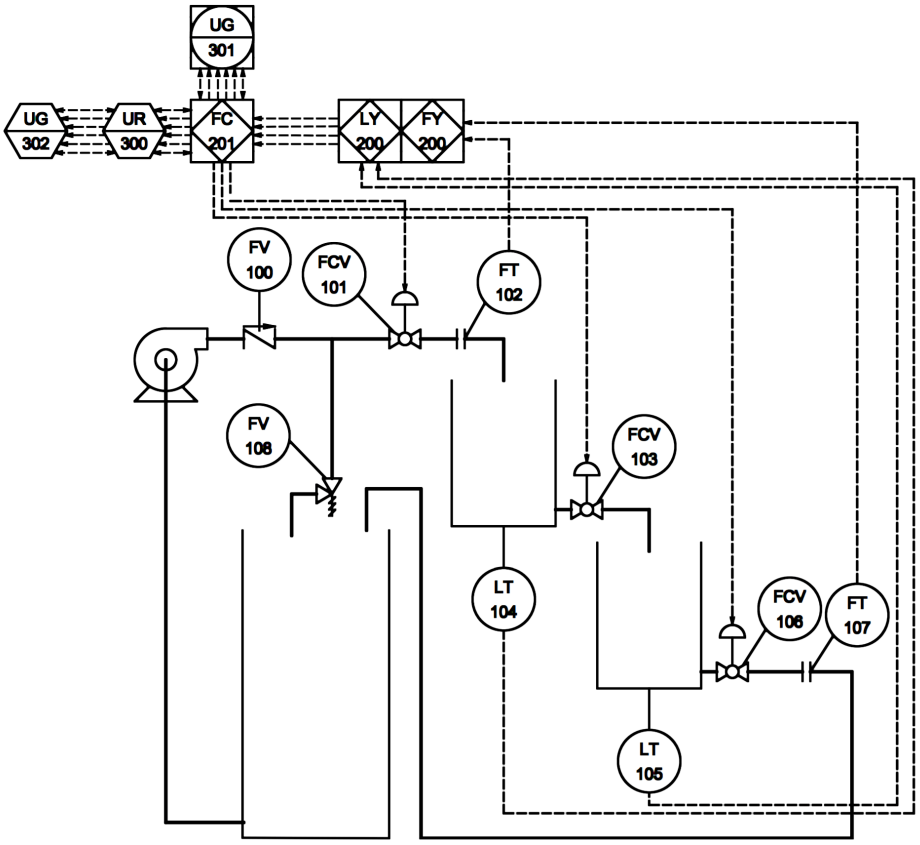


Figura 4.2: Diagrama de proceso e instrumentación del sistema.

4.3. Descripción de la arquitectura

El sistema utiliza la arquitectura CIM truncada de cuatro capas que se muestra en la Figura 4.3.

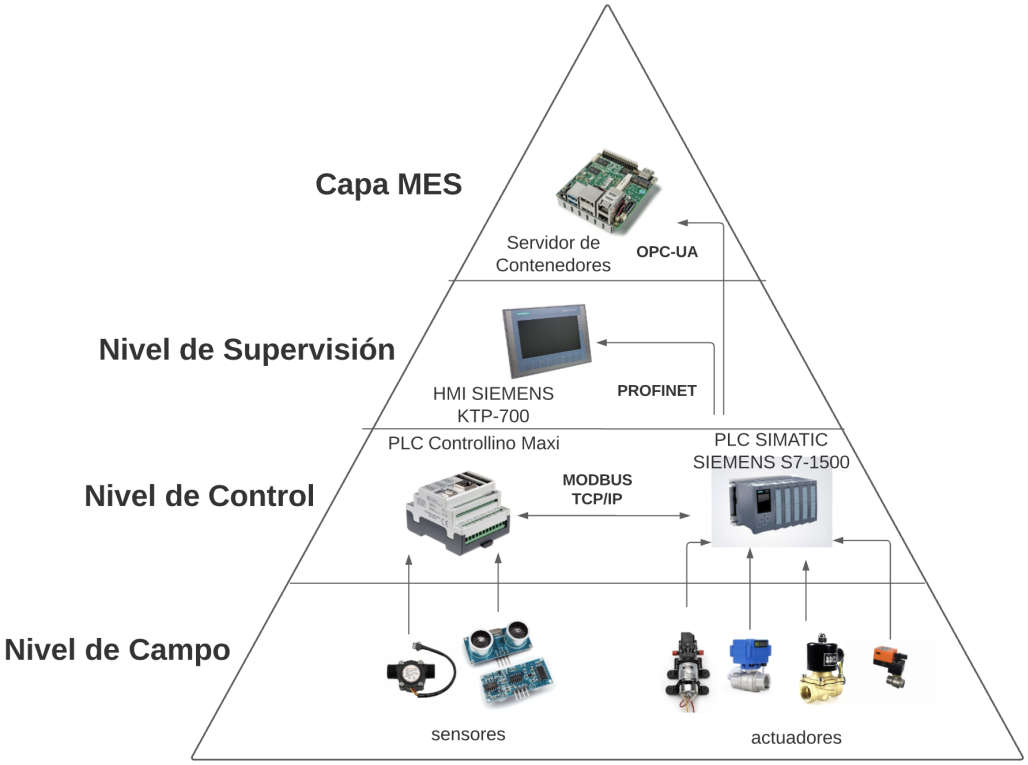


Figura 4.3: Arquitectura planteada del sistema.

4.3.1. Primera capa o nivel de campo

En el nivel de campo o primera capa, se ubican los sensores de caudal, los sensores de flujo, la bomba de agua y las tres diferentes electroválvulas. Esta capa se encarga de recolectar la información de los sensores y enviar a la capa superior, así como activar las electroválvulas para controlar el nivel de los tanques.

4.3.2. Segunda capa o nivel de control

La segunda capa se encarga de recibir los datos de los sensores con un tiempo de muestreo de 1 segundo y procesar esos datos con la finalidad de realizar el sistema de control de la planta. En esta capa se utilizan los PLC Controllino y Siemens. El PLC Controllino se encarga de recibir las señales de los sensores, acondicionarlas y enviarlas al PLC Siemens mediante el protocolo Modbus TCP/IP. El PLC Siemens se encarga de recibir la información y realizar el sistema de control de la planta

El PLC Controllino utiliza interrupciones para calcular la cantidad de pulsos por segundo que envía el sensor de flujo, luego se encuentra el flujo en L/min aplicando la Ecuación 4.1

$$flujo = \frac{Cantidad\_de\_pulsos}{Factor\_de\_calibracion} \tag{4.1}$$

Siendo $Factor\_de\_calibracion = 7.5$ ya que el fabricante indica que el sensor recibe 7.5 pulsos en cada L/min.

Se utiliza la librería *NewPing* de Arduino para recibir los datos de los sensores de nivel, esta librería envía

una onda y obtiene el tiempo que tarda la onda reflejada en llegar, la distancia se calcula con la Ecuación 4.2

$$distancia = \frac{tiempo\_medido \times velocidad\_del\_sonido}{2} \tag{4.2}$$

Donde *velocidad_del_sonido* = 340m/s.

El PLC Controllino redondea el valor obtenido de los sensores a dos decimales, luego multiplica esos valores por un factor de 1000 para así obtener valores enteros de las variables.

Como se mencionó, la comunicación entre los PLC se realiza mediante el protocolo Modbus TCP/IP. El Controllino actúa como maestro y el Siemens como esclavo. Para implementar el protocolo en el Controllino se utiliza la librería *mgsmodbus* que presenta compatibilidad con el módulo *Ethernet* que integra el PLC. La librería otorga transparencia en el uso de registros para el envío de información, únicamente se utiliza la dirección del primer registro (40001). Para la configuración del protocolo se requiere un puerto, que se utiliza por defecto el 502 y las direcciones IP del maestro y esclavo que se observa en el diagrama de red de la Figura 4.4. En el caso del PLC Siemens, se utiliza el bloque llamado *MB_CLIENT* en donde se requiere la dirección IP del maestro y la dirección del registro (40001).

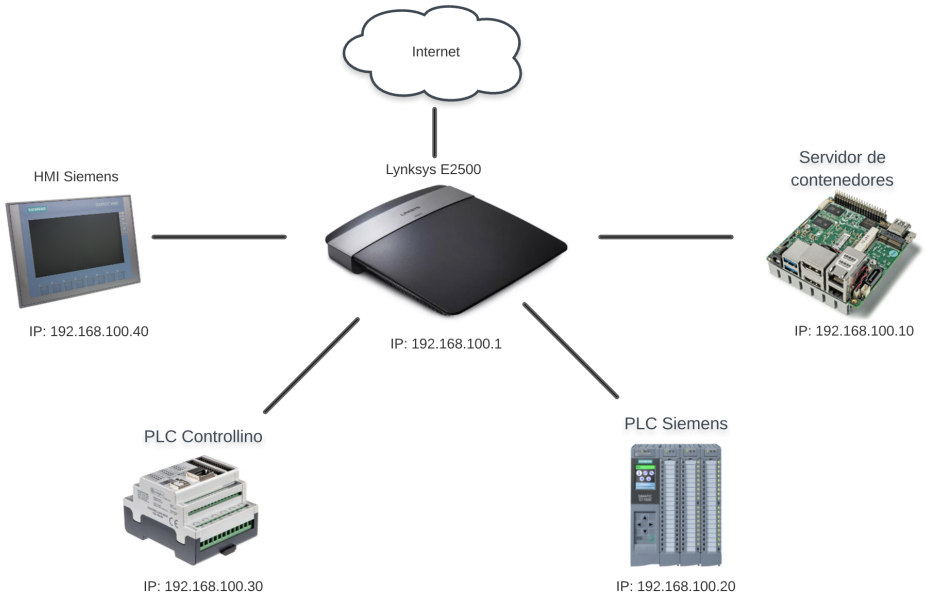


Figura 4.4: Diagrama de red del sistema.

Luego de obtener los datos de los sensores en el PLC Siemens, se utiliza el valor de los tanques para realizar el sistema de control. Los valores pasan por un proceso en el cual se normalizan y escalan para obtener valores de 0 al 100 %. El sistema de control de la planta se observa en la Figura 4.5

Se utilizan tres sistemas de control, el primero es de tipo PID y se encarga de comandar la válvula proporcional Winner WVA4-3. Es el responsable de regular el caudal de ingreso a la planta; toma como referencia el nivel objetivo del tanque 1 y su labor es minimizar el error e_1. El segundo, es de tipo ON-OFF y se encarga de comandar la válvula de tipo solenoide BACOENG 2W-15; por lo que abrirá y cerrará el flujo de agua del tanque

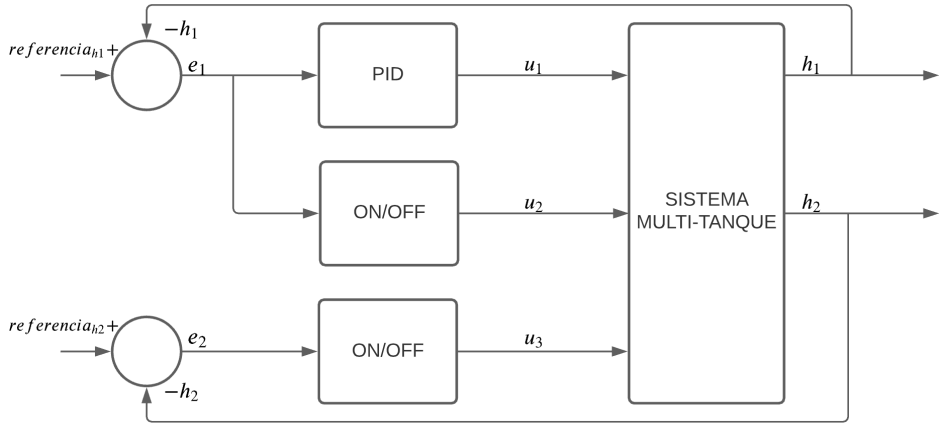


Figura 4.5: Sistema de control planteado.

1 al tanque 2. Su función es mantener el error e_1 siempre en signo positivo, esto significa impedir que el tanque 1 sobrepase el nivel objetivo. El tercero, es de tipo ON-OFF, se encarga de regular la válvula de tipo bola U.S. SOLID USS-MSV00002 y con ésta, el caudal de salida de la planta. Toma como referencia el nivel objetivo del tanque 2 y su función es minimizar el error e_2.

4.3.3. Tercera capa o nivel de supervisión

La tercera capa se encarga de monitorear las variables de proceso de la planta, para eso se utiliza el HMI Siemens, en donde se cuenta con indicadores para el nivel de la planta, nivel de apertura de la válvula proporcional, caudal de entrada y caudal de salida. Además, posee la función de ingresar manualmente los niveles objetivo de los tanques.

La comunicación entre el PLC y HMI Siemens se realiza a través de Profinet, respetando el objetivo de tener una solución heterogénea. Para implementar este protocolo, no es necesario utilizar librerías, porque los componentes comparten marca y tienen la facilidad que la comunicación es transparente al usuario.

4.3.4. Cuarta capa o MES

En la cuarta capa se tienen todas las funcionalidades IIoT virtualizadas, en el dispositivo Up Squared Two, con la ayuda de la herramienta Docker. Para comunicar los servicios de esta capa con la planta, se crea un servidor OPC-UA en el PLC Siemens. Donde, el PLC recibe como parámetros la dirección IP y el puerto, que de manera estándar se utiliza el 4840. Con esta información se crea un *endpoint*, necesario para que los clientes OPC-UA se comuniquen con el servidor.

En la Tabla 4.2 se observa las variables que contiene el servidor OPC-UA, las cuales son utilizadas por los servicios implementados en la capa MES.

Tabla 4.2: Variables implementadas en el servidor OPC-UA.

Variable de proceso	Variable en el servidor OPC-UA
Nivel actual del tanque 1	NivelTanque1
Nivel actual del tanque 2	NivelTanque2
Caudal de entrada	CaudalEntrada
Caudal de salida	CaudalSalida
Estado de la bomba	Bomba
Apertura de la válvula proporcional	ValvulaProporcional
Estado de la válvula solenoide	ValvulaSolenoide
Estado de la válvula de bola	ValvulaBola
Nivel objetivo del tanque 1	Setpoint1
Nivel objetivo del tanque 2	Setpoint2
Fuga en el tanque 1	FallaFugaTanque1
Fuga en el tanque 2	FallaFugaTanque2
Falla en la válvula de entrada	FallaValvulaEntrada
Falla en la válvula de salida	FallaValvulaSalida
Estado del paro de emergencia	Stop
Estado de funcionamiento de la planta	Manual

4.3.4.1. *Dashboard*

Se implementa un *dashboard* utilizando la plataforma Node-RED. Esta es una herramienta de programación para conectar dispositivos de *hardware*, API y servicios en línea de formas nuevas e interesantes. Cabe recalcar que Node-RED se instala en un contenedor Docker con la finalidad de ejecutarse en forma paralela y aislada al sistema de control y supervisión. Para la programación en esta herramienta de *software* libre se requiere la instalación de un nodo para comunicarse a través de OPC-UA basado en la librería node-red-contrib-opcua y un nodo indicador de estado LED simple basado en la librería node-red-contrib-ui-led.

Este panel de visualización se diseña en cuatro secciones separadas. Éstas se detallan a continuación:

1. Corresponde al control automático, ésta presenta los niveles de los dos tanques frente a cada nivel objetivo mediante *widgets* de nivel y tipo dona respectivamente, los valores de caudal de entrada, caudal de salida y apertura de la válvula proporcional mediante *widgets* tipo brújula, indicadores de estado de la bomba, válvula solenoide y válvula de bola mediante *widgets* tipo LED y la gráfica de señales de control mediante el *widget* tipo *chart*.

2. Corresponde al control manual, ésta presenta los campos de texto en los que se pueden definir los niveles objetivo de los dos tanques mediante los *widgets* de cuadro de texto, la activación de la opción de control manual mediante un *widget* de cambio de estado, los controles para la apertura de la válvula proporcional mediante un *widget* tipo *slider*, los controles de activación de la bomba, válvula solenoide y válvula de bola mediante *widgets* de cambio de estado y la gráfica del nivel del tanque 1 mediante el *widget* tipo *chart*. Se accede a estos controles siempre y cuando esté activado el control manual, caso contrario permanecen desactivados y no se los puede utilizar.

3. Corresponde únicamente al paro de emergencia o estado *STOP* mediante un *widget* de cambio de estado.

4. Corresponde a indicadores de fallas del sistema mediante *widgets* tipo LED y la gráfica del nivel del

tanque 2 mediante el *widget* tipo *chart*.

Más adelante, en la Sección 5.2 se presenta la visualización del diseño de este *dashboard*.

4.3.4.2. Base de datos local

Para el almacenamiento de datos de manera local se utiliza herramientas de *software* libre como el sistema de gestión de bases de datos MySQL en conexión con el lenguaje de programación Python. MySQL se instala en un contenedor Docker con la finalidad que se ejecute de forma paralela y aislada al sistema de control, supervisión y al panel de visualización. En él se crea una base de datos de nombre "TESIS" y una tabla que almacenará las variables definidas en la Tabla 4.2.

El envío de datos se ejecuta dentro de otro contenedor y es construido desde un archivo *dockerfile*, este es un documento de texto que contiene las instrucciones para crear un contenedor a partir de una imagen base. La Figura 4.6 muestra el diagrama de flujo correspondiente a la creación del contenedor. Se utiliza Python3 como imagen base y se instalan paquetes adicionales mediante el comando pip3, estos son:

- *mysql-connector-python* para la conexión entre la herramienta de programación y el gestor de base de datos,
- *cryptography* para el correcto funcionamiento del cliente OPC-UA,
- *python-time* para la suspensión momentánea de ejecución del código mediante la función *sleep()*,
- *opcua* para la conexión con el servidor OPC-UA,
- *ntplib* para la consulta de hora y fecha mediante un servidor de protocolo de tiempo de red (Network Time Protocol (NTP)).

Adicionalmente, se implementa un programa en Python que se encarga de recolectar las variables de la planta mediante un cliente OPC-UA, consultar la hora y la fecha mediante un cliente NTP y finalmente enviar la información hacia la base de datos creada utilizando sentencias SQL. La Figura 4.7 muestra el diagrama de flujo correspondiente a la estructura del programa.

4.3.4.3. Envío de datos a la nube

Para el envío de datos a la nube se utiliza Thinger.io, esta es una plataforma de *software* libre para aplicaciones IoT que proporciona una infraestructura de nube escalable lista para conectar diversos dispositivos o sistemas. Es necesario la creación de un *data bucket*, que actúa como un espacio de memoria virtual dentro de la plataforma, con esto se consigue guardar el comportamiento de la planta dentro de la nube.

La información se envía desde el *dashboard* creado en la Sección 4.3.4.1, mediante el protocolo de transferencia de hipertexto (Hypertext Transfer Protocol (HTTP)) desde un nodo tipo *http request*. Para esto, en Thinger.io se define un *token* de acceso al cual se le asigna el permiso de escritura en el *databucket*. El nodo HTTP de Node-RED se configura con los siguientes datos de conexión hacia Thinger.io:

- método *POST*,
- dirección de localizador uniforme de recursos (Uniform Resource Locator (URL)) correspondiente al *databucket* en Thinger.io,
- autenticación tipo portador (*bearer*) y asignación de *token* de acesso definido en Thinger.io,
- habilitación de conexión *keep-alive*, y,

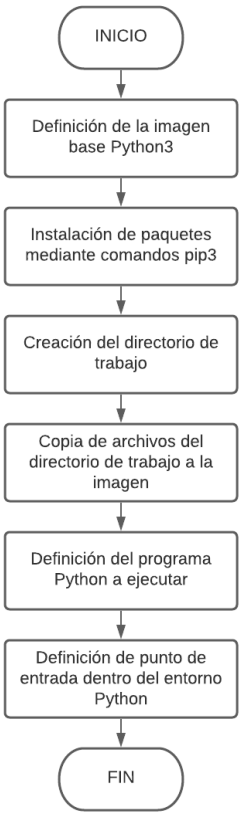


Figura 4.6: Diagrama de flujo *dockerfile*.

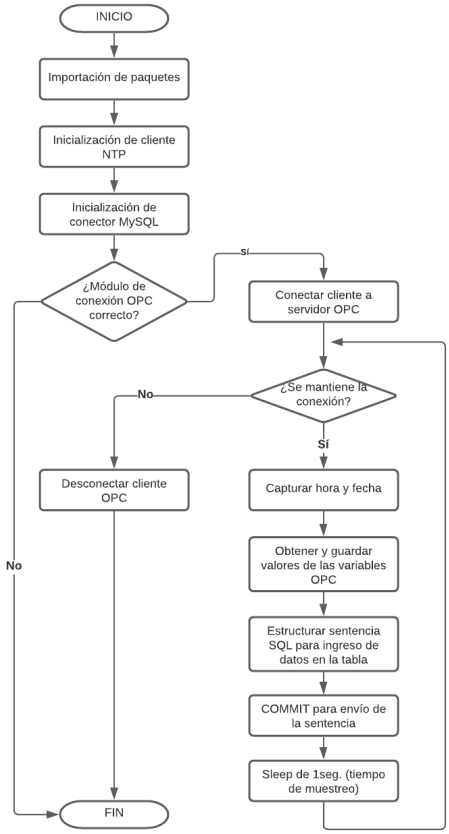


Figura 4.7: Diagrama de flujo código Python.

- asignación de cadena UTF-8 como *return* del método asignado.

Con la finalidad de visualizar los datos, se diseña un *dashboard* en la plataforma Thinger.io. A este *dashboard* se añaden los *widgets* necesarios, a los que se les asigna las variables que recibe el *databucket* desde Node-RED.

4.3.4.4. Sistema de detección de fallas con *Machine Learning*

El sistema de detección de fallas impone un punto de equilibrio, debido a que caracterizar el comportamiento de la planta en toda su dinámica resulta computacionalmente muy costoso y en la práctica los sistemas multi-tanques son utilizados como tanques de reserva y siempre mantienen un nivel definido. El punto de equilibrio que se establece es del 25 % en el tanque 1 y 35 % en el tanque 2. Se toma una base de datos de 900 muestras, esto es, el comportamiento de la planta durante 15 minutos. La misma esta conformada por las 10 primeras variables del servidor OPC-UA que se observa en la Tabla 4.2 y 4 variables adicionales correspondientes a la tasa de cambio del nivel del tanque 1, tanque 2, caudal de entrada y caudal de salida.

A la base de datos obtenida se aplica un proceso estadístico llamado análisis de componente principal PCA con la finalidad de encontrar un nuevo conjunto de datos de menor dimensión para caracterizar el comportamiento de la planta de manera gráfica. Esta nueva base de datos se utiliza para entrenar un algoritmo de *machine learning* llamado máquina de vectores de soporte SVM cuya función es clasificar el comportamiento de la planta. El algoritmo detecta 5 comportamientos que son: comportamiento normal, fuga en el tanque 1, fuga en el tanque 2, falla en la válvula de entrada y falla en la válvula de salida.

Para simular una falla en el tanque 1 o en el tanque 2 se libera un tapón que se encuentra en la base del tanque que tiene una apertura de 3/8". Para simular una falla en la válvula de entrada o en la válvula de salida se desconecta esa válvula del tablero de control. En la Figura 4.8 se observa un diagrama de flujo del entrenamiento del clasificador y en la Figura 4.9 se observa el diagrama de flujo del proceso de evaluación del comportamiento de la planta.

4.3.4.5. *Chatbot*

El servicio *chatbot* utiliza la plataforma Telegram para realizar consultas a la planta y a la base de datos local. Para crear el *bot* se utiliza BotFather, que es un *bot* nativo de Telegram. Éste, únicamente con información de un nombre y un usuario, crea un *token* que sirve para comunicar el lenguaje de programación Python a la plataforma Telegram. La comunicación entre Python y Telegram se desarrolla por una API. En Python se utiliza el paquete *telegram.ext*. Cabe recalcar que este *bot* de consulta por *chat* se instala en un contenedor Docker con la finalidad de ejecutarse en forma paralela y aislada al sistema de control y supervisión. Las funcionalidades que el servicio *chatbot* implementa son:

- Conocer el funcionamiento actual de la planta (manual o automático).
- Conocer el nivel de los tanques.
- Conocer el nivel de los *setpoints*.
- Conocer el nivel de los caudales.
- Conocer el estado de las válvulas (abiertas o cerradas).
- Consultar si existe fallas en la planta

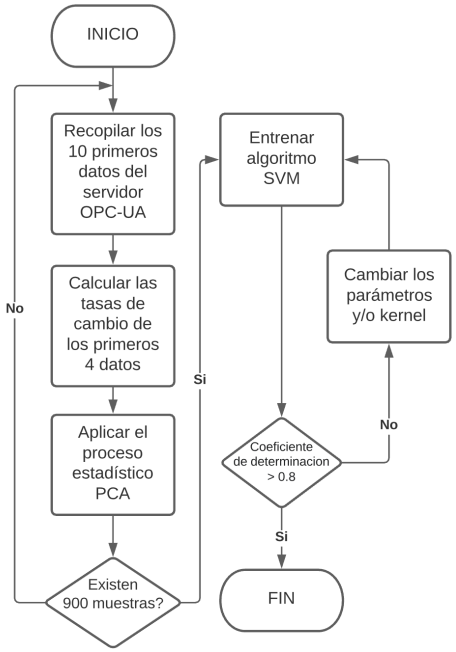


Figura 4.8: Diagrama de flujo del entrenamiento del clasificador.

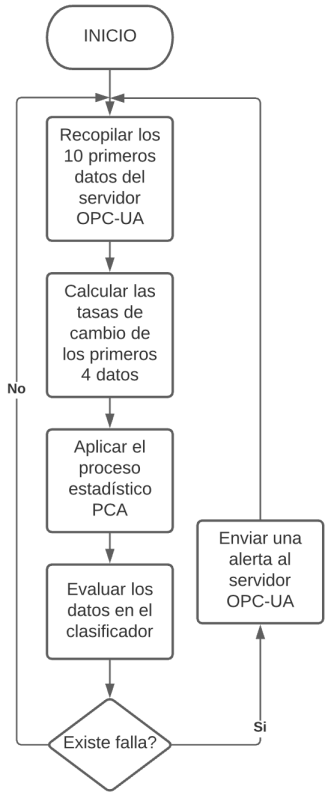


Figura 4.9: Diagrama de flujo de la evaluación del comportamiento de la planta.

- Consultar el registro de fallas en la base de datos

En la Figura 4.10 se observa el diagrama de flujo para obtener la consulta de Telegram y enviar la respuesta utilizando el lenguaje de programación Python.

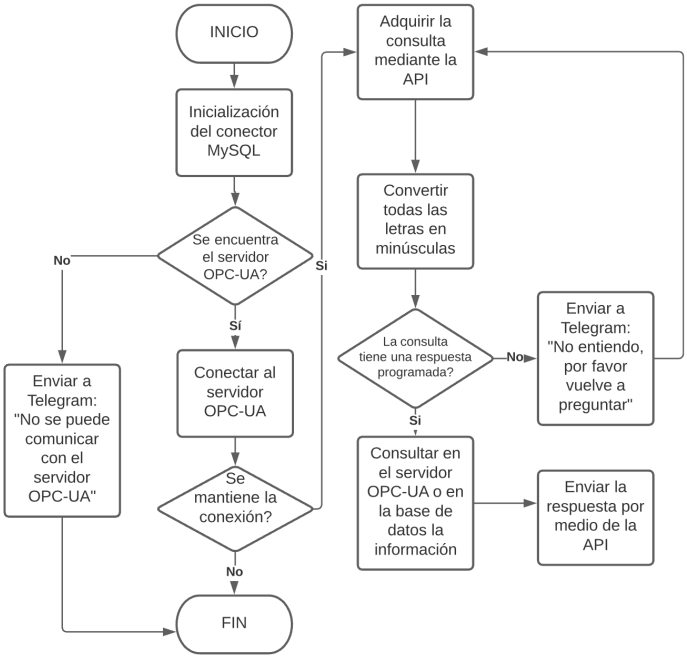


Figura 4.10: Diagrama de flujo del Chatbot.

4.4. Diseño del *hardware*

En la Figura 4.11 se observa el esquema del sistema de tanques. El diseño se reutilizó de la tesis presentada en [54], en el anexo A se muestra en detalle el sistema.

El sistema cuenta con 3 tanques que se observan de color gris, un tanque de bombeo ubicado en la parte inferior del sistema, con capacidad para 19.162 L y tanques de reserva 1 y 2, ambos con capacidad para 12 L, a una altura de 61.55 cm y 19.9 cm respectivamente. Además los tanques de reserva cuentan con una abertura de 0.95 cm en la parte inferior para simular fugas. La bomba que alimenta al sistema, que se presenta en color marrón, está ubicada en la parte trasera del segundo tanque de reserva a una altura de 19.9 cm y está anclada al armazón con ayuda de 4 tuercas y tornillos de 1/8”. Así mismo, la válvula proporcional está a una altura de 43 cm de la base del tanque de reserva 1, la válvula solenoide está a una altura de 41 cm de la base del tanque de reserva 2 y la válvula de tipo bola está a una altura de 19.9 cm del tanque de bombeo. Todas estas se ilustran en color azul. Por último, los sensores ultrasónicos están a una altura de 33 cm de la base de los tanques de reserva y los sensores de caudal se posicionan a continuación de la válvula proporcional y la válvula de bola. Todos los sensores se visualizan en color morado.

En la Figura 4.12 se observa el diseño del tablero de instrumentación. En el anexo B se muestra a detalle.

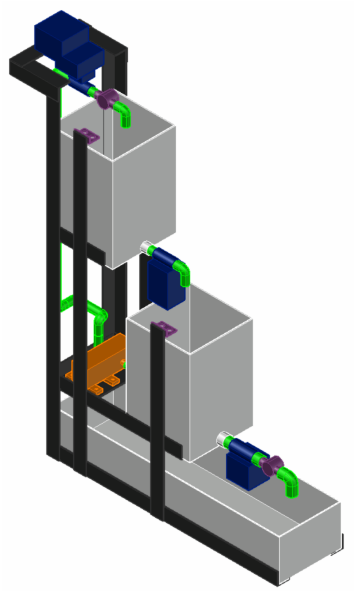


Figura 4.11: Diseño del sistema de tanques.

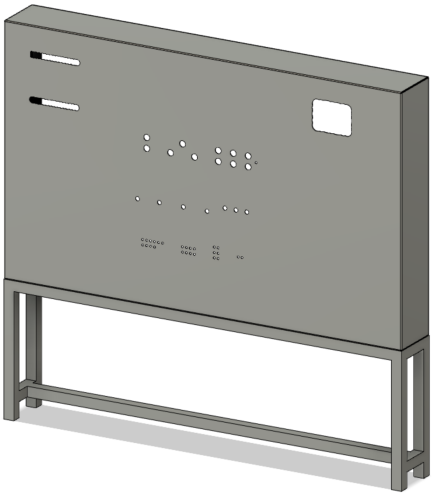


Figura 4.12: Diseño del tablero de instrumentación.

El tablero es un gabinete de acero con 180 cm de ancho, 100 cm de alto y 15 cm de espesor, contiene en la parte interior canaletas para el manejo de cables y 2 riel din para la colocación de los componentes. En la parte superior izquierda posee dos aperturas las cuales sirven para el manejo de cables del PLC Siemens y el enrutador Lynksys. En la parte superior derecha se observa un agujero para el HMI. En la parte central existe tres niveles de agujeros, los agujeros superiores sirven para alojar 6 pulsantes, un selector de dos posiciones, un botón de paro de emergencia y dos luces piloto, esto destinado para el control manual de la planta. En el segundo nivel se encuentran 7 luces piloto, destinadas a indicar las fallas en el sistema así como el estado de la bomba y las electroválvulas. Finalmente en el tercer nivel se encuentran 26 borneras para la conexión de las válvulas, sensores y alimentación del tablero.

Capítulo 5

Resultados

Este capítulo presenta los resultados del funcionamiento integrado del sistema multi-tanque. Adicionalmente, se presenta un análisis de los sistemas de interacción con la planta, el envío de información a la base de datos local y a la nube de almacenamiento, el servicio de detección de fallas y una comparación de la solución propuesta en este trabajo con una solución IIoT del fabricante Siemens.

5.1. Funcionamiento del sistema multi-tanque

La Figura 5.1 presenta el sistema multi-tanque construido. Éste cuenta con cable helicoidal para proteger los cables de los componentes además de dar mejor estética a la planta. La Figura 5.2 presenta el tablero de instrumentación construido. Éste es manejado por personal operador de la planta y cuenta con la función de control automático y manual. La función de control manual permite activar o desactivar las válvulas y bomba mediante botones y manejar la apertura de la válvula proporcional utilizando un potenciómetro. La Figura 5.3 presenta la parte interna del tablero de instrumentación. Se aprecia los dispositivos utilizados, fuentes de alimentación, PLC Controllino, módulo de relés y servidor de contenedores, todos estos acoplados mediante *riel din*.

La Figura 5.4 presenta el comportamiento de la planta frente a un nivel objetivo en el tanque 1 del 35 % y en el tanque 2 del 45 %. Se observa un rizado en las mediciones, consecuencia de las ondulaciones y salpicaduras que se crean al ser tanques pequeños. Para mejorar la toma de datos se incorpora flotadores en los dos tanques. Estos están construidos en madera y tienen un tamaño de 20 cm x 20 cm, además presentan un agujero recubierto con una malla con el propósito de amortiguar las salpicaduras. En la Figura 5.5 se muestra el comportamiento de la planta al adicionar los flotadores, donde se aprecia la reducción del rizado en las mediciones, mejorando así la toma de datos y por ende el rendimiento del sistema.

5.1.1. Porcentaje de sobreimpulso y tiempo de estabilización del sistema

Para determinar el porcentaje de sobreimpulso y el tiempo de estabilización del sistema, se realizaron 3 pruebas, en todas ellas se estableció un nivel objetivo en el tanque 1 del 35 % y en el tanque 2 del 45 %. En la

Figura 5.1: Sistema multi-tanque ensamblado.

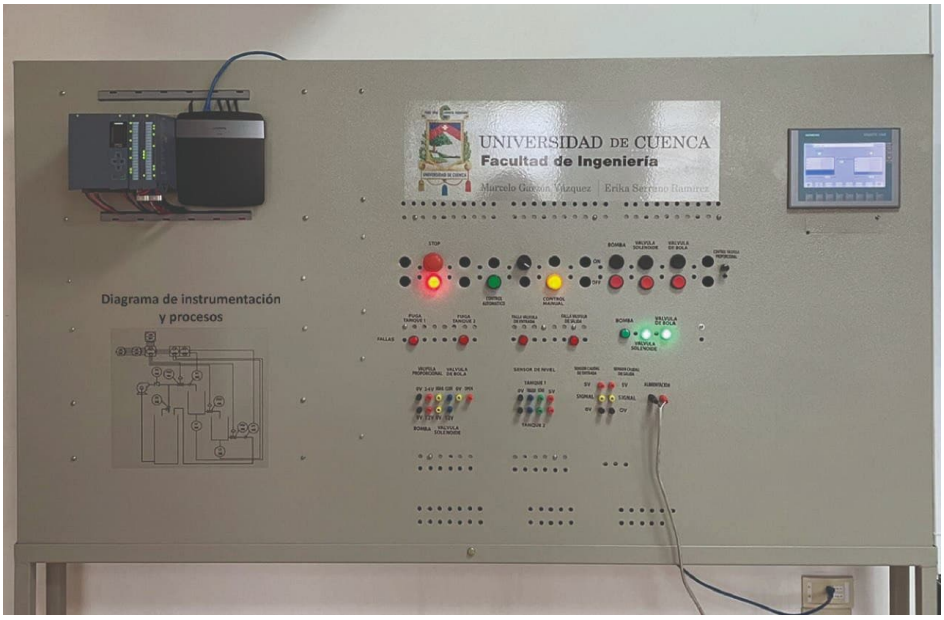


Figura 5.2: Tablero de instrumentación ensamblado.

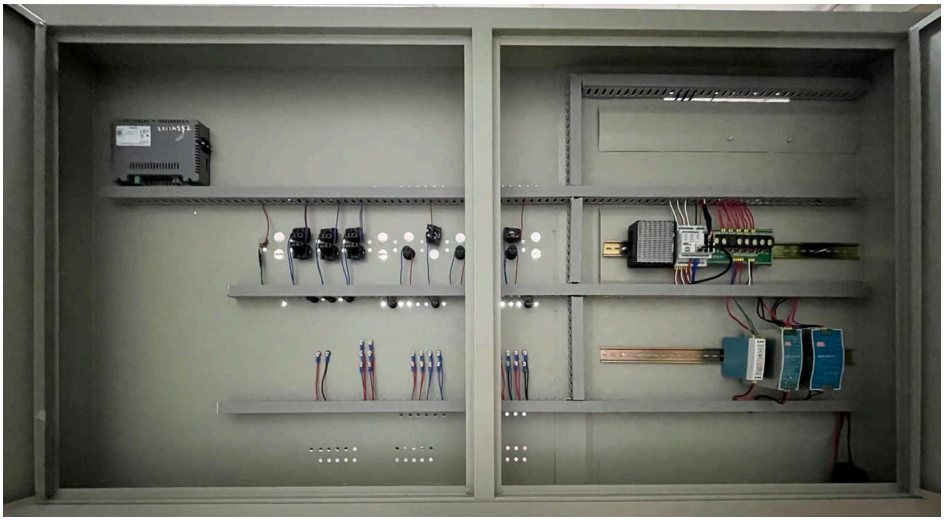


Figura 5.3: Tablero de instrumentación ensamblado, parte interna.

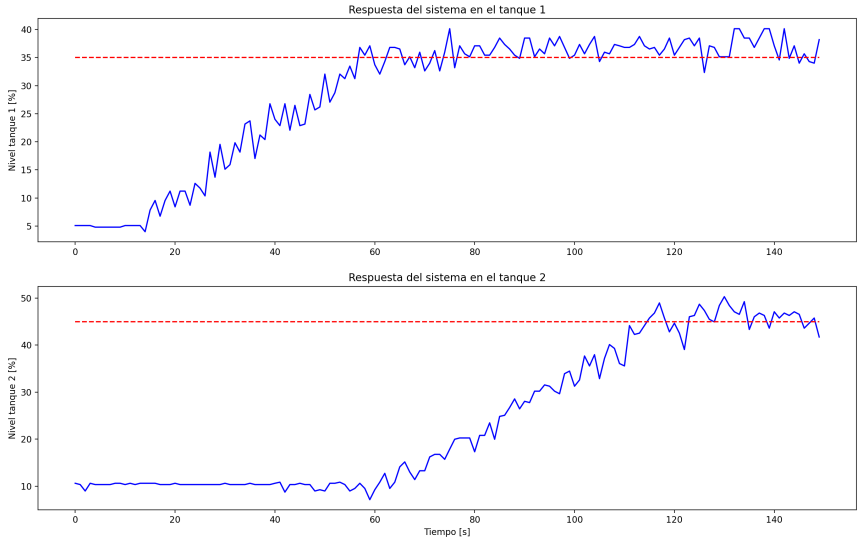


Figura 5.4: Comportamiento transitorio del sistema multi-tanque.

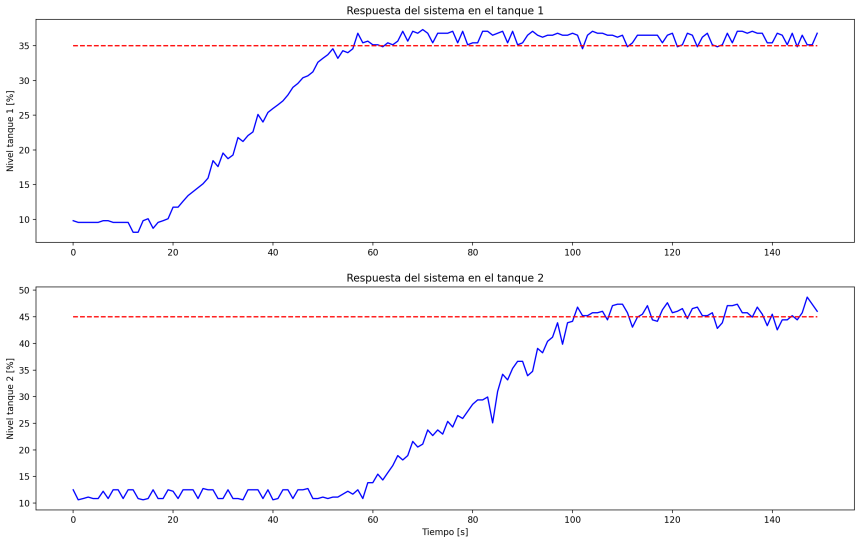


Figura 5.5: Comportamiento transitorio del sistema multi-tanque utilizando flotadores.

Tabla 5.1: Resultados de las pruebas de sobreimpulso y tiempo de estabilización.

	Prueba 1	Prueba 2	Prueba 3	Promedio
Porcentaje de sobreimpulso en el tanque 1	8.31 %	8.31 %	6.69 %	7.77 %
Porcentaje de sobreimpulso en el tanque 2	6.88 %	8.23 %	8.84 %	7,65 %
Tiempo de estabilización en el tanque 1	52 s	54 s	52 s	52.66 s
Tiempo de estabilización en el tanque 2	112 s	115 s	109 s	112 s

Tabla 5.1 se aprecia los resultados obtenidos en las pruebas. El tanque 1 presenta en promedio un porcentaje de sobreimpulso del 7.77 % bastante similar al 7.67 % del tanque 2. Además, el tiempo de estabilización del tanque 1 en promedio es de 52.66 segundos, mucho menor a los 112 segundos del tanque 2. Estos resultados muestran que la planta tiene un comportamiento definido ya que no existe gran diferencia entre los tres experimentos. La Figura 5.6 presenta una gráfica del comportamiento promedio que tiene el sistema en las pruebas realizadas.

5.1.2. Cambios de referencia en el sistema

Para determinar el comportamiento de la planta ante cambios de referencia, se realizan 3 pruebas. La planta parte con un nivel objetivo en el tanque 1 del 35 % y en el tanque 2 del 45 %. A los 150 segundos, se cambia del 35 % al 25 % el nivel objetivo del tanque 1 y del 45 % al 35 % el nivel objetivo del tanque 2. Finalmente, a los 250 segundos se retorna a los niveles objetivos originales. La Tabla 5.2 muestra los resultados obtenidos. Donde los tiempos de estabilización al disminuir el nivel objetivo son claramente superiores que al aumentarlos, esto se debe a la falta de comunicación entre los controladores de las válvulas. La Figura 5.7 presenta el comportamiento promedio del sistema en las 3 pruebas realizadas.

Como datos adicionales obtenidos en las pruebas, se aprecia que la planta tiene un tiempo muerto de 15 s en el tanque 1 y de 53 s en el tanque 2. Esto se debe a que la válvula proporcional es lenta y necesita abrirse

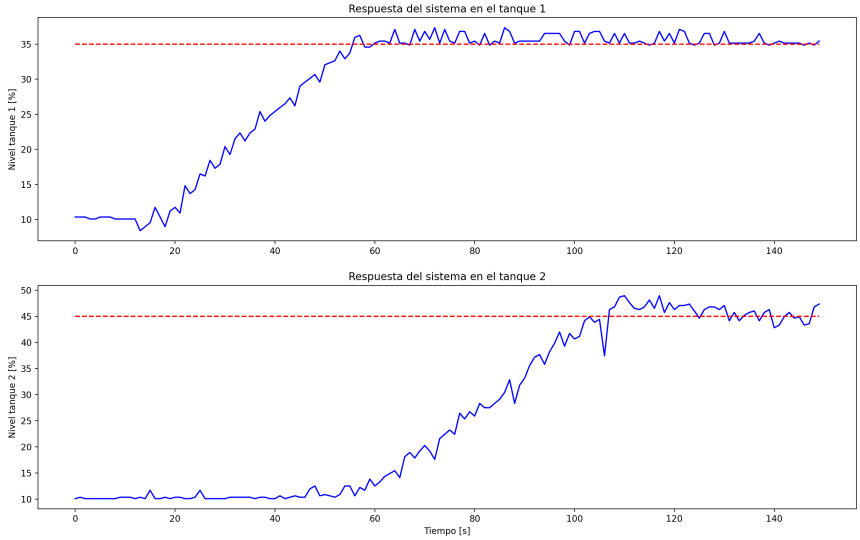


Figura 5.6: Comportamiento promedio del sistema en las pruebas de sobreimpulso y tiempo de estabilización.

Tabla 5.2: Resultados de las pruebas de cambio de referencia

	Prueba 1	Prueba 2	Prueba 3	Promedio
Tiempo de estabilización del tanque 1 del 35 % al 25 %	70 s	68 s	67 s	68.33 s
Tiempo de estabilización del tanque 1 del 25 % al 35 %	17 s	19 s	19 s	18.33 s
Tiempo de estabilización del tanque 2 del 45 % al 35 s	102 s	110 s	107 s	106.33 s
Tiempo de estabilización del tanque 2 del 35 % al 45 s	37 s	34 s	37 s	36 s

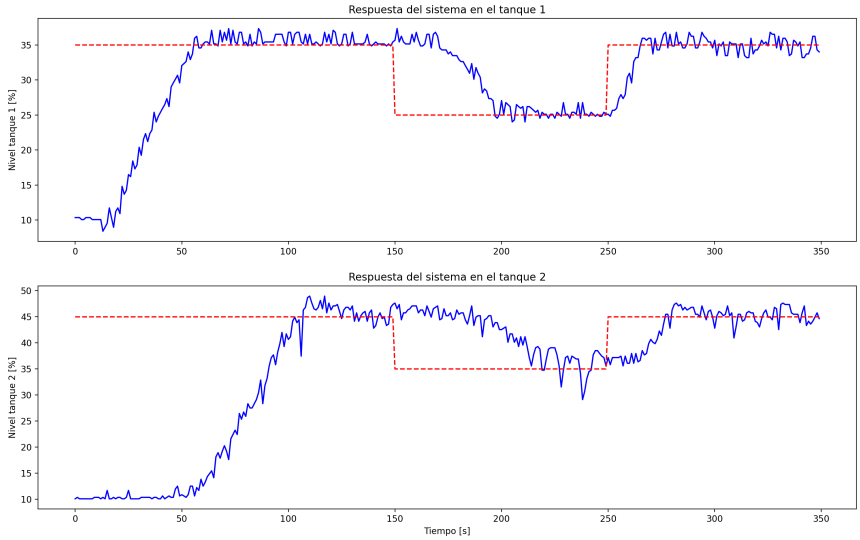


Figura 5.7: Comportamiento promedio del sistema en las pruebas de cambio de referencia.

mínimamente hasta el 13 % para que la bomba se active. Además, el tanque 2 presenta el tiempo muerto ya que el sistema de control se diseñó de tal manera que el tanque 1 alcanza el nivel objetivo y a continuación el agua fluye al tanque 2.

5.2. Análisis de los sistemas de interacción con la planta

Los sistemas de interacción con la planta implementados son cuatro y se listan a continuación. Cada uno tiene su propia funcionalidad, esto se profundiza más adelante.

- Sistema de supervisión y control en planta: tablero de instrumentación y HMI.
- Sistema de monitoreo y control administrativo: *dashboard* de Node-RED.
- Sistema de monitoreo externo a la planta: *dashboard* de la nube Thinger.io.
- Sistema de consulta por *chat*: *chatbot* de Telegram.

En las subsecciones siguientes se presentan figuras en las que se aprecia el diseño final de estos sistemas de interacción. Para el servicio de supervisión o monitoreo, lo principal es conocer el nivel de los tanques frente al nivel objetivo al que se ha fijado cada uno, esto para verificar el funcionamiento de los sistemas de control implementados. Para el servicio de control, lo indispensable es permitir el reajuste de los niveles objetivos de los tanques, esto para verificar que los sistemas de control responden de manera adecuada ante un cambio de referencia.

Los paneles de visualización de la información sobre el funcionamiento del sistema son tres, el HMI, el *dashboard* de Node-RED y el *dashboard* de la nube Thinger.io. Éstos tienen la finalidad de supervisión, en el caso del HMI, de virtualización del servicio de monitoreo y control, para el caso del *dashboard* de Node-RED, y la visuaización de información en la nube de información, para el caso del *dashboard* de Thinger.io.

5.2.1. Sistema de supervisión y control en planta

El sistema de supervisión y control en planta se compone del tablero de instrumentación y el HMI. La Figura 5.8 presenta el diseño final del sistema de supervisión del HMI. Éste cuenta con los indicadores de nivel de los tanques, de apertura de la válvula proporcional, de caudal de entrada, de caudal de salida y la función de ingreso manual de los niveles objetivo de los tanques. El sistema de supervisión del HMI, destinado para el uso de personal operador de la planta, se encuentra incluido en el tablero de instrumentación que se ubica junto al sistema multi-tanque.

La Figura 5.2 presentada en la Sección 5.1 permite apreciar el diseño final del tablero de instrumentación. Éste incluye el sistema de supervisión del HMI, y además, como se explicó en mencionada sección, permite el control ya sea manual o automático de la planta. Al final, HMI y tablero de instrumentación constituyen un solo sistema de interacción total con la planta.

5.2.2. Sistema de monitoreo y control administrativo

El sistema de monitoreo y control administrativo es el *dashboard* implementado en la plataforma Node-RED. El diseño final del *dashboard* se ha dividido en dos partes, la parte superior se presenta en la Figura 5.9 y la parte

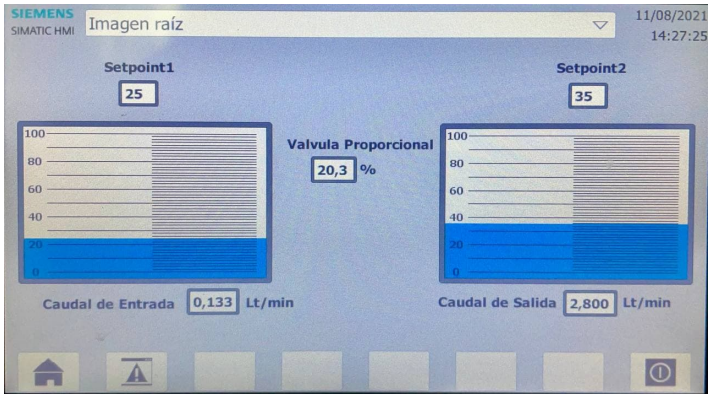


Figura 5.8: Diseño del HMI.

inferior en 5.10. El panel de visualización cuenta con las secciones descritas en 4.3.4.1, de manera que se tenga por separado el control automático y el control manual, además de los indicadores de fallas, funcionamiento de válvulas y gráficas de comportamiento de los sistemas de control. El panel de visualización diseñado en Node-RED, destinado para el uso de personal administrativo de la planta, se encuentra en oficinas de la empresa.

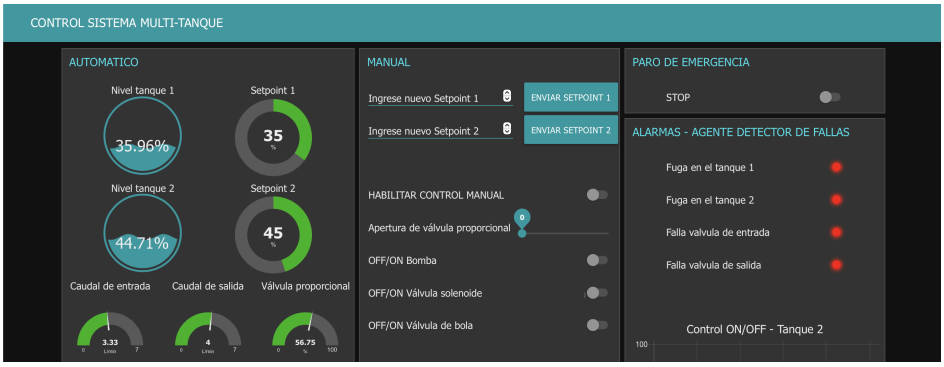


Figura 5.9: Diseño del *dashboard* en Node-RED, parte superior.

Figura 5.10: Diseño del *dashboard* en Node-RED, parte inferior.

5.2.3. Sistema de monitoreo externo a la planta

El sistema de monitoreo externo a la planta es el *dashboard* implementado en la nube de almacenamiento de información Thinger.io. La Figura 5.11 presenta el diseño del panel de visualización. Éste se concibe como un sistema de monitoreo desde la nube debido a que no existe la opción de controlar la planta desde esta plataforma, a diferencia del *dashboard* de Node-RED. El *dashboard* de Thinger.io, puede ser consultado por cualquier miembro o personal de la empresa que se encuentre fuera de ella.

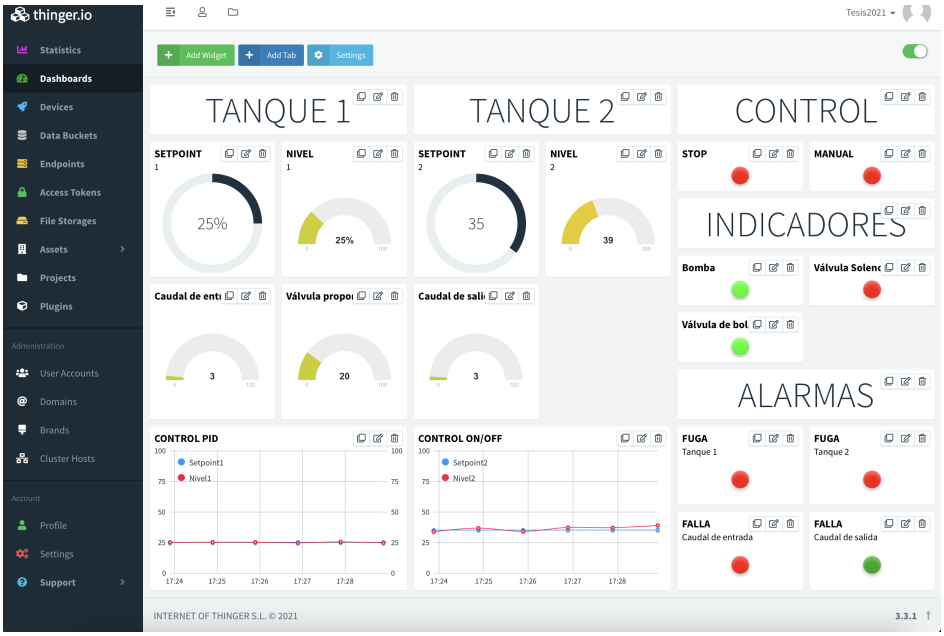


Figura 5.11: Diseño del *dashboard* en la nube de almacenamiento de información de Thinger.io.

5.2.4. Sistema de consulta externo por *chat*

El sistema de consulta por *chat* consiste en el *chatbot* implementado en Telegram. La Figura 5.12 presenta una captura de la conversación en la cual se hace la consulta de varios parámetros y valores que se desean conocer del funcionamiento del sistema. El *chatbot*, destinado también para monitoreo externo, puede ser consultado por cualquier miembro o personal de la empresa, con la finalidad de conocer el funcionamiento del sistema en el instante de la consulta.

5.2.5. Análisis de longitudes de paquetes enviados por la planta

La planta envía paquetes de información hacia todos los sistemas que funcionan en torno a ella, para analizarlos se utiliza la herramienta Wireshark. Éste es un *software* gratuito que permite analizar el tráfico de red en tiempo real. Mediante el uso de esta herramienta y la explotación de sus capacidades de análisis de red, se filtran los paquetes de acuerdo a los protocolos que se desean analizar.

El propósito de esta prueba es determinar el tamaño de los paquetes manejados por las soluciones IIoT, éstos son Node-RED, la nube Thinger.io, el *chatbot* de consulta y la base de datos local MySQL. Al estar todos estos servicios virtualizados en sus propios contenedores Docker, a excepción de la nube que recibe la información

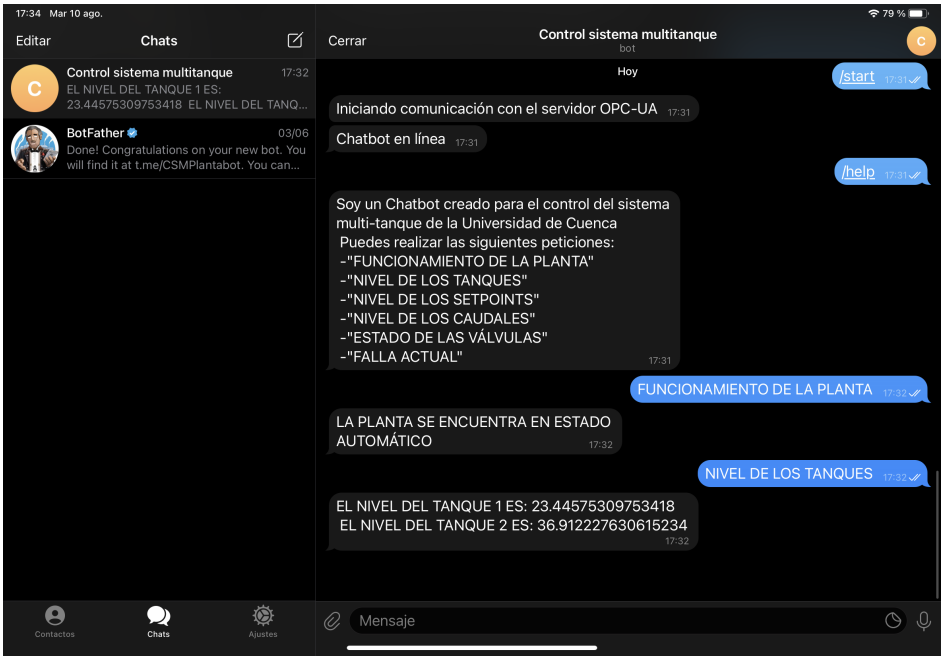


Figura 5.12: Consulta de información del sistema al *chatbot*.

desde Node-RED, es posible ejecutar la herramienta Wireshark dentro del servidor de contenedores para analizar la red de propia de Docker.

Se capturan los datos del tráfico de la red de Docker mientras se ejecutan varios procesos de comunicación. Éstos se detallan a continuación. Se menciona la menor y mayor longitud de los paquetes capturados, de manera que se analice el caso de cada protocolo por separado.

1. El servidor OPC-UA envía los datos del funcionamiento de la planta al contenedor de Node-RED bajo el protocolo OPC-UA. La Figura 5.13 presenta la captura de un paquete desde la dirección 192.168.100.20 del servidor hacia la dirección 172.17.0.2 del contenedor Node-RED. Se verifica el uso correcto del protocolo OPC-UA. En este caso, la longitud de los paquetes enviados es única, 136 bytes.

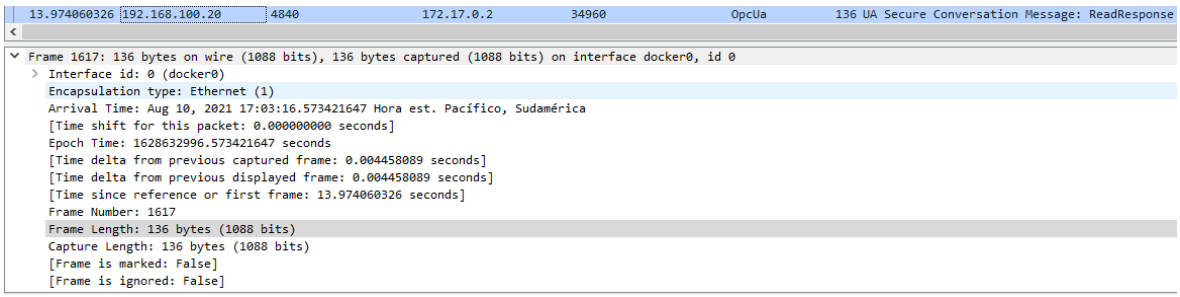


Figura 5.13: Captura de envío de paquete del servidor a Node-RED en Wireshark.

2. El contenedor Node-RED envía los datos del funcionamiento de la planta a la nube Thinger.io bajo el protocolo HTTP. La Figura 5.14 presenta la captura de un paquete desde la dirección 172.17.0.2 del contenedor Node-RED hacia la dirección 192.168.100.133 de la nube de información. Se verifica el uso correcto del protocolo HTTP. La Figura 5.15 presenta y permite verificar el método POST asignado para

esta comunicación, además se aprecia la dirección del servidor de contenedores como el *host* de este paquete de envío. En este caso, la mínima y máxima longitud de paquete analizado es 120 bytes y 583 bytes, respectivamente.

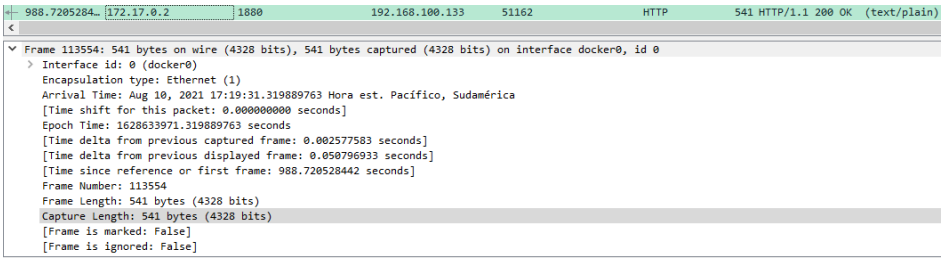


Figura 5.14: Captura de envío de paquete de Node-RED a la nube Thinger.io en Wireshark.

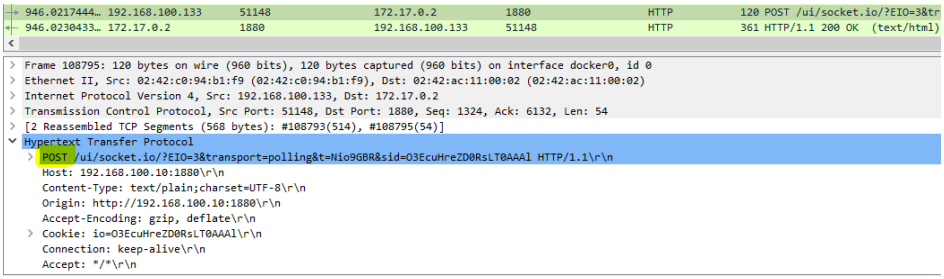


Figura 5.15: Verificación de método POST en protocolo HTTP.

3. El servidor OPC-UA envía los datos del funcionamiento de la planta al contenedor de Python bajo el protocolo OPC-UA. La Figura 5.16 presenta la captura de un paquete desde la dirección 192.168.100.20 del servidor hacia la dirección 172.17.0.5 del contenedor Python. Se verifica el uso correcto del protocolo OPC-UA. La mínima y máxima longitud de paquete analizado es 126 bytes y 128 bytes, respectivamente.

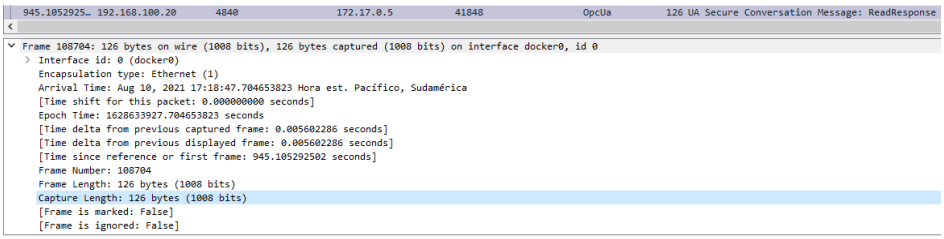


Figura 5.16: Captura de envío de paquete del servidor a Python en Wireshark.

4. El contenedor Python envía los datos del funcionamiento de la planta al contenedor de la base de datos MySQL bajo el protocolo de control de transmisión (Transmission Control Protocol (TCP)). La Figura 5.17 presenta la captura de un paquete desde la dirección 172.17.0.5 del contenedor Python hacia la dirección 192.168.100.10 de la base de datos MySQL. Se verifica el uso correcto del protocolo TCP. Para este caso, la mínima y máxima longitud de paquete analizado es 66 bytes y 286 bytes, respectivamente.
5. El servidor OPC-UA envía los datos del funcionamiento de la planta al contenedor del *chatbot* bajo el protocolo OPC-UA, cuando éste le hace una consulta. La Figura 5.18 presenta la captura de un paquete desde la dirección 192.168.100.20 del servidor hacia la dirección 172.17.0.4 del contenedor *chatbot*. Se

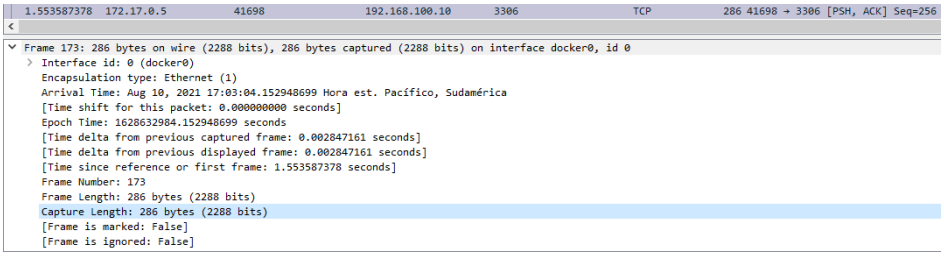


Figura 5.17: Captura de envío de paquete de Python a MySQL en Wireshark.

verifica el uso correcto del protocolo OPC-UA. La mínima y máxima longitud de paquete analizado es 150 bytes y 1086 bytes, respectivamente.

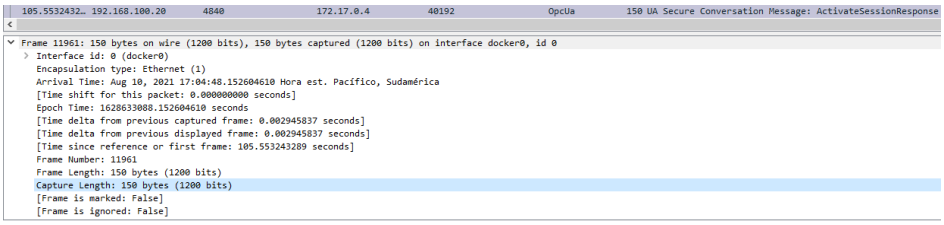


Figura 5.18: Captura de envío de paquete del servidor al *chatbot* en Wireshark.

En los paquetes de comunicación mediante protocolo OPC-UA, es interesante analizar la secuencia de éstos, desde iniciar la comunicación con un mensaje tipo *Hello* mediante protocolo OPC-UA, mensajes de confirmación tipo ACK (*acknowledgement*) mediante protocolo TCP y mensajes que habilitan el canal de comunicación, crean la sesión y la mantienen activa mediante protocolo OPC-UA. Esta secuencia de intercambio de mensajes se presenta en la Figura 5.19. Ésta corresponde a una captura del intercambio de paquetes entre el servidor y el contenedor *chatbot*, sin embargo, paquetes similares se apreciaron para las comunicaciones con Node-RED y Python durante la captura en Wireshark.

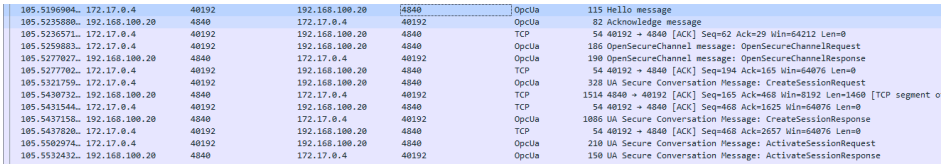


Figura 5.19: Secuencia de paquetes capturados entre servidor y *chatbot* en Wireshark.

La Tabla 5.3 presenta un resumen de las longitudes mínimas y máximas de los paquetes analizados mediante Wireshark. De ésta se analizan los siguientes resultados:

- La longitud de los paquetes bajo el protocolo OPC-UA tiende a ser fija o tiene muy poca variación, como se aprecia en el caso del envío a Node-RED y a Python. Esto ocurre debido a que, a ambas plataformas, el servidor envía el mismo paquete de información cada 1 segundo, es decir, la misma cantidad de variables.
- Para el caso de envío al *chatbot*, a pesar de ser mediante el mismo protocolo que se analiza en el punto anterior, el servidor no envía el mismo paquete cuando el *bot* le hace una consulta, sino depende de qué variables le solicite para que envíe más o menos cantidad de información. Por esta razón hay un rango más grande de variación en la longitud de los paquetes en esta comunicación.

Tabla 5.3: Longitudes mínima y máxima de paquetes analizados en el servidor de contenedores.

Comunicación	Protocolo	Longitud mínima	Longitud máxima
Servidor → *Node – RED*	OPC-UA	136 bytes	136 bytes
Node-RED → *Thinger.io*	HTTP	120 bytes	583 bytes
Servidor → *Python*	OPC-UA	126 bytes	128 bytes
Python → *MysQL*	TCP	66 bytes	286 bytes
Servidor → *Chatbot*	OPC-UA	150 bytes	1086 bytes

- La secuencia de los mensajes bajo los protocolos HTTP y TCP requiere cumplir varias instancias para el establecimiento de la comunicación. Para HTTP hay mensajes de método GET, POST, mensajes de confirmación tipo 200 OK y mensajes de cierre o reuso de conexión; y todos ellos de longitudes diferentes. Para TCP la comunicación inicia con el apretón de manos de 3 pasos (*3-Way Handshake*), confirma con mensajes ACK, envía los datos para volver a recibir una confirmación y cierra la conexión; similar al caso anterior, todos de longitudes distintas. Es esta la razón detrás de que las variaciones en longitudes de los paquetes para estos dos protocolos sea tan diferenciada.

5.3. Análisis del envío de información a la base de datos local y a la nube de almacenamiento de información

La base de datos local MySQL implementada en el servidor de contenedores Docker, recibe la información sobre el funcionamiento de la planta, sistemas de control, indicadores de fallas y demás desde el cliente Python implementado en un contenedor de Docker distinto.

El cliente Python consulta la información de la planta al servidor OPC-UA cada 1 segundo, con la finalidad de simular el tiempo de muestreo. Al mismo tiempo que el cliente consulta la información, la envía a la base de datos local. Entonces, la base de datos creada en MySQL recibe la información del sistema cada 1 segundo.

Por otra parte, la nube de almacenamiento de información Thinger.io, recibe los datos en su *databucket* de acuerdo al envío desde el cliente implementado en Node-RED. Éste envía la información que recibe del servidor cada 1 segundo, igual que Python.

Se realiza una captura de datos en la nube de almacenamiento y en la base de datos local mientras la planta funciona por 1 hora. Si el cliente Python consulta y envía los datos cada 1 segundo durante una hora, se esperaría recibir 3600 paquetes de datos en MySQL; el mismo razonamiento aplica para el envío a Thinger.io.

En la Tabla 5.4 se presenta el total de paquetes enviados, recibidos y un porcentaje que representa la pérdida de paquetes al realizar esta prueba. Se encuentra que la base de datos local recibe 2991 paquetes y la nube de información recibe sólo 60, con base en esto, se aprecia que el porcentaje de pérdida en Thinger.io es casi 6 veces el porcentaje de pérdida en MySQL. Esto se debe a las siguientes razones:

1. El programa diseñado para Python, además de establecer un cliente OPC-UA, establece un cliente NTP que permite la consulta de fecha y hora exactas. Este es un servidor de protocolo de tiempo de red externo al sistema y servidores implementados en este trabajo. Se presume que la consulta a este servidor

Tabla 5.4: Resultados de paquetes enviados, recibidos y pérdidas para MySQL y Thinger.io durante 1 hora de almacenamiento de información.

	Total paquetes enviados	Total paquetes recibidos	Porcentaje de pérdida
Base de datos local MySQL	3600	2991	17 %
Almacenamiento en nube Thinger.io	3600	60	98 %

consume un tiempo adicional al 1 segundo de envío. El diagrama de flujo presentado en la Figura 4.7 de la Sección 4.3.4.2 aclara esta explicación. Por esta razón, MySQL no recibe los paquetes exactamente cada 1 segundo, sino 1 segundo y un tiempo adicional. En consecuencia, no habrán 3600 paquetes. Una solución a este problema sería la implementación de un servidor NTP de manera interna, esta función viene dada por defecto en algunos routers, como por ejemplo en el router Mikrotik RB3011UiAS-RM.

2. La nube de información Thinger.io recibe cada 1 minuto los datos que Node-RED envía. Esto se verifica en la primera columna del *databucket* de la plataforma, pues se aprecia que un nuevo dato es almacenado exactamente cada 60 segundos, esto se presenta en la Figura 5.20. Razón por la cual se almacenan únicamente 60 paquetes al final de 1 hora. La capacidad de Thinger.io de almacenar la información cada 1 minuto es predeterminada y no es posible reajustarla. La solución a este problema sería el cambio de plataforma.

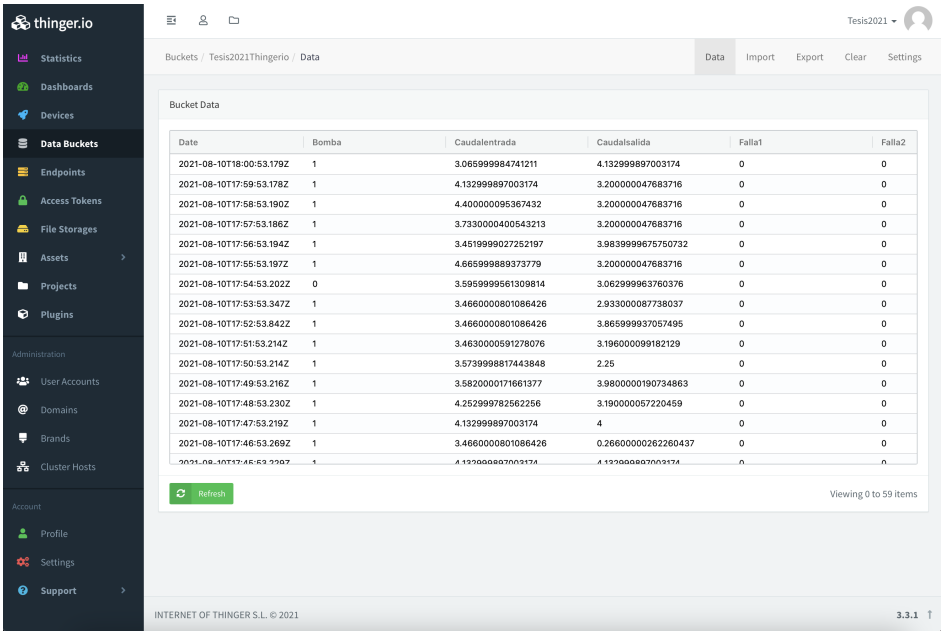


Figura 5.20: Paquetes de datos recibidos en el *databucket* de Thinger.io.

Después del conocimiento de esta información, se realiza una gráfica de comportamiento del sistema de control en el tanque 1 con la información almacenada en MySQL y en Thinger.io, con la finalidad de comparar la dinámica de la planta de acuerdo a los datos almacenados en las dos plataformas. Se toman los datos correspondientes a las variables nivel actual del tanque 1 y nivel objetivo del tanque 1 vs. tiempo, éstos se recortan a un tiempo de 15 minutos (900 segundos) de manera que las gráficas no sean extensas y se aprecien de mejor

manera.

La Figura 5.21 presenta las gráficas de comportamiento de la planta. La gráfica superior se genera tomando los datos almacenados en MySQL y la inferior de los datos de Thinger.io. Claramente al disponer de una mayor cantidad de valores para la base de datos local, ésta muestra un comportamiento definido de la planta, mientras que con los escasos datos de la nube de información, la dinámica de la planta se pierde y se aprecia un comportamiento erróneo.

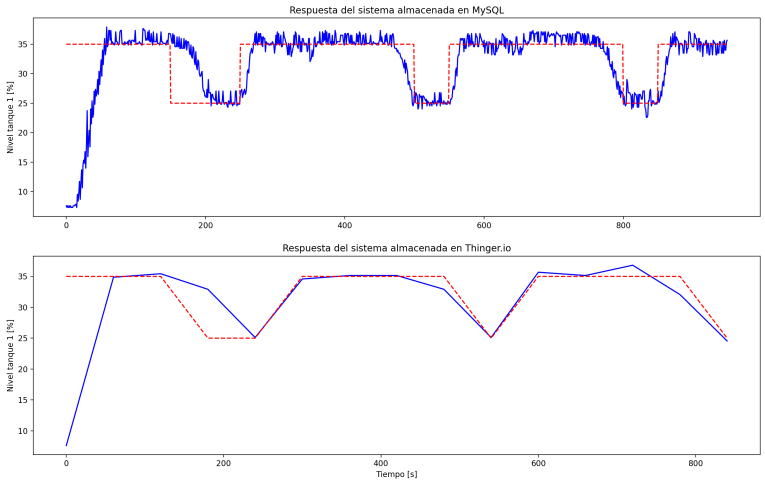


Figura 5.21: Comparación de gráficas de respuesta del sistema con los datos de MySQL y Thinger.io.

Debido a este reducido almacenamiento en la nube Thinger.io se concibe a este sistema de monitoreo desde una perspectiva de análisis de comportamiento de la planta a un nivel más general. Para un monitoreo más específico y casi a tiempo real se dispone de los demás sistemas de interacción con la planta, HMI, *dashboard* de Node-RED y *chatbot* de consulta.

5.4. Análisis del servicio de detección de fallas

Para realizar el sistema de detección de fallas se utiliza una base de datos de 900 muestras, mismas que contienen los 5 tipos de comportamientos a clasificar y se explican en la Sección 4.3.4.4. La base de datos contiene 14 variables, lo que provoca que encontrar un comportamiento específico de las variables para un tipo de fallo del sistema sea una tarea compleja. Para solventar este problema se redujo la dimensión de los datos a dos variables utilizando el algoritmo de análisis de componente principal PCA.

Siendo X la base de datos, el algoritmo consiste en encontrar la matriz de carga P, siendo esta la encargada de la reducción de la dimensión del problema. Para esto se calcula la matriz de covarianza de X, como se observa en la Ecuación 5.1, esta matriz tiene el nombre de S. A esta matriz de covarianza se aplica una descomposición en valores singulares, utilizando Python y la librería Numpy, como se observa en la Ecuación 5.2. Así, obteniendo las matrices U que corresponde a los autovectores y s que es un vector de los valores singulares. Finalmente

se extrae de la matriz U dos columnas correspondientes a las dos dimensiones objetivo del problema, como se observa en la Ecuación 5.3. A esta porción de la matriz U se la nombra P y tiene el nombre de matriz de carga y se observa en la Ecuación 5.4.

$$S = Cov(X) \tag{5.1}$$

$$U, s, Vh = linalg.svd(S) \tag{5.2}$$

$$P = U[:, 0:2] \tag{5.3}$$

$$P = \begin{bmatrix} -0{,}299 & 0{,}136 \\ -0{,}595 & 0{,}513 \\ -0{,}031 & 0{,}040 \\ -0{,}015 & 0{,}018 \\ -0{,}012 & 5{,}2e^{-4} \\ -0{,}485 & -0{,}839 \\ -0{,}004 & 0{,}009 \\ -0{,}007 & 0{,}010 \\ -0{,}329 & 0{,}005 \\ -0{,}461 & 0{,}008 \\ 1{,}84e^{-4} & 0{,}003 \\ -0{,}001 & 0{,}003 \\ 3{,}37e^{-5} & -7{,}14e^{-4} \\ -1{,}08e^{-4} & 6{,}38e^{-4} \end{bmatrix} \tag{5.4}$$

Al eliminar columnas de la matriz, se elimina información; con el vector s se puede determinar la cantidad de información que se está utilizando. Como se observa en la Ecuación 5.5, se utiliza el 97.13 % de información, lo que nos indica que es suficiente tener dos variables y no aumentar a una tercera.

$$Porcentaje = \frac{\sum_0^2 s}{\sum_0^\infty s} = 0{,}9713 \tag{5.5}$$

Al reducir el problema a dos dimensiones, se puede representar los datos de una manera geométrica. En la Figura 5.22 se muestra la gráfica del problema de clasificación. Se aprecia en color azul el comportamiento normal de la planta, en color rojo la fuga en el tanque 1, en color verde la fuga en el tanque 2, en color amarillo la falla en la válvula de entrada y en celeste la falla en la válvula de salida. Se observa zonas en las cuales existe una acumulación de puntos del mismo color, pero existen otras zonas en las cuales existen puntos de varios colores.

Estas zonas de varios colores se da porque la dinámica de la planta no presenta saltos, es decir para trasladarse de un comportamiento normal de color azul ubicado en el centro del gráfico, a una falla en la válvula de entrada presentada en color amarillo en la esquina inferior del gráfico, debe pasar por la falla de color rojo. Y así las

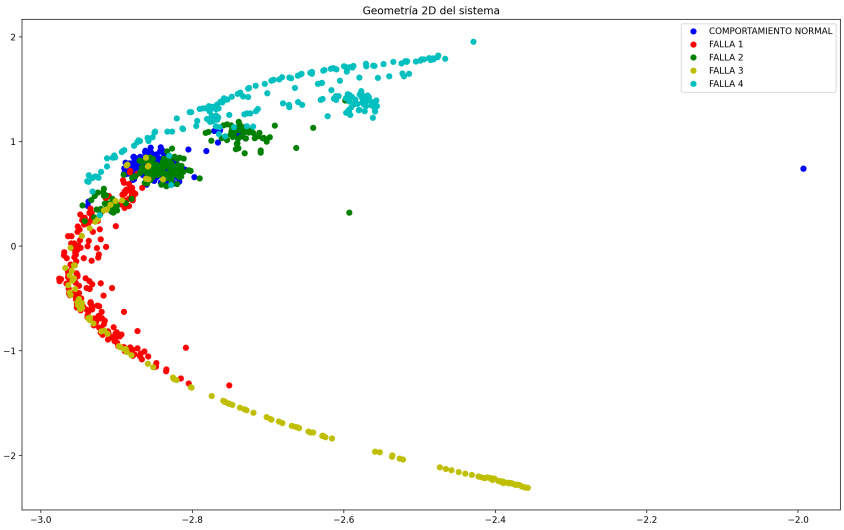


Figura 5.22: Geometría del sistema.

Tabla 5.5: Parámetros utilizados en el clasificador SVM.

Parámetro	Valor
Kernel	rbf
Parámetro de regularización Ç"	1500
Coeficiente de Kernel "gamma"	"scale"
Forma de la función de decisión "decision_function_shape"	.ºvr"

demás fallas. Esto repercute en un tiempo de espera para la detección. Los puntos que aparecen alejados de la dinámica de la planta son resultado de las perturbaciones en los tanques, esos puntos se los desecha para construcción del clasificador.

Para realizar la clasificación de las fallas se utiliza el algoritmo de *machine learning* SVM. Este algoritmo utiliza el 80 % de los datos para entrenamiento y el 20 % para evaluar el rendimiento. Los parámetros utilizados en el clasificador se muestra en la Tabla 5.5. Estos resultados se obtuvieron al construir varios clasificadores y evaluar el rendimiento.

Se aplica el algoritmo SVM a la geometría del sistema. En la Figura 5.23 se muestra los resultados. Gracias al clasificador se logra desaparecer las zonas en las que existen puntos de varios colores con esto se logra una mayor robustez en el clasificador pero incrementa los tiempos de detección.

En la Figura 5.24 se muestra la matriz de confusión del clasificador, esta se utiliza para verificar el rendimiento del algoritmo. Se observa que la falla 1 correspondiente a la fuga en el tanque 1 y la falla 4 correspondiente a la falla en la válvula de salida son clasificadas de la mejor manera, teniendo el 97.5 % de efectividad. El comportamiento normal tienen un rendimiento medio ya que presentan un 77.5 % de efectividad. Por último la falla 2 correspondiente a la fuga en el tanque 2 y la falla 3 que representa a la falla en la válvula de entrada

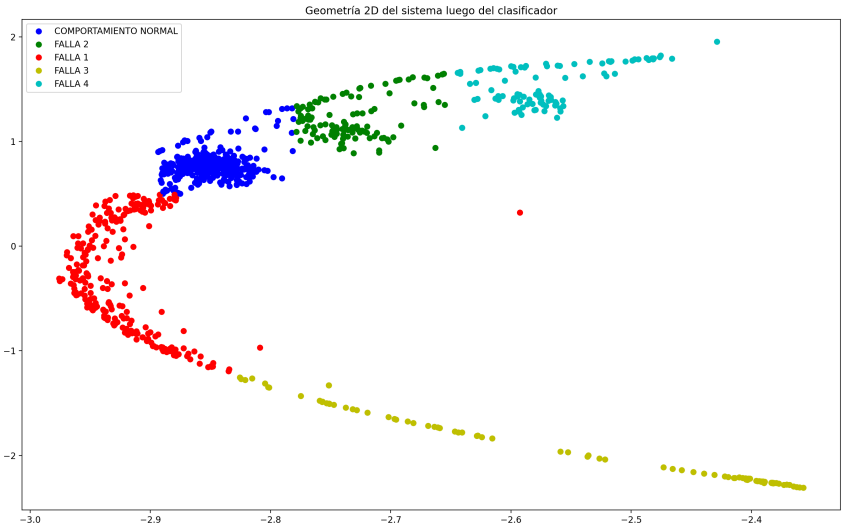


Figura 5.23: Geometría del sistema luego del clasificador.

tienen el rendimiento más bajo, con el 57.5 % y 60.71 % respectivamente.

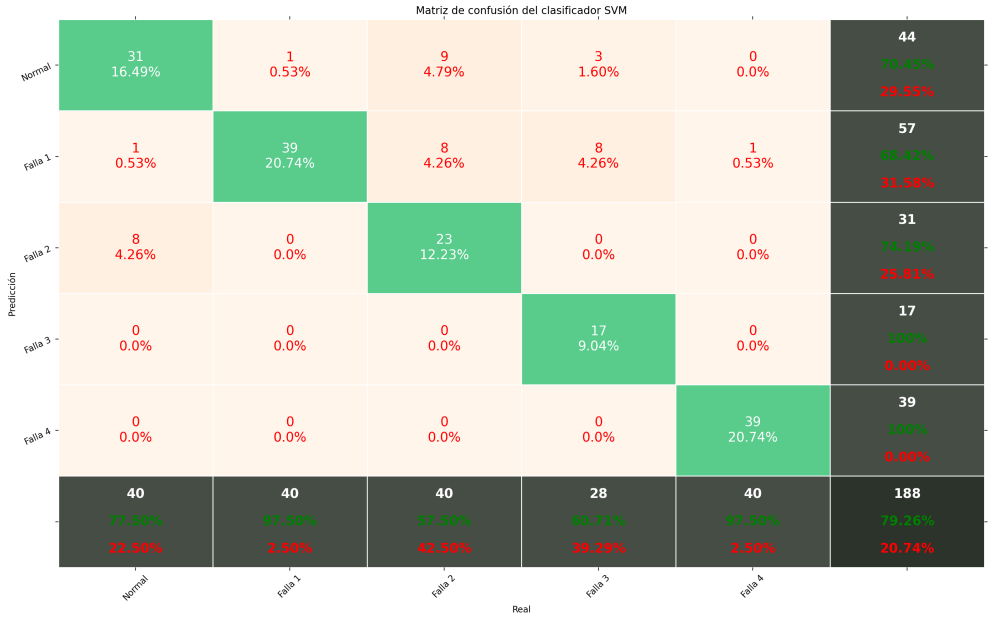


Figura 5.24: Matriz de confusión del clasificador.

5.5. Comparación con la solución IIoT de Siemens

En la industria existen varios exponentes de desarrollos IIoT, entre los más destacados está Siemens. Éste presenta tres soluciones, *Industrial Edge* enfocado a ser un primer encuentro con IIoT, diseñado para extraer la

información de la planta y procesarla de manera centralizada y en tiempo real. *MindSphere* es una herramienta para adquirir y visualizar datos y resultados analíticos inmediatos en tiempo real en una ubicación centralizada, además de tener compatibilidad con las nubes AWS, Azure, Alibaba Cloud. *Mendix* es un servicio que posee una tienda de aplicaciones en donde desarrolladores trabajan para generar nuevas propuestas.

En la Tabla 5.6 se muestra una comparación entre las características de la solución de Siemens con respecto a la solución presentada en el trabajo de titulación. La solución propuesta por Siemens tiene la ventaja de utilizar una única plataforma para el desarrollo de los servicios IIoT, esto permite un rápido despliegue del servicio, pero presenta el inconveniente de la falta de heterogeneidad. La solución propuesta tiene una complejidad mayor a la hora de desplegar un servicio, pero por otra parte es altamente configurable, posee compatibilidad con herramientas como MySQL, Node-RED, Docker, Python y esto hace que el servicio se adapte de mejor manera a la planta. En la Figura 5.25 se aprecia un diagrama de araña con la comparación de las dos soluciones, se le asigna a cada característica un valor del 1 al 5. El valor de 1 significa que la solución posee poca integración con esa característica y el valor de 5 significa que la solución posee una excelente integración con la característica.

Tabla 5.6: Comparación entre la solución IIoT Siemens y la solución desarrollada.

Característica	**Solución Siemens**	**Solución desarrollada**
Heterogeneidad	Para utilizar los servicios, es necesario que todos los dispositivos sean de la marca Siemens	Se puede integrar múltiples dispositivos y tecnologías industriales.
Compatibilidad con nubes	Posee compatibilidad con las nubes AWS, Azure, Alibaba Cloud.	Posee compatibilidad únicamente con la nube Thinger.io
Compatibilidad lenguajes de programación	Todas las soluciones IIoT se desarrollan en una única plataforma llamada *Mendix*	Es compatible con todos los lenguajes de programación que puedan comunicarse con un servidor OPC-UA
Facilidad en el despliegue de soluciones	Al tener la plataforma *Mendix*, se puede adquirir soluciones que están implementadas o desarrolladas por personas externas a la compañía	Se necesita desarrollar la solución utilizando múltiples lenguajes de programación e integrando cada uno en el servidor de contenedores.
Integración con *machine learning*	La plataforma *Mendix* posee módulos especializados en *machine learning* que son fácilmente aplicables	Al tener compatibilidad con Python, se puede crear modelos únicos para adaptarse al requerimiento específico de la solución.

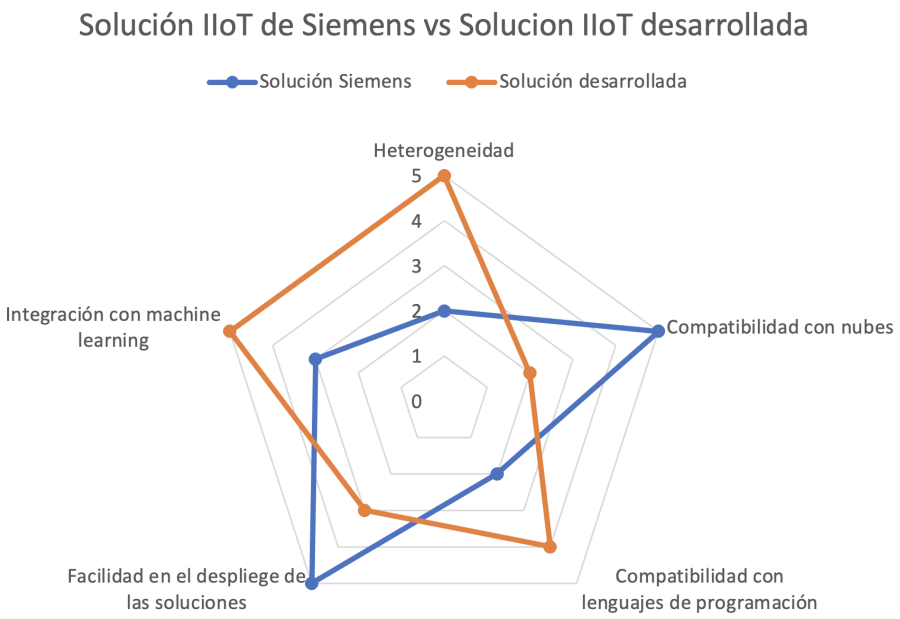


Figura 5.25: Diagrama de araña de la solución IIoT de Siemens vs. la solución IIoT desarrollada

Capítulo 6

Conclusiones y trabajos futuros

Este capítulo final presenta las conclusiones del trabajo de titulación con base en los análisis de los resultados, teniendo a consideración los objetivos planteados en el capítulo inicial de este documento. Se presentan las propuestas respecto a maneras de ampliar la investigación a futuro.

6.1. Conclusiones

- El internet de las cosas (IoT) es ampliamente aplicado a diferentes sectores de la sociedad, área de la salud, sector doméstico, ciudades inteligentes, sector productivo, entre otros. Dentro del sector productivo se introduce el concepto de la Industria 4.0 y el internet industrial de las cosas (IIoT). A pesar de que el IoT tiene una vasta gama de aplicaciones dentro de este sector, es imprescindible para su conformación la integración de diversas tecnologías, equipos y dispositivos. Este trabajo de titulación propone un servidor OPC-UA como base fundamental que permite la comunicación de tecnologías heterogéneas, y a partir de la cual se implementan demás comunicaciones de nivel superior que dan paso a la concepción de las soluciones IIoT. La ventaja de utilizar este servidor es que es *software* libre y multi-plataforma. Para los desarrolladores es especialmente útil debido a que se integra fácilmente con el lenguaje de programación Python, abriendo las puertas a una variedad de aplicaciones enfocadas en ciencia de datos.

- La arquitectura CIM utilizada para la implementación de este trabajo, permite un acercamiento real a sistemas industriales existentes en el mercado. El proceso que supone una implementación bajo esta arquitectura es secuencial. Siempre se parte de la capa de campo en la cual se integran sensores y actuadores de la planta. Posteriormente, se desarrolla la capa de control, que toma como base los datos de los dispositivos de la capa inferior para una implementación de diferentes sistemas de control y automatización. La siguiente capa, la de supervisión, se encarga de monitorear el desempeño de los sistemas de control de la capa anterior con la finalidad de predecir anomalías que se puedan corregir a tiempo, verificar el estado de los dispositivos para un futuro proceso de mantenimiento, entre otras. Asimismo, la capa posterior aloja las soluciones IIoT, que incluyen servicios de visualización, monitoreo, consulta y control por medio de varias plataformas, almacenamiento de datos de manera local y en la nube y servicio de detección de fallas. Siendo éstas, soluciones novedosas que utilizan los últimos avances en las teleco-

municaciones. Con esto se comprueba que el orden de implementación de esta arquitectura es lógico, ya que no se puede concebir una solución IIoT a partir de una planta que no ha sido debidamente desarrollada.

- Una de las partes fundamentales de este trabajo es la utilización de distintos protocolos de comunicación industrial. Desde protocolos tradicionales como Modbus hasta más novedosos como OPC-UA. Esta constituye la base de la integración de tecnologías de diferentes marcas, modelos y costos. Con esta explotación de protocolos industriales se logra comunicar dispositivos tan complejos y robustos como el PLC Siemens con el Controllino, un PLC basado en Arduino. Lo destacable de este trabajo es que la integración de estos protocolos convierten a este sistema en uno fácilmente aplicable a una planta totalmente diferente, lo que le da escalabilidad a este proyecto dentro de la industria.

- La variedad de *software* que se aplica en el desarrollo del trabajo en su totalidad, presenta varias ventajas y desventajas. Por ejemplo, el *software* Tia Portal que permite la programación de los dispositivos Siemens mediante bloques, tiene la ventaja de la facilidad de adición de funcionalidades y la transparencia en la comunicación entre estos equipos. Por otra parte, el PLC Controllino, cuya programación es estructurada, tiene la desventaja que convierte la tarea de integración de nuevas funcionalidades en una muy compleja. Para el desarrollo de las soluciones IIoT se utiliza Python y Node-RED. La principal ventaja con el lenguaje Python es su gran capacidad para la explotación y análisis de datos. Node-RED utiliza programación basada en nodos y flujos, lo que lo convierte en un *software* amigable al desarrollador, pues no se necesitan profundos conocimientos de programación para desarrollar una aplicación.

- Con la finalidad de otorgar a la solución propuesta un contexto industrial, se dispone de un tablero de instrumentación en el cual se adaptan los componentes de laboratorio utilizados. Además de esto, en él se constituyen las fases de control, desde los botones, indicadores LED y selectores, hasta la fase de supervisión con el dispositivo HMI.

- El PLC Siemens es el encargado de ejecutar el sistema de control. Este sistema regula el nivel de los tanques de manera correcta, pero presenta dificultades al momento que disminuye la referencia. Esto se debe a que los tres sistemas de control implementados no se comunican entre sí. Este problema se solventaría utilizando esquemas de control más complejos, sin embargo, éstos no son objetivo fundamental del desarrollo de este trabajo de titulación.

- Para el despliegue de la solución IIoT de monitoreo de la planta, se utilizan principalmente dos herramientas con enfoques diferentes. La primera es el *dashboard* implementado en Node-RED. Ésta se enfoca en la visualización y el control de los procesos de la planta y está orientada al personal administrativo, pues su implementación es local. La ventaja de implementarlo en esta plataforma es que es de fácil accesibilidad, pues únicamente se requiere de un navegador web. Adicionalmente a esto, las capacidades de recepción y despliegue de la información son adecuadas para el sistema, pues cada segundo se actualizan los datos. La segunda solución se implementa en la plataforma Thinger.io, debido a que cuenta con la función de panel de visualización. Ésta se enfoca únicamente en monitoreo y está orientada a personal externo a la planta, pues se encuentra alojada en la nube. Presenta la misma ventaja de la solución antes mencionada, pero las capacidades de recepción y despliegue de la información son limitadas, debido a que los datos se actualizan aproximadamente a los 60 segundos.

- La implementación de un servidor de contenedores en la plataforma Docker permite la virtualización del servicio de detección de fallas. Este servicio tradicionalmente se desarrolla en base al modelamiento de la planta; la tendencia actual es inclinarse por los datos correspondientes al comportamiento del sistema. Con base en esto, la propuesta de este trabajo, para el servicio de detección de fallas, es la utilización de herramientas de aprendizaje de máquina potenciado con análisis estadístico. Dentro del enfoque estadístico se utiliza el análisis de componente principal (PCA) cuyo objetivo es reducir la dimensión del problema de 14 a 2 variables. Esto potencia la capacidad de clasificación del algoritmo de *machine learning*.

- La ventaja que proporciona Docker en la virtualización de la capa MES es que los servicios se ejecutan de manera paralela y aislada al sistema de control y supervisión. Esto aporta un esquema modular a la solución propuesta, de manera que resultaría sencilla la implementación de nuevos servicios, sin que esto afecte al diseño actual. Este enfoque resulta beneficioso para la fase de monitoreo, puesto que en caso de presentarse una falla en alguno de los servicios de visualización o consulta de la información, los otros servicios no se verían perjudicados.

6.2. Trabajos futuros

- Implementar un esquema que comunique a los tres sistemas de control para mejorar los tiempos de estabilización al presentarse un cambio de referencia.
- Evaluar otras nubes de almacenamiento de información que mejoren el tiempo de recepción de los datos.
- Integrar el *bot* de consulta en Telegram con la base de datos para generar reportes del comportamiento de la planta. Esto, en un entorno empresarial, facilitaría al personal administrativo el acceso a información relevante.

Anexo A

Diseño de los tanques y armazón de la planta del sistema multi-tanque

En este anexo se encuentra la lámina del montaje completo de la planta (incluyendo actuadores, sensores y tuberías), así como, láminas técnicas del armazón de la planta, del tanque de bombeo y los tanques de reserva.

A6
105x148

A5
148x210

A4
210x297

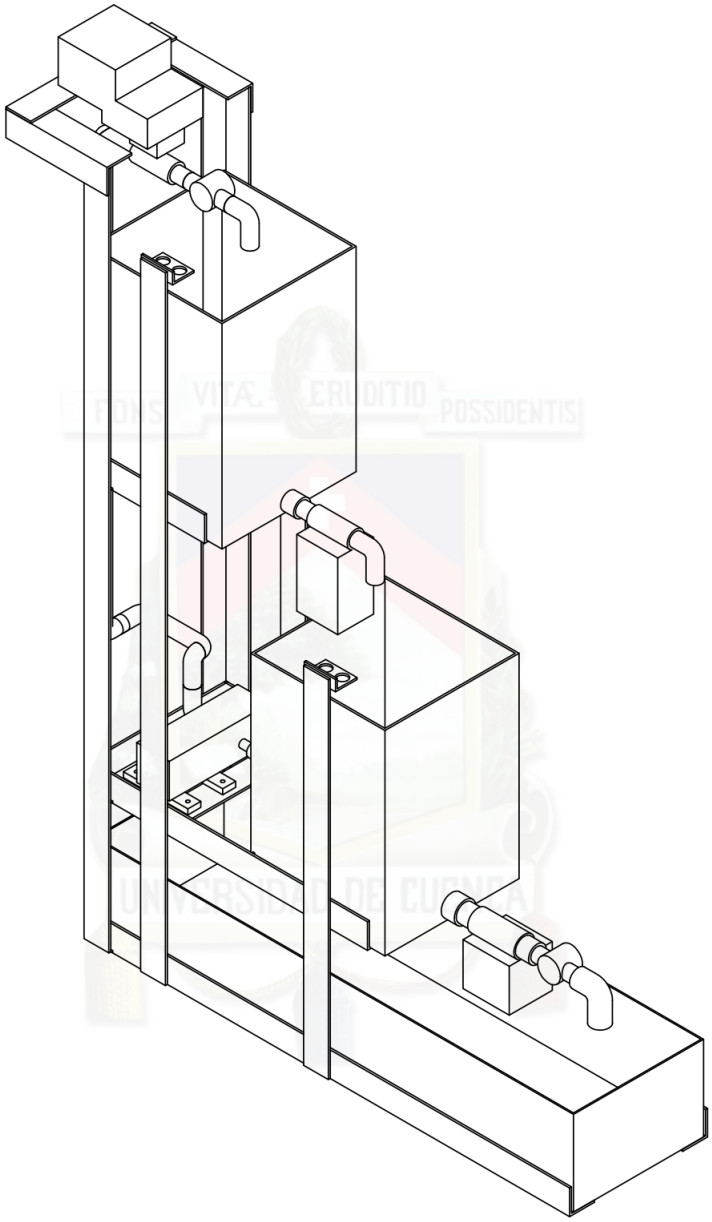
A3
297x420

A2
420x594

A1
594x841

A0
841x1189

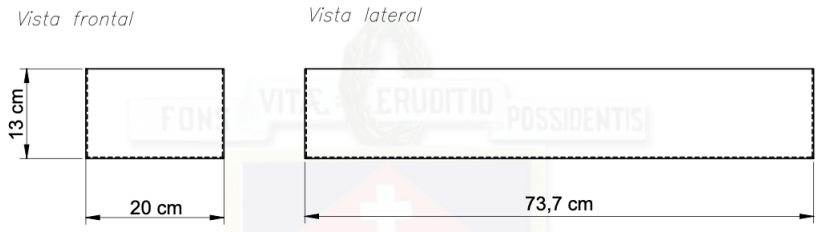
Hoja de enseñanza Técnica

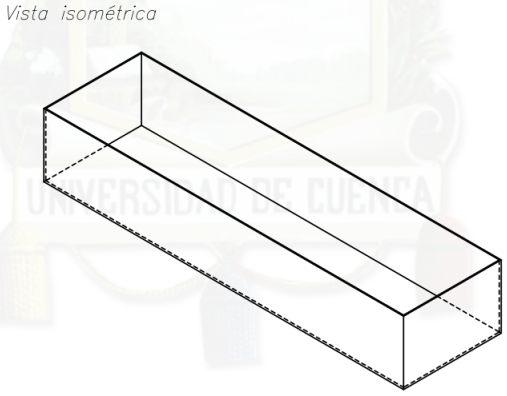


	Nombre	Fecha	Escuela de Ingeniería Electrónica y Telecomunicaciones	UNIVERSIDAD DE CUENCA	
Dibujado por:	Ricardo Enderica Fabricio López	25/06/2018			
Proyección	PLANTA DEL SISTEMA MULTITANQUE				Escala: 1:6 Lámina N° 1

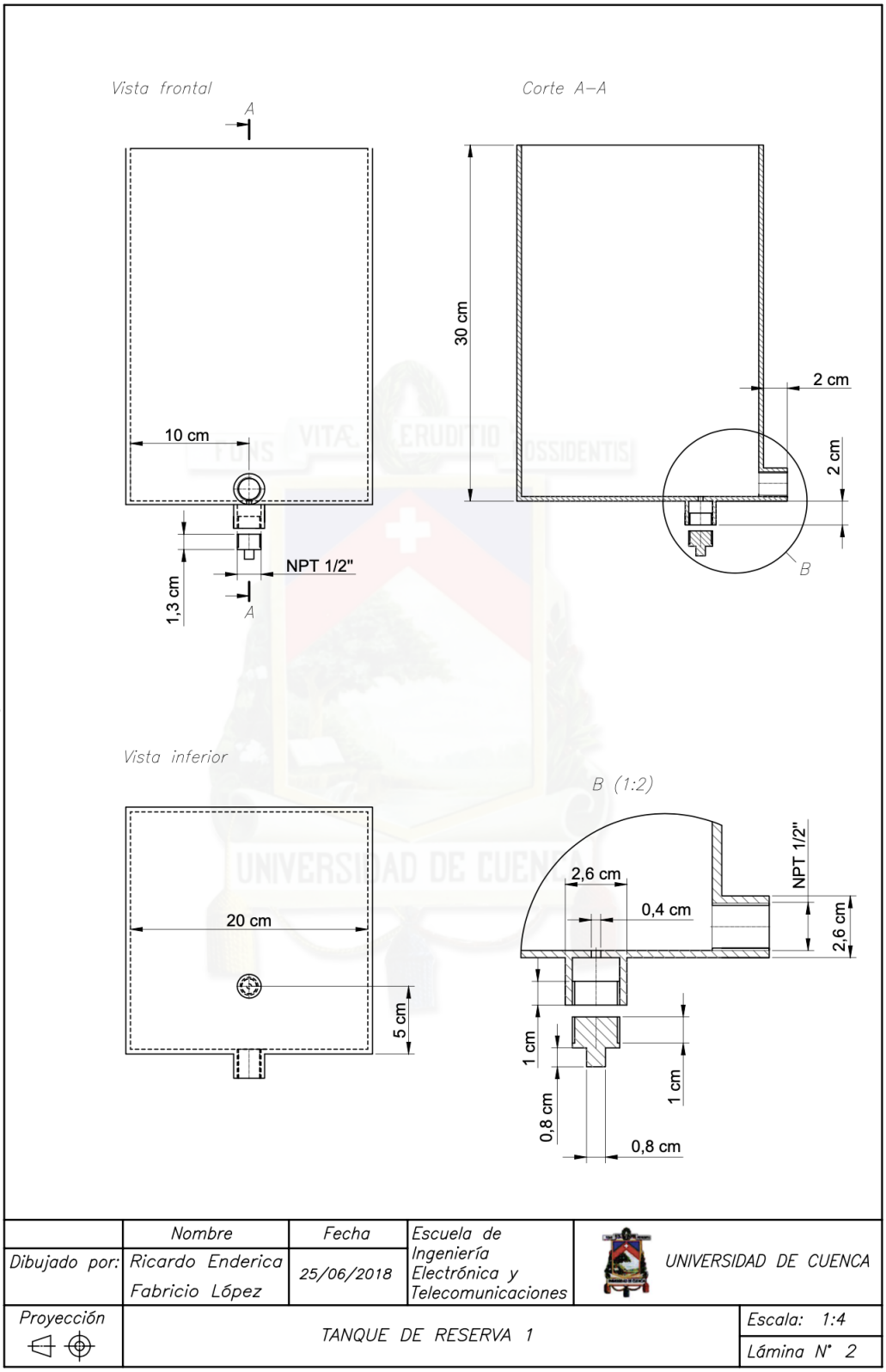
A6 105x148
A5 148x210
A4 210x297
A3 297x420
A2 420x594
A1 594x841
A0 841x1189

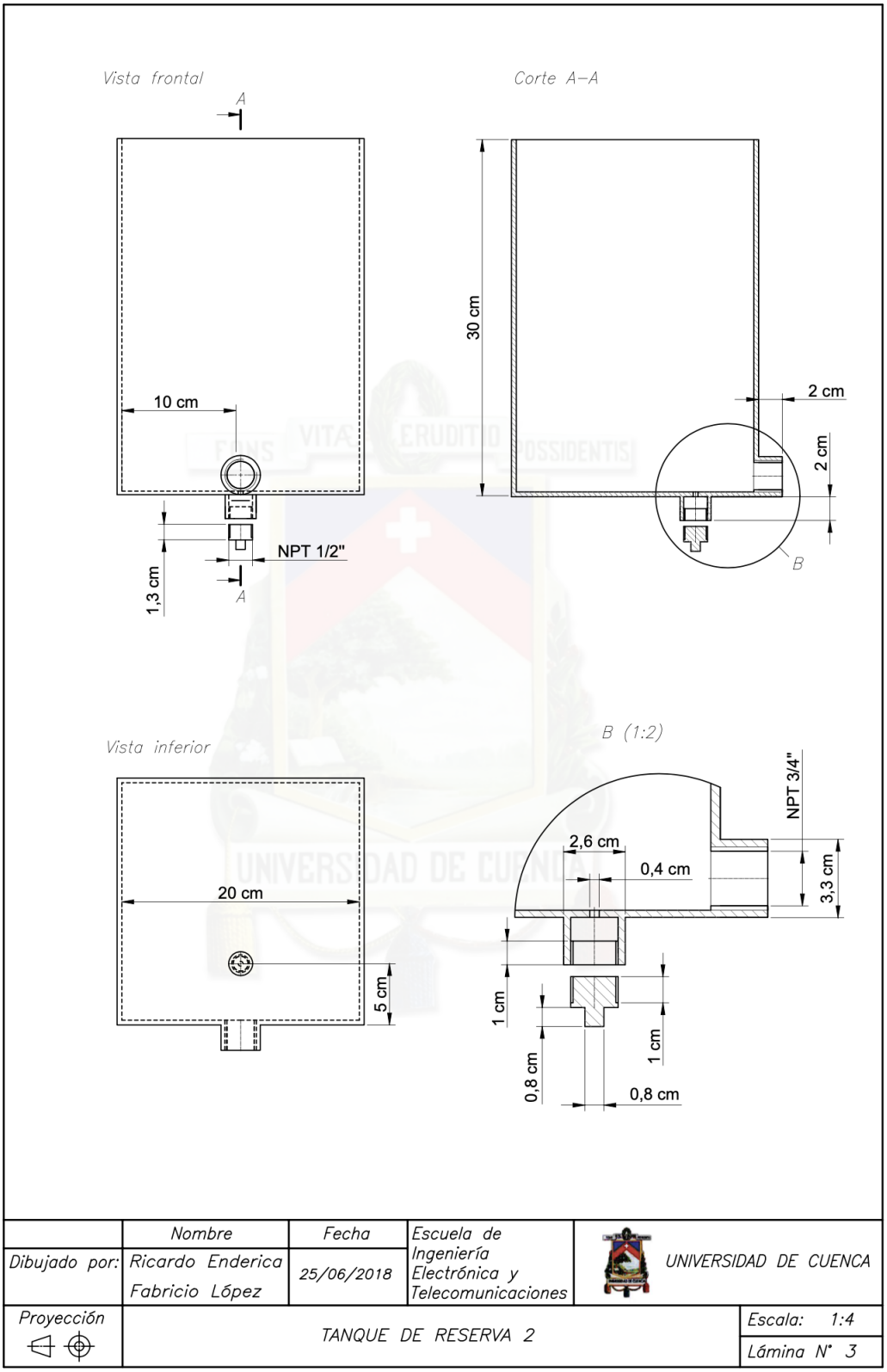
Hoja de enseñanza Técnica

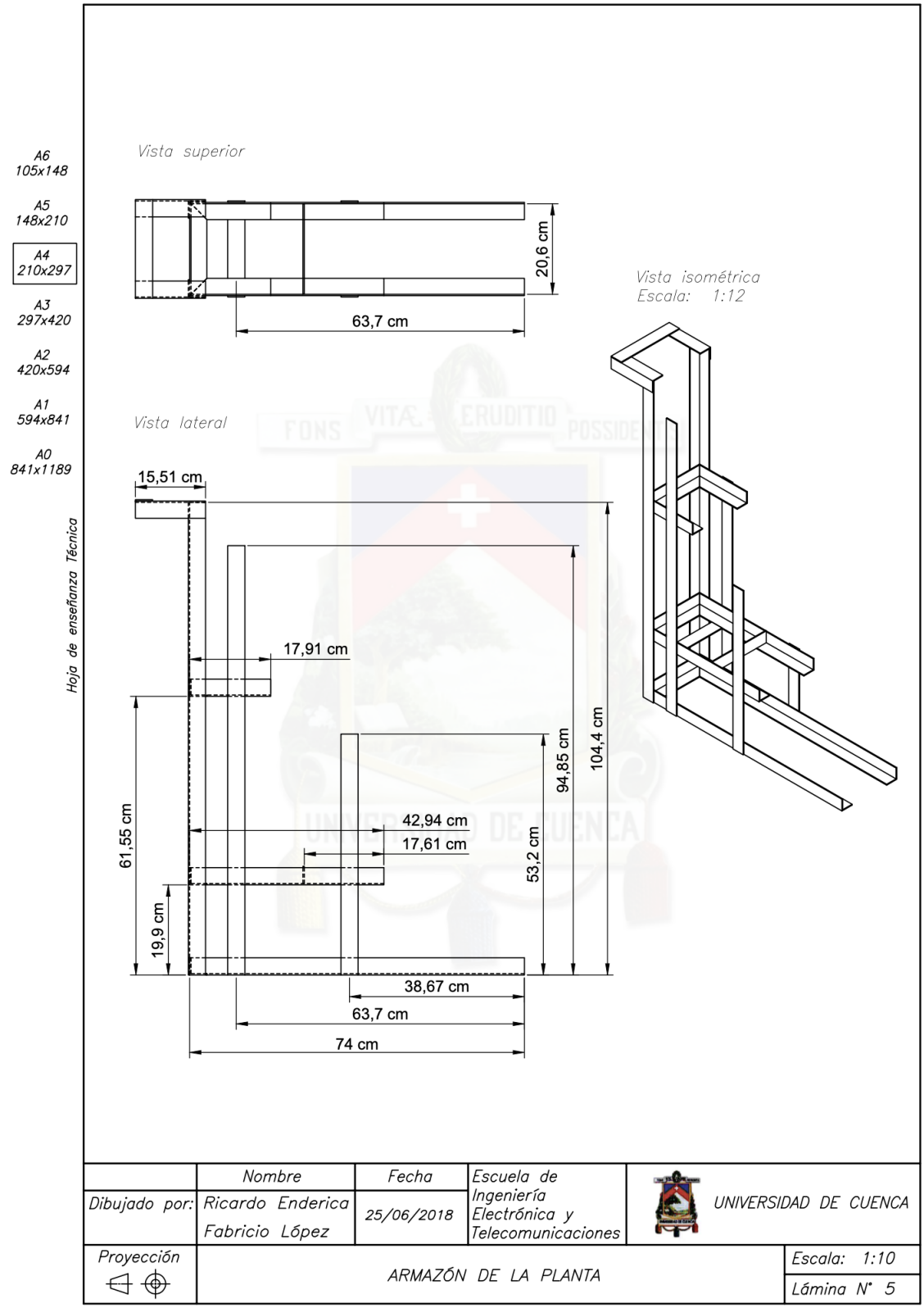




	Nombre	Fecha	Escuela de Ingeniería Electrónica y Telecomunicaciones	UNIVERSIDAD DE CUENCA
Dibujado por:	Ricardo Enderica Fabricio López	25/06/2018		
Proyección	TANQUE DE BOMBEO		Escala: 1:8	
			Lámina N° 4	


Vista frontal
Corte A–A
A
10 cm
NPT 1/2"
1,3 cm
30 cm
2 cm
2 cm
B
Vista inferior
B (1:2)
20 cm
5 cm
2,6 cm
0,4 cm
NPT 1/2"
2,6 cm
1 cm
1 cm
0,8 cm
0,8 cm
A6 105x148
A5 148x210
A4 210x297
A3 297x420
A2 420x594
A1 594x841
A0 841x1189
Hoja de enseñanza Técnica
Nombre
Fecha
Escuela de Ingeniería Electrónica y Telecomunicaciones
UNIVERSIDAD DE CUENCA
Dibujado por:
Ricardo Enderica
Fabricio López
25/06/2018
Proyección
TANQUE DE RESERVA 1
Escala: 1:4
Lámina N° 2

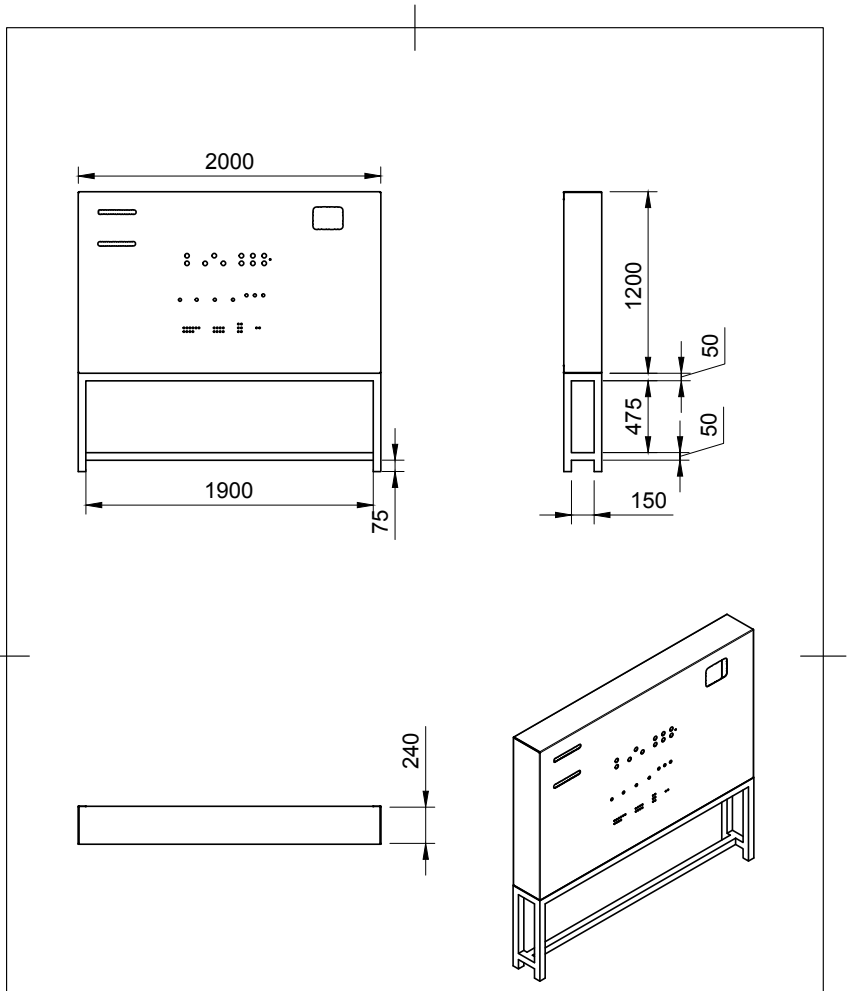

Vista frontal
Corte A–A
A
10 cm
NPT 1/2"
1,3 cm
30 cm
2 cm
2 cm
B
A6 105x148
A5 148x210
A4 210x297
A3 297x420
A2 420x594
A1 594x841
A0 841x1189
Hoja de enseñanza Técnica
Vista inferior
B (1:2)
20 cm
5 cm
NPT 3/4"
2,6 cm
0,4 cm
3,3 cm
1 cm
1 cm
0,8 cm
0,8 cm
Nombre
Fecha
Escuela de Ingeniería Electrónica y Telecomunicaciones
UNIVERSIDAD DE CUENCA
Dibujado por:
Ricardo Enderica
Fabricio López
25/06/2018
Proyección
TANQUE DE RESERVA 2
Escala: 1:4
Lámina N° 3


A6 105x148
A5 148x210
A4 210x297
A3 297x420
A2 420x594
A1 594x841
A0 841x1189
Hoja de enseñanza Técnica
Vista superior
20,6 cm
63,7 cm
Vista isométrica
Escala: 1:12
Vista lateral
15,51 cm
17,91 cm
104,4 cm
94,85 cm
61,55 cm
42,94 cm
17,61 cm
53,2 cm
19,9 cm
38,67 cm
63,7 cm
74 cm
Nombre
Fecha
Dibujado por:
Ricardo Enderica
Fabricio López
25/06/2018
Escuela de Ingeniería Electrónica y Telecomunicaciones
UNIVERSIDAD DE CUENCA
Proyección
ARMAZÓN DE LA PLANTA
Escala: 1:10
Lámina N° 5

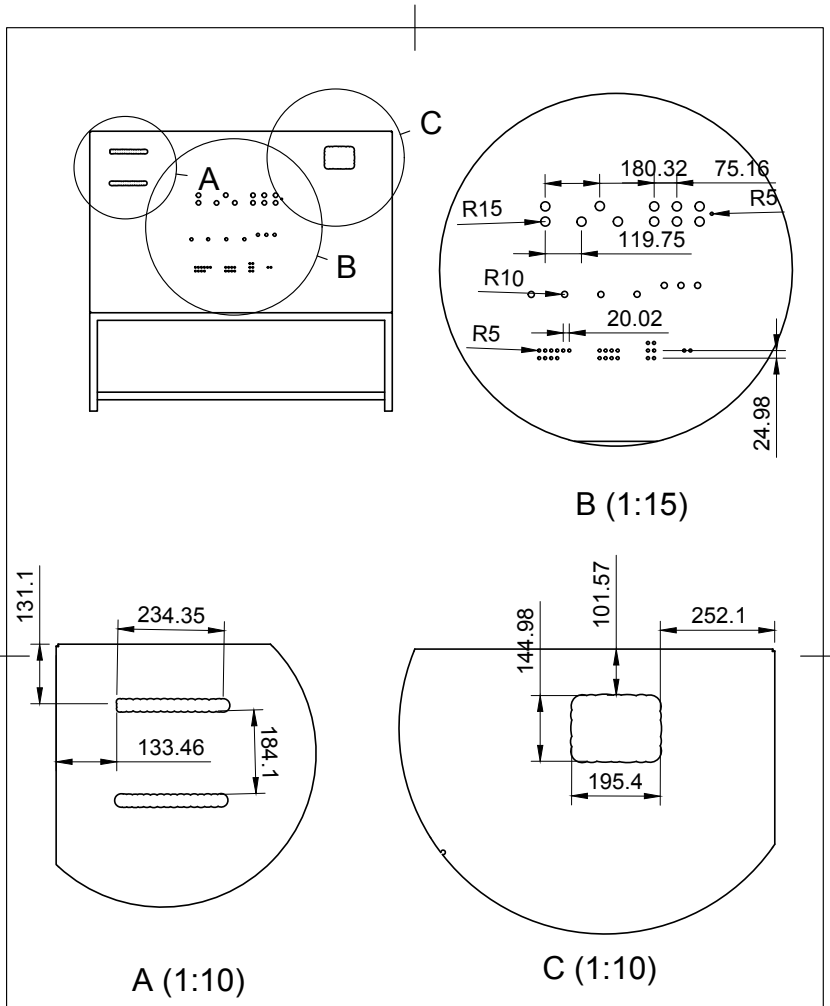
ANEXO B

Diseño del tablero de instrumentación

En este anexo se presenta las láminas técnicas del tablero de instrumentación de la planta.



	Nombre:	Fecha:	Escuela de ingeniería electrónica y telecomunicaciones	
Dibujado por:	Marcelo Garzon Erika Serrano	25/07/21		
	Tablero de instrumentación			Escala: 1:20 Lamina N* 1



	Nombre:	Fecha:	Escuela de ingeniería electrónica y telecomunicaciones	
Dibujado por:	Marcelo Garzon Erika Serrano	25/07/21		
	Detalles del tablero de instrumentación			Escala: 1:20
				Lamina N* 2

Bibliografía

[1] C. Ynzunza, J. Izar, J. Bocarando, F. Aguilar, y M. Larios, "El Entorno de la Industria 4.0: Implicaciones y Perspectivas Futuras," *CONCIENCIA TECNOLÓGICA*, num. 54, pp. 33–45, 2017.

[2] N. HuanSheng y L. Hong, "Cyber-physical-social-thinking space based science and technology framework for the Internet of Things," *SCIENCE CHINA Information Sciences*, vol. 58, pp. 1–19, 2015.

[3] H. Kagermann, W. Wahlster, y J. Helbig, "Recommendations for implementing the strategic initiative INDUSTRIE 4.0," *Industrie 4.0 Working Group*, 2013.

[4] A. Gilchrist, *INDUSTRY 4.0 THE INDUSTRIAL INTERNET OF THINGS*, ser. INDUSTRY 4.0. Apress, 2016.

[5] J. Val Román, "Industria 4.0: la transformación digital de la industria," *CONFERENCIA DE DIRECTORES Y DECANOS DE INGENIERÍA INFORMÁTICA*, p. 120, 2012.

[6] A. Arbildo, "El control de procesos industriales y su influencia en el mantenimiento," *Ingeniería Industrial*, num. 29, pp. 35–49, 2011.

[7] M. López, "INDUSTRIA 4.0 PARA LA MONITORIZACIÓN DE UN PROCESO INDUSTRIAL," *Facultad de Tecnologías de la Información, Telecomunicaciones e Industrial, Universidad Técnica de Ambato*, 2019.

[8] H. Boyes, B. Hallaq, J. Cunningham, y T. Watson, "The industrial internet of things (IIoT): An analysis framework," *Computers in Industry*, num. 101, pp. 1–12, 2018.

[9] R. Neugebauer, S. Hippmann, M. Leis, y M. Landherr, "Industrie 4.0 - From the perspective of applied research," *CIRP Conference on Manufacturing Systems*, num. 57, pp. 2–7, 2016.

[10] I. Butun, *Industrial IoT Challenges, Design Principles, Applications, and Security*, ser. Industrial IoT. Springer, 2020.

[11] A. Karmakar, N. Dey, T. Baral, y M. Chowdhury, "Industrial Internet of Things: A Review," *Proc. Int. Conf. Opto-Electron. Appl. Opt. (Optronix)*, pp. 1–6, 2019.

[12] G. Kanagachidambaresan, R. Anand, E. Balasubramanian, y V. Mahima, *Internet of Things for Industry 4.0 Design, Challenges and Solutions*, ser. EAI/Springer Innovations in Communication and Computing. Springer, 2020.

[13] M.-A. Castro, G. Díaz, F. Pérez, R. Fernández, E. Ruiz, V. Sempere, J. Blanes, J. Fuertes, P. Colom, J. Yepez, P. Espiñeira, M. Domínguez, y R. Bayón, *COMUNICACIONES INDUSTRIALES: PRINCIPIOS BÁSICOS*,

ser. Ingeniería Industrial - Unidades Didácticas. UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA (UNED), 2007.

[14] C. Yu, X. Xu, y Y. Lu, "Computer-Integrated Manufacturing, Cyber-Physical Systems and Cloud Manufacturing – Concepts and relationships," *ScienceDirect Manufacturing Letters*, num. 6, pp. 5–9, 2015.

[15] G. Doumeingts, B. Vallespir, y D. Chen, "Metlhodologies for designing CIM systems: A survey," *Computers in Industry*, num. 25, pp. 263–280, 1995.

[16] E. Gómez, L. García, y L. Hernández, "Buses de campo. Estrategias de aplicación," *Ciencias de la Ingeniería y Tecnología, Handbook*, pp. 291–298, 2014.

[17] J. Torres y A. Vega, "UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA," *Carrera de Ingeniería Electrónica, Universidad Politécnica Salesiana Sede Cuenca*, 2015.

[18] N. Mathivanan, "PC-based Instrumentation: Concepts and Practice," *Prentice-Hall*, 2007.

[19] F. OPC. ¿Qué es OPC? [En línea]. Disponible: https://opcfoundation.org/about/what-is-opc/

[20] P. Drahos, E. Kucera, O. Haffner, y I. Klimo, "Trends in Industrial Communication and OPC UA," *Cybernetics & Informatics (K&I)*, 2018.

[21] S. Cavalieri y F. Chiacchio, "Analysis of OPC UA performances," *Computer Standards Interfaces*, num. 23, pp. 165–177, 2013.

[22] A. Maxim, D. Copot, C. Copot, y C. Ionescu, "The 5W's for Control as Part of Industry 4.0: Why, What, Where, Who, and When—A PID and MPC Control Perspective," *Inventions*, vol. 4, num. 10, pp. 1–10, 2019.

[23] A. Visioli, *Basics of PID Control. In: Practical PID Control.* Springer, 2006, ch. 1.

[24] M. Johnson y M. Moradi, *PID Control New Identification and Design Methods*, ser. PID Control. Springer, 2005.

[25] C. Boettiger, "An introduction to Docker for reproducible research," *ACM SIGOPS Operating Systems Review*, num. 49, pp. 71–79, 2015.

[26] I. Miell y A. Sayers, *Docker In Practice, Second Edition*, ser. Docker. MANNING Shelter Island, 2019.

[27] J. Turnbull, "The Docker Book: Containerization is the new virtualization," 2019.

[28] B. Rad, H. Bhatti, y M. Ahmadi, "An Introduction to Docker and Analysis of its Performance," *IJCSNS International Journal of Computer Science and Network Security*, vol. 17, num. 3, pp. 228–235, 2017.

[29] Z. Gao, S. Ding, y C. Cecati, "Real-Time Fault Diagnosis and Fault-Tolerant Control," *IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS*, vol. 62, num. 6, pp. 3752–3759, 2015.

[30] S. Aranguren y R. Tarantino, "METODOLOGIAS Y TECNOLOGIAS DE DETECCION Y DIAGNOSTICO DE FALLAS APLICADAS A PROCESOS INDUSTRIALES," *Revista Colombiana de Tecnologías de Avanzada*, vol. 1, num. 13, pp. 106–116, 2009.

[31] X. He, Z. Wang, Y. Liu, L. Qin, y D. Zhou, "Fault Tolerant Control for an Internet-Based Three-Tank System: Accommodation to Sensor Bias Faults," *IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS*, 2016.

[32] S. Gajjar, M. Kulahci, y A. Palazoglu, "Real-time fault detection and diagnosis using sparse principalcomponent analysis," *Journal of Process Control*, num. 67, pp. 112–128, 2018.

[33] L. Hurtado-Cortés, E. Villarreal-López, y L. Villarreal-López, "Fault detection and diagnosis through artificial intelligence techniques, a state of art," *Dyna*, vol. 83, num. 199, pp. 19–28, 2016.

[34] S. Yin, X. Gao, H. Karimi, y X. Zhu, "Study on Support Vector Machine-Based Fault Detection in Tennessee Eastman Process," *Hindawi Publishing Corporation - Abstract and Applied Analysis*, vol. 2014, 2014.

[35] S. Gunn, "Support Vector Machines for Classification and Regression," *Science and Mathematics School of Electronics and Computer Science*, 1998.

[36] F. Gamboa Quintanilla, O. Cardin, A. L'Anton, y P. Castagna, "A modeling framework for manufacturing services in service-oriented holonic manufacturing systems," *Eng. Appl. Artif. Intell.*, num. 55, pp. 26–36, 2016.

[37] T. Borangiu, D. Trentes, A. Thomas, P. Leitao, y J. Barata, "Digital transformation of manufacturing through cloud services and resource virtualization," *Computers in Industry*, num. 108, pp. 150–162, 2019.

[38] W. Lizhe, T. Jie, M. Kunze, A. C. Castellanos, D. Kramer, y W. Karl, "Scientific Cloud Computing: Early Definition and Experience," *10th IEEE Int. Conference on High Performance Computing and Communications*, pp. 825–830, 2008.

[39] T. Swathi, K. Srikanth, y S. Raghunath, "VIRTUALIZATION IN CLOUD COMPUTING," *International Journal of Computer Science and Mobile Computing*, vol. 3, num. 5, pp. 540–546, 2014.

[40] C. Wei, Z. Yu, y S. Fong, "How to Build a Chatbot: Chatbot Framework and its Capabilities," *Association for Computing Machinery*, 2018.

[41] F. Marcondes, J. Almeida, y P. Novais, "A Short Survey on Chatbot Technology: Failure in Raising the State of the Art," *Springer*, pp. 28–36, 2020.

[42] A. Adikari, D. De Silva, D. Alahakoon, y X. Yu, "A Cognitive Model for Emotion Awareness in Industrial Chatbots," *IEEE*, pp. 183–186, 2019.

[43] N. Brito, P. Ribeiro, F. Soares, C. Monteiro, V. Carvalho, y R. Vasconcelos, "A Remote System for Water Tank Level Monitoring and Control - a Collaborative Case-study," *IEEE*, pp. 19–23, 2009.

[44] S. Singh, S. Singh, y R. Dohare, "Optimal PID controller design for level control of three tank system," *International Journal of Advanced Technology and Engineering Exploration*, vol. 4, num. 26, 2016.

[45] F. Alharbi, "WIRELESS TANK CONTROL AND MONITORING USING IOT," *International Journal of Mechanical Engineering and Technology (IJMET)*, vol. 10, num. 11, pp. 192–198, 2019.

[46] D. Shah y D. Patel, "Design of sliding mode control for quadruple-tank MIMO processwith time delay compensation," *Journal of Process Control*, vol. 76, pp. 46–61, 2019.

[47] S. Profanter, A. Tekat, K. Dorofeev, y A. Knoll, "OPC UA versus ROS, DDS, and MQTT: Performance Evaluation of Industry 4.0 Protocols," *IEEE*, pp. 955–962, 2019.

[48] Y. Xun y S. Hong, "An AutomationML/OPC UA-based Industry 4.0 Solution for a Manufacturing System," *IEEE*, pp. 543–550, 2018.

[49] S. Nagalingam y G. Lin, "CIM—still the solution for manufacturing industry," *Robotics and Computer-Integrated Manufacturing*, num. 24, pp. 332–344, 2008.

[50] A. Remli, A. Khtira, y B. El Asri, "Reference Architecture for Efficient Computer Integrated Manufacturing," *Proceedings of the 23rd International Conference on Enterprise Information Systems*, vol. 1, pp. 328–334, 2021.

[51] E. Oztemel y S. Gursev, "Literature review of Industry 4.0 and related technologies," *Journal of Intelligent Manufacturing*, 2018.

[52] J. Jasperniete y J. Feld, "PROFINET: An Integration Platform for heterogeneous Industrial Communication Systems," *IEEE*, vol. 1, pp. 815–822, 2005.

[53] B. Vogel-Heuser, C. Diedrich, D. Pantförder, y P. Göhner, "Coupling heterogeneous production systems by a multi-agent based cyber-physical production system," *IEEE*, pp. 713–719, 2014.

[54] R. Enderica y F. López, "Diseño e implementación de los sistemas de instrumentación, comunicaciones y control de un sistema multi-tanque," *Facultad de Ingeniería, Electrónica y Telecomunicaciones, Universidad de Cuenca*, 2018.