

UNIVERSIDAD EUROPEA DE MADRID

ESCUELA DE ARQUITECTURA, INGENIERÍA, CIENCIA Y COMPUTACIÓN (STEAM)

GRADO EN INGENIERÍA EN SISTEMAS INDUSTRIALES

TRABAJO FIN DE GRADO

INTERCONEXIÓN MATLAB/SIMULINK – PLC SIEMENS MEDIANTE OPC-UA

Alumno: D. Roberto Castedo Montes

Director: D. Nourdine Aliane

JUNIO 2026

TÍTULO: Interconexión MATLAB/Simulink – PLC Siemens mediante OPC-UA

AUTOR: Roberto Castedo Montes

DIRECTOR DEL PROYECTO: Nourdine Aliane

FECHA: 07 de junio de 2026

Resumen

Este Trabajo Fin de Grado desarrolla una arquitectura de control distribuida para regular el nivel de agua en un depósito virtual, integrando MATLAB/Simulink, TIA Portal, SIMATIC PLCSIM Advanced y comunicación industrial OPC-UA. El objetivo principal es validar la interacción entre un modelo dinámico de planta y un controlador industrial ejecutado en un PLC virtual, reproduciendo un entorno de ensayo previo a una posible implementación física.

El proceso se modela en Simulink mediante una ecuación lineal de primer orden que describe la evolución del nivel del depósito en función del caudal de entrada y de la descarga. Sobre este modelo actúa un regulador PID_Compact configurado en TIA Portal y ejecutado en una CPU Siemens S7-1500 simulada. La comunicación bidireccional se realiza mediante OPC-UA: Simulink envía al PLC la altura del líquido y recibe la acción de control calculada por el regulador.

Además, se desarrolla una interfaz HMI para operar y supervisar el sistema, permitiendo encender y detener la aplicación, modificar la consigna, introducir perturbaciones sobre la descarga y visualizar las variables principales del lazo. También se implementa una animación del tanque con gráfica en tiempo real.

La validación se realiza mediante pruebas de comunicación, seguimiento de consignas, análisis del controlador PID y respuesta ante perturbaciones. Los resultados muestran que la arquitectura propuesta permite cerrar correctamente el lazo de control, alcanzar las referencias establecidas y recuperar el régimen estacionario tras alteraciones externas.

Palabras clave: OPC-UA, MATLAB/Simulink, TIA Portal, PLCSIM Advanced, PID_Compact, PLC, gemelo digital, control de nivel, HMI, automatización industrial.

Abstract

This Final Degree Project develops a distributed control architecture for regulating the water level in a virtual tank, integrating MATLAB/Simulink, TIA Portal, SIMATIC PLCSIM Advanced, and OPC-UA industrial communication. The main objective is to validate the interaction between a dynamic plant model and an industrial controller executed on a virtual PLC, reproducing a testing environment prior to a possible physical implementation.

The process is modeled in Simulink using a first-order linear equation that describes the evolution of the tank level according to the inlet flow and the outlet dynamics. This model is controlled by a PID_Compact regulator configured in TIA Portal and executed on a simulated Siemens S7-1500 CPU. Bidirectional communication is performed through OPC-UA: Simulink sends the liquid level to the PLC and receives the control action calculated by the regulator.

In addition, an HMI interface is developed to operate and supervise the system, allowing the user to start and stop the application, modify the setpoint, introduce disturbances in the outlet dynamics, and monitor the main control-loop variables. A tank animation with a real-time level graph is also implemented.

The system is validated through communication tests, setpoint tracking, PID controller analysis, and disturbance response evaluation. The results show that the proposed architecture successfully closes the control loop, reaches the defined references, and recovers steady-state operation after external disturbances.

Keywords: OPC-UA, MATLAB/Simulink, TIA Portal, PLCSIM Advanced, PID_Compact, PLC, digital twin, level control, HMI, industrial automation.

ÍNDICE

Resumen 1
Abstract 2
Capítulo 1 Introducción 9
1.1 Antecedentes y justificación 9
1.2 Objetivo general 10
1.3 Objetivos específicos 10
1.4 Metodología y planificación del proyecto 12
1.4.1 Cronograma 13
1.5 Alcance del proyecto 15
Capítulo 2 Antecedentes 17
2.1 El contexto de la industria 4.0 17
2.1.1 Transformación digital e integración de sistemas industriales 17
2.1.2 Interoperabilidad y estandarización 19
2.1.3 Datos y analítica industrial 20
2.2 El gemelo digital 21
2.2.1 Definición, arquitectura y estandarización 21
2.3 El protocolo OPC 23
2.3.1 Evolución a OPC-UA y principios de diseño 23
2.3.2 Modelado de información, especificaciones complementarias, PubSub y seguridad 24
2.4 Ejemplos de aplicaciones 26
2.4.1 Puesta en marcha virtual con gemelo digital 26
2.4.2 Integración de datos y trazabilidad con OPC-UA 27
2.4.3 Monitorización y mantenimiento predictivo 27
Capítulo 3 Desarrollo e Implementación 29
3.1 Planteamiento del caso de estudio 29
3.1.1 Descripción física del sistema seleccionado 30
3.1.2 Análisis de los modelos alternativos revisados 34
3.1.3 Justificación final del modelo seleccionado 35
3.2 Modelado virtual en MATLAB/Simulink 36
3.2.1 Configuración del cliente OPC-UA en Simulink 40
3.2.2 Bloque OPC-UA-Read 41
3.2.3 Bloque OPC-UA-Write 42

3.2.4 Bloque de animación y visualización 45
3.3 Control PID en el entorno TIA Portal 47
3.3.1 TIA Portal como entorno de ingeniería 48
3.3.2 Selección del PLC 48
3.3.3 Configuración de la comunicación OPC-UA en TIA Portal 49
3.3.4 Organización de los bloques de datos 49
3.3.5 Fundamentos del regulador PID 52
3.3.6 Implementación del regulador mediante PID_Compact 53
3.3.7 Ajuste de los parámetros del PID 58
3.3.8 Relación entre la comunicación OPC-UA y el lazo de control 63
3.4 Implementación del sistema y supervisión 64
3.4.1 Programa de operación desarrollado en TIA Portal 64
3.4.2 Interfaz HMI para la operación y supervisión del sistema 67
3.4.3 Funcionamiento general de la aplicación 74
Capítulo 4 Pruebas y validación 79
4.1 Definición de las pruebas 80
4.1.1 Pruebas de comunicación mediante OPC-UA 81
4.1.2 Validación del modelo dinámico y de la animación en tiempo real 82
4.1.3 Pruebas del controlador PID 83
4.1.4 Pruebas de funcionamiento del sistema completo 84
4.2 Análisis e interpretación de los resultados 85
4.2.1 Resultados de las pruebas de comunicación OPC-UA 86
4.2.2 Resultados de la validación del modelo dinámico y de la animación 89
4.2.3 Resultados de las pruebas del controlador PID 93
4.2.4 Resultados del funcionamiento global del sistema 99
4.3 Posibles mejoras 110
Capítulo 5 Conclusiones 112
5.1 Resumen del trabajo 112
5.2 Conclusiones sobre los objetivos del proyecto 113
5.3 Líneas futuras de trabajo 115
Bibliografía 117
Anexos 122
Anexo A. Código de la función calcular_PID_tanque_optimizacion.m 122
Anexo B. Código de la función tank_anim_gui_simple.m 128
Anexo C. Código de la S-Function tank_anim_sfun_simple.m 137

Índice de Figuras

Figura 1: Esquema de integración vertical y horizontal en Industria 4.0. Fuente: elaboración propia a partir de [6]. 19
Figura 2: Arquitectura general de un gemelo digital industrial. Fuente: elaboración propia a partir de [16]. 22
Figura 3: Evolución de OPC clásico a OPC-UA. Fuente: adaptado de [21]. 24
Figura 4: Esquema básico del espacio de direcciones de un servidor OPC-UA. Fuente: adaptado de [25]. 25
Figura 5: Estructura física simplificada del depósito utilizado como caso de estudio. Fuente: adaptado de [36]. 31
Figura 6: Implementación de la ecuación del depósito en Simulink. 37
Figura 7: Modelo Simulink del sistema de control de nivel. 38
Figura 8: Configuración del Solver de Simulink. 39
Figura 9: Configuración del cliente OPC-UA. 40
Figura 10: Configuración del bloque OPC-UA Read. 42
Figura 11: Configuración OPC-UA Write. 43
Figura 12: Integración de los bloques OPC en el sistema de control. 44
Figura 13: Integración de la perturbación mediante OPC-UA. 45
Figura 14: Ventana de animación con el tanque y la gráfica. 47
Figura 15: Activación del servidor OPC-UA. 49
Figura 16: Creación bloque Global. 50
Figura 17: Principales variables Global. 50
Figura 18: Creación bloque OpcInterface. 51
Figura 19: Configuración OpcInterface. 51
Figura 20: Principales variables OpcInterface. 52
Figura 21: Configuración del periodo de muestreo. 54
Figura 22: Conexionado PID_Compact. 55
Figura 23: Configuración básica PID_Compact. 56
Figura 24: Configuración límites PID_Compact. 57
Figura 25: Parámetros PID_Compact. 58
Figura 26: Función de optimización. 62
Figura 27: Valores resultantes del autoajuste. 63
Figura 28: Diagrama de flujo del programa de operación. 66
Figura 29: Captura de la pantalla inicial. 68
Figura 30: Captura de la pantalla principal de la HMI. 69

Figura 31: Captura de la señalización del valor de perturbación no válido en la pantalla principal. 70
Figura 32: Captura de la pantalla principal del sistema de control. 71
Figura 33: Captura de la pantalla de Información del sistema. 72
Figura 34: Captura de la pantalla de créditos de autoría. 73
Figura 35: Captura de la pantalla de monitorización PID-OPC con evolución de consigna, variable de proceso y salida del regulador. 74
Figura 36: Captura de la configuración de la instancia virtual del PLC en PLCSIM Advanced. 75
Figura 37: Captura de conexión del cliente OPC-UA de Simulink con el servidor del PLC simulado. 76
Figura 38: Señal generada en Simulink. 86
Figura 39: Señal recibida en TIA Portal. 87
Figura 40: Señal generada en TIA Portal. 88
Figura 41: Señal recibida en Simulink. 88
Figura 42: Modelo Simulink para prueba. 90
Figura 43: Captura de prueba realizada en Simulink. 91
Figura 44: Primera fase de la prueba animación. 91
Figura 45: Segunda fase de la prueba animación. 92
Figura 46: Respuesta ante una consigna de 16 unidades en Simulink. 93
Figura 47: Respuesta del sistema con PID_Compact ante una activación inmediata desde la animación Simulink. 94
Figura 48: Respuesta del sistema con PID_Compact ante una activación inmediata desde la monitorización TIA Portal. 95
Figura 49: Respuesta con PID_Compact tras una espera previa, visualizada en Simulink. 96
Figura 50: Respuesta del sistema con PID_Compact tras una breve espera previa a la activación, visualizada en TIA Portal. 97
Figura 51: Respuesta del sistema ante un incremento de consigna de 0 cm a 20 cm, visualizada en TIA Portal. 100
Figura 52: Respuesta del sistema ante un incremento de consigna de 0 cm a 20 cm, visualizada en la animación de Simulink. 101
Figura 53: Respuesta del sistema ante una reducción de consigna de 20 cm a 10 cm, visualizada en TIA Portal. 102
Figura 54: Respuesta del sistema ante una reducción de consigna de 20 cm a 10 cm, visualizada en la animación de Simulink. 102
Figura 55: Respuesta del sistema ante un incremento de consigna de 10 cm a 29 cm, visualizada en TIA Portal. 103
Figura 56: Respuesta del sistema ante un incremento de consigna de 10 cm a 29 cm, visualizada en la animación de Simulink. 104

Figura 57: Respuesta del sistema durante el vaciado ante una reducción de consigna de 29 cm a 0 cm, visualizada en TIA Portal. 105
Figura 58: Respuesta del sistema durante el vaciado ante una reducción de consigna de 29 cm a 0 cm, visualizada en la animación de Simulink. 106
Figura 59: Respuesta del sistema ante una perturbación de −0,9, visualizada en TIA Portal. 107
Figura 60: Respuesta del sistema ante una perturbación de −0,9, visualizada en la animación de Simulink. 107
Figura 61: Respuesta del sistema ante una perturbación de 1,9, visualizada en TIA Portal. 108
Figura 62: Respuesta del sistema ante una perturbación de 1,9, visualizada en la animación de Simulink. 109

Índice de Tablas

Tabla 1: Cronograma de planificación del proyecto. 13
Tabla 2: Resumen de las pruebas de validación realizadas. 79

Capítulo 1 Introducción

La idea principal de este Trabajo Fin de Grado consiste en implementar una estrategia de control distribuido, en la cual se propone el control de una planta virtual (nivel de un depósito de agua) desarrollada en MATLAB/Simulink mediante un controlador PID ejecutado sobre un PLC Siemens, donde ambos elementos se comunican a través del protocolo OPC-UA.

En este capítulo se justifica este enfoque en el marco de la Industria 4.0. A continuación, se detalla el objetivo general y los objetivos específicos del proyecto y se delimitan con precisión todos los aspectos relacionados con el sistema de control en los niveles de simulación, de implementación del control sobre el PLC, así como las comunicaciones OPC-UA.

1.1 Antecedentes y justificación

En el marco de la Industria 4.0, la integración entre sistemas de automatización industrial y modelos virtuales ha adquirido un papel cada vez más relevante en el diseño, análisis y validación de procesos de control. El desarrollo de modelos capaces de reproducir el comportamiento dinámico de una planta permite anticipar su respuesta, comprobar estrategias de regulación y detectar posibles problemas antes de disponer de una instalación física completamente operativa.

Dentro de este contexto, el concepto de gemelo digital hace referencia a la representación virtual de un sistema físico o proceso industrial, empleada para estudiar su funcionamiento, ensayar escenarios de operación y apoyar la toma de decisiones durante las fases de diseño y puesta en marcha. En aplicaciones de control, esta aproximación resulta especialmente útil cuando se pretende validar la interacción entre un controlador industrial y una planta sin asumir los costes, riesgos o limitaciones asociados a las pruebas directas sobre equipos reales.

Tradicionalmente, la puesta en marcha de sistemas de regulación basados en PLC requiere realizar ensayos sobre la propia instalación, lo que puede implicar tiempos elevados de ajuste, indisponibilidad del proceso y posibles riesgos si el comportamiento del controlador no ha sido previamente verificado. Frente a esta situación, herramientas como MATLAB/Simulink, TIA Portal y SIMATIC PLCSIM Advanced permiten desarrollar entornos de validación distribuidos en los que un modelo dinámico de la planta se ejecuta en simulación, mientras que la lógica de control se implementa en un autómata industrial virtualizado.

Desde esta perspectiva, el empleo de entornos virtuales de validación también presenta una contribución indirecta a la sostenibilidad del proceso de ingeniería. En el plano económico, permite reducir tiempos de puesta en marcha, anticipar errores de

configuración y disminuir la necesidad de realizar ensayos preliminares sobre equipos físicos completos. Desde el punto de vista social y operativo, facilita la comprobación del comportamiento del controlador en un entorno seguro, sin exponer a operadores, instalaciones o procesos reales a respuestas no verificadas. Asimismo, aunque el presente trabajo no desarrolla una evaluación ambiental cuantitativa, la sustitución parcial de pruebas físicas por simulaciones puede contribuir a reducir el uso de recursos materiales y energéticos durante las fases iniciales de diseño, ajuste y validación.

Para que esta interacción sea posible, es necesario disponer de un mecanismo de comunicación capaz de intercambiar datos entre plataformas heterogéneas. En este trabajo se emplea OPC-UA como estándar de comunicación industrial, ya que permite establecer un intercambio bidireccional de variables entre Simulink y el PLC configurado en TIA Portal. Gracias a esta comunicación, el modelo virtual del proceso envía la variable de proceso al controlador y recibe de este la señal de actuación necesaria para cerrar el lazo de regulación.

El proyecto se centra en el control del nivel de agua en un depósito, representado mediante un modelo dinámico lineal de primer orden implementado en MATLAB/Simulink. Sobre dicho modelo actúa un regulador industrial PID_Compact configurado en TIA Portal y ejecutado sobre una CPU simulada mediante PLCSIM Advanced. Además, se desarrolla una interfaz HMI para la operación y supervisión del sistema, así como herramientas complementarias de visualización, entre ellas una animación del tanque y una gráfica de evolución del nivel en tiempo real.

Por tanto, el presente Trabajo Fin de Grado se justifica por la necesidad de estudiar y validar una arquitectura de control distribuida que integre simulación dinámica, automatización industrial y comunicación OPC-UA. Este enfoque permite reproducir un entorno representativo de puesta en marcha virtual, analizar el comportamiento del lazo de control ante cambios de consigna y perturbaciones, y demostrar la viabilidad de emplear herramientas de simulación como apoyo al desarrollo de soluciones de automatización industrial.

1.2 Objetivo general

El objetivo general de este Trabajo Fin de Grado es diseñar, implementar y validar una arquitectura de control distribuida entre MATLAB/Simulink y TIA Portal mediante comunicación OPC-UA, aplicada a la regulación del nivel de agua en un depósito virtual. En dicha arquitectura, el modelo dinámico de la planta se ejecuta en Simulink, mientras que el algoritmo de control PID se implementa en un PLC Siemens simulado mediante SIMATIC PLCSIM Advanced.

1.3 Objetivos específicos

Para alcanzar el objetivo general, se establecen los siguientes objetivos específicos:

- Modelado virtual del proceso
  - Definir el modelo matemático del depósito y obtener una expresión adecuada para su implementación en simulación.
  - Desarrollar en MATLAB/Simulink el modelo dinámico del nivel de agua.
  - Configurar el entorno de simulación con criterios temporales coherentes con la ejecución del controlador.
  - Implementar una herramienta gráfica de animación del tanque y una representación en tiempo real de la evolución del nivel.

- Implementación del control en TIA Portal
  - Configurar una CPU Siemens S7-1500 en el entorno de TIA Portal y simular su funcionamiento mediante PLCSIM Advanced.
  - Implementar el regulador de nivel mediante el objeto tecnológico PID_Compact.
  - Definir la ejecución cíclica del controlador con un periodo de muestreo compatible con el modelo de Simulink.
  - Calcular valores iniciales de regulación mediante herramientas desarrolladas en MATLAB y emplear las funciones de ajuste disponibles en TIA Portal para refinar la respuesta del controlador.

- Configuración de la comunicación OPC-UA
  - Activar y configurar el servidor OPC-UA de la CPU simulada.
  - Definir los bloques de datos y variables necesarias para el intercambio de información entre el PLC y Simulink.
  - Configurar en Simulink el cliente OPC-UA y los bloques de lectura y escritura empleados para cerrar el lazo de control.
  - Verificar la correcta transferencia bidireccional de datos mediante pruebas específicas de comunicación.
- Desarrollo de la interfaz HMI
  - Diseñar una interfaz gráfica en TIA Portal para la operación y supervisión del sistema.
  - Permitir el encendido y apagado de la aplicación, la modificación de consignas y la aplicación de perturbaciones.
  - Mostrar las principales variables del lazo de control, incluyendo la referencia, la variable de proceso y la salida del regulador.
  - Incorporar pantallas de información y monitorización que faciliten la interpretación del funcionamiento global del sistema.

- Integración y validación del sistema completo
  - Integrar el modelo virtual, el controlador PID, la comunicación OPC-UA y la HMI en una única arquitectura funcional.
  - Realizar pruebas de seguimiento de consigna en distintos puntos de operación.
  - Evaluar la capacidad del regulador para rechazar perturbaciones introducidas sobre la dinámica de descarga del depósito.

- Comparar la respuesta del sistema controlado íntegramente en Simulink con la obtenida en la arquitectura distribuida Simulink–OPC-UA–TIA Portal.
- Analizar las diferencias de comportamiento derivadas de la discretización, la comunicación entre plataformas y el uso de un regulador industrial ejecutado externamente al modelo.

1.4 Metodología y planificación del proyecto

Para el desarrollo del proyecto se adopta un enfoque metodológico propio, denominado Co-Sim Incremental con OPC-UA, orientado a la integración progresiva entre el entorno de automatización industrial Siemens y el entorno de simulación MATLAB/Simulink. La metodología se basa en un enfoque iterativo de validación incremental, permitiendo verificar cada componente del sistema antes de avanzar hacia niveles superiores de integración. El desarrollo del proyecto se plantea en un período estimado de cinco meses, distribuyendo las actividades de forma progresiva para garantizar la validación incremental de cada fase antes de avanzar hacia la integración completa del sistema.

Mes 1. Preparación del entorno y revisión teórica

Durante el primer mes se realiza la revisión bibliográfica y técnica relacionada con sistemas de control PID, co-simulación industrial, OPC-UA y herramientas Siemens. Paralelamente, se instalan y configuran TIA Portal, SIMATIC PLCSIM Advanced y MATLAB/Simulink. También se establece la red Ethernet virtual y se habilita la comunicación OPC-UA entre el PLC virtual y MATLAB. Al finalizar este mes se debe contar con un entorno completamente operativo y validado mediante pruebas básicas de conectividad y transferencia de datos.

Mes 2. Desarrollo del modelo virtual del depósito

En el segundo mes se desarrolla el modelo matemático y dinámico del depósito en Simulink. Se implementan las ecuaciones diferenciales del sistema hidráulico y se ajustan los parámetros físicos necesarios para representar el comportamiento del tanque. Asimismo, se incorporan herramientas gráficas para visualizar la evolución del nivel de agua y las variables del proceso en tiempo real. Finalmente, se realizan simulaciones en lazo abierto para validar que la respuesta dinámica coincide con el comportamiento teórico esperado.

Mes 3. Diseño e implementación del controlador PID

El tercer mes se dedica al diseño del controlador PID en TIA Portal. Durante esta etapa se configura e integra el regulador PID_Compact dentro del PLC Siemens S7-1500 virtual y se configuran las variables de intercambio mediante OPC-UA. Posteriormente

se ajustan las ganancias del controlador y se realizan pruebas independientes utilizando PLCSIM Advanced para evaluar la estabilidad, el tiempo de respuesta y el error estacionario del sistema de control.

Mes 4. Integración de la co-simulación PLC–Simulink

En el cuarto mes se lleva a cabo la integración completa entre el PLC virtual y el modelo desarrollado en Simulink. Se implementa el intercambio bidireccional de datos mediante OPC-UA y se sincronizan los tiempos de simulación y muestreo. Además, se realizan pruebas continuas de funcionamiento para evaluar la estabilidad de la comunicación, los retardos temporales y la coherencia entre las variables intercambiadas. Esta fase permite validar el funcionamiento global de la arquitectura de co-simulación.

Mes 5. Validación experimental, análisis y documentación final

El quinto mes se orienta a la evaluación experimental del sistema integrado y a la elaboración de la documentación final del proyecto. Se ejecutan distintos escenarios de prueba, incluyendo cambios de referencia y perturbaciones en el sistema, con el objetivo de analizar el desempeño del controlador y la robustez de la co-simulación. Finalmente, se procesan los resultados obtenidos, se elaboran conclusiones técnicas y se redacta la memoria final del proyecto, incluyendo diagramas, resultados experimentales y recomendaciones para trabajos futuros.

1.4.1 Cronograma

Con el fin de sintetizar la planificación temporal descrita, se ha elaborado un cronograma de tareas distribuido en cinco meses. Este cronograma permite representar de forma ordenada la secuencia de desarrollo del proyecto, desde la preparación inicial del entorno hasta la validación final de la arquitectura integrada. La planificación temporal de las actividades se recoge en la Tabla 1.

Tabla 1: Cronograma de planificación del proyecto.

Fase / Actividad principal	Mes 1	Mes 2	Mes 3	Mes 4	Mes 5
Revisión bibliográfica y estudio del estado del arte	X				
Estudio de OPC-UA, co-simulación y herramientas Siemens	X				

Fase / Actividad principal	Mes 1	Mes 2	Mes 3	Mes 4	Mes 5
Instalación y configuración de MATLAB/Simulink, TIA Portal y PLCSIM Advanced	X				
Configuración inicial de la red virtual y del servidor OPC-UA	X				
Definición del modelo matemático del depósito		X			
Implementación del modelo dinámico en MATLAB/Simulink		X			
Configuración de parámetros de simulación, solver y tiempos de muestreo		X			
Desarrollo de la animación y visualización en tiempo real		X			
Configuración del PLC Siemens S7-1500 simulado			X		
Implementación del regulador PID_Compact en TIA Portal			X		
Creación de bloques de datos y variables de intercambio OPC-UA			X		
Cálculo y ajuste inicial de los parámetros del controlador			X		
Integración bidireccional entre Simulink y el PLC mediante OPC-UA				X	
Pruebas de lectura y escritura de variables OPC-UA				X	
Sincronización entre ciclo de PLC, comunicación y simulación				X	
Validación del lazo cerrado distribuido PLC–Simulink				X	

Fase / Actividad principal	Mes 1	Mes 2	Mes 3	Mes 4	Mes 5
Ejecución de pruebas de seguimiento de consigna					X
Ejecución de pruebas ante perturbaciones					X
Análisis e interpretación de resultados					X
Redacción de conclusiones, mejoras y documentación final					X

La planificación presentada en la Tabla 1 refleja una estrategia de desarrollo progresiva, en la que cada fase se apoya en la validación de la etapa anterior. En primer lugar, se establece la base teórica y se prepara el entorno software necesario. Posteriormente, se desarrolla el modelo dinámico de la planta en Simulink y se implementa el controlador industrial en TIA Portal. Una vez verificados ambos elementos por separado, se realiza la integración mediante OPC-UA y se comprueba el intercambio bidireccional de variables entre el PLC virtual y el modelo de simulación. Finalmente, se ejecutan las pruebas de validación del sistema completo, se analizan los resultados obtenidos y se elabora la documentación final del trabajo.

1.5 Alcance del proyecto

El alcance del proyecto comprende el desarrollo y validación de una arquitectura de control distribuida basada en la interconexión entre un entorno de simulación dinámica y un sistema de automatización industrial mediante el estándar OPC-UA. El sistema se compone de un modelo virtual del nivel de agua en un depósito, implementado en MATLAB/Simulink, y de un controlador PID industrial configurado en TIA Portal sobre una CPU Siemens S7-1500 simulada con SIMATIC PLCSIM Advanced.

El trabajo incluye la formulación matemática del proceso, su implementación mediante bloques de Simulink y la configuración de los parámetros de simulación necesarios para operar de forma coordinada con el lazo de control externo. Asimismo, se desarrolla una animación gráfica del depósito junto con una curva en tiempo real de la altura, utilizadas como apoyo visual para la supervisión del comportamiento del modelo.

En el entorno de TIA Portal, el proyecto abarca la configuración del servidor OPC-UA, la creación de bloques de datos específicos para el intercambio de señales y la implementación del regulador PID_Compact. También se desarrolla la lógica de operación asociada al encendido del sistema, la gestión de consignas, la aplicación

temporal de perturbaciones y la validación de los valores introducidos desde la interfaz de usuario.

La comunicación entre Simulink y TIA Portal se realiza mediante un intercambio bidireccional de variables OPC-UA. La planta virtual envía al PLC la altura instantánea del depósito, mientras que el controlador devuelve la señal de actuación que modifica el caudal de entrada del modelo. Esta arquitectura permite cerrar el lazo de regulación entre dos entornos diferentes y reproducir un escenario de validación próximo a una puesta en marcha virtual.

El alcance contempla además el diseño de una interfaz HMI en TIA Portal para la supervisión y operación del sistema. Dicha interfaz permite modificar la consigna, activar o detener la aplicación, introducir perturbaciones y visualizar la evolución de las variables principales del controlador y del proceso.

La validación del sistema se realiza mediante pruebas funcionales sobre la arquitectura integrada. Entre ellas se incluyen ensayos de comunicación OPC-UA, análisis del comportamiento del controlador PID, seguimiento de distintas consignas de altura y respuesta ante perturbaciones en la dinámica de descarga. También se compara el resultado de la regulación ejecutada completamente en Simulink con el obtenido cuando el PID se ejecuta en TIA Portal y la planta se mantiene en Simulink.

En relación con la sostenibilidad, el alcance del trabajo se limita a una valoración cualitativa de los beneficios potenciales asociados al uso de la simulación y la puesta en marcha virtual. No se realiza un cálculo cuantitativo de impacto ambiental, social o económico, ya que el objetivo principal del proyecto se centra en la validación técnica de la comunicación OPC-UA y del lazo de control distribuido. No obstante, la arquitectura desarrollada se considera alineada con criterios de eficiencia en ingeniería, al permitir ensayar estrategias de control en un entorno virtual antes de recurrir a una instalación física, reduciendo potencialmente tiempos de prueba, errores de puesta en marcha, riesgos operativos y consumo de recursos durante las fases iniciales de desarrollo.

El proyecto no contempla la implementación sobre una planta física real ni la validación final con un PLC conectado a un proceso industrial físico. Tampoco se abordan análisis exhaustivos de rendimiento en tiempo real estricto, estudios avanzados de ciberseguridad OPC-UA ni un desarrollo completo de estrategias de control distintas al PID. El enfoque del trabajo es principalmente funcional y demostrativo, orientado a comprobar la viabilidad técnica de la integración entre simulación dinámica, control industrial y comunicación OPC-UA dentro de un entorno de validación virtual.

Capítulo 2 Antecedentes

El presente capítulo establece los fundamentos teóricos y tecnológicos sobre los que se sustenta este Trabajo Fin de Grado, abordando los conceptos clave de la Industria 4.0, el gemelo digital y el protocolo OPC-UA. En primer lugar, se contextualiza la transformación digital en el ámbito industrial, analizando la integración de sistemas OT e IT, los principios de interoperabilidad y estandarización, así como el papel de los datos y la analítica industrial como habilitadores de nuevas funcionalidades. A continuación, se introduce el concepto de gemelo digital, describiendo su arquitectura y su relevancia como herramienta de simulación y optimización de procesos. Posteriormente, se presenta el protocolo OPC-UA, detallando su evolución. Finalmente, se ilustran aplicaciones prácticas de estas tecnologías mediante tres ejemplos.

En conjunto, este capítulo proporciona el marco de referencia necesario para comprender y justificar las decisiones tecnológicas adoptadas en el desarrollo del sistema de control distribuido propuesto en este Trabajo Fin de Grado.

2.1 El contexto de la industria 4.0

La Industria 4.0 se asocia con la digitalización avanzada de la manufactura y con la creación de entornos productivos en los que procesos, equipos y sistemas se conectan de forma inteligente mediante tecnologías de la información y la comunicación. Desde una perspectiva institucional, la Plattform Industrie 4.0 [1] describe este enfoque como la “interconexión inteligente de máquinas y procesos para la industria con ayuda de las tecnologías de la información y la comunicación”, poniendo el énfasis en la conectividad como base para la capacidad de adaptación y la eficiencia operativa.

En Alemania, el término se consolidó como una iniciativa estratégica ligada a la modernización industrial, respaldada por informes de referencia orientados a identificar necesidades de investigación, líneas de acción y barreras de adopción. En particular, el informe final del grupo de trabajo Industrie 4.0 publicado por Forschungsunion y acatech [2] recoge recomendaciones para implementar la iniciativa y documenta la identificación de requisitos y acciones de investigación a medio y largo plazo, lo que contribuyó a estructurar el concepto y su despliegue en el tejido industrial.

2.1.1 Transformación digital e integración de sistemas industriales

La Industria 4.0 se apoya en la integración funcional entre el mundo operativo, conocido habitualmente como OT, y los sistemas de información corporativos, conocidos como IT. Esta convergencia permite que los datos generados por equipos, sensores, controladores y procesos industriales puedan utilizarse para supervisión,

optimización, trazabilidad y toma de decisiones. La guía técnica de National Instruments sobre conexión de entornos de desarrollo a IoT industrial [3] subraya que este paradigma no se limita a incorporar tecnología nueva, sino que enfatiza la integración y el mejor aprovechamiento de tecnologías existentes, habilitando conectividad digital entre industria, máquinas, productos y personas con el objetivo de aumentar la flexibilidad y la capacidad de personalización.

La transición hacia este modelo requiere estrategias y hojas de ruta que conecten objetivos de negocio con decisiones arquitectónicas y tecnológicas, incluyendo la alineación de componentes y restricciones en arquitecturas viables para entornos productivos y la mitigación de riesgos en la adopción de nuevas tecnologías. En este sentido, Siemens [4] plantea la necesidad de definir una hoja de ruta de digitalización que aporte contexto de Industria 4.0 para mejorar procesos y alinear componentes y limitaciones dentro de una arquitectura aplicable en producción, integrando además criterios de valor y viabilidad para priorizar iniciativas.

La conectividad en Industria 4.0 suele describirse en dos direcciones complementarias: integración vertical e integración horizontal. La integración vertical enlaza los niveles de campo (sensores, controladores y máquinas) con sistemas de supervisión y ejecución (por ejemplo, SCADA/MES) y con aplicaciones corporativas (por ejemplo, ERP), extendiéndose a servicios en la nube cuando aporta valor; la integración horizontal articula el intercambio de información entre celdas, líneas, plantas y, potencialmente, socios de la cadena de valor. Esta visión aparece reflejada en descripciones industriales como la de OPC Router [5], donde se enfatiza la automatización del intercambio de datos en procesos de producción interconectados y el enlace “de sensor y controlador a ERP y nube”, así como la conexión entre sistemas de planta y sistemas empresariales.

La relación entre integración vertical e integración horizontal en el contexto de la Industria 4.0 se muestra de forma esquemática en la Figura 1. En ella se representa la conexión vertical entre los distintos niveles de automatización, desde el nivel de campo hasta los sistemas empresariales, así como la integración horizontal entre diferentes líneas o plantas dentro del entorno productivo.

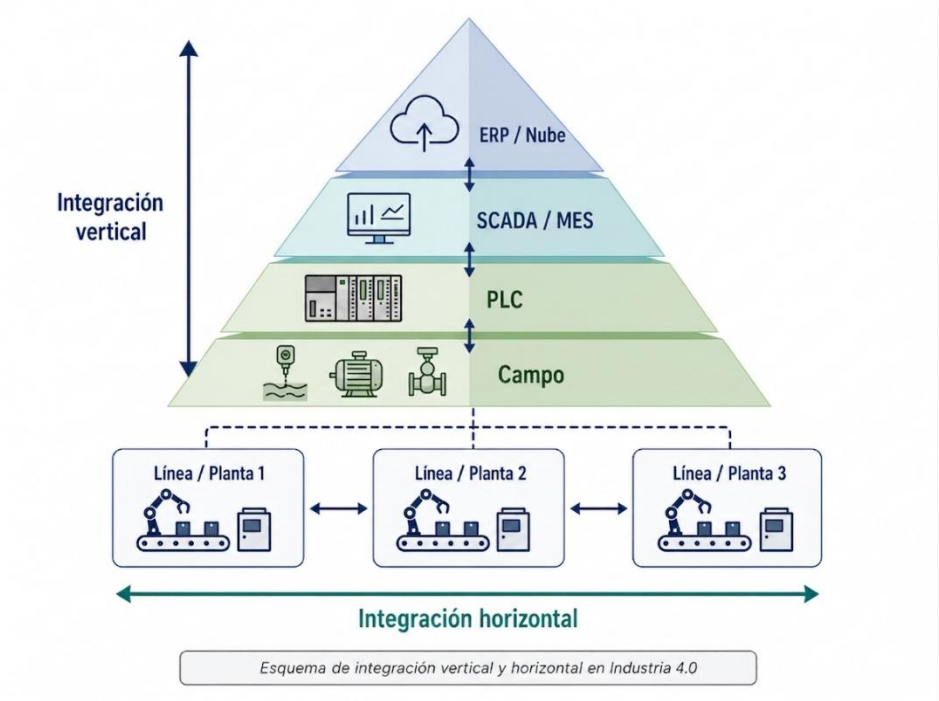


Figura 1: Esquema de integración vertical y horizontal en Industria 4.0. Fuente: elaboración propia a partir de [6].

2.1.2 Interoperabilidad y estandarización

El carácter heterogéneo de los entornos industriales, con múltiples fabricantes, generaciones tecnológicas y requisitos operativos, convierte la interoperabilidad en un principio central para la Industria 4.0. Esta interoperabilidad no solo implica conectividad técnica (comunicación fiable y segura), sino también interoperabilidad semántica (significado coherente de los datos compartidos) para que distintas aplicaciones puedan consumir información de forma consistente y sin transformaciones frágiles. En este marco, la norma de referencia RAMI 4.0 recogida en la DIN SPEC 91345 [7] aporta un modelo de arquitectura que ayuda a estructurar la complejidad, situando activos y componentes a lo largo de capas funcionales, jerarquías y fases del ciclo de vida.

En el plano de las comunicaciones industriales, OPC-UA se considera un estándar ampliamente adoptado para habilitar interoperabilidad entre sistemas heterogéneos. La definición de OPC-UA proporcionada por OPC Foundation [8] lo presenta como un estándar abierto y libre de regalías, disponible además como IEC 62541, y destaca que, más allá del transporte de datos, incorpora un marco de modelado de información

que permite representar datos con modelos semánticos y exponer información definida por fabricantes de manera estandarizada, con capacidad de escalar desde dispositivos hasta sistemas empresariales. En entornos conectados, la seguridad es un requisito determinante, y la documentación de seguridad de OPC-UA en la especificación de OPC Foundation [9] describe mecanismos para alinear el intercambio de información con requisitos de confidencialidad, integridad y autenticación, contemplando tanto arquitecturas cliente/servidor como publicar/suscribir (PubSub).

Complementariamente, la estandarización de la representación digital de los activos se considera un habilitador de la interoperabilidad en Industria 4.0. La International Electrotechnical Commission, mediante IEC 63278-1:2023 [10], define la estructura de la Asset Administration Shell (AAS) como una representación digital estandarizada de un activo que proporciona acceso uniforme a información y servicios, con el propósito de permitir que dos o más aplicaciones intercambien información y la utilicen de forma mutuamente consistente y segura. En coherencia con esta visión, la Industrial Digital Twin Association [11] describe la AAS como un soporte para el intercambio significativo de información sobre activos y componentes de Industria 4.0 entre socios de una red de creación de valor, señalando su orientación hacia implementaciones interoperables del gemelo digital.

2.1.3 Datos y analítica industrial

La Industria 4.0 es, en la práctica, un paradigma orientado a datos: la mejora de procesos y la automatización avanzada dependen de la capacidad de capturar datos fiables, transportarlos con garantías, integrarlos entre dominios y explotarlos para tomar decisiones. La guía de National Instruments sobre Industrial IoT [3] destaca el papel de arquitecturas que conectan entornos de adquisición y control con sistemas de TI y servicios en la nube, apoyando casos de uso como optimización del flujo de producción, supervisión avanzada y analítica.

En este contexto, el dato industrial adquiere valor cuando puede utilizarse de forma estructurada para supervisar el estado del proceso, detectar desviaciones, analizar tendencias y apoyar decisiones de operación o mantenimiento. Para ello, no basta con registrar variables de campo, sino que resulta necesario garantizar su disponibilidad, coherencia y trazabilidad dentro de la arquitectura de automatización. De este modo, los datos generados por sensores, controladores y sistemas de supervisión pueden transformarse en información útil para mejorar el rendimiento del proceso y reducir incertidumbres durante la operación.

La relación entre conectividad, datos, semántica y arquitectura refuerza una idea clave: el valor del dato industrial no depende solo de su captación, sino también de su contextualización mediante modelos de información, su trazabilidad a lo largo del ciclo de vida y su disponibilidad para diferentes aplicaciones y niveles, aspectos que el

modelo RAMI 4.0 documentado en la DIN SPEC 91345 [7] ayuda a estructurar al situar activos y componentes dentro de una arquitectura de referencia común.

Por tanto, los datos y la analítica industrial constituyen una base fundamental para la Industria 4.0, ya que permiten conectar el nivel físico de la planta con herramientas de supervisión, diagnóstico y optimización. A partir de esta base se desarrollan aplicaciones más avanzadas, como la simulación conectada, la puesta en marcha virtual o los modelos digitales del proceso, que se tratan con mayor detalle en el apartado dedicado al gemelo digital.

2.2 El gemelo digital

El gemelo digital constituye una de las aplicaciones más representativas de la digitalización industrial, ya que permite conectar el comportamiento de un sistema físico con una representación virtual capaz de reproducir, analizar y anticipar su funcionamiento. En el contexto de este trabajo, este concepto resulta especialmente relevante porque la planta física se sustituye por un modelo desarrollado en MATLAB/Simulink, sobre el que se prueba un controlador industrial implementado en TIA Portal. Por ello, antes de abordar la comunicación OPC-UA y la implementación del sistema, es necesario introducir el concepto de gemelo digital, su arquitectura general y los principales marcos de estandarización asociados.

2.2.1 Definición, arquitectura y estandarización

El concepto de gemelo digital se fundamenta en la creación de una representación virtual de un activo físico que permite analizar su comportamiento, simular escenarios y optimizar su funcionamiento a partir de datos reales. La página corporativa de Siemens [12] describe los gemelos digitales como representaciones virtuales de productos, procesos o rendimiento que conectan distintas etapas del ciclo de vida mediante datos, lo que permite mejorar la eficiencia y reducir fallos. Del mismo modo, la sección de preguntas frecuentes de Siemens [13] introduce la relación con el denominado “hilo digital”, entendido como la continuidad de datos y procesos a lo largo del ciclo de vida, donde el gemelo digital actúa como una entidad operativa dentro de ese flujo de información.

Desde una perspectiva más técnica, el artículo divulgativo de National Instruments [14] define el gemelo digital como un modelo virtual que refleja un objeto físico instrumentado mediante sensores, cuyos datos de funcionamiento, como temperatura o consumo energético, se utilizan para alimentar la copia digital y realizar simulaciones orientadas a la mejora del sistema. Esta visión se refuerza en el caso técnico de National Instruments [15], donde se describe el gemelo digital como una réplica virtual estrechamente vinculada a su equivalente físico, destacando que la sincronización mediante datos en tiempo real es esencial para que el modelo digital reproduzca con

precisión los efectos del sistema real y pueda ajustarse mediante procesos de validación.

La estructura general de un gemelo digital aplicado a un sistema industrial se muestra en la Figura 2. En ella se representan los tres elementos principales de la arquitectura: el activo físico o planta, la comunicación e intercambio de datos, y el modelo virtual o gemelo digital. Las flechas bidireccionales reflejan la sincronización continua entre el sistema físico y su representación digital.

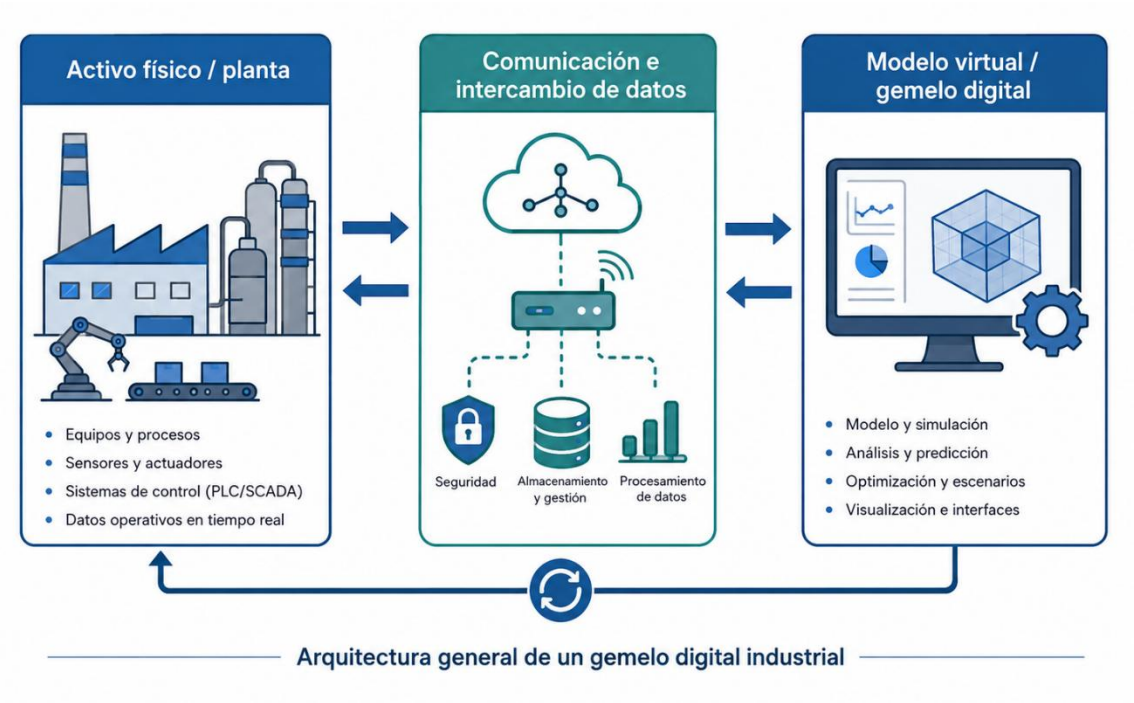


Figura 2: Arquitectura general de un gemelo digital industrial. Fuente: elaboración propia a partir de [16].

La necesidad de estructurar y normalizar este concepto ha dado lugar al desarrollo de marcos de referencia específicos. La norma ISO 23247-1 [17] establece un marco general para la implementación de gemelos digitales en entornos de fabricación, definiendo principios, términos y requisitos que permiten describir de manera homogénea los distintos componentes implicados. En paralelo, la norma IEC 63278-1 [10] introduce el concepto de Asset Administration Shell como una representación digital estandarizada de un activo, diseñada para proporcionar acceso uniforme a información y servicios y facilitar el intercambio de datos entre aplicaciones de forma coherente y segura.

En el ámbito de la estandarización aplicada, la Industrial Digital Twin Association desempeña un papel relevante en la definición de modelos interoperables. La página de especificaciones de la IDTA [18] indica que la Asset Administration Shell define la estructura software, las interfaces y la semántica necesarias para construir gemelos digitales estandarizados, lo que permite superar limitaciones de integraciones

específicas entre sistemas. De forma complementaria, el preámbulo de la especificación de la IDTA [19] establece como objetivo principal habilitar el intercambio significativo de información entre activos y sistemas dentro de redes de valor industriales, reforzando la interoperabilidad entre organizaciones.

En conjunto, el gemelo digital se configura como un elemento clave dentro de la Industria 4.0, al integrar datos, modelos y capacidades de simulación en una representación coherente del sistema físico. Su valor no reside únicamente en la visualización del activo, sino en la capacidad de establecer una conexión continua con el sistema real, aplicar análisis avanzados y facilitar la toma de decisiones basada en datos, todo ello apoyado en marcos de estandarización que garantizan interoperabilidad y escalabilidad.

2.3 El protocolo OPC

El protocolo OPC es un estándar de comunicación industrial cuyo objetivo principal es permitir que diferentes dispositivos, programas y sistemas intercambien datos de forma sencilla y uniforme, independientemente del fabricante. En términos generales, OPC actúa como un “lenguaje común” que facilita la comunicación entre elementos de un sistema industrial, evitando la necesidad de desarrollar conexiones específicas para cada caso.

2.3.1 Evolución a OPC-UA y principios de diseño

El protocolo OPC surge como una solución para facilitar la interoperabilidad entre sistemas industriales mediante un estándar común de comunicación. El folleto técnico de OPC Foundation [20] describe el origen de OPC como “OLE for Process Control”, basado en tecnologías COM/DCOM de Microsoft, lo que permitió inicialmente la integración entre aplicaciones industriales, aunque con una fuerte dependencia del entorno Windows. Esta limitación motivó la evolución hacia una nueva arquitectura más flexible y abierta.

En este contexto, OPC-UA se desarrolla como una arquitectura independiente de plataforma y lenguaje de programación. El mismo documento de OPC Foundation [20] explica que OPC-UA se diseñó para superar las restricciones del modelo anterior, permitiendo su implementación en diferentes sistemas operativos y entornos de ejecución. Esta transición se consolida con su estandarización como norma IEC 62541, lo que refuerza su adopción como referencia en comunicaciones industriales. De forma coherente, la página de OPC Router [21] describe esta evolución como el paso de tecnologías dependientes de COM/DCOM a arquitecturas basadas en TCP/IP y otros protocolos modernos, destacando la independencia de plataforma y la incorporación de modelos semánticos como avances clave.

La evolución desde OPC clásico hacia OPC-UA se muestra de forma resumida en la Figura 3.

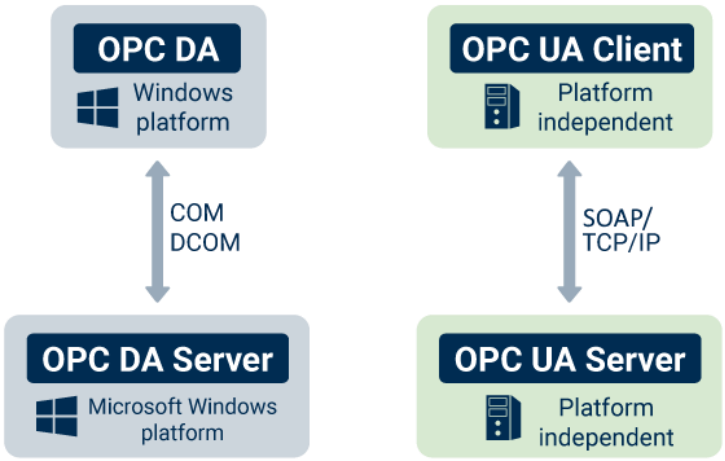


Figura 3: Evolución de OPC clásico a OPC-UA. Fuente: adaptado de [21].

Una de las principales características de OPC-UA es su modelo de información estructurado. La documentación de IEC 62541-3 [22] define el AddressSpace como un espacio donde se organizan los datos del sistema de forma estructurada, similar a un árbol de información. Por su parte, IEC 62541-5 [23] describe cómo se definen los distintos elementos dentro de ese espacio y cómo se relacionan entre sí. Esto permite que los datos intercambiados no solo tengan un valor, sino también un contexto, facilitando su interpretación por diferentes sistemas.

2.3.2 Modelado de información, especificaciones complementarias, PubSub y seguridad

OPC-UA no se limita a transmitir datos, sino que también permite describirlos de forma estructurada. El documento de OPC Foundation [20] define OPC-UA como un estándar de intercambio de información seguro y fiable, capaz de representar distintos tipos de información como variables de proceso, alarmas, históricos o incluso funciones. Esta información se organiza en el AddressSpace, donde cada elemento, denominado nodo, tiene un significado concreto, tal y como se detalla en la referencia técnica MDIS [24].

La estructura básica de comunicación y modelado de información en OPC-UA se muestra en la Figura 4. En ella se representa cómo el servidor OPC-UA expone la información del sistema mediante un espacio de direcciones compuesto por nodos, que pueden ser consultados o modificados por clientes OPC-UA dentro de una arquitectura cliente-servidor.

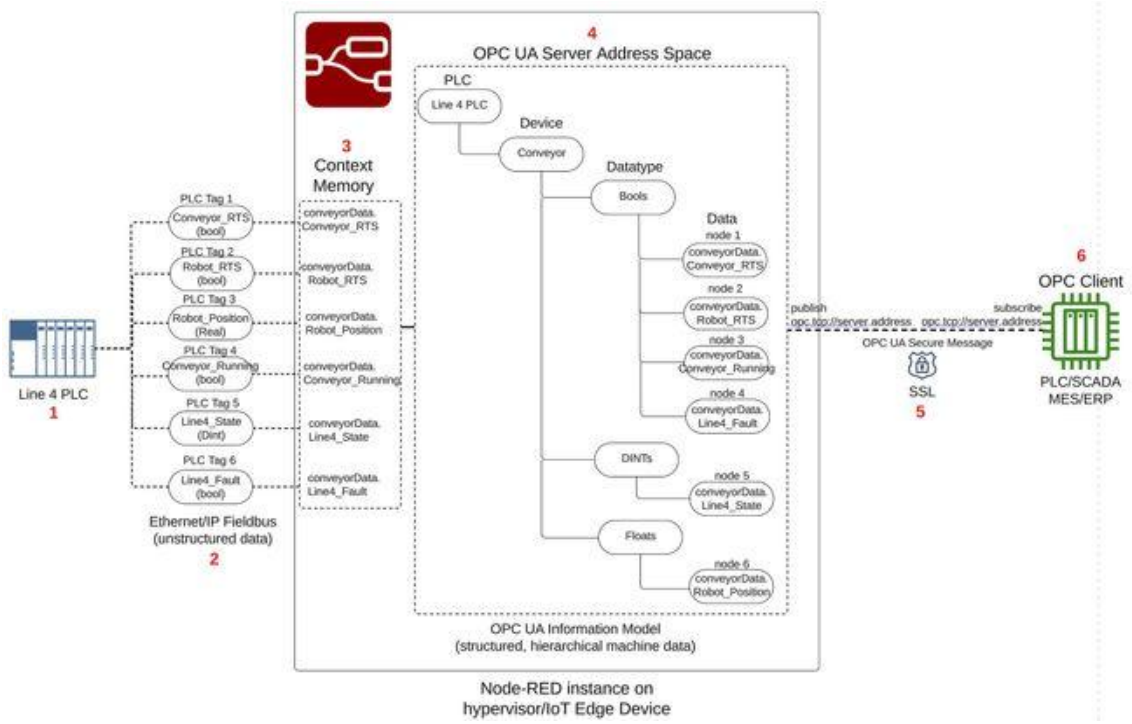


Figura 4: Esquema básico del espacio de direcciones de un servidor OPC-UA. Fuente: adaptado de [25].

Para adaptar este modelo a distintos sectores industriales, OPC-UA introduce las denominadas especificaciones complementarias, conocidas en la documentación técnica como Companion Specifications. La página de OPC Foundation [26] explica que estas especificaciones permiten definir modelos de información específicos para cada ámbito, como energía, automatización o robótica. En esta misma línea, la guía de OPC Foundation [27] describe estas especificaciones como una forma de estandarizar la estructura y el significado de los datos, evitando que cada sistema utilice formatos diferentes.

Además del modelo clásico cliente/servidor, en el que un sistema solicita información a otro, OPC-UA incorpora el modelo publicar/suscribir (PubSub). La especificación OPC-UA Part 14 [28] describe este enfoque como un sistema en el que los dispositivos pueden enviar información sin necesidad de que otro sistema la solicite directamente, permitiendo que múltiples sistemas reciban esos datos al mismo tiempo. Esto facilita la escalabilidad y simplifica la comunicación en sistemas complejos.

En cuanto a la seguridad, OPC-UA incorpora mecanismos integrados para proteger la información. La especificación Part 2 [29] define el uso de canales seguros que garantizan la confidencialidad (los datos no pueden ser leídos por terceros), la integridad (los datos no se modifican durante la transmisión) y la autenticación (se verifica la identidad de los sistemas que se comunican). Esto se consigue mediante el uso de certificados digitales y técnicas de cifrado, permitiendo un intercambio de información fiable incluso en entornos distribuidos.

En aplicaciones reales, OPC-UA se utiliza como elemento de conexión entre sistemas industriales y sistemas de gestión. La página de OPC Router [30] muestra ejemplos en los que OPC-UA permite integrar datos de planta con bases de datos o sistemas empresariales, facilitando el flujo de información en toda la organización. De forma complementaria, la plataforma de monitorización de Paessler [31] destaca la importancia de supervisar estos sistemas interconectados y señala el uso de protocolos como OPC-UA como base para garantizar una comunicación eficiente y mantenible en entornos industriales.

2.4 Ejemplos de aplicaciones

Para complementar las secciones anteriores del presente capítulo, esta sección presenta tres ejemplos que ilustran la aplicación práctica de OPC-UA y el gemelo digital en entornos industriales. El primero, basado en una aplicación de puesta en marcha virtual, muestra el uso del gemelo digital para validar la interacción entre el sistema modelado y el programa de control antes de disponer del equipo físico. El segundo, basado en casos de inray con OPC Router, muestra la integración estandarizada de datos entre planta y sistemas superiores, como bases SQL, MongoDB y MES, para trazabilidad y operación en tiempo real. El tercero, extraído de Paessler y National Instruments, aborda la monitorización de servidores OPC-UA y el mantenimiento predictivo mediante gemelos digitales, destacando la convergencia IT/OT y la detección proactiva de fallos.

2.4.1 Puesta en marcha virtual con gemelo digital

El artículo de Siemens Tecnomatix [32] describe la puesta en marcha virtual (virtual commissioning) como un enfoque que conecta el gemelo digital de la máquina o sistema con el PLC mediante los mismos programas de control destinados al “gemelo físico”, permitiendo observar la respuesta del sistema, identificar carencias y optimizar el diseño antes de disponer del equipo real. En ese mismo texto, Siemens Tecnomatix destaca que esta aproximación permite probar y refinar, en un entorno virtual, la integración entre diseños mecánicos, eléctricos y lógicos antes del ensamblaje en planta, reduciendo el riesgo de retrasos y paradas asociadas a correcciones tardías. Esta lógica encaja con el planteamiento de Siemens [33], que sitúa el valor del gemelo digital en la simulación y optimización previa a la inversión en prototipos o activos, y en la mejora progresiva de precisión mediante datos operacionales y algoritmos orientados a optimización y mantenimiento predictivo.

2.4.2 Integración de datos y trazabilidad con OPC-UA

Los casos de uso publicados por inray en la página de OPC Router [34] ilustran aplicaciones donde OPC-UA se emplea como vía de acceso estandarizada a datos industriales para automatizar flujos entre planta y sistemas superiores. En un ejemplo del sector alimentario, la descripción de inray indica que datos de producción de controladores S7 se obtienen mediante un servidor OPC-UA y se utilizan, a través de OPC Router, para conectar con un sistema de gestión, almacenando simultáneamente datos en una base SQL con un proceso de comunicación automatizado. En el ámbito de impresión y embalaje, inray describe un caso en el que OPC Router se conecta a un servidor OPC-UA, convierte datos a JSON y los envía a MongoDB para suministrar datos operacionales en tiempo real a un MES y a un sistema de análisis, señalando además mejoras en mantenimiento, resolución de incidencias y aseguramiento de calidad a partir del registro de parámetros de proceso.

La página de OPC Foundation dedicada a Companion Specifications [26] explica que los modelos de información se apoyan en una infraestructura común (descubrimiento, transporte, acceso a información y seguridad/robustez) y que las Companion Specifications suelen constituir “modelos estándar de industria” porque abordan problemas específicos de un sector, de modo que la interoperabilidad se desplaza del mero transporte al nivel semántico. En escenarios de integración con múltiples consumidores de datos (MES, historiadores, analítica y servicios de gemelo digital), la especificación OPC-UA PubSub [28] define mecanismos de publicación/suscripción basados en DataSetMessages y NetworkMessages y explicita que la seguridad de mensaje puede configurarse con firma y cifrado, proporcionando protección extremo a extremo del payload. De forma complementaria, la arquitectura de seguridad descrita en OPC-UA Part 2 [29] atribuye al SecureChannel funciones de cifrado para confidencialidad, firmas para integridad y certificados para autenticación de aplicaciones, lo que resulta relevante cuando el intercambio de datos de planta alimenta procesos de trazabilidad y decisión con requisitos de confianza.

2.4.3 Monitorización y mantenimiento predictivo

La página de Paessler sobre monitorización OPC-UA [31] presenta un caso de aplicación centrado en mantener la salud y disponibilidad de servidores y clientes OPC-UA y en detectar errores tempranamente en procesos de fabricación, apoyándose en alertas personalizadas y visualización para identificar y prevenir problemas en redes OT. En esa misma fuente, Paessler [31] relaciona este enfoque con la convergencia IT/OT y con la necesidad de una visión combinada de redes IT y OT, y vincula la monitorización de componentes OT con el objetivo de implementar mantenimiento predictivo y un enfoque proactivo de resolución de incidencias.

En el plano del gemelo digital aplicado al mantenimiento, la página de National Instruments [35] incluye un ejemplo de uso donde datos en tiempo real de gemelos digitales, combinados con monitorización de salud de equipos, permiten predecir fallos

y reducir tiempos de inactividad, reforzando el valor práctico de conectar modelos digitales con señales operacionales. En conjunto, estas aplicaciones ponen de manifiesto que la utilidad del gemelo digital en producción se materializa cuando existe captura e integración sistemática de datos y cuando la comunicación industrial aporta interoperabilidad y seguridad, aspectos ejemplificados por las capacidades de observabilidad y alertado descritas por Paessler [31] y por los mecanismos de seguridad normalizados en OPC-UA Part 2 [29].

Capítulo 3 Desarrollo e Implementación

En este capítulo se presenta el desarrollo e implementación completa del sistema de control distribuido, desde la definición del caso de estudio hasta la integración final de todos los componentes. El capítulo comienza con el planteamiento del caso de estudio del depósito, donde se revisan tres enfoques de modelado y se justifica la elección del modelo lineal de primer orden adoptado. A continuación, se describe la implementación del modelo virtual en MATLAB/Simulink, incluyendo la configuración del cliente OPC-UA, los bloques de lectura y escritura, y el sistema de animación desarrollado. Posteriormente, se detalla la implementación del regulador PID en TIA Portal, abarcando la selección del PLC, la configuración del servidor OPC-UA, la parametrización del objeto tecnológico PID_Compact y el cálculo de los parámetros del controlador. Finalmente, se presenta la implementación funcional completa, incluyendo la lógica de operación desarrollada en el PLC y la interfaz HMI para la supervisión y control del sistema.

3.1 Planteamiento del caso de estudio

Para definir el caso de estudio se han revisado diferentes enfoques de modelado aplicables a la dinámica de un depósito de nivel. La memoria de proyecto disponible en un repositorio de GitHub sobre el modelo de una planta de depósito de agua [36] presenta una formulación lineal de primer orden, en la que la evolución del nivel se obtiene a partir del caudal de entrada, la sección transversal del tanque y una resistencia hidráulica equivalente asociada a la salida. Este planteamiento resulta adecuado para el desarrollo de sistemas de control, ya que proporciona una dinámica sencilla, interpretable y directamente implementable en MATLAB/Simulink.

Por otra parte, el artículo web de Control Automático Educación [37] desarrolla el modelado matemático de un tanque de nivel considerando una válvula de entrada y una válvula de salida. En este caso, el sistema se obtiene a partir de un balance de masa y de una descarga dependiente de la raíz cuadrada de la altura. Además, el artículo aborda el cálculo del punto de equilibrio, la linealización mediante series de Taylor y la obtención de funciones de transferencia, por lo que constituye un enfoque más completo desde el punto de vista del análisis de sistemas de control.

Finalmente, la página web de la Universidad del País Vasco [38] explica el proceso de llenado y vaciado de un depósito abierto a partir de la conservación de volumen y de la ecuación de Bernoulli. Este enfoque permite comprender con mayor detalle la base física de la descarga gravitatoria, aunque conduce a una formulación más compleja que la necesaria para el objetivo principal de este trabajo.

De los tres planteamientos anteriores, el modelo adoptado en este proyecto es el descrito en la memoria de proyecto disponible en GitHub [36]. Esta elección responde

a varios motivos. En primer lugar, se trata de un modelo suficientemente representativo de la dinámica de un depósito, ya que conserva los fenómenos esenciales de acumulación de líquido, entrada de caudal y vaciado. En segundo lugar, su formulación lineal de primer orden facilita tanto la implementación en Simulink como el posterior diseño y análisis del control de nivel. Además, el propio documento de referencia emplea este modelo como base para la incorporación de un controlador PID, lo que lo hace especialmente coherente con el objetivo de este trabajo, centrado en la interconexión entre una planta virtual desarrollada en MATLAB/Simulink y un controlador implementado en TIA Portal mediante comunicación OPC-UA.

Los modelos descritos en el artículo web [37] y en la página web [38] se consideran referencias complementarias. Ambos permiten contextualizar otras formas de representar la dinámica de un depósito, pero introducen una complejidad adicional que no resulta imprescindible para validar la arquitectura propuesta. En consecuencia, sus ecuaciones se presentan únicamente de forma resumida, y la formulación matemática se centra principalmente en el modelo lineal finalmente adoptado.

3.1.1 Descripción física del sistema seleccionado

El sistema seleccionado consiste en un depósito de sección transversal constante A, en el que el nivel de líquido viene descrito por la variable $h(t)$, que varía con el tiempo desde una altura inicial $h(0) = h_0$. El tanque recibe un caudal de entrada $Q_{in}(t)$, mientras que el líquido abandona el depósito a través de una salida cuya oposición al flujo se modela mediante una resistencia hidráulica equivalente R.

La estructura física simplificada del depósito utilizado como caso de estudio se muestra en la Figura 5.

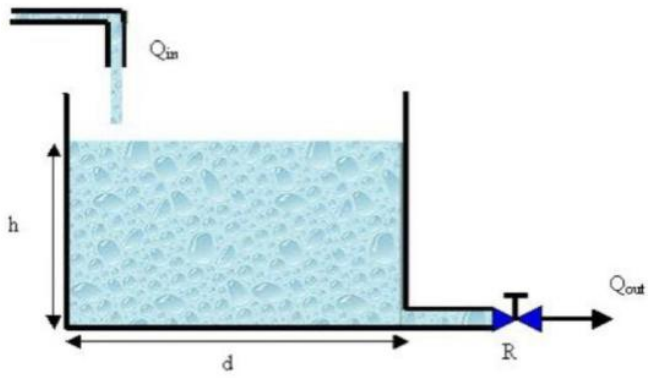


Figura 5: Estructura física simplificada del depósito utilizado como caso de estudio. Fuente: adaptado de [36].

Desde un punto de vista físico, el depósito actúa como un elemento de almacenamiento. Cuando el caudal que entra es mayor que el caudal que sale, el nivel aumenta; cuando la descarga supera a la entrada, el nivel disminuye; y cuando ambos caudales se equilibran, la altura permanece constante. El objetivo del modelo consiste, por tanto, en describir matemáticamente la evolución temporal de $h(t)$ a partir de este balance entre entrada y salida.

Hipótesis de modelado

Para establecer el modelo se adoptan varias hipótesis simplificadoras. En primer lugar, se considera que el depósito posee una sección transversal constante A, de manera que el volumen almacenado depende linealmente de la altura del líquido. Asimismo, se supone que la salida puede representarse mediante una resistencia hidráulica equivalente R, lo que permite expresar el caudal de descarga como una relación lineal con el nivel [36].

También se considera que los parámetros A y R permanecen constantes durante la operación del sistema y que ambos toman valores positivos. Esta última condición resulta necesaria para que el modelo sea físicamente coherente y matemáticamente válido, ya que valores nulos de área o resistencia impedirían representar correctamente la dinámica del depósito [36].

Estas hipótesis permiten obtener un modelo lineal de primer orden que conserva la interpretación física básica del proceso y, al mismo tiempo, resulta sencillo de implementar mediante bloques elementales de Simulink.

Formulación matemática del modelo adoptado

La deducción del modelo parte de la relación entre el volumen de líquido almacenado y la altura alcanzada en el depósito. Dado que la sección transversal del tanque es constante, dicha relación se expresa mediante la ecuación (1):

$$V(t) = A\,h(t) \tag{1}$$

donde $V(t)$ representa el volumen de líquido contenido en el depósito, A es el área de su sección transversal y $h(t)$ es la altura del nivel en cada instante.

A partir de la ecuación anterior, la tasa de variación del volumen almacenado se obtiene derivando respecto al tiempo. Esta relación se recoge en la ecuación (2):

$$\frac{dV(t)}{dt} = A\frac{dh(t)}{dt} \tag{2}$$

La variación temporal del volumen debe coincidir con la diferencia entre el caudal de entrada y el caudal de salida del depósito. Este balance de caudales se formula mediante la ecuación (3):

$$A\frac{dh(t)}{dt} = Q_{in}(t) - Q_{out}(t) \tag{3}$$

donde $Q_{in}(t)$ representa el caudal de entrada y $Q_{out}(t)$ el caudal de salida.

En el modelo adoptado, la salida del depósito se representa mediante una resistencia hidráulica equivalente. Bajo esta aproximación, el caudal de descarga es proporcional a la altura del líquido e inversamente proporcional a la resistencia al flujo. Esta relación se expresa mediante la ecuación (4):

$$Q_{out}(t) = \frac{h(t)}{R} \tag{4}$$

Sustituyendo la ecuación (4) en el balance de caudales definido en la ecuación (3), se obtiene la ecuación dinámica intermedia del sistema, mostrada en la ecuación (5):

$$A\frac{dh(t)}{dt} = Q_{in}(t) - \frac{h(t)}{R} \tag{5}$$

Finalmente, al despejar la derivada temporal de la altura en la ecuación anterior, se obtiene la ecuación diferencial que define el modelo dinámico adoptado en este trabajo. Dicha expresión se presenta en la ecuación (6):

$$\frac{dh(t)}{dt} = \frac{1}{A}Q_{in}(t) - \frac{1}{RA}h(t) \tag{6}$$

La ecuación (6) constituye la expresión fundamental del caso de estudio. El primer término, $\frac{1}{A}Q_{in}(t)$, representa el efecto del caudal de entrada sobre el incremento del nivel. El segundo término, $-\frac{1}{RA}h(t)$, describe la disminución del nivel provocada por la descarga del depósito. La dinámica queda así definida por la competencia entre el aporte de líquido y la pérdida a través de la salida.

La ecuación anterior también permite interpretar de forma directa el papel de los parámetros del sistema. Un área A mayor implica que, para un mismo caudal de entrada, la altura varía más lentamente. Por otra parte, una resistencia R elevada reduce el efecto del vaciado, haciendo que el tanque conserve el líquido durante más tiempo. Esta relación entre parámetros y comportamiento dinámico resulta especialmente útil para la posterior implementación y análisis del sistema en simulación.

Interpretación física del estado estacionario

El estado estacionario se alcanza cuando el nivel deja de variar con el tiempo. Matemáticamente, esta condición se establece imponiendo que la derivada de la altura sea nula, tal como se indica en la ecuación (7):

$$\frac{dh(t)}{dt} = 0 \tag{7}$$

Al aplicar la condición de régimen estacionario de la ecuación (7) sobre la ecuación dinámica del modelo definida en la ecuación (6), se obtiene la relación mostrada en la ecuación (8):

$$0 = \frac{1}{A}Q_{in} - \frac{1}{RA}h_{ss} \tag{8}$$

donde h_{ss} representa la altura alcanzada por el líquido en estado estacionario.

Despejando h_{ss} en la ecuación anterior se obtiene la expresión del nivel de equilibrio, recogida en la ecuación (9):

$$h_{ss} = R \cdot Q_{in} \tag{9}$$

La ecuación (9) indica que, para un caudal de entrada constante, el nivel final del depósito depende directamente de la resistencia hidráulica de salida. Desde el punto de vista físico, el estado estacionario aparece cuando el caudal que entra en el tanque queda exactamente compensado por el caudal que abandona el sistema. En ese instante, el depósito deja de acumular o perder líquido y la altura permanece constante.

Esta interpretación resulta relevante para el desarrollo posterior del trabajo, ya que el sistema de control deberá modificar el caudal de entrada para conducir el nivel hacia una consigna determinada y mantenerlo estable ante cambios en la referencia o en las condiciones de operación.

3.1.2 Análisis de los modelos alternativos revisados

Modelo físico de llenado y vaciado basado en Bernoulli

Aunque el modelo adoptado en este trabajo es el modelo lineal de primer orden descrito en la memoria de proyecto disponible en GitHub [36], también se han revisado otros dos enfoques de modelado con el objetivo de contextualizar la elección realizada.

Por un lado, la página web de la Universidad del País Vasco [38] presenta un modelo físico de llenado y vaciado basado en la conservación de volumen y en la ecuación de Bernoulli. En dicho planteamiento se considera un depósito abierto, alimentado por un caudal de entrada Q, y con una descarga a través de un orificio inferior de sección a. Este modelo permite describir con mayor detalle la relación entre la altura del líquido y la velocidad de salida, teniendo en cuenta la energía asociada a la columna de fluido.

La expresión final obtenida en este enfoque conduce a una ecuación dinámica no lineal, recogida en la ecuación (10):

$$\left(\frac{A^2}{a^2} - 1\right)\left(\frac{dh}{dt}\right)^2 - 2Q\frac{A}{a^2}\frac{dh}{dt} + \frac{Q^2}{a^2} - 2gh = 0 \tag{10}$$

La ecuación (10) muestra que la dinámica del depósito queda ligada de forma explícita a la descarga gravitatoria y a la geometría del orificio de salida. Sin embargo, también

evidencia una mayor complejidad matemática, ya que la derivada de la altura aparece elevada al cuadrado. Por este motivo, aunque el modelo resulta útil para comprender la base física del vaciado, no se considera el más adecuado para la implementación principal de este trabajo.

Por otro lado, el artículo web de Control Automático Educación [37] desarrolla un modelo no lineal de un tanque de nivel con válvula de entrada y válvula de salida. En este caso, la dinámica se obtiene a partir de un balance de masa, considerando las aperturas de ambas válvulas y constantes hidráulicas asociadas al flujo. La ecuación característica de este modelo se presenta en la ecuación (11):

$$A\frac{dH}{dt} = k_1 a_1 - k_2 a_2 \sqrt{2gH} \tag{11}$$

donde A representa el área transversal del tanque, H la altura del líquido, a_1 y a_2 las aperturas de las válvulas de entrada y salida, respectivamente, y k_1 y k_2 las constantes hidráulicas asociadas a dichas válvulas.

La ecuación (11) destaca por incorporar el término $\sqrt{H}$, que introduce una relación no lineal entre la altura del depósito y el caudal de descarga. A partir de este modelo, el artículo desarrolla el cálculo del punto de equilibrio y la posterior linealización mediante series de Taylor, con el objetivo de obtener funciones de transferencia útiles para el análisis y diseño de controladores.

Ambos enfoques permiten representar el comportamiento de un depósito con mayor detalle físico que el modelo finalmente seleccionado. No obstante, también requieren una formulación más compleja, un mayor número de parámetros y, en el caso del modelo con válvulas, una etapa adicional de linealización alrededor de un punto de operación. En comparación, el modelo lineal de primer orden descrito en la memoria de proyecto disponible en GitHub [36] proporciona una dinámica más sencilla, directamente implementable en Simulink y suficiente para validar la arquitectura de control distribuido planteada en este trabajo.

3.1.3 Justificación final del modelo seleccionado

A la vista de los enfoques revisados, el modelo lineal de primer orden descrito en la memoria de proyecto disponible en GitHub [36] se considera el más adecuado para el desarrollo del presente trabajo. El artículo web [37] aporta una formulación no lineal orientada al análisis de control y a la obtención de funciones de transferencia linealizadas. Por su parte, el recurso web [38] permite describir con mayor detalle la física del llenado y vaciado de un depósito abierto. Sin embargo, ambos enfoques presentan una complejidad superior a la necesaria para el objetivo central de este proyecto.

El propósito del caso de estudio no es reproducir con el máximo grado de fidelidad todos los fenómenos hidráulicos de un depósito real, sino disponer de una planta dinámica representativa, estable, interpretable y adecuada para validar la comunicación bidireccional entre MATLAB/Simulink y un PLC Siemens mediante OPC-UA, así como para implementar un control de nivel sobre dicha planta. Desde esta perspectiva, el modelo lineal de la memoria de proyecto disponible en GitHub [36] ofrece una solución especialmente conveniente, ya que combina una formulación matemática clara, una implementación directa en Simulink y una relación inmediata con el diseño posterior del controlador PID.

Por todo ello, el modelo adoptado para el desarrollo del sistema será el descrito en la memoria de proyecto disponible en GitHub [36], mientras que los planteamientos del artículo web [37] y del recurso web [38] se mantienen como referencias complementarias para contextualizar la elección realizada y mostrar otras posibles formas de abordar el modelado de un depósito de nivel.

3.2 Modelado virtual en MATLAB/Simulink

MATLAB y Simulink se han empleado como entorno principal para el modelado virtual de la planta, con el objetivo de disponer de una representación dinámica del proceso sobre la que poder desarrollar y validar posteriormente el sistema de control. En este trabajo, Simulink no se utiliza únicamente como herramienta de simulación matemática, sino como el entorno en el que se ejecuta el modelo virtual del depósito, que intercambia información en tiempo real con el regulador implementado en TIA Portal mediante comunicación OPC-UA. De este modo, el modelo desarrollado actúa como una planta virtual funcional, permitiendo estudiar el comportamiento del sistema y cerrar el lazo de control con el PLC sin necesidad de disponer físicamente de la instalación. Según la documentación web [39], Simulink es un entorno gráfico basado en diagramas de bloques para simulación multidominio y diseño basado en modelos, integrado con MATLAB y provisto de bibliotecas de bloques, herramientas gráficas y solvers para representar sistemas dinámicos. Asimismo, la documentación web [40] describe la posibilidad de configurar clientes OPC-UA en Simulink para leer y escribir datos desde servidores industriales, como el generado en este caso desde TIA Portal.

El sistema modelado corresponde a un depósito de sección transversal constante A, alimentado por un caudal de entrada $Q_{in}(t)$ y provisto de una salida cuya oposición al vaciado se representa mediante una resistencia hidráulica equivalente R. La variable de interés es la altura del nivel de agua $h(t)$, ya que representa el estado dinámico del proceso y constituye la magnitud que se desea regular mediante el controlador.

Como se ha desarrollado en el apartado anterior, el modelo matemático finalmente adoptado describe la evolución temporal del nivel a partir del equilibrio entre el caudal de entrada y la descarga del depósito. La ecuación diferencial empleada para su implementación en MATLAB/Simulink se recoge en la ecuación (12):

$$\frac{dh(t)}{dt} = \frac{1}{A} Q_{in}(t) - \frac{1}{RA} h(t) \tag{12}$$

Esta expresión constituye la base del modelo dinámico empleado en la simulación. A partir de ella se calcula, en cada instante, la derivada del nivel $\frac{dh}{dt}$, que posteriormente se integra para obtener la evolución temporal de la altura h(t). De esta forma, la ecuación (12) permite trasladar al entorno de simulación el comportamiento dinámico del depósito obtenido analíticamente en el apartado previo.

La implementación de la ecuación (12) en el entorno de MATLAB/Simulink se muestra en la Figura 6.

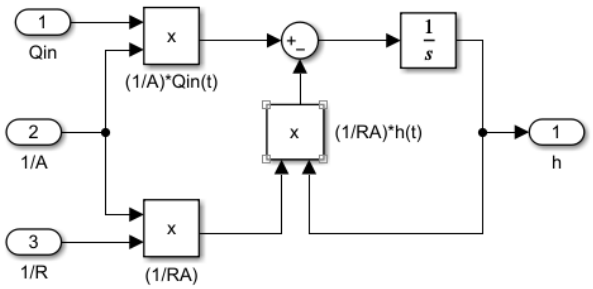


Figura 6: Implementación de la ecuación del depósito en Simulink.

Una vez definida la ecuación del proceso, se construyó el modelo completo en Simulink, incorporando tanto la dinámica de la planta como los bloques necesarios para su comunicación con el PLC. La estructura general del modelo desarrollado se muestra en la Figura 7.

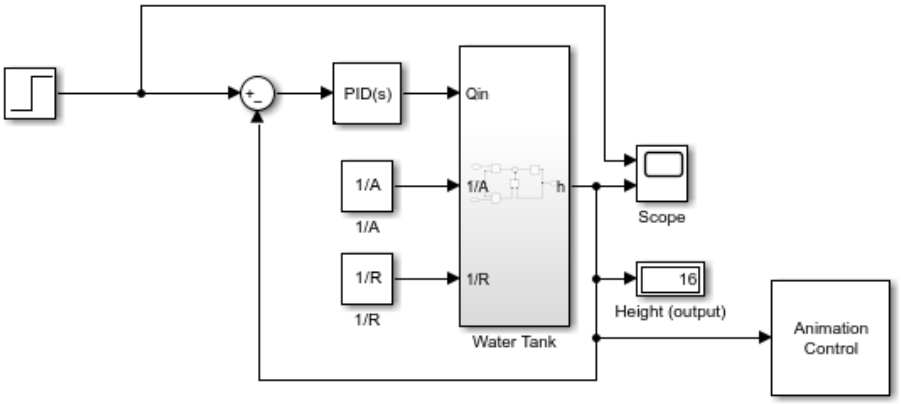


Figura 7: Modelo Simulink del sistema de control de nivel.

El elemento central del diagrama es el subsistema Water Tank, encargado de representar la dinámica del depósito. Este bloque recibe como entrada principal la señal de actuación Q_{in}, que corresponde al valor calculado por el regulador PID ejecutado en TIA Portal. A partir de dicha señal y de los parámetros definidos en el modelo, el subsistema calcula la evolución del nivel de agua y genera como salida la variable h, que representa la altura instantánea del líquido en el tanque.

La salida del modelo se ramifica hacia distintos elementos. Por un lado, se conduce al bloque de escritura OPC-UA, que la envía al PLC para que pueda ser utilizada como variable de proceso por el regulador. Por otro, se conecta a elementos de visualización empleados durante la simulación, como un bloque de salida numérica y los sistemas gráficos utilizados para observar la respuesta temporal del nivel. Además, esta señal se emplea posteriormente en el módulo de animación desarrollado para representar visualmente el llenado y vaciado del depósito.

Desde el punto de vista temporal, la configuración del modelo se definió en dos niveles diferenciados. Por un lado, en los parámetros generales de simulación de Simulink se seleccionó un solver de paso fijo (Fixed-step), concretamente ode4 (Runge-Kutta), con un paso fundamental de 0,01 s. Esta configuración determina la resolución temporal interna con la que Simulink integra la dinámica del depósito. De este modo, la planta virtual se actualiza cada 10 ms, permitiendo representar su evolución de forma más continua que el intercambio de datos con el PLC.

La elección de un paso de simulación inferior al periodo de comunicación resulta coherente con el comportamiento de un proceso físico real. En una instalación industrial, la planta no permanece inmóvil entre dos ejecuciones consecutivas del controlador, sino que continúa evolucionando de forma continua mientras el PLC lee las variables, calcula la acción de control y actualiza sus salidas de manera periódica. En el modelo desarrollado, esta situación se aproxima haciendo que la dinámica del depósito se resuelva internamente con mayor frecuencia que la lectura y escritura de variables mediante OPC-UA. Así, aunque el controlador recibe y entrega información cada 20 ms, la altura del líquido se calcula con una resolución intermedia de 10 ms, ofreciendo una representación temporal más fina de la planta virtual.

La configuración del solver y del paso fijo empleada en Simulink se muestra en la Figura 8.

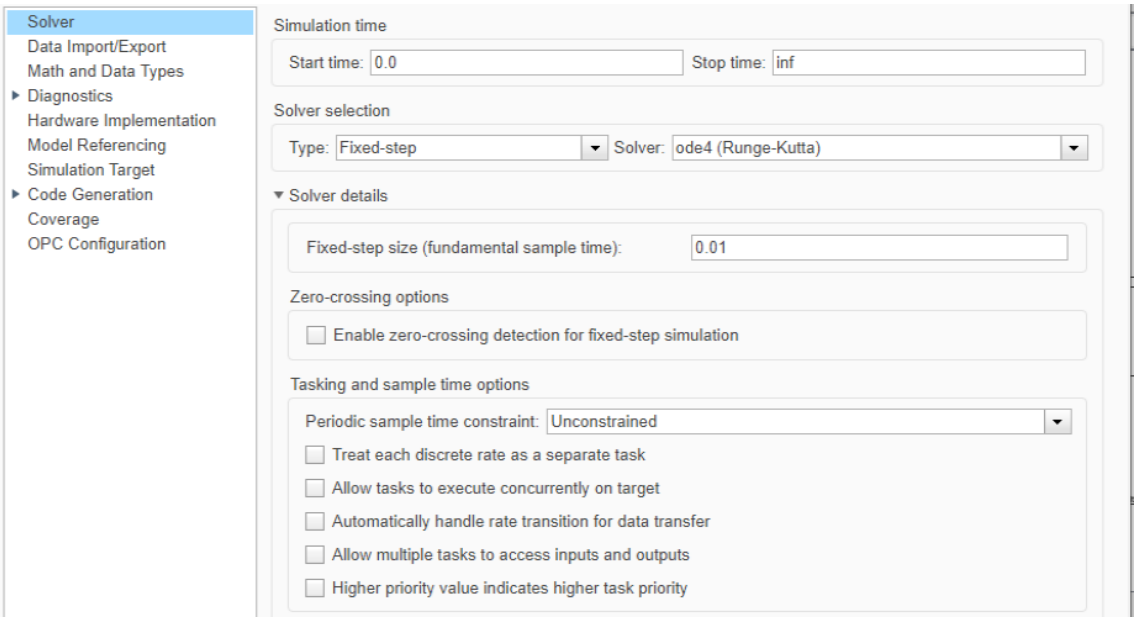


Figura 8: Configuración del Solver de Simulink.

Aunque inicialmente se valoró emplear pasos de simulación menores, como 0,001 s, esta opción incrementaba de forma notable la carga computacional del modelo. Al reducir el paso fijo, Simulink debe realizar un número mucho mayor de cálculos de integración por cada segundo simulado, lo que ralentizaba perceptiblemente la ejecución, especialmente al trabajar de forma simultánea con la comunicación OPC-UA, la animación del depósito y la representación gráfica en tiempo real. Dado que el proceso modelado presenta una dinámica relativamente lenta, se consideró que un paso de 0,01 s ofrecía un compromiso adecuado entre resolución temporal, estabilidad de la simulación y fluidez de ejecución.

Además, se activó la opción Simulation Pacing con un valor de 1. Esta configuración hace que la simulación avance aproximadamente a razón de un segundo de tiempo

simulado por cada segundo de tiempo real. Su uso resulta adecuado en este trabajo por dos motivos. En primer lugar, facilita la interacción entre Simulink y el PLC simulado, evitando que el modelo avance a máxima velocidad sin guardar una relación intuitiva con el tiempo de ejecución del sistema externo. En segundo lugar, permite observar de forma adecuada la evolución del nivel, las gráficas y la animación asociada al tanque durante el transcurso de la simulación.

3.2.1 Configuración del cliente OPC-UA en Simulink

Para establecer la comunicación entre el modelo de Simulink y el PLC configurado en TIA Portal, se añadió al modelo un cliente OPC-UA asociado al servidor publicado por la CPU simulada. Esta configuración se realiza desde el apartado OPC Configuration de los parámetros del modelo, accesible también desde los propios bloques OPC-UA mediante la opción Configure servers. Según la documentación web [40], este panel permite crear, configurar, conectar y desconectar los clientes OPC-UA empleados en el modelo.

En la implementación realizada, el cliente quedó vinculado al servidor identificado como SIMATIC.S7-1500.OPC-UA.Application:PLC_1, asociado al host 192.168.1.1, correspondiente a la CPU simulada en el entorno de Siemens. Esta configuración permite que Simulink localice el servidor OPC-UA generado desde TIA Portal y pueda acceder posteriormente a las variables publicadas por el PLC, que serán empleadas para intercambiar la acción de control y la variable de proceso.

La configuración del cliente OPC-UA incorporado al modelo de Simulink se muestra en la Figura 9.

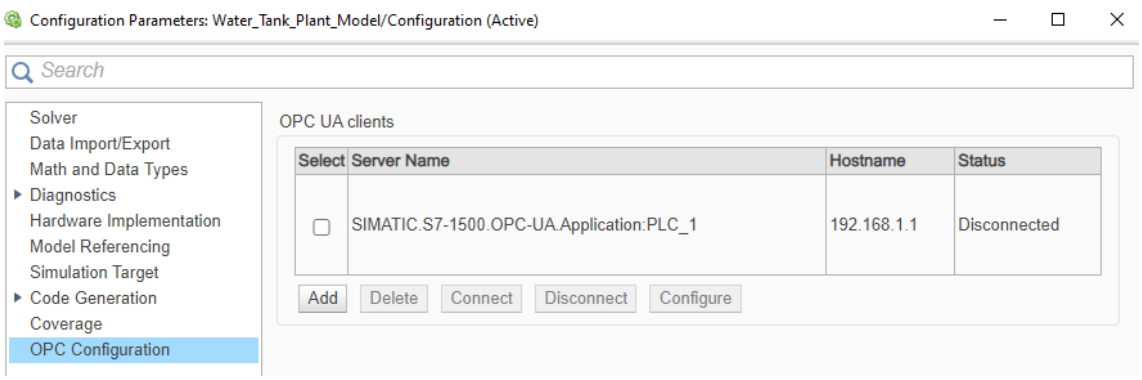


Figura 9: Configuración del cliente OPC-UA.

3.2.2 Bloque OPC-UA-Read

La lectura de datos desde el PLC hacia Simulink se realiza mediante el bloque OPC-UA Read. Su función dentro del modelo es recuperar la señal de actuación calculada por el regulador PID en TIA Portal e introducirla como entrada del modelo dinámico del depósito. Por este motivo, el bloque se sitúa antes del subsistema Water Tank, ya que su salida alimenta directamente la entrada Q_{in} de la planta virtual.

En la configuración adoptada, el bloque se asocia al servidor SIMATIC.S7-1500.OPC-UA.Application:PLC_1 y se selecciona el nodo salidaPID, perteneciente al índice de espacio de nombres 3. Esta variable contiene la salida generada por el bloque PID_Compact del PLC, es decir, la magnitud de control que debe aplicarse al modelo del depósito en cada instante de simulación.

La configuración del bloque OPC-UA Read, incluyendo el servidor seleccionado, el nodo salidaPID y su índice de espacio de nombres, se muestra en la Figura 10.

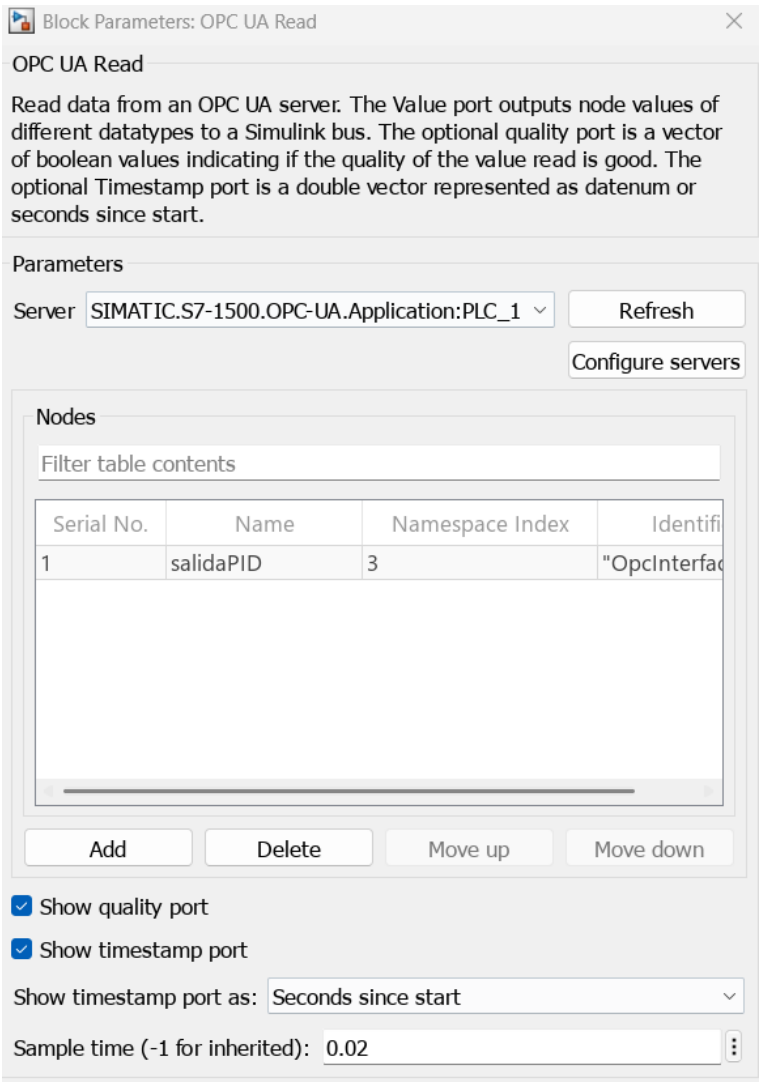


Figura 10: Configuración del bloque OPC-UA Read.

Además de la selección del servidor y del nodo, el bloque se configuró con un tiempo de muestreo de 0,02 s. Este valor determina la periodicidad con la que Simulink realiza la lectura de salidaPID desde el servidor OPC-UA. Su elección permite coordinar la adquisición de la acción de control con el ciclo de ejecución del regulador en TIA Portal, implementado en el bloque organizativo Cyclic20ms, también ejecutado cada 20 ms.

Asimismo, en el bloque se mantienen habilitadas las salidas auxiliares Timestamp y Quality, visibles en el diagrama del modelo. La salida Timestamp permite disponer de información temporal asociada al dato recibido, mientras que Quality proporciona una indicación sobre la validez de la lectura realizada desde el servidor OPC-UA. Aunque la señal empleada para cerrar el lazo de control es únicamente la salida Value, estas señales adicionales resultan útiles durante las pruebas de comunicación y la supervisión del intercambio de datos. Según la documentación web [41], el bloque OPC-UA Read permite leer nodos de un servidor OPC-UA de forma síncrona a un tiempo de muestreo especificado y, opcionalmente, generar salidas adicionales asociadas a la marca temporal y la calidad del dato.

3.2.3 Bloque OPC-UA-Write

La escritura de datos desde Simulink hacia el PLC se implementa mediante el bloque OPC-UA Write. Este bloque se sitúa a la salida del modelo del depósito, ya que su finalidad es transmitir al PLC la altura del nivel calculada por la planta virtual. De este modo, el valor de h(t) obtenido en Simulink queda disponible en TIA Portal como variable de proceso del regulador.

En la configuración adoptada, el bloque se asocia al servidor SIMATIC.S7-1500.OPC-UA.Application:PLC_1 y se selecciona el nodo entradaPID, perteneciente al índice de espacio de nombres 3. Esta variable recibe el nivel simulado del depósito y se conecta posteriormente a la entrada Input del bloque PID_Compact en el PLC.

La configuración del bloque OPC-UA Write, incluyendo el servidor empleado, el nodo entradaPID y su índice de espacio de nombres, se muestra en la Figura 11.

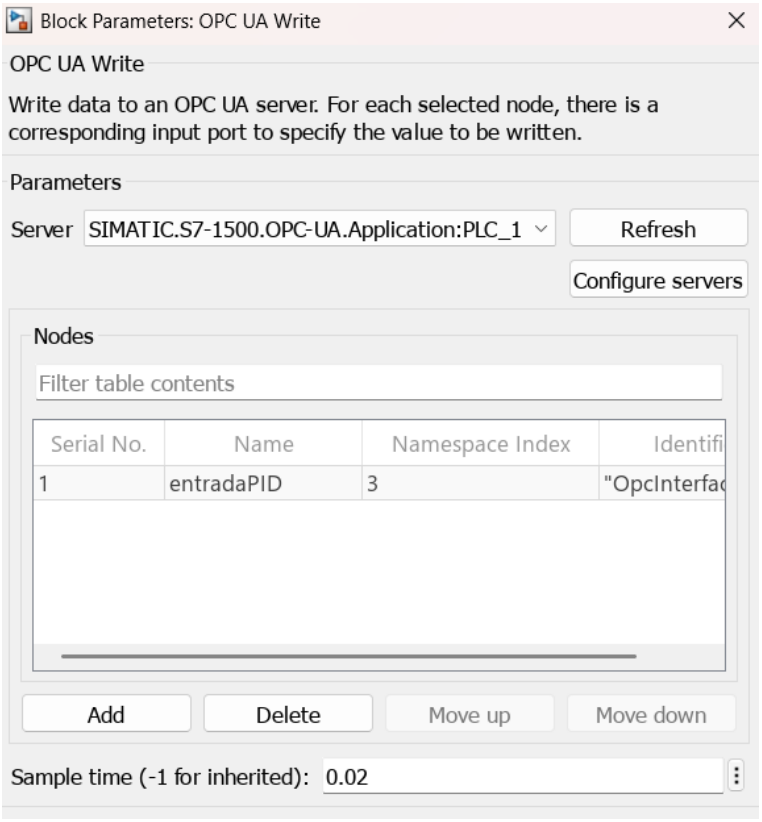


Figura 11: Configuración OPC-UA Write.

El bloque se configuró igualmente con un tiempo de muestreo de 0,02 s. De esta forma, la altura calculada por el modelo se escribe en el servidor OPC-UA con la misma periodicidad con la que el regulador ejecuta su algoritmo y con la que Simulink lee la señal de control. Esta correspondencia temporal evita que el PLC trabaje con valores de proceso actualizados a una frecuencia distinta de la utilizada para generar la acción de control.

Dado que en esta implementación únicamente se transmite una variable desde Simulink hacia el PLC, el bloque dispone de un solo puerto de entrada, asociado directamente al nodo entradaPID. Cada nuevo valor de altura generado por el modelo del depósito se introduce por dicho puerto y se escribe sobre la variable correspondiente del servidor OPC-UA. Según la documentación web [42], el bloque OPC-UA Write realiza escrituras síncronas sobre los nodos seleccionados a un tiempo de muestreo definido, asignando un puerto de entrada a cada nodo configurado.

En conjunto, la disposición de los bloques OPC-UA permite cerrar el lazo de control entre ambos entornos. El PLC calcula la acción de control y la almacena en salidaPID; Simulink lee esa variable mediante el bloque OPC-UA Read, la utiliza como entrada del modelo del depósito y calcula la nueva altura del líquido; finalmente, dicha altura se escribe en entradaPID mediante el bloque OPC-UA Write, quedando disponible de nuevo para el regulador. Esta organización convierte al modelo de Simulink en una

planta virtual gobernada por un controlador industrial externo, reproduciendo una arquitectura de co-simulación en la que la dinámica del proceso y la lógica de control se ejecutan en plataformas distintas, pero coordinadas mediante un intercambio periódico de variables a través de OPC-UA.

La integración de ambos bloques dentro del modelo general de Simulink se muestra en la Figura 12. En ella puede observarse cómo el bloque OPC-UA Read recibe desde el PLC la variable salidaPID, correspondiente a la acción de control calculada por PID_Compact, y la aplica como caudal de entrada Q_{in} al subsistema Water Tank. A su vez, la altura del líquido calculada por el modelo se envía de vuelta al PLC mediante el bloque OPC-UA Write, que escribe dicho valor en la variable entradaPID. Esta misma señal se utiliza también para la visualización del nivel, tanto en el bloque de salida como en el sistema de animación desarrollado para representar gráficamente la evolución del depósito.

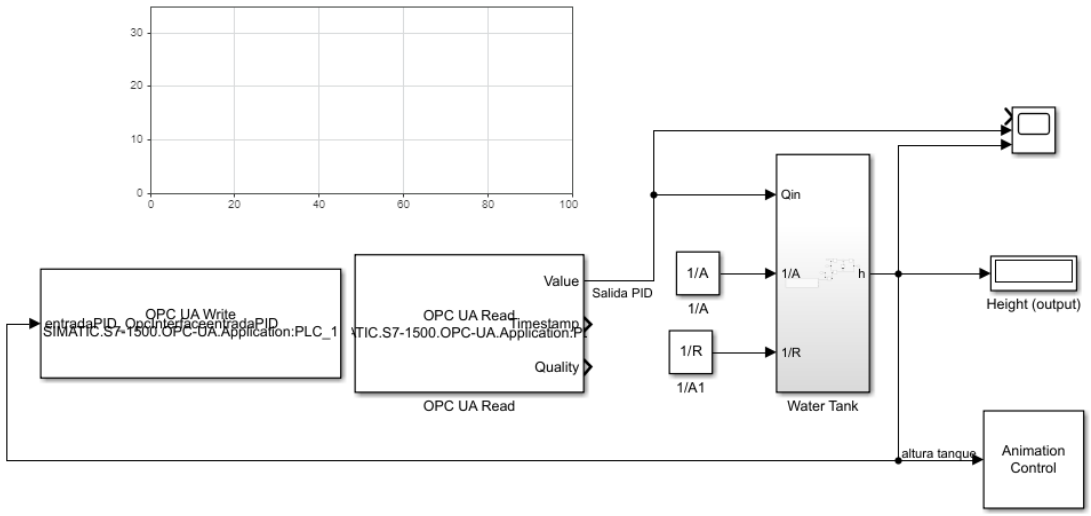


Figura 12: Integración de los bloques OPC en el sistema de control.

Además del intercambio principal de señales asociado al lazo de control, el modelo incorpora una lectura adicional desde el PLC destinada a introducir perturbaciones sobre la dinámica de descarga del depósito. Para ello, se añadió un segundo bloque OPC-UA Read específico, configurado sobre el servidor SIMATIC.S7-1500.OPC-UA.Application:PLC_1, encargado de leer el nodo perturbacionopc, perteneciente al índice de espacio de nombres 3. Esta variable contiene el valor de perturbación calculado en TIA Portal a partir de la orden introducida por el usuario desde la HMI.

El bloque se configuró con un tiempo de muestreo de 0,02 s, manteniendo la misma periodicidad que el resto de los intercambios OPC-UA del modelo. Asimismo, se conservaron habilitadas las salidas auxiliares Timestamp y Quality, útiles durante las pruebas de comunicación para verificar la marca temporal asociada al dato recibido y

la validez de la lectura, aunque la señal empleada directamente en la dinámica del depósito es únicamente la salida Value.

La forma en que esta perturbación se integra en el modelo se muestra en la Figura 13. En el interior del subsistema Water Tank, el valor recibido desde el PLC se suma al valor nominal de $1/R$. El resultado de esta operación modifica el término asociado al caudal de salida del depósito, permitiendo representar variaciones temporales en la resistencia hidráulica equivalente. De este modo, las perturbaciones se generan desde el entorno de supervisión y control del PLC, pero su efecto se aplica directamente sobre la planta virtual desarrollada en Simulink, lo que permite evaluar la capacidad del regulador para rechazar cambios externos en las condiciones de funcionamiento del proceso.

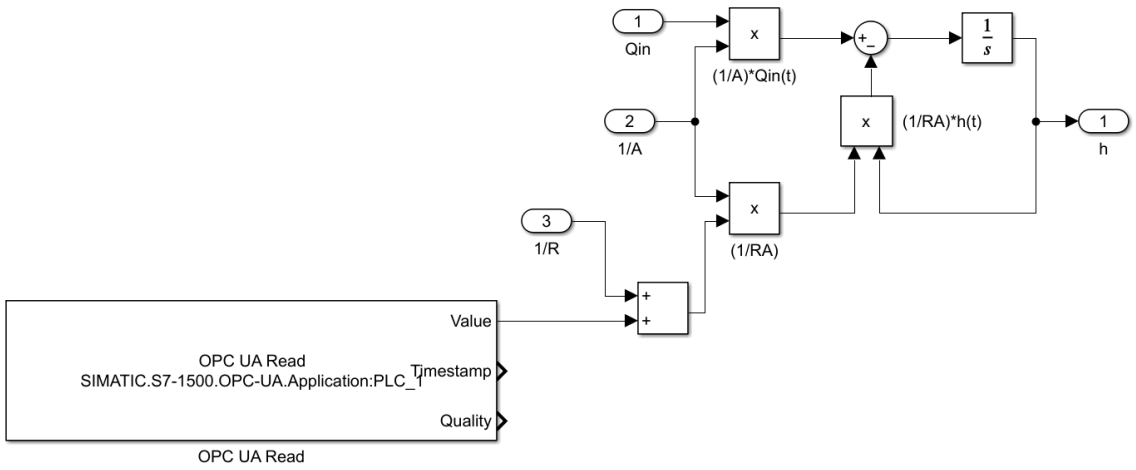


Figura 13: Integración de la perturbación mediante OPC-UA.

3.2.4 Bloque de animación y visualización

Como complemento al modelo dinámico del depósito y a los bloques de comunicación OPC-UA, se incorporó al diagrama de Simulink un bloque de visualización denominado Animation Control. Este bloque se conecta directamente a la señal de salida del tanque, correspondiente a la altura instantánea $h(t)$, y permite representar de forma gráfica la evolución del nivel durante la simulación. Su finalidad no es intervenir en el cálculo de la dinámica del sistema ni en el cierre del lazo de control, sino facilitar la supervisión visual del comportamiento de la planta virtual mientras esta intercambia datos con el PLC.

El funcionamiento de este módulo se apoya en dos archivos desarrollados en MATLAB: tank_anim_sfun_simple.m y tank_anim_gui_simple.m. Ambos trabajan de forma coordinada. El primero actúa como interfaz entre el modelo de Simulink y el entorno gráfico de MATLAB, mientras que el segundo contiene la lógica encargada de crear y actualizar la ventana de animación.

El archivo tank_anim_sfun_simple.m define una S-Function empleada como bloque dentro del modelo. Esta función dispone de una única entrada, a la que llega el valor de la altura del depósito calculado por Simulink, y no genera ninguna salida, ya que su propósito es exclusivamente visual. Durante el inicio de la simulación, la S-Function llama al archivo gráfico para crear la ventana de animación, o reutilizarla si ya existe, y reinicia la gráfica temporal del nivel. Posteriormente, en cada actualización del modelo, recibe el valor actual de $h(t)$, obtiene el tiempo de simulación correspondiente y transmite ambas magnitudes a la función gráfica para que la representación visual se actualice de manera sincronizada con la evolución del sistema.

Por su parte, el archivo tank_anim_gui_simple.m implementa la interfaz gráfica propiamente dicha. Esta función genera una ventana en MATLAB que contiene, por un lado, una representación esquemática del depósito con sus tuberías de entrada y salida y, por otro, una gráfica temporal de la altura del líquido. A medida que la S-Function le envía nuevos valores de nivel, la interfaz modifica la altura dibujada del agua dentro del tanque, actualiza el valor numérico mostrado en pantalla y añade un nuevo punto a la curva $h(t)$. De esta forma, la animación refleja visualmente el llenado o vaciado del depósito al mismo tiempo que la gráfica permite analizar su evolución cuantitativa.

La comunicación entre ambos archivos se realiza mediante llamadas sucesivas desde la S-Function a la función gráfica. En el arranque de la simulación se ejecutan las acciones de creación o recuperación de la interfaz y de reinicio de la gráfica. Durante la ejecución, cada nuevo valor de altura se envía a la interfaz mediante una acción de actualización, acompañada del instante de simulación en el que se ha obtenido. Así, el bloque `Animation Control` convierte la señal numérica generada por el modelo en una representación visual dinámica, manteniendo la correspondencia entre la animación, la gráfica y el estado real de la simulación.

Además, la interfaz incluye un control de inicio y parada de la simulación, que permite actuar directamente sobre el modelo desde la propia ventana gráfica. Este elemento facilita la realización de pruebas y la observación del comportamiento del sistema sin necesidad de volver continuamente al entorno principal de Simulink. Aunque esta funcionalidad es auxiliar, contribuye a mejorar la interacción con el modelo durante la fase de validación.

La interfaz de animación y seguimiento gráfico del nivel desarrollada en MATLAB se muestra en la Figura 14.

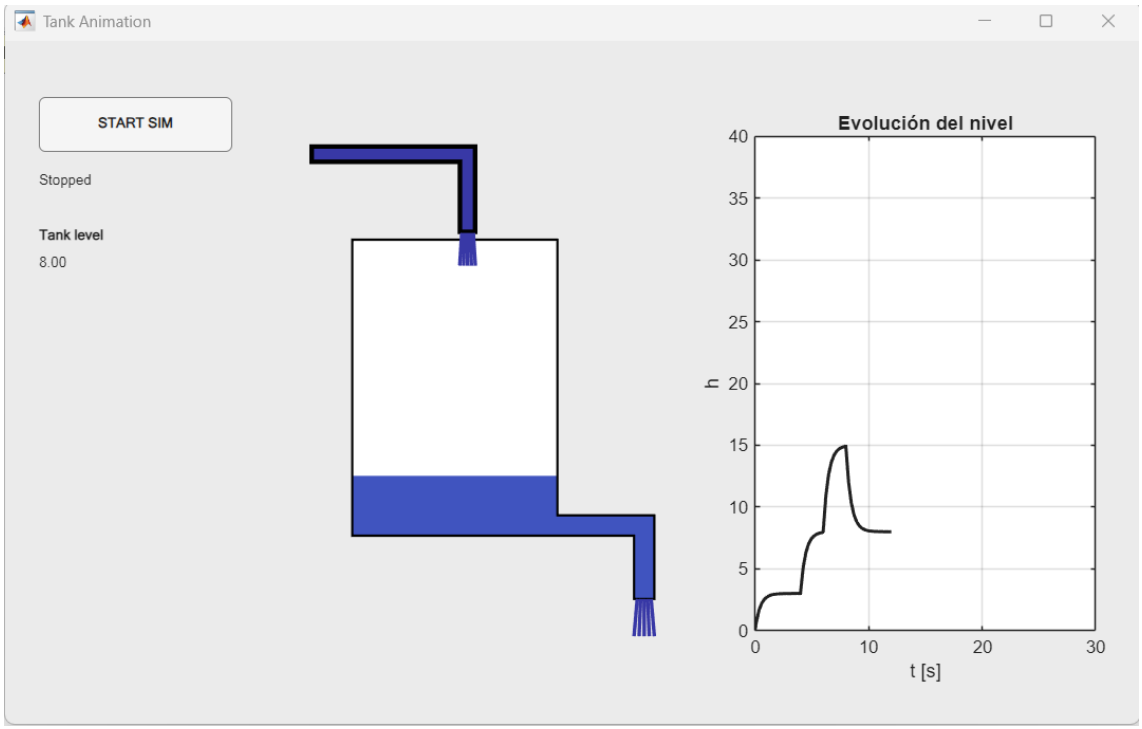


Figura 14: Ventana de animación con el tanque y la gráfica.

En conjunto, este módulo de visualización permite interpretar de forma más intuitiva la respuesta del depósito ante las acciones de control enviadas desde el PLC. La representación simultánea del nivel de agua, su valor numérico y su evolución temporal facilita la detección de comportamientos transitorios, sobreimpulsos, cambios de régimen o posibles anomalías durante las pruebas de funcionamiento del sistema completo.

3.3 Control PID en el entorno TIA Portal

En esta sección se describe el despliegue del control PID en el PLC SIMATIC S7-1500 utilizando el entorno TIA Portal para el control en lazo cerrado distribuido entre el PLC y la planta virtual desarrollada en Simulink. La sección comienza presentando el fundamento teórico del regulador PID y su expresión en tiempo continuo. A continuación, se detalla la implementación práctica mediante el objeto tecnológico PID_Compact de Siemens, incluyendo la configuración del periodo de muestreo (o de ejecución cíclica), el conexionado de las señales de consigna, variable de proceso y salida, así como los ajustes básicos de escalado y límites. Posteriormente, se aborda el ajuste de los parámetros del controlador, partiendo de una estimación inicial mediante optimización numérica en MATLAB y culminando con un autoajuste fino realizado desde la propia herramienta de TIA Portal, que proporciona la

parametrización final adoptada. Finalmente, se analiza la relación entre la comunicación OPC-UA y el lazo de control, explicando cómo el intercambio cíclico de variables entre Simulink y el PLC permite cerrar el bucle de regulación de forma distribuida.

3.3.1 TIA Portal como entorno de ingeniería

TIA Portal constituye el entorno de ingeniería empleado para desarrollar la parte de automatización del sistema. En una misma plataforma se integran la configuración del hardware, la programación del autómata, la parametrización de objetos tecnológicos, la gestión de comunicaciones industriales y el diseño de interfaces HMI.

En el presente trabajo, TIA Portal se utiliza para implementar la lógica de control ejecutada en el PLC, configurar el servidor OPC-UA necesario para el intercambio de datos con Simulink, parametrizar el regulador PID_Compact y organizar las variables empleadas durante la operación del sistema. Esta función resulta especialmente relevante en la arquitectura planteada, ya que el controlador se ejecuta en un entorno propio de automatización industrial, mientras que la planta física se sustituye por el modelo virtual desarrollado en MATLAB/Simulink.

De este modo, el PLC mantiene su papel como elemento de control, pero actúa sobre una planta simulada con la que intercambia variables de proceso y mando. Esta aproximación permite desarrollar, depurar y validar el comportamiento del sistema de regulación en condiciones representativas antes de su aplicación sobre una instalación física.

3.3.2 Selección del PLC

Para la implementación del sistema se seleccionó una CPU SIMATIC S7-1500, modelo CPU 1516F-3 PN/DP, coincidente con los equipos disponibles en el laboratorio de la universidad. Esta elección permite que el desarrollo realizado en el entorno virtual mantenga una correspondencia directa con el hardware sobre el que podría reproducirse posteriormente la solución. Además, la documentación técnica de Siemens recoge el uso del objeto tecnológico PID_Compact en controladores SIMATIC S7-1200 y S7-1500, lo que hace que esta familia de PLC resulte adecuada para la implementación del regulador industrial empleado en el proyecto [43].

Asimismo, la CPU seleccionada resulta apropiada para la arquitectura planteada porque dispone de funcionalidad de servidor OPC-UA. Esta característica permite que clientes externos, como el modelo desarrollado en Simulink, accedan a las variables publicadas desde el PLC para realizar operaciones de lectura, escritura o suscripción. Siemens documenta esta capacidad en la hoja de datos de la CPU 1516F-3 PN/DP, donde se indica la disponibilidad de servidor OPC-UA y acceso a datos mediante dichos mecanismos [44].

Aunque en este trabajo se emplea una CPU S7-1500 por coherencia con el entorno de laboratorio, la estrategia general no queda limitada exclusivamente a este modelo. La arquitectura podría trasladarse a otras CPUs Siemens compatibles con PID_Compact y que dispongan de servidor OPC-UA integrado. En cualquier caso, la disponibilidad concreta de esta funcionalidad dependerá del modelo de CPU, de la versión de firmware y de las opciones habilitadas en cada dispositivo [44].

3.3.3 Configuración de la comunicación OPC-UA en TIA Portal

La integración entre el PLC y el modelo de planta desarrollado en Simulink se realiza mediante comunicación OPC-UA. Esta comunicación permite distribuir el lazo de control entre ambos entornos: el regulador se ejecuta en TIA Portal, mientras que la dinámica del depósito se resuelve en Simulink. De esta forma, el controlador industrial actúa sobre una planta virtual externa y recibe de ella la variable de proceso necesaria para cerrar la realimentación. Este planteamiento coincide con la arquitectura descrita por Siemens [45] para la conexión de modelos de Simulink con controladores SIMATIC mediante OPC-UA en entornos de validación virtual.

El primer paso de configuración consistió en activar el servidor OPC-UA de la CPU dentro de las propiedades del dispositivo. En la implementación realizada, el servidor queda habilitado en la dirección opc.tcp://192.168.1.1:4840, que será utilizada posteriormente desde Simulink para registrar el cliente OPC-UA y acceder a las variables publicadas por el PLC.

La activación del servidor OPC-UA y la dirección de conexión configurada en la CPU se muestran en la Figura 15.

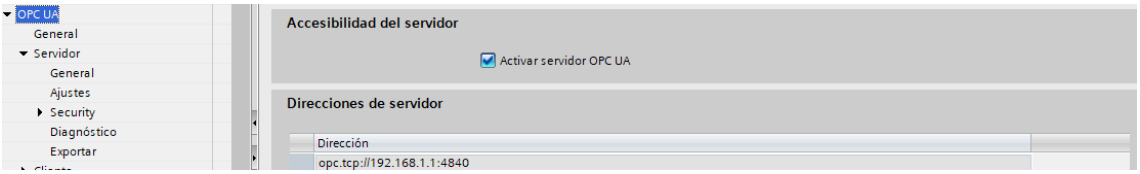


Figura 15: Activación del servidor OPC-UA.

3.3.4 Organización de los bloques de datos

Para estructurar las variables del proyecto se emplearon dos bloques de datos globales diferenciados. Por un lado, se creó el bloque Global (DB1), destinado a almacenar variables de funcionamiento interno del sistema, tales como consignas, modos de operación, órdenes procedentes de la HMI y señales auxiliares asociadas a perturbaciones o maniobras específicas. La creación de este bloque de datos se muestra en la Figura 16.

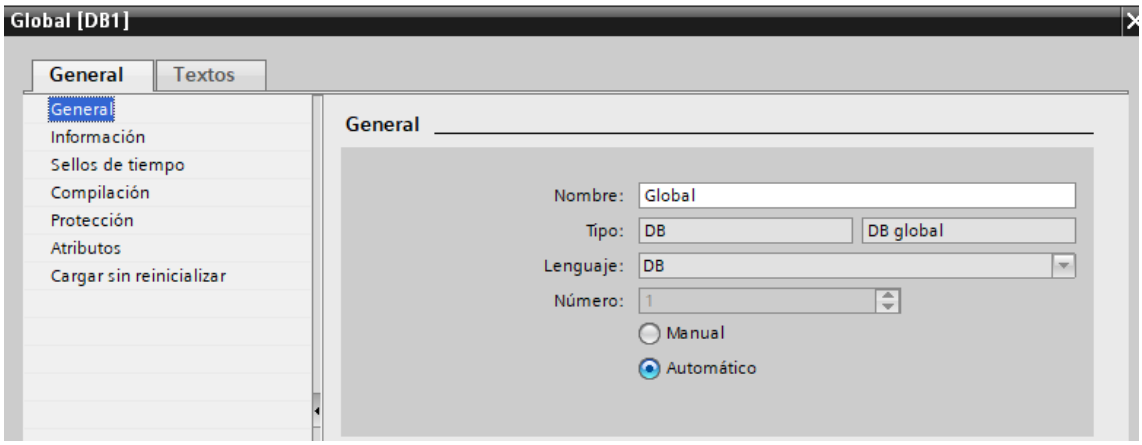


Figura 16: Creación bloque Global.

Las principales variables definidas en el bloque Global se presentan en la Figura 17. Entre ellas se encuentran setpoint_PID, utilizada como consigna del regulador; manualvalue, empleada para fijar una salida manual del controlador; y las variables modoactivate y modo, utilizadas para gestionar el modo de operación del PID_Compact. También se incorporan señales vinculadas a la lógica de interacción con la HMI, como órdenes de encendido, parada o modificación temporal de la referencia.

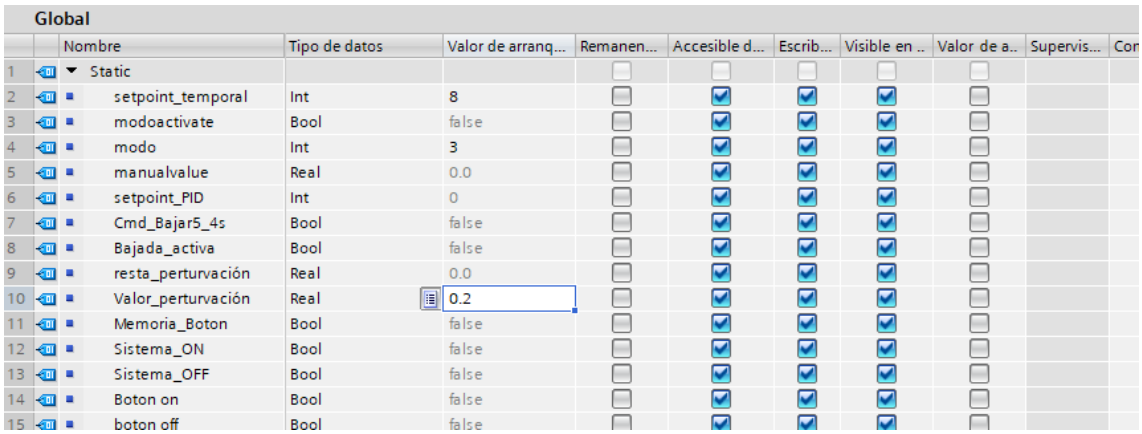


Figura 17: Principales variables Global.

Por otro lado, se creó el bloque OpcInterface (DB2), reservado específicamente para el intercambio de señales entre el PLC y Simulink mediante OPC-UA. Esta separación facilita la organización del proyecto y permite identificar de forma clara qué variables forman parte de la interfaz de comunicación externa. La definición de este bloque de datos se recoge en la Figura 18.

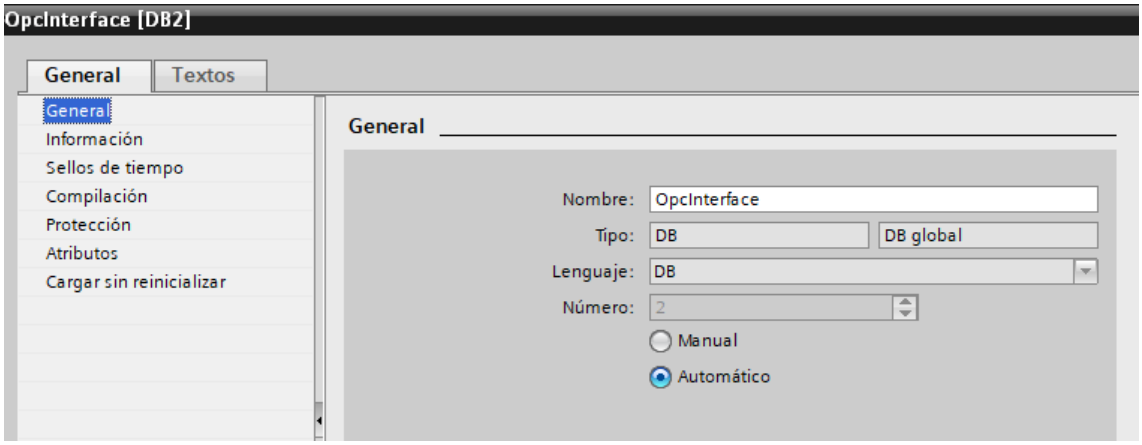


Figura 18: Creación bloque OpcInterface.

Para que el bloque pudiera ser utilizado como interfaz de comunicación, se habilitó la opción Bloque de datos accesible desde OPC-UA dentro de sus atributos. Asimismo, se mantuvo activado el acceso optimizado al bloque. Esta configuración se muestra en la Figura 19.

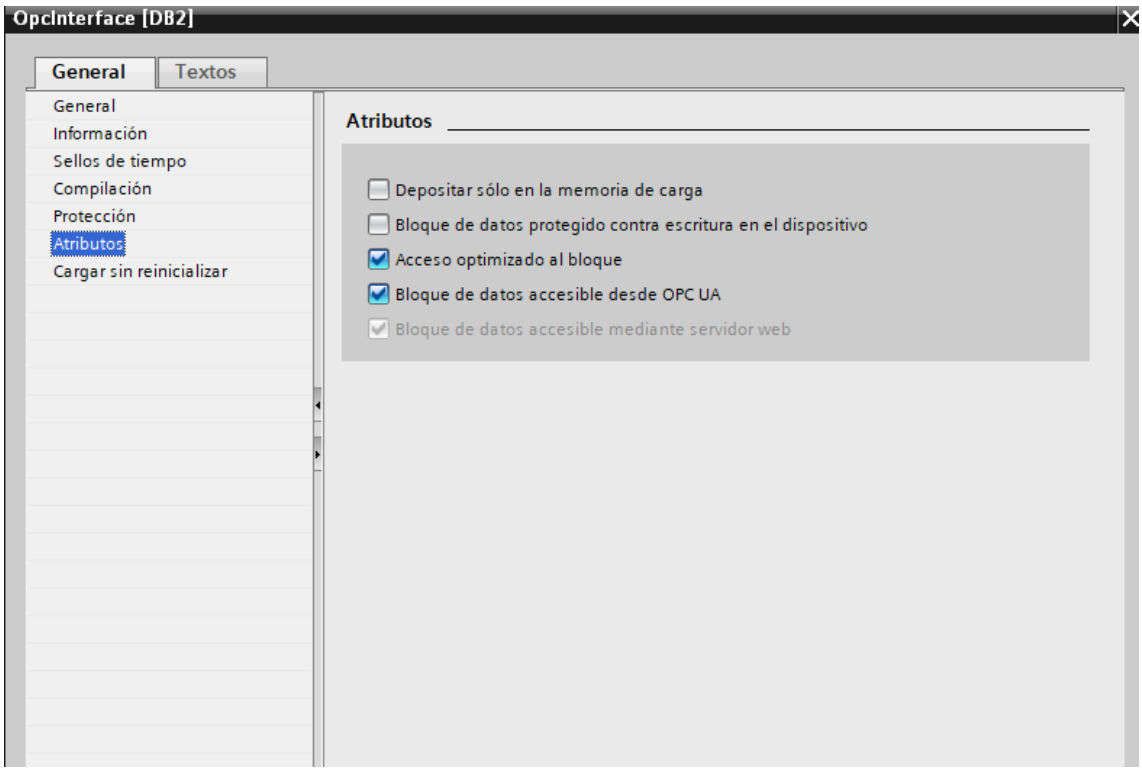
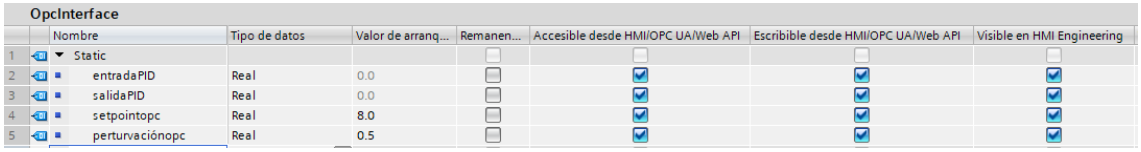


Figura 19: Configuración OpcInterface.

Dentro de OpcInterface se definieron cuatro variables de tipo Real: entradaPID, salidaPID, setpointopc y perturbacionopc. Todas ellas se configuraron como accesibles y escribibles desde HMI/OPC-UA/Web API, lo que permite su utilización

durante las pruebas de comunicación y supervisión. La estructura interna de este bloque se muestra en la Figura 20.



OpcInterface

	Nombre	Tipo de datos	Valor de arranq...	Remanen...	Accesible desde HMI/OPC UA/Web API	Escribible desde HMI/OPC UA/Web API	Visible en HMI Engineering
1	Static						
2	entradaPID	Real	0.0				
3	salidaPID	Real	0.0				
4	setpointopc	Real	8.0				
5	perturvaciónopc	Real	0.5				

Figura 20: Principales variables OpcInterface.

En la implementación actual, las dos variables fundamentales para cerrar el lazo de control son entradaPID y salidaPID. La variable salidaPID almacena la acción de control generada por el regulador ejecutado en el PLC, mientras que entradaPID recibe desde Simulink la altura calculada del depósito y actúa como variable de proceso del controlador. Las variables setpointopc y perturbacionopc se mantienen como señales de interfaz adicionales, útiles para pruebas específicas y para posibles ampliaciones posteriores del sistema.

El intercambio de información se produce de forma cíclica. En cada periodo de ejecución, el regulador calcula una nueva salida y la deposita en salidaPID. Simulink lee este valor, lo emplea como entrada del modelo del tanque y genera una nueva altura de nivel. A continuación, dicha altura se escribe en entradaPID, quedando disponible para que el PLC la utilice en el siguiente cálculo del regulador. De este modo, OPC-UA actúa como nexo entre el controlador industrial y la planta virtual.

3.3.5 Fundamentos del regulador PID

Un regulador PID es un controlador en lazo cerrado que calcula una señal de mando a partir del error existente entre la consigna y la variable de proceso, con el objetivo de que esta última siga el valor deseado de forma estable y con una dinámica adecuada. El texto de referencia [46] expone que la acción proporcional genera una respuesta inmediata frente al error instantáneo, la acción integral acumula dicho error para eliminar desviaciones en régimen permanente y la acción derivativa anticipa la evolución del sistema a partir de la variación temporal del error.

La expresión ideal en tiempo continuo del regulador PID se presenta en la ecuación (13):

$$u(t) = K_p\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{de(t)}{dt}\right) \quad (13)$$

donde $\mathrm{u(t)}$ es la señal de control, $e(t)$ representa el error entre consigna y variable de proceso y K_p, T_i, T_d son los parámetros proporcional, integral y derivativo.

En el sistema desarrollado, este regulador se encarga de controlar la altura del líquido en el depósito. Para ello, compara la consigna de nivel con la altura calculada por el modelo de Simulink y genera una señal de actuación que se transmite de nuevo a la planta virtual.

3.3.6 Implementación del regulador mediante PID_Compact

En TIA Portal, la regulación se implementa mediante el objeto tecnológico PID_Compact, disponible para controladores SIMATIC S7-1200 y S7-1500. Este bloque permite configurar reguladores de tipo P, PI, PD o PID completo, e incorpora funciones específicas para aplicaciones industriales, como la limitación de salida, la operación en modo manual o automático, el tratamiento de saturaciones y el filtrado de la acción derivativa.

Aunque el principio de funcionamiento responde al controlador PID descrito anteriormente, PID_Compact incorpora una estructura de regulación más flexible, con ponderación de consigna y filtrado derivativo. De manera general, el comportamiento del bloque puede representarse mediante la ecuación (14):

$$\mathrm{u(s)} = \mathrm{K_p}\left[\left(\mathrm{bw(s)} - \mathrm{x(s)}\right) + \frac{1}{\mathrm{T_i\ s}}\left(\mathrm{w(s)} - \mathrm{x(s)}\right) + \frac{\mathrm{T_d\ s}}{\mathrm{aT_d\ s} + 1}\left(\mathrm{cw(s)} - \mathrm{x(s)}\right)\right] \qquad (14)$$

donde $u(s)$ es la salida del controlador, $w(s)$ la consigna, $x(s)$ la variable de proceso, a el coeficiente de filtrado de la acción derivativa y b y c los factores de ponderación asociados a la consigna.

Ejecución cíclica del regulador

El algoritmo del regulador debe ejecutarse con un periodo temporal constante. Para ello, en el proyecto se creó un bloque organizativo de interrupción cíclica denominado Cyclic20ms, configurado con un tiempo de ciclo de 20000 µs = 20 ms. Esta configuración se muestra en la Figura 21.

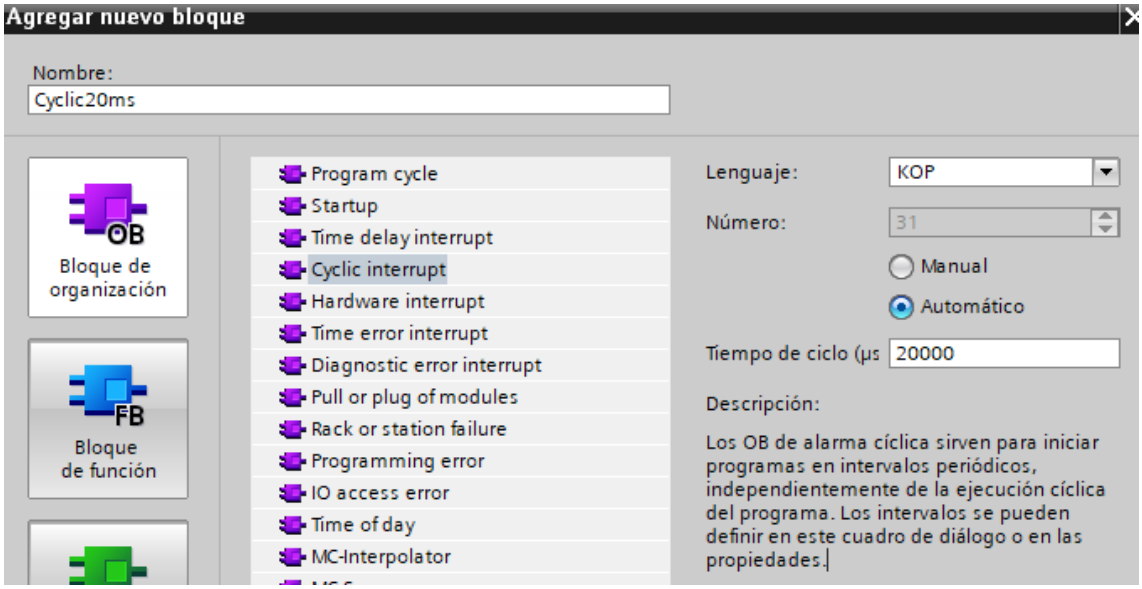


Figura 21: Configuración del periodo de muestreo.

La elección de un periodo de 20 ms responde a la necesidad de sincronizar la ejecución del PID_Compact con el tiempo de muestreo utilizado en Simulink y en los bloques OPC-UA de lectura y escritura. En cada ciclo, el PLC recibe el valor actualizado de la variable de proceso, calcula la nueva acción de control y la deja disponible para que Simulink la lea en el siguiente intercambio. Esta coincidencia temporal resulta fundamental para evitar desfases, lecturas obsoletas o comportamientos inconsistentes dentro del lazo de control.

Conexionado del bloque PID_Compact

Una vez definido el ciclo de ejecución, se incorporó al programa el bloque PID_Compact. La llamada al regulador y la conexión de sus principales entradas y salidas se muestran en la Figura 22.

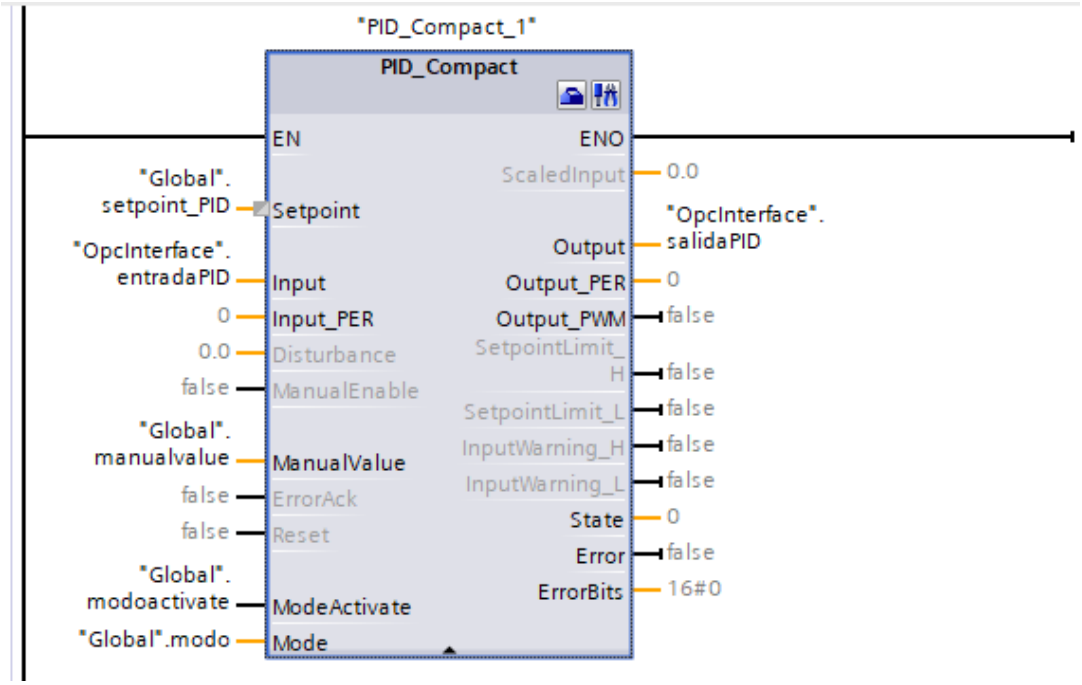


Figura 22: Conexionado PID_Compact.

La consigna del regulador se asoció a la variable "Global".setpoint_PID ubicada en el bloque de datos Global. Esta señal está definida como Int y representa la altura deseada del líquido en el depósito.

La variable de proceso se conectó a "OpcInterface".entradaPID, que contiene la altura enviada desde Simulink a través de OPC-UA. Esta señal se asigna a la entrada Input del PID_Compact, permitiendo que el regulador trabaje con la respuesta real del modelo virtual.

La salida del controlador se almacena en "OpcInterface".salidaPID y constituye la magnitud de actuación que Simulink lee para alimentar el modelo dinámico del depósito.

Además de estas señales principales, se configuraron varias entradas auxiliares del bloque. La entrada ManualValue se vinculó a la variable "Global".manualvalue para permitir la imposición de una salida manual cuando el regulador trabaja en ese modo. La entrada ModeActivate se conectó a "Global".modoactivate y se utiliza para habilitar

o confirmar el cambio de modo. Por su parte, la entrada Mode se asoció a "Global".modo, donde se emplea el valor 3 para el modo automático y el valor 4 para el modo manual. El resto de las entradas y salidas auxiliares del bloque se mantuvieron sin uso específico durante las pruebas principales.

Ajustes básicos del PID_Compact

En la configuración básica del objeto tecnológico se seleccionó una magnitud de proceso de tipo longitud. TIA Portal muestra en este caso la unidad µm de forma predeterminada, aunque dicha indicación no condiciona la escala empleada en el modelo desarrollado; en este trabajo, los valores de nivel se interpretan en centímetros. Asimismo, se mantuvo desactivada la inversión del sentido de regulación y se habilitó la opción Activar Mode tras rearrancar la CPU, estableciendo como estado de arranque el modo automático. De este modo, tras reiniciar el controlador, el bloque PID_Compact recupera directamente su funcionamiento en lazo cerrado sin necesidad de seleccionar manualmente el modo de regulación. La configuración básica del objeto PID_Compact se muestra en la Figura 23.

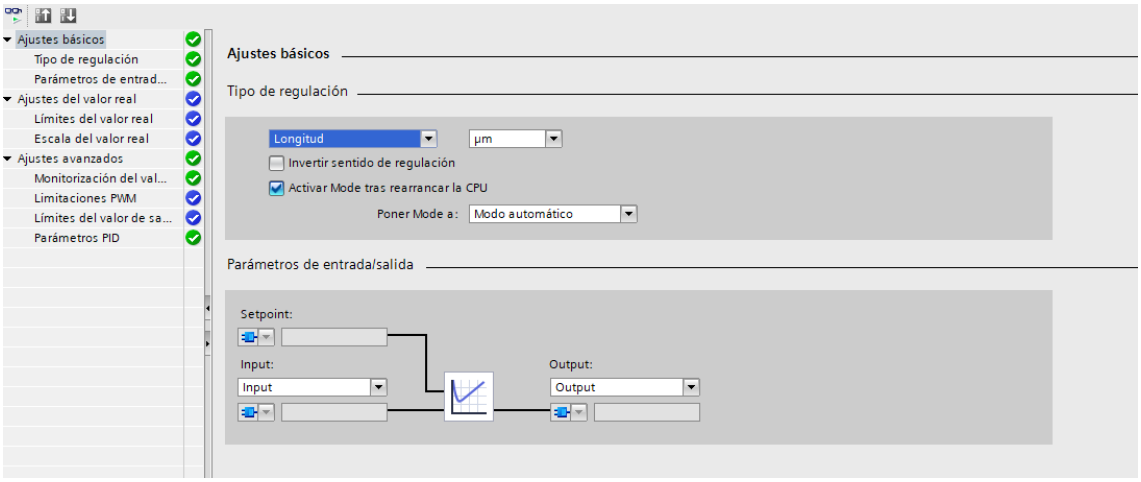


Figura 23: Configuración básica PID_Compact.

Límites y escala del valor real

El bloque PID_Compact se configuró con un límite inferior del valor real de 0,0 µm, ya que una altura negativa carece de sentido físico dentro del sistema. El límite superior se estableció en 120,0 µm, proporcionando un margen por encima de la escala principal de trabajo y evitando que transitorios elevados provoquen una saturación prematura de la variable de proceso.

La escala del valor real se definió entre 0,0 µm y 100,0 µm. Además, la entrada Input_PER se mantuvo desactivada, puesto que la variable de proceso no se recibe

como una señal periférica normalizada, sino directamente como un valor real procedente del modelo de Simulink a través de la entrada Input. De esta forma, el controlador opera directamente con la magnitud física intercambiada mediante OPC-UA.

La configuración de los límites y de la escala del valor real se muestra en la Figura 24.

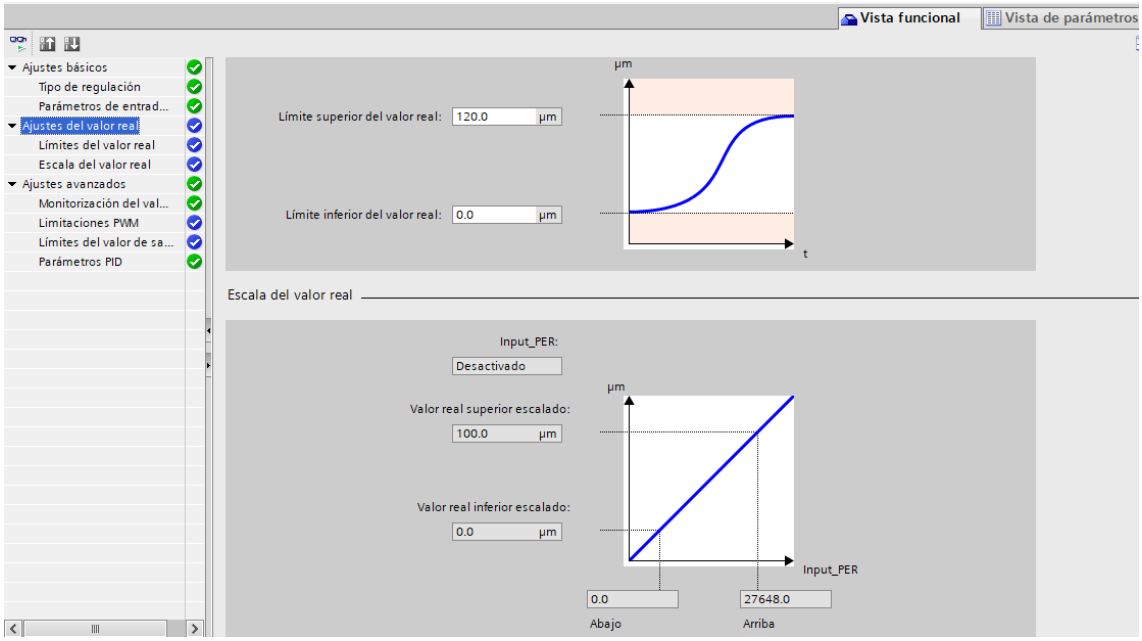


Figura 24: Configuración límites PID_Compact.

Parámetros del regulador

Una vez definida la estructura general del objeto tecnológico, se accedió al apartado destinado a la parametrización del regulador. Desde esta ventana, PID_Compact permite configurar los principales valores asociados al control PID, como la ganancia proporcional, el tiempo de integración, el tiempo derivativo, el coeficiente de retardo derivativo y las ponderaciones de las acciones proporcional y derivativa. Asimismo, permite seleccionar la estructura del regulador que se desea emplear.

En esta configuración se estableció el tiempo de muestreo del algoritmo PID en:

$$T_{PID} = 0{,}02\ s$$

Este valor se fijó para mantener la coherencia temporal con el bloque organizativo Cyclic20ms, encargado de ejecutar periódicamente el controlador cada 20 ms. La

ventana de configuración de los parámetros disponibles en PID_Compact se muestra en la Figura 25.

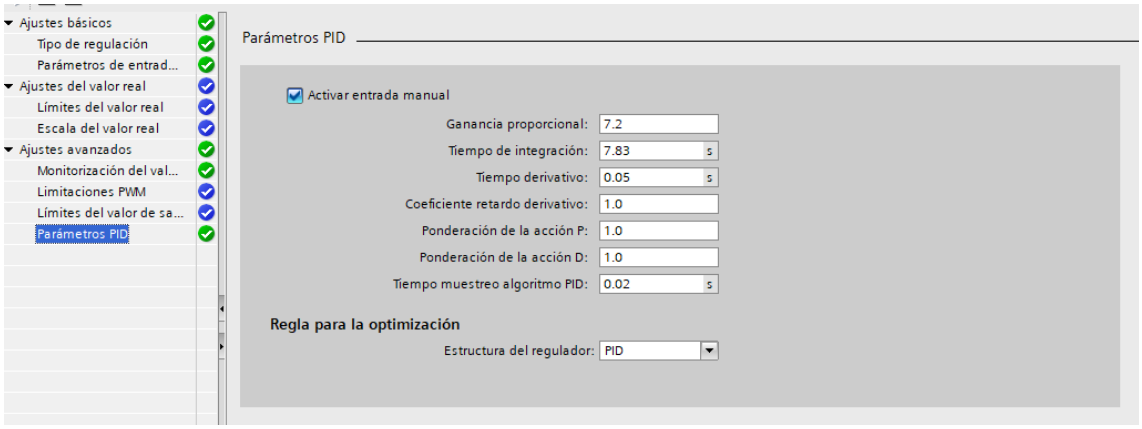


Figura 25: Parámetros PID_Compact.

3.3.7 Ajuste de los parámetros del PID

Con el fin de obtener una primera estimación de los parámetros del controlador, se desarrolló en MATLAB la función calcular_PID_tanque_optimizacion. El procedimiento implementado no se basa en una regla clásica de sintonización, sino en una optimización numérica de los parámetros del regulador a partir de la respuesta temporal simulada del sistema.

El programa parte de una representación simplificada del depósito en forma de sistema de primer orden, adecuada para obtener unos valores iniciales de regulación antes de su aplicación sobre el modelo completo. La dinámica considerada por la función se expresa en la ecuación (15):

$$\frac{dh}{dt} = \frac{1}{A} Q_{in} - \frac{1}{RA} h \tag{15}$$

donde A es el área transversal del depósito, R representa la resistencia equivalente asociada a la descarga, Q_{in} es el caudal de entrada aplicado por el regulador y h la altura del líquido.

Además, el programa incorpora un retardo L en la señal de realimentación. Este retardo se introduce con el propósito de aproximar, de forma simplificada, el efecto temporal asociado al intercambio de datos entre el modelo de planta desarrollado en Simulink y el regulador ejecutado en el PLC mediante OPC-UA. En la configuración empleada, los bloques OPC-UA Read y OPC-UA Write actualizan sus datos cada 0,02 s, por lo que se adopta inicialmente:

$$L = 0{,}02 \text{ s}$$

Este valor no pretende reproducir de manera exacta todos los efectos de la comunicación OPC-UA, sino representar un retardo característico asociado al periodo de actualización de las señales intercambiadas. En la implementación real, además del retardo temporal, intervienen otros efectos propios de la arquitectura distribuida, como el muestreo discreto, la retención de valores entre actualizaciones y la ejecución cíclica del algoritmo de control en el PLC.

Para calcular los parámetros, la función simula el depósito en lazo cerrado mediante un controlador expresado inicialmente en forma PID paralela. La ley de control utilizada por el programa se recoge en la ecuación (16):

$$Q_{in}(t) = Pe(t) + I\int e(t)\,dt + D\frac{de(t)}{dt} \tag{16}$$

donde P, I y D son las ganancias proporcional, integral y derivativa optimizadas por el programa, y $e(t)$ es el error entre la altura de referencia y la altura realimentada.

La función evalúa múltiples combinaciones de P, I y D mediante un proceso de búsqueda numérica. Para cada combinación de parámetros se simula la respuesta completa del tanque durante un intervalo de tiempo prefijado y se calcula una función de coste que penaliza los comportamientos no deseados. En particular, se consideran el sobreimpulso, el error a los 20 segundos, el error final en régimen estacionario, el error acumulado durante toda la respuesta y la aparición de ganancias excesivamente elevadas. De esta forma, el algoritmo no busca únicamente que el sistema alcance la consigna, sino que lo haga con una respuesta suficientemente rápida, estable y con bajo error residual.

Con el fin de reducir la dependencia respecto a un único punto de partida, la optimización se ejecuta desde varias condiciones iniciales diferentes. Además, los parámetros se optimizan internamente en escala logarítmica, lo que garantiza que las ganancias P, I y D obtenidas sean siempre positivas. Una vez evaluadas todas las soluciones, el programa conserva aquella que presenta el menor valor de la función de coste. En las pruebas realizadas se emplearon los parámetros:

$$A = 8; \quad R = 1; \quad L = 0{,}02 \text{ s}$$

mediante la llamada:

calcular_PID_tanque_optimizacion(8, 1, 0.02)

Al no introducirse de forma explícita la altura objetivo, la función utiliza por defecto una consigna de $H_{in} = 8$. Para esta configuración, MATLAB proporcionó los siguientes valores:

$$P = 18{,}090444 \approx 18{,}090; \quad I = 2{,}264136 \approx 2{,}264; \quad D \approx 0$$

El valor derivativo obtenido es prácticamente nulo, por lo que, para el retardo considerado, la solución óptima propuesta por el algoritmo equivale en la práctica a un regulador PI. Esto indica que la acción proporcional e integral son suficientes para alcanzar una respuesta rápida, estable y sin sobreimpulso apreciable en el modelo analizado.

Estos resultados constituyen una estimación inicial del regulador en la forma paralela expresada en la ecuación (16). Sin embargo, el objeto tecnológico PID_Compact de TIA Portal no solicita directamente las ganancias I y D en esa forma, sino que emplea como parámetros principales la ganancia proporcional K_p, el tiempo de integración T_i y el tiempo derivativo T_d. La correspondencia entre ambas representaciones se obtiene a partir de las relaciones recogidas en la ecuación (17):

$$K_p = P, T_i = \frac{P}{I}, T_d = \frac{D}{P} \tag{17}$$

Aplicando estas conversiones a los valores calculados en MATLAB se obtiene:

$$K_p \approx 18{,}090; \quad T_i = \frac{18{,}090444}{2{,}264136} \approx 7{,}99 \text{ s}; \quad T_d \approx 0 \text{ s}$$

Por tanto, los parámetros equivalentes para su introducción en PID_Compact son:

$$K_p \approx 18{,}09; \quad T_i \approx 7{,}99 \text{ s}; \quad T_d = 0 \text{ s}$$

La salida del programa de MATLAB no debe trasladarse directamente campo a campo al objeto tecnológico, sino que debe convertirse previamente a la estructura de parámetros utilizada por PID_Compact. En TIA Portal, estos resultados se introducen en los campos Ganancia proporcional, Tiempo de integración y Tiempo derivativo, definidos conforme a la parametrización propia del regulador Siemens.

No obstante, los parámetros obtenidos mediante esta optimización numérica no reproducen de forma exacta el comportamiento observado en la implementación final con PLC. Esta diferencia se debe a que el programa de MATLAB trabaja sobre un modelo simplificado, mientras que el lazo real incorpora la ejecución discreta del

regulador PID_Compact con un periodo de muestreo de 0,02 s, la comunicación OPC-UA entre Simulink y TIA Portal, la retención de las señales entre instantes de actualización y la propia estructura interna del bloque de control Siemens. Por este motivo, los valores calculados en MATLAB se emplean únicamente como una aproximación inicial de sintonización, útil para obtener un primer orden de magnitud de los parámetros, pero no como la configuración definitiva del regulador.

Herramienta de autoajuste del PID de TIA Portal

Además de la estimación previa realizada en MATLAB, TIA Portal incorpora herramientas específicas de sintonización automática para el objeto tecnológico PID_Compact. En este proyecto se empleó la función de optimización fina, orientada a refinar los parámetros del controlador a partir del comportamiento real del lazo de regulación en torno al punto de operación. Siemens indica que esta herramienta genera una oscilación limitada de la variable de proceso y utiliza la amplitud y frecuencia de dicha respuesta para recalcular los parámetros del regulador.

Durante este procedimiento, PID_Compact analiza la respuesta del sistema conectado y recalcula los parámetros del regulador a partir de la oscilación controlada de la variable de proceso. Siemens indica que la optimización fina se realiza sobre el lazo en funcionamiento y que, una vez completada correctamente, los parámetros obtenidos pasan a ser los utilizados por el controlador. Por este motivo, el ajuste automático resulta especialmente adecuado en la arquitectura desarrollada, ya que tiene en cuenta de forma implícita el comportamiento conjunto del modelo de planta en Simulink, la comunicación OPC-UA y la ejecución cíclica del regulador en el PLC.

La ventana de monitorización utilizada durante la optimización fina del regulador se muestra en la Figura 26. En ella se representan de forma simultánea la consigna, la variable de proceso y la señal de salida del controlador, lo que permite observar la evolución del sistema durante el ajuste y comprobar la estabilización del lazo.

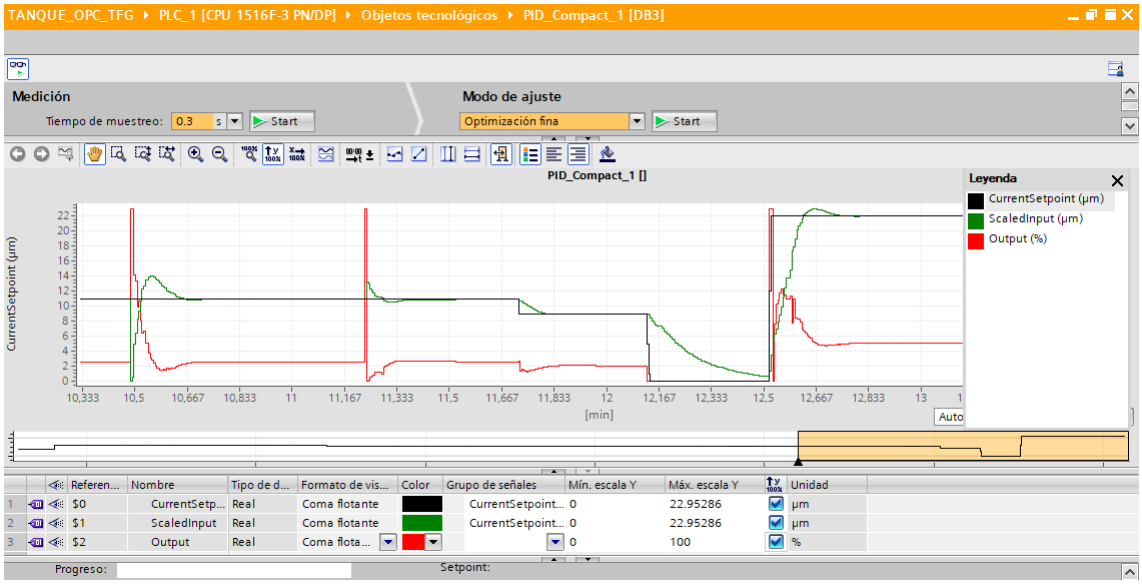


Figura 26: Función de optimización.

Como resultado de este procedimiento, TIA Portal proporcionó una nueva parametrización del controlador, más representativa del comportamiento real del sistema que la estimación previa obtenida mediante MATLAB. Los parámetros calculados automáticamente y finalmente adoptados en el objeto tecnológico PID_Compact fueron:

$$K_p \approx 12{,}01; \quad T_i \approx 3{,}37\ \text{s}; \quad T_d \approx 0{,}601\ \text{s}$$

junto con los parámetros complementarios:

$$a = 0{,}1; \quad b = 0{,}255; \quad c = 0$$

donde a es el coeficiente de retardo derivativo, b la ponderación de la acción proporcional y c la ponderación de la acción derivativa. El tiempo de muestreo del algoritmo PID se mantuvo en $T_s = 0{,}02$ s de acuerdo con la ejecución periódica del regulador en el bloque de interrupción cíclica del PLC.

Los valores resultantes del autoajuste se muestran en la Figura 27.

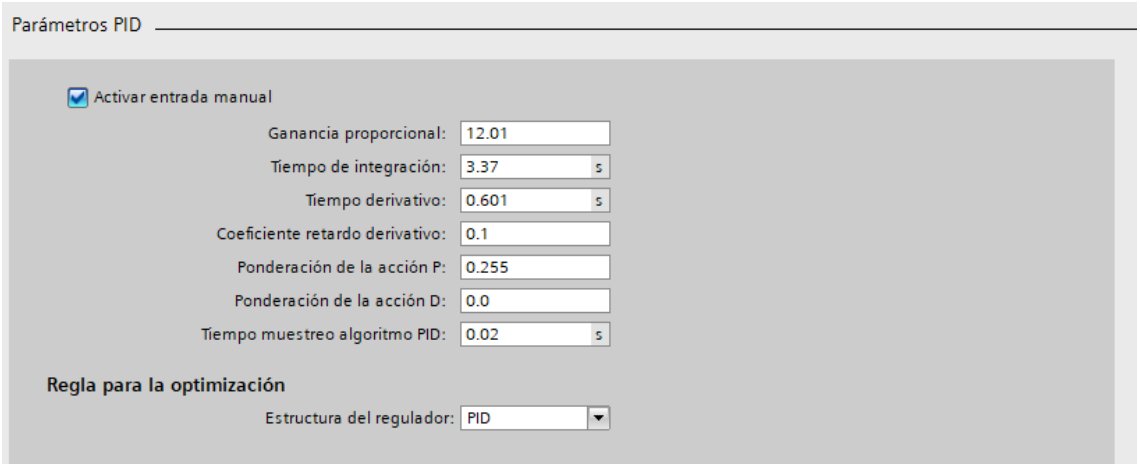


Figura 27: Valores resultantes del autoajuste.

La elección de estos parámetros se justifica porque fueron obtenidos directamente sobre la arquitectura de control finalmente empleada, incluyendo el intercambio de datos mediante OPC-UA y la ejecución discreta del regulador en el PLC. Frente a la sintonización inicial calculada en MATLAB, que se basa en un modelo simplificado del lazo cerrado, la optimización fina de TIA Portal proporciona una parametrización ajustada al comportamiento efectivo del sistema implementado. Por ello, los parámetros calculados automáticamente se adoptaron como configuración final del controlador.

Tras su aplicación, se comprobó que el sistema presentaba una respuesta estable, con seguimiento adecuado de la consigna y un sobreimpulso moderado, manteniendo un compromiso satisfactorio entre rapidez de respuesta y amortiguación.

3.3.8 Relación entre la comunicación OPC-UA y el lazo de control

La implementación desarrollada permite cerrar el lazo de control de forma distribuida entre TIA Portal y Simulink. En cada periodo de ejecución, el regulador PID_Compact calcula una acción de control y la almacena en salidaPID. El modelo de Simulink lee esta variable, la aplica al depósito virtual y obtiene una nueva altura de nivel. Dicho valor se escribe en entradaPID, quedando disponible para que el PLC lo emplee como variable de proceso en el siguiente ciclo.

Esta organización convierte al PLC en el controlador real del sistema y a Simulink en la representación dinámica de la planta. La comunicación OPC-UA no actúa únicamente como un mecanismo de transferencia de datos, sino como el elemento que hace posible la validación del control sobre una planta virtual en condiciones cercanas a las de una implementación real. La sincronización entre el ciclo de ejecución del PLC, el tiempo de muestreo de Simulink y la actualización de las señales OPC-UA permite ensayar el comportamiento del sistema de regulación, comprobar la

respuesta frente a cambios de consigna y analizar el efecto de perturbaciones antes de trabajar sobre una instalación física.

3.4 Implementación del sistema y supervisión

Una vez configurada la comunicación OPC-UA y parametrizado el regulador PID_Compact, se desarrolló en TIA Portal la lógica complementaria necesaria para operar el sistema de forma controlada y facilitar su supervisión desde una interfaz HMI. Mientras que el objeto tecnológico se encarga del cálculo de la acción de control, el programa del PLC incorpora las funciones auxiliares que permiten gestionar la puesta en marcha, modificar la consigna, seleccionar modos de funcionamiento e introducir determinadas acciones de prueba sobre el sistema.

La implementación realizada no se limita, por tanto, a ejecutar un regulador PID de forma aislada, sino que conforma una pequeña aplicación de automatización completa. En ella, el PLC actúa como núcleo de control, Simulink reproduce la dinámica de la planta virtual, OPC-UA permite el intercambio de variables entre ambos entornos y la HMI proporciona al usuario un medio de interacción directa con el proceso.

3.4.1 Programa de operación desarrollado en TIA Portal

La lógica general de funcionamiento se implementó en el bloque principal Main [OB1], programado en lenguaje KOP. Este bloque se organiza en varios segmentos, cada uno de ellos destinado a resolver una parte concreta de la operación del sistema: inicialización, gestión de los estados de encendido y apagado, activación de perturbaciones, mantenimiento temporal de dichas perturbaciones y validación de los valores introducidos por el usuario. La división en redes diferenciadas permite estructurar el programa de forma clara y facilita la comprensión de la secuencia global de funcionamiento.

De forma general, el programa trabaja con dos estados principales: sistema apagado y sistema encendido. Cuando el sistema se encuentra apagado, se fuerza una condición segura de operación, estableciendo la consigna del regulador en cero y fijando la variable perturbacionopc en su valor inicial. Desde este estado, la activación del pulsador de encendido desde la HMI permite pasar al modo de funcionamiento normal.

Cuando el sistema se encuentra encendido, la lógica habilita la transferencia de la consigna definida por el usuario hacia la variable setpoint_PID, que es la que posteriormente utiliza el bloque PID_Compact como referencia. En este estado también queda habilitada la aplicación de perturbaciones. Para ello, el operador introduce desde la HMI el valor deseado y activa la orden correspondiente mediante

el pulsador de perturbación. Esta acción genera una transición interna que inicia el cálculo del nuevo valor a aplicar sobre la señal perturbacionopc.

La perturbación no se aplica únicamente de forma instantánea, sino que queda mantenida durante un intervalo temporal específico. Tras recibir la orden desde la HMI, el programa calcula el nuevo valor a partir del valor actual de perturbacionopc y del incremento o decremento introducido por el usuario. A continuación, se activa el estado M_Comparacion, que da paso a la fase de validación y aplicación de la perturbación. Mientras este estado permanece activo, el valor calculado se mantiene sobre OpcInterface.perturbacionopc, de modo que la perturbación continúa actuando sobre el sistema durante el tiempo establecido.

Antes de aplicar definitivamente la nueva perturbación, el programa comprueba que el valor calculado se mantiene dentro del rango permitido. Si el resultado queda por debajo de $-0{,}9$ o supera $2{,}0$, se satura automáticamente en el límite correspondiente y se activa la variable Valor_invalido, utilizada posteriormente en la HMI para advertir al operador de que el valor solicitado no era admisible. Si el valor se encuentra dentro del intervalo permitido, se transfiere directamente a OpcInterface.perturbacionopc.

La lógica incorpora dos temporizaciones auxiliares. En primer lugar, se emplea un retardo breve de seguridad de 0,1 s antes de ejecutar la validación del valor calculado. En segundo lugar, el estado M_Comparacion se mantiene activo durante 6 s, periodo durante el cual la perturbación permanece aplicada sobre el sistema. Una vez finalizada esta temporización, el programa desactiva dicho estado y retorna al funcionamiento normal.

El funcionamiento global del programa puede resumirse mediante el diagrama de flujo de la Figura 28.

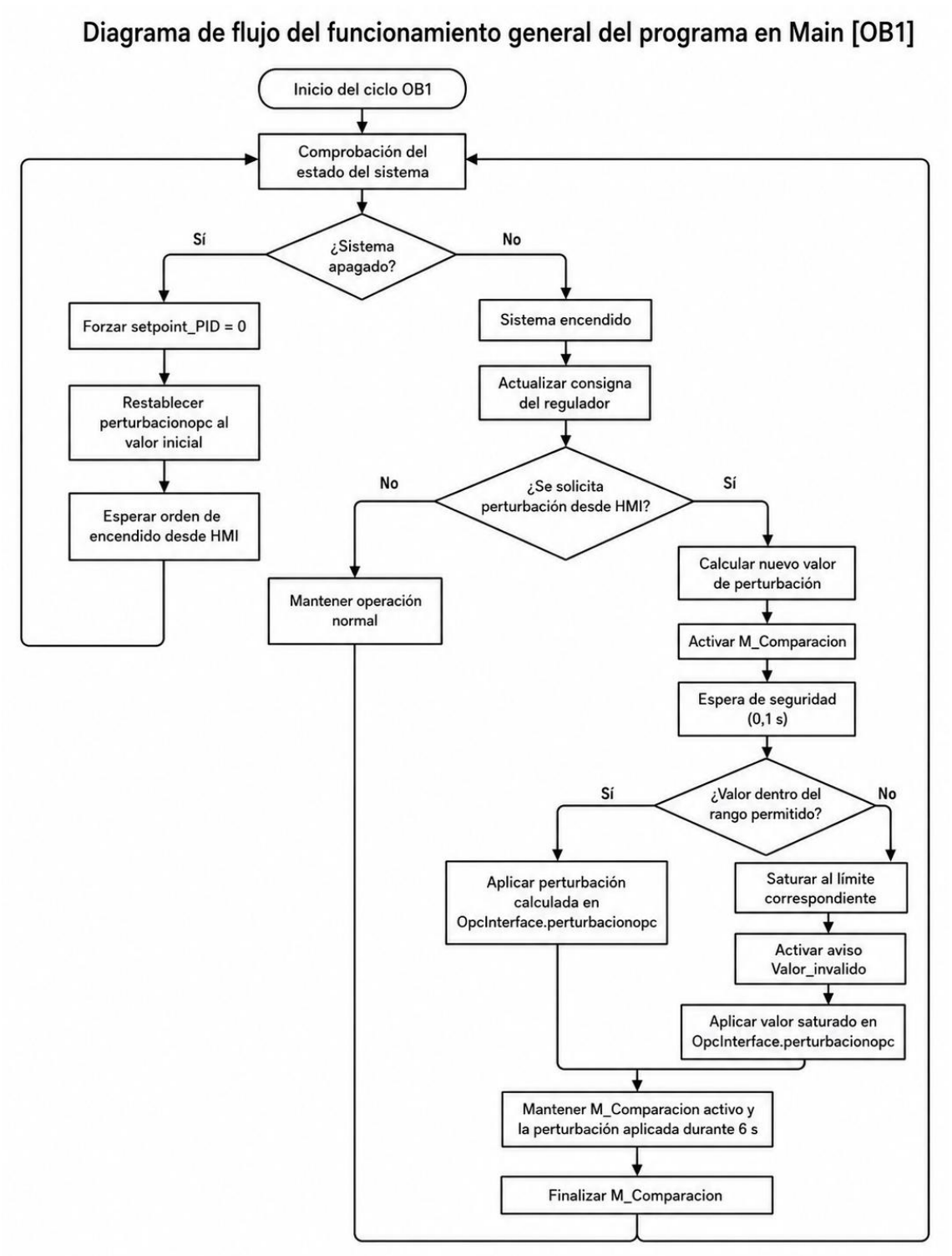


Figura 28: Diagrama de flujo del programa de operación.

Este programa actúa como lógica de supervisión y operación del sistema. El cálculo del control propiamente dicho continúa realizándose en el objeto tecnológico PID_Compact, mientras que el bloque Main [OB1] organiza las condiciones de uso, la

interacción con la HMI, la generación y mantenimiento temporal de perturbaciones y la gestión de las señales auxiliares necesarias para las pruebas de funcionamiento.

3.4.2 Interfaz HMI para la operación y supervisión del sistema

Con el objetivo de facilitar la interacción con el proceso, se desarrolló una interfaz HMI asociada al proyecto de TIA Portal. Esta interfaz permite actuar sobre el sistema de forma gráfica, supervisar su estado y consultar la evolución de las variables principales sin necesidad de acceder directamente al programa del PLC.

La HMI se organiza en varias pantallas, cada una de ellas con una finalidad específica. La navegación entre pantallas se realiza mediante botones integrados en la interfaz, de forma que el usuario puede acceder desde la pantalla principal a los apartados de información del proyecto, información del sistema y monitorización de las variables PID-OPC.

Pantalla de inicio

La primera pantalla de la interfaz corresponde a la imagen raíz o pantalla de inicio de la HMI. Esta pantalla actúa como punto de acceso inicial a la aplicación y presenta una portada identificativa del proyecto, incluyendo el título del trabajo, la finalidad general del sistema, el estándar de comunicación empleado y los elementos principales de la arquitectura desarrollada: el depósito virtual, la comunicación OPC-UA y el PLC Siemens. Además, incorpora el botón de acceso a las imágenes de sistema, desde el cual el usuario puede pasar a la pantalla principal de operación y supervisión del proceso.

La pantalla de inicio de la interfaz HMI se muestra en la Figura 29.

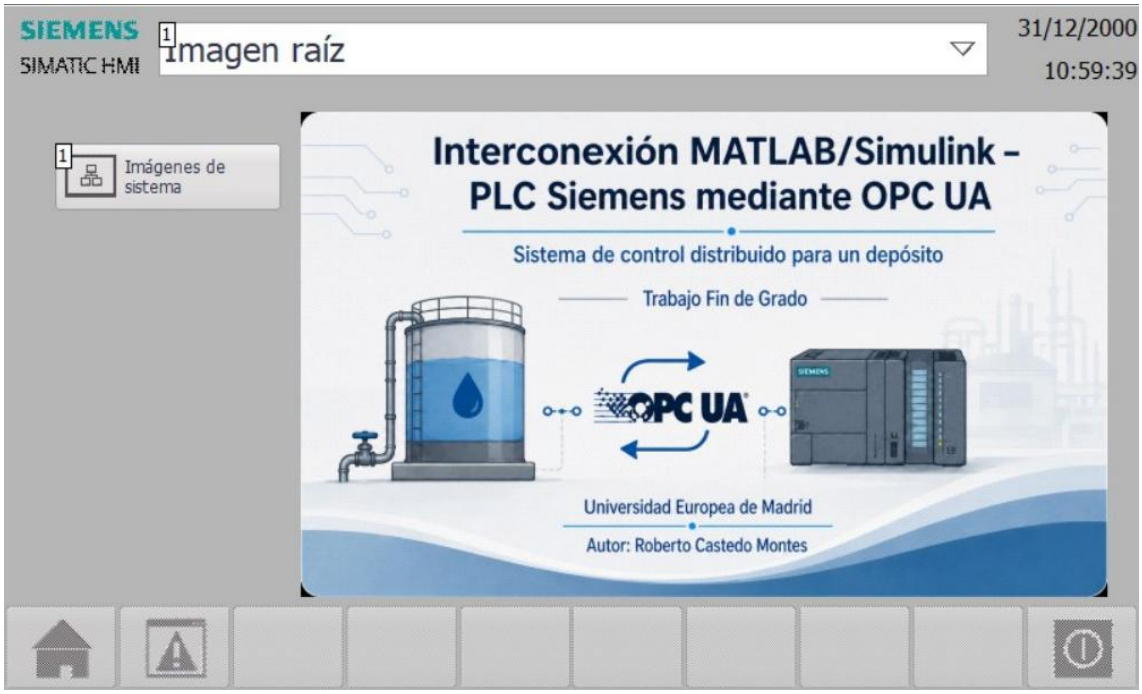


Figura 29: Captura de la pantalla inicial.

Pantalla principal de operación

La pantalla denominada Imágenes de sistema constituye la vista principal de operación. En ella se concentra la interacción directa con el proceso y se representa de forma esquemática el depósito controlado. Esta pantalla incorpora los elementos necesarios para poner en marcha o detener el sistema, establecer la consigna de funcionamiento, introducir perturbaciones y observar de forma inmediata el estado general de la aplicación.

En la zona de operación se incluyen dos pulsadores de control, ON y OFF, vinculados a las órdenes de encendido y parada del sistema. Junto a ellos se dispone de un indicador visual de estado, representado mediante señalización de color, que permite identificar rápidamente si la instalación se encuentra habilitada o detenida. Cuando el sistema se encuentra apagado, el indicador aparece en color rojo; cuando el sistema está habilitado, el indicador pasa a color verde.

El estado de funcionamiento normal del sistema desde la pantalla principal de la HMI se muestra en la Figura 30. En esta condición, el indicador superior aparece en color verde, la consigna se encuentra aplicada al regulador y el depósito muestra gráficamente el nivel correspondiente al valor controlado.



Figura 30: Captura de la pantalla principal de la HMI.

La consigna del regulador puede modificarse desde esta misma pantalla mediante un deslizador asociado a la referencia de nivel. El valor introducido se utiliza posteriormente en la lógica del PLC para actualizar la señal de consigna que recibe el PID_Compact. Además, se muestra de forma numérica el valor aplicado, facilitando una lectura precisa del objetivo definido por el usuario.

La pantalla también incorpora una zona destinada a la introducción de perturbaciones. El operador puede escribir el valor deseado y activar su aplicación mediante el botón Perturbación. La lógica del PLC procesa esta orden y comprueba si el valor resultante se encuentra dentro de los límites establecidos. Cuando el valor introducido queda fuera del rango permitido, el programa no descarta la orden, sino que aplica la perturbación saturándola al límite correspondiente, máximo o mínimo, según el valor solicitado. De esta forma, se evita introducir una condición no admisible en el modelo, pero se mantiene la ejecución de la prueba con el valor límite permitido.

Esta condición se refleja visualmente en la HMI mediante el cambio de apariencia del campo de perturbación. Si el valor introducido es válido, el campo mantiene su aspecto normal; en cambio, si el valor solicitado supera los límites definidos, el campo aparece en color rojo para indicar que la entrada no ha sido aceptada de forma directa y que

el PLC ha aplicado una saturación. Además, mientras la perturbación permanece activa, la válvula de salida cambia de color gris a azul, proporcionando una indicación visual directa de que se está modificando temporalmente la dinámica de descarga del depósito.

Un ejemplo de aplicación de una perturbación con valor no válido se muestra en la Figura 31. En esta situación, el campo de perturbación aparece en rojo para indicar que el valor introducido queda fuera del rango permitido. No obstante, la válvula de salida aparece en color azul mientras dura la perturbación, ya que el programa aplica igualmente la perturbación utilizando el valor límite correspondiente. De este modo, la HMI informa simultáneamente de dos estados: por un lado, que el valor introducido no es admisible, y por otro, que la perturbación se encuentra activa sobre el término asociado a la descarga del tanque.



Figura 31: Captura de la señalización del valor de perturbación no válido en la pantalla principal.

Por último, la pantalla principal incluye botones de navegación hacia las páginas de información del proyecto, información del sistema y seguimiento PID-OPC, facilitando el acceso ordenado al resto de funciones de supervisión.

La pantalla principal de operación de la HMI se muestra en la Figura 32.

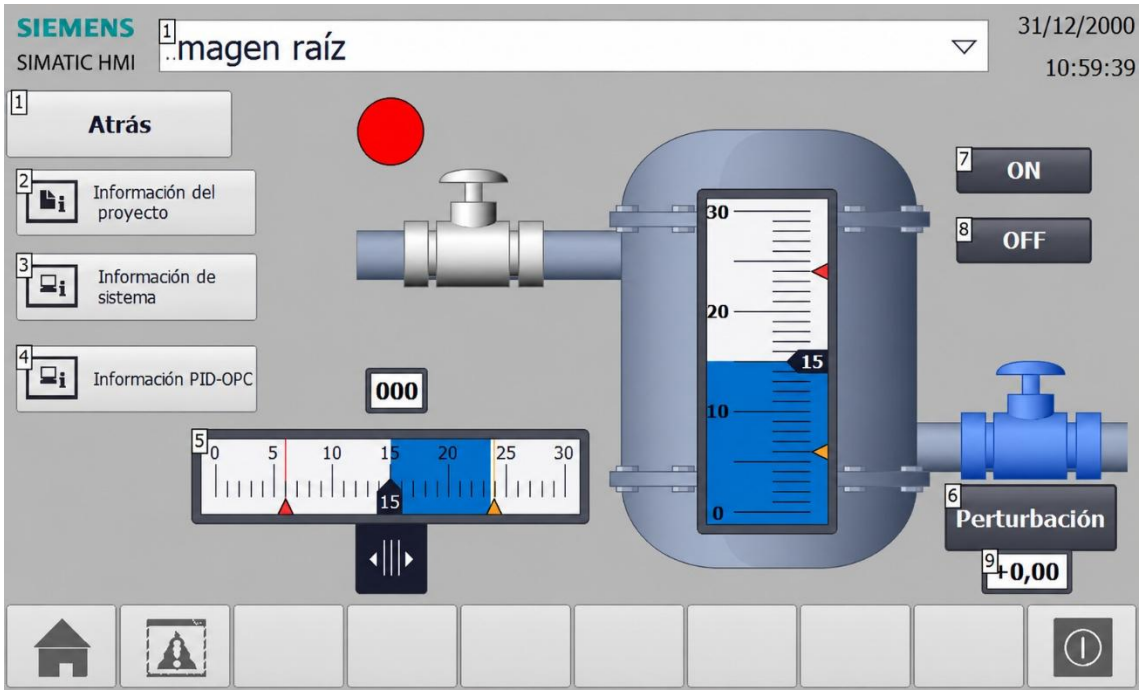


Figura 32: Captura de la pantalla principal del sistema de control.

Pantalla de información del sistema

La pantalla Información de sistema se utiliza para presentar datos generales sobre la plataforma de supervisión y el entorno de automatización empleado. En ella se recoge el tipo de panel HMI, la versión asociada, el tipo de conexión utilizada y la familia del controlador configurado. En la implementación desarrollada se identifica un panel TP700 Comfort, comunicación mediante Ethernet y un controlador SIMATIC S7-1500.

Aunque esta pantalla no interviene directamente en la operación del lazo de control, aporta información útil para documentar la configuración del sistema y verificar, desde la propia interfaz, los elementos principales que forman parte de la arquitectura implementada.

La pantalla de información del sistema se muestra en la Figura 33.

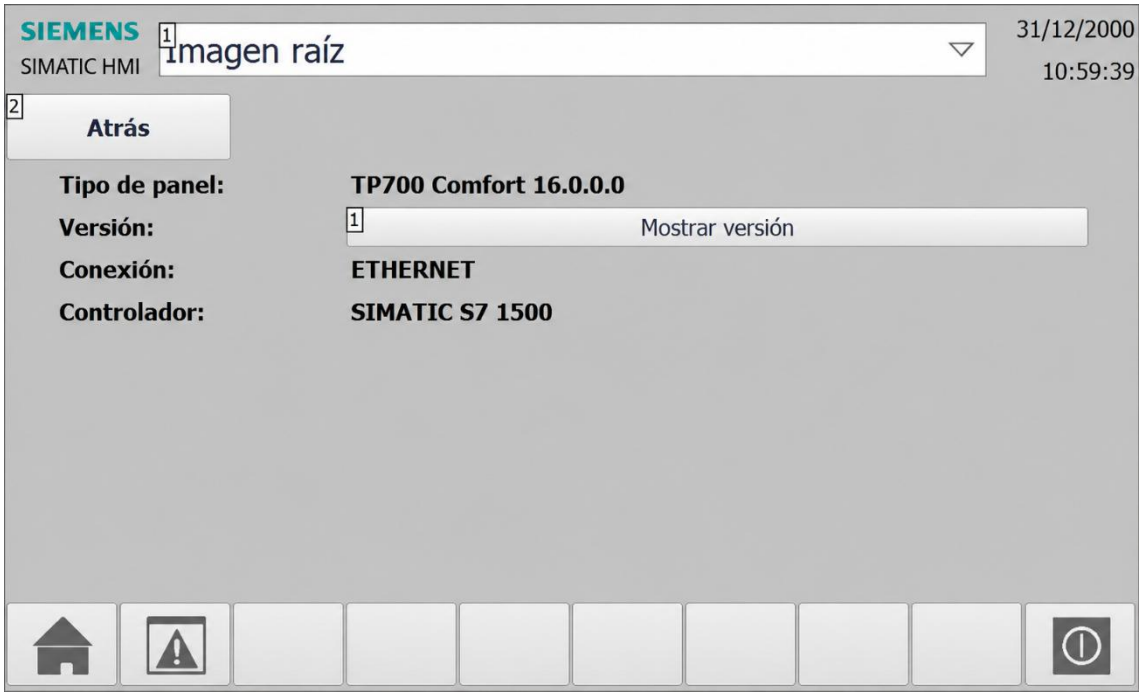


Figura 33: Captura de la pantalla de Información del sistema.

Pantalla de créditos de autoría

La pantalla Información del proyecto recoge los datos identificativos de la aplicación desarrollada. En ella se presenta el nombre del proyecto, la fecha de creación, el autor, el tutor y una breve descripción general del trabajo. Esta pantalla cumple una función principalmente documental, integrando dentro de la propia HMI la información básica del desarrollo implementado.

La pantalla de información del proyecto se muestra en la Figura 34.

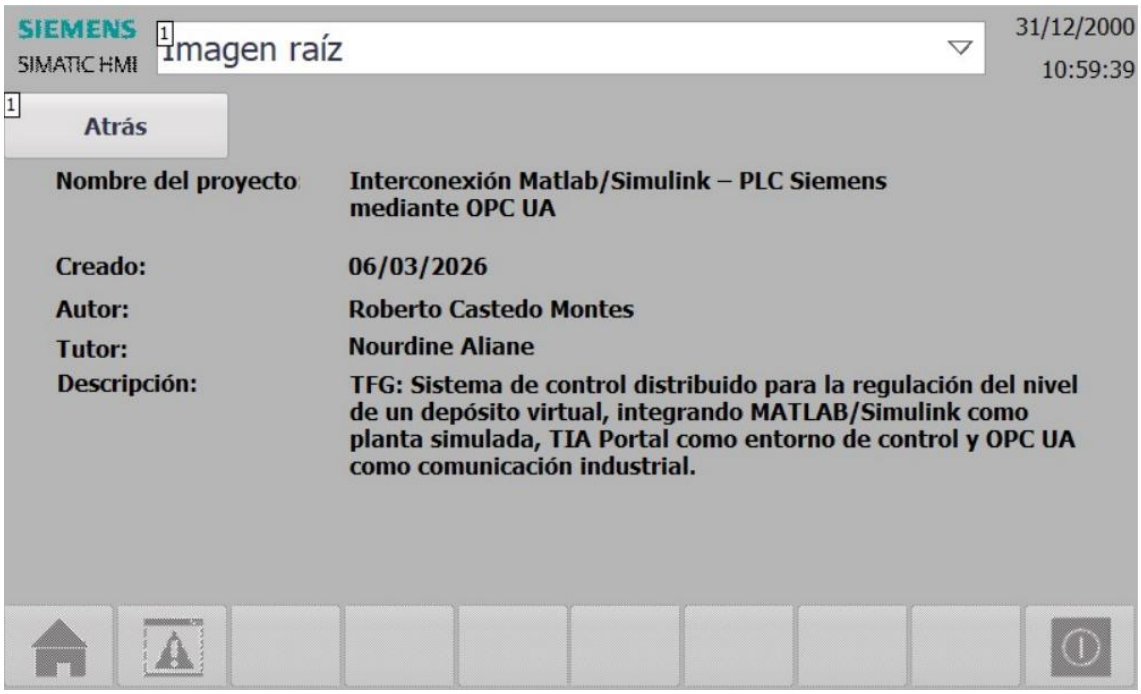


Figura 34: Captura de la pantalla de créditos de autoría.

Pantalla de monitorización PID-OPC

La pantalla Información PID-OPC se diseñó para supervisar de forma más detallada el comportamiento del lazo de control y el intercambio de variables entre el PLC y el modelo desarrollado en Simulink. Su elemento principal es un visor de curvas en tiempo real que representa simultáneamente tres señales:

- La consigna del sistema, asociada a Global_setpoint.
- La variable de proceso recibida desde Simulink, asociada a OpcInterface_entradaPID.
- La salida generada por el regulador, asociada a OpcInterface_salidaPID.

Estas tres curvas permiten observar la relación entre la referencia introducida, la respuesta efectiva del modelo de planta y la acción de control calculada por el PLC. De este modo, la pantalla facilita la valoración visual del comportamiento dinámico del sistema durante las pruebas de seguimiento de consigna y ante la aplicación de perturbaciones.

Además del visor de tendencias, la pantalla incorpora campos numéricos que muestran los valores instantáneos de las señales principales, permitiendo

complementar la lectura gráfica con una supervisión precisa de los datos intercambiados por OPC-UA.

La pantalla de monitorización permite comprobar en tiempo real la relación entre la consigna, la variable de proceso recibida desde Simulink y la salida generada por el regulador. Un ejemplo de esta supervisión durante el funcionamiento del sistema se muestra en la Figura 35.

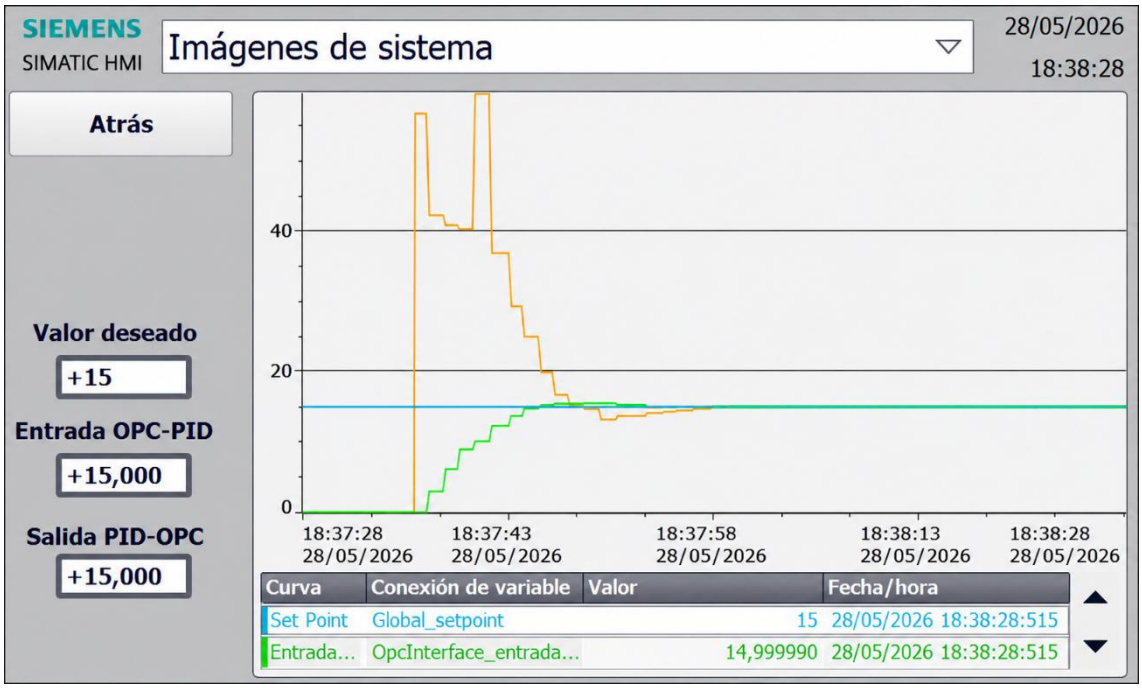


Figura 35: Captura de la pantalla de monitorización PID-OPC con evolución de consigna, variable de proceso y salida del regulador.

3.4.3 Funcionamiento general de la aplicación

Una vez configurados el modelo de Simulink, el proyecto de TIA Portal y la comunicación OPC-UA, la puesta en marcha de la aplicación requiere seguir una secuencia ordenada de inicialización. Este procedimiento se planteó de forma análoga al caso de uso de Siemens para la conexión de modelos de Simulink con SIMATIC PLCSIM Advanced mediante OPC-UA, adaptándolo a la arquitectura desarrollada en este trabajo.

En primer lugar, debe iniciarse PLCSIM Advanced y crearse la instancia del controlador virtual que actuará como PLC simulado. Para ello, se selecciona la interfaz virtual correspondiente y se configura la instancia con la dirección IP asignada a la CPU dentro del proyecto de TIA Portal. En la implementación desarrollada, la instancia virtual recibe el nombre TanqueOPC y queda asociada a la dirección 192.168.1.1, correspondiente al PLC simulado utilizado durante las pruebas.

La configuración de la instancia virtual del PLC en PLCSIM Advanced se muestra en la Figura 36.

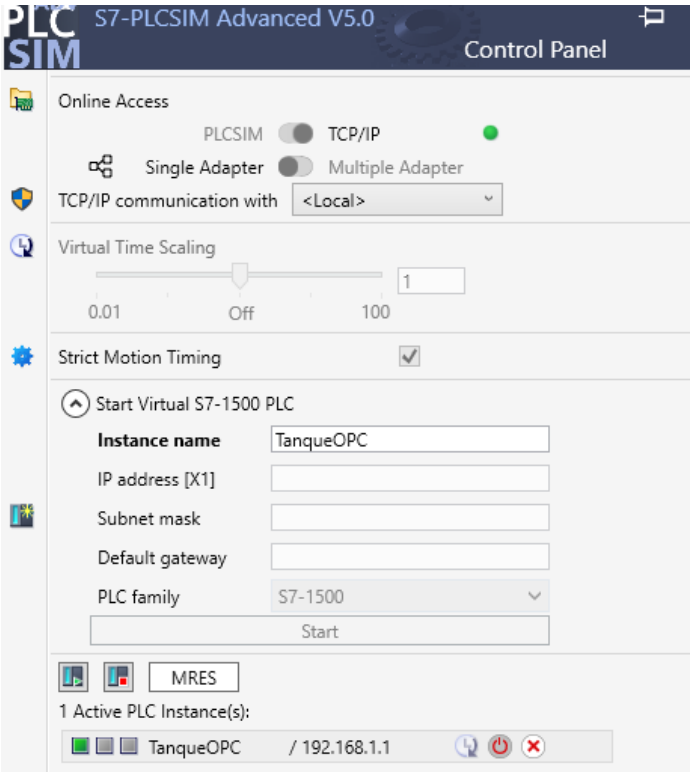


Figura 36: Captura de la configuración de la instancia virtual del PLC en PLCSIM Advanced.

Una vez creada la instancia, esta se pone en funcionamiento desde el panel de control de PLCSIM Advanced, quedando preparada para recibir el programa del autómata. A continuación, desde TIA Portal se carga el proyecto en el controlador virtual. Para ello, se selecciona la CPU del proyecto y se realiza una descarga extendida al dispositivo, escogiendo como destino la instancia de PLCSIM Advanced previamente iniciada. Tras completarse la transferencia, el controlador virtual se establece en modo RUN, de forma que los bloques organizativos, el regulador PID_Compact y las variables empleadas para la comunicación queden activos. Este paso permite que el PLC simulado ejecute la lógica de control como si se tratase de un equipo físico.

Con el PLC virtual ya operativo, se abre el modelo de Simulink y se accede al apartado de configuración OPC-UA del propio modelo. En esta ventana debe comprobarse que aparece registrado el servidor OPC-UA asociado a la CPU simulada y, además, debe establecerse manualmente la conexión con dicho servidor antes de iniciar la simulación. En el caso desarrollado, el cliente OPC-UA de Simulink se conecta al servidor SIMATIC.S7-1500.OPC-UA.Application:PLC_1, alojado en el host 192.168.1.1. Cuando la conexión se establece correctamente, el estado del servidor aparece como Connected.

La conexión activa del cliente OPC-UA en Simulink con el servidor del PLC simulado se muestra en la Figura 37.

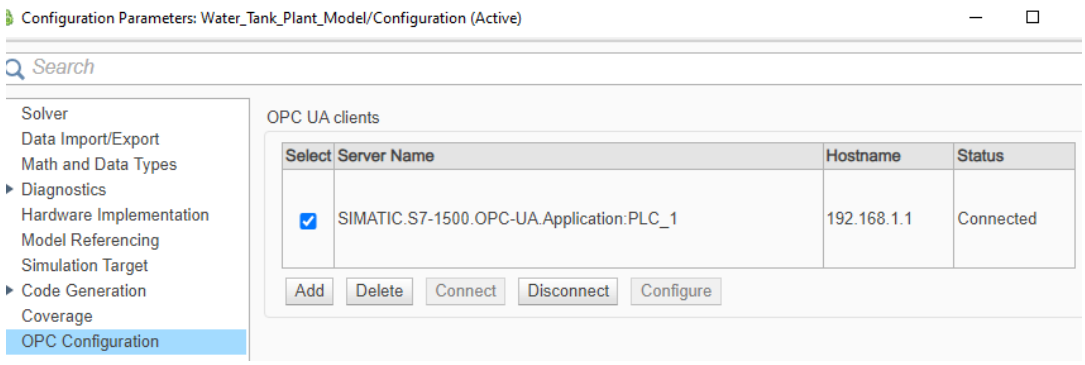


Figura 37: Captura de conexión del cliente OPC-UA de Simulink con el servidor del PLC simulado.

Esta acción es necesaria porque, en la configuración empleada, el modelo de Simulink no realiza la conexión al servidor OPC-UA de forma automática al arrancar. Si el servidor permanece en estado Disconnected, los bloques OPC-UA Read y OPC-UA Write no pueden acceder correctamente a las variables del PLC y la simulación genera errores de comunicación.

Una vez establecida la conexión con el servidor OPC-UA, también debe verificarse que los bloques OPC-UA Read y OPC-UA Write estén asociados al servidor correcto y que los nodos seleccionados correspondan con las variables de intercambio definidas en el bloque de datos OpcInterface. Esta comprobación garantiza que Simulink pueda leer la acción de control calculada por el PLC y escribir de vuelta las variables procedentes de la planta virtual.

Después de confirmar la conexión OPC-UA, se inicia la simulación del modelo de Simulink. A partir de este momento, el modelo virtual del depósito comienza a evolucionar y se establece el intercambio cíclico de información con el PLC. El regulador implementado en TIA Portal calcula la acción de control a partir de la consigna y del nivel recibido, y dicha salida se transmite a Simulink mediante OPC-UA. El modelo de planta utiliza esta señal como caudal de entrada, actualiza la altura del líquido y devuelve el nuevo valor al PLC, cerrando así el lazo de control.

Por último, se inicia la simulación de la interfaz HMI desde TIA Portal, que constituye el punto principal de operación del sistema. A través de ella, el usuario puede activar o detener el funcionamiento general de la aplicación, modificar la consigna de nivel y ejecutar las pruebas de perturbación previstas. La aplicación se organiza en torno a dos estados principales: sistema apagado y sistema encendido. Al arrancar, el sistema permanece inicialmente apagado, manteniendo la consigna del regulador en cero y evitando que el controlador actúe sobre la planta virtual. Desde la HMI, el usuario puede habilitar el funcionamiento mediante el pulsador de encendido; esta orden es

procesada por el PLC, que activa el estado de operación y permite que la consigna seleccionada pase a ser utilizada por el regulador PID_Compact.

Una vez encendido el sistema, el usuario puede modificar la consigna de nivel mediante el elemento de entrada dispuesto en la HMI. El valor introducido se almacena en el PLC como referencia temporal y, mientras el sistema permanece activo, se transfiere a la consigna efectiva del regulador. De este modo, cualquier cambio de nivel solicitado desde la interfaz se traduce en una nueva referencia para el lazo de control. El PLC compara dicha consigna con la altura real recibida desde Simulink y calcula la acción de control necesaria para corregir la diferencia. La respuesta del sistema puede observarse tanto en la HMI como en la interfaz gráfica desarrollada en MATLAB/Simulink, donde se representa visualmente el nivel del depósito y su evolución temporal.

La interfaz gráfica asociada al modelo de Simulink actúa como apoyo visual a la simulación. En ella se muestra una representación animada del depósito, cuya altura de agua se actualiza conforme evoluciona la variable h. De forma simultánea, se presenta una gráfica en tiempo real de la evolución del nivel y un indicador numérico con el valor instantáneo de la altura. También se visualiza el estado de la simulación, diferenciando entre ejecución activa y parada. Esta representación permite comprobar de manera directa cómo responde la planta virtual ante cada modificación introducida desde la HMI, facilitando la interpretación del comportamiento dinámico del sistema.

El apagado del sistema se realiza desde la HMI mediante el pulsador correspondiente. Al recibirse esta orden, el PLC abandona el estado de funcionamiento activo, fuerza nuevamente la consigna del regulador a cero y devuelve las variables auxiliares a su condición normal de reposo. De esta manera, el sistema queda preparado para una nueva puesta en marcha sin conservar órdenes de control anteriores que puedan alterar el siguiente ensayo. La transición entre los estados de encendido y apagado permite reproducir un comportamiento similar al de una instalación industrial en la que el operador habilita o detiene el proceso desde una pantalla de supervisión.

Además del control del nivel mediante cambios de consigna, la aplicación permite ejecutar pruebas de perturbación sobre el proceso. Estas pruebas se realizan modificando temporalmente el valor asociado a la descarga del depósito. En concreto, el usuario introduce desde la HMI una variación que se suma al parámetro 1/R, el cual interviene en el término de salida de la ecuación dinámica del tanque. Al activar la orden de perturbación, el PLC calcula el nuevo valor resultante y comprueba que permanezca dentro de los límites establecidos para evitar configuraciones no admisibles. Si la entrada supera dichos márgenes, el programa fuerza el valor al límite correspondiente y activa una señal de aviso de valor no válido. La lógica que implementa esta secuencia, junto con la temporización y la validación de los límites, se desarrolla en el bloque principal del PLC.

Cuando la perturbación es aceptada, el nuevo valor se aplica durante un intervalo temporal definido. Durante ese periodo, la dinámica de descarga del depósito se

modifica y el nivel tiende a desviarse de la consigna. El regulador PID_Compact debe entonces compensar dicha variación actuando sobre el caudal de entrada, lo que permite comprobar la capacidad del lazo de control para rechazar perturbaciones externas. Una vez finalizado el tiempo de prueba, el PLC restablece automáticamente la condición nominal de operación, retirando la perturbación y devolviendo el sistema a su régimen normal. Esta funcionalidad permite evaluar no solo el seguimiento de consignas, sino también la robustez del controlador frente a cambios temporales en las condiciones del proceso.

Durante toda la operación, la HMI cumple una doble función. Por un lado, actúa como interfaz de mando, desde la que se realizan las acciones de encendido, apagado, modificación de consigna y activación de perturbaciones. Por otro, sirve como herramienta de supervisión, permitiendo observar las variables principales del proceso y el estado de las órdenes ejecutadas. De forma complementaria, la animación de Simulink ofrece una representación visual más intuitiva del comportamiento físico del depósito, mientras que la gráfica temporal facilita el análisis de la evolución de la altura ante cada acción del usuario.

En consecuencia, la aplicación desarrollada integra en un único entorno funcional la simulación de la planta, el control industrial ejecutado en un PLC virtual, el intercambio de información mediante OPC-UA, la operación desde una HMI y la visualización dinámica del proceso. Esta estructura permite reproducir de forma coordinada el funcionamiento de un lazo de regulación industrial, analizar la respuesta ante cambios de consigna, verificar el rechazo de perturbaciones y disponer de una supervisión gráfica clara del comportamiento del sistema.

Capítulo 4 Pruebas y validación

El presente capítulo recoge el conjunto de pruebas realizadas para verificar el funcionamiento del sistema desarrollado. La validación se plantea de forma progresiva, comenzando por la comprobación de la comunicación entre MATLAB/Simulink y TIA Portal mediante OPC-UA, continuando con la verificación del modelo dinámico y de la animación asociada, y finalizando con el análisis del comportamiento del lazo de control tanto en simulación como en la arquitectura integrada.

Este planteamiento permite comprobar por separado cada uno de los elementos principales del proyecto antes de evaluar el funcionamiento global del sistema. De este modo, se busca confirmar que la comunicación es correcta, que el modelo representa adecuadamente la evolución del nivel del depósito, que la visualización en tiempo real es coherente con dicha evolución y que el controlador PID es capaz de regular el proceso de forma satisfactoria ante diferentes condiciones de operación.

Para organizar la validación, las pruebas se han agrupado en varios bloques, cada uno orientado a verificar una parte concreta de la arquitectura desarrollada. En la Tabla 2 se resumen las pruebas realizadas y el objetivo principal de cada una.

Tabla 2: Resumen de las pruebas de validación realizadas.

Código	Objetivo de la prueba
P1	Validar la comunicación OPC-UA en el sentido Simulink → TIA Portal, comprobando que una señal generada en Simulink puede escribirse correctamente en una variable del PLC.
P2	Validar la comunicación OPC-UA en el sentido TIA Portal → Simulink, comprobando que una señal generada en el PLC puede leerse correctamente desde Simulink.
P3	Comprobar la coherencia entre el modelo dinámico del depósito y la animación desarrollada en MATLAB, verificando que ambas representaciones muestran la misma evolución de la altura.

Código	Objetivo de la prueba
P4	Evaluar el comportamiento del controlador PID implementado íntegramente en MATLAB/Simulink, utilizado como referencia inicial del sistema de control.
P5	Evaluar el comportamiento del controlador PID_Compact implementado en TIA Portal, con el modelo del depósito ejecutándose en Simulink y el lazo cerrado mediante OPC-UA.
P6	Comparar la respuesta del PID de Simulink con la respuesta del PID_Compact de TIA Portal, analizando las diferencias derivadas de la arquitectura distribuida.
P7	Validar el funcionamiento global del sistema ante cambios de consigna, comprobando que el nivel del depósito puede regularse en distintos puntos de operación.
P8	Evaluar la respuesta del sistema ante perturbaciones aplicadas sobre la dinámica de descarga del depósito, comprobando la capacidad de recuperación del regulador.

A continuación, se desarrolla cada una de estas pruebas, indicando el procedimiento seguido, las evidencias gráficas obtenidas y la interpretación de los resultados. Las capturas y gráficas se incorporan en el apartado correspondiente a cada ensayo, de forma que puedan relacionarse directamente con la prueba que justifican. De este modo, las figuras no se presentan únicamente como elementos ilustrativos, sino como evidencias del comportamiento observado durante la validación del sistema.

4.1 Definición de las pruebas

En este apartado se describen las pruebas planteadas para validar el sistema desarrollado, indicando el objetivo de cada una, el procedimiento seguido y los aspectos que se pretenden comprobar. Las pruebas se agrupan en cuatro bloques: validación de la comunicación OPC-UA, comprobación del modelo y de la animación, evaluación del controlador PID y validación del funcionamiento global del sistema.

4.1.1 Pruebas de comunicación mediante OPC-UA

El primer conjunto de ensayos tiene como objetivo verificar el intercambio correcto de variables entre MATLAB/Simulink y TIA Portal a través del protocolo OPC-UA. Antes de analizar el comportamiento del sistema completo, resulta necesario comprobar que ambos entornos son capaces de enviar y recibir información sin alteraciones significativas.

Prueba P1. Envío de datos desde Simulink hacia TIA Portal

En esta prueba se genera en Simulink una señal senoidal conocida, que se transmite al PLC mediante un bloque de escritura OPC-UA. La señal recibida se monitoriza desde TIA Portal con el fin de comprobar que reproduce correctamente la forma, amplitud y frecuencia de la señal original.

Objetivo de la prueba:

- Verificar que la comunicación en el sentido Simulink → TIA Portal se realiza correctamente y que los datos enviados desde el modelo llegan al PLC sin modificaciones apreciables.

Evidencias utilizadas para la validación:

- Captura del generador de señal en Simulink.
- Captura de la variable recibida en TIA Portal.
- Comparación visual entre ambas señales.

Prueba P2. Envío de datos desde TIA Portal hacia Simulink

De forma complementaria, se realiza la prueba inversa. En este caso, la señal senoidal se genera desde el entorno de TIA Portal y se envía a Simulink mediante OPC-UA. La señal recibida se representa en un Scope para comprobar que mantiene el comportamiento esperado.

Objetivo de la prueba:

- Validar la comunicación en el sentido TIA Portal → Simulink, asegurando que las variables escritas desde el PLC pueden ser leídas correctamente en el modelo.

Evidencias utilizadas para la validación:

- Captura de la variable o señal generada en TIA Portal.
- Captura del Scope de Simulink con la señal recibida.
- Comparación entre la señal transmitida y la señal observada.

4.1.2 Validación del modelo dinámico y de la animación en tiempo real

Prueba P3. Validación del modelo dinámico y de la animación

Una vez comprobada la comunicación, se evalúa la coherencia entre el modelo matemático del depósito y la animación desarrollada para representar visualmente la evolución de la altura del agua.

Para ello, se plantea una comparación entre dos situaciones:

- Una simulación ejecutada durante un tiempo finito determinado, obteniendo la evolución del nivel en un Scope.
- Una simulación ejecutada en tiempo real o con duración indefinida, observando simultáneamente la animación del depósito y la gráfica asociada.

Tras dejar transcurrir el mismo intervalo de tiempo en ambos casos, se comparan las curvas obtenidas para verificar que la animación reproduce de forma fiel la evolución calculada por el modelo.

Objetivo de la prueba:

- Comprobar que la animación no introduce discrepancias respecto al modelo dinámico y que la representación gráfica en tiempo real coincide con la simulación convencional del sistema.

Evidencias utilizadas para la validación:

- Gráfica del Scope en una simulación de duración fija.
- Captura de la animación y de la gráfica generada en tiempo real.
- Comparación de ambas curvas para el mismo intervalo temporal.

4.1.3 Pruebas del controlador PID

El siguiente bloque de ensayos se centra en la validación del sistema de control. Dado que el controlador se ha estudiado inicialmente en MATLAB/Simulink y posteriormente se ha implementado en TIA Portal mediante el bloque PID_Compact, se plantea una comparación entre ambos casos.

Prueba P4. Evaluación del PID en MATLAB/Simulink

En primer lugar, se analiza el comportamiento del sistema en un entorno puramente simulado, incorporando al modelo del depósito un controlador PID configurado en Simulink. Se estudia la respuesta del sistema ante un cambio de consigna en la altura deseada.

Objetivo de la prueba:

- Evaluar la capacidad del controlador PID para regular el nivel del depósito en un entorno de simulación controlado y disponer de una referencia con la que comparar la implementación posterior en TIA Portal.

Aspectos analizados:

- Tiempo de subida.
- Sobreimpulso.
- Tiempo de establecimiento.
- Error en régimen permanente.
- Suavidad de la señal de control, si se considera relevante.

Prueba P5. Evaluación del PID implementado en TIA Portal

A continuación, se realiza la misma prueba utilizando el controlador PID_Compact programado en TIA Portal. En este caso, el modelo del depósito permanece en Simulink, mientras que el cálculo de la acción de control se lleva a cabo en el PLC virtual, intercambiando las variables mediante OPC-UA.

Objetivo de la prueba:

- Verificar que el controlador implementado en TIA Portal es capaz de regular el modelo del depósito y que el comportamiento obtenido resulta coherente con el observado previamente en MATLAB/Simulink.

Aspectos analizados:

- Evolución de la altura frente a la consigna.
- Estabilidad del sistema.
- Rapidez de respuesta.
- Posibles diferencias respecto al PID implementado directamente en Simulink.

Prueba P6. Comparación entre ambos controladores

Tras realizar las pruebas anteriores, se plantea una comparación directa entre la respuesta obtenida con el PID de MATLAB/Simulink y la respuesta del PID_Compact de TIA Portal.

Objetivo de la prueba:

- Analizar si ambos controladores proporcionan un comportamiento equivalente y estudiar las posibles diferencias derivadas de la implementación, del tiempo de muestreo, de la discretización o de la propia estructura interna de cada regulador.

Evidencias utilizadas para la validación:

- Superposición o comparación de las curvas de respuesta.
- Tabla resumen de parámetros característicos, si procede.
- Comentario razonado de las diferencias observadas.

4.1.4 Pruebas de funcionamiento del sistema completo

Finalmente, se realizan pruebas sobre la arquitectura integrada completa, formada por el modelo del depósito en Simulink, la comunicación OPC-UA, el controlador PID implementado en TIA Portal y la visualización en tiempo real.

Estas pruebas buscan comprobar el comportamiento del sistema en situaciones representativas de funcionamiento, más allá de una única respuesta a escalón.

Prueba P7. Seguimiento de distintas consignas de altura

Se ensaya la capacidad del sistema para alcanzar diferentes valores de referencia de nivel. Para ello, se modifican sucesivamente las consignas de altura y se observa la respuesta del sistema.

Objetivo de la prueba:

- Comprobar que el lazo de control es capaz de regular el depósito en distintos puntos de operación y no únicamente alrededor de una altura concreta.

Casos de prueba:

- Alcanzar una altura inicial baja.
- Incrementar la consigna a un valor intermedio o alto.
- Reducir posteriormente la consigna.

Prueba P8. Respuesta ante perturbaciones

Además del seguimiento de consignas, se analiza la robustez del controlador ante perturbaciones introducidas de forma controlada. Estas perturbaciones representan modificaciones temporales en la dinámica de descarga del depósito, al alterar el término asociado a la resistencia hidráulica equivalente del modelo.

Objetivo de la prueba:

- Evaluar la capacidad del controlador para recuperar el nivel deseado después de que el sistema se vea alterado por una perturbación externa.

Aspectos analizados:

- Magnitud de la desviación causada por la perturbación.
- Tiempo necesario para recuperar la consigna.
- Presencia o ausencia de oscilaciones excesivas.
- Comportamiento de la acción de control durante la corrección.

4.2 Análisis e interpretación de los resultados

Este apartado recoge los resultados obtenidos en las pruebas definidas anteriormente y analiza su significado en relación con los objetivos del proyecto. Para mantener la coherencia del capítulo, la presentación de resultados seguirá el mismo orden que la definición de las pruebas: comunicación OPC-UA, validación del modelo y la animación, comportamiento del controlador PID y validación global del sistema.

4.2.1 Resultados de las pruebas de comunicación OPC-UA

Comunicación Simulink → TIA Portal

Para comprobar el funcionamiento de la comunicación desde Simulink hacia TIA Portal, se generó en Simulink una señal senoidal de amplitud unitaria, conectada simultáneamente a un bloque de visualización y al bloque OPC-UA Write, encargado de escribir su valor en la variable correspondiente del servidor OPC-UA. La señal empleada durante la prueba se muestra en la Figura 38.

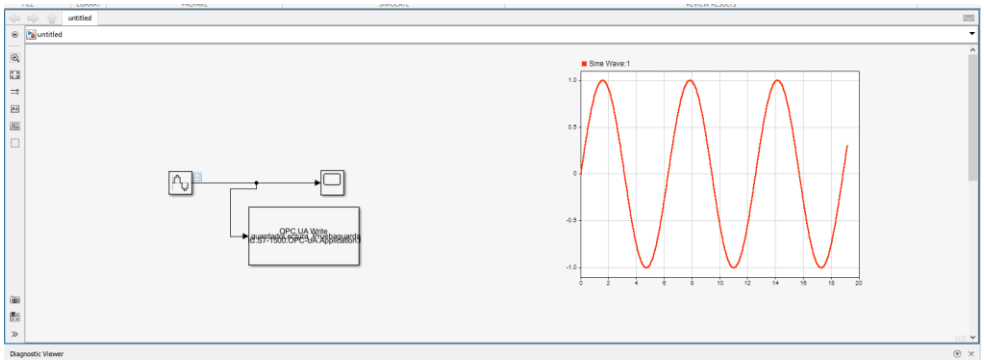


Figura 38: Señal generada en Simulink.

Como puede observarse, la señal generada presenta la evolución periódica esperada, oscilando entre valores próximos a −1 y 1. Esta misma variable fue monitorizada posteriormente desde TIA Portal mediante una traza asociada al dato recibido a través de OPC-UA, tal como se recoge en la Figura 39.

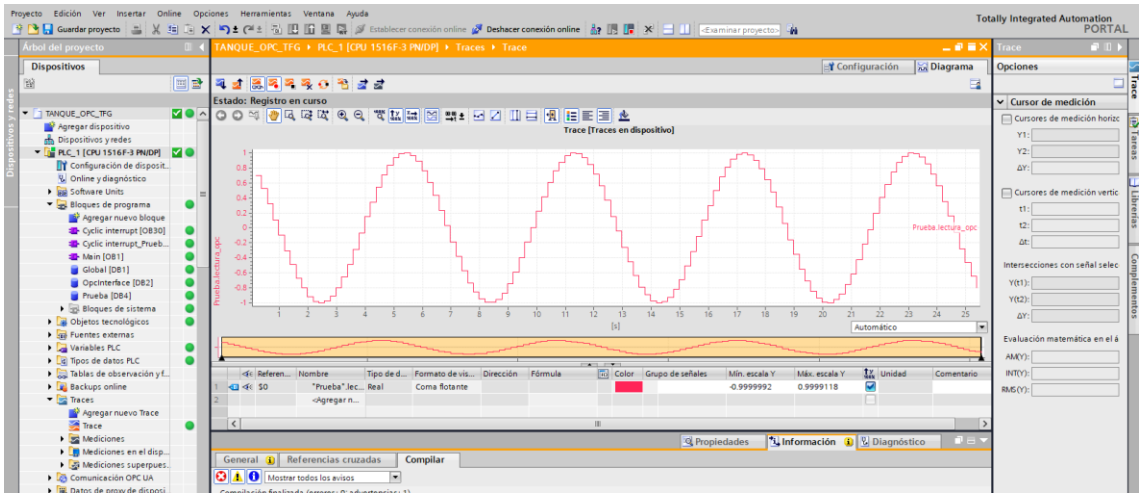


Figura 39: Señal recibida en TIA Portal.

La evolución registrada en TIA Portal reproduce correctamente el comportamiento senoidal generado en Simulink. Se mantienen tanto el rango de valores como la periodicidad de la señal, sin apreciarse deformaciones significativas ni pérdidas de información durante la transmisión. La representación obtenida en la traza presenta un aspecto ligeramente escalonado, coherente con el carácter muestreado del intercambio de datos y del propio registro temporal realizado en el entorno del PLC.

Estos resultados permiten confirmar que la escritura de variables desde Simulink hacia TIA Portal mediante OPC-UA se realiza de forma correcta, validando el sentido de comunicación necesario para enviar al controlador variables procedentes del modelo dinámico, como puede ser la altura calculada del depósito.

Comunicación TIA Portal → Simulink

La segunda prueba se realizó en sentido inverso, con el objetivo de verificar la lectura desde Simulink de una señal generada en TIA Portal. Para ello, se implementó en el PLC un pequeño programa ejecutado dentro de una interrupción cíclica con un periodo de 0,02 s, coincidente con el tiempo de muestreo configurado en Simulink. En dicho programa, una variable auxiliar x se incrementa en 0,1 en cada ciclo de ejecución hasta alcanzar un valor igual o superior a 31,416, momento en el que vuelve a reiniciarse a cero. De forma simultánea, se calcula el seno de esta variable mediante el bloque correspondiente de TIA Portal y el resultado se almacena en una segunda variable. Esta última es la señal senoidal empleada en la prueba, registrada en el entorno del PLC y mostrada en la Figura 40.

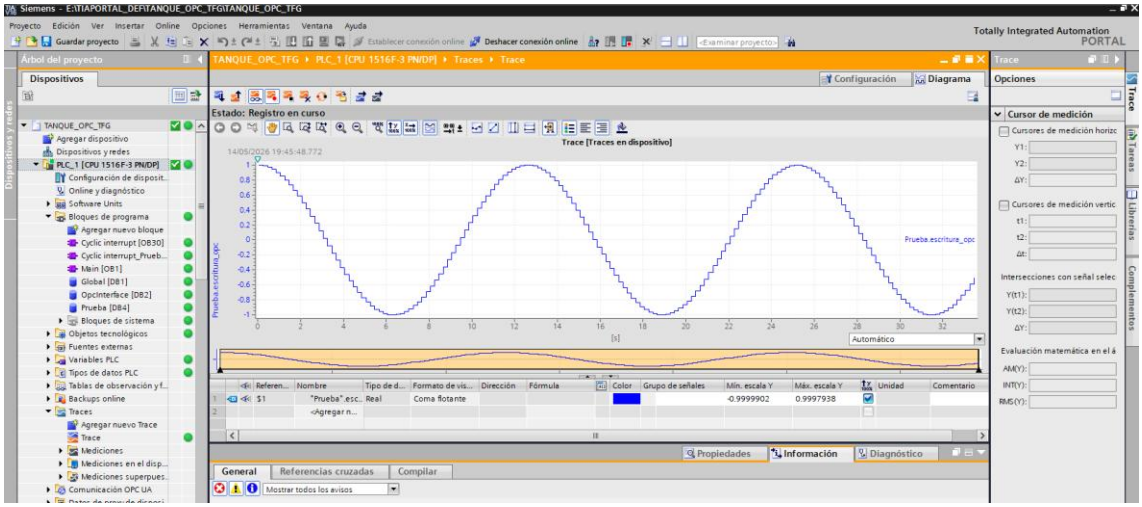


Figura 40: Señal generada en TIA Portal.

La traza obtenida en TIA Portal presenta de nuevo una oscilación periódica comprendida entre valores próximos a −1 y 1. Esta variable fue leída desde Simulink mediante el bloque OPC-UA Read y representada gráficamente durante la simulación, como se observa en la Figura 41.

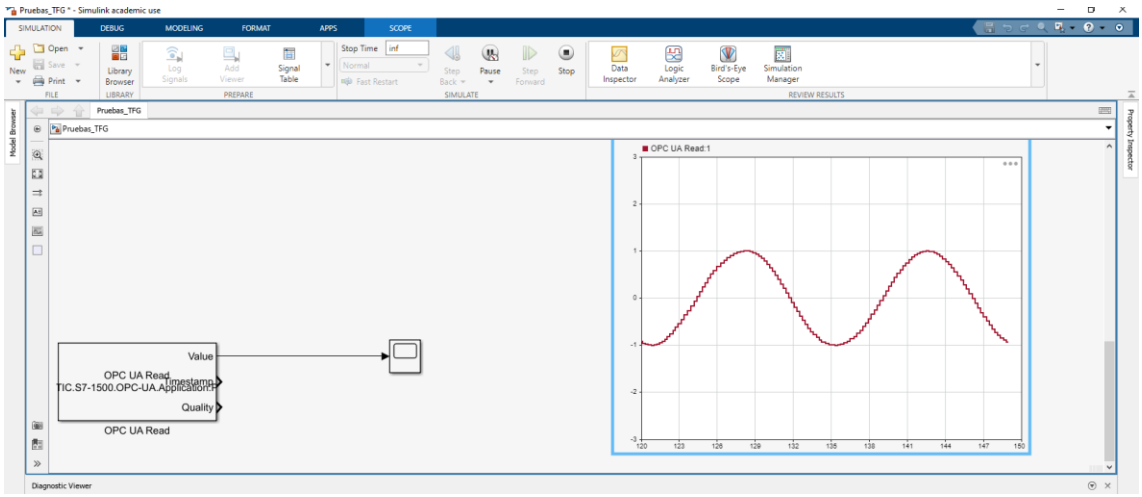


Figura 41: Señal recibida en Simulink.

La señal recibida en Simulink mantiene el mismo comportamiento que la señal registrada en TIA Portal, conservando su forma, amplitud y periodicidad. Al igual que en la prueba anterior, la gráfica muestra una evolución discretizada, asociada a la actualización periódica de los datos intercambiados mediante OPC-UA. No obstante, esta característica no impide identificar con claridad que la información transmitida se recibe correctamente y conserva su evolución temporal.

Por tanto, la prueba confirma que Simulink es capaz de leer adecuadamente variables procedentes de TIA Portal a través del servidor OPC-UA. Este sentido de comunicación resulta esencial para el funcionamiento del sistema completo, ya que permite enviar desde el controlador implementado en el PLC la acción de control que posteriormente actúa sobre el modelo virtual del depósito.

Interpretación conjunta

Los resultados obtenidos en ambas pruebas permiten validar la comunicación bidireccional entre MATLAB/Simulink y TIA Portal mediante OPC-UA. En los dos sentidos de intercambio se ha comprobado que una señal de referencia conocida puede transmitirse de un entorno al otro manteniendo su comportamiento temporal y su rango de valores.

La correcta recepción de las señales senoidales confirma que los bloques OPC-UA Write y OPC-UA Read utilizados en Simulink, así como las variables configuradas en TIA Portal, se encuentran correctamente enlazados. Aunque las representaciones gráficas presentan el aspecto discretizado propio de un sistema de adquisición y comunicación muestreado, no se observan alteraciones relevantes que comprometan el intercambio de información.

Esta validación constituye un paso previo fundamental para el resto de pruebas del proyecto. Una vez comprobado que la comunicación funciona de forma fiable en ambos sentidos, puede abordarse con mayor garantía la conexión entre el modelo dinámico del depósito y el controlador PID implementado en TIA Portal, que requiere precisamente un intercambio continuo de variables entre ambos entornos.

4.2.2 Resultados de la validación del modelo dinámico y de la animación

Para llevar a cabo esta comprobación se utilizó el modelo de Simulink mostrado en la Figura 42. En dicho esquema, la altura del depósito calculada por el bloque Water Tank se emplea simultáneamente como señal de realimentación del lazo de control, como variable representada en el Scope y como entrada del bloque Animation Control, encargado de actualizar la animación en tiempo real. De este modo, tanto la gráfica convencional como la representación visual del depósito reciben exactamente la misma señal de nivel.

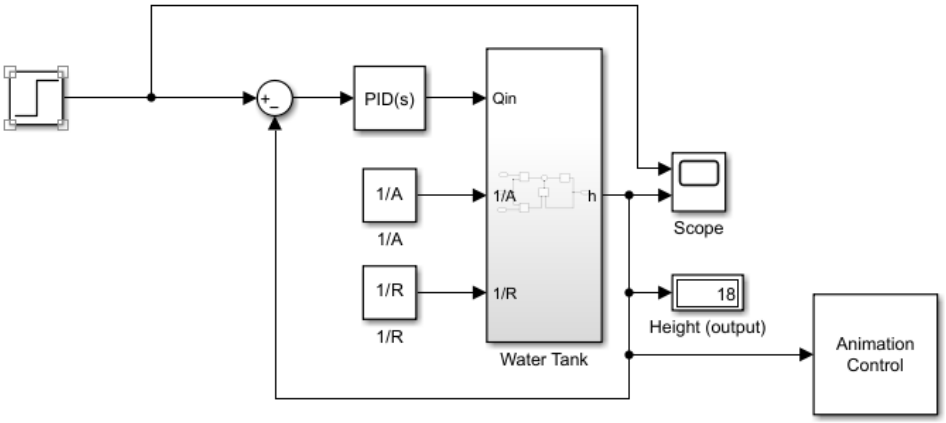


Figura 42: Modelo Simulink para prueba.

En primer lugar, se ejecutó una simulación de duración finita para obtener una referencia gráfica completa del comportamiento del sistema. En esta prueba se aplicó inicialmente una consigna de altura de 4 unidades y, posteriormente, se modificó hasta alcanzar un valor de 18 unidades. La evolución obtenida en el Scope se recoge en la Figura 43.

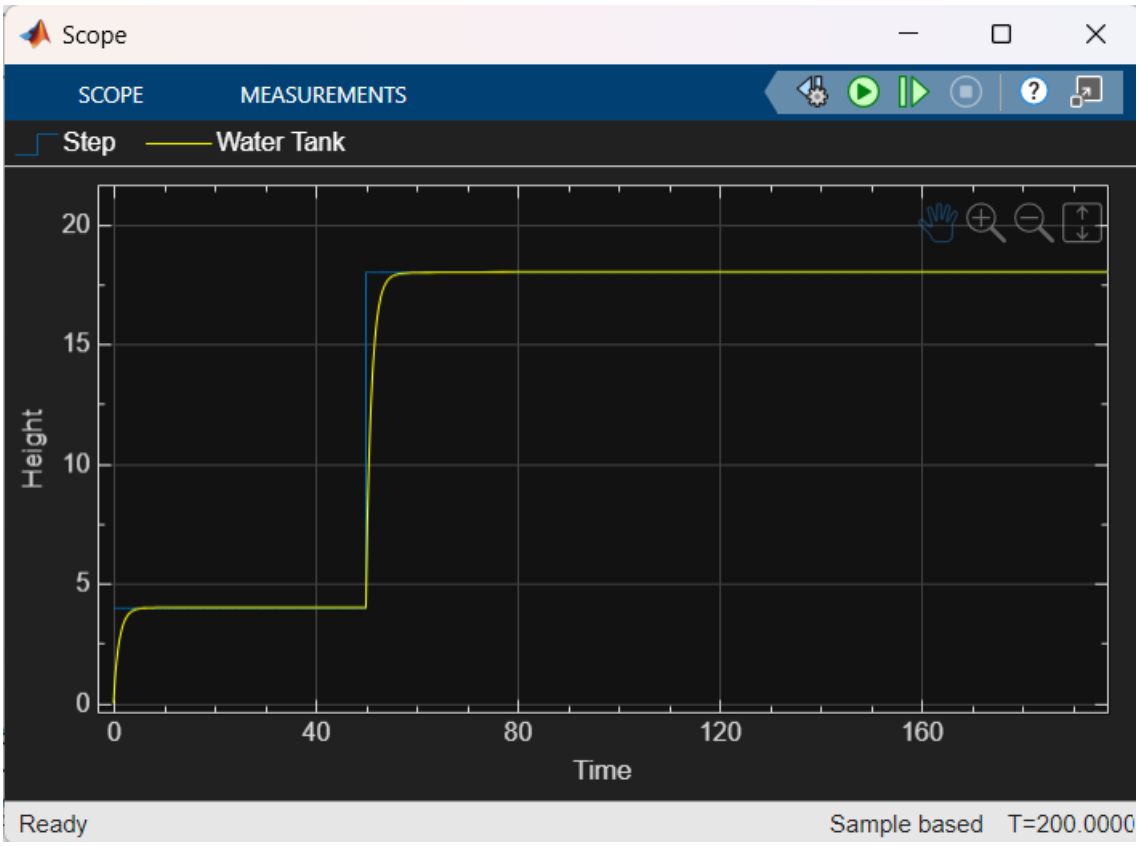


Figura 43: Captura de prueba realizada en Simulink.

Como se observa, el nivel del depósito aumenta desde su valor inicial hasta estabilizarse en torno a 4 unidades durante la primera fase de la prueba. Tras el cambio de consigna, producido aproximadamente en $t = 50\,\mathrm{s}$, la altura vuelve a incrementarse hasta alcanzar el nuevo valor de referencia de 18 unidades, manteniéndose estable a partir de ese instante.

A continuación, se repitió el mismo ensayo utilizando la simulación en tiempo real con la animación activada. En la primera fase de la prueba, correspondiente a la consigna de 4 unidades, la interfaz refleja tanto el nivel alcanzado por el depósito como la evolución temporal asociada, tal como se muestra en la Figura 44.

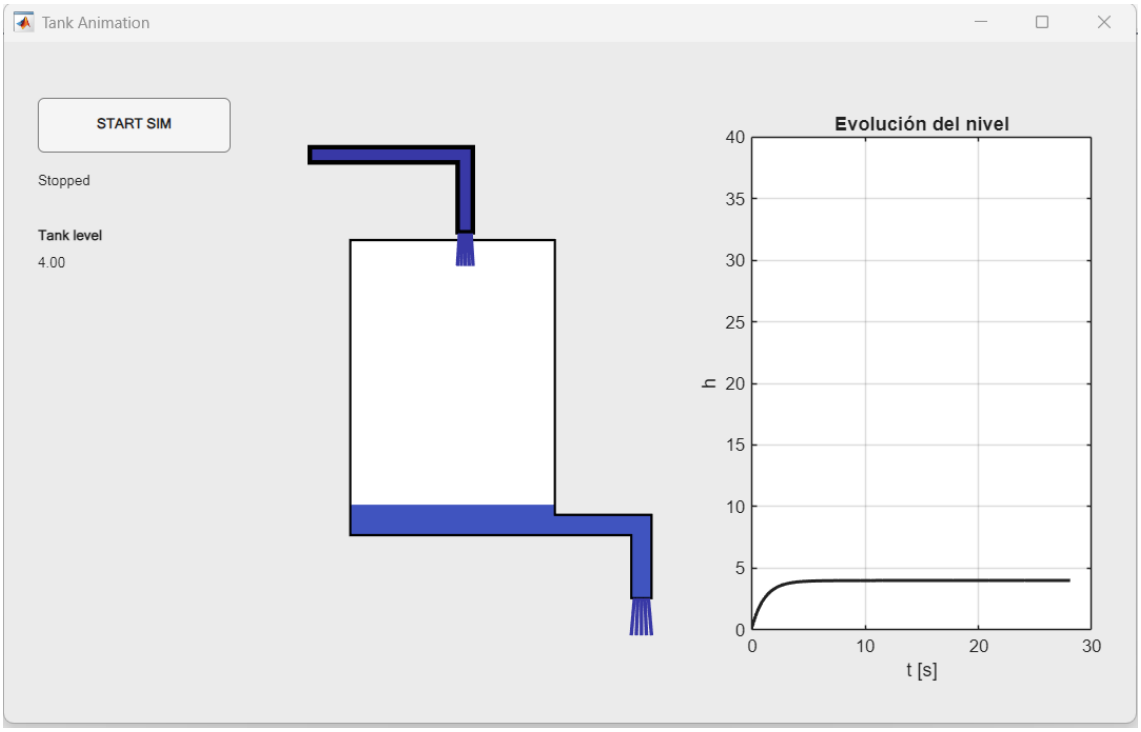


Figura 44: Primera fase de la prueba animación.

La gráfica incluida en la propia animación reproduce el mismo comportamiento observado en la simulación de referencia: el nivel parte de cero, aumenta de forma progresiva y se estabiliza en torno a 4 unidades. Además, la representación visual del líquido en el depósito coincide con el valor numérico indicado en la interfaz, confirmando que la animación refleja correctamente el estado calculado por el modelo.

La misma comprobación se realizó tras el cambio de consigna hasta 18 unidades. El resultado obtenido en la interfaz de animación se presenta en la Figura 45.

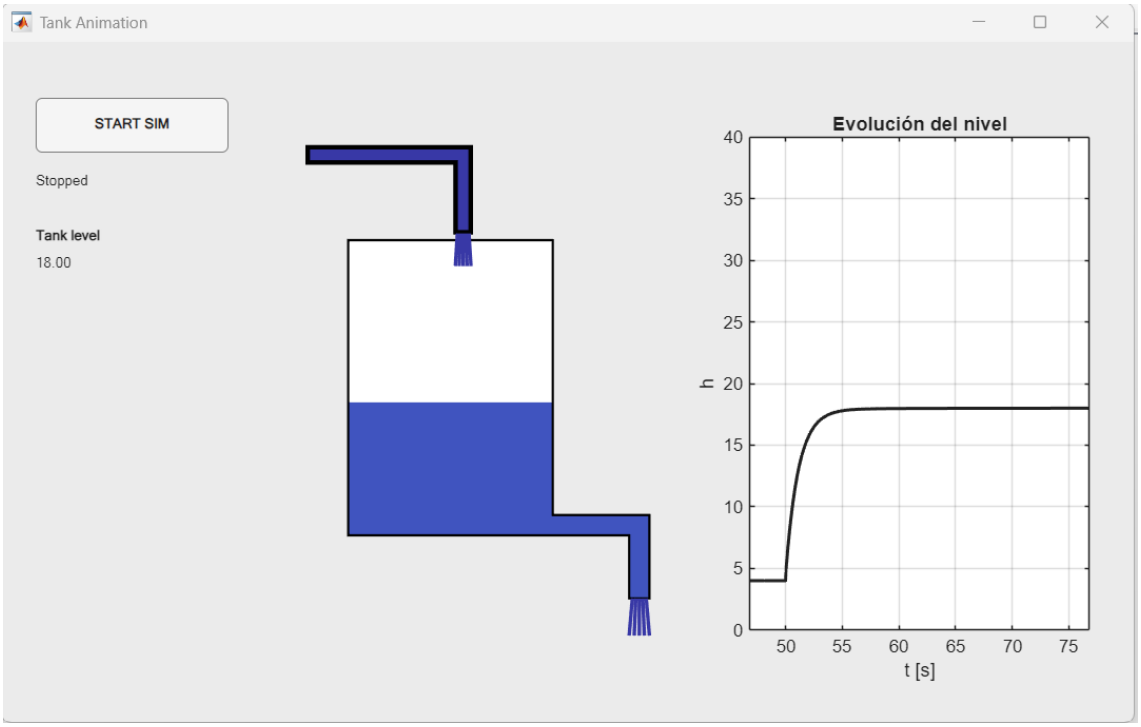


Figura 45: Segunda fase de la prueba animación.

En este segundo caso, la curva mostrada en tiempo real reproduce la transición desde el nivel anterior de 4 unidades hasta el nuevo valor de equilibrio de 18 unidades. Este comportamiento coincide con el registrado previamente en el Scope, tanto en la forma general de la respuesta como en los valores alcanzados antes y después del cambio de consigna. De igual forma, la altura representada gráficamente en el depósito se corresponde con el valor numérico indicado en la interfaz.

A partir de estos resultados, puede concluirse que la animación desarrollada no introduce alteraciones sobre la dinámica del sistema, sino que actúa únicamente como una representación visual del nivel calculado por el modelo. La concordancia entre la simulación convencional y la visualización en tiempo real valida el correcto funcionamiento de la herramienta gráfica implementada y permite utilizarla posteriormente como apoyo para interpretar de forma más intuitiva el comportamiento del sistema completo.

4.2.3 Resultados de las pruebas del controlador PID

Comportamiento del PID en MATLAB/Simulink

En primer lugar, se evaluó el comportamiento del sistema utilizando únicamente el entorno de MATLAB/Simulink. En esta configuración, tanto el modelo dinámico del depósito como el controlador PID se ejecutan dentro de Simulink, sin intervención de TIA Portal ni del intercambio de variables mediante OPC-UA. Esta prueba permite disponer de una referencia inicial del comportamiento esperado del sistema en un entorno de simulación completamente integrado.

Para ello, se aplicó una consigna escalón de 16 unidades de altura y se registró la evolución de la salida del depósito mediante un bloque Scope. La respuesta obtenida se muestra en la Figura 46.

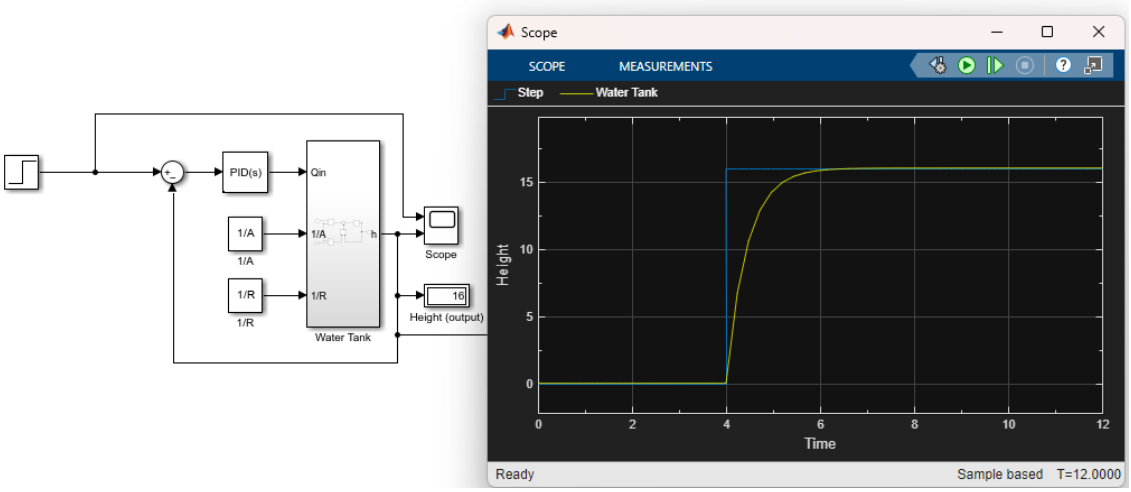


Figura 46: Respuesta ante una consigna de 16 unidades en Simulink.

Como puede observarse, tras la aplicación de la consigna, el nivel del depósito aumenta de forma rápida y progresiva hasta alcanzar el valor de referencia. La respuesta presenta un comportamiento estable, sin oscilaciones apreciables ni sobreimpulso significativo, y la variable controlada converge al entorno de las 16 unidades de altura.

Asimismo, una vez alcanzado el régimen permanente, no se aprecia error estacionario relevante entre la consigna y la salida del sistema. Por tanto, la simulación confirma que el controlador PID configurado en Simulink es capaz de regular correctamente el nivel del depósito y constituye una referencia adecuada para comparar posteriormente la implementación realizada en TIA Portal.

Comportamiento del PID implementado en TIA Portal

Una vez verificado el funcionamiento del control en Simulink, se analizó la respuesta del sistema cuando la regulación se realiza mediante el bloque PID_Compact de TIA Portal. En esta configuración, el modelo del depósito continúa ejecutándose en Simulink, mientras que el cálculo de la acción de control se lleva a cabo en el PLC virtual. El intercambio entre ambos entornos se realiza mediante OPC-UA, de modo que Simulink envía al controlador la altura actual del depósito y recibe desde TIA Portal la señal de control correspondiente.

Durante las primeras pruebas se inició la simulación en Simulink y, prácticamente de forma inmediata, se activó el sistema desde la HMI de TIA Portal introduciendo una consigna de 16 unidades. La respuesta registrada en la animación del depósito se muestra en la Figura 47.

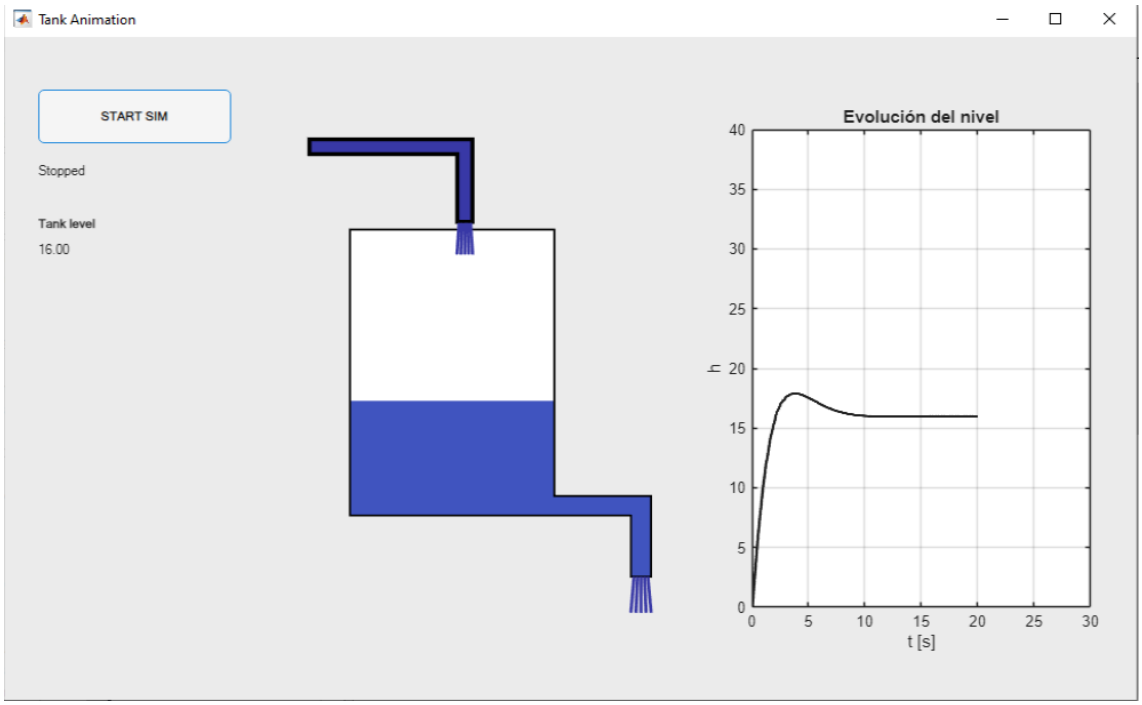


Figura 47: Respuesta del sistema con PID_Compact ante una activación inmediata desde la animación Simulink.

En esta condición, el nivel asciende rápidamente hasta superar de forma moderada la consigna, alcanzando un valor máximo próximo a las 18 unidades, para después descender y estabilizarse en torno al valor de referencia. La respuesta sigue siendo estable y converge correctamente, aunque el sobreimpulso inicial resulta más visible que en la prueba realizada únicamente en Simulink.

La misma situación puede observarse desde la ventana de monitorización de PID_Compact en TIA Portal, representada en la Figura 48.

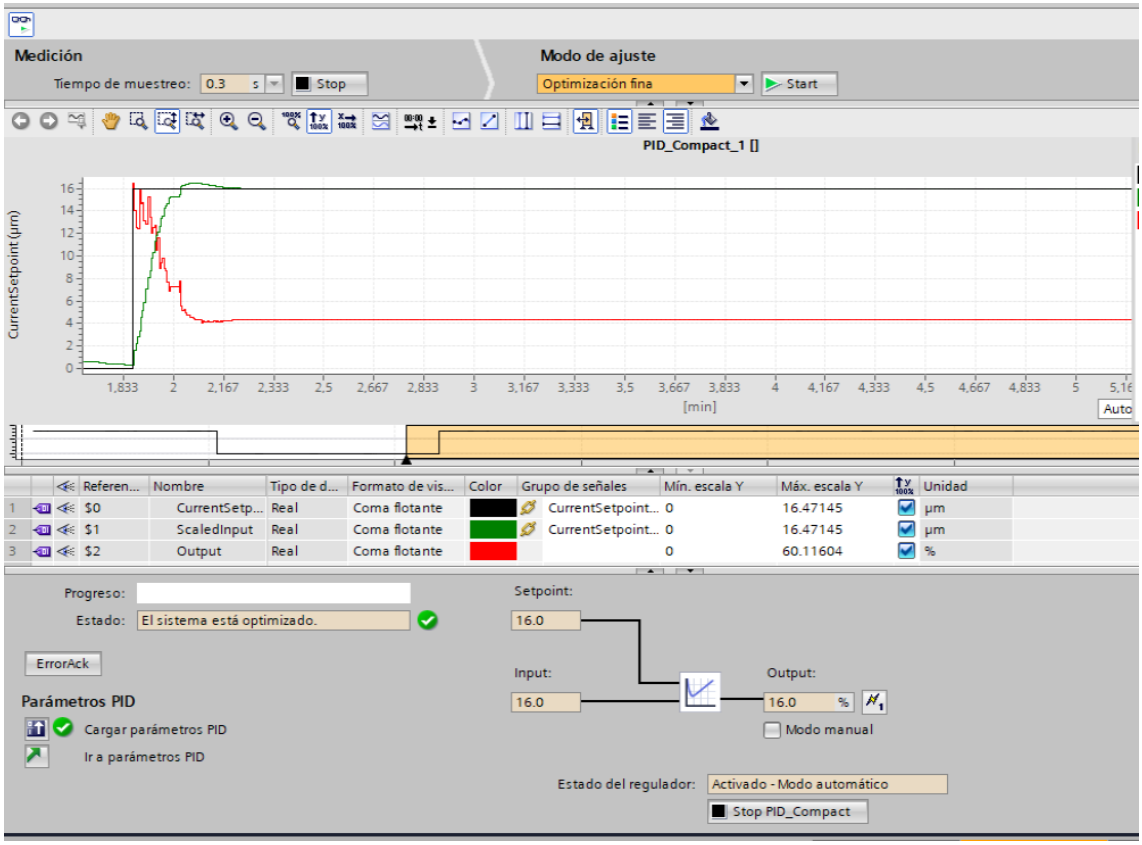


Figura 48: Respuesta del sistema con PID_Compact ante una activación inmediata desde la monitorización TIA Portal.

En esta gráfica se aprecia la evolución simultánea de la consigna, la variable de proceso escalada y la salida del regulador. Tras la activación inicial, la variable controlada aumenta con rapidez, presenta un sobreimpulso transitorio y posteriormente converge hacia la referencia. La señal de salida del controlador muestra una actuación intensa durante el arranque, necesaria para impulsar el llenado inicial del depósito, y se reduce después hasta alcanzar un valor estacionario compatible con el mantenimiento del nivel.

Influencia del instante de activación del sistema

Al repetir la prueba dejando transcurrir unos segundos desde el inicio de la simulación en Simulink antes de activar el sistema desde la HMI e introducir la consigna, se observó una respuesta transitoria más suave. Esta condición permite que el modelo, la comunicación OPC-UA y el entorno de supervisión se encuentren ya en ejecución antes de iniciar el cambio de referencia, evitando que el transitorio propio de arranque del conjunto coincida con el instante de aplicación de la consigna.

La respuesta registrada desde la animación de MATLAB/Simulink en esta segunda condición se muestra en la Figura 49.

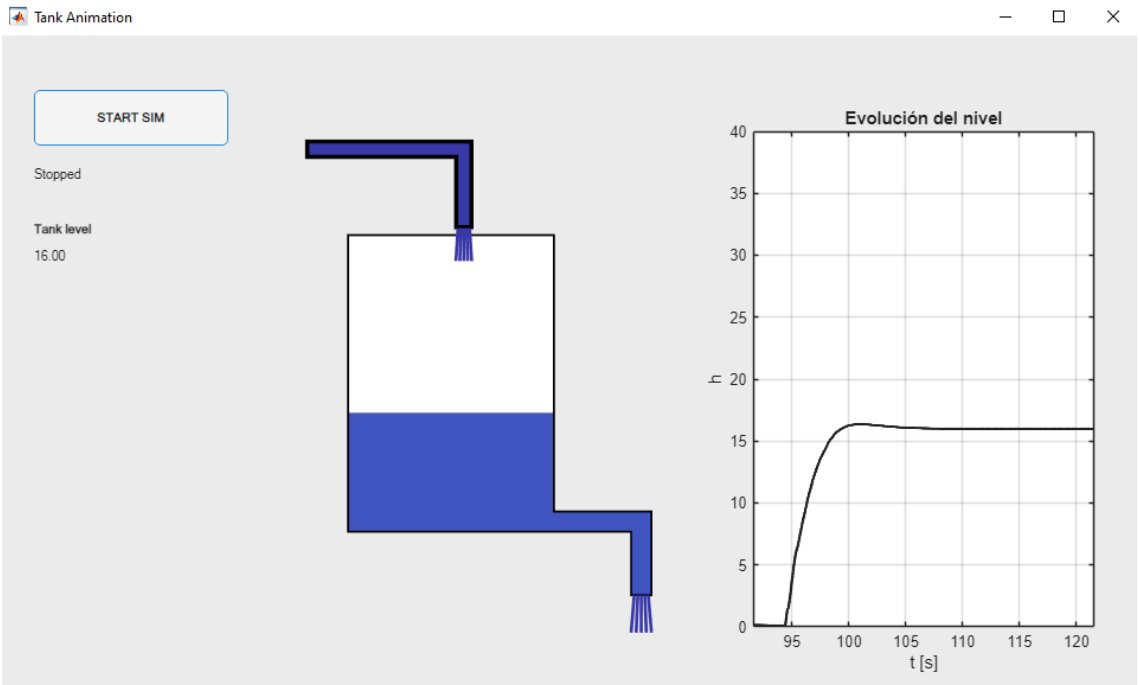


Figura 49: Respuesta con PID_Compact tras una espera previa, visualizada en Simulink.

En esta captura se observa que el nivel del depósito asciende hacia la consigna de 16 unidades con un sobreimpulso mucho más reducido que en la prueba anterior. La variable controlada supera ligeramente el valor de referencia y, a continuación, converge de forma estable hacia el régimen permanente. La evolución representada confirma que el sistema mantiene un comportamiento correcto y que la forma de la respuesta mejora cuando la orden de control se aplica una vez estabilizada la ejecución inicial del entorno distribuido. La misma prueba, supervisada desde TIA Portal, se recoge en la Figura 50.

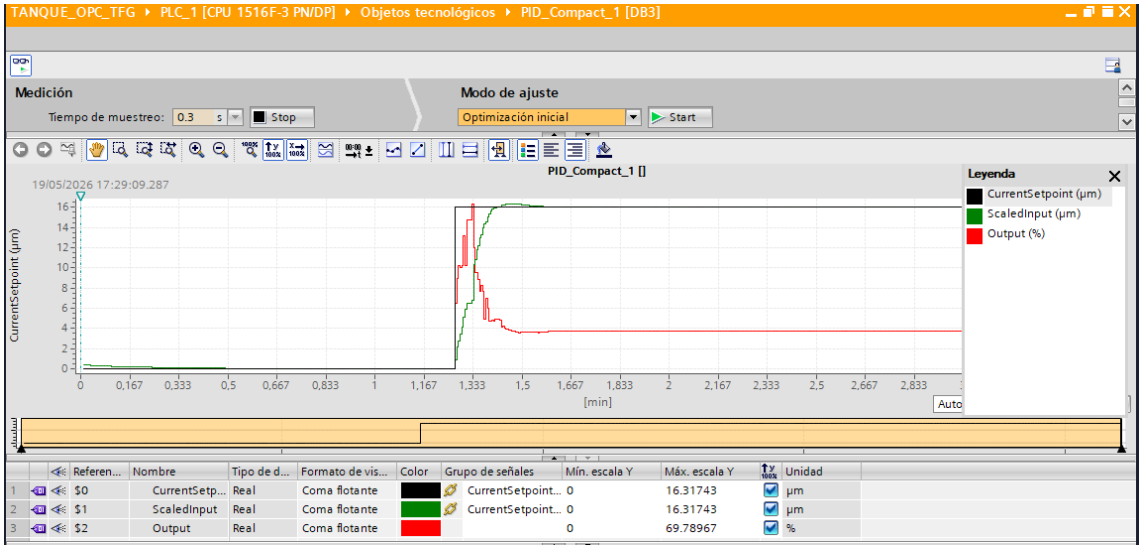


Figura 50: Respuesta del sistema con PID_Compact tras una breve espera previa a la activación, visualizada en TIA Portal.

La gráfica de PID_Compact reproduce el mismo comportamiento observado en la animación de Simulink. La variable de proceso alcanza la consigna de 16 unidades con una sobreoscilación limitada y se estabiliza posteriormente en torno a dicho valor. Asimismo, la salida del regulador presenta un pico inicial asociado al esfuerzo de control necesario para elevar el nivel, seguido de una reducción progresiva hasta alcanzar un régimen estacionario.

La coincidencia entre las capturas de Simulink y TIA Portal confirma que ambas herramientas están representando correctamente la misma evolución del lazo de control: TIA Portal muestra la variable recibida desde el modelo mediante OPC-UA, mientras que la animación de MATLAB refleja directamente la altura calculada por la planta virtual.

A partir de estas observaciones, para valorar el comportamiento propio del regulador resulta más representativo considerar la prueba en la que se deja un breve margen de tiempo entre el inicio de la simulación y la activación del sistema. La prueba con activación inmediata sigue siendo válida, pero incorpora de forma simultánea el transitorio de puesta en marcha de la arquitectura completa y el transitorio provocado por la aplicación de la consigna.

Comparación de los resultados obtenidos

La comparación entre la simulación íntegramente realizada en MATLAB/Simulink y la implementación distribuida mediante TIA Portal, PLCSIM Advanced y OPC-UA muestra que ambos enfoques son capaces de llevar el nivel del depósito hasta la consigna de 16 unidades y mantenerlo estable en régimen permanente. Sin embargo, la evolución temporal de la respuesta no es idéntica.

En el ensayo desarrollado únicamente en Simulink, el controlador y el modelo del depósito se ejecutan dentro del mismo entorno de simulación y bajo una gestión temporal común. La acción de control se calcula directamente a partir de la señal realimentada del modelo, sin intercambio de datos con sistemas externos. Como resultado, la respuesta es más suave y prácticamente no presenta sobreimpulso.

En la implementación con PID_Compact, el lazo de control queda distribuido entre dos entornos. El modelo dinámico del depósito se integra en Simulink, mientras que el cálculo de la acción de control se realiza en el PLC virtual. La planta se resuelve internamente con un paso de simulación de 0,01 s, mientras que el intercambio de variables mediante OPC-UA Read y OPC-UA Write, así como la ejecución del regulador en el bloque cíclico del PLC, se producen cada 0,02 s. Por tanto, aunque el depósito sigue evolucionando con una resolución temporal más fina, la variable medida recibida por el controlador y la acción de control devuelta al modelo se actualizan de forma discreta cada 20 ms.

Este funcionamiento se aproxima mejor a una arquitectura de control digital real, en la que la planta evoluciona entre dos instantes consecutivos de muestreo, mientras que el controlador lee las variables, calcula su salida y actualiza la actuación de manera periódica. No obstante, esta discretización del lazo, unida al intercambio de información entre Simulink y TIA Portal mediante OPC-UA, hace que la respuesta obtenida no tenga por qué reproducir exactamente la observada en la simulación completamente interna de MATLAB/Simulink.

Además, debe tenerse en cuenta la diferencia entre los reguladores empleados. El bloque PID de Simulink y el objeto tecnológico PID_Compact de TIA Portal no constituyen implementaciones idénticas, ya que presentan formulaciones internas, parámetros de configuración y tratamientos de la acción derivativa propios de cada entorno. Por este motivo, aun persiguiendo la misma finalidad de regulación, pueden aparecer diferencias en la rapidez inicial, el nivel de sobreimpulso o la forma concreta de aproximación a la consigna.

Las pruebas realizadas también muestran que el instante de activación del sistema influye en la forma del transitorio observado. Cuando la consigna se aplica prácticamente en el mismo momento en que se inicia la simulación, la respuesta presenta un sobreimpulso más acusado. Este comportamiento puede atribuirse a que, durante los primeros instantes, la arquitectura completa todavía se encuentra iniciando su funcionamiento coordinado: Simulink comienza la ejecución del modelo, los bloques OPC-UA inician el intercambio periódico de variables y el PLC recibe las primeras muestras de la planta virtual mientras el regulador pasa a actuar sobre una consigna no nula. En esas condiciones, es posible que los primeros ciclos del lazo se produzcan con una sincronización menos asentada entre la lectura de la variable de proceso y la aplicación de la acción de control, lo que provoca una actuación inicial más intensa del regulador y, en consecuencia, una mayor sobreoscilación.

En cambio, cuando se deja transcurrir un breve intervalo antes de activar el sistema desde la HMI, la simulación, la comunicación OPC-UA y la ejecución cíclica del PLC ya se encuentran estabilizadas. Al introducir entonces la consigna, el lazo de control parte de unas condiciones temporales más regulares y la respuesta obtenida resulta más representativa del comportamiento propio del regulador, mostrando un sobreimpulso menor y una aproximación más progresiva al valor de referencia.

Por este motivo, la prueba con una breve espera previa se considera más adecuada para analizar el comportamiento del controlador en régimen de funcionamiento normal. La prueba con activación inmediata sigue siendo útil, ya que permite observar el comportamiento del sistema durante la puesta en marcha completa de la arquitectura distribuida, pero incorpora simultáneamente el transitorio de inicialización de la comunicación y el transitorio provocado por la aplicación de la consigna.

En conjunto, los resultados confirman que la arquitectura basada en Simulink–OPC-UA–TIA Portal permite controlar de forma efectiva el modelo virtual del depósito. El sistema alcanza la consigna establecida, mantiene estabilidad en régimen permanente y presenta una evolución coherente tanto en la animación de MATLAB como en la monitorización de PID_Compact. Aunque el comportamiento transitorio difiere ligeramente del obtenido en la simulación completamente interna de Simulink, dicha diferencia es coherente con la naturaleza distribuida, discreta y más próxima a un entorno industrial de la implementación desarrollada.

4.2.4 Resultados del funcionamiento global del sistema

Seguimiento de distintas consignas de altura

Una vez validado el comportamiento del controlador, se realizaron varios ensayos sobre el sistema completo con el objetivo de comprobar su capacidad para trabajar en distintos puntos de operación. Para ello, se aplicaron cambios de consigna tanto ascendentes como descendentes, cubriendo valores bajos, intermedios y próximos al límite superior definido para el depósito.

En el modelo desarrollado se considera una altura máxima útil del tanque de 30 cm. No obstante, la consigna introducida desde la interfaz se limitó a un valor máximo de 29 cm, con el fin de evitar que el regulador demande directamente el valor extremo de capacidad y conservar un pequeño margen respecto al límite superior del depósito.

Conviene señalar que, en las capturas obtenidas desde TIA Portal, la variable de altura aparece representada con la unidad µm. Esta indicación corresponde a la unidad de longitud mostrada por defecto en la configuración del bloque PID_Compact y no afecta al análisis de los resultados. En el contexto de este trabajo, los valores numéricos representados deben interpretarse en centímetros, de acuerdo con la escala utilizada en el modelo del depósito.

El primer ensayo consistió en modificar la consigna desde un nivel inicial de 0 cm hasta un valor de 20 cm. La evolución de la consigna, de la variable controlada y de la salida del regulador registrada en TIA Portal se muestra en la Figura 51.



Figura 51: Respuesta del sistema ante un incremento de consigna de 0 cm a 20 cm, visualizada en TIA Portal.

La misma prueba, visualizada desde la animación desarrollada en Simulink, se recoge en la Figura 52.

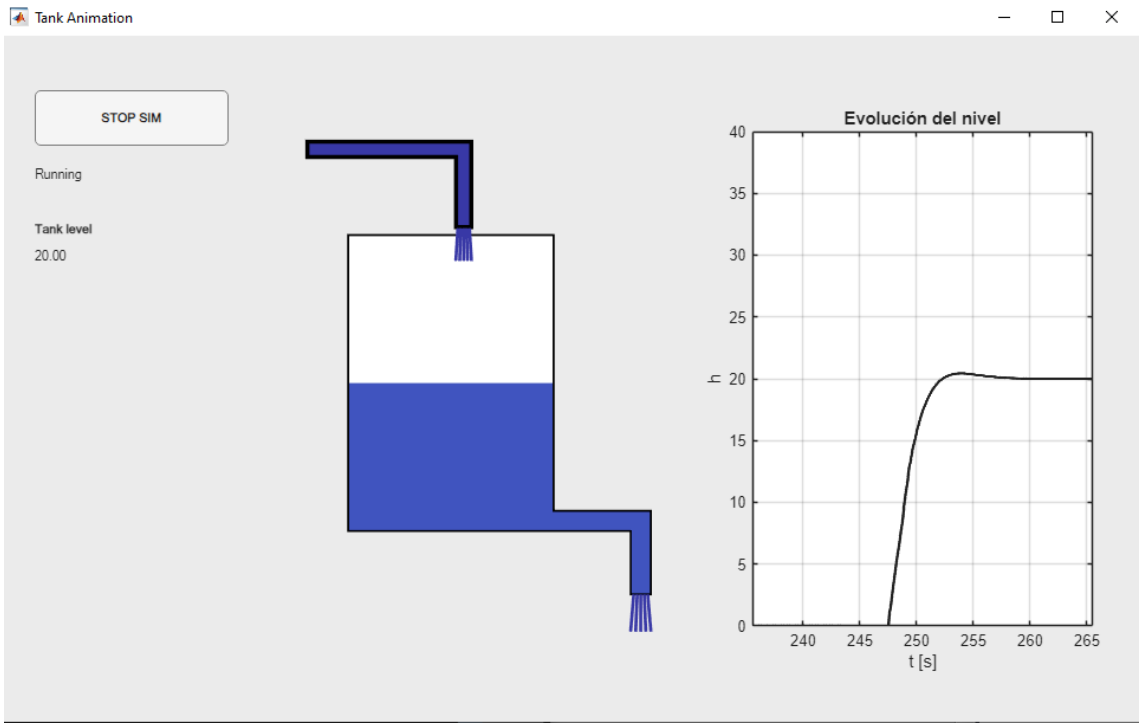


Figura 52: Respuesta del sistema ante un incremento de consigna de 0 cm a 20 cm, visualizada en la animación de Simulink.

En este caso, el sistema parte de un depósito prácticamente vacío y eleva el nivel hasta alcanzar la referencia establecida. La salida del controlador aumenta inicialmente para impulsar el llenado y se reduce progresivamente a medida que la altura se aproxima a la consigna. La respuesta presenta un ligero sobreimpulso antes de estabilizarse en torno a los 20 cm, manteniendo posteriormente el nivel deseado. Tras este primer incremento, se evaluó la respuesta del sistema ante una reducción de consigna desde 20 cm hasta 10 cm. La evolución registrada en TIA Portal se presenta en la Figura 53.

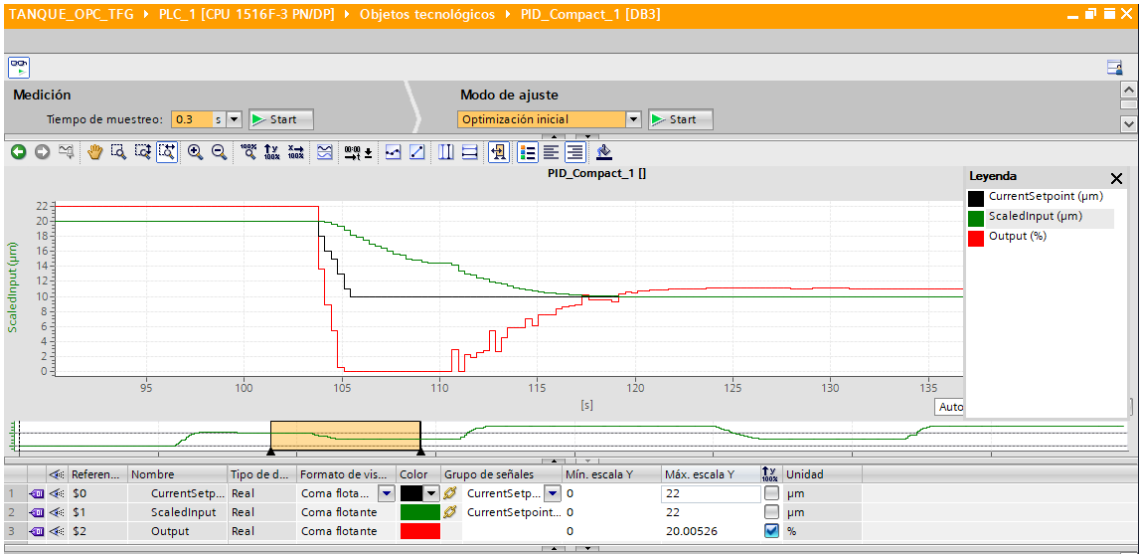


Figura 53: Respuesta del sistema ante una reducción de consigna de 20 cm a 10 cm, visualizada en TIA Portal.

La transición equivalente observada en la animación del depósito se muestra en la Figura 54.

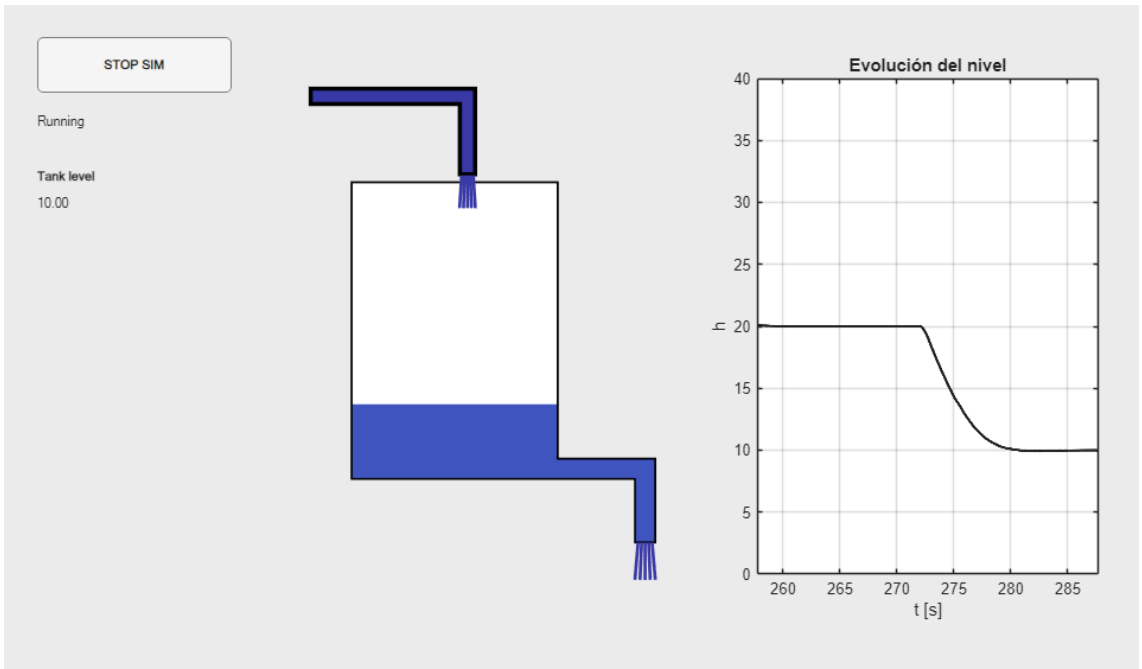


Figura 54: Respuesta del sistema ante una reducción de consigna de 20 cm a 10 cm, visualizada en la animación de Simulink.

En este ensayo, la salida del controlador disminuye de forma marcada tras el cambio de referencia, reduciendo el caudal de entrada y permitiendo que el nivel del depósito descienda por efecto de la descarga. La altura evoluciona de forma progresiva hasta

situarse en torno a los 10 cm, sin presentar oscilaciones acusadas ni una pérdida de estabilidad durante el transitorio. Este comportamiento confirma que el sistema no solo es capaz de aumentar el nivel, sino también de conducirlo de forma controlada hacia valores inferiores.

A continuación, se estudió la respuesta ante un incremento de consigna desde 10 cm hasta 29 cm, valor próximo al límite máximo de funcionamiento establecido. El resultado registrado en TIA Portal se muestra en la Figura 55.

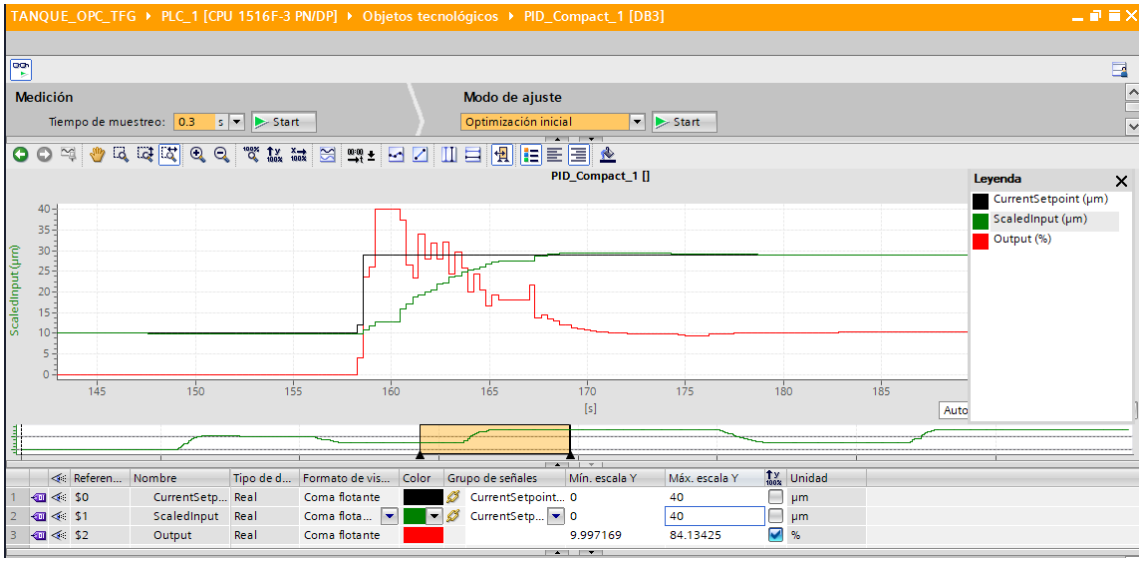


Figura 55: Respuesta del sistema ante un incremento de consigna de 10 cm a 29 cm, visualizada en TIA Portal.

La representación correspondiente obtenida en la animación de Simulink se recoge en la Figura 56.

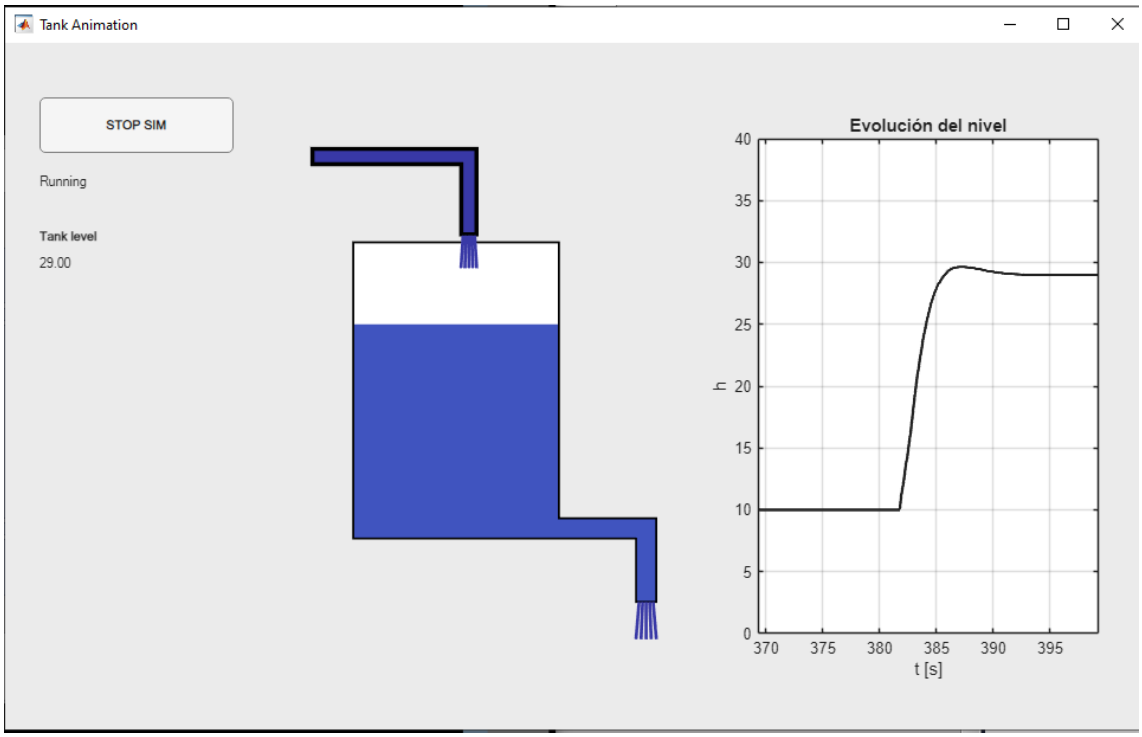


Figura 56: Respuesta del sistema ante un incremento de consigna de 10 cm a 29 cm, visualizada en la animación de Simulink.

La evolución observada confirma que el regulador mantiene un comportamiento adecuado también en la zona alta de operación del depósito. Tras el cambio de referencia, el nivel aumenta con rapidez desde los 10 cm hasta aproximarse a los 29 cm. Se aprecia un pequeño sobreimpulso transitorio, tras el cual la altura converge hacia la nueva consigna y permanece estable. Este ensayo resulta especialmente relevante al realizarse cerca del límite superior fijado para el sistema, mostrando que la regulación continúa siendo eficaz en un punto de operación exigente.

Por último, se ensayó una transición desde el valor máximo permitido de consigna, 29 cm, hasta 0 cm, con el fin de observar el comportamiento del sistema durante un vaciado completo. La evolución registrada desde TIA Portal se presenta en la Figura 57.

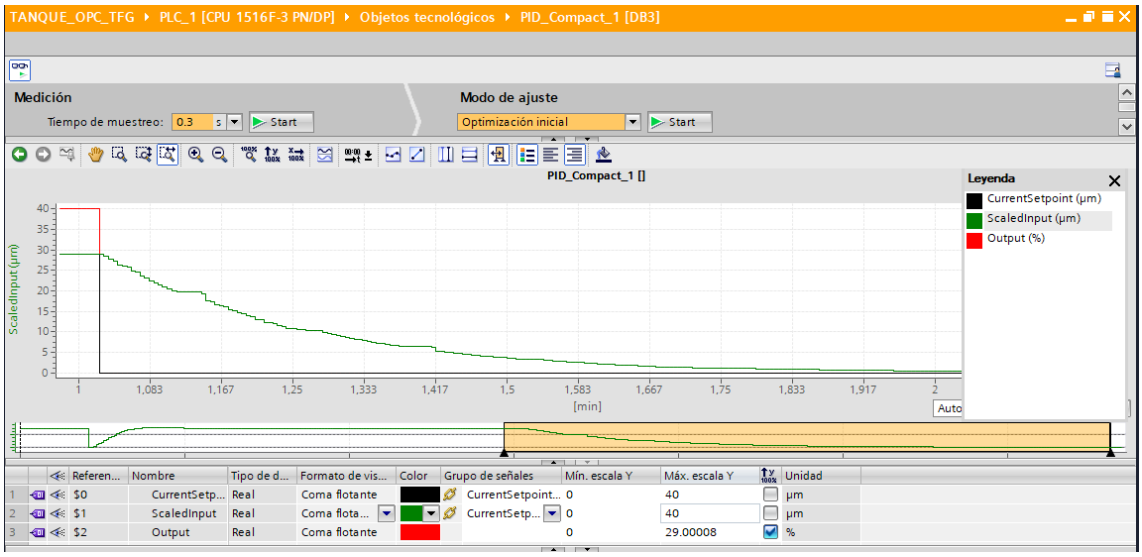


Figura 57: Respuesta del sistema durante el vaciado ante una reducción de consigna de 29 cm a 0 cm, visualizada en TIA Portal.

La representación equivalente obtenida en la animación de Simulink se muestra en la Figura 58.

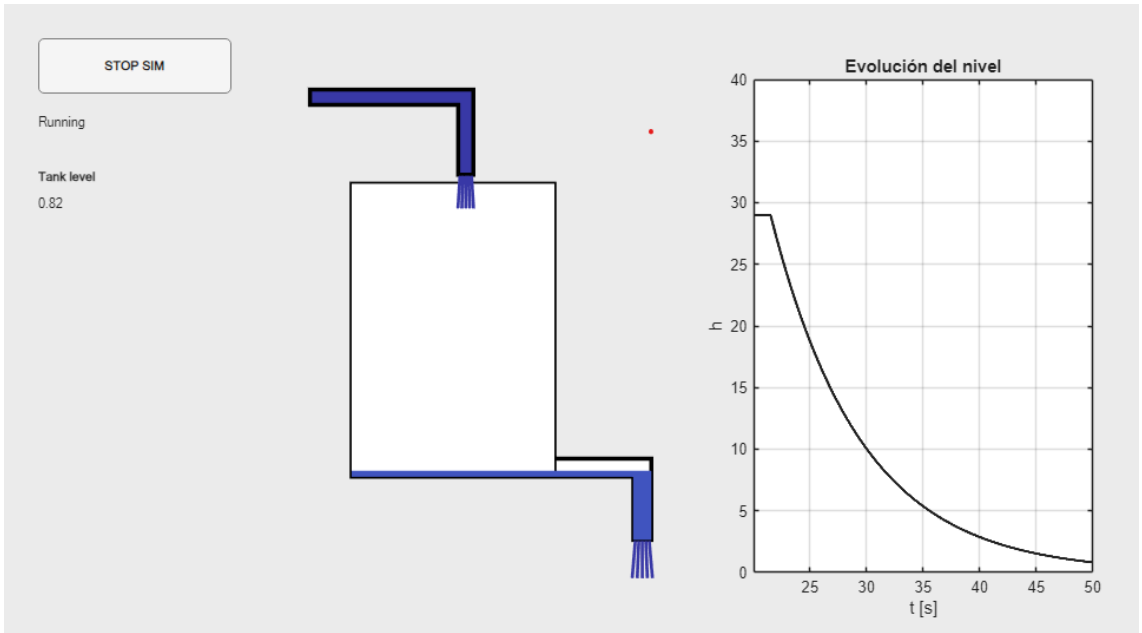


Figura 58: Respuesta del sistema durante el vaciado ante una reducción de consigna de 29 cm a 0 cm, visualizada en la animación de Simulink.

A diferencia de los casos de llenado, el descenso hasta 0 cm se produce de forma notablemente más lenta. Este comportamiento es coherente con la dinámica física del modelo: el controlador puede reducir la acción de entrada hasta anularla, pero el vaciado depende del caudal de salida del depósito y no de una actuación activa que fuerce una disminución inmediata del nivel. Como consecuencia, la altura presenta una caída gradual hasta aproximarse a cero, siguiendo la evolución gradual propia de la dinámica de descarga del tanque.

En conjunto, las pruebas de seguimiento de consignas muestran que el sistema integrado es capaz de regular el nivel del depósito en un amplio rango de funcionamiento. El controlador responde adecuadamente ante incrementos y reducciones de referencia, alcanza los valores deseados y mantiene la estabilidad una vez finalizados los transitorios. Además, las capturas obtenidas desde TIA Portal y desde la animación de Simulink presentan comportamientos coherentes entre sí, confirmando que la monitorización del controlador y la visualización del modelo representan la misma evolución dinámica.

Respuesta ante perturbaciones

Además de los cambios de consigna, se evaluó la capacidad del sistema para rechazar perturbaciones introducidas sobre la dinámica de descarga del depósito. Como se describió previamente, estas perturbaciones modifican temporalmente el término asociado a la salida del tanque, alterando el equilibrio del sistema aun cuando la consigna permanece constante. Para estas pruebas se mantuvo una referencia de 16 cm y se aplicaron dos perturbaciones de signo y magnitud diferentes. En el primer ensayo se introdujo una perturbación de valor −0,9. La evolución registrada desde TIA Portal se muestra en la Figura 59.

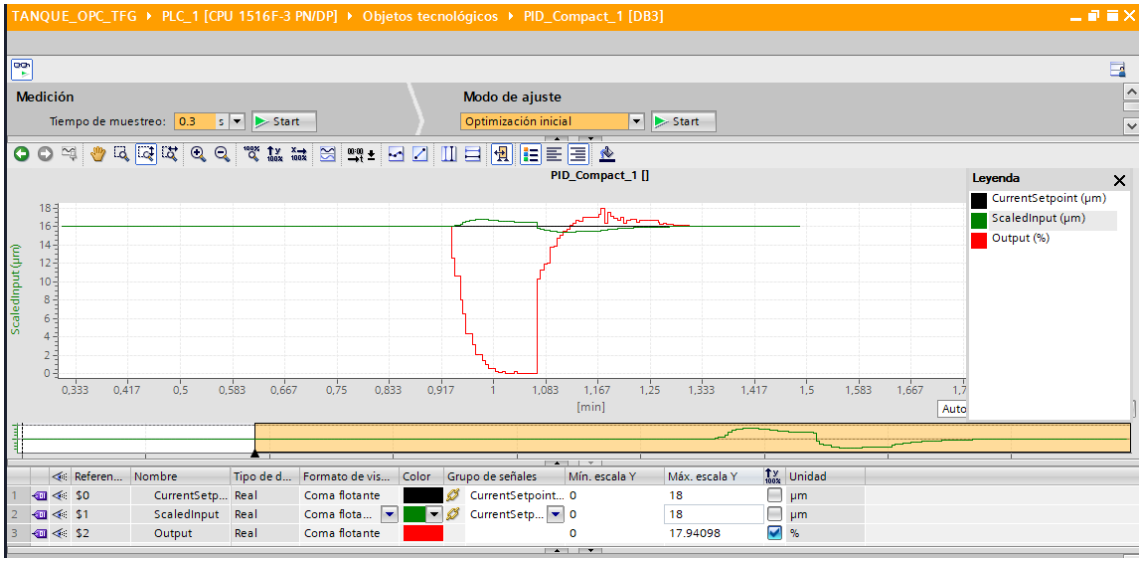


Figura 59: Respuesta del sistema ante una perturbación de −0,9, visualizada en TIA Portal.

La misma prueba observada desde la animación de Simulink se recoge en la Figura 60.

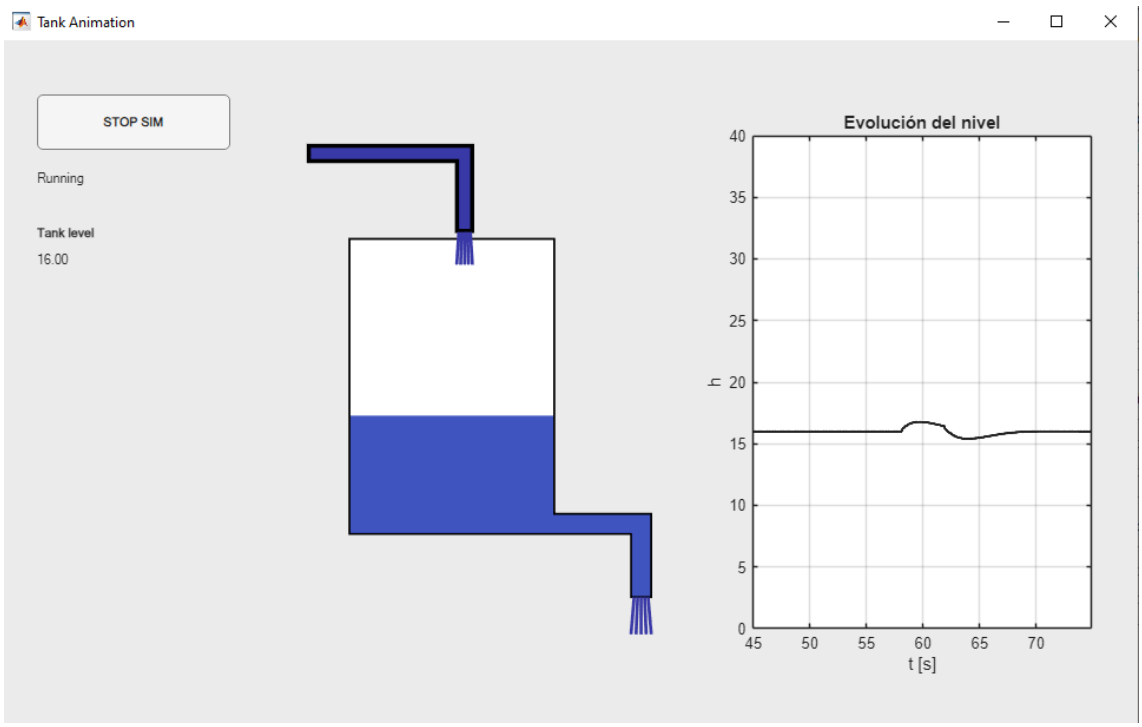


Figura 60: Respuesta del sistema ante una perturbación de −0,9, visualizada en la animación de Simulink.

Tras la aplicación de la perturbación, el nivel se desvía temporalmente respecto a la consigna de 16 cm, elevándose por encima del valor de referencia. Esta variación es compensada por el controlador mediante la modificación de su salida, que reduce la actuación sobre el sistema para corregir el incremento de altura. Como consecuencia, el nivel retorna progresivamente hacia la consigna y vuelve a estabilizarse en torno al valor deseado.

En el segundo ensayo se aplicó una perturbación de valor 1,9, de mayor magnitud que la anterior. La respuesta registrada en TIA Portal se presenta en la Figura 61.

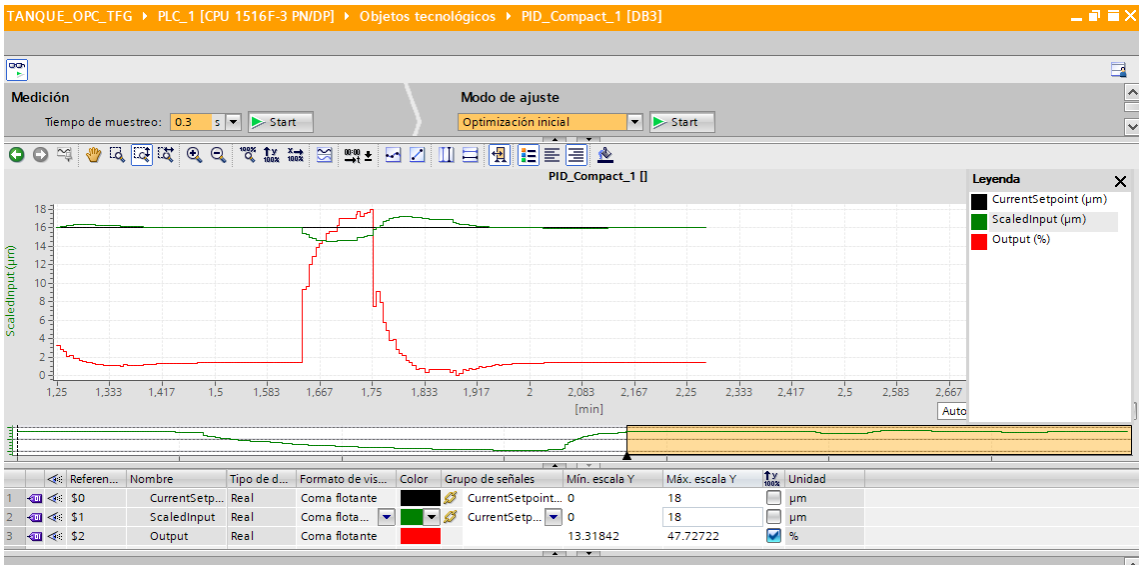


Figura 61: Respuesta del sistema ante una perturbación de 1,9, visualizada en TIA Portal.

La evolución equivalente visualizada en la animación de Simulink se muestra en la Figura 62.

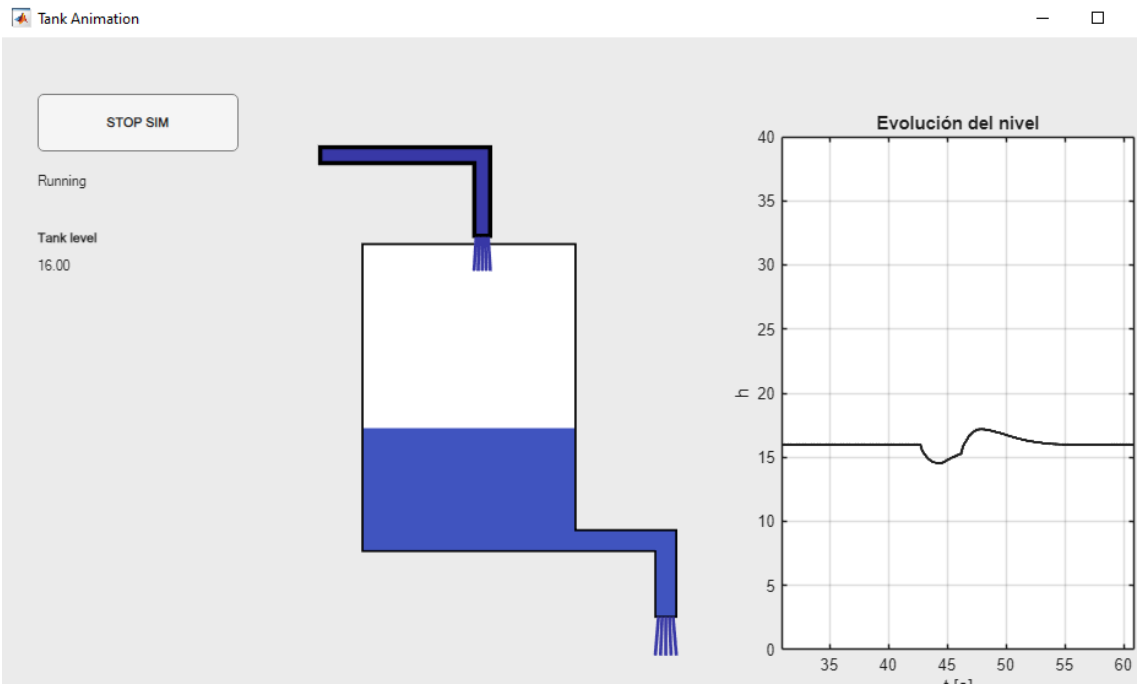


Figura 62: Respuesta del sistema ante una perturbación de 1,9, visualizada en la animación de Simulink.

En esta situación, la desviación inicial respecto a la consigna resulta más acusada y se produce en sentido descendente. El nivel del depósito disminuye de forma apreciable tras la perturbación y, como respuesta, el controlador incrementa su actuación para recuperar la altura deseada. La corrección provoca posteriormente una pequeña superación de la referencia antes de que el sistema vuelva a estabilizarse en torno a los 16 cm.

En conjunto, las pruebas realizadas muestran que el sistema es capaz de corregir las desviaciones provocadas por ambas perturbaciones y recuperar la consigna de 16 cm. La magnitud de la respuesta depende del valor introducido: la perturbación de −0,9 genera una variación más moderada, mientras que la de 1,9 provoca una desviación mayor y requiere una actuación correctora más intensa. En ambos casos, el regulador modifica su salida en el sentido adecuado para compensar el efecto de la perturbación y restablecer el nivel deseado. Aunque aparecen pequeñas oscilaciones durante el transitorio, estas se amortiguan progresivamente y no derivan en un comportamiento inestable. Por tanto, los resultados confirman que el controlador mantiene una capacidad de rechazo de perturbaciones satisfactoria dentro de las condiciones ensayadas.

4.3 Posibles mejoras

A partir de las pruebas realizadas, el sistema desarrollado puede considerarse funcional para el objetivo planteado: permite intercambiar información entre MATLAB/Simulink y TIA Portal mediante OPC-UA, controlar el modelo virtual del depósito desde un regulador industrial, visualizar su evolución en tiempo real y comprobar su respuesta ante diferentes consignas y perturbaciones. No obstante, los ensayos también permiten identificar varias líneas de mejora que podrían abordarse en futuros desarrollos.

Una primera mejora estaría relacionada con la puesta en marcha del sistema integrado. Durante las pruebas del controlador se observó que la respuesta transitoria varía dependiendo del instante en el que se activa el control desde la HMI respecto al inicio de la simulación en Simulink. Cuando la consigna se aplica de forma inmediata, el sobreimpulso resulta más acusado que cuando se deja transcurrir un breve intervalo previo. Como mejora, podría implementarse una lógica de habilitación coordinada que impida activar el regulador hasta que el modelo de Simulink, la comunicación OPC-UA y el PLC virtual se encuentren correctamente inicializados y transfiriendo datos de manera estable. De este modo, se evitaría que el transitorio de arranque de la arquitectura distribuida influya sobre la respuesta inicial del lazo de control.

También sería conveniente profundizar en el análisis temporal de la comunicación y del control. En el sistema desarrollado, la planta virtual se integra en Simulink con un paso de simulación inferior al periodo de intercambio OPC-UA y al ciclo de ejecución del regulador en el PLC. Esta configuración resulta coherente con la naturaleza continua del proceso físico, pero introduce una diferencia respecto a la simulación realizada íntegramente en Simulink, donde todo el lazo se gestiona dentro de un mismo entorno. Como trabajo futuro, podría estudiarse de forma específica la influencia de distintos tiempos de muestreo, periodos de comunicación y configuraciones de simulación sobre variables como el sobreimpulso, el tiempo de establecimiento o la capacidad de rechazo de perturbaciones. Este análisis permitiría determinar con mayor precisión el compromiso entre fidelidad temporal, estabilidad del control y carga computacional.

Otra posible mejora consiste en realizar una evaluación más cuantitativa del rendimiento del controlador. En este trabajo se ha comprobado de forma gráfica que el sistema alcanza las consignas establecidas, mantiene la estabilidad y recupera el nivel deseado tras las perturbaciones aplicadas. Sin embargo, sería posible complementar este análisis mediante indicadores numéricos, como el tiempo de subida, el tiempo de establecimiento, el sobreimpulso máximo, el error en régimen permanente o el tiempo de recuperación tras una perturbación. La obtención sistemática de estos parámetros facilitaría una comparación más precisa entre el PID implementado en MATLAB/Simulink y el bloque PID_Compact de TIA Portal, así como entre diferentes configuraciones del regulador.

En relación con el control, también podría estudiarse una sintonización más específica del PID dentro de la arquitectura integrada. Las pruebas muestran que el controlador implementado en TIA Portal regula adecuadamente el nivel, aunque su respuesta presenta diferencias respecto a la simulación exclusivamente realizada en Simulink. Además de la distinta estructura interna de ambos reguladores, estas diferencias pueden estar condicionadas por la discretización del lazo, la comunicación OPC-UA y la ejecución cíclica del PLC. Por ello, una posible continuación sería comparar distintos métodos de ajuste del regulador directamente sobre el sistema Simulink–OPC-UA–TIA Portal, valorando cuál ofrece el mejor equilibrio entre rapidez de respuesta, sobreimpulso y robustez ante perturbaciones. Asimismo, podrían analizarse con mayor detalle las limitaciones de la señal de control y la incorporación de estrategias de protección frente a saturaciones o comportamientos no deseados en condiciones extremas.

El modelo de planta también admite futuras ampliaciones orientadas a aumentar su realismo físico. Aunque el depósito implementado permite reproducir de forma adecuada la dinámica principal del nivel, podrían añadirse efectos propios de una instalación real, como ruido en la medida, incertidumbre en los parámetros, retardos adicionales, dinámicas de actuadores o restricciones más detalladas en el caudal de entrada y salida. Del mismo modo, las perturbaciones podrían generarse mediante perfiles más variados o aleatorios, en lugar de limitarse a valores concretos, con el fin de evaluar la robustez del controlador ante condiciones de operación más diversas.

Otra posible línea de mejora sería incorporar indicadores específicos de sostenibilidad asociados al uso de la arquitectura virtual. En futuros desarrollos, podría estimarse el ahorro de tiempo durante la puesta en marcha, la reducción de ensayos físicos necesarios, el menor uso de recursos materiales y energéticos durante la fase de validación, o la disminución de riesgos operativos al probar previamente el controlador en simulación. Este análisis permitiría complementar la validación técnica del sistema con una valoración más amplia de su impacto económico, ambiental y operativo.

Por último, una mejora especialmente relevante sería trasladar la arquitectura desde el entorno completamente virtual a una validación con hardware real. La sustitución del PLC simulado mediante PLCSIM Advanced por un controlador físico permitiría comprobar el comportamiento de la comunicación OPC-UA y del regulador en condiciones más próximas a una aplicación industrial. En una fase posterior, incluso podría plantearse la conexión con una planta real o con un banco de pruebas físico, manteniendo la estructura de supervisión y análisis desarrollada en este trabajo. Esta evolución permitiría convertir la solución actual en una plataforma de validación más completa para estrategias de control industrial.

Capítulo 5 Conclusiones

En este capítulo se recogen las principales conclusiones obtenidas tras el desarrollo, integración y validación del sistema propuesto. En primer lugar, se presenta una síntesis del trabajo realizado, destacando los elementos más relevantes de la arquitectura implementada. A continuación, se valoran los resultados alcanzados en relación con los objetivos planteados al inicio del proyecto. Finalmente, se proponen posibles líneas futuras de trabajo que permitirían ampliar, mejorar o trasladar la solución desarrollada hacia un entorno más próximo a una aplicación industrial real.

5.1 Resumen del trabajo

En este Trabajo Fin de Grado se ha desarrollado un sistema de control distribuido para la regulación del nivel de agua de un depósito virtual. La arquitectura propuesta combina MATLAB/Simulink, TIA Portal, SIMATIC PLCSIM Advanced y comunicación OPC-UA, de forma que el modelo de la planta se ejecuta en Simulink y el regulador se implementa en un PLC Siemens simulado.

El trabajo parte del estudio del contexto tecnológico en el que se enmarca el proyecto, incluyendo la Industria 4.0, el concepto de gemelo digital y el uso de OPC-UA como protocolo de comunicación industrial. A partir de esta base, se planteó un caso de estudio sencillo pero representativo: un depósito de nivel cuya dinámica puede utilizarse para validar la comunicación entre una planta virtual y un controlador industrial.

Para el modelado del depósito se revisaron varios enfoques posibles y se adoptó finalmente un modelo lineal de primer orden. Esta elección permitió disponer de una planta suficientemente clara para analizar el comportamiento del lazo de control, sin introducir una complejidad matemática excesiva. El modelo se implementó en MATLAB/Simulink mediante bloques básicos, configurando un solver de paso fijo y un tiempo de integración compatible con el intercambio de datos con el PLC.

La comunicación entre Simulink y TIA Portal se realizó mediante bloques OPC-UA de lectura y escritura. Simulink escribe en el PLC la altura calculada del depósito, que actúa como variable de proceso, y lee desde el PLC la salida generada por el regulador PID_Compact, que se aplica como caudal de entrada al modelo. Además, se incorporó una señal adicional de perturbación para modificar temporalmente la dinámica de descarga del tanque y comprobar la respuesta del controlador ante variaciones externas.

En TIA Portal se configuró una CPU Siemens S7-1500 simulada con PLCSIM Advanced. Sobre esta CPU se activó el servidor OPC-UA, se crearon los bloques de datos necesarios para el intercambio de variables y se implementó el regulador

mediante el objeto tecnológico PID_Compact. También se desarrolló una lógica auxiliar en el PLC para gestionar el encendido y apagado del sistema, la transferencia de consignas desde la HMI y la aplicación temporal de perturbaciones.

Junto al control, se diseñó una interfaz HMI que permite operar el sistema de forma más cercana a una aplicación industrial. Desde esta pantalla se puede activar o detener el funcionamiento, modificar la consigna de nivel, introducir perturbaciones y supervisar las principales variables del lazo de control. Como apoyo visual, también se desarrolló una animación en MATLAB que representa el nivel del depósito y su evolución temporal durante la simulación.

Por último, se realizaron diferentes pruebas para validar el sistema. En primer lugar, se comprobaron los dos sentidos de comunicación OPC-UA. Después, se verificó que la animación representaba correctamente la evolución calculada por el modelo de Simulink. Posteriormente, se comparó el comportamiento del PID implementado únicamente en Simulink con el regulador PID_Compact ejecutado en TIA Portal. Finalmente, se ensayó el funcionamiento global del sistema ante cambios de consigna y perturbaciones. Los resultados obtenidos permiten afirmar que la arquitectura desarrollada cumple la función prevista y permite controlar correctamente el nivel del depósito virtual.

5.2 Conclusiones sobre los objetivos del proyecto

El objetivo principal del trabajo era diseñar, implementar y validar una arquitectura de control distribuida entre MATLAB/Simulink y TIA Portal mediante OPC-UA. A partir del desarrollo realizado, puede concluirse que este objetivo se ha alcanzado, ya que el sistema permite cerrar el lazo de control entre una planta virtual ejecutada en Simulink y un regulador industrial implementado en un PLC simulado.

En relación con el modelado de la planta, se ha conseguido implementar un modelo dinámico del depósito capaz de representar la evolución del nivel en función del caudal de entrada y de la descarga. Aunque se trata de un modelo simplificado, resulta adecuado para el objetivo del proyecto, ya que permite observar con claridad la respuesta del sistema ante cambios de consigna y perturbaciones. Además, su implementación en Simulink facilita la modificación de parámetros y la visualización del comportamiento de la planta.

También se ha conseguido integrar el modelo con TIA Portal mediante OPC-UA. Las pruebas de comunicación realizadas en ambos sentidos han permitido comprobar que las variables se transmiten correctamente entre Simulink y el PLC. Esta parte era fundamental, ya que el controlador necesita recibir la altura del depósito y devolver una acción de control para que el lazo pueda cerrarse correctamente.

Respecto al controlador, el uso del objeto tecnológico PID_Compact ha permitido trabajar con una herramienta propia del entorno industrial Siemens. El regulador se ha ejecutado de forma cíclica cada 20 ms, manteniendo una relación temporal coherente con los bloques de comunicación OPC-UA configurados en Simulink. Los ensayos realizados muestran que el controlador es capaz de llevar el nivel del depósito hasta la consigna establecida y mantenerlo estable en régimen permanente.

La comparación entre la simulación íntegramente realizada en Simulink y la implementación distribuida con TIA Portal muestra que ambas soluciones permiten regular el sistema, aunque no presentan exactamente la misma respuesta transitoria. En el caso distribuido aparecen diferencias asociadas al muestreo, a la ejecución cíclica del PLC, al intercambio OPC-UA y a la propia implementación interna de PID_Compact. Estas diferencias son esperables y, de hecho, reflejan mejor el comportamiento de una arquitectura de control digital en la que la planta y el controlador no se ejecutan dentro del mismo entorno.

Las pruebas de seguimiento de consigna muestran que el sistema puede trabajar en distintos puntos de operación. El depósito responde correctamente tanto ante incrementos como ante reducciones de la referencia, alcanzando los valores definidos desde la HMI y manteniéndose estable tras el transitorio. También se ha comprobado que el vaciado hasta valores bajos se produce de forma más lenta que el llenado, lo cual es coherente con la dinámica del modelo, ya que el descenso del nivel depende principalmente del caudal de salida.

Por otra parte, las pruebas con perturbaciones permiten comprobar que el regulador mantiene capacidad de corrección cuando se modifica temporalmente la dinámica de descarga del depósito. En los casos ensayados, el nivel se desvía de la referencia, pero el controlador actúa en el sentido adecuado y consigue recuperar la consigna. Esto confirma que el sistema no solo es válido para seguir cambios de referencia, sino también para analizar la respuesta del lazo ante alteraciones externas.

La HMI y la animación desarrollada en MATLAB han resultado útiles para la supervisión del sistema. La HMI permite operar el proceso desde TIA Portal, mientras que la animación facilita la interpretación visual del comportamiento del depósito. La coincidencia entre las señales observadas en TIA Portal y en Simulink confirma que ambas herramientas representan la misma evolución del sistema, cada una desde su función dentro de la arquitectura.

En conjunto, el trabajo demuestra que es posible utilizar Simulink como planta virtual y TIA Portal como entorno de control, comunicando ambos mediante OPC-UA. Esta solución permite ensayar un controlador industrial frente a un modelo dinámico antes de disponer de una planta física, lo que resulta útil para tareas de validación, ajuste preliminar y puesta en marcha virtual.

De forma complementaria, esta aproximación presenta beneficios indirectos desde el punto de vista de la sostenibilidad económica, operativa y ambiental. La posibilidad de validar el lazo de control en un entorno virtual permite anticipar errores, reducir pruebas físicas preliminares y limitar riesgos durante las primeras fases de desarrollo. Aunque el trabajo no cuantifica estos impactos, la arquitectura propuesta muestra cómo la simulación conectada mediante OPC-UA puede contribuir a procesos de ingeniería más eficientes, seguros y orientados a una mejor utilización de recursos.

5.3 Líneas futuras de trabajo

Aunque el sistema desarrollado cumple los objetivos planteados, existen varias mejoras que podrían abordarse en una continuación del proyecto.

Una primera mejora sería automatizar o coordinar mejor la puesta en marcha del sistema. Durante las pruebas se observó que el instante en el que se activa el control desde la HMI influye en el transitorio inicial. Cuando la consigna se aplica justo al iniciar la simulación, el sobreimpulso puede ser mayor que cuando se espera unos segundos. Por ello, sería conveniente implementar una lógica que solo permita activar el regulador cuando el PLC, el servidor OPC-UA, los bloques de comunicación y el modelo de Simulink estén correctamente inicializados.

También sería interesante estudiar con más detalle la influencia de los tiempos de muestreo. En este trabajo se ha utilizado un paso de simulación de 0,01 s en Simulink y un periodo de 0,02 s para la comunicación OPC-UA y la ejecución del regulador. En futuros ensayos podrían compararse distintas combinaciones de tiempos para analizar su efecto sobre el sobreimpulso, el tiempo de establecimiento, la estabilidad y la respuesta ante perturbaciones.

Otra mejora consistiría en incorporar un análisis más cuantitativo de los resultados. En este trabajo la validación se ha basado principalmente en la observación de las gráficas y en la interpretación de la respuesta del sistema. Como ampliación, podrían calcularse indicadores como el tiempo de subida, el tiempo de establecimiento, el sobreimpulso máximo, el error en régimen permanente o el tiempo de recuperación tras una perturbación. Esto permitiría comparar de forma más precisa diferentes configuraciones del controlador.

En cuanto al control, podría estudiarse una sintonización más completa del regulador dentro de la propia arquitectura Simulink–OPC-UA–TIA Portal. Aunque los parámetros obtenidos mediante el autoajuste de TIA Portal han proporcionado una respuesta adecuada, se podrían comparar otros métodos de ajuste y analizar cuál ofrece mejor equilibrio entre rapidez, estabilidad y rechazo de perturbaciones.

El modelo del depósito también podría hacerse más realista. Entre las posibles ampliaciones se encuentran la incorporación de ruido en la medida, retardos

adicionales, variaciones en los parámetros, saturaciones más detalladas o una dinámica de actuadores. También podrían probarse modelos no lineales de descarga para acercar el comportamiento de la planta virtual al de un sistema físico real.

Otra línea de mejora estaría relacionada con la HMI. Podrían añadirse alarmas, históricos de variables, tendencias más configurables, mensajes de estado y una gestión más completa de los modos de operación. De esta forma, la interfaz se aproximaría más a una pantalla de supervisión industrial real.

Finalmente, la continuación más directa del trabajo sería sustituir la CPU simulada por un PLC físico Siemens. En una primera fase, el modelo del depósito podría seguir ejecutándose en Simulink, manteniendo la comunicación mediante OPC-UA. Posteriormente, el sistema podría conectarse a una planta real o a un banco de pruebas físico de nivel. Este paso permitiría comprobar el comportamiento de la arquitectura en condiciones más próximas a una aplicación industrial completa.

Bibliografía

[1] «What is Industrie 4.0?» Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.plattform-i40.de/IP/Navigation/EN/Industrie40/WhatIsIndustrie40/what-is-industrie40.html

[2] «Recommendations for implementing the strategic initiative INDUSTRIE 4.0. Final report of the Industrie 4.0 Working Group», acatech - National Academy of Science and Engineering. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://en.acatech.de/publication/recommendations-for-implementing-the-strategic-initiative-industrie-4-0-final-report-of-the-industrie-4-0-working-group/

[3] «A Practical Guide for Connecting LabVIEW to the Industrial IoT». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.ni.com/en/solutions/industrial-machinery/smart-machine-control/a-practical-guide-for-connecting-labview-to-the-industrial-iot.html

[4] «Estrategia de transformación digital | Siemens Software», Siemens. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.siemens.com/es-es/products/digital-transformation-strategy/

[5] «The Industry 4.0 software for system integration», OPC Router - The Communication Middleware. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.opc-router.com/

[6] P. Liegl, «EDI und sicherer CAD-Daten-austausch als Basis für moderne Industrie-prozesse», ecosio. Accedido: 24 de mayo de 2026. [En línea]. Disponible en: https://ecosio.com/de/blog/edi-und-sicherer-cad-datenaustausch-als-basis-fuer-moderne-industrieprozesse/

[7] «DIN Media Standards Solutions», dinmedia-en. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.dinmedia.de/en/fachdaten-einzelsicht

[8] «MDIS OPC UA Companion Specification - 4.3 OPC UA Overview». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://reference.opcfoundation.org/MDIS/v130/docs/4.3

[9] «UA Part 2: Security - 4.5 OPC UA security architecture». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://reference.opcfoundation.org/Core/Part2/v104/docs/4.5

[10] «IEC 63278-1:2023». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://webstore.iec.ch/en/publication/65628

[11] «Preamble». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://industrialdigitaltwin.io/aas-specifications/IDTA-01001/v3.1/preamble.html

[12] «El gemelo Digital», Siemens. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.siemens.com/es-es/company/insights/digital-twin/

[13] «Digital Twin: Software de Siemens Digital Industries», Siemens. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.siemens.com/es-es/campaigns/digital-twin/

[14] «NI: Getting Complex Products to Market Faster». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.ni.com/en/perspectives/ni-getting-complex-products-to-market-faster.html

[15] «National Instruments. Designing and Developing a Digital Twin for Low-Voltage Vehicular Power Systems. Caso técnico / customer story (PDF)». 2021. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.ni.com/content/dam/web/pdfs/NI_AutoJournal_2021_Q2_DesigningandDevelopingaDigitalTwin_CustomerStory.pdf?srsltid=AfmBOoqy-YhVqupjOwunl68BEEe6L9KOw9eeb7mVUajIoA2nd9awroLS

[16] «What is a Digital Twin? Transform Asset Management Now», OpenText. Accedido: 24 de mayo de 2026. [En línea]. Disponible en: https://www.opentext.com/what-is/digital-twin

[17] «ISO 23247-1:2021», ISO. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.iso.org/standard/75066.html

[18] «AAS Spezifikationen», IDTA. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://industrialdigitaltwin.org/en/content-hub/aasspecifications

[19] «Preamble». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://industrialdigitaltwin.io/aas-specifications/IDTA-01001/v3.1/preamble.html

[20] «OPC-UA-Interoperability-For-Industrie4-and-IoT-EN.pdf». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://opcfoundation.org/wp-content/uploads/2026/01/OPC-UA-Interoperability-For-Industrie4-and-IoT-EN.pdf

[21] M. Krohn, «What is OPC UA? A practical introduction», OPC Router - The Communication Middleware. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.opc-router.com/what-is-opc-ua/

[22] «IEC 62541-3:2020». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://webstore.iec.ch/en/publication/61112

[23] «IEC 62541-5:2020». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://webstore.iec.ch/en/publication/61114

[24] «MDIS OPC UA Companion Specification - 4.3.4.1 Concepts». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://reference.opcfoundation.org/MDIS/v130/docs/4.3.4.1

[25] «Building Secure OPC-UA Server in Node-RED. • FlowFuse». Accedido: 24 de mayo de 2026. [En línea]. Disponible en: https://flowfuse.com/node-red/protocol/opc-ua/

[26] «UA Companion Specifications», OPC Foundation. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://opcfoundation.org/about/opc-technologies/opc-ua/ua-companion-specifications/

[27] «UA Modelling Best Practices - 6 How to create a Companion Specification». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://reference.opcfoundation.org/Model-Best/v102/docs/6

[28] «UA Part 14: PubSub - 5 PubSub Concepts». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://reference.opcfoundation.org/Core/Part14/v105/docs/5

[29] «UA Part 2: Security - 4 OPC UA security architecture». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://reference.opcfoundation.org/Core/Part2/v104/docs/4

[30] «The Industry 4.0 software for system integration», OPC Router - The Communication Middleware. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.opc-router.com/

[31] «IIoT Monitoring | PRTG», Paessler - The Monitoring Experts. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.paessler.com/industry/industrial-iot-monitoring

[32] «Virtual commissioning with the digital twin - Tecnomatix». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://blogs.sw.siemens.com/tecnomatix/virtual-commissioning-with-the-digital-twin/

[33] «Digital Twin», Siemens. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.siemens.com/en-us/company/digital-twin/comprehensive-digital-twin-for-industry/

[34] M. Krohn, «OPC Router Use Cases - our international application examples», OPC Router - The Communication Middleware. Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.opc-router.com/opc-router-international-use-cases/

[35] «Optimice las pruebas de sistemas de control electrónico con un depósito digital ». Accedido: 12 de abril de 2026. [En línea]. Disponible en: https://www.ni.com/es/solutions/aerospace-defense/digital-transformation/optimize-testing-electronic-control-systems-with-digital-depot.html

[36] DeanLonergan, «Water_Tank_Plant_Model/Project Document.pdf at f889b18a156781680baf3a42ae59035292b8fb17 · DeanLonergan/Water_Tank_Plant_Model», GitHub. Accedido: 22 de abril de 2026. [En línea]. Disponible en: https://github.com/DeanLonergan/Water_Tank_Plant_Model/blob/f889b18a156781680baf3a42ae59035292b8fb17/Project%20Document.pdf

[37] S. A. C. Giraldo, «Tanque de Nivel - Modelo Matemático, Simulación y Control», Control Automático Educación. Accedido: 22 de abril de 2026. [En línea]. Disponible en: https://controlautomaticoeducacion.com/analisis-de-sistemas/tanque-nivel/

[38] «Llenado y vaciado de un depósito», Universidad del País Vasco. Accedido: 22 de abril de 2026. [En línea]. Disponible en: http://www.sc.ehu.es/sbweb/fisica3/fluidos/llenado/llenado.html

[39] «Introducción a Simulink». Accedido: 14 de abril de 2026. [En línea]. Disponible en: https://es.mathworks.com/help/simulink/getting-started-with-simulink.html

[40] «Configure OPC UA Client in Simulink - MATLAB & Simulink». Accedido: 14 de abril de 2026. [En línea]. Disponible en: https://es.mathworks.com/help/icomm/ug/configure-opc-ua-client-in-simulink.html

[41] «OPC UA Read - Read data from OPC UA server - Simulink». Accedido: 28 de febrero de 2026. [En línea]. Disponible en: https://es.mathworks.com/help/icomm/ug/opcuaread.html

[42] «OPC UA Write - Write data to OPC UA server - Simulink». Accedido: 13 de febrero de 2026. [En línea]. Disponible en: https://es.mathworks.com/help/icomm/ug/opcuawrite.html

[43] «79047707_PidCompactV2_2_DOC_V2_0_1_en.pdf». Accedido: 14 de abril de 2026. [En línea]. Disponible en: https://cache.industry.siemens.com/dl/files/707/79047707/att_915339/v2/79047707_PidCompactV2_2_DOC_V2_0_1_en.pdf

[44] «Data sheet 6ES7516-3FN02-0AB0, SIMATIC S7-1500F, CPU 1516F-3 PN/DP», Siemens Industry Support. Accedido: 28 de mayo de 2026. [En línea]. Disponible en: https://support.industry.siemens.com/teddatasheet/?caller=SIOS&format=pdf&language=en&mlfbs=6ES7516-3FN02-0AB0

[45] Siemens, «OPC UA communication between SIMATIC S7-1500 and MATLAB/Simulink using SIMATIC PLCSIM Advanced», Siemens Industry Support. Accedido: 14 de abril de 2026. [En línea]. Disponible en: https://cache.industry.siemens.com/dl/files/187/109749187/att_935977/v1/109749187_DIGI_Usecases_OPC_DOC_V10_en.pdf

[46] «PID.pdf». Accedido: 14 de abril de 2026. [En línea]. Disponible en: https://www.eng.newcastle.edu.au/~jhb519/teaching/caut1/Apuntes/PID.pdf

Anexos

En este apartado se recogen los programas principales desarrollados en MATLAB para apoyar la implementación, la visualización y la validación del sistema. Se incluyen únicamente los tres archivos utilizados en la versión final del trabajo: la función de cálculo inicial de parámetros del regulador, la interfaz gráfica de animación del depósito y la S-Function empleada para conectar dicha interfaz con el modelo de Simulink.

Anexo A. Código de la función calcular_PID_tanque_optimizacion.m

El siguiente programa permite obtener una primera estimación de los parámetros del regulador a partir del modelo simplificado del depósito. El cálculo se realiza mediante optimización numérica de la respuesta temporal, evaluando distintas combinaciones de parámetros y conservando aquella que proporciona el menor valor de la función de coste.

```
function resultados = calcular_PID_tanque_optimizacion(A, R, L, Hin)
% calcular_PID_tanque_optimizacion
%
% Funcion para obtener una primera estimacion de los parametros P, I y D
% de un regulador aplicado al modelo simplificado de nivel de un deposito.
%
% El modelo considerado es:
%
%       dh/dt = (1/A)*Qin - (1/(R*A))*h
%
% donde:
%   h    -> altura del liquido
%   Qin  -> caudal de entrada aplicado por el controlador
%   A    -> seccion transversal del deposito
%   R    -> resistencia equivalente asociada a la descarga
%
% La funcion incluye un retardo L en la senal de realimentacion, con el fin
% de representar de forma aproximada el retraso existente entre la planta y
% el controlador.
%
% El ajuste se realiza mediante optimizacion numerica. Para cada conjunto
% de valores P, I y D, el programa simula la respuesta del sistema en lazo
% cerrado y calcula un coste asociado a la calidad de dicha respuesta.
%
% El objetivo de la optimizacion es obtener una respuesta con:
%   - Bajo sobreimpulso
%   - Error reducido a los 20 segundos
%   - Error final practicamente nulo
%   - Ganancias de control razonables
%
% Entradas:
%   A   -> area transversal del deposito
%   R   -> resistencia asociada al caudal de salida
%   L   -> retardo de la senal de realimentacion [s]
%   Hin -> consigna de altura. Si no se introduce, se toma Hin = 8
%
% Ejemplo de uso:
%   calcular_PID_tanque_optimizacion(8, 1, 0.5)
%
```

```
% ============================================================
    % 1. Comprobacion de los parametros de entrada
    % ============================================================
    % Se exige introducir, como minimo, los parametros fisicos A y R,
    % junto con el retardo L. La consigna Hin es opcional.
    if nargin < 3
        error('Debes introducir al menos A, R y L. Ejemplo:
calcular_PID_tanque_optimizacion(8,1,0.5)');
    end

    % Si no se indica una consigna, se utiliza un valor por defecto.
    % Este valor solo afecta a la escala de la respuesta, ya que el modelo
    % empleado es lineal.
    if nargin < 4
        Hin = 8;
    end

    % Validacion basica de los parametros fisicos introducidos.
    % El area y la resistencia deben ser positivas, el retardo no puede ser
    % negativo y la consigna debe ser mayor que cero.
    if A <= 0
        error('A debe ser mayor que cero.');
    end

    if R <= 0
        error('R debe ser mayor que cero.');
    end

    if L < 0
        error('L no puede ser negativo.');
    end

    if Hin <= 0
        error('Hin debe ser mayor que cero.');
    end

    % ============================================================
    % 2. Configuracion de la simulacion
    % ============================================================
    % dt define el paso temporal utilizado para integrar numericamente el
    % modelo. Tsim fija la duracion total de cada simulacion evaluada.
    dt = 0.01;          % Paso de integracion [s]
    Tsim = 30;          % Tiempo total de simulacion [s]

    % ============================================================
    % 3. Puntos iniciales para la optimizacion
    % ============================================================
    % La busqueda se ejecuta desde varios conjuntos iniciales de P, I y D.
    % Esto reduce la dependencia respecto a una unica condicion inicial y
    % aumenta la probabilidad de encontrar una solucion adecuada.
    condicionesIniciales = [
        5.0   0.8   0.2
        7.0   1.0   0.3
        3.0   0.5   0.1
        10.0  1.0   0.1
    ];

    mejorCoste = inf;
    mejorZ = [];

    % Opciones del algoritmo de optimizacion. Se desactiva la salida
    % intermedia por pantalla y se fijan limites de iteraciones, evaluaciones
    % y tolerancias numericas.
    opciones = optimset( ...
        'Display', 'off', ...
        'MaxIter', 800, ...
        'MaxFunEvals', 3000, ...
        'TolX', 1e-8, ...
        'TolFun', 1e-10);

    fprintf('\n');
    fprintf('============================================================\n');
    fprintf(' AJUSTE PID POR OPTIMIZACION NUMERICA\n');
```

```
fprintf('============================================================\n\n');

fprintf('Parametros introducidos:\n');
fprintf('A   = %.6f\n', A);
fprintf('R   = %.6f\n', R);
fprintf('L   = %.6f s\n', L);
fprintf('Hin = %.6f\n\n', Hin);

fprintf('Buscando parametros P, I y D...\n\n');

% ============================================================
% 4. Optimizacion de los parametros del regulador
% ============================================================
% Para cada punto inicial se ejecuta fminsearch. El conjunto que obtenga
% el menor valor de la funcion de coste se conserva como solucion final.
for k = 1:size(condicionesIniciales, 1)

    P0 = condicionesIniciales(k, 1);
    I0 = condicionesIniciales(k, 2);
    D0 = condicionesIniciales(k, 3);

    % Se optimizan los logaritmos de P, I y D en lugar de las ganancias
    % directamente. Al recuperar las ganancias mediante exp(), se evita
    % que el optimizador proponga valores negativos.
    z0 = log([P0, I0, D0]);

    [zOpt, costeOpt] = fminsearch( ...
        @(z) coste_pid(z, A, R, L, Hin, dt, Tsim), ...
        z0, ...
        opciones);

    ganancias = exp(zOpt);

    fprintf('Inicio %d -> P = %.4f, I = %.4f, D = %.4f, coste = %.6f\n', ...
        k, ganancias(1), ganancias(2), ganancias(3), costeOpt);

    % Se guarda la solucion con menor coste.
    if costeOpt < mejorCoste
        mejorCoste = costeOpt;
        mejorZ = zOpt;
    end
end

% ============================================================
% 5. Obtencion de las ganancias finales
% ============================================================
% La mejor solucion se encuentra almacenada en escala logaritmica.
% Se transforma de nuevo a escala normal para obtener P, I y D.
gananciasFinales = exp(mejorZ);

P = gananciasFinales(1);
I = gananciasFinales(2);
D = gananciasFinales(3);

% ============================================================
% 6. Simulacion final con los parametros obtenidos
% ============================================================
% Con la mejor combinacion de parametros se realiza una ultima simulacion
% para calcular los indicadores principales de la respuesta.
[t, h, u] = simular_tanque_pid(P, I, D, A, R, L, Hin, dt, Tsim);

% Sobreimpulso porcentual respecto a la consigna.
sobreimpulso = max(0, max(h) - Hin) / Hin * 100;

% Indice correspondiente aproximadamente al instante t = 20 s.
indice20 = round(20 / dt) + 1;
if indice20 > length(h)
    indice20 = length(h);
end

% Errores porcentuales a los 20 s y al final de la simulacion.
error20 = (h(indice20) - Hin) / Hin * 100;
errorFinal = (h(end) - Hin) / Hin * 100;
```

```
    % ============================================================
    % 7. Salida de resultados por pantalla
    % ============================================================
    fprintf('\n');
    fprintf('============================================================\n');
    fprintf(' RESULTADO FINAL\n');
    fprintf('============================================================\n\n');

    fprintf('Parametros PID calculados:\n');
    fprintf('P = %.6f\n', P);
    fprintf('I = %.6f\n', I);
    fprintf('D = %.6f\n\n', D);

    fprintf('Comportamiento de la respuesta:\n');
    fprintf('Sobreimpulso        = %.4f %%\n', sobreimpulso);
    fprintf('Error a los 20 s    = %.4f %%\n', error20);
    fprintf('Error final         = %.4f %%\n\n', errorFinal);

    fprintf('============================================================\n');

    % ============================================================
    % 8. Estructura de resultados
    % ============================================================
    % Se devuelven los parametros calculados, las metricas de respuesta y
    % los vectores temporales obtenidos en la simulacion final.
    resultados.A = A;
    resultados.R = R;
    resultados.L = L;
    resultados.Hin = Hin;
    resultados.P = P;
    resultados.I = I;
    resultados.D = D;
    resultados.sobreimpulso = sobreimpulso;
    resultados.error20 = error20;
    resultados.errorFinal = errorFinal;
    resultados.t = t;
    resultados.h = h;
    resultados.u = u;

end


% ========================================================================
% FUNCION DE COSTE UTILIZADA POR EL OPTIMIZADOR
% ========================================================================
function J = coste_pid(z, A, R, L, Hin, dt, Tsim)
% Calcula el coste asociado a una combinacion concreta de parametros PID.
%
% El optimizador trabaja con el vector z, que contiene los logaritmos de
% P, I y D. La funcion simula la respuesta del sistema y devuelve un valor
% numerico J: cuanto menor sea J, mejor se considera la respuesta obtenida.

    % Recuperacion de las ganancias reales a partir de la escala logaritmica.
    ganancias = exp(z);

    P = ganancias(1);
    I = ganancias(2);
    D = ganancias(3);

    % Se rechazan directamente ganancias excesivas. Esta penalizacion evita
    % que el optimizador explore zonas poco realistas o numericamente inestables.
    if P > 30 || I > 5 || D > 3
        J = 1e6;
        return;
    end

    % Simulacion temporal del tanque con los parametros propuestos.
    [t, h, u] = simular_tanque_pid(P, I, D, A, R, L, Hin, dt, Tsim);

    % Si la simulacion produce valores no validos, se asigna un coste alto
    % para descartar automaticamente esa combinacion de parametros.
    if any(~isfinite(h)) || any(~isfinite(u))
```

```
        J = 1e6;
        return;
    end

    % ------------------------------------------------------------
    % Indicadores utilizados para evaluar la respuesta
    % ------------------------------------------------------------

    % Sobreimpulso relativo. Solo se penaliza si la altura supera la consigna.
    sobreimpulso = max(0, max(h) - Hin) / Hin;

    % Error relativo en torno a t = 20 s.
    indice20 = round(20 / dt) + 1;
    if indice20 > length(h)
        indice20 = length(h);
    end

    error20 = abs(h(indice20) - Hin) / Hin;

    % Error relativo al final de la simulacion.
    errorFinal = abs(h(end) - Hin) / Hin;

    % IAE: integral del valor absoluto del error.
    % Representa el error acumulado durante toda la simulacion.
    IAE = trapz(t, abs(h - Hin) / Hin) / Tsim;

    % ITAE: integral del error absoluto ponderado por el tiempo.
    % Penaliza especialmente los errores que permanecen al final de la respuesta.
    ITAE = trapz(t, t .* abs(h - Hin) / Hin) / (Tsim^2);

    % ------------------------------------------------------------
    % Funcion objetivo
    % ------------------------------------------------------------
    % El coste combina varios criterios de comportamiento. Los pesos de cada
    % termino se han elegido para priorizar bajo sobreimpulso, buen error a
    % 20 s, error final reducido y ganancias moderadas.
    J = ...
        25  * sobreimpulso^2 + ...
        150 * error20^2 + ...
        50  * errorFinal^2 + ...
        0.5 * IAE + ...
        0.5 * ITAE + ...
        0.005 * D^2 + ...
        1e-5 * P^2 + ...
        1e-4 * I^2;

end


% ========================================================================
% SIMULACION DEL TANQUE EN LAZO CERRADO CON PID Y RETARDO
% ========================================================================
function [t, h, u] = simular_tanque_pid(P, I, D, A, R, L, Hin, dt, Tsim)
% Simula la respuesta temporal del deposito en lazo cerrado.
%
% El controlador se aplica en forma paralela:
%
%       u = P*e + I*integral(e) + D*de/dt
%
% y la planta se integra con una aproximacion de Euler explicita:
%
%       dh/dt = (1/A)*u - (1/(R*A))*h
%
% La senal realimentada al controlador se toma con un retardo L.

    % Numero total de muestras de la simulacion.
    N = round(Tsim / dt) + 1;

    % Inicializacion de los vectores de tiempo, altura y accion de control.
    t = (0:N-1)' * dt;
    h = zeros(N, 1);
    u = zeros(N, 1);
```

```
    % Variable acumuladora de la accion integral.
    integralError = 0;

    % Conversion del retardo temporal L a numero de pasos de simulacion.
    pasosRetardo = round(L / dt);

    % En el instante inicial se considera h = 0, por lo que el error inicial
    % coincide con la consigna.
    errorAnterior = Hin;

    for k = 2:N

        % ---------------------------------------------------------
        % 1. Altura utilizada por el controlador
        % ---------------------------------------------------------
        % El controlador no emplea la altura instantanea, sino una version
        % retardada de la misma para representar el retraso de realimentacion.
        indiceRetardado = k - pasosRetardo;

        if indiceRetardado >= 1
            hRetardada = h(indiceRetardado);
        else
            hRetardada = 0;
        end

        % ---------------------------------------------------------
        % 2. Error de control
        % ---------------------------------------------------------
        % Diferencia entre la consigna y la altura medida con retardo.
        error = Hin - hRetardada;

        % ---------------------------------------------------------
        % 3. Termino integral
        % ---------------------------------------------------------
        % Aproximacion discreta de la integral del error.
        integralError = integralError + error * dt;

        % ---------------------------------------------------------
        % 4. Termino derivativo
        % ---------------------------------------------------------
        % Aproximacion discreta de la derivada del error.
        derivadaError = (error - errorAnterior) / dt;

        % ---------------------------------------------------------
        % 5. Salida del regulador
        % ---------------------------------------------------------
        % La variable u representa el caudal de entrada aplicado al deposito.
        u(k) = P * error + I * integralError + D * derivadaError;

        % ---------------------------------------------------------
        % 6. Actualizacion de la altura del deposito
        % ---------------------------------------------------------
        % Se calcula la derivada de la altura a partir del caudal de entrada
        % y del termino de descarga proporcional al nivel.
        dhdt = (1 / A) * u(k) - (1 / (R * A)) * h(k-1);

        % Integracion explicita de Euler.
        h(k) = h(k-1) + dt * dhdt;

        % Se guarda el error para calcular la derivada en el siguiente paso.
        errorAnterior = error;

        % Control de seguridad numerica. Si la respuesta crece de forma
        % excesiva, se marca la simulacion como no valida.
        if abs(h(k)) > 1e6 || abs(u(k)) > 1e6
            h(k:end) = NaN;
            u(k:end) = NaN;
            return;
        end
    end

end
```

Anexo B. Código de la función tank_anim_gui_simple.m

El siguiente programa define la interfaz gráfica utilizada para representar la evolución del nivel del depósito. La ventana incorpora una animación del tanque, el valor numérico de la altura, una gráfica temporal de la variable controlada y un botón para iniciar o detener la simulación del modelo de Simulink.

```
function varargout = tank_anim_gui_simple(action, varargin)
% tank_anim_gui_simple
%
% Ventana auxiliar para visualizar el nivel del depósito durante la simulación.
% La función se controla mediante el argumento "action", que permite crear la
% ventana, actualizar el dibujo, reiniciar la gráfica, arrancar/parar el modelo
% o cerrar la interfaz.
%
% Acciones disponibles:
%   'get_or_create' -> crea la ventana si no existe o recupera la existente
%   'update'        -> actualiza el nivel del tanque y la gráfica h(t)
%   'reset_plot'    -> borra los puntos de la gráfica temporal
%   'toggle_sim'    -> arranca o detiene el modelo de Simulink
%   'close'         -> cierra la ventana de animación

if nargin < 1
    action = 'get_or_create';
end

switch lower(action)

    case 'get_or_create'
        % Obtiene la ventana de animación. Si no existe, la crea.
        [fig, h] = LocalGetOrCreateFigure();
        varargout{1} = fig;

        if nargout > 1
            varargout{2} = h;
        end

    case 'update'
        % Actualiza el dibujo con la altura actual y el tiempo de simulación.
        tankheight = varargin{1};
        simTime    = varargin{2};

        [~, h] = LocalGetOrCreateFigure();
        LocalUpdateDrawing(h, tankheight, simTime);
        LocalRefreshRunButton(h);

    case 'reset_plot'
        % Borra la curva de nivel acumulada en la gráfica.
        [~, h] = LocalGetOrCreateFigure();
        LocalResetPlot(h);

    case 'toggle_sim'
        % Cambia el estado de la simulación: si está parada, la inicia;
        % si está en ejecución, la detiene.
        [~, h] = LocalGetOrCreateFigure();
        LocalToggleSimulation(h);

    case 'close'
        % Cierra la ventana si está abierta.
        fig = LocalGetFigure();

        if ishghandle(fig, 'figure')
            delete(fig);
        end
```

```
end

end


function mdl = LocalModelName()
% Nombre del modelo de Simulink que se controla desde el botón de la interfaz.

mdl = 'Water_Tank_Plant_Model';

end


function fig = LocalGetFigure()
% Busca una figura abierta con la etiqueta asignada a la animación.
% Si existe, devuelve su manejador; si no existe, devuelve -1.

fig = findall(0, 'Type', 'figure', 'Tag', 'SimpleTankAnimationFigure');

if isempty(fig)
    fig = -1;
else
    fig = fig(1);
end

end


function [fig, h] = LocalGetOrCreateFigure()
% Recupera la ventana existente o crea una nueva si todavía no se ha abierto.
% También comprueba que los elementos principales de la interfaz siguen siendo
% válidos. Si la figura está corrupta o incompleta, se elimina y se genera otra.

fig = LocalGetFigure();

if ishghandle(fig, 'figure')

    h = get(fig, 'UserData');

    if isempty(h) || ...
       ~isfield(h, 'ax') || ~ishghandle(h.ax) || ...
       ~isfield(h, 'axPlot') || ~ishghandle(h.axPlot)

        delete(fig);
        [fig, h] = LocalCreateFigure();
    end

else
    [fig, h] = LocalCreateFigure();
end

end


function [fig, h] = LocalCreateFigure()
% Crea desde cero la ventana de animación, incluyendo:
%   - Botón de arranque/parada de la simulación
%   - Indicador textual de estado
%   - Valor numérico del nivel
%   - Dibujo del depósito y tuberías
%   - Gráfica temporal del nivel h(t)

% ============================================================
% Parámetros visuales generales
% ============================================================
edge       = 18;      % Anchura visual del tanque
tankTop    = 26;      % Altura visual del tanque
bgColor    = [0.92 0.92 0.92];
tankColor  = [1 1 1];
pipeColor  = [0.22 0.22 0.65];
waterColor = [0.25 0.33 0.75];
lineColor  = [0 0 0];
```

```
% ------------------------------------------------------------
% Geometría de la tubería superior de entrada
% ------------------------------------------------------------
topThk   = 1.35;      % Espesor de la tubería
topHoriz = 14.5;      % Longitud horizontal
topDrop  = 6.2;       % Bajada vertical hacia el tanque

topRightX = edge/2 + 1.8;
topLeftX  = topRightX - topHoriz;
topBaseY  = tankTop + 7.0;

% ------------------------------------------------------------
% Geometría de la tubería inferior de salida
% ------------------------------------------------------------
outThk   = 1.60;      % Espesor de la tubería de salida
outHoriz = 7.0;       % Longitud horizontal de salida
outDrop  = 5.6;       % Bajada vertical de la salida

% ============================================================
% Creación de la ventana principal
% ============================================================
fig = figure( ...
    'Visible', 'off', ...
    'Tag', 'SimpleTankAnimationFigure', ...
    'NumberTitle', 'off', ...
    'Name', 'Tank Animation', ...
    'MenuBar', 'none', ...
    'ToolBar', 'none', ...
    'Color', bgColor, ...
    'HandleVisibility', 'callback', ...
    'CloseRequestFcn', 'tank_anim_gui_simple(''close'')');

% ============================================================
% Panel izquierdo: botón, estado y valor numérico
% ============================================================

% Botón para arrancar o detener la simulación de Simulink.
h.ui.runstop = uicontrol( ...
    'Parent', fig, ...
    'Style', 'pushbutton', ...
    'Units', 'normalized', ...
    'Position', [0.03 0.84 0.17 0.08], ...
    'String', 'START SIM', ...
    'FontWeight', 'bold', ...
    'Callback', @(src,evt)tank_anim_gui_simple('toggle_sim'));

% Texto que indica si la simulación está parada o en ejecución.
h.ui.status = uicontrol( ...
    'Parent', fig, ...
    'Style', 'text', ...
    'Units', 'normalized', ...
    'Position', [0.03 0.77 0.17 0.04], ...
    'BackgroundColor', bgColor, ...
    'HorizontalAlignment', 'left', ...
    'String', 'Stopped');

% Etiqueta del valor mostrado.
h.ui.value_label = uicontrol( ...
    'Parent', fig, ...
    'Style', 'text', ...
    'Units', 'normalized', ...
    'Position', [0.03 0.69 0.17 0.04], ...
    'BackgroundColor', bgColor, ...
    'HorizontalAlignment', 'left', ...
    'FontWeight', 'bold', ...
    'String', 'Tank level');

% Valor numérico de la altura real del tanque.
h.ui.value_text = uicontrol( ...
    'Parent', fig, ...
    'Style', 'text', ...
    'Units', 'normalized', ...
    'Position', [0.03 0.65 0.17 0.04], ...
```

```
    'BackgroundColor', bgColor, ...
    'HorizontalAlignment', 'left', ...
    'String', '0.00');

% ============================================================
% Eje de animación del depósito
% ============================================================
ax = axes( ...
    'Parent', fig, ...
    'Position', [0.24 0.08 0.38 0.84], ...
    'Visible', 'off', ...
    'Tag', 'picture', ...
    'Color', bgColor, ...
    'DataAspectRatio', [1 1 1]);

hold(ax, 'on');
set(ax, 'SortMethod', 'childorder');

% Variables que se guardan para poder actualizar la animación posteriormente.
h.ax = ax;
h.edge = edge;
h.tankTop = tankTop;
h.displayMaxLevel = 40;   % Valor real que se representa como depósito lleno
h.outThk = outThk;
h.outHoriz = outHoriz;
h.outDrop = outDrop;

% Datos de la tubería superior, necesarios para actualizar las gotas.
h.top.topRightX = topRightX;
h.top.topBaseY  = topBaseY;
h.top.topThk    = topThk;
h.top.topDrop   = topDrop;

% ============================================================
% Eje de la gráfica temporal h(t)
% ============================================================
axPlot = axes( ...
    'Parent', fig, ...
    'Position', [0.66 0.14 0.30 0.72], ...
    'Box', 'on', ...
    'Color', [1 1 1], ...
    'XGrid', 'on', ...
    'YGrid', 'on', ...
    'FontSize', 10);

hold(axPlot, 'on');
xlabel(axPlot, 't [s]');
ylabel(axPlot, 'h');
title(axPlot, 'Evolución del nivel');
xlim(axPlot, [0 30]);
ylim(axPlot, [0 40]);

h.axPlot = axPlot;
h.plot.line = animatedline(axPlot, 'LineWidth', 1.8);
h.plot.window = 30;   % Ventana visible de la gráfica, en segundos

% ============================================================
% Dibujo del contorno del tanque
% ============================================================
tankX = [0 edge edge 0 0];
tankY = [0 0 tankTop tankTop 0];

h.patch.tankBody = patch( ...
    'Parent', ax, ...
    'XData', tankX, ...
    'YData', tankY, ...
    'FaceColor', tankColor, ...
    'EdgeColor', lineColor, ...
    'LineWidth', 2.6);

% ============================================================
% Dibujo de la tubería superior de entrada
% ============================================================
```

```
topPipeX = [ ...
    topLeftX, ...
    topRightX-topThk, ...
    topRightX-topThk, ...
    topRightX, ...
    topRightX, ...
    topLeftX, ...
    topLeftX];

topPipeY = [ ...
    topBaseY, ...
    topBaseY, ...
    topBaseY-topDrop, ...
    topBaseY-topDrop, ...
    topBaseY+topThk, ...
    topBaseY+topThk, ...
    topBaseY];

h.patch.topPipe = patch( ...
    'Parent', ax, ...
    'XData', topPipeX, ...
    'YData', topPipeY, ...
    'FaceColor', pipeColor, ...
    'EdgeColor', lineColor, ...
    'LineWidth', 2.6);

% ============================================================
% Dibujo de la tubería inferior de salida
% ============================================================
bottomPipeX = [ ...
    edge, ...
    edge+outHoriz, ...
    edge+outHoriz, ...
    edge+outHoriz+outThk, ...
    edge+outHoriz+outThk, ...
    edge, ...
    edge];

bottomPipeY = [ ...
    0, ...
    0, ...
    -outDrop, ...
    -outDrop, ...
    outThk, ...
    outThk, ...
    0];

h.patch.bottomPipeBody = patch( ...
    'Parent', ax, ...
    'XData', bottomPipeX, ...
    'YData', bottomPipeY, ...
    'FaceColor', tankColor, ...
    'EdgeColor', lineColor, ...
    'LineWidth', 2.6);

% ============================================================
% Agua dentro del tanque
% ============================================================
firsty = tankTop/2;

% Zona azul correspondiente al agua.
h.patch.mainWater = patch( ...
    'Parent', ax, ...
    'XData', tankX, ...
    'YData', [0 0 firsty firsty 0], ...
    'FaceColor', waterColor, ...
    'EdgeColor', 'none');

% Zona blanca superior que representa el volumen sin agua.
h.patch.airMain = patch( ...
    'Parent', ax, ...
    'XData', tankX, ...
    'YData', [firsty firsty tankTop tankTop firsty], ...
```

```
    'FaceColor', tankColor, ...
    'EdgeColor', 'none');

% ============================================================
% Agua en la tubería inferior
% ============================================================
tubeheight = min(firsty, outThk);

h.patch.bottomPipeWater = patch( ...
    'Parent', ax, ...
    'XData', bottomPipeX, ...
    'YData', [ ...
        0, ...
        0, ...
        -outDrop, ...
        -outDrop, ...
        tubeheight, ...
        tubeheight, ...
        0], ...
    'FaceColor', waterColor, ...
    'EdgeColor', 'none');

% ============================================================
% Líneas de entrada de agua desde la tubería superior
% ============================================================
nTop = 5;
topDripX = linspace(topRightX-topThk+0.16, topRightX-0.16, nTop);
topDripDx = [-0.18 -0.09 0 0.09 0.18];

for k = 1:nTop

    x0 = topDripX(k);
    y0 = topBaseY-topDrop-0.10;
    y1 = tankTop - 2.2;

    h.drip.top(k) = line( ...
        [x0 x0 + topDripDx(k)], ...
        [y0 y1], ...
        'Parent', ax, ...
        'Color', pipeColor, ...
        'LineWidth', 2.0, ...
        'Clipping', 'off');
end

% ============================================================
% Líneas de salida de agua por la tubería inferior
% ============================================================
nBot = 5;
botDripX = linspace(edge+outHoriz+0.18, edge+outHoriz+outThk-0.18, nBot);
botDripDx = [-0.28 -0.14 0 0.14 0.28];

for k = 1:nBot

    x0 = botDripX(k);
    y0 = -outDrop - 0.10;
    y1 = -outDrop - 3.4;

    h.drip.bot(k) = line( ...
        [x0 x0 + botDripDx(k)], ...
        [y0 y1], ...
        'Parent', ax, ...
        'Color', pipeColor, ...
        'LineWidth', 2.0, ...
        'Clipping', 'off');
end

% ============================================================
% Límites del dibujo
% ============================================================
set(ax, ...
    'XLim', [topLeftX-3 edge+outHoriz+outThk+5], ...
    'YLim', [-outDrop-5 topBaseY+topThk+3]);
```

```
% Se guarda toda la información gráfica en la figura para poder recuperarla
% desde las funciones de actualización.
set(fig, 'UserData', h, 'Visible', 'on');

LocalRefreshRunButton(h);
drawnow;

end


function LocalUpdateDrawing(h, tankheight, simTime)
% Actualiza la animación y la gráfica temporal con los valores procedentes
% de la simulación de Simulink.

tankheight = double(tankheight);
simTime    = double(simTime);

% ============================================================
% Escalado de la altura real a la altura visual del dibujo
% ============================================================
displayHeight = max(min(tankheight, h.displayMaxLevel), 0);
displayHeight = h.tankTop * displayHeight / h.displayMaxLevel;

% Actualización de la zona de agua del tanque.
set(h.patch.mainWater, ...
    'YData', [0 0 displayHeight displayHeight 0]);

% Actualización de la zona de aire. Si el tanque está lleno visualmente,
% se oculta la zona blanca superior.
if displayHeight <= h.tankTop
    set(h.patch.airMain, ...
        'Visible', 'on', ...
        'YData', [displayHeight displayHeight h.tankTop h.tankTop displayHeight]);
else
    set(h.patch.airMain, 'Visible', 'off');
end

% ============================================================
% Actualización del agua en la tubería inferior
% ============================================================
tubeheight = min(max(displayHeight, 0), h.outThk);

edge = h.edge;
outHoriz = h.outHoriz;
outThk = h.outThk;
outDrop = h.outDrop;

bottomPipeX = [ ...
    edge, ...
    edge+outHoriz, ...
    edge+outHoriz, ...
    edge+outHoriz+outThk, ...
    edge+outHoriz+outThk, ...
    edge, ...
    edge];

bottomPipeY = [ ...
    0, ...
    0, ...
    -outDrop, ...
    -outDrop, ...
    tubeheight, ...
    tubeheight, ...
    0];

set(h.patch.bottomPipeWater, ...
    'XData', bottomPipeX, ...
    'YData', bottomPipeY);

% Las líneas de salida se muestran solo cuando hay agua en el depósito.
if tankheight > 0.02
    set(h.drip.bot, 'Visible', 'on');
else
```

```
    set(h.drip.bot, 'Visible', 'off');
end

% ============================================================
% Actualización de las líneas de entrada
% ============================================================
nTop = numel(h.drip.top);
topRightX = h.top.topRightX;
topBaseY  = h.top.topBaseY;
topThk    = h.top.topThk;
topDrop   = h.top.topDrop;

topDripX = linspace(topRightX-topThk+0.16, topRightX-0.16, nTop);
topDripDx = [-0.18 -0.09 0 0.09 0.18];

% El extremo inferior de las líneas de entrada se adapta al nivel visual.
yEnd = max(displayHeight + 0.4, h.tankTop - 2.2);

for k = 1:nTop

    x0 = topDripX(k);
    y0 = topBaseY-topDrop-0.10;

    set(h.drip.top(k), ...
        'XData', [x0 x0 + topDripDx(k)], ...
        'YData', [y0 yEnd], ...
        'Visible', 'on');
end

% Se muestra el valor real de altura, no el valor escalado de la animación.
set(h.ui.value_text, 'String', sprintf('%.2f', tankheight));

% ============================================================
% Actualización de la gráfica temporal h(t)
% ============================================================
addpoints(h.plot.line, simTime, tankheight);

% La gráfica muestra una ventana temporal móvil. Al principio se mantiene
% entre 0 y 30 s; después se desplaza con el tiempo de simulación.
if simTime <= h.plot.window
    xlim(h.axPlot, [0 h.plot.window]);
else
    xlim(h.axPlot, [simTime - h.plot.window, simTime]);
end

% Si el nivel supera el límite superior de la gráfica, se amplía el eje Y.
yl = ylim(h.axPlot);

if tankheight > yl(2)
    ylim(h.axPlot, [0 max(tankheight*1.10, yl(2)+1)]);
end

% Refresco limitado para evitar sobrecargar la simulación.
drawnow limitrate nocallbacks;

end


function LocalResetPlot(h)
% Borra la curva almacenada y devuelve los ejes de la gráfica a sus límites
% iniciales.

clearpoints(h.plot.line);
xlim(h.axPlot, [0 h.plot.window]);
ylim(h.axPlot, [0 40]);

drawnow;

end


function LocalToggleSimulation(h)
% Arranca o detiene el modelo de Simulink asociado a la animación.
```

```
% Si el modelo no está cargado, se abre automáticamente.

mdl = LocalModelName();

if ~bdIsLoaded(mdl)
    open_system(mdl);
end

status = get_param(mdl, 'SimulationStatus');

switch lower(status)

    case 'running'
        set_param(mdl, 'SimulationCommand', 'stop');

    otherwise
        set_param(mdl, 'SimulationCommand', 'start');
end

LocalRefreshRunButton(h);

end


function LocalRefreshRunButton(h)
% Actualiza el texto del botón y del indicador de estado según el estado
% actual de la simulación de Simulink.

mdl = LocalModelName();

if ~bdIsLoaded(mdl)
    set(h.ui.runstop, 'String', 'START SIM');
    set(h.ui.status, 'String', 'Stopped');
    return
end

status = get_param(mdl, 'SimulationStatus');

switch lower(status)

    case 'running'
        set(h.ui.runstop, 'String', 'STOP SIM');
        set(h.ui.status, 'String', 'Running');

    otherwise
        set(h.ui.runstop, 'String', 'START SIM');
        set(h.ui.status, 'String', 'Stopped');
end

end
```

Anexo C. Código de la S-Function tank_anim_sfun_simple.m

El siguiente programa define la S-Function de nivel 2 empleada como bloque de visualización en Simulink. Su función es recibir la altura calculada por el modelo, obtener el tiempo actual de simulación y enviar ambos valores a la interfaz gráfica para actualizar la animación y la curva temporal.

```
function tank_anim_sfun_simple(block)
% tank_anim_sfun_simple
%
% S-Function de nivel 2 utilizada para conectar la salida del modelo de
% Simulink con la interfaz gráfica del tanque.
%
% Esta función no calcula ninguna dinámica del sistema. Su única finalidad
% es recibir la altura del depósito desde Simulink y enviarla a la ventana
% de animación para actualizar el dibujo y la gráfica en tiempo real.

    setup(block);

end


function setup(block)
% Configuración inicial de la S-Function.
%
% En esta función se definen los puertos de entrada y salida, el tiempo de
% muestreo y los métodos que se ejecutan al iniciar, actualizar y finalizar
% la simulación.

    % La S-Function no utiliza parámetros configurables desde el bloque.
    block.NumDialogPrms = 0;

    % Se define un único puerto de entrada y ningún puerto de salida.
    % La entrada corresponde al nivel del tanque calculado por el modelo.
    block.NumInputPorts  = 1;
    block.NumOutputPorts = 0;

    % Configuración del puerto de entrada:
    %   DatatypeID = 0  -> tipo double
    %   Dimensions = 1  -> señal escalar
    %   DirectFeedthrough = true -> la salida visual depende directamente
    %                               del valor de entrada actual
    block.InputPort(1).DatatypeID        = 0;
    block.InputPort(1).Dimensions        = 1;
    block.InputPort(1).DirectFeedthrough = true;

    % Tiempo de muestreo heredado/continuo.
    % Con [0 0], el bloque se actualiza siguiendo el ritmo de la simulación.
    block.SampleTimes = [0 0];

    % Configuración de compatibilidad del estado de simulación.
    block.SimStateCompliance = 'DefaultSimState';

    % Se indica a Simulink que este bloque actúa como dispositivo de
    % visualización. De esta forma, se evita que se trate como un bloque
    % que modifica la dinámica del modelo.
    block.SetSimViewingDevice(true);

    % Registro de los métodos principales de ejecución del bloque:
    %   Start     -> se ejecuta al comenzar la simulación
    %   Update    -> se ejecuta durante la simulación para actualizar la GUI
    %   Terminate -> se ejecuta al finalizar la simulación
    block.RegBlockMethod('Start',     @mdlStart);
    block.RegBlockMethod('Update',    @mdlUpdate);
```

```
    block.RegBlockMethod('Terminate', @mdlTerminate);

end


function mdlStart(block)
% Método ejecutado al iniciar la simulación.
%
% Se crea la ventana de animación si aún no existe y se borra la gráfica
% anterior para que cada simulación empiece con una representación limpia.

    tank_anim_gui_simple('get_or_create');
    tank_anim_gui_simple('reset_plot');

end


function mdlUpdate(block)
% Método ejecutado durante la simulación.
%
% En cada actualización se toma el valor actual de la altura del tanque y
% el tiempo de simulación. Ambos valores se envían a la interfaz gráfica
% para actualizar el nivel dibujado y añadir un nuevo punto a la gráfica.

    % Altura actual del tanque procedente de la entrada del bloque.
    tankheight = block.InputPort(1).Data;

    % Tiempo actual de simulación de Simulink.
    simTime = block.CurrentTime;

    % Actualización de la animación y de la curva h(t).
    tank_anim_gui_simple('update', tankheight, simTime);

end


function mdlTerminate(block)
% Método ejecutado al finalizar la simulación.
%
% La ventana no se cierra automáticamente para que el usuario pueda revisar
% la última gráfica obtenida después de detener la simulación.

end
```