UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Industrial

Desarrollo e implementación del sistema de control industrial de un biorreactor para la producción de etanol.

Trabajo Fin de Máster

Máster Universitario en Ingeniería Industrial (Acceso desde Grado I. Electrónica Industrial y Automática)

AUTOR/A: Pérez Gutiérrez, Daniel

Tutor/a: Blasco Ferragud, Francesc Xavier

CURSO ACADÉMICO: 2022/2023

AGRADECIMIENTOS

Quiero expresar mi más sincero agradecimiento en este trabajo, que marca el fin de mi trayectoria universitaria, a todas las personas que han contribuido con su ayuda y apoyo para alcanzar esta valiosa meta.

Quiero dedicar un especial agradecimiento a mi familia, quienes han sido el pilar fundamental que me ha apoyado incondicionalmente a lo largo de mis estudios. Su amor, apoyo y constante motivación han sido el motor que me ha impulsado a dar lo mejor de mí en este proyecto y en cada paso de mi formación académica. Agradezco profundamente su presencia y confianza en mí, y valoro enormemente el tiempo y los sacrificios que han dedicado para mi crecimiento y éxito. Su inquebrantable apoyo ha sido crucial en cada etapa de este camino y estoy infinitamente agradecido por tenerlos a mi lado.

Un agradecimiento especial a Ekko, mi leal compañero canino, por estar siempre a mi lado y brindarme su cariño durante todo el proceso del proyecto. Tu presencia ha sido reconfortante y has hecho de este viaje una experiencia aún más especial.

Quiero agradecer de todo corazón a mis amigos y compañeros, en especial a Miguel Padrón González y Jesús Elías Soto. Han estado a mi lado durante todo este proyecto, brindándome su apoyo incondicional y compartiendo momentos inolvidables.

En especial, quiero expresar mi más sincero agradecimiento a mi tutor, Francesc Xavier Blasco Ferragud, por su guía y asesoramiento durante todo el proceso. Su experiencia y dedicación han sido invaluables para la realización de este proyecto. Agradezco su apoyo constante, sus sugerencias y su infinita paciencia, los cuales han sido fundamentales para superar los desafíos y alcanzar los objetivos establecidos. Sus valiosas contribuciones han sido indispensables para el éxito de este proyecto.

Por último, quiero agradecer a todas las fuentes bibliográficas, cursos y documentos que he consultado durante la realización de este proyecto. Su información ha enriquecido mi trabajo y ha sido de gran utilidad para el desarrollo y fundamentación de las ideas presentadas.

RESUMEN

El presente trabajo se enfoca en el desarrollo e implementación de un sistema de control industrial para el control de un biorreactor destinado a la producción de etanol. Dado el nivel de complejidad del proceso, se ha optado por utilizar un enfoque basado en un gemelo digital.

El proyecto se divide en varias fases fundamentales:

1. Implementación de un modelo no lineal del proceso para la creación del gemelo digital. Se desarrollarán dos versiones del modelo: una que se conectará al controlador programable lógico (PLC) para simular operaciones realistas y otra que se utilizará en Matlab para el desarrollo y ajuste de la estructura de control.
2. Desarrollo, ajuste y validación de la estructura de control más adecuada. Se obtendrán modelos lineales a partir del modelo no lineal, se desarrollará la estructura de control basada en estos modelos lineales y se validará su rendimiento utilizando el modelo no lineal como referencia.
3. Implementación de la estructura de control en un PLC industrial y realización de ensayos de validación. Se transferirá la estructura de control desarrollada al entorno de control del PLC, que se realizará mediante el software Codesys, y se llevarán a cabo pruebas exhaustivas para asegurar su correcto funcionamiento y desempeño.
4. Desarrollo gracias al programa IGSS de un sistema de supervisión, control y adquisición de datos (SCADA) industrial para la monitorización y operación del proceso. Se crearán interfaces gráficas de usuario que permitirán a los operadores visualizar y operar el sistema, además de proporcionar acceso a registros históricos y otras funcionalidades relevantes.
5. Documentación exhaustiva del proyecto. Se elaborará un informe detallado que describa en profundidad cada etapa del desarrollo, incluyendo los aspectos teóricos, los métodos utilizados, los resultados obtenidos y las conclusiones relevantes.

Mediante este enfoque integral, se busca establecer un sistema de control eficiente y confiable para optimizar la producción de etanol en el biorreactor, asegurando la calidad del producto y minimizando los costos operativos.

Palabras Clave: Control industrial, Biorreactor, Etanol, Gemelo digital, Modelo no lineal, PLC, SCADA, Matlab, Codesys, IGSS.

RESUM

El present treball se centra en el desenvolupament i implementació d'un sistema de control industrial per al control d'un biorreactor destinat a la producció d'etanol. Donat el nivell de complexitat del procés, s'ha optat per utilitzar un enfocament basat en un bessó digital.

El projecte es divideix en diverses fases fonamentals:

1. Implementació d'un model no lineal del procés per a la creació del bessó digital. Es desenvoluparan dues versions del model: una que es connectarà al controlador programable lògic (PLC) per a simular operacions realistes i una altra que s'utilitzarà en Matlab per al desenvolupament i ajust de l'estructura de control.
2. Desenvolupament, ajust i validació de l'estructura de control més adequada. S'obtindran models lineals a partir del model no lineal, es desenvoluparà l'estructura de control basada en aquests models lineals i es validarà el seu rendiment utilitzant el model no lineal com a referència.
3. Implementació de l'estructura de control en un PLC industrial i realització d'assaigs de validació. Es transferirà l'estructura de control desenvolupada a l'entorn de control del PLC, que es realitzarà mitjançant el programari Codesys, i es duran a terme proves exhaustives per assegurar el seu correcte funcionament i rendiment.
4. Desenvolupament gràcies al programa IGSS d'un sistema de supervisió, control i adquisició de dades (SCADA) industrial per a la monitorització i operació del procés. Es crearan interfícies gràfiques d'usuari que permetran als operadors visualitzar i operar el sistema, a més de proporcionar accés a registres històrics i altres funcionalitats rellevants.
5. Documentació exhaustiva del projecte. Es redactarà un informe detallat que descriurà en profunditat cada etapa del desenvolupament, incloent els aspectes teòrics, els mètodes utilitzats, els resultats obtinguts i les conclusions rellevants.

Mitjançant aquest enfocament integral, es busca establir un sistema de control eficient i fiable per a optimitzar la producció d'etanol en el biorreactor, assegurant la qualitat del producte i minimitzant els costos operatius.

Paraules clau: Control industrial, Biorreactor, Etanol, Bessó digital, Model no lineal, PLC, SCADA, Matlab, Codesys, IGSS.

ABSTRACT

The present work focuses on the development and implementation of an industrial control system for the control of a bioreactor intended for ethanol production. Given the level of complexity of the process, an approach based on a digital twin has been chosen.

The project is divided into several fundamental phases:

1. Implementation of a non-linear model of the process for the creation of the digital twin. Two versions of the model will be developed: one that will be connected to the programmable logic controller (PLC) to simulate realistic operations, and another that will be used in Matlab for the development and tuning of the control structure.

2. Development, tuning, and validation of the most suitable control structure. Linear models will be derived from the non-linear model, the control structure based on these linear models will be developed, and its performance will be validated using the non-linear model as a reference.

3. Implementation of the control structure in an industrial PLC and conducting validation tests. The developed control structure will be transferred to the control environment of the PLC, which will be done using the Codesys software, and comprehensive tests will be carried out to ensure its correct functioning and performance.

4. Development of an industrial Supervisory Control and Data Acquisition (SCADA) system using the IGSS program for process monitoring and operation. User graphical interfaces will be created, allowing operators to visualize and operate the system, as well as providing access to historical records and other relevant functionalities.

5. Comprehensive documentation of the project. A detailed report will be elaborated, describing in-depth each stage of the development, including theoretical aspects, methods used, results obtained, and relevant conclusions.

Through this comprehensive approach, the aim is to establish an efficient and reliable control system to optimize ethanol production in the bioreactor, ensuring product quality and minimizing operating costs.

Keywords: Industrial control, Bioreactor, Ethanol, Digital twin, Non-linear model, PLC, SCADA, Matlab, Codesys, IGSS.

ÍNDICE GENERAL

ÍNDICE

Escuela Técnica Superior de Ingeniería Industrial

ETSII

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Máster Universitario en Ingeniería Industrial (Acceso desde Grado I. Electrónica Industrial y Automática)

Desarrollo e implementación de un sistema de control industrial de un biorreactor para la producción de etanol

Trabajo Fin de Máster

1. MEMORIA

Autor: Daniel Pérez Gutiérrez

Tutor académico: Francesc Xavier Blasco Ferragud

Julio 2023

CAPÍTULO 1. MEMORIA TÉCNICA

1.1 Objeto del proyecto

1.1.1 Objeto técnico

El objeto técnico de este trabajo es el desarrollo e implementación de un sistema de control industrial para el biorreactor de producción de etanol. Este sistema tiene como objetivo principal regular y optimizar las variables de proceso del biorreactor, como la temperatura, la concentración de nutrientes y el caudal, con el fin de asegurar un rendimiento eficiente y consistente en la producción de etanol. El objeto técnico abarca la creación de un gemelo digital del proceso, la obtención y ajuste de modelos de control, la implementación de la estructura de control en un PLC industrial, el desarrollo de un sistema SCADA para la supervisión y operación del proceso, y la documentación de todo el proyecto.

1.1.2 Objeto académico

El objeto académico del proyecto es la realización de la asignatura Trabajo Fin de Máster, que pertenece al segundo año del Máster Universitario en Ingeniería Industrial por la Universidad Politécnica de Valencia. Para su elaboración, se ha hecho uso de los conocimientos aprendidos en el máster, en especial en la asignatura Instrumentación y Control Industrial y de la ayuda facilitada por el tutor académico. Se obtienen nuevos conocimientos de control y automática industrial que complementan a los ya adquiridos.

1.2 Alcance

El alcance de este proyecto incluye las siguientes actividades y entregables:

1. Obtención del modelo no lineal mediante programación en Matlab, a partir de un documento detallado [1] que describe las ecuaciones que definen el modelo del proceso del biorreactor de producción de etanol. Se llevará a cabo una implementación precisa de estas ecuaciones en Matlab para obtener el modelo no lineal requerido que conformará en gemelo digital.
2. Desarrollo, ajuste y validación de la estructura de control más adecuada, utilizando el software Simulink junto con Matlab.
3. Implementación de la estructura de control en un PLC industrial mediante Codesys, asegurando una comunicación adecuada por MODBUS TCP/IP y la correcta configuración de los parámetros de control.
4. Realización de ensayos de validación para verificar el funcionamiento y desempeño del sistema de control implementado.

5. Desarrollo de un sistema SCADA industrial en el programa IGSS que permita la monitorización y operación del proceso. Esto incluye la creación de pantallas de operación, registros históricos y otras funcionalidades relevantes.
6. Documentación completa del proyecto, incluyendo informes técnicos, manual de operación, diagramas de control y cualquier otra documentación necesaria para comprender y replicar el sistema de control implementado.

1.3 Antecedente

El antecedente de este proyecto surge de la necesidad de mejorar el control de los biorreactores en la producción de etanol, buscando optimizar su rendimiento y eficiencia. Se plantea el desarrollo e implementación de un sistema de control industrial mediante un gemelo digital y un autómata programable simulado. Este proyecto pretende superar las limitaciones existentes y lograr un control más preciso y adaptable para maximizar la producción de etanol.

1.4 Normas y referencias

- Norma **UNE 157001** (Criterios generales para la elaboración formal de los documentos que constituyen un proyecto técnico)
- Norma **ISA 5.1-1984** (R2009) (Símbolos de Instrumentación y definición. Empleada para la elaboración del P&ID del proceso)
- Norma **ISO 9241** - parte 10, principios de diálogo (Recomendaciones y buenas prácticas tenidas en cuenta para el desarrollo del SCADA)
- [1] **Model based control of a yeast fermentation bioreactor using optimally designed artificial neural networks** - Chemical Engineering Journal 127 (2007) 95–109 (Zoltan Kalman Nagy Department of Chemical Engineering, Loughborough University, Loughborough, Leics LE11 3TU, United Kingdom)
- Guía de buenas prácticas en la elaboración de trabajos académicos (Normativa de Honestidad Académica de la ETSII)

1.5 Glosario de términos y abreviaturas

1.5.1 Definiciones relativas al control

- **Control continuo:** aplicación de técnicas y algoritmos de control para mantener una variable de proceso dentro de límites deseados en todo momento. En contraste con el control discreto, que opera en intervalos discretos de tiempo, el control continuo permite ajustes y correcciones constantes y suaves en respuesta a las condiciones cambiantes del proceso.
- **Linealización:** proceso de aproximación de un sistema no lineal a un modelo lineal en un punto de operación específico. Se utiliza para simplificar el análisis y el diseño de controladores, ya que los sistemas lineales son más fáciles de entender y manejar

matemáticamente. La linealización se basa en la expansión de Taylor y permite representar el comportamiento local de un sistema no lineal mediante un modelo lineal.

- **PID:** acrónimo de Proporcional, Integral y Derivativo, que son los términos que componen un controlador PID. Un controlador PID es un tipo de controlador de retroalimentación que calcula la salida de control en función del error actual, la suma acumulada de errores pasados y la tasa de cambio del error. El controlador PID es ampliamente utilizado en sistemas de control industrial debido a su simplicidad y capacidad para proporcionar un control preciso y estable.
- **Emparejamiento:** elección entre las variables manipuladas y las manipulables en un sistema de control. Se busca un rendimiento óptimo y reducir las influencias cruzadas entre las variables. El objetivo es asegurar que cada variable controlada pueda ser regulada de manera independiente y precisa.
- **Matriz de transferencia:** representación matemática de un sistema dinámico lineal. Es una matriz que relaciona las variables de entrada del sistema con las variables de salida, describiendo cómo las señales de entrada se transforman en las señales de salida a través de una combinación de operaciones matemáticas.
- **RGA (Relative Gain Array):** herramienta utilizada en el análisis de la interacción entre variables en un sistema de control multivariable. Proporciona información sobre la relación entre los cambios en las variables de entrada y las variables de salida del sistema. Deben emparejarse las entradas y salidas cuyo índice de la matriz se encuentre entre 0,5 y 10 para que predomine el efecto directo y sea más sencillo el control.
- **Niederlinski:** índice utilizado en control multibucle, que complementa a la RGA en la selección de emparejamientos. El criterio de Niederlinski (NI > 0 para plantas y controladores estables en bucle abierto –excepto acción integral en controlador–) complementa la información de ganancia relativa en plantas de dimensión 3 o mayor. Deberían ser elegidos emparejamientos que, tras permutación para estar en la diagonal, tengan tanto NI como los elementos diagonales de la RGA positivos. El criterio cambia para el caso de plantas inestables. Se define como:

$$\mathrm{NI} = \frac{\det\big(\mathrm{G}(0)\big)}{\det(\tilde{\mathrm{G}}\,(0))} = \frac{\det\big(\mathrm{G}(0)\big)}{\Pi_{i=1}^{m} g_{ii}(0)}$$

- **Método de Euler:** método numérico utilizado para aproximar soluciones de ecuaciones diferenciales. Es un enfoque simple y ampliamente utilizado que se basa en la aproximación de la derivada de una función mediante un cociente de diferencia finita. Se utiliza para discretizar las ecuaciones diferenciales y resolverlas de forma iterativa.
- **Modelo en variables de estado:** representación matemática de un sistema dinámico que utiliza un conjunto de ecuaciones diferenciales de primer orden, conocidas como ecuaciones de estado, para describir el comportamiento del sistema en función de sus variables internas, llamadas variables de estado.
- **Perturbación:** cambios no deseados o no controlados en las variables de entrada o condiciones del proceso que pueden afectar el rendimiento del sistema.

- **Realimentación:** principio fundamental en el control de sistemas donde se utiliza la información de salida para ajustar las entradas y mantener el sistema en un estado deseado. La realimentación permite la corrección continua de las desviaciones entre la salida deseada y la salida real del sistema, lo que permite un control más preciso y estable.
- **Override:** estrategia de control que sobrescribe el funcionamiento regular de control. Se utiliza para responder rápidamente a situaciones particulares o condiciones inusuales, permitiendo tomar el control directo de ciertas variables o funciones del sistema para garantizar una respuesta adecuada**.**
- **Histéresis:** condición en la cual se establece un rango de valores donde no se realiza ninguna acción de control hasta que el valor se salga de ese rango. Se utiliza para evitar cambios frecuentes en la acción de control y mejorar la estabilidad del sistema.

1.5.2 Definiciones relativas a componentes y comunicaciones

- **Modbus TCP/IP:** implementación del protocolo Modbus sobre TCP/IP (Protocolo de Control de Transmisión/Protocolo de Internet) en redes Ethernet. Permite la comunicación eficiente entre dispositivos en entornos industriales. Se utiliza para tareas como lectura de sensores, control de actuadores y monitoreo de variables en sistemas de automatización. Permite a un sistema, denominado servidor, enviar información a uno o varios sistemas, denominados clientes.
- **OPC UA:** (OLE for Process Control Unified Architecture) es un estándar de comunicación industrial utilizado para el intercambio seguro y confiable de datos en sistemas de automatización y control. Proporciona una arquitectura unificada para la interoperabilidad entre diferentes dispositivos y plataformas**.** OPC UA está basado en el modelo cliente/servidor y utiliza el protocolo TCP/IP para la comunicación. Permite el intercambio de datos en tiempo real, así como información de configuración, alarmas y eventos entre diferentes componentes de un sistema de automatización, como sensores, actuadores, controladores y sistemas de supervisión.

1.5.3 Otras definiciones

- **Software in the loop:** (SIL) técnica utilizada en el desarrollo y prueba de sistemas de control. Consiste en ejecutar el software de control en un entorno de simulación o emulación en lugar de utilizar el hardware físico real. Esto permite evaluar y depurar el software antes de implementarlo en el hardware, lo que ahorra tiempo y recursos. El SIL se utiliza comúnmente en aplicaciones de ingeniería y sistemas de control para verificar y validar el funcionamiento del software en un entorno virtual antes de la implementación en tiempo real.
- **SCADA:** (Supervisory Control And Data Acquisition)

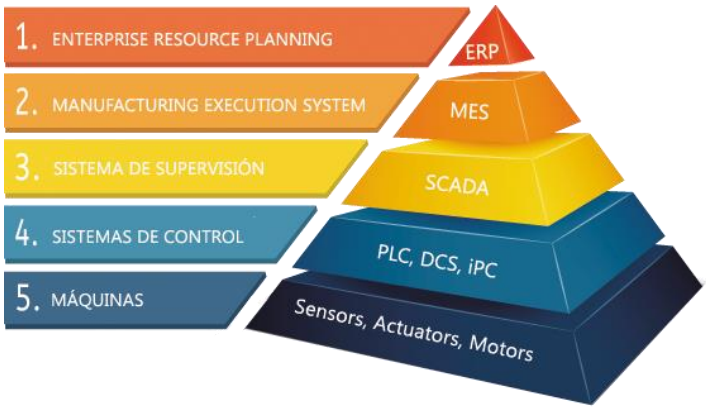

Figura 1 Pirámide de la automatización

1.6 Software utilizado

Durante el desarrollo de este proyecto, se emplearon varios programas y herramientas de software para diferentes etapas y tareas. A continuación, se proporciona una breve definición de cada uno de ellos:

- **Matlab:** Entorno de programación y desarrollo de software ampliamente utilizado en aplicaciones científicas y de ingeniería. Proporciona un amplio conjunto de funciones y herramientas para el análisis numérico, la simulación y la visualización de datos. En este proyecto, Matlab se utilizó para el desarrollo y ajuste del modelo de control, así como para el análisis de datos y la implementación del gemelo digital. Versión: R2022b, con licencia facilitada por la Universidad Politécnica de Valencia.
  Enlace: MATLAB - El lenguaje del cálculo técnico (mathworks.com)
- **Simulink:** Extensión de Matlab que permite la simulación y el modelado de sistemas dinámicos mediante diagramas de bloques. Es ampliamente utilizado para el diseño y la implementación de sistemas de control en tiempo real. En este proyecto, Simulink se utilizó para desarrollar y simular modelos del proceso y para diseñar y probar la estructura de control.
  Enlace: Simulación y diseño basado en modelos con Simulink - MATLAB (mathworks.com)
- **CODESYS:** Entorno de desarrollo integrado (IDE) utilizado para la programación de controladores programables lógicos (PLCs). Proporciona un conjunto de herramientas y una interfaz de programación estandarizada para la implementación de aplicaciones de automatización industrial. En este proyecto, CODESYS se utilizó para programar y configurar el PLC industrial donde se implementó la estructura de control. Se ha elegido este programa ya que es gratuito y compatible con el resto de aplicaciones. Versión: V3.5 SP18 Patch 4.
  Enlace: CODESYS Group
- **IGSS:** (Interactive Graphical SCADA System) sistema de desarrollo de SCADAs. Proporciona una interfaz gráfica de usuario para la supervisión y operación de

procesos, así como capacidades de registro de datos y generación de informes. En este proyecto, IGSS se utilizó para desarrollar la interfaz gráfica de usuario del sistema SCADA, permitiendo la monitorización y operación del proceso del biorreactor. Versión: IGSS V16, License FREE50

Enlace: IGSS | Industrial Automation and SCADA Systems since 1984 (schneider-electric.com)

- **Visio:** Herramienta de software utilizada para la creación de diagramas y visualizaciones técnicas. Proporciona una amplia gama de plantillas y herramientas para la creación de diagramas de flujo, diagramas de bloques y otros diagramas técnicos. En este proyecto, Visio se utilizó para la creación de diagramas P&ID que representan la estructura del sistema de control. Versión: Visio de Office365 mediante ordenador remoto a Polilabs.

  Enlace: Microsoft Visio - Diagramas de flujo - Visio online

Estos programas y herramientas de software desempeñaron un papel fundamental en el desarrollo, la simulación, la implementación y la documentación de este proyecto de control continuo sobre el biorreactor de etanol, proporcionando las capacidades necesarias para el diseño y la puesta en marcha del sistema de control industrial.

1.7 Descripción de la instalación

1.7.1 Modelo del biorreactor

El funcionamiento del biorreactor está definido por las siguientes ecuaciones diferenciales, extraídas del documento [1]:

Balances de masas

$$dV = Fi - Fe$$

$$dCx = NUx * Cx * Cs/(Ks + Cs) * exp(-Cp * Kp) - (Fe/V) * Cx$$

$$dCp = NUp * Cx * Cs/(Ks1 + Cs) * exp(-Cp * Kp1) - (Fe/V) * Cp$$

$$dCs = -(1/Rsx) * NUx * Cx * Cs/(Ks + Cs) * exp(-Cp * Kp) - (1/Rsp) * NUp * Cx * Cs/(Ks1 + Cs) * exp(-Cp * Kp1) + (Fi/V) * cS\_in - (Fe/V) * Cs$$

$$dCo2 = kla * (Co2A - Co2) - Ro2 - (Fe/V) * Co2$$

Balances de energía

$$dTr = (Fi/V) * (Tin + 273) - (Fe/V) * (Tr + 273) + ((Ro2 * Hr)/(32 * Ror * Cheat\_r)) - ((Kt * At * (Tr - Tag))/(V * Ror * Cheat\_r))$$

$$dTag = (Fag/Vj) * (Tin\_ag - Tag) + ((Kt * At * (Tr - Tag))/(Vj * Roag * Cheat\_ag))$$

1.7.2 Equipo utilizado

En el desarrollo de este proyecto, todas las actividades se llevaron a cabo en un mismo ordenador, utilizando el enfoque conocido como **"Software on the Loop" (SIL)**. Este enfoque implica realizar pruebas y simulaciones del sistema utilizando únicamente software, sin la necesidad de equipos físicos adicionales.

El equipo utilizado consistió en un ordenador con los requisitos de hardware necesarios para ejecutar las aplicaciones y herramientas de software utilizadas, como Matlab, Simulink, CODESYS, IGSS y Visio. El ordenador contaba con suficiente capacidad de procesamiento y memoria para llevar a cabo las tareas requeridas de simulación, programación, modelado y visualización.

El uso de un único ordenador en todo el proceso de desarrollo presenta ventajas significativas, como la facilidad de acceso a las herramientas de software y la posibilidad de realizar pruebas rápidas e iterativas. Además, el enfoque SIL permite reducir los costos asociados a la adquisición y configuración de equipos físicos adicionales.

1.8 Requerimientos

1.8.1 Requerimientos de control

- El control se realiza sobre un gemelo digital basado en el modelo no lineal del biorreactor [1].
- Se utiliza principalmente controladores PID debido a su aplicación sencilla.
- Las variables controladas deben estar cerca del punto de operación para lograr una respuesta rápida y sin grandes oscilaciones.
- Se debe tener en cuenta el rechazo de perturbaciones.
- El control se realiza en un entorno de simulación SIL utilizando el autómata simulado CODESYS.
- Se deben controlar la concentración de etanol, la temperatura del reactor y el volumen del tanque.

1.8.2 Requerimientos de los protocolos de comunicación

- La comunicación entre el biorreactor y el PLC se realiza mediante Modbus TCP/IP, utilizando variables WORD de 16 bits. Que simula la comunicación entre sensores/actuadores y PLC
- El cliente SCADA recibe la información del PLC a través de comunicación OPC UA.

1.8.3 Requerimientos del SCADA

- Mostrar los valores actuales de las variables del sistema.
- Permitir la modificación de los parámetros de control.
- Visualización de gráficas históricas.
- Visualización de la planta.
- Mostrar el diagrama P&ID.
- Estado de alarmas.

1.9 Análisis de soluciones

Este apartado consiste en un resumen de cómo se llegó a las soluciones para los requerimientos del proyecto. Se desarrollan en más detalle en la sección 2. Cálculos Justificativos e Implementación.

1.9.1 Solución relativa al control

Modelo del biorreactor y linealización

El control del biorreactor resultó ser complicado debido a su naturaleza no lineal y la fuerte interdependencia de las variables, especialmente las relacionadas con las concentraciones, que deben ser precisas. Se obtuvo un modelo basado en el documento [1], que define el proceso, aunque se realizaron modificaciones en ciertos parámetros y fórmulas que contenían errores, como en el caso de la ecuación que modela la variación de concentración de CO2. En la que se añade el término que corresponde al efecto del caudal de entrada, quedando:

$$\frac{dCo2}{dt} = kla * (Co2A - Co2) - Ro2 - (\boldsymbol{Fe/V}) * \boldsymbol{Co2}$$

Primero, se implementó este modelo no lineal en un script de Matlab. El tiempo de ejecución se ajustó para lograr un rendimiento óptimo en el dispositivo, considerando que tanto el proceso no lineal simulado, como el control y el SCADA se ejecuta en un mismo equipo.

El proceso tiene tiempos de respuesta muy lentos (horas). Por ello se ajusta el periodo para la simulación del modelo del biorreactor. Este tiempo de ejecución se ha establecido en Ts = 0.02 segundos, acelerando el proceso en comparación con el tiempo real para poder realizar las pruebas con los controladores diseñados sin tener que esperar horas para ver los cambios. Para obtener el tiempo real de proceso, se debe multiplicar Ts por 3600.

En cuanto a los puntos de operación del proceso (Tabla 1), se determinaron fijando las entradas en el punto de operación establecido en el documento [1] y observando gráficamente en Simulink cuándo se estabilizaban.

PUNTOS DE OPERACIÓN		
NOMBRE	DESCRIPCIÓN	VALOR
Fi	Caudal de entrada al reactor	51 l/h
Fag	Caudal del refrigerante	18 l/h
Tin	Temperatura de entrada al reactor	25 ºC
cS_in	Concentración de soluto de entrada al reactor	60 g/l

Tin_ag	Temperatura de entrada al refrigerante	15 ºC
Fe	Caudal de salida del reactor	51 l/h
V	Volumen del reactor	1000 l
Cx	Concentración de biomasa en la salida del reactor	0.9047 g/l
Cs	Concentración de soluto en la salida del reactor	29.7389 g/l
Cp	Concentración de soluto en la salida del reactor	12.5152 g/l
Co2	Concentración de dióxido de carbono en la salida del reactor	3.1069 mg/l
Tr	Temperatura en el interior del reactor	29.57 ºC
Tag	Temperatura en la salida del refrigerante	27.05 ºC

Tabla 1. Puntos de operación

Una vez planteado el modelo no lineal del sistema, el siguiente paso consiste en realizar la linealización de este. Para llevar a cabo esta linealización, se utilizó inicialmente la herramienta ***Linearize*** de Simulink. Esta herramienta, que toma los puntos de operación definidos, genera un conjunto de matrices en el espacio de estado (A, B, C y D) que representa la aproximación lineal del sistema.

Sin embargo, durante el proceso de linealización, se encontró que esta matriz presentaba un alto índice de la variable compleja 's', en torno a la potencia 260. Debido a esto y para que los modelos resulten útiles para el diseño de los controladores, se ha optado por simplificar los modelos obteniendo la dinámica más significativa. Se aplicó un método individual a cada elemento de la matriz del espacio de estado. De esta manera, se transformó cada elemento en una función de transferencia y dando como resultado al conjunto como una matriz de 7 filas y 6 columnas, correspondientes a las salidas y entradas del sistema.

La matriz de transferencia GG presenta el siguiente aspecto:

Figura 2. Matriz de transferencia GG

Posteriormente, se realizó una simplificación de estas funciones de transferencia, eliminando polos y ceros que tuvieran un impacto poco significativo en la dinámica. Un ejemplo sería la siguiente función que relaciona Cx con Fi:

$$GG(2,1) = \frac{0.000112\ s\text{^}2\ +\ 1.104e-05\ s\ +\ 2.168e-08}{0.1109\ s\text{^}4\ +\ 0.01801\ s\text{^}3\ +\ 0.001102\ s\text{^}2\ +\ 2.217e-05\ s}$$

Con esto, se obtuvo el modelo linealizado, el cual presenta pequeñas diferencias con el modelo real para una variación de +0.5 en las variables de entrada (Figura 3 y Figura 4). Por lo tanto, se considera un resultado aceptable de cara a trabajar con este modelo en la parte de control.

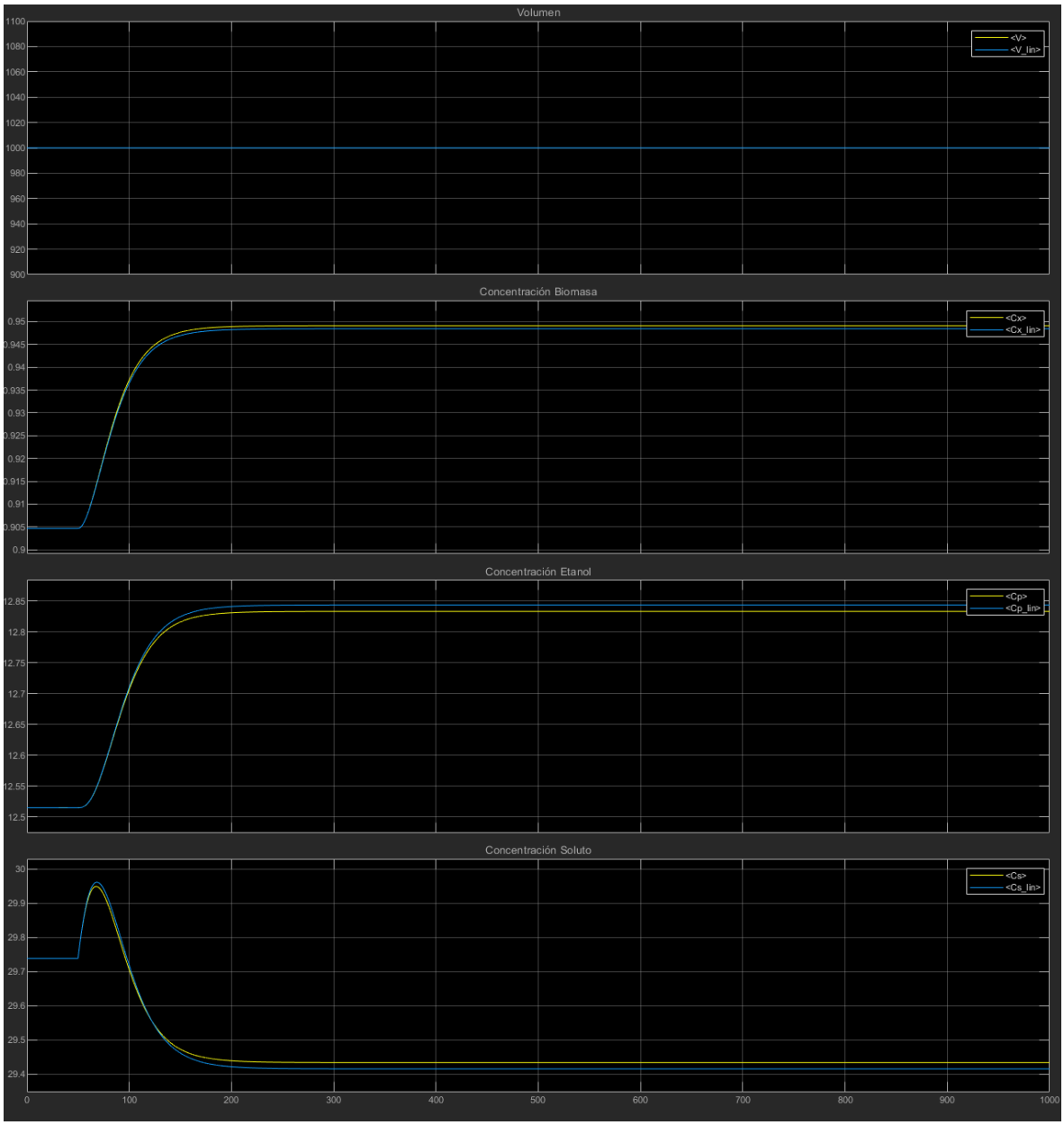

Figura 3. Gráficas en Simulink del modelo no lineal frente al modelo linealizado 1

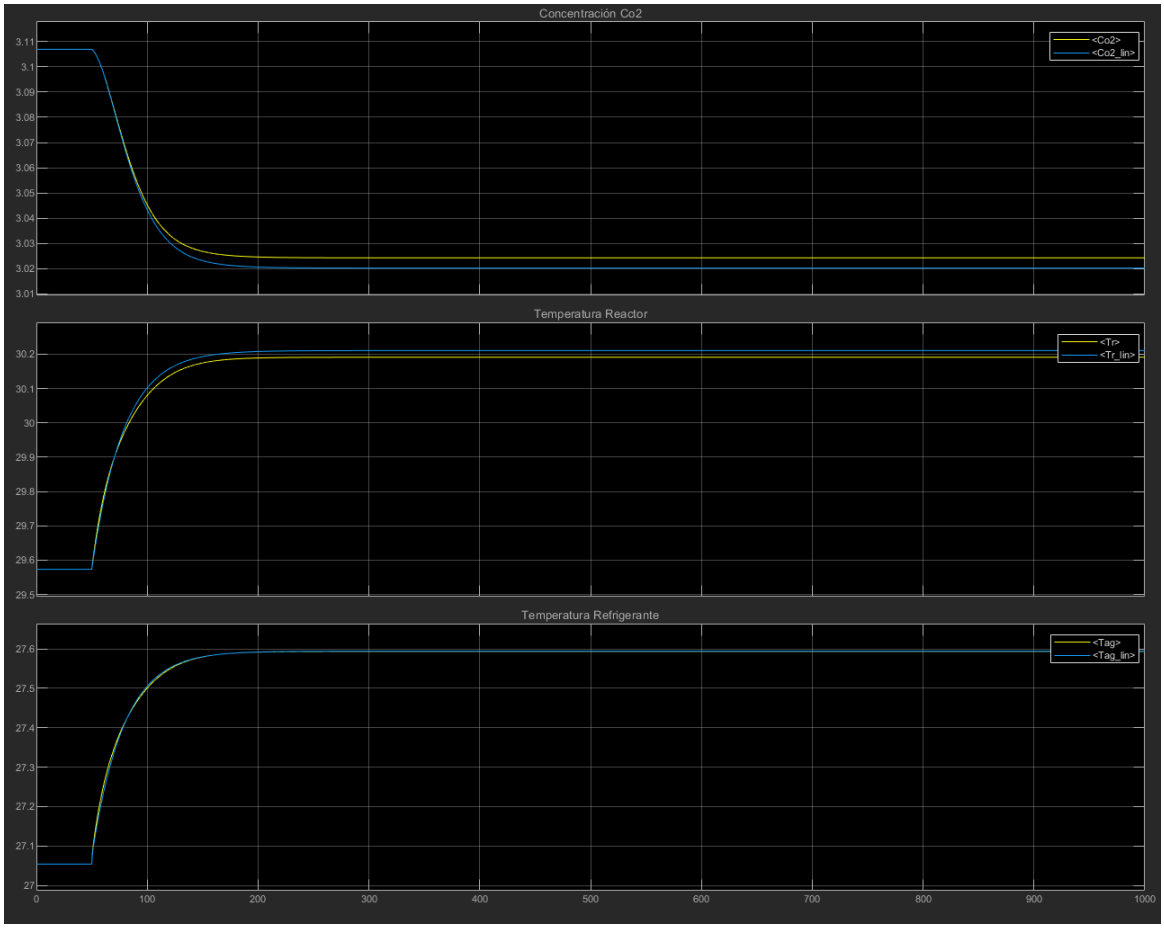

Figura 4. Gráficas en Simulink del modelo no lineal frente al modelo linealizado 2

Emparejamientos

A partir de estas funciones de transferencia simplificadas, es momento de pensar qué se desea controlar realmente y qué emparejamientos son los más idóneos para ello.

En la primera aproximación realizada y siguiendo lo que se ha visto en la asignatura de Instrumentación y Control Industrial, se estudiaron los emparejamientos utilizando el método RGA y el índice de Niederlinski. Para ello, se realizó el siguiente procedimiento en código de Matlab.

Lo primero es crear la matriz de ganancias K a partir de la matriz de transferencia GG:

```
%% Cargar la matriz de transferencia GG
load matriz_transferencia.mat
%% Cálculo de matriz de ganancias K
% se deben quitar los integradores (y luego se
% vuelven a poner, la primera y sexta columna tiene integradores)
% ganancias de las fdts que no tienen integradores (filas 1-7, columnas 2-5)
K=dcgain(GG(1:7,2:5));
% ganancias de las fdts que tienen integradores (filas 1-7, col 1 y 6)
s=tf('s');
K2=dcgain(GG(1:7,[1,6])*s);
% como hay integradores, para obtener la matriz completa se sustituye 1/s
```

```
% por la variable simbolica I (utiliza el Symbolic Toolbox de Matlab)
syms I
K2=K2*I;
% en notación simbólica salen fracciones, se escribe con decimales con vpa()
K2=vpa(K2,4); %4 decimales
% se completa toda la matriz
K=[K2(:,1) K K2(:,2)];
% ver la matriz entera
K=vpa(K,4)
```

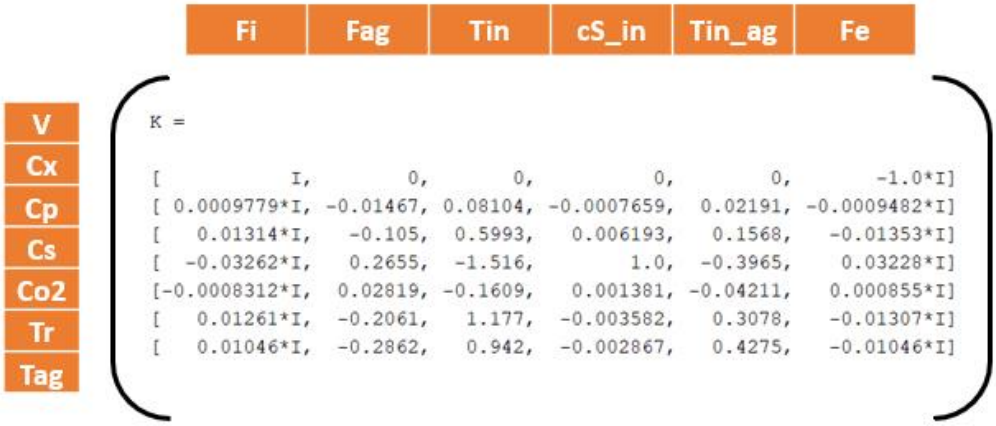

Figura 5. Matriz de ganancias K

Se observa que esta matriz K no es cuadrada y no se ve claro qué variables se pueden quitar de cara a poder estudiarla por RGA. Se realizó un análisis del funcionamiento del reactor y se tomaron las siguientes consideraciones, teniendo en cuenta que la matriz K presenta las mismas filas y columnas que la matriz de transferencia GG de la Figura 2:

- El caudal de entrada (Fi) se supuso como una variable a monitorizar y una perturbación, por lo que se eliminó la primera columna.
- Para el control del volumen, como se deduce la matriz de ganancias, solo le afecta Fi y Fe. Como ya Fi se ha considerado una perturbación, solo queda el caudal de salida (Fe) para el control de V. Además, se implementó un control PID ratio con el caudal de entrada (Fi). Estas consideraciones llevan a eliminar la sexta columna.
- La temperatura de entrada al reactor Tin también se considera una perturbación (se puede usar en alguna prealimentación), se elimina la tercera columna.
- Si se supone la temperatura del refrigerante (Tin_ag) constante (o una perturbación, utilizable en alguna prealimentación) se puede eliminar columna cinco.
- Solo quedarían las columnas dos (Fag) y cuatro (Cs_in) para el control.

Por lo tanto, las variables que se pueden controlar son 6 (todas las filas menos la primera, porque la V se ha emparejado ya) y puesto que solo quedan 2 variables manipulables, (Fag y Cs_in) se debe elegir entre estos dos posibles emparejamientos de control. Parece razonable querer controlar la concentración de producto Cp (fila 3) y la temperatura del reactor Tr (fila 6) puesto que influye mucho en la producción.

Con estas consideraciones, se reordena la matriz de ganancias y se calcula la RGA y el Niederlinski:

```
KK=K([3,6],[2,4]);
RGA=vpa(KK.*transpose(inv(KK)),2); al ser simbólico
RGA=double(RGA); %para pasar de simbólica a matriz numérica
KK_ordenada(1,:)=KK(2,:);
KK_ordenada(2,:)=KK(1,:);
Niederlinski=det(KK_ordenada)/prod(diag(KK_ordenada))
```

$$RGA = \begin{pmatrix} 0.2276 & 0.7724 \\ 0.7724 & 0.2276 \end{pmatrix}$$

$$Ni = 1.2946$$

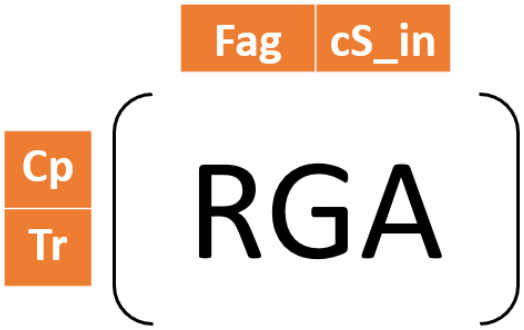

Figura 6. Matriz RGA

Como se observa, la RGA muestra que el emparejamiento óptimo sería Cp-cS_in y Tr-Fag.

Sin embargo, los resultados de control no fueron satisfactorios con estos emparejamientos, debido a que se tardaba un tiempo demasiado largo en llegar al set point de control establecido.

Se decidió entonces utilizar el caudal del refrigerante (Fag) como variable manipulada principal para controlar la concentración de etanol (Cp), ya que, de forma lógica, la parte de refrigeración debería de ser un mecanismo de control. El control del volumen (V) resultó ser sencillo debido a su relación con los caudales de entrada (Fi) y salida (Fe), que actúan como un integrador. Se eligió tomar como variable manipulada el caudal de salida para mantener el volumen del reactor en el punto de operación. En cuanto a la temperatura del reactor (Tr), se intentó controlarla simultáneamente a las otras. Pero se observó una relación clara entre la temperatura y las demás variables, lo que dificultaba el control simultáneo de la concentración de etanol (Cp) y la temperatura del reactor (Tr). Por lo tanto, se implementó un control override, que se activa cuando la temperatura supera el valor de 32.57 ºC y la devuelve al punto de operación. Para evitar cambios constantes en este control, se utilizó una condición de histéresis.

Toda esta parte de control se realizó primero en Simulink, con el objetivo de poder ir tuneando sobre la marcha los diferentes parámetros y configuraciones para obtener el control deseado.

Cuando estas pruebas dieron un buen resultado, se procedió a replicarlo en CODESYS e instalarlo en el PLC simulado.

Implementación en dispositivo industrial (PLC CODESYS)

El análisis de soluciones para la implementación del diseño de control realizado en Simulink en CODESYS tuvo en cuenta los siguientes aspectos:

- El objetivo principal era transferir el diseño de control desarrollado en Simulink a la plataforma CODESYS. Esto implicaba traducir el modelo de control en un programa con lenguaje Ladder y ST, de manera que se pudiera ejecutar en un controlador lógico programable (PLC). Esto se ha realizado en dos POUs, para tener separado el cálculo realizado en ST y los bloques de control en Ladder.
- Durante el proceso de implementación, se encontraron dificultades en el escalado de las señales. Las señales provenientes del sistema de control desarrollado en Simulink necesitaban ser ajustadas adecuadamente, ya que al enviarse estas variables por comunicación MODBUS TCP/IP, se deben convertir a tipo WORD. Para poder mantener el signo y los decimales de cada señal, se ha realizado un escalado que permita transportar el valor más preciso posible. Se debe tener en cuenta que el rango de valores representables en un entero de 16 bits va desde -32,768 hasta 32,767. Se realizaron ajustes y conversiones de escala para garantizar una correcta representación de las variables de control en CODESYS. Dentro de la programación, se usa **LREAL_TO_WORLD** (Figura 7) para enviar el valor digitalizado y **WORD_TO_REAL** para traducir el enviado desde Matlab.
-

```
//ENTRADAS DEL SISTEMA
    FIO.Fi_dig :=  LREAL_TO_WORD(Fi*(30000.0/200.0));
    FIO.Fag_dig :=  LREAL_TO_WORD(Fag*(30000.0/200.0));
    FIO.Tin_dig :=  LREAL_TO_WORD((30000.0/255.0)*(Tin+55.0));
    FIO.cS_in_dig :=  LREAL_TO_WORD(cS_in*(30000.0/100.0));
    FIO.Tin_ag_dig :=  LREAL_TO_WORD((30000.0/255.0)*(Tin_ag+55.0));
    FIO.Fe_dig :=  LREAL_TO_WORD(Fe*(30000.0/200.0));
```

Figura 7. Cambio de variables de tipo LREAL a WORD

- Para la implementación de los controladores PID en CODESYS, facilitados por el **DISA** de la **UPV**, se tuvo que importar manualmente la librería ***SysTimeGetMs()*** que permitiera acceder y utilizar las funciones relacionadas con el tiempo de la CPU.
- Durante la programación en CODESYS, se encontró un problema relacionado con la detección de números decimales. Para solucionarlo, fue necesario agregar ".0" a los valores numéricos para que el programa reconociera correctamente la presencia de decimales. De esta manera, se evitó que las operaciones aritméticas devolvieran valores erróneos o nulos.
- Para la implementación del override, se realizó con dos condiciones que hace referencia a los rangos de histéresis (Figura 8).

```
// Implementación de la histeresis
IF Tr > (referencia + hysteresis1) THEN
    override := TRUE;
ELSIF Tr < (referencia + hysteresis2) THEN
    override := FALSE;
END_IF
```

Figura 8. Override en CODESYS

1.9.2 Solución relativa al SCADA

Durante el proceso de desarrollo del sistema SCADA, se realizaron múltiples revisiones y ajustes en el diseño de la pantalla principal. El principal objetivo era crear una interfaz práctica, accesible y de fácil comprensión para los operarios, de manera que pudieran interactuar de manera rápida y sencilla con el sistema. Con este objetivo en mente, se llevaron a cabo varios cambios de diseño para lograr una pantalla principal que cumpliera con estos requisitos.

En el diseño de la pantalla principal, se consideraron factores como la disposición de los elementos, el tamaño y la legibilidad de los textos, la organización de la información y la usabilidad general. Se buscó simplificar y optimizar la visualización de los valores del biorreactor y las variables relevantes, así como la sección de ingreso de valores de control y la indicación del estado de las alarmas.

Además, parte del proceso de desarrollo del SCADA implicó la creación de un menú que permite acceder a pestañas adicionales. Estas pestañas incluyen una descripción de cómo utilizar el SCADA y un diagrama P&ID de la planta. Estas adiciones fueron pensadas para proporcionar información adicional y ayudar a los operarios a comprender mejor el funcionamiento del sistema.

Sin embargo, se presentó un problema significativo al establecer las alarmas, ya que esto causaba que el programa se cerrara inesperadamente o se produjera un fallo total del sistema. Ante esta situación, se contactó al servicio de soporte técnico de IGSS, el proveedor del SCADA utilizado, para buscar una solución.

Tras comunicar el problema al servicio de soporte, se determinó que el fallo estaba relacionado con una interacción entre el SCADA y la comunicación OPC UA. Se identificó un error en el software proporcionado por el proveedor, lo que ocasionaba el cierre del programa al intentar establecer las alarmas. Para abordar esta situación, se trabajó en colaboración con el servicio de soporte de IGSS, brindando información detallada sobre el problema y los pasos seguidos para reproducirlo.

Después de una semana, el servicio de soporte de IGSS lanzó una actualización del software que solucionó el problema con la configuración de las alarmas. La actualización corrigió la interacción defectuosa entre el SCADA y la comunicación OPC UA, lo que permitió establecer y gestionar las alarmas de manera adecuada sin provocar cierres inesperados del programa.

1.10 Resultados finales

1.10.1 Hardware empleado

El dispositivo empleado, que contiene la planta, el PLC y el SCADA ejecutándose todo simultáneamente, es un OMEN HP Laptop 16-c0xxx. Dispone de un procesador AMD Ryzen 7 5800H de 3.20 GHz y una RAM de 16 GB. El motivo de esta elección es principalmente que ya se disponía de este componente, el cual permite un uso fluido de todo el sistema.

1.10.2 Software in the Loop

Matlab

Durante el desarrollo del proyecto, se ha utilizado MATLAB como la herramienta principal para llevar a cabo el montaje del gemelo digital del biorreactor y diseñar el sistema de control asociado.

El desarrollo del sistema de control se lleva a cabo mediante un script principal que consta de un bucle. Al ejecutarse, este script carga los parámetros iniciales y los puntos de operación del sistema. Los puntos de operación se muestran en pantalla para su visualización. A continuación, el script ejecuta el modelo del biorreactor cada 0.02 segundos, lo que permite realizar un seguimiento continuo del comportamiento del sistema en tiempo real.

Además, se utiliza otro script independiente para establecer la comunicación MODBUS TCP/IP con el PLC. Este script se encarga de establecer la comunicación entre MATLAB y el PLC, lo cual es fundamental para la interacción y control del sistema. Se escalan las variables y se realizan operaciones de escritura y lectura de los datos. Esta comunicación y actualización de variables se realiza cada 0.5 segundos, asegurando una sincronización adecuada entre el biorreactor y el PLC.

Por último, con el objetivo de simular un entorno lo más realista posible, se ejecuta un script adicional que introduce perturbaciones en las variables principales del sistema. Estas perturbaciones pueden afectar tanto a las variables de entrada como a los sensores, permitiendo evaluar el rendimiento y la respuesta del sistema de control ante condiciones variables.

En cuanto a los emparejamientos utilizados en el sistema de control, se han establecido los siguientes:

- **GG (1,6):** control del **Volumen** (V) manipulando el **Caudal de salida** (Fe)
- **GG (3,2):** se controla la **Concentración de etanol** (Cp) con el **Caudal del refrigerante** (Fag)
- **GG (6,2):** se controla la **Temperatura del reactor** (Tr) haciendo un **override** y con el **Caudal del refrigerante**

Estos emparejamientos permiten establecer estrategias de control específicas para cada variable y asegurar un funcionamiento óptimo del biorreactor en términos de volumen, concentración de etanol y temperatura.

A continuación, se presentan los diagramas finales del esquema de control implementado y los resultados gráficos del control de los emparejamientos mencionados.

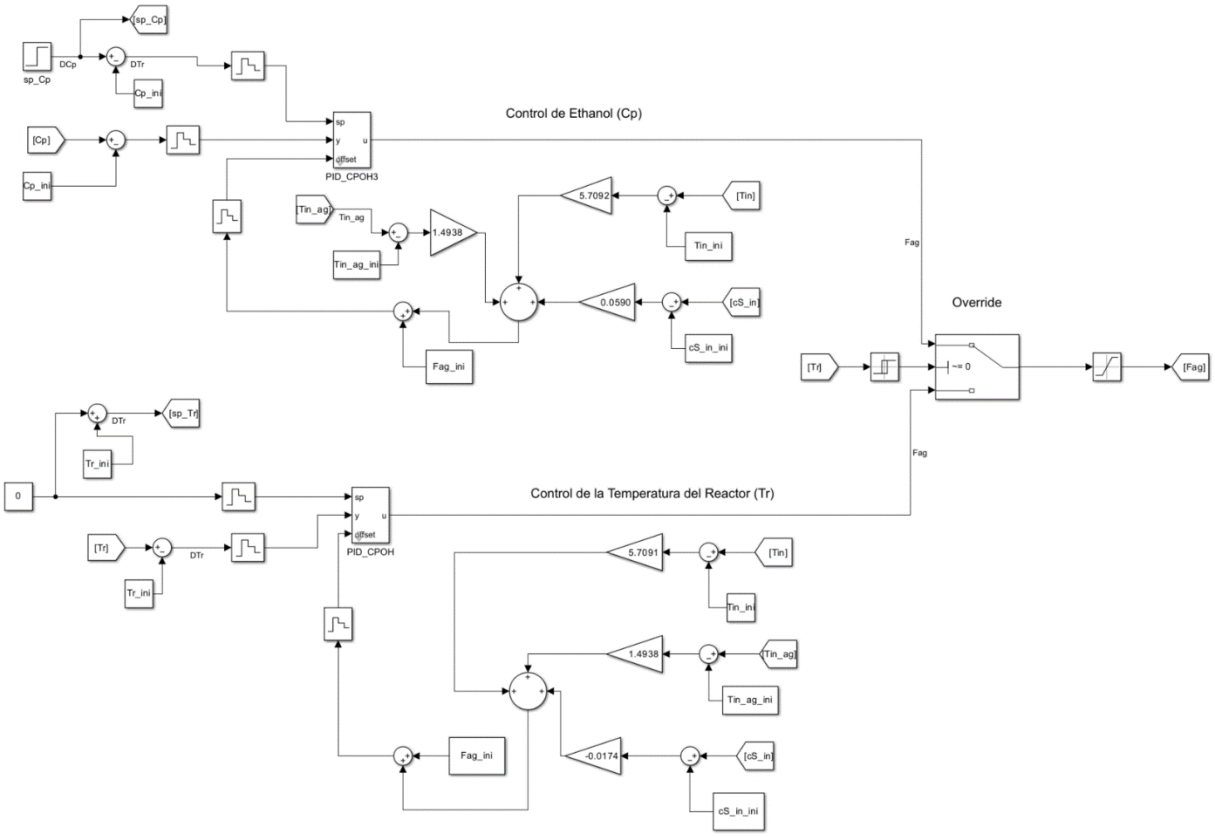

Figura 9. Bucle de control Cp y Tr sobre modelo no lineal

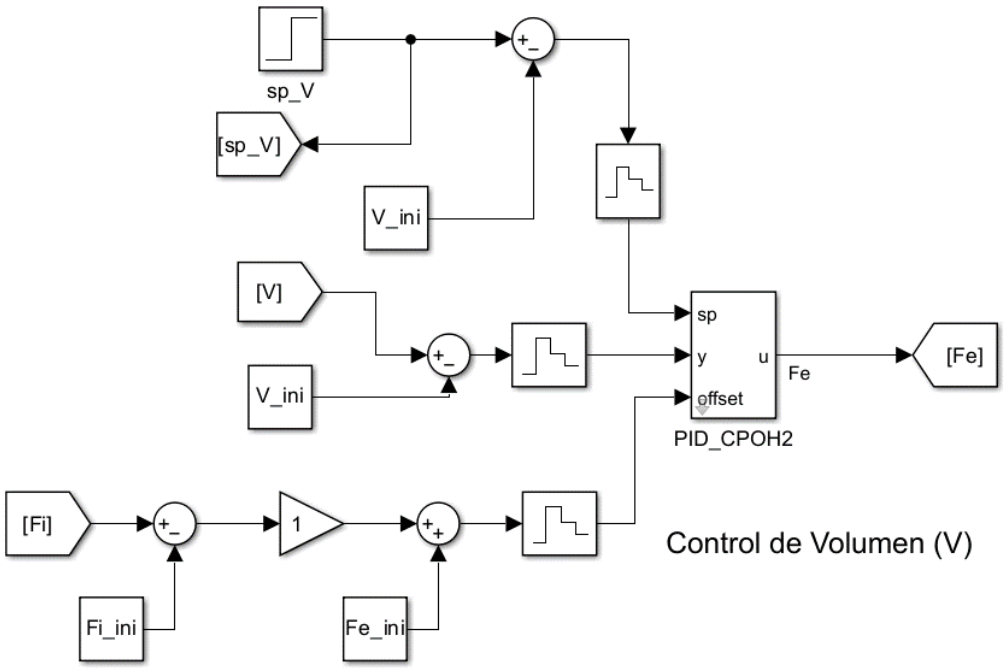

Figura 10. Bucle de control V sobre modelo no lineal

El diagrama del esquema de control implementado muestra la interconexión de los diferentes bloques y componentes que forman parte del sistema de control. En este diagrama, se representan los lazos de retroalimentación y los diferentes puntos de control y señales de entrada y salida. Se muestra claramente la secuencia y relación entre los bloques, brindando una representación visual del flujo de control en el sistema.

Se presentan también los resultados gráficos obtenidos del control de los emparejamientos mencionados (Figura 11 y Figura 12). Estos resultados muestran las respuestas del sistema de control en términos de las variables controladas y manipuladas. Las gráficas permiten analizar el desempeño del control en términos de estabilidad, seguimiento de referencia y rechazo a perturbaciones.

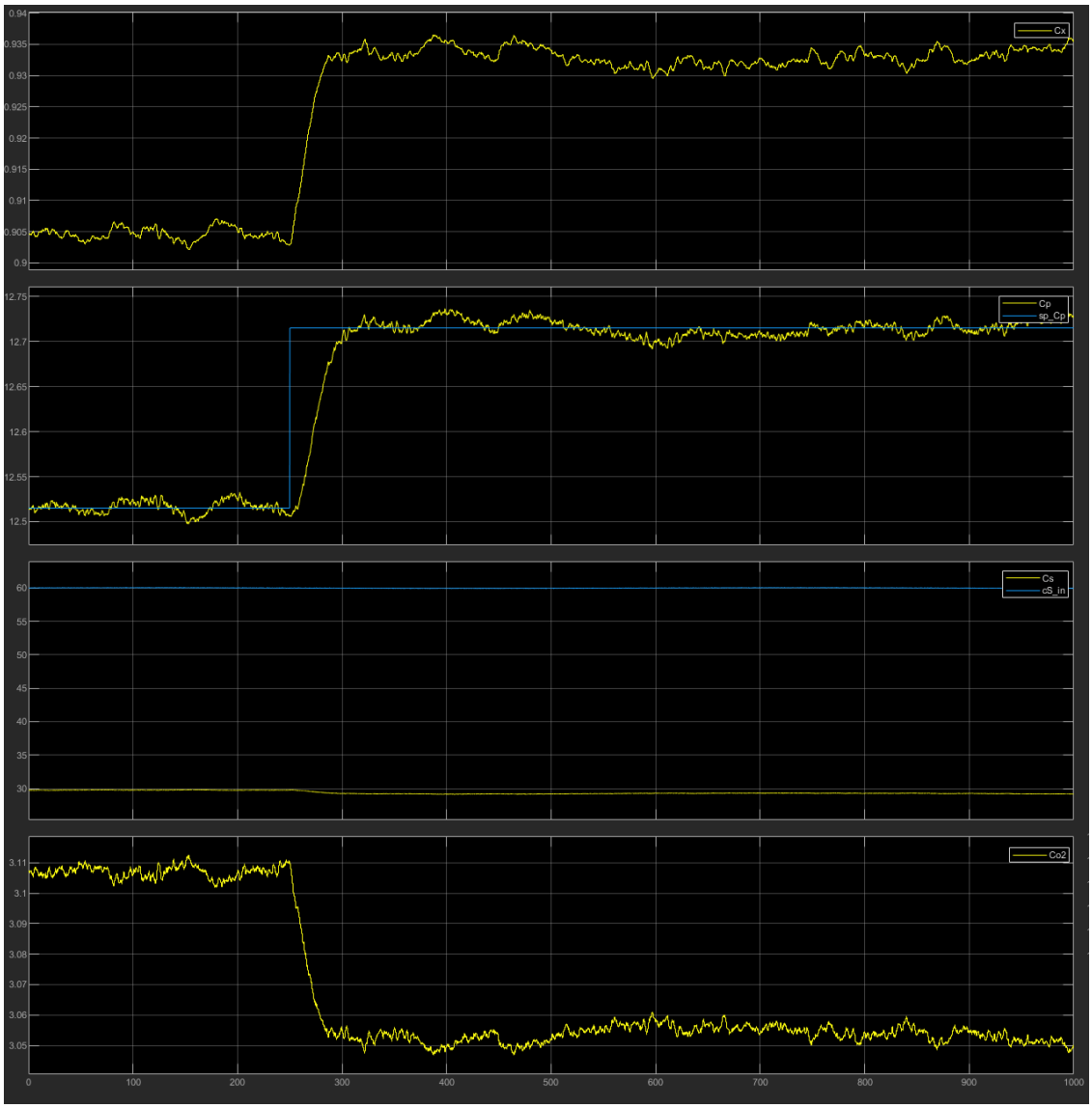

Figura 11. Gráficos de respuesta no lineal 1

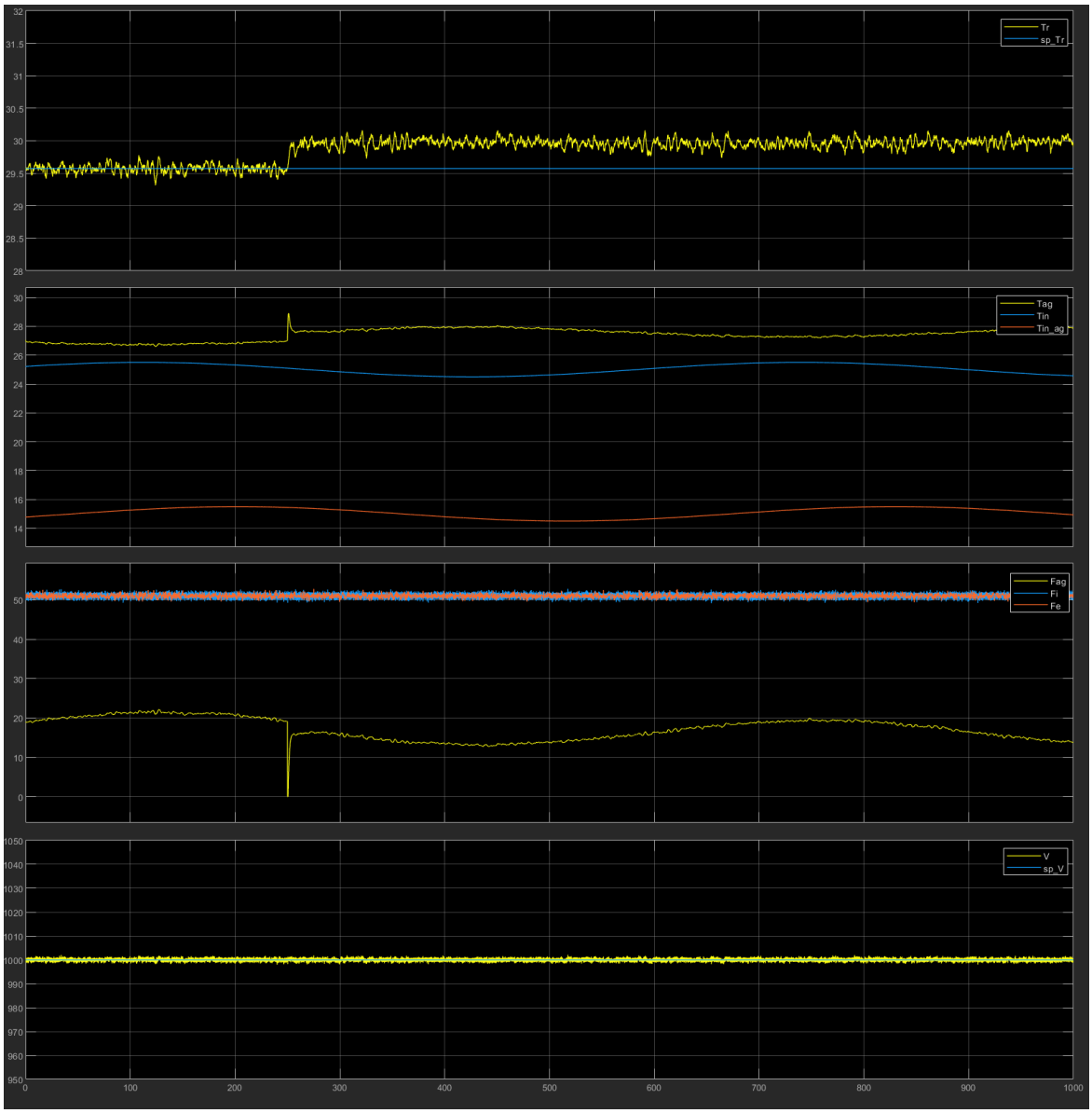

Figura 12. Gráficos de respuesta no lineal 2

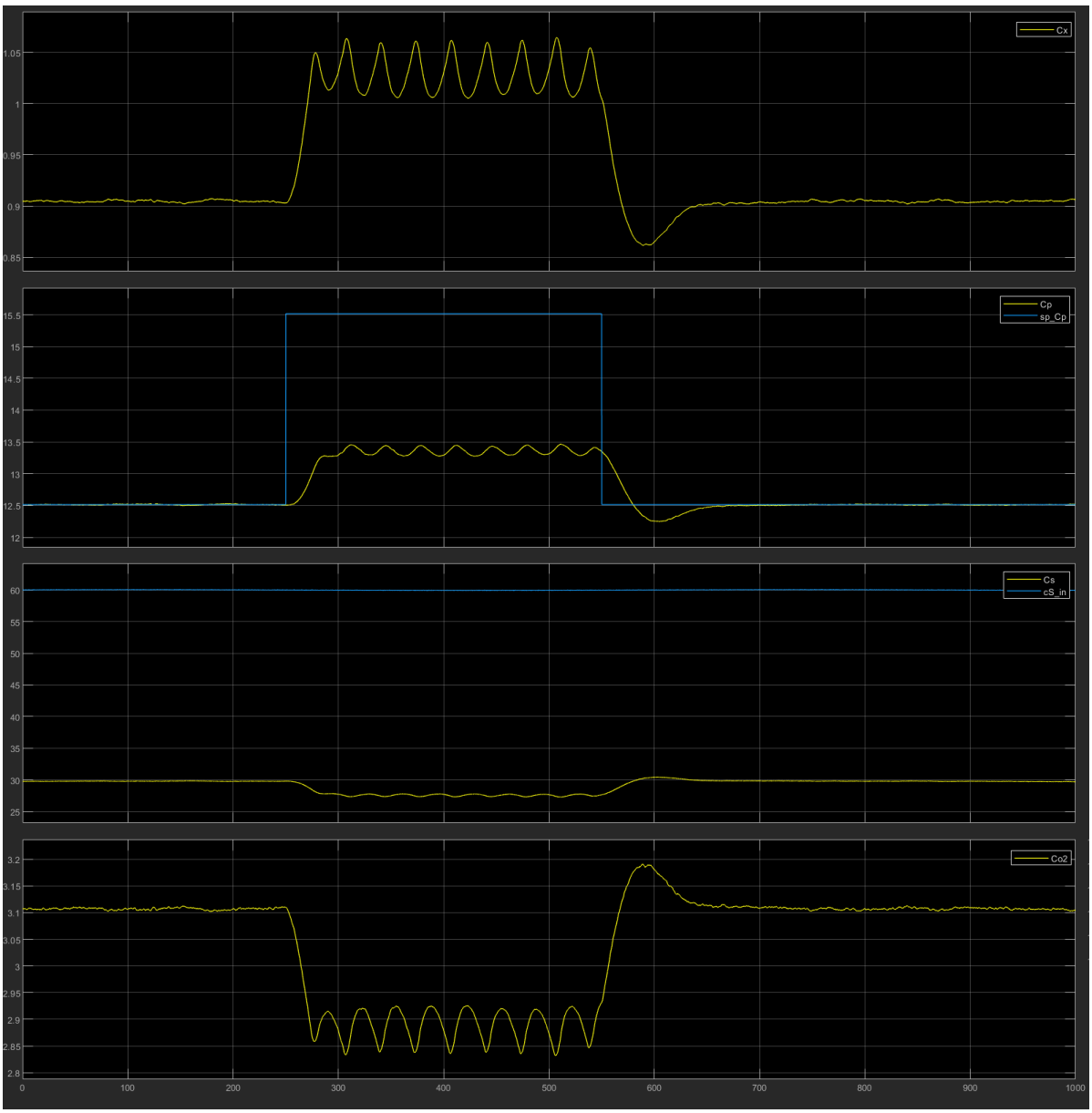

Figura 13. Gráficas control override 1

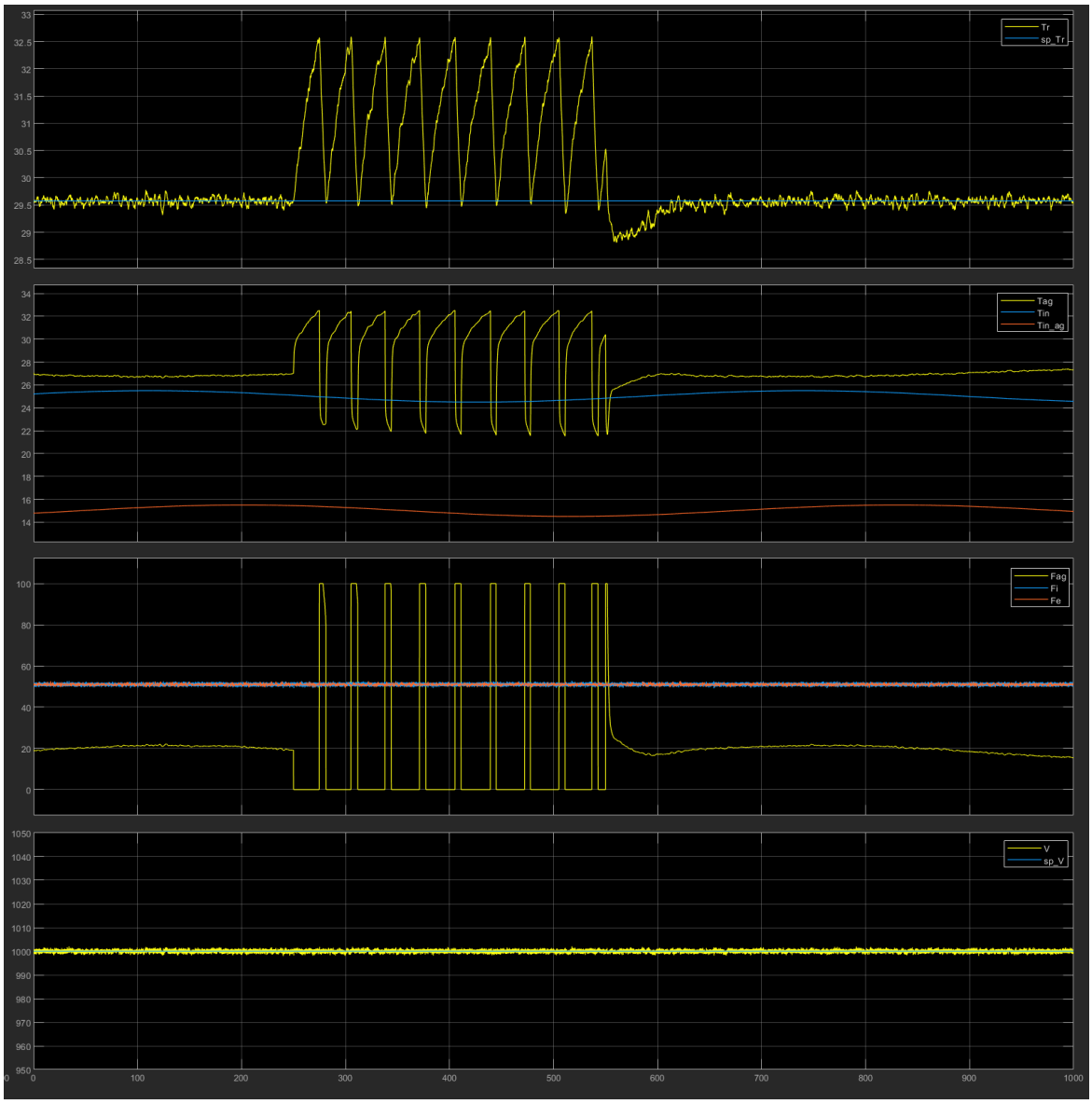

Figura 14. Gráficas control override 2

PLC

La implementación del control anteriormente diseñado y su consecuente programación se ha realizado en CODESYS y en el PLC simulado que ofrece este software.

Se han llevado a cabo varias acciones y configuraciones para implementar el sistema de control diseñado previamente en Simulink y permitir la lectura y escritura de variables mediante la comunicación MODBUS TCP/IP. A continuación, se describen los pasos realizados en CODESYS:

- **Agregar el dispositivo PLC simulado:** Se agregó el dispositivo que simula el PLC. Se eligió la opción "CODESYS Control Win V3 x64" y se configuró la IP como "localhost" y el hardware como "OMEN-PC".
- **Adición de los Program Organization Units (POUs):** Se agregaron dos POUs en la sección ***Application*** del proyecto en CODESYS. El primero es ***PLC_PRG (PRG)***, que se

encarga de realizar el escalado de las señales, inicializar valores, calcular los offsets e implementar la histéresis utilizando el lenguaje ST (AWL). El segundo POU es ***POU (PRG)***, que alberga los bloques de los controladores PID y está programado en lenguaje Ladder (KOP).

- **Configuración de la ejecución cíclica de los POUs:** Se indicó en CODESYS que los POUs se ejecuten de forma cíclica dentro de la sección ***MainTask***. También se estableció el valor del tiempo en el que se realizará cada ciclo de ejecución (0,2 segundos).
- **Creación de un Function Block (FB):** Se creó un Function Block llamado ***POU_2*** con el código que conforma los controladores PID. Este bloque fue proporcionado por el DISA de la UPV, al igual que el bloque de PID en Simulink.
- **Configuración de la comunicación MODBUS TCP/IP:** Se agregó un dispositivo de tipo Ethernet en CODESYS y se incluyó el componente ***ModbusTCP Slave Device***. En la configuración del dispositivo, se realizaron ajustes como la asignación del puerto esclavo (502), el número de registros holdings (7) y el número de registros inputs (6) (Figura 15). También se creó un bloque de ***Lista de variables globales*** para el envío de los datos en formato digital.
- **Configuración de la comunicación OPC UA:** Se agregaron los módulos necesarios para establecer la comunicación OPC UA con el SCADA.
- **Adición de gráficas:** Como una funcionalidad adicional, se agregaron gráficas en CODESYS para verificar el correcto funcionamiento del control (Figura 16, Figura 17 y Figura 18).

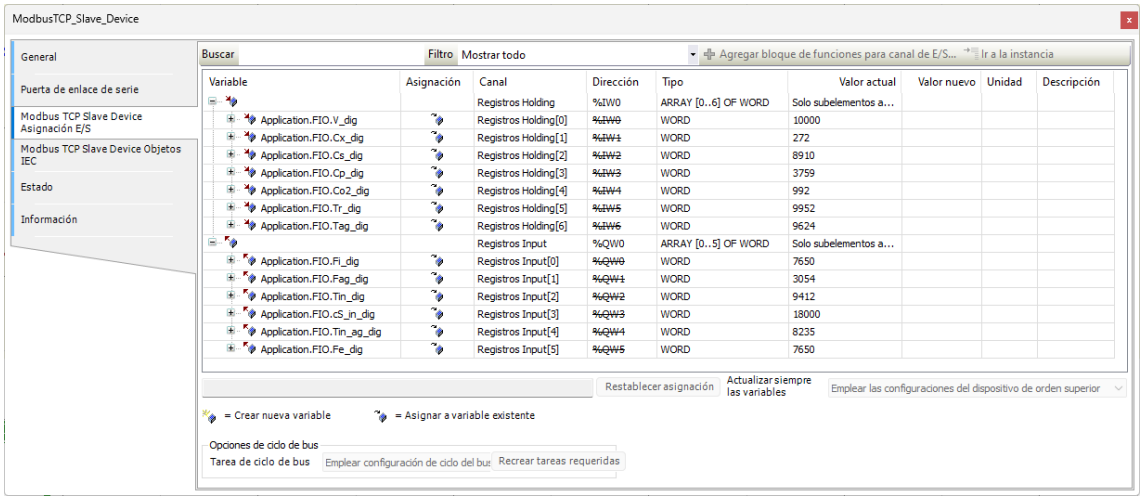

Figura 15. Lista de variables enviadas y recibidas por Modbus TCP/IP

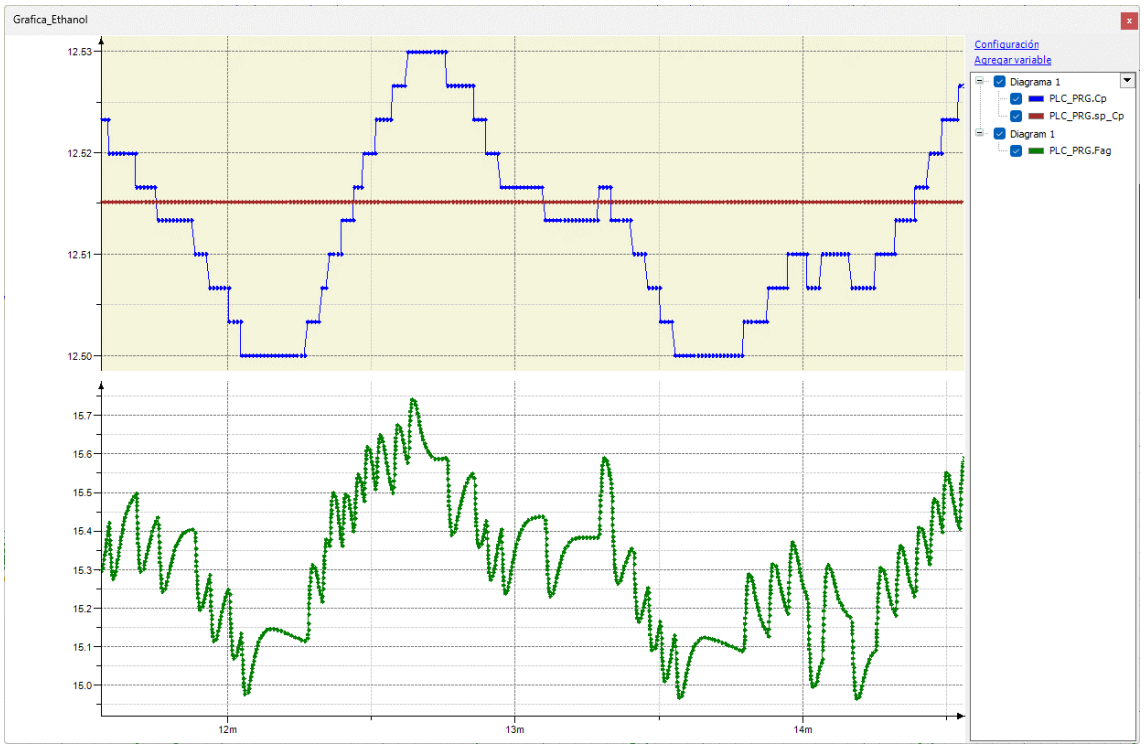

Figura 16. Gráfica control de etanol con perturbaciones en el entorno de CODESYS

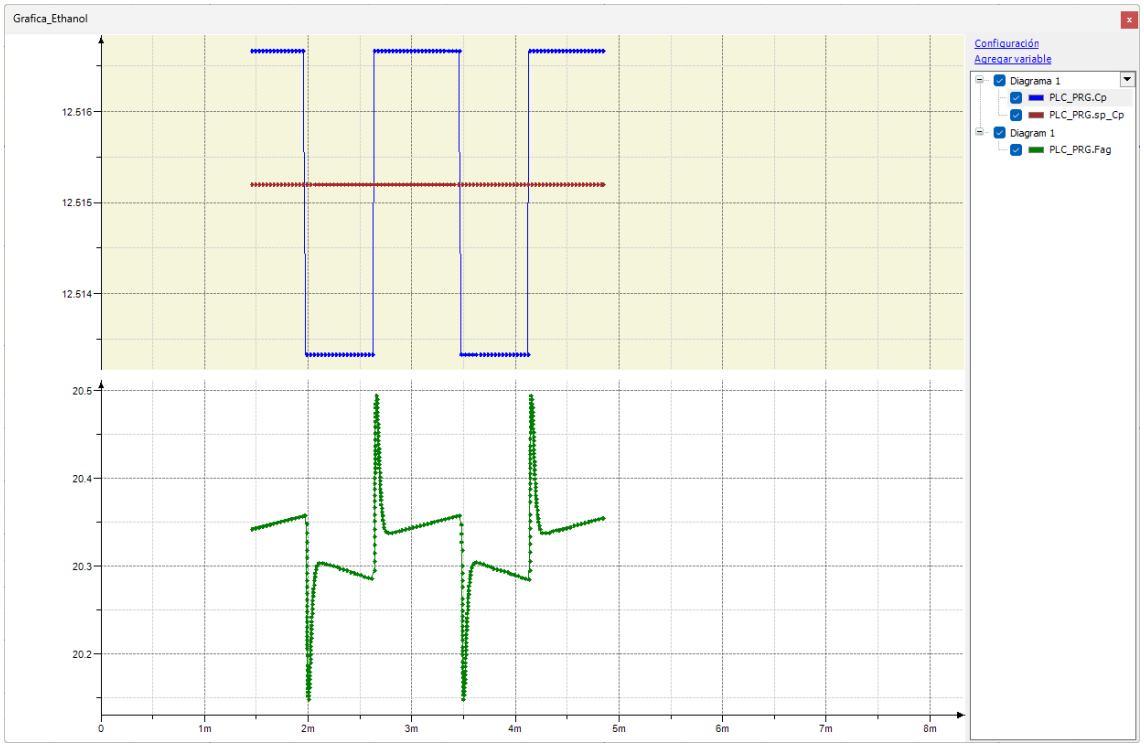

Figura 17. Gráfica control de volumen sin perturbaciones en el entorno de CODESYS

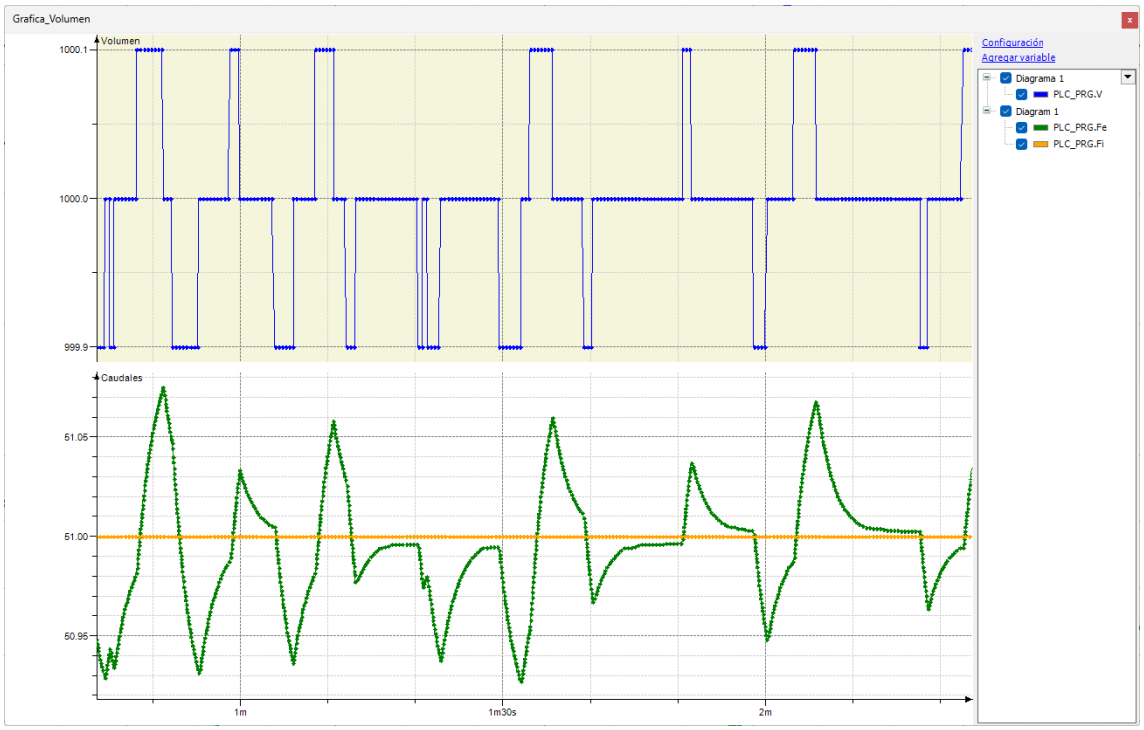

Figura 18. Gráfica control de volumen con perturbaciones en el entorno de CODESYS

SCADA

Para este proyecto se ha desarrollado el SCADA en el software IGSS, mediante la implementación de pantallas personalizadas para la supervisión y control de procesos. Estas pantallas incluyen elementos como gráficos, tablas, alarmas y controles interactivos, diseñados para visualizar y gestionar variables y parámetros en tiempo real.

Se ha configurado el sistema de alarmas en IGSS para detectar condiciones anormales y generar notificaciones en pantalla. Esto permite llevar un control en tiempo real del proceso, tomando medidas rápidas y efectivas en caso de errores.

Además, se ha implementado el control de los dispositivos y actuadores, permitiendo ajustar y modificar parámetros de forma remota para una gestión eficiente del sistema.

La capacidad de registro y almacenamiento de datos históricos de IGSS se ha utilizado para analizar tendencias, generar informes y realizar un seguimiento detallado del rendimiento del sistema a lo largo del tiempo.

La interfaz de usuario en IGSS ha sido personalizada para organizar las pantallas, gráficos y elementos de control según las necesidades del proyecto, facilitando la navegación y el acceso a la información relevante.

A continuación, se muestran las pantallas de la interfaz SCADA:

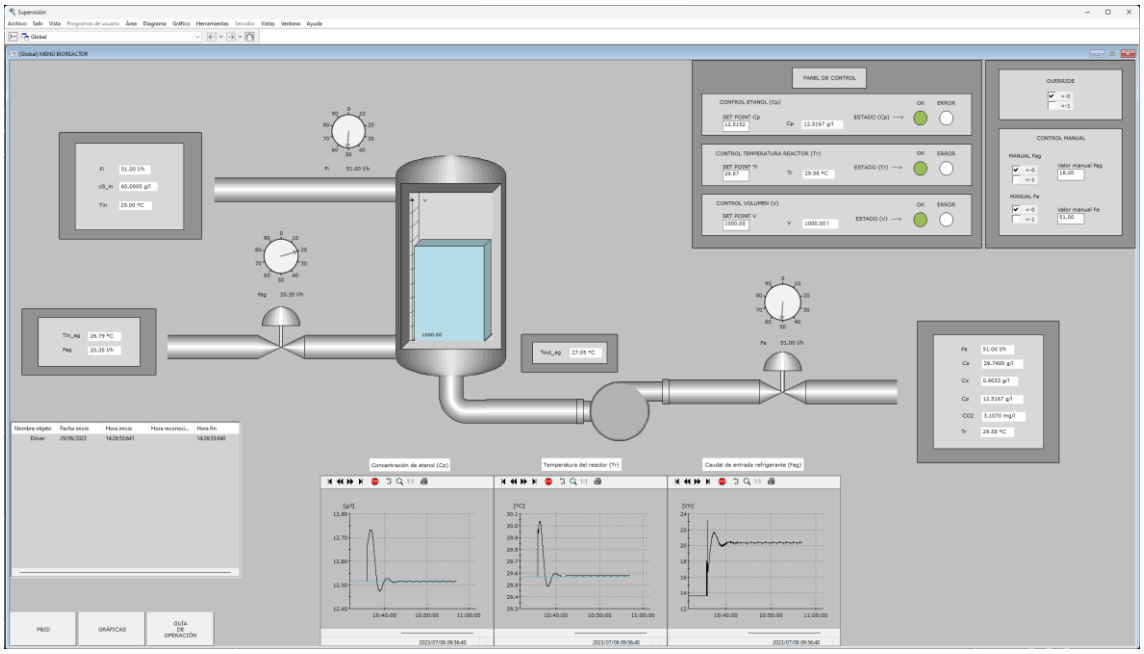

Figura 19. Pantalla principal SCADA en IGSS

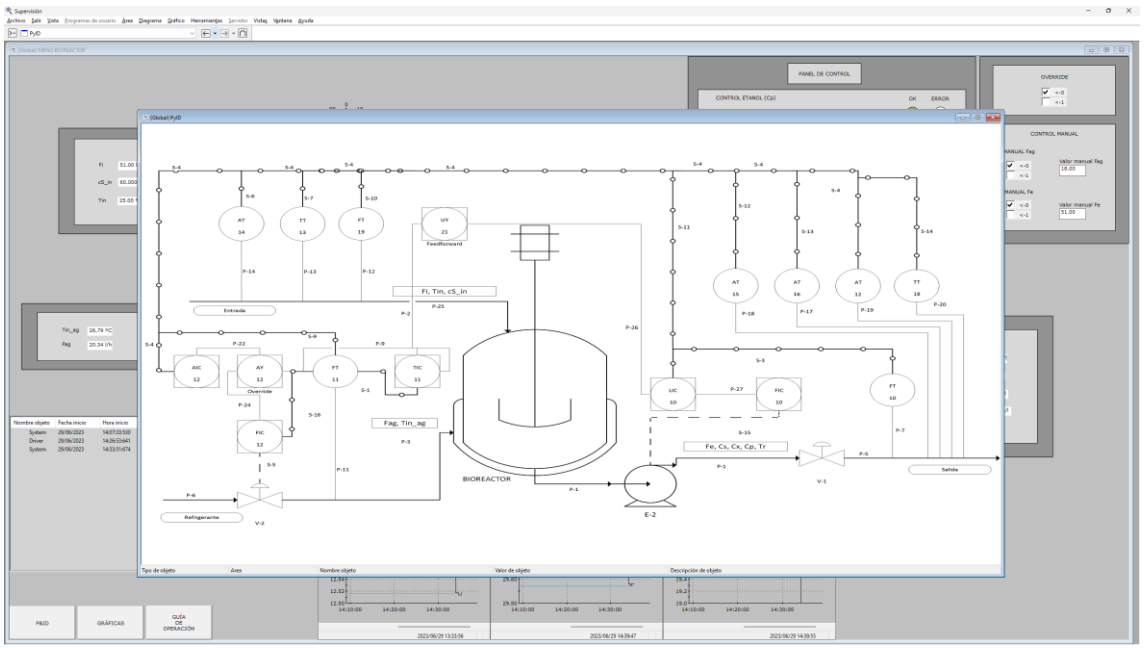

Figura 20. Ventana P&ID en IGSS

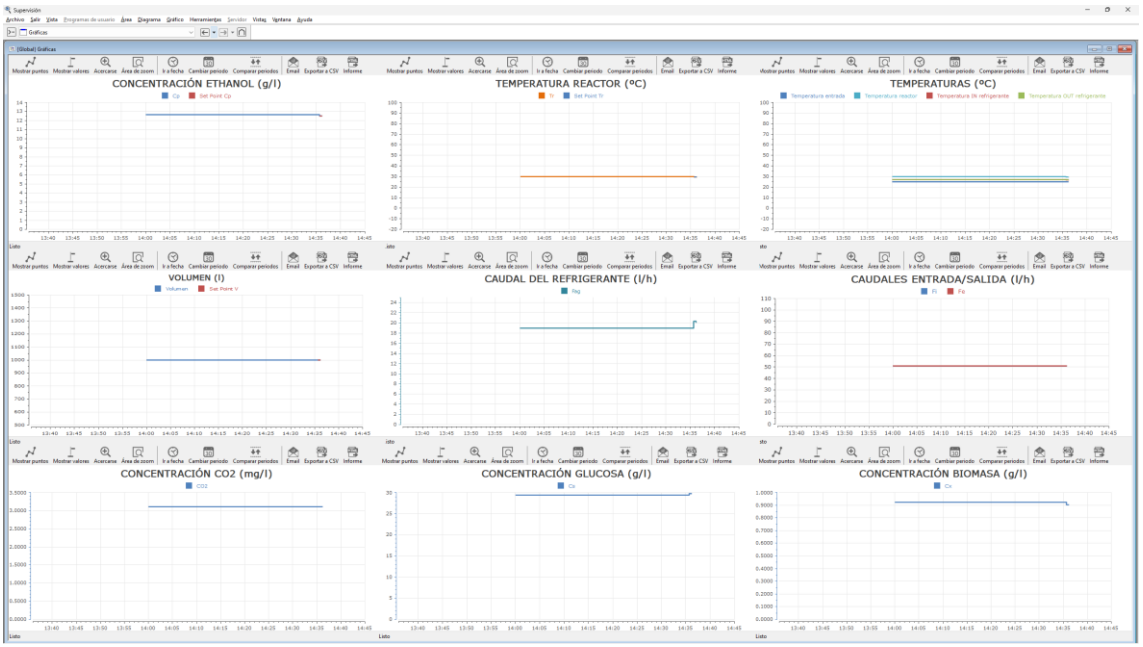

Figura 21. Ventana de gráficas históricas en IGSS

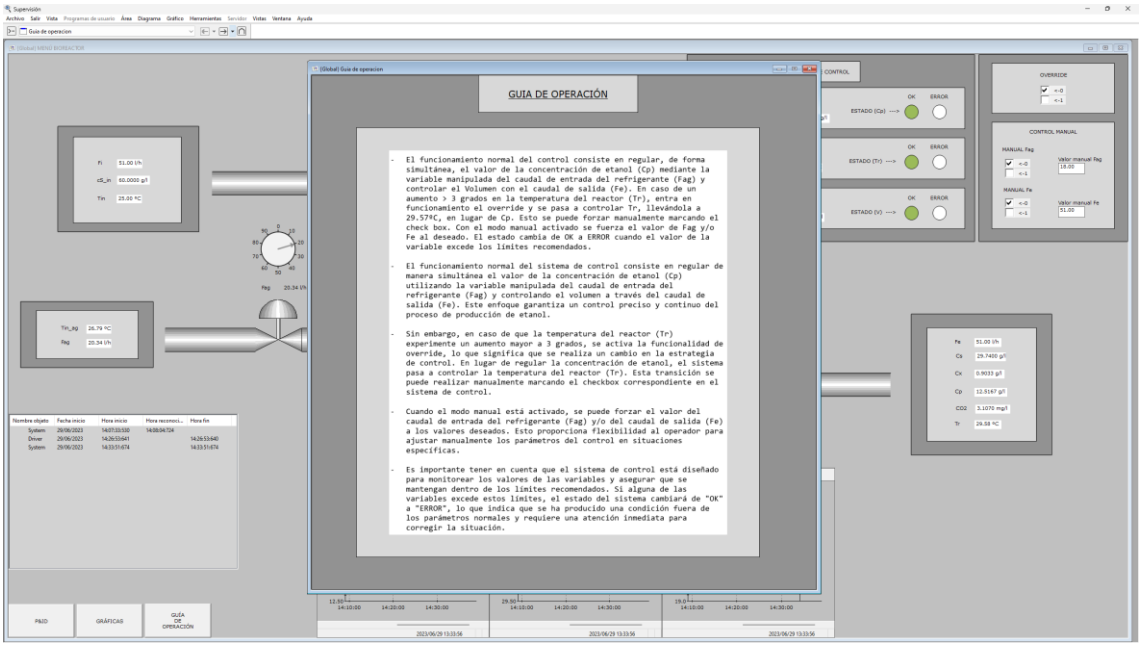

Figura 22. Ventana guía de operación de la planta en IGSS

1.10.3 Esquema final

El esquema final de la planta consta de varios componentes interconectados:

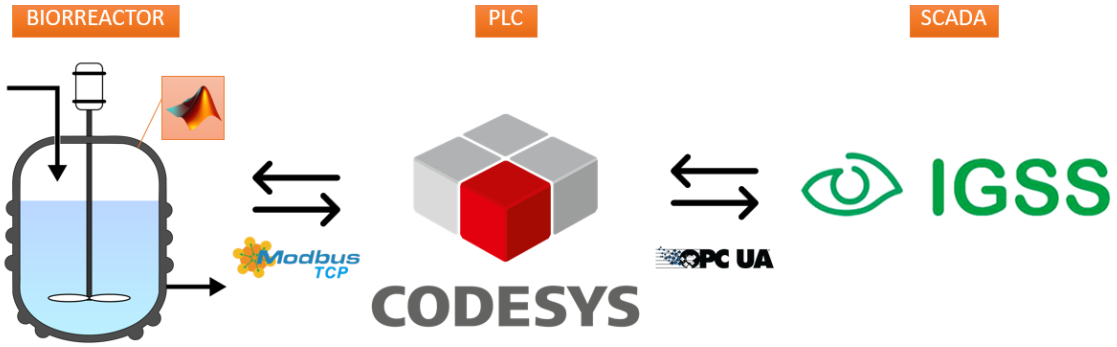

Figura 23. Software in the Loop (SIL)

En primer lugar, se encuentra el gemelo digital del biorreactor, ejecutado en Matlab. Modela su funcionamiento en tiempo real. Este sistema adquiere y procesa diferentes variables, como la temperatura, las concentraciones y el nivel. A partir de estos datos, se aplican algoritmos y estrategias de control para ajustar los parámetros del proceso y optimizar su funcionamiento.

Las variables procesadas y calculadas en el gemelo digital se envían a través de comunicación MODBUS TCP/IP al PLC simulado. El PLC simulado, que se ejecuta en CODESYS, actúa como el controlador del sistema. Recibe las variables del biorreactor y las utiliza para supervisar los sensores de la planta y controlar los dispositivos actuadores.

A su vez, el PLC simulado se comunica con el SCADA a través de una conexión OPC UA. El SCADA, implementado en IGSS, recibe los datos del PLC simulado y los muestra en tiempo real en su interfaz de usuario. El SCADA permite supervisar y controlar el funcionamiento de la planta mediante pantallas personalizadas, gráficos, alarmas y controles interactivos. También proporciona capacidades de registro y almacenamiento de datos históricos, lo que permite el análisis de tendencias y la generación de informes.

Para poder replicar todo este proceso, es necesario seguir de forma secuencial los siguientes pasos:

1. **Arrancar el PLC simulado:** Inicia la instancia del PLC simulado en CODESYS utilizando el CODESYS Control Win SysTray. Hacer clic derecho en la instancia y seleccionar "Start PLC" para que comience a ejecutarse
2. **Reset en caliente del PLC:** Se debe realizar un reset en caliente en CODESYS del PLC para reiniciar el valor de las variables iniciales del proceso
3. **Ejecutar el script yeast_reactor_RUN:** Inicia el script de run en MATLAB, el cual se encarga de inicializar y configurar el gemelo digital del biorreactor
4. **Ejecutar el script yeast_reactor_MODBUS:** Inicia el script de Modbus en MATLAB, el cual se encarga de establecer la comunicación MODBUS TCP/IP con el PLC simulado. Este script se encarga de enviar las variables procesadas por el gemelo digital al PLC para su control y supervisión
5. **Ejecutar el script yeast_reactor_PERT (opcional):** Inicia el script que genera perturbaciones aleatorias en el sistema.
6. **Arrancar IGSS y modo Supervisión:** Inicia el software IGSS y permite visualizar las pantallas diseñadas en el SCADA y obtener información en tiempo real sobre el funcionamiento de la planta. Se puede navegar a través de las diferentes pantallas para supervisar y controlar los parámetros y variables relevantes del proceso

Escuela Técnica Superior de Ingeniería Industrial

ETSII

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Máster Universitario en Ingeniería Industrial (Acceso desde Grado I. Electrónica Industrial y Automática)

Desarrollo e implementación de un sistema de control industrial de un biorreactor para la producción de etanol

Trabajo Fin de Máster

2. CÁLCULOS JUSTIFICATIVOS E IMPLEMENTACIÓN

Autor: Daniel Pérez Gutiérrez

Tutor académico: Francesc Xavier Blasco Ferragud

Julio 2023

CAPÍTULO 2. CÁLCULOS JUSTIFICATIVOS E IMPLEMENTACIÓN

2.1 Implementación del gemelo digital (Matlab)

En este apartado se va a desarrollar toda la programación realizada para la ejecución del biorreactor y posterior parte de diseño del sistema de control.

2.1.1 Gemelo digital del biorreactor

El primer paso fue llevar a Matlab todas las ecuaciones del documento [1], que define el modelo no lineal del biorreactor mediante parámetros y ecuaciones diferenciales. Esto se realiza mediante los siguientes tres scripts:

- **yeast_raector_params:** Carga los parámetros necesarios para los cálculos posteriores.
- **yeast_reactor_ValoresIni:** Carga el valor inicial de las variables de estado y de las entradas al sistema, además del tiempo de ejecución Ts. Muestra por pantalla estos valores (Figura 24).

```
---------------VALORES INICIALES---------------
Tiempo: 0.00000 [s]
Variables manipuladas
  Fi: 51.00 [l/h]; Fag:  18.00 [l/h]; Fe:  51.00 [l/h]
  Tin: 25.00 [oC]; Tin_ag:  15.00 [oC]
  Cs_in: 60.000 [g/l]
Variables estado:
  V: 1000.0 [l]
  Cx: 0.905 [g/l]; Cp: 12.515 [g/l]; Cs: 29.739 [g/l]; Co2: 3.107 [mg/l]
  Tr: 29.57 [oC]; Tag: 27.05 [oC]
```

Figura 24. Valores Iniciales en la Command Window

- **yeast_reactor_modelo:** Constituye el script principal donde se realiza todo cálculo de las derivadas. Este código, a partir de las 6 entradas del sistema, resuelve el valor para 7 variables de salida haciendo uso del método de Euler, que se realimentan de nuevo en el cálculo como variables de estado. Como, por ejemplo:

$$V = dV * Ts + V$$

  Se ajustan también las condiciones de saturación:
  - El volumen del tanque no puede ser superior a 1500l ni inferior a 10l.
  - El valor de las concentraciones no puede ser inferior a 0.

- **yeast_reactor_run:** Es el programa que se encarga de llamar a todos los demás y ejecuta un timer para llamar de forma repetida a yeast_reactor_modelo, calculando así de forma continua las variables de estado (Figura 25).

```
%% RUN REACTOR
yeast_reactor_params;
yeast_reactor_ValoresIni;
%% TIMER EJECUCIÓN
tt=timer;
tt.StartDelay=0;
tt.Period=Ts;
tt.TimerFcn="yeast_reactor_modelo";
tt.ExecutionMode='fixedRate';
start(tt)
```

Figura 25. Script yeast_reactor_run

2.1.2 Comunicación MODBUS TCP/IP

La comunicación entre Matlab y el PLC se realiza mediante comunicación MODBUS TCP/IP. En el lado de Matlab, esto lo conforman los siguientes dos scripts:

- yeast_reactor_escalado: en este script se realiza la escritura y lectura de las variables comunicadas:

$$write(mm,'holdingregs',1,[V_{dig}Cx_{dig}Cs_{dig}Cp_{dig}Co2_{dig}Tr_{dig}Tag_{dig}],2,'uint16');$$
$$vector\_read = read(mm,'inputregs',1,6,2,'int16');$$

  El envío de datos se hace con enteros de 16 bits y el PLC devuelve un tipo WORD, es por esto por lo que se debe de realizar un correcto escalado a digital de las señales y así mantener el mayor número de decimales con una alta precisión. Los escalados realizados son los siguientes:
  - Concentraciones [0 100] -> [0 30000]
  - Temperaturas [-55 200] -> [0 30000]
  - Caudales [0 200] -> [0 30000]
  - Volumen [0 3000] -> [0 30000]
- yeast_reactor_MODBUS: este código es el segundo timer de Matlab. Realiza la apertura de la comunicación modbus en la **IP local 192.168.1.58.** Crea el bucle que llama a yeast_reactor_escalado (Figura 26).

```
%% APERTURA DE CONEXIÓN MODBUS A CODESYS
mm = modbus('tcpip','192.168.1.58',502);
%% TIMER EJECUCIÓN MODBUS
tt2=timer;
tt2.StartDelay=0;
tt2.ExecutionMode='fixedRate';
tt2.Period=0.5;
tt2.TimerFcn="yeast_reactor_escalado";
start(tt2)
```

Figura 26. Script yeast_reactor_MODBUS

2.2 Diseño del sistema de control (Matlab/Simulink)

Ya con el biorreactor funcionando, es momento de diseñar la parte de control mediante el módulo de Simulink, el cual necesita de los siguientes módulos para realizar todo el desarrollo:

- **Control System Toolbox** versión 10.12
- **Industrial Communication Toolbox** versión 6.1
- **Simulink Control Design** versión 6.2
- **Symbolic Math Toolbox** versión 9.2

Ya dentro de Simulink, se configura el ***Stop Time*** entre 300 y 1000, tiempo con el que se puede observar bien la evolución del proceso. Accediendo a ***Configuration Parameters*** se termina de ajustar la herramienta (Figura 27), destacando la solución del ***Solver*** en tipo ***Variable-step***, al tratarse de un proceso continuo.

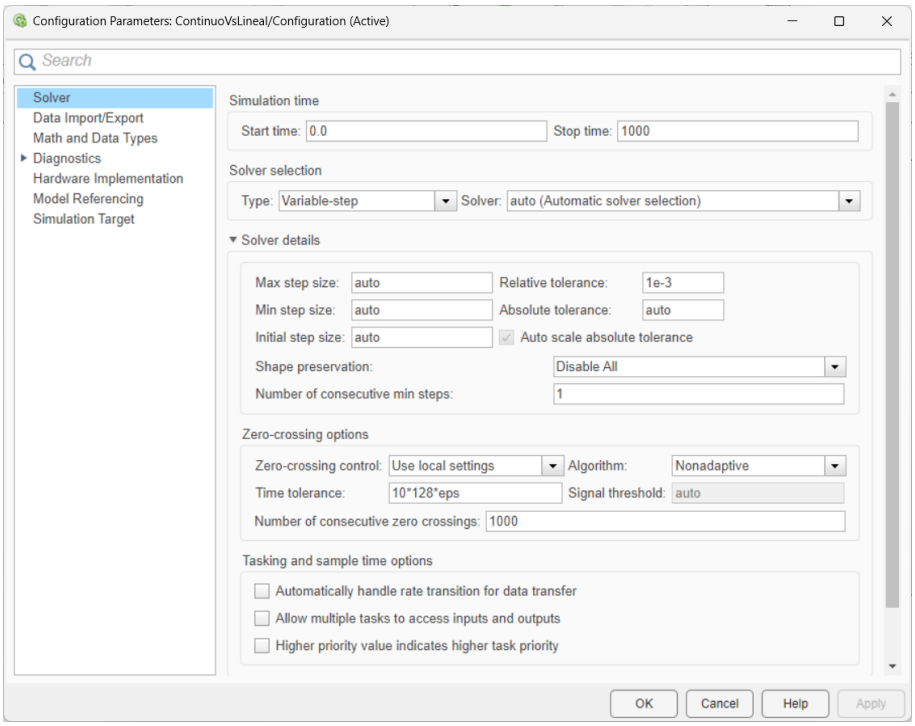

Figura 27. Configuration Parameters. Matlab.

A continuación, se crea un bloque de tipo ***Matlab Function***, en el que se copia el código del biorreactor. Este bloque tiene 6 entradas más el Ts y 7 estados, que se realimentan al sistema. En cada una de estas salidas, se pone un bloque ***Unit Delay*** que tendrá el valor inicial de las variables (Figura 28). Esto último se hace para que no se produzcan warnings de bucle algebraico, al intentar resolver una ecuación con el mismo valor.

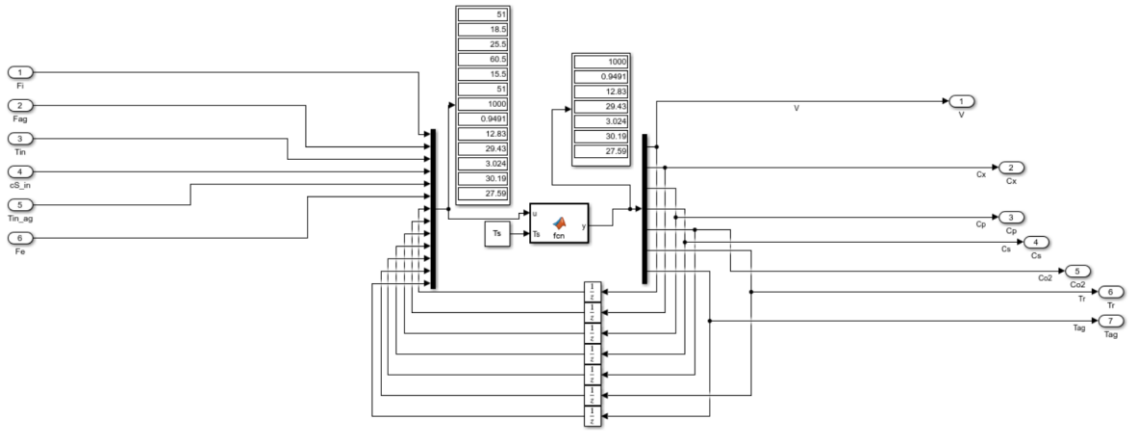

Figura 28. Bloque Matlab Function del Biorreactor

Para conseguir el punto de operación, se comienza con los valores iniciales del documento [1] y se les da un step en la entrada (Figura 29). El valor donde se estabilice cada una de las entradas será el punto de operación.

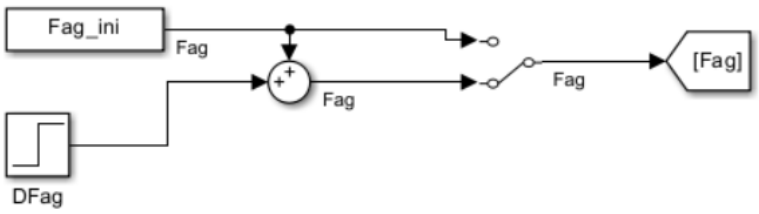

Figura 29. Esquema de variable de entrada Fag

Ahora viene la parte importante, que consiste en linealizar el proceso. Para esto, se ha hecho uso de la herramienta ***Linearize***, presente en el módulo ***Control System Toolbox***.

Haciendo click izquierdo en el bloque del biorreactor y seleccionando ***Linear Analysis – Linearize Block…***, se abre una nueva ventana con las opciones de linealización (Figura 30).

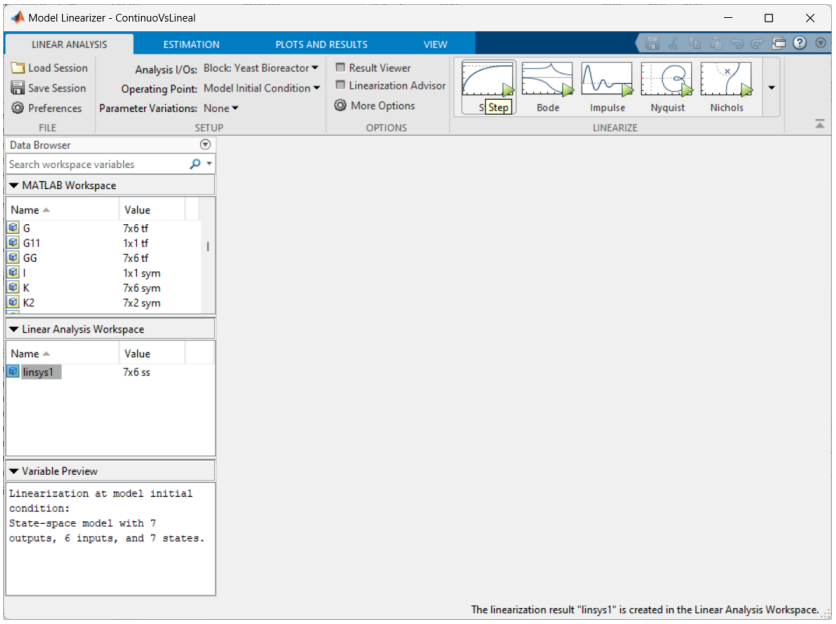

Figura 30. Herramienta Linearizer

Haciendo clic en Step, la herramienta genera una matriz de 7x6 en el espacio de estados, llamada ***linsys1***. Esto lo realiza a partir de los valores de punto de operación de las entradas.

Teniendo esta matriz, hay que pasarla ahora a una función de transferencia, para poder trabajar con ella y luego realizar el proceso de simplificación. El siguiente código muestra un ejemplo del proceso realizado, el cual debe de aplicarse de forma individual a cada uno de los elementos de la matriz:

```
--------------------------
load('linearizacion')
load('matriz_transferencia.mat')

s=tf('s'); %se establece la variable compleja s
A=linsys1.A;
```

```
B=linsys1.B;
C=linsys1.C;
D=linsys1.D;

% %  CÁLCULO DE MATRIZ DE TRANSFERENCIA GG
[num,den]=ss2tf(A,B,C,D,1) % transforma de espacio de estados a función de
transferencia y obtiene la columna 1 de la matriz
g11=tf(num(1,:),den) %función de transferencia en la posición 1-1
zpk(g11) %muestra la función de transferencia en polos y zeros
minreal(g11,0.1) %simplifica la función de transferencia eliminando los
términos que estén tanto en el numerador como en el denominador
pole(g11) zero(g11)
g11b= %ft simplificada
figure %abre una figura
step(g11,g11b) %pinta la ft con la simplificada y se observa si no se ha
perdido la dinámica
GG(1,1)=g11b %guardar en la matriz de transferencia GG la función de
transferencia simplificada en su posición correspondiente
--------------------------
```

A algunas de estas funciones de transferencia, se les realizó una simplificación extra, eliminando términos que afectaran poco a la dinámica y comprobándolo con la función original en una gráfica de step (Figura 31). Un ejemplo de la GG (2,1) antes y después de simplificar es el siguiente:

gg21:

$$\frac{-2.6087e-08\ (s-200)\ (s+200)\ (s+124.1)\ (s+2.108)\ (s+0.09657)\ (s+0.05092)\ (s+0.002004)}{s\ (s+124.1)\ (s+2.174)\ (s+0.05101)\ (s+0.03897)\ (s\text{^}2\ +\ 0.1234s\ +\ 0.005129)}$$

gg21d:

$$\frac{0.00101\ (s+0.09657)\ (s+0.002004)}{s\ (s+0.03897)\ (s\text{^}2\ +\ 0.1234s\ +\ 0.005129)}$$

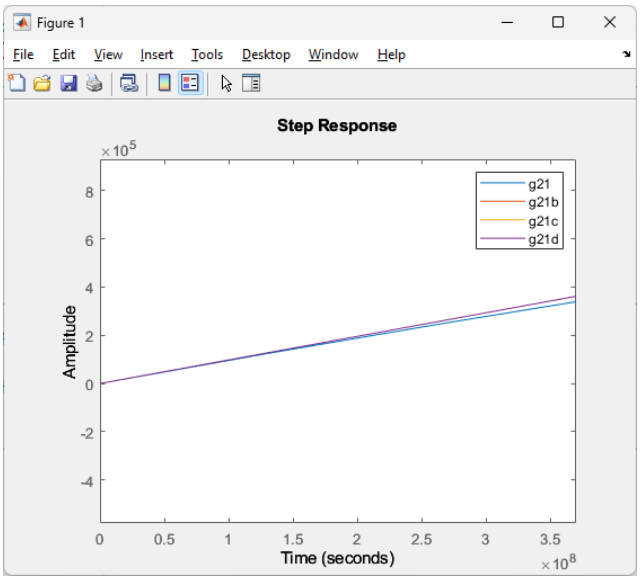

Figura 31. Step Response GG (2,1)

Este método se aplica de forma individual a cada uno de los elementos de la matriz de funciones de transferencia hasta quedar completa. Ya con ella, toca ahora compararla con el modelo no lineal, para comprobar que se haya realizado una buena linealización.

En Simulink, se agrega un bloque de tipo ***LTI System***, y se apunta a la matriz GG. En el caso del modelo linealizado, es necesario restarle a la entrada el punto de operación y sumárselo a la salida, pues esta matriz trabaja con variaciones sobre dichos puntos.

El siguiente paso es analizar los posibles emparejamientos y estudiar cuál es la mejor opción. Este proceso de selección se ha explicado ya con detalle en la sección Análisis de soluciones - Emparejamientos. Pero en la práctica, se han calculado todos los posibles emparejamientos. Para ello se utiliza el esquema mostrado en la Figura 32.

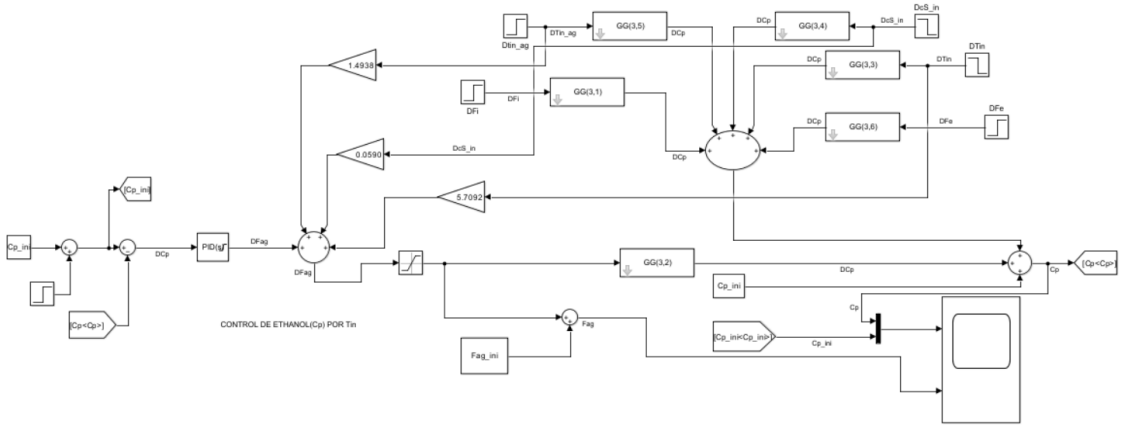

Figura 32. Esquema de diseño PID

Se configuran primero los parámetros del PID, teniendo desconectadas las perturbaciones y sus respectivas prealimentaciones. Se pone el bloque ***LTI System*** apuntando a la función de transferencia de ese emparejamiento y se cambia el punto de consigna, o set point, mediante un step.

El siguiente paso es elegir los parámetros del PID, para esto se realiza una primera aproximación mediante la herramienta ***Tune*** (Figura 33), que permite ajustar el ***Response Time*** y ***Transient Behavior***, hasta obtener una respuesta rápida y con apenas sobre oscilación.

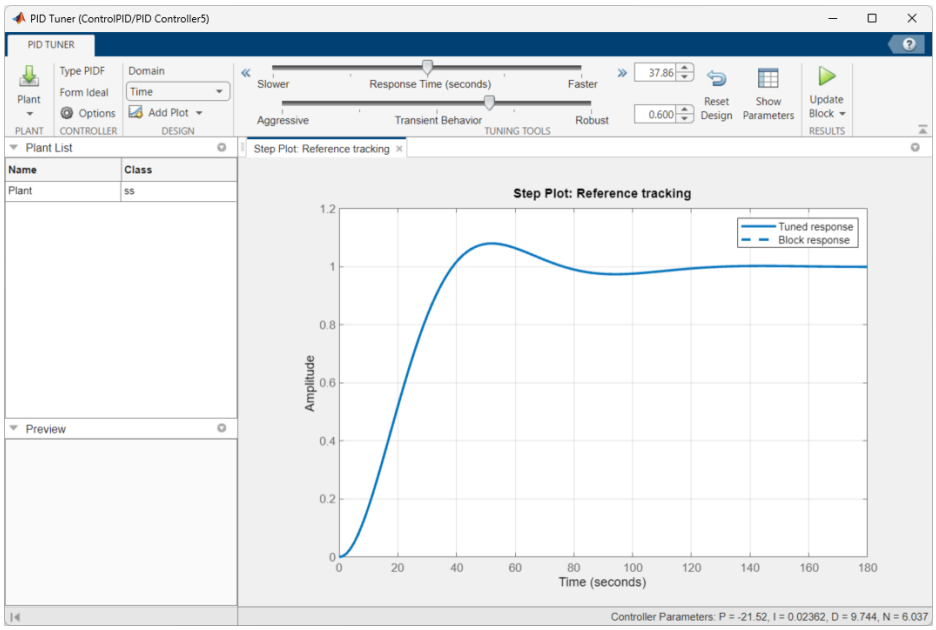

Figura 33. Herramienta Tune para PID

Se terminan de ajustar manualmente algunos parámetros hasta obtener un resultado adecuado. Ahora, ya con el PID configurado, se agregan las perturbaciones y se estudia el poner o no prealimentaciones, que en la mayoría de los casos mejoran el control.

Finalmente, para el caso del control principal, se obtienen las siguientes respuestas, con y sin prealimentaciones:

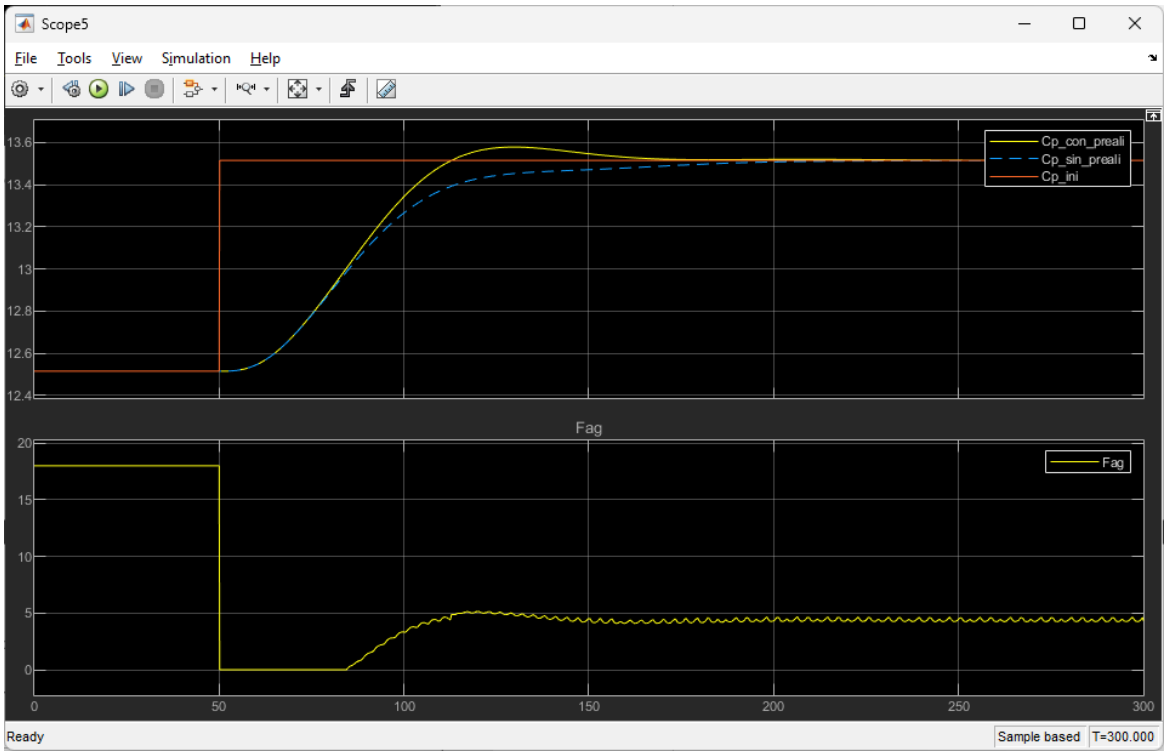

Figura 34. Respuesta al control de Cp con y sin prealimentaciones

Ya con los controladores configurados, es momento de pasarlo todo al proceso real (Figura 35).

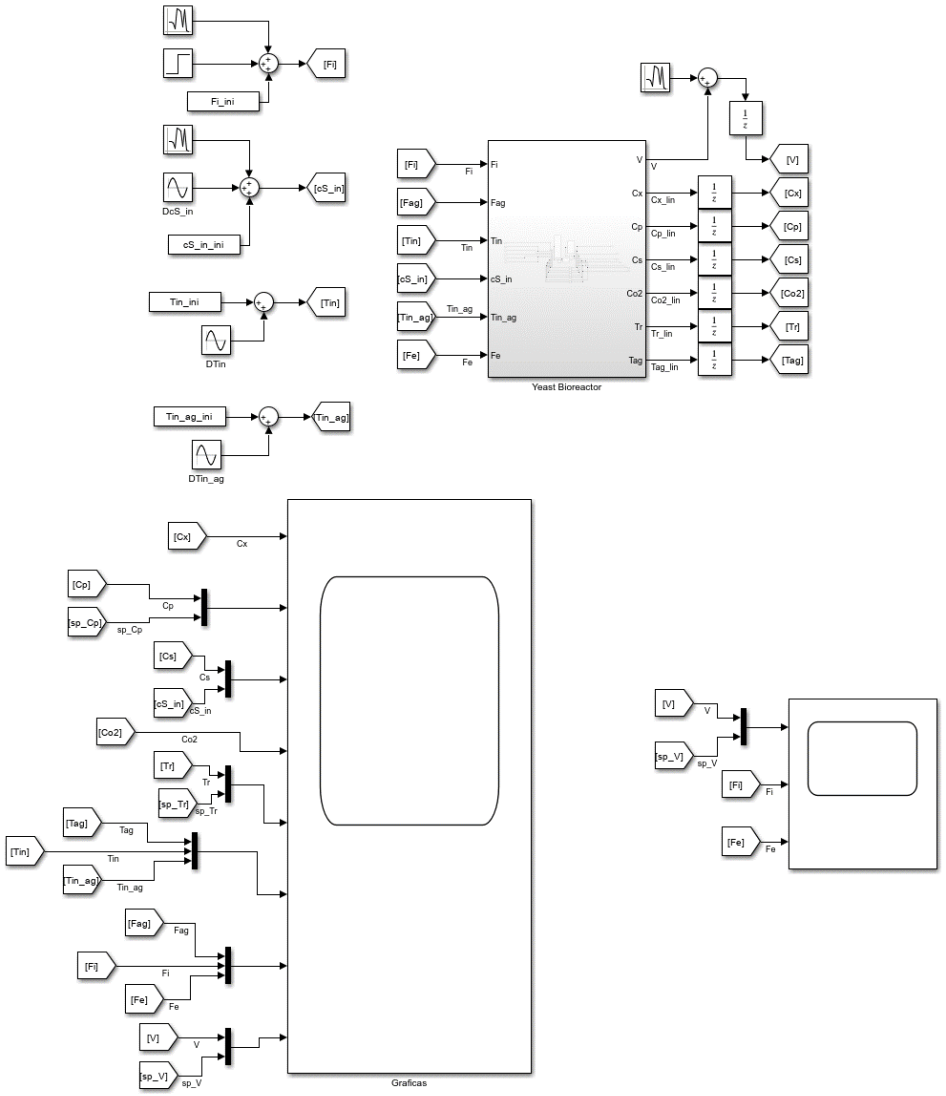

Figura 35. Diagrama de control no lineal 1

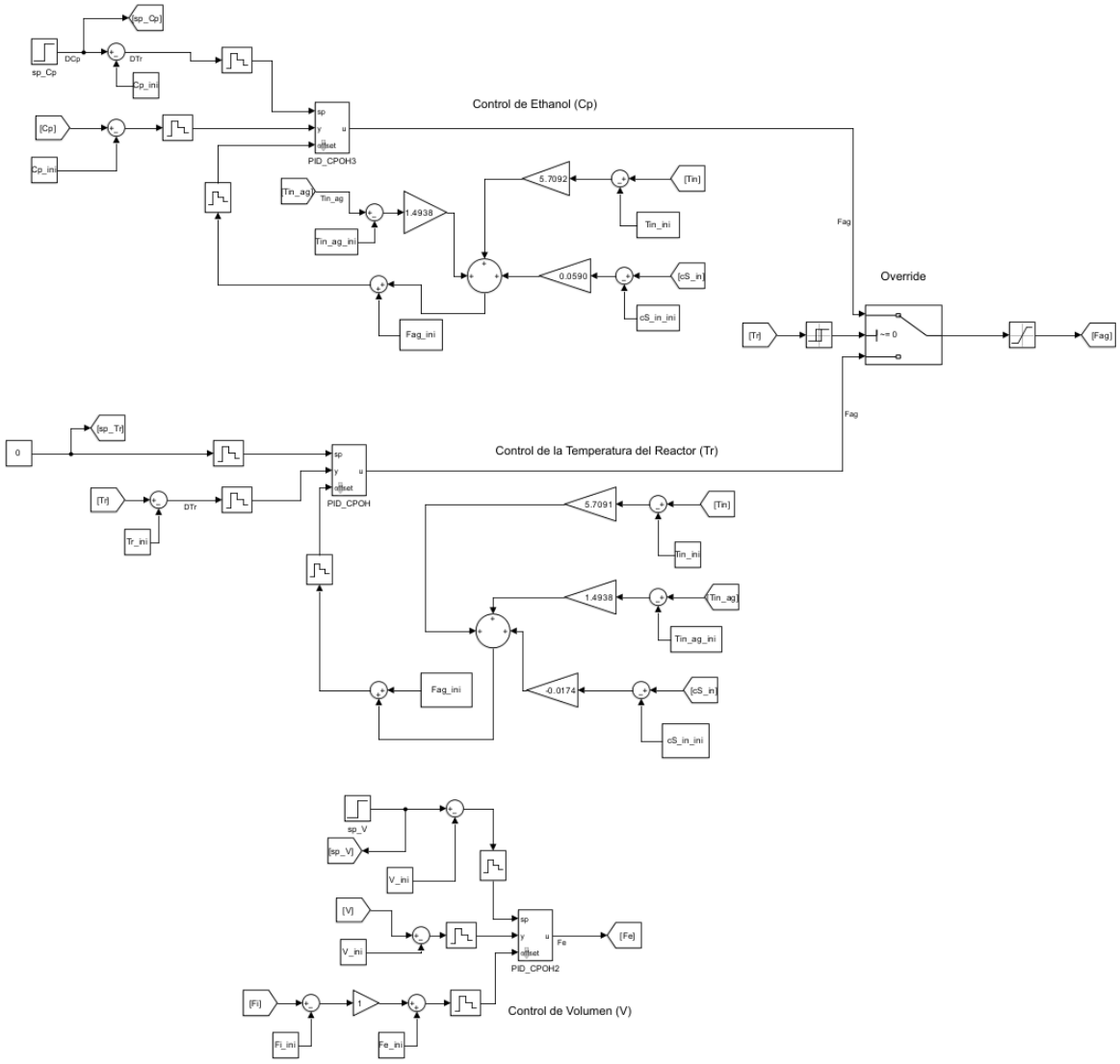

Figura 36. Diagrama de control no lineal 1

Para ello, hay que tener en cuenta que se debe transformar el PID de Simulink a un bloque personalizado para luego poder utilizarlo en Codesys. Este bloque está en formato ISA y dispone de filtros, antiwindup y también los valores de saturación dentro del propio PID. Los parámetros finales de los controladores son los siguientes:

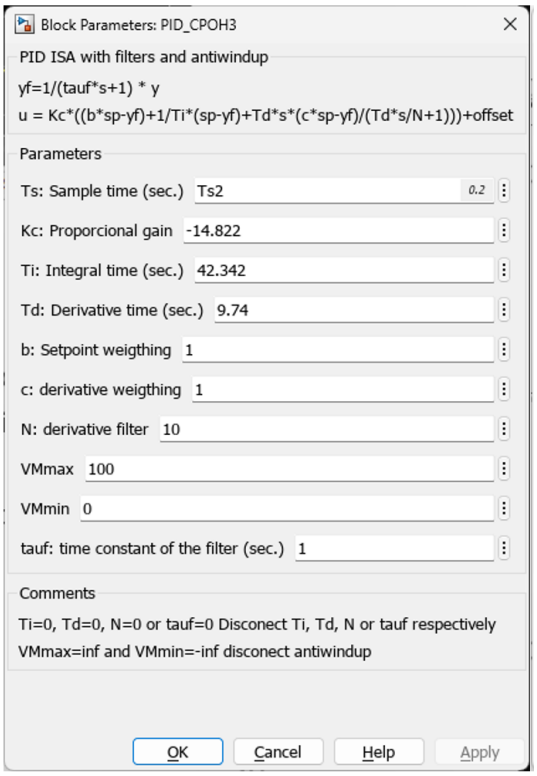

Figura 37. Parámetros PID_CPOH3

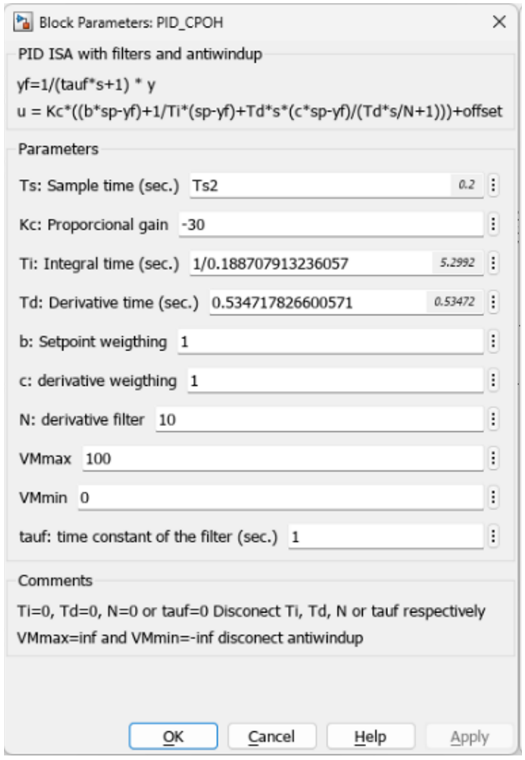

Figura 38. Parámetros PID_CPOH

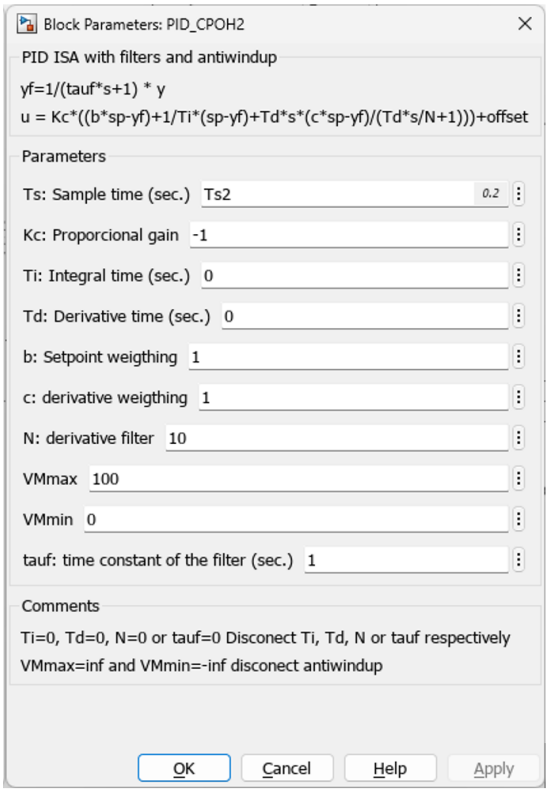

Figura 39. Parámetros PID_CPOH2

En cuanto al control override, se ha implementado con los bloques ***Relay*** y ***Switch***, que permiten que el control de Tr no se esté encendiendo y apagando constantemente al tener una histéresis (Figura 40).

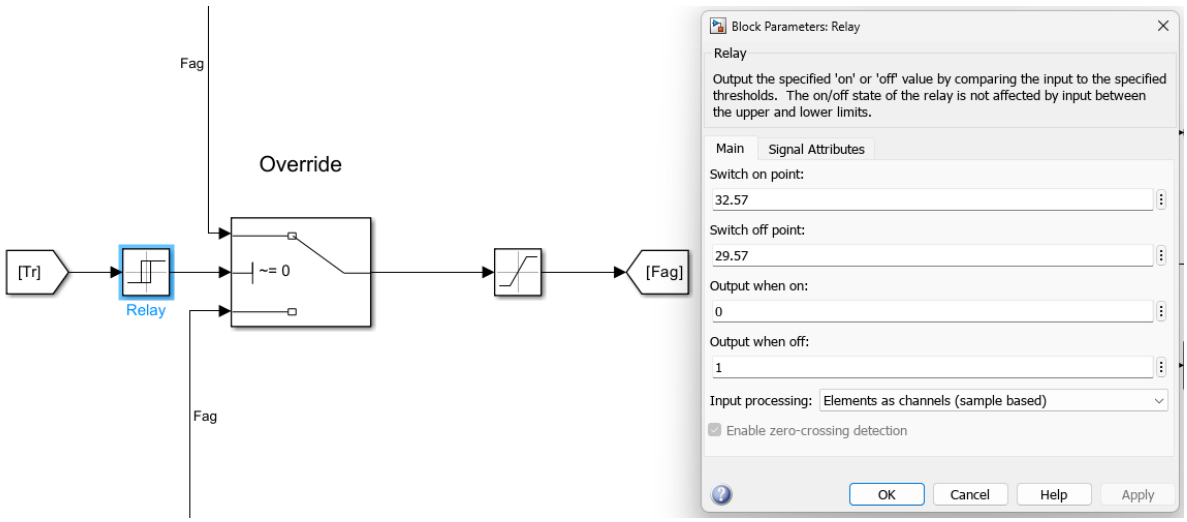

Figura 40. Control override con histéresis

También se han introducido también perturbaciones y ruidos en las entradas, además de un pequeño ruido en la salida V. Esto se ha hecho con el objetivo de asemejar el modelo lo máximo posible a la realidad (Figura 41).

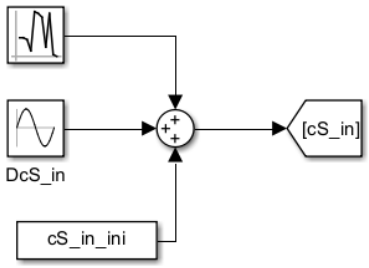

Figura 41. Perturbaciones y ruidos de entrada en cS_in

El resultado gráfico del control final para un set point de +0.2 en la concentración de etanol (Cp) es el siguiente:

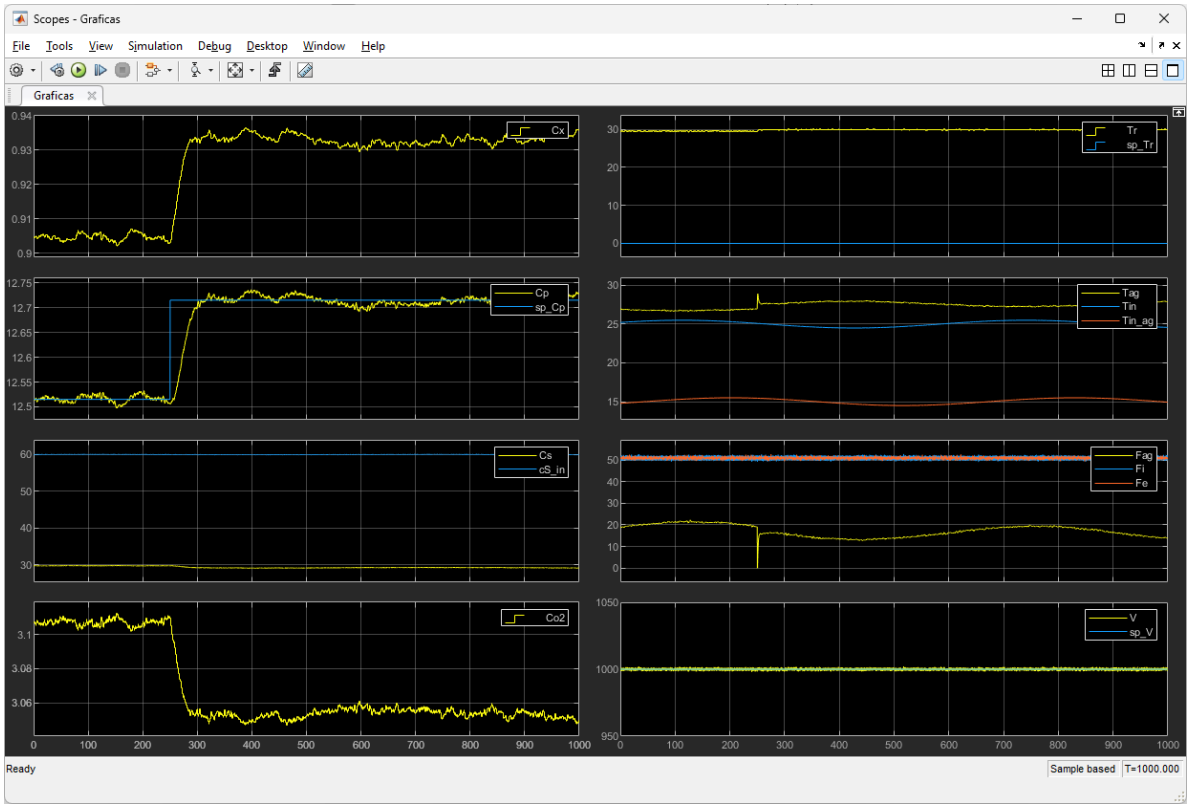

Figura 42. Gráficas finales control no lineal

2.3 Implementación estructura de control en PLC industrial (CODESYS)

En este apartado se desarrollan los pasos realizados para llevar a CODESYS el sistema de control diseñado previamente en Simulink, además de la configuración necesaria para la lectura y escritura de las variables por comunicación MODBUS TCP/IP.

Todo el código utilizado se encuentra en la sección 5. CÓDIGO - CODESYS.

El primer paso al abrir CODESYS es crear un nuevo proyecto. Se crea uno a partir de ***Proyecto vacío*** (Figura 43), esto permite editar desde cero todo el programa.

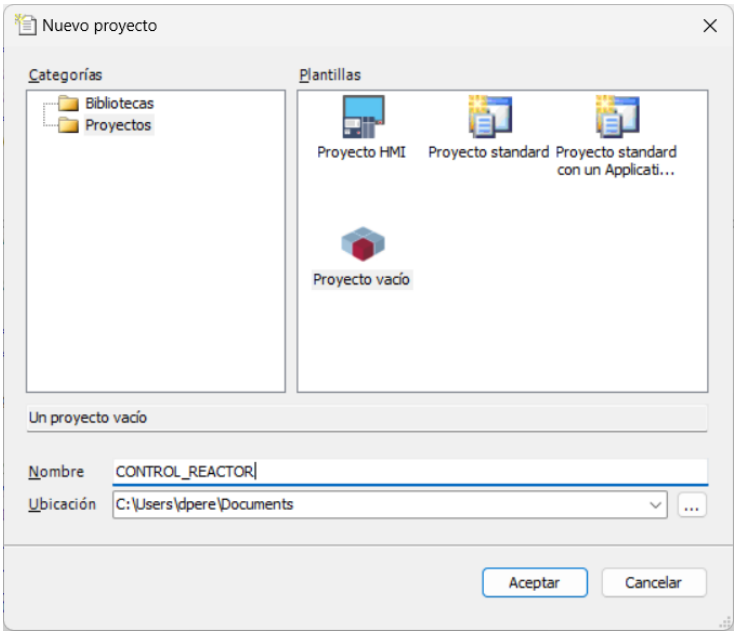

Figura 43. Nuevo proyecto CODESYS

Para agregar los diferentes componentes del programa, se hace uso de la columna de la izquierda. Lo primero es agregar el dispositivo que simula el PLC, en este caso se elige la opción de ***CODESYS Control Win V3 x64*** (Figura 44) y se configura la IP como localhost y OMEN-PC como el hardware. Debe arrancarse primero la instancia, que se hace mediante el icono en la sección derecha de la barra de herramientas  llamado ***CODESYS Control Win SysTray – x64***. Se hace clic derecho y start PLC.

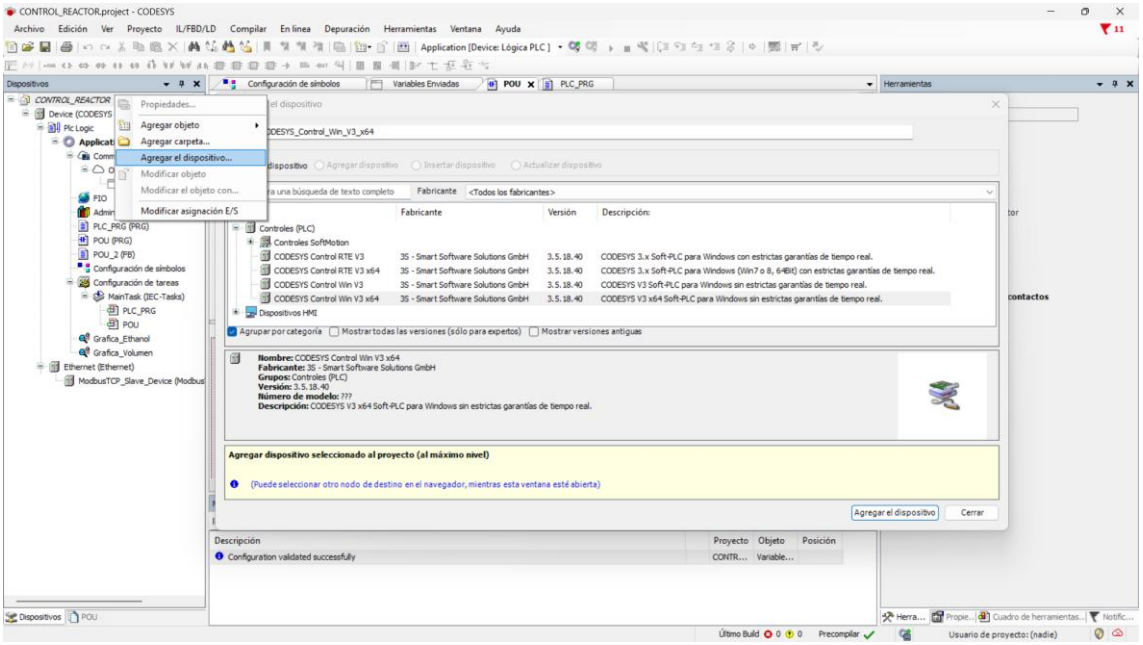

Figura 44. Dispositivo PLC, CODESYS Control Win V3 x64

Se agrega ahora en la sección ***Application*** los dos POUs del programa. Estos son:

- **PLC_PRG (PRG):** es el POU que se encarga, en lenguaje ST (AWL), del escalado de las señales, inicializar valores, cálculo de los offsets e implementación de la histéresis.
- **POU (PRG):** es el bloque donde se alojan los PIDs (). Este bloque se ha programado en lenguaje Ladder (KOP).

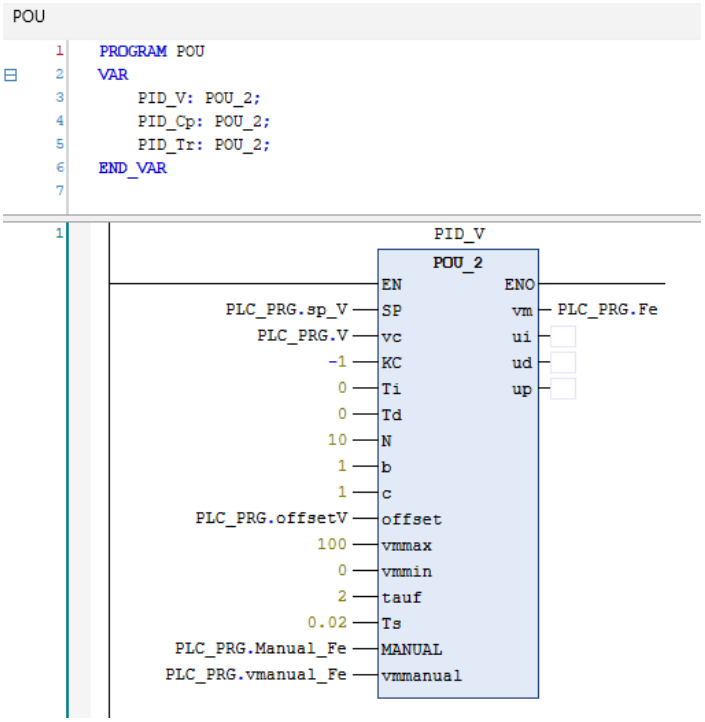

Figura 45. PID Control de Volumen. POU CODESYS

Para que estos dos POUs se ejecuten de forma cíclica, en CODESYS se le indica dentro de la sección ***MainTask***, donde también se coloca el valor del tiempo cada cuánto se ejecuta un ciclo (Figura 46).

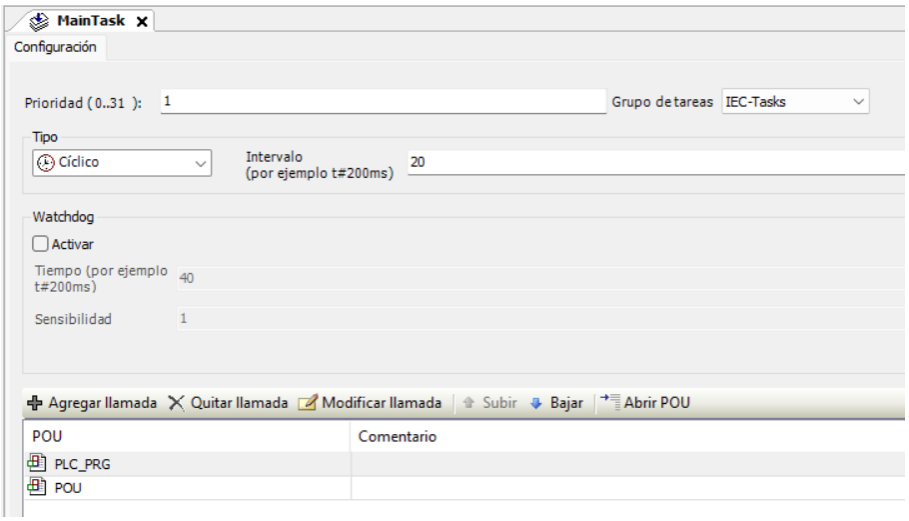

Figura 46. MainTask

De forma adicional, se crea un ***Function Block*** llamado POU_2 (FB) con el código que conforman los PIDs. Esto, al igual que el bloque de Simulink de PID, lo ha proporcionado el **DISA** de la **UPV**.

Para la comunicación MODBUS TCP/IP, se agrega un nuevo dispositivo de tipo ***Ethernet*** y dentro se agrega el componente ***ModbusTCP Slave:***

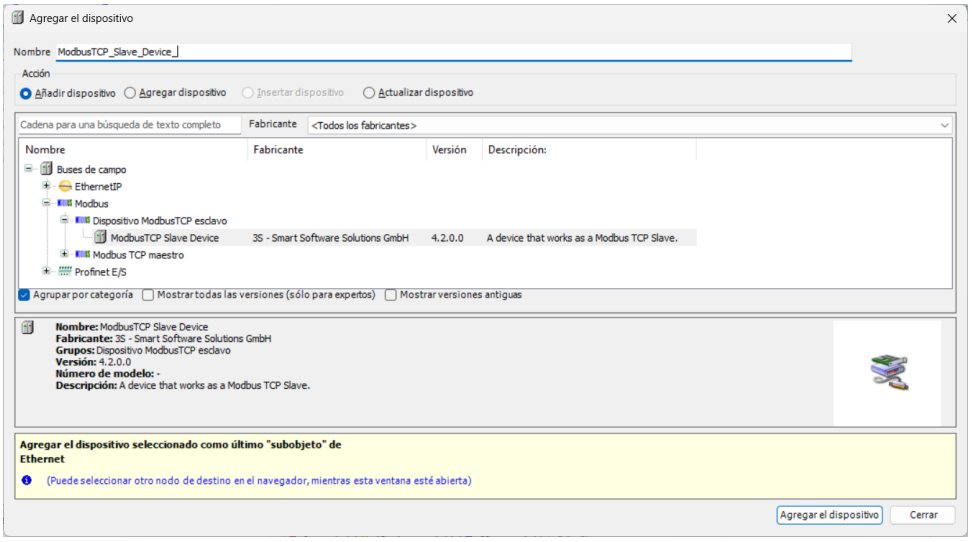

Figura 47. Dispositivo ModbusTCP Slave Device

Dentro de la configuración del dispositivo, se modifica lo siguiente en la pestaña ***General***:

- Puerto esclavo: 502
- Registros Holdings: 7 (entradas de CODESYS)
- Registros Inputs: 6 (salidas de CODESYS)

Para poder enviar estos datos, también hay que activar el check box en la pestaña ***Configuración de símbolos*** y crear un bloque de ***Lista de variables globales*** (Figura 48) para los datos en digital.

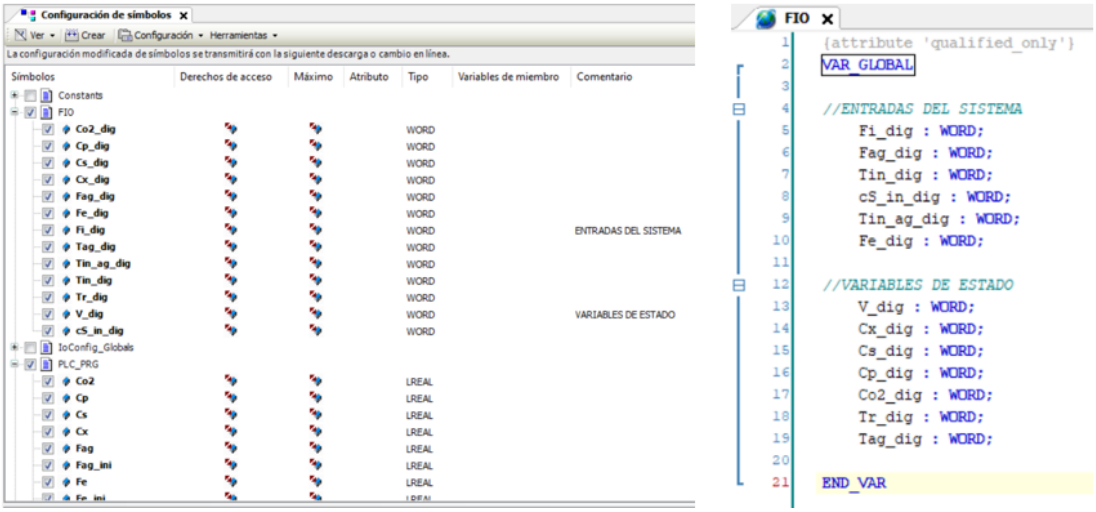

Figura 48. Configuración de símbolos (izquierda). Lista de variables globales FIO (derecha)

Por último, solo queda configurar la comunicación OPC UA con el SCADA. Para ello, se agrega el módulo ***Communication Manager > OPC UA Server > Variables Enviadas > Symbol Type Editor > Device.Application*** y se arrastra el POU ***PLC_PRG (PROGRAM)*** a la ventana de la derecha. Esto indica al PLC, qué variables serán exportadas (Figura 49).

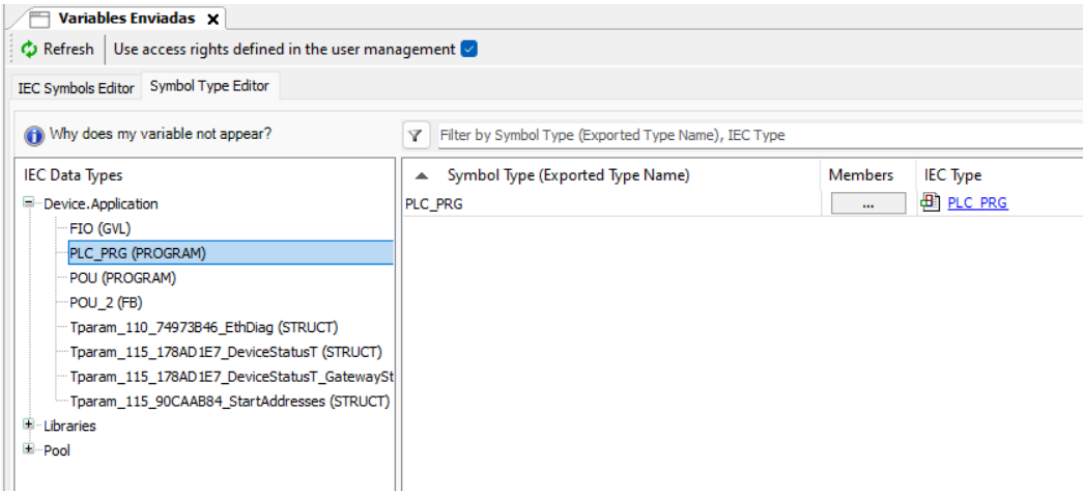

Figura 49 Variables enviadas OPC UA

Como extra, CODESYS permite añadir gráficas (Figura 50), con el objetivo de comprobar que se esté realizando el control de igual forma que lo diseñado en Simulink. Para ello se hace click con botón derecho en ***Application > Agregar objeto > Trace***.

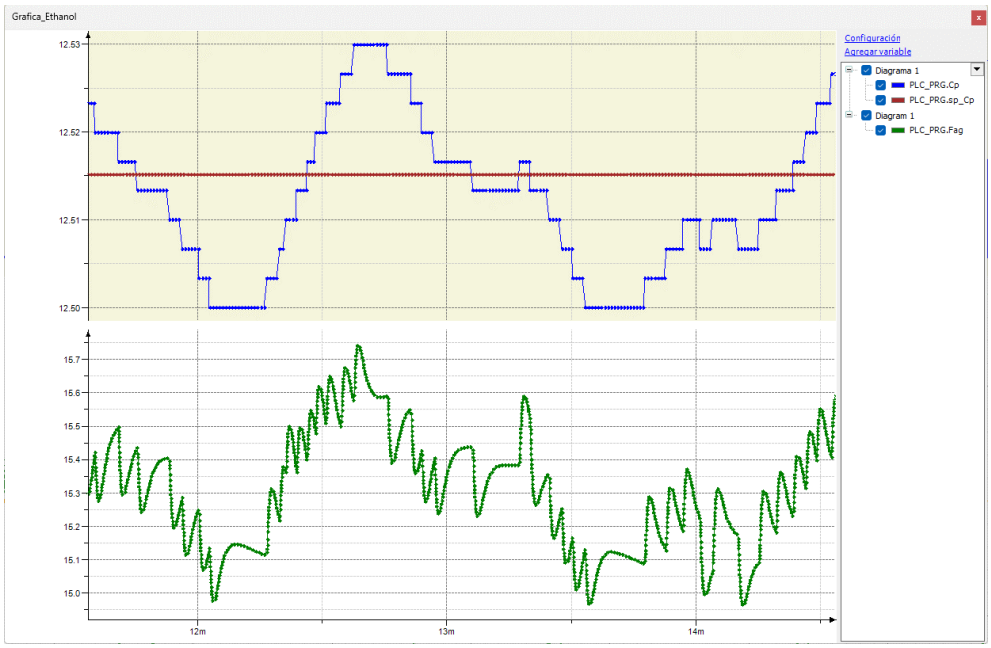

Figura 50. Gráficas de control de Cp con Fag

Los módulos resultantes de todo este proceso, y que se muestran en la pestaña izquierda ***Dispositivos*** son los siguientes:

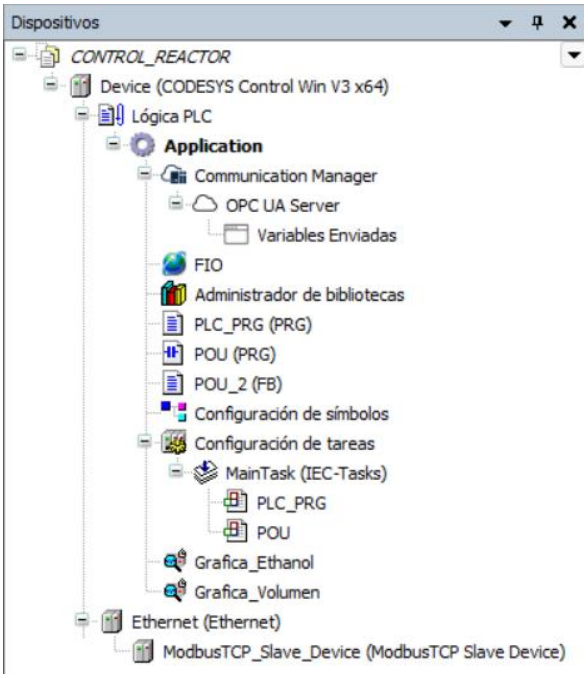

Figura 51. Columna de dispositivos CODESYS

2.4 Desarrollo del SCADA (IGSS)

2.4.1 Configuración comunicación OPC UA

Para configurar la parte cliente del servidor OPC UA, se sigue el siguiente procedimiento:

1. Creación del proyecto, instalación de licencia FREE50 (Figura 52) y acceso al Modo Diseño.

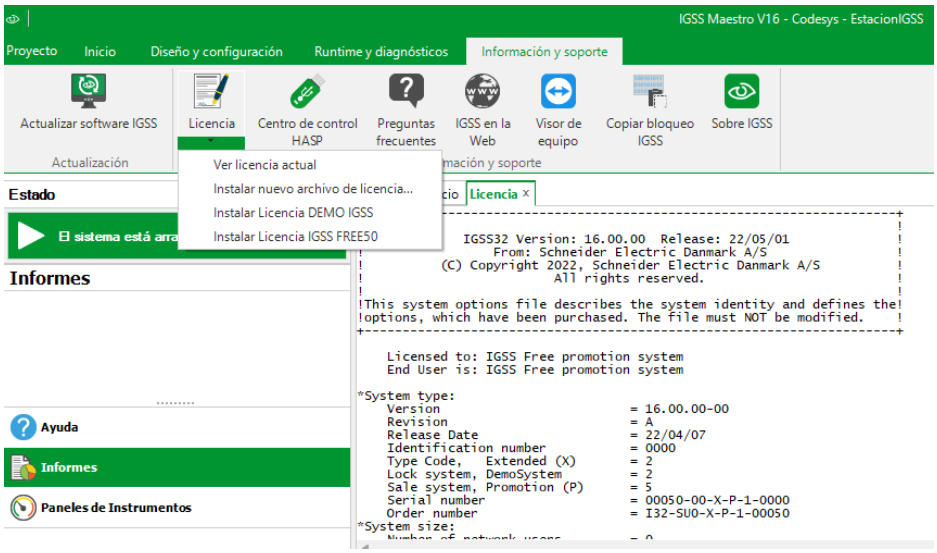

Figura 52. Licencia FREE50 IGSS

2. En la pestaña ***Diseño y configuración***, se selecciona la opción ***Configuración del sistema*** que permite configurar la conexión de la aplicación con las diferentes fuentes de datos, en este caso con CODESYS.

3. Configuración de la estación: se crea una nueva estación que representará la conexión con el servidor OPC. Se le añade el driver ***OPC UA Client Side*** para establecer la comunicación con el servidor.

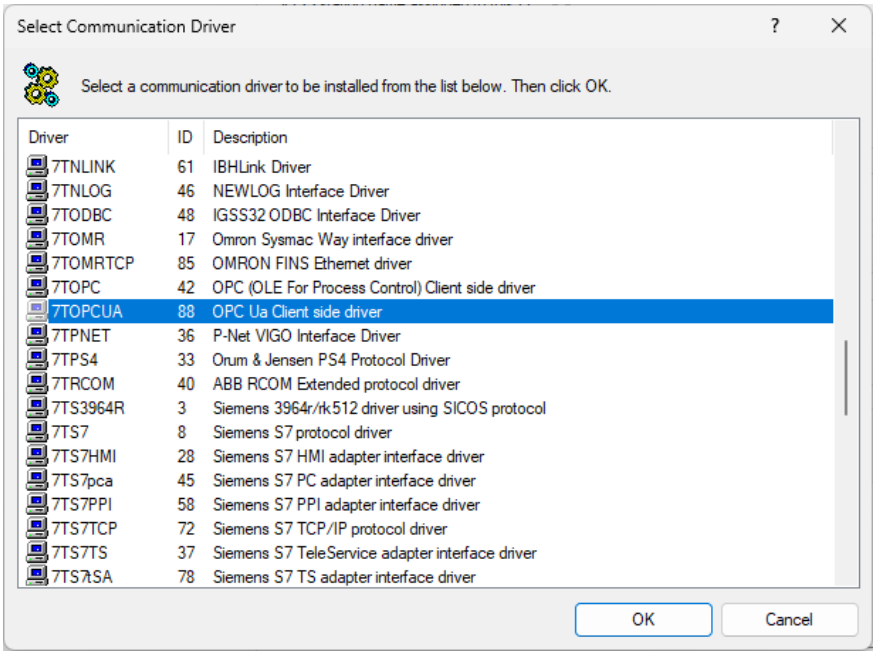

Figura 53. OPC UA Cliente side driver

4. Configuración del canal: Dentro de la estación, se pueden configurar varios canales de comunicación. Para este caso, basta con configurar un canal. En el nodo correspondiente al canal, se establece la dirección IP y el puerto del servidor OPC: ***opc.tcp://OMEN-PC:4840***.

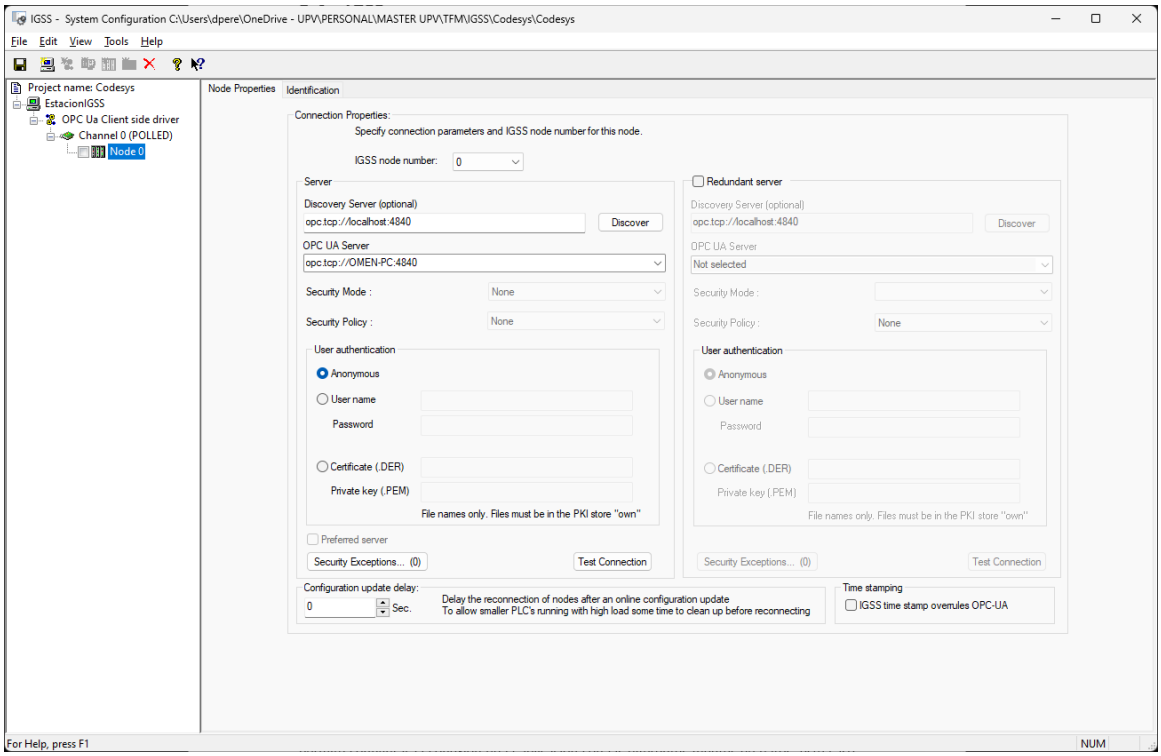

Figura 54. Configuración del cliente OPC UA

Una vez se ha completado esta configuración, se podrán asociar los valores almacenados en el servidor OPC UA con las variables definidas en la interfaz de la aplicación. Esto permitirá la lectura y escritura de los datos del servidor, lo cual es fundamental para el desarrollo del sistema de control industrial.

2.4.2 Desarrollo de la interfaz

Para comenzar a crear la interfaz, se accede al módulo ***Definición*** dentro de la pestaña ***Diseño y configuración***.

El esquema que utiliza IGSS para el flujo de datos y creación de componentes es el siguiente:

Figura 55. Esquema flujo de datos IGSS

Por lo tanto, se crea primero un área ***General*** que va a alojar todos los demás componentes.

Ahora, se crean sobre esta área los diagramas que conforman las diferentes pantallas del SCADA. En este proyecto se han creado los siguientes 4 diagramas:

- **Menú Biorreactor:** es la pantalla principal que se muestra al abrir el sistema. Permite realizar las funciones principales de control y monitorizar todas las variables, además del estado de las alarmas.
- **P&ID:** pantalla que muestra el diagrama de P&ID de la planta.
- **Guía de operación:** guía general para el uso adecuado de este SCADA.
- **Gráficas:** se muestran un histórico gráfico de todas las variables.

Tras esto, se proceden a añadir las variables que se están enviando por el canal de OPC UA. Esto se puede realizar de dos formas, o con botón derecho sobre el fondo y ***Nuevo > Tipo de objeto > Analógico/Digital***, o abriendo el ***Examinador de objetos*** (Figura 56) con el comando ***Control + E*** y seleccionando el tipo de variable que se desee añadir.

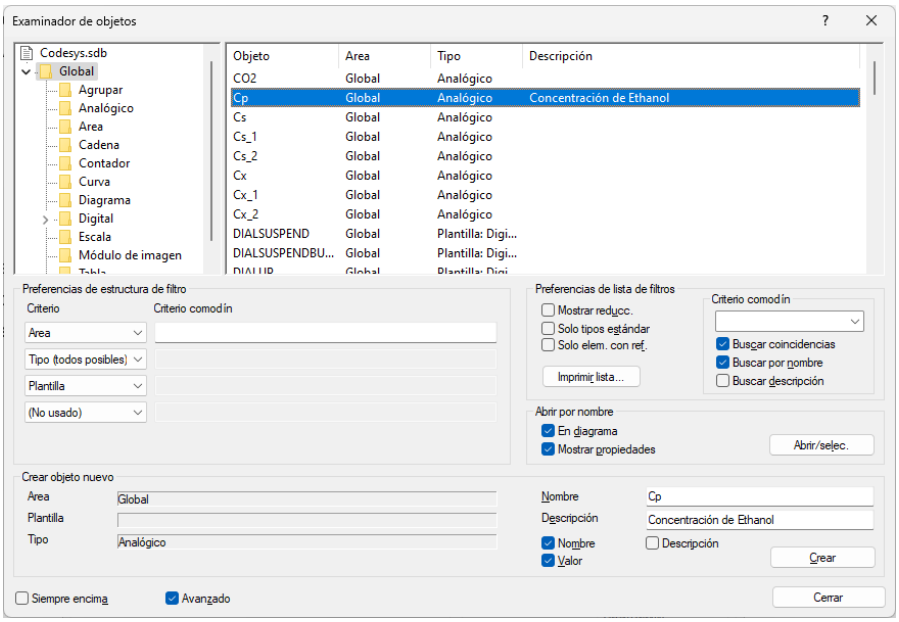

Figura 56. Examinador de objetos

Dentro de la configuración de cada variable, se deben ajustar múltiples parámetros, el más importante de ellos es la propia conexión con la variable enviada de CODESYS. Esto se realiza en la pestaña ***Mapeo de átoms***. En el átom de ***Valor actual***, se abre una pestaña nueva llamada ***IGSS OPC UA Browser*** (Figura 57), mediante la que debemos buscar la variable a la que corresponde para vincularla. Dada la configuración realizada en CODESYS, estas variables se alojan en la carpeta ***Objects > DeviceSet > CODESYS Control Win V3 x64 > Resources > Application > Programs > PLC_PRG.*** En el caso de variables que se desea no solo leer si no también escribir su valor, se modifica el desplegable ***Modo E/S*** y se pone en ***e/s***. Se añaden también los átoms de ***Alarma alta/Alarma baja*** para poder configurar el estado de alarmas en las variables que se deseen monitorizar.

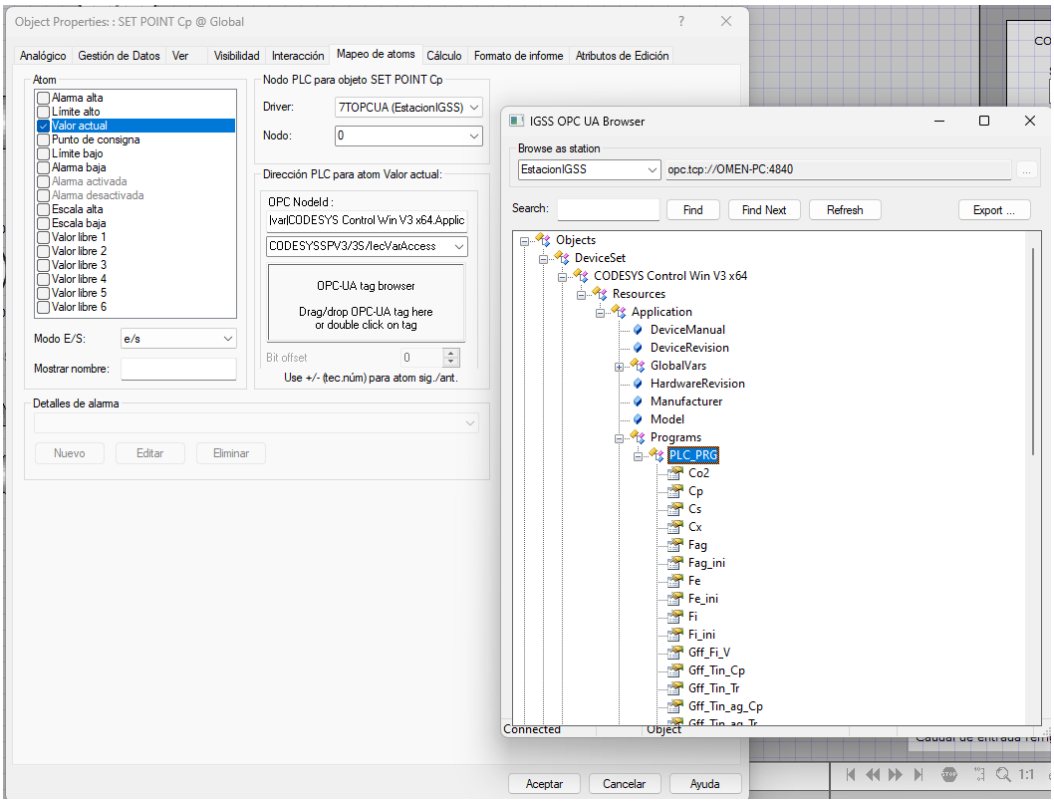

Figura 57. Vinculación variable Set Point Cp. IGSS – CODESYS

En la pestaña ***Analógico*** (Figura 58) se añade el valor inicial de la variable, su límite máximo y mínimo, las unidades y el número de decimales que se desean mostrar.

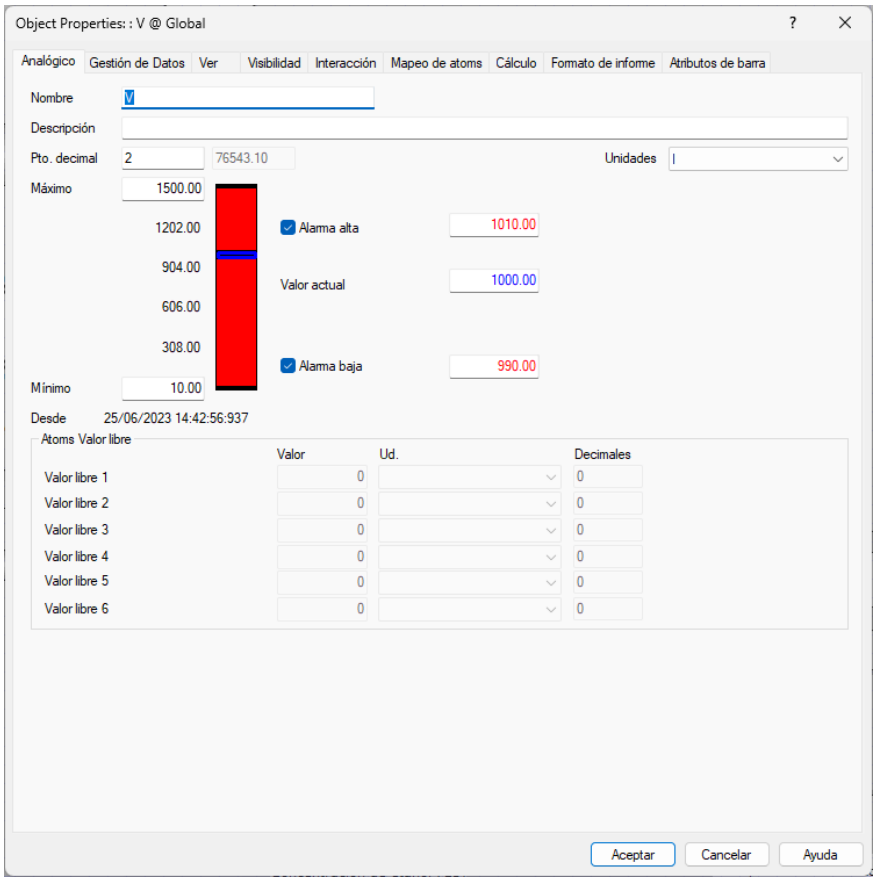

Figura 58. Pestaña configuración variables analógicas

Para agregar los bloques que definen visualmente el biorreactor, se añaden ***Descriptores***. Este icono  se encuentra en la ***Barra de herramientas de dibujo***. Para poder visualizarla, primero hay que activarla en la pestaña ***Vista*** (Figura 59).

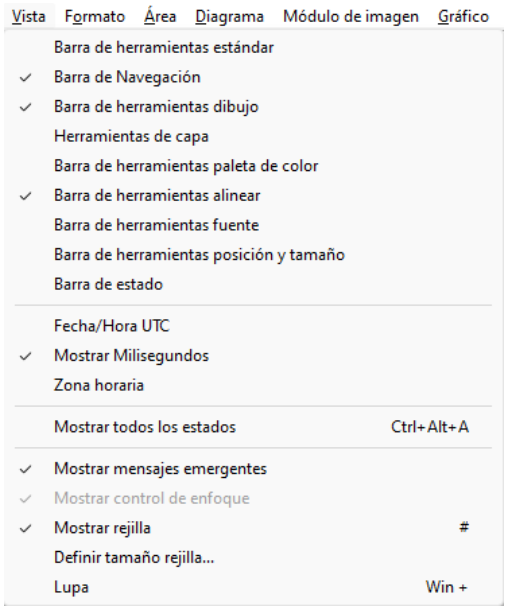

Figura 59. Vistas activadas en Modo Diseño

Siguiendo este proceso se configuran las variables a mostrar, las controlables y los bloques del biorreactor. Se añaden ***check box*** para el modo override y el modo manual (Figura 60).

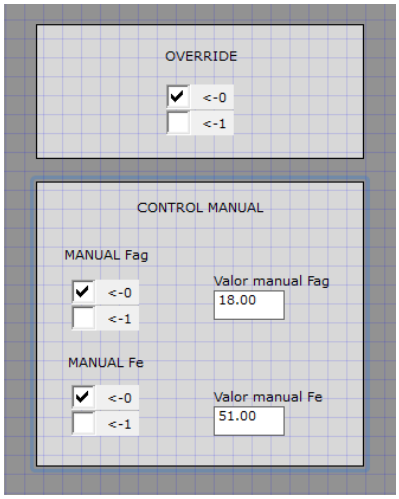

Figura 60. Check box para override y control manual

El siguiente paso es añadir la visualización de las alarmas, para ello se agregan indicadores visuales que se muestran en verde cuando el estado está en OK y en rojo cuando están en ERROR. Esto se añade con botón izquierdo sobre el diagrama ***Nuevo > Descriptores estándar > elipse.***

Para configurarla se hace doble clic sobre la elipse y se modifica el color en ***Atributos de elipse > Color de fondo***.

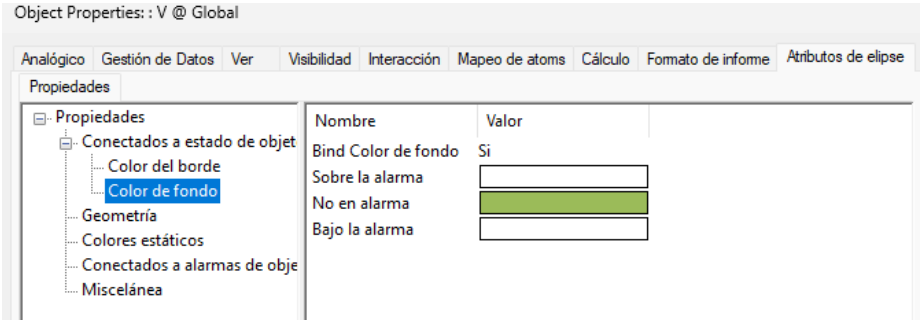

Figura 61. Configurar color de alarma en elipse

De manera similar, se añade también una lista de alarmas y tres gráficos básicos a la pantalla principal. Con todo esto, la pantalla principal se muestra en el modo ***Definición*** como muestra la siguiente imagen:

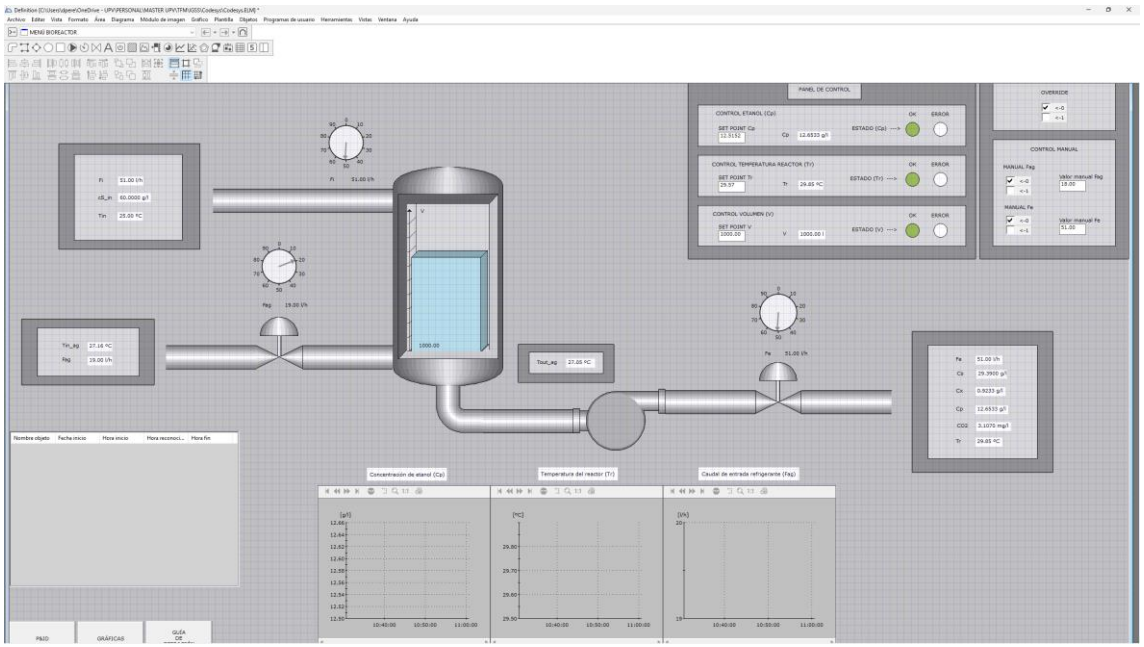

Figura 62. Pantalla principal SCADA. Módulo Definición

A continuación, se pretende establecer el menú con el que poder acceder a los demás diagramas. Ya con los diagramas creados, lo primero es añadir los botones que hacen de vínculo, seleccionando el icono ***Botón***  de la ***Barra de herramientas dibujo***. Tras esto, se da botón derecho sobre el bloque, ***Conectar…*** y se selecciona el diagrama que corresponde. Con esto se consigue que al hacer clic sobre alguna pestaña del menú, se abra una nueva ventana con la información extra.

Ya con el menú creado y vinculado, se procede a diseñar estos nuevos diagramas.

En el caso de **Gráficas**, se añaden gráficas individuales y compartidas de todas las variables. Botón derecho sobre el fondo y ***Nuevo > Visualización de datos > Gráfico…***

Dentro de las propiedades de estos gráficos (Figura 63), se deben añadir las variables que se quiere mostrar en cada una de ellas, además de limitar los ejes, activar la leyenda y establecer el periodo inicial de muestra.

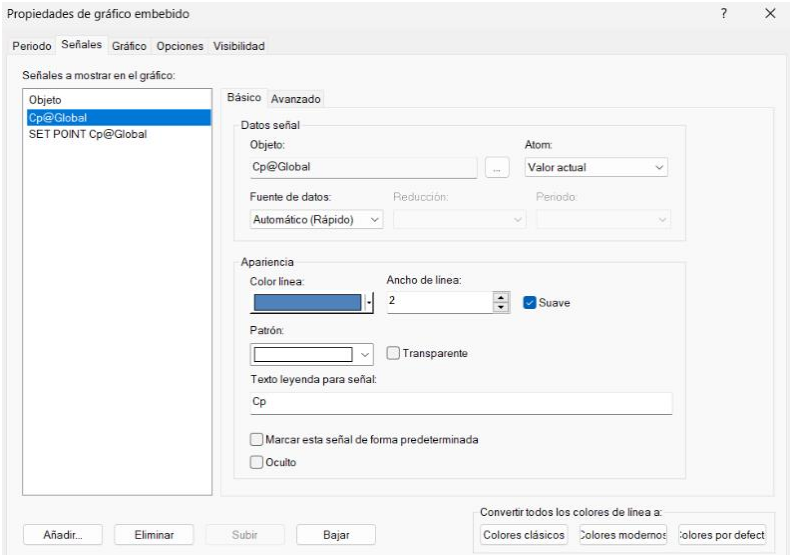

Figura 63. Propiedades de gráfico embebido

Es importante destacar, que para que los valores se puedan mostrar de forma histórica, es necesario marcar previamente la opción de registro (Figura 64) en las propiedades de cada variable. ***Object Properties > Gestión de Datos > Registro > Todos los valores***.

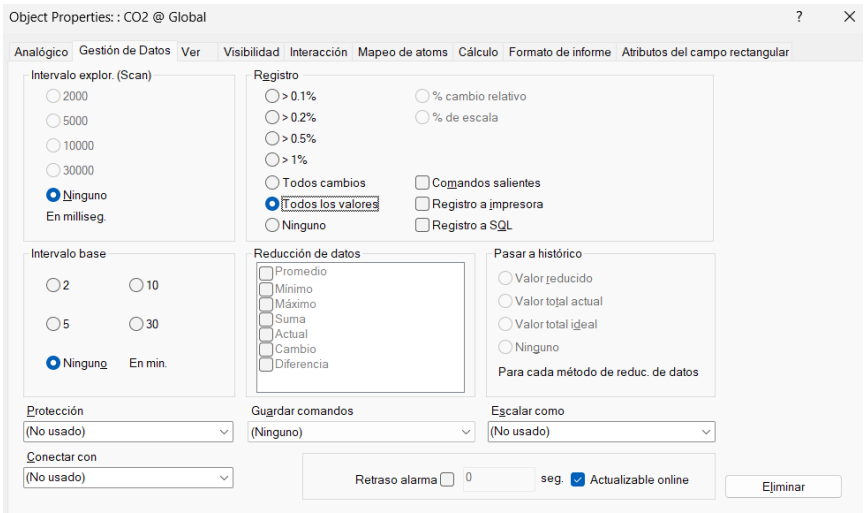

Figura 64. Registro histórico de valores

El diagrama del ***P&ID***, se ha diseñado primero mediante el Software ***Visio*** de Microsoft, haciendo uso de la herramienta Polilabs que pone a disposición de los estudiantes un PC remoto.

Para mostrarlo en el SCADA, se debe hacer desde la propia configuración del diagrama (Figura 65). Se accede a ***Propiedades de diagrama > Fondo > Imagen*** y se selecciona en ***Examinar*** la imagen en png del ***P&ID***. Se configura también el tamaño y posición de la ventana, que en este caso se ha optado por ser menor que la pantalla principal y así poder ver ambas a la vez.

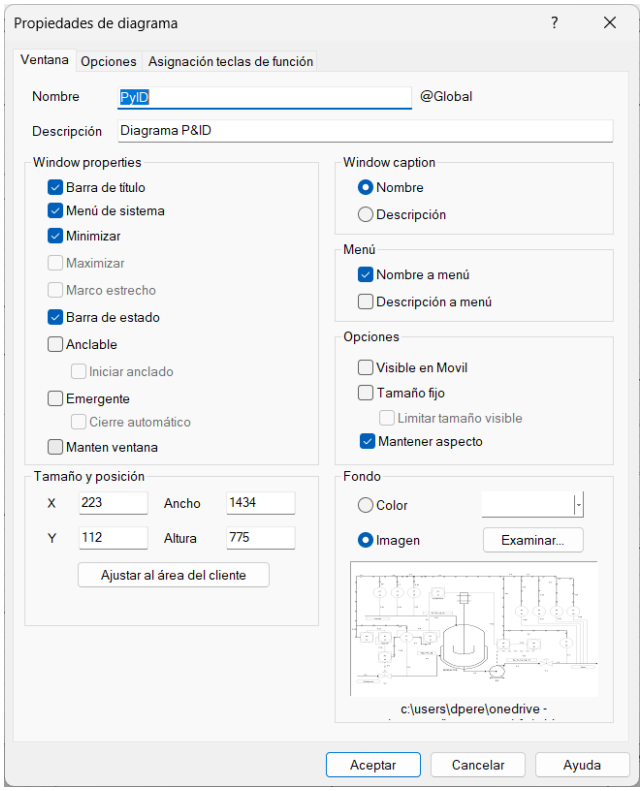

Figura 65. Propiedades diagram a P&ID

El último diagrama que se añade al SCADA es el de la ***Guía de operación***, que consiste en una ventana a la que se le agrega el objeto de tipo ***Imagen*** en el que se muestra el texto de la descripción.

Escuela Técnica Superior de Ingeniería Industrial

ETSII

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Máster Universitario en Ingeniería Industrial (Acceso desde Grado I. Electrónica Industrial y Automática)

Desarrollo e implementación de un sistema de control industrial de un biorreactor para la producción de etanol

Trabajo Fin de Máster

3. PRESUPUESTO

Autor: Daniel Pérez Gutiérrez

Tutor académico: Francesc Xavier Blasco Ferragud

Julio 2023

CAPÍTULO 3. PRESUPUESTO

En esta sección del documento, se presentará información detallada sobre los costos asociados al proyecto. Se dividirán en dos grupos principales: costos de mano de obra y costos de materiales utilizados. El objetivo es proporcionar una visión clara de los recursos económicos requeridos para la realización del proyecto. No se incluyen los precios de la instalación real, (va por cuenta del cliente: sensores, PLCs, biorreactor, …) solo se detalla el precio de desarrollo del sistema de control. Tampoco se incluyen los precios de puesta en marcha.

Para encontrar las referencias de los precios mencionados en este presupuesto, se recomienda consultar el apartado de 3.4 Bibliografía del presupuesto, que se encuentra al final de este capítulo. En ese apartado, se proporcionarán las fuentes utilizadas para obtener los precios de los diferentes elementos y unidades de obra incluidos en el presupuesto. Estas referencias bibliográficas permitirán acceder a la información detallada sobre los precios utilizados, brindando transparencia y respaldo a la elaboración del presupuesto

3.1 Mediciones

Tabla 2. Unidades de obra del proyecto. Distribución de tiempo

CÓDIGO	NOMBRE	UNIDADES	HORAS
U-01	**PROGRAMACIÓN MATLAB**		
U-01-01	IMPLEMENTACIÓN GEMELO DIGITAL	1	60
U-01-02	IMPLEMENTACIÓN COMUNICACIÓN MODBUS TCP/IP	1	10
U-02	**DISEÑO DEL CONTROL**		
U-02-01	ANÁLISIS LINEAL DEL MODELO	1	20
U-02-01	CONFIGURACIÓN PIDs	1	20
U-03	**IMPLEMENTACIÓN DEL CONTROL**		
U-03-01	PROGRAMACIÓN CONTROL PLC	1	45
U-03-02	PROGRAMACIÓN PARTE SERVIDOR MODBUS	1	20
U-03-03	PROGRAMACIÓN COMUNICACIÓN OPC-UA	1	4
U-04	**PROGRAMACIÓN SCADA**	1	40
U-05	**PRUEBAS**		
U-05-01	PRUEBAS SOBRE MATLAB	1	24
U-05-02	PRUEBAS SOBRE PLC	1	10
U-05-03	PRUEBAS DEL SCADA	1	10
U-06	**ELABORACIÓN DEL DOCUMENTO**	1	45
	TOTAL	12	308

Tabla 3. Unidades de obra

UNIDADES DE OBRA	DESCRIPCIÓN
IMPLEMENTACIÓN GEMELO DIGITAL	Programación y montaje del modelo no lineal del biorreactor
IMPLEMENTACIÓN COMUNICACIÓN MODBUS TCP/IP	Programación y montaje del script para comunicación Modbus entre biorreactor y PLC
ANÁLISIS LINEAL DEL MODELO	Aplicación de las téncnicas de linealización sobre el modelo no lineal
CONFIGURACIÓN PIDs	Diseño y configuración de los parámetros de control de los PIDs
PROGRAMACIÓN CONTROL PLC	Implementación del algoritmo de control para gobernar la planta
PROGRAMACIÓN PARTE SERVIDOR MODBUS	Programación de la comunicación Modbus en CODESYS
PROGRAMACIÓN COMUNICACIÓN OPC-UA	Programación de la comunicación OPC-UA en el CODESYS
PROGRAMACIÓN SCADA	Diseño y programación de la interfaz de sistema en IGSS
PRUEBAS SOBRE MATLAB	Pruebas realizadas para verificar la correcta definición de los modelos y los resultados obtenidos
PRUEBAS SOBRE PLC	Pruebas realizadas para verificar la correcta implementación de las comunicaciones y el algoritmo de control
PRUEBAS DEL SCADA	Pruebas realizadas para la comprobación del buen funcionamiento de la aplicación SCADA en IGSS
ELABORACIÓN DEL DOCUMENTO	Redacción del proyecto y elaboración de la presentación

3.2 Cuadros de precios

3.2.1 Cuadro de precios unitarios

En este apartado se presentarán los precios unitarios de cada elemento que intervienen en la consecución del proyecto, diferenciando entre mano de obra y material necesario. Se tomarán en cuenta los siguientes aspectos:

Mano de obra

Para la mano de obra, se consideran 250 días laborales al año, con una jornada de 8 horas diarias. Se ha empleado un contrato indefinido como estándar. El cálculo de la contribución a la seguridad social a cargo de la empresa se ha realizado utilizando el siguiente porcentaje:

- Contribución a la seguridad social: 23,60% (cotización por contingencias comunes) + 5,50% (cotización por desempleo) + 1,00% (cotización por formación profesional) + 0,60% (cotización por FOGASA) + 0,20% (cotización por accidentes de trabajo y enfermedades profesionales) = 30,90%

Tabla 4. Costes horarios de mano de obra

Rol	Salario Bruto Anual (€)	Seguridad Social (€)	Pagas Extra (€)	Coste Anual (€)	Coste Hora (€/h)
Ingeniero Industrial Junior	20,244.14	6,255.44	3,374.04	29,873.62	14.94

Material

El apartado de materiales empleados se divide en dos grupos: hardware y software.

Para el hardware, se ha tomado como referencia las tablas de amortización de la Agencia Tributaria de España para estimar la vida útil de los diferentes equipos.

Tabla 5. Costes horarios de hardware empleado

Código	Hardware	Unidades	Precio (€)	Vida Útil (h)	Coste Horario (€/h)
(MH.1)	**Ordenador de sobremesa** (teclado+ratón+monitor+windows)	1	1,400.0	16000	0.084

Tabla 6. Costes horarios de software empleado

Código	Programa	Unidades	Precio (€)	Duración Licencia (años)	Horas de Funcionamiento (h)	Coste Horario (€/h)
(MS.1)	Matlab	1	800.0	1	2000	0.40
(MS.2)	CODESYS	1	0.0	-	-	0.00
(MS.3)	Microsoft Office	1	105.6	1	2000	0.05
(MS.4)	IGSS	1	0.0	-	-	0.00

3.2.2 Cuadro de precios descompuesto

A continuación, se detallan los costes asociados a cada una de las unidades de obra enunciadas con anterioridad:

Tabla 7. Unidad de obra. Programación Matlab

CÓDIGO	NOMBRE	UNIDADES	CANTIDAD	PRECIO	IMPORTE
U-01	**PROGRAMACIÓN MATLAB**				
M0.1	Ingeniero Industrial Junior	h	70	14,94 €	1.045,80 €
MH.1	Ordenador portátil	h	70	0,08 €	5,60 €
MS.1	Matlab	h	70	0,40 €	28,00 €
				TOTAL	1.079,40 €

Tabla 8. Unidad de obra. Diseño de control

CÓDIGO	NOMBRE	UNIDADES	CANTIDAD	PRECIO	IMPORTE
U-02	**DISEÑO DEL CONTROL**				
M0.1	Ingeniero Industrial Junior	h	40	14,94 €	597,60 €
MH.1	Ordenador portátil	h	40	0,08 €	3,20 €
MS.1	Matlab	h	40	0,40 €	16,00 €
MS.3	Microsoft Office	h	8	0,05 €	0,40 €
				TOTAL	617,20 €

Tabla 9. Unidad de obra. Implementación del control

CÓDIGO	NOMBRE	UNIDADES	CANTIDAD	PRECIO	IMPORTE
U-03	**IMPLEMENTACIÓN DEL CONTROL**				
M0.1	Ingeniero Industrial Junior	h	69	14,94 €	1.030,86 €
MH.1	Ordenador portátil	h	69	0,08 €	5,52 €
MS.1	Matlab	h	69	0,40 €	27,60 €
MS.2	CODESYS	h	24	0,00 €	0,00 €
				TOTAL	1.063,98 €

Tabla 10. Unidad de obra. Programación SCADA

CÓDIGO	NOMBRE	UNIDADES	CANTIDAD	PRECIO	IMPORTE
U-04	**PROGRAMACIÓN SCADA**				
M0.1	Ingeniero Industrial Junior	h	20	14,94 €	298,80 €
MH.1	Ordenador portátil	h	20	0,08 €	1,60 €
MS.4	IGSS	h	20	0,00 €	0,00 €
				TOTAL	300,40 €

Tabla 11. Unidad de obra. Elaboración del documento

CÓDIGO	NOMBRE	UNIDADES	CANTIDAD	PRECIO	IMPORTE
U-06	**ELABORACIÓN DEL DOCUMENTO**				
M0.1	Ingeniero Industrial Junior	h	45	14,94 €	672,30 €
MH.1	Ordenador portátil	h	45	0,08 €	3,60 €
MS.3	Microsoft Office	h	45	0,05 €	2,25 €
				TOTAL	678,15 €

Tabla 12. Unidad de obra. Pruebas

CÓDIGO	NOMBRE	UNIDADES	CANTIDAD	PRECIO	IMPORTE
U-05	**PRUEBAS**				
M0.1	Ingeniero Industrial Junior	h	44	14,94 €	657,36 €
MH.1	Ordenador portátil	h	44	0,08 €	3,52 €
MS.1	Matlab	h	30	0,40 €	12,00 €
MS.2	CODESYS	h	16	0,00 €	0,00 €
MS.4	IGSS	h	10	0,00 €	0,00 €
				TOTAL	672,88 €

3.3 Presupuesto final

3.3.1 Presupuesto de ejecución de material

Este presupuesto constituye el coste total de las unidades de obra, excluyendo gastos generales, beneficio industrial e impuestos.

Tabla 13. Presupuesto de ejecución de material (PEM)

CÓDIGO - UNIDAD DE OBRA	IMPORTE
U-01 PROGRAMACIÓN MATLAB	1.079,40 €
U-02 DISEÑO DEL CONTROL	617,20 €
U-03 IMPLEMENTACIÓN DEL CONTROL	1.063,98 €
U-04 PROGRAMACIÓN SCADA	300,40 €
U-05 PRUEBAS	672,88 €
U-06 ELABORACIÓN DEL DOCUMENTO	678,15 €
TOTAL	**4.412,01 €**

3.3.2 Presupuesto de contrata

Este presupuesto contempla además de lo considerado en el PEM, los gastos generales, y el beneficio industrial.

Tabla 14. Presupuesto de contrata

PRESUPUESTO DE EJECUCIÓN DE MATERIAL	4.412,01 €
GASTOS GENERALES	293,15 €
BENEFICIO INDUSTRIAL	197,20 €
PRESUPUESTO DE CONTRATA	**4.902,36 €**

3.3.3 Presupuesto total

Tabla 15. Presupuesto total

PRESUPUESTO DE CONTRATA	4.902,36 €
IVA (21%)	1.029,50 €
PRESUPUESTO TOTAL	**5.931,86 €**

Asciende el presupuesto del presente proyecto a la expresada cantidad de **CINCO MIL NOVECIENTOS TREINTA Y UNO EUROS con OCHENTA Y SEIS CÉNTIMOS.**

3.4 Bibliografía del presupuesto

- Días laborables calculadora en España | Islas Canarias (dias-laborables.es)
- Disposición 542 del BOE núm. 15 de 2017
- Tabla de amortización de inversiones para sociedades | Cuéntica (cuentica.com)
- HP OMEN 16-c0013ns AMD Ryzen 7 5800H/16GB/512GB SSD/RTX 3050Ti/16.1" | PcComponentes.com
- Monitor gaming - Samsung Odyssey LC27G55TQWRXEN, 27" WQHD, Curvo, 1 ms – MediaMarkt Canarias
- Teclado gaming - Logitech G815, Retroiluminado, Negro – MediaMarkt Canarias
- Logitech G305 LIGHTSPEED Ratón Gaming Inalámbrico, Captor HERO 12K, 12,000 DPI, Ultra-ligero, Batería de 250h, 6 Botones Programables, Memoría Integrada, PC/Mac - Negro : Logitech: Amazon.es: Videojuegos
- Comparar todos los planes de Microsoft 365 (anteriormente Office 365): Microsoft Store
- Pricing and Licensing - MATLAB & Simulink (mathworks.com)
- ¿Cuánto cuesta contratar un trabajador? - Infoautonomos

Escuela Técnica Superior de Ingeniería Industrial

ETSII

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Máster Universitario en Ingeniería Industrial (Acceso desde Grado I. Electrónica Industrial y Automática)

Desarrollo e implementación de un sistema de control industrial de un biorreactor para la producción de etanol

Trabajo Fin de Máster

4. DIAGRAMAS

Autor: Daniel Pérez Gutiérrez

Tutor académico: Francesc Xavier Blasco Ferragud

Julio 2023

CAPÍTULO 4. DIAGRAMAS

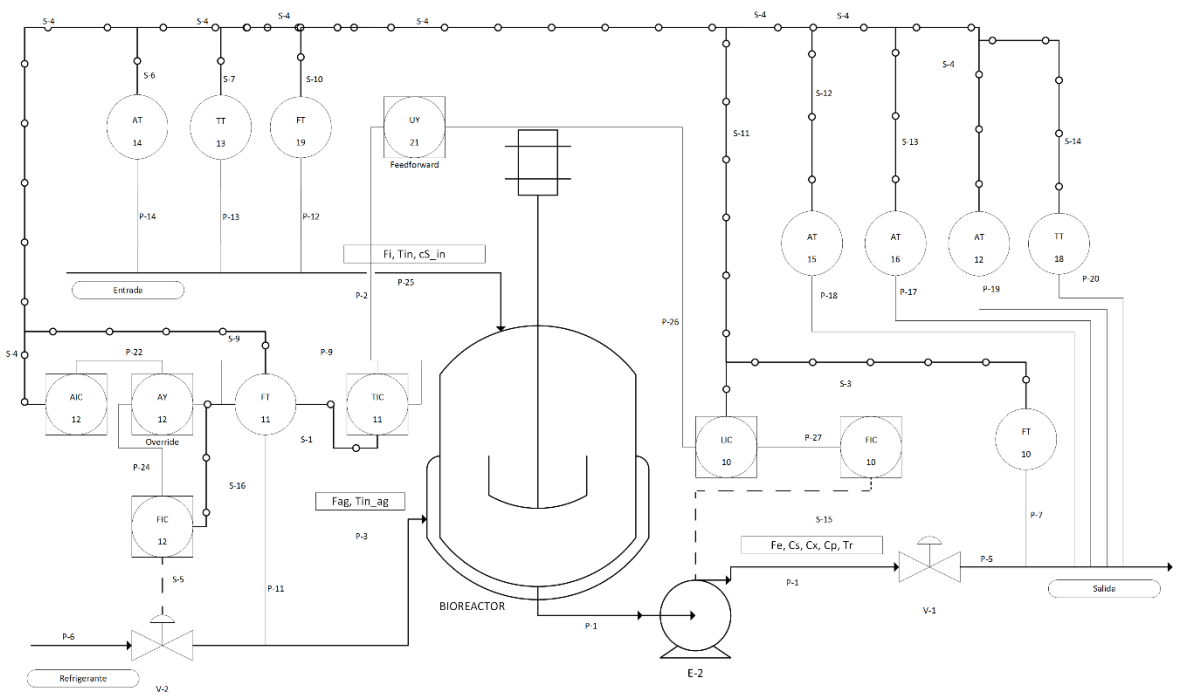

Diagrama 1. Diagrama P&ID

Escuela Técnica Superior de Ingeniería Industrial

ETSII

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Máster Universitario en Ingeniería Industrial (Acceso desde Grado I. Electrónica Industrial y Automática)

Desarrollo e implementación de un sistema de control industrial de un biorreactor para la producción de etanol

Trabajo Fin de Máster

5. CÓDIGO

Autor: Daniel Pérez Gutiérrez

Tutor académico: Francesc Xavier Blasco Ferragud

Julio 2023

CAPÍTULO 5. CÓDIGO

5.1 Matlab

5.1.1 Matriz GG

From input 1 to output...

1: 1/s

2: (0.000112 s^2 + 1.104e-05 s + 2.168e-08) / (0.1109 s^4 + 0.01801 s^3 + 0.001102 s^2 + 2.217e-05 s)

3: (0.0007038 s^2 + 0.0007331 s + 2.627e-06) / (s^4 + 0.1624 s^3 + 0.009938 s^2 + 0.0001999 s)

4: (-0.0001854 s - 7.229e-07) / (0.0005688 s^2 + 2.217e-05 s)

5: (-0.04844 s^2 - 0.002511 s - 2.287e-06) / (1.384 s^3 + 0.1245 s^2 + 0.002751 s)

6: (0.06179 s^3 + 0.006181 s^2 + 0.00016 s + 2.794e-07) / (0.2683 s^4 + 0.03529 s^3 + 0.001537 s^2 + 2.217e-05 s)

7: (0.05112 s^2 + 0.002513 s + 4.544e-06) / (0.2683 s^3 + 0.0216 s^2 + 0.0004345 s)

From input 2 to output...

1: 0

2: (-6.479e-05 s - 6.374e-06) / (2.174 s^3 + 0.353 s^2 + 0.0216 s + 0.0004345)

3: (-4.561e-05) / (2.174 s^3 + 0.353 s^2 + 0.0216 s + 0.0004345)

4: (0.0001154) / (2.174 s^3 + 0.353 s^2 + 0.0216 s + 0.0004345)

5: (0.01367 s + 0.00152) / (269.8 s^3 + 43.81 s^2 + 2.681 s + 0.05393)

6: (-0.01943 s^2 - 0.001913 s - 8.955e-05) / (2.174 s^3 + 0.353 s^2 + 0.0216 s + 0.0004345)

7: (-0.2435 s^3 - 0.05615 s^2 - 0.004046 s - 0.0001244) / (s^4 + 2.336 s^3 + 0.3629 s^2 + 0.0218 s + 0.0004345)

From input 3 to output...

```
   1:  0

   2:  (0.000358 s + 3.522e-05) / (2.174 s^3 + 0.353 s^2 + 0.0216 s +
0.0004345)

   3:  (0.0001198) / (s^3 + 0.1624 s^2 + 0.009938 s + 0.0001999)

   4:  (-0.0003029) / (s^3 + 0.1624 s^2 + 0.009938 s + 0.0001999)

   5:  (-0.07832 s^2 - 0.04461 s - 0.003992) / (124.1 s^3 + 20.15 s^2
+ 1.233 s + 0.0248)

   6:   (0.005023 s + 0.0002352) / (0.1234 s^2 + 0.009938 s +
0.0001999)

   7:  (0.008742 s + 0.0004093) / (0.1234 s^3 + 0.2782 s^2 + 0.0218 s
+ 0.0004345)
```

From input 4 to output...

```
   1:  0

   2:  (2.651e-06 s^2 + 4.763e-08 s - 7.809e-09) / (s^4 + 0.2134 s^3 +
0.01822 s^2 + 0.0007068 s + 1.02e-05)

   3:  (5.856e-05 s^2 + 5.618e-06 s + 6.314e-08) / (s^4 + 0.2134 s^3 +
0.01822 s^2 + 0.0007068 s + 1.02e-05)

   4:  0.05101 / (s + 0.05101)

   5:  (-0.0005749 s^2 - 1.151e-05 s + 1.748e-06) / (124.1 s^4 + 26.48
s^3 + 2.261 s^2 + 0.08772 s + 0.001265)

   6:  (-4.923e-07 s^2 + 9.849e-05 s - 4.532e-06) / (124.1 s^4 + 26.48
s^3 + 2.261 s^2 + 0.08772 s + 0.001265)

   7:  (0.0001713 s - 7.887e-06) / (269.8 s^4 + 57.57 s^3 + 4.916 s^2
+ 0.1907 s + 0.002751)
```

From input 5 to output...

```
   1:  0

   2:  (9.678e-05 s + 9.521e-06) / (2.174 s^3 + 0.353 s^2 + 0.0216 s +
0.0004345)

   3:  (6.814e-05) / (2.174 s^3 + 0.353 s^2 + 0.0216 s + 0.0004345)

   4:  (-0.0001723) / (2.174 s^3 + 0.353 s^2 + 0.0216 s + 0.0004345)

   5:  (-0.02042 s - 0.002271) / (2.174 s^4 + 270.1 s^3 + 43.83 s^2 +
2.682 s + 0.05393)

   6:  (0.02903 s^2 + 0.002857 s + 0.0001338) / (2.174 s^3 + 0.353 s^2
+ 0.0216 s + 0.0004345)
```

```
   7:  (0.3638 s^3 + 0.08387 s^2 + 0.006044 s + 0.0001858) / (s^4 +
2.336 s^3 + 0.3629 s^2 + 0.0218 s + 0.0004345)
```

From input 6 to output...

```
   1:  (-1) / s

   2:  (-0.0009682 s^2 - 9.826e-05 s - 1.895e-07) / (s^4 + 0.1624 s^3
+ 0.009938 s^2 + 0.0001999 s)

   3:  (-0.01251 s^3 - 0.002152 s^2 - 0.0007937 s - 2.704e-06) / (s^4
+ 0.1624 s^3 + 0.009938 s^2 + 0.0001999 s)

   4:  (0.001734 s + 6.451e-06) / (s^4 + 0.1624 s^3 + 0.009938 s^2 +
0.0001999 s)

   5:  (0.2085 s^2 + 0.02352 s + 2.121e-05) / (124.1 s^4 + 20.15 s^3 +
1.233 s^2 + 0.0248 s)

   6:  (-0.3028 s^3 - 0.03065 s^2 - 0.001467 s - 2.612e-06) / (s^4 +
0.1624 s^3 + 0.009938 s^2 + 0.0001999 s)

   7:  (-0.527 s^3 - 0.05335 s^2 - 0.002554 s - 4.546e-06) / (2.174 s^4 + 0.353 s^3 +
0.0216 s^2 + 0.0004345 s)


These are the transfer functions for each input-output combination in
the given matrix of transfer functions.

Continuous-time transfer function.
```

5.1.2 yeast_reactor_RUN.m

```
%% RUN REACTOR
yeast_reactor_params;
yeast_reactor_ValoresIni;
%% TIMER EJECUCIÓN
tt=timer;
tt.StartDelay=0;
tt.Period=Ts;
tt.TimerFcn="yeast_reactor_modelo";
tt.ExecutionMode='fixedRate';
start(tt)
%stop(tt)
```

5.1.3 yeast_reactor_ValoresIni.m

```
%% Periodo muestreo (ud tiempo)
Ts=0.02;
Tiempo=0;

%% Valor inicial variables estado
V  = 1000; %l
Cx   =  0.9047; %g/l
Cs   = 29.7389;  %g/l
Cp   = 12.5152; %g/l
```

```
Co2  =  3.1069; %mg/l
Tr   = 29.57; % ºC
Tag  = 27.05; % ºC

%% Entradas del sistema
Fi      = 51; %l/h
Fag     = 18; % l/h
Tin     = 25; %◦C
cS_in   = 60; %g/l
Tin_ag  = 15; %◦C
Fe      = Fi ; %l h-1

%% Mensaje con valores iniciales por pantalla
fprintf("---------------VALORES INICIALES---------------\n")
fprintf("Tiempo: %.5f [s]\n",Tiempo)
fprintf("Variables manipuladas\n")
fprintf("  Fi: %.2f [l/h]; Fag:  %.2f [l/h]; Fe:  %.2f [l/h]\n",Fi,Fag,Fe)
fprintf("  Tin: %.2f [oC]; Tin_ag:  %.2f [oC]\n",Tin,Tin_ag)
fprintf("  Cs_in: %.3f [g/l]\n",cS_in)
fprintf("Variables estado:\n")
fprintf("  V: %0.1f [l]\n",V)
fprintf("  Cx: %.3f [g/l]; Cp: %.3f [g/l]; Cs: %.3f [g/l]; Co2: %.3f
[mg/l]\n", Cx,Cp,Cs,Co2)
fprintf("  Tr: %.2f [oC]; Tag: %.2f [oC]\n",Tr,Tag)
```

5.1.4 yeast_reactor_params.m

```
%% PARÁMETROS DEL PROCESO
A1   = 9.5e8;
A2   = 2.55e33;
At   = 1; %m2
Cheat_ag = 4.18; %J g-1 K-1
Cheat_r  = 4.18; %J g-1 K-1
Ea1 = 55000; %J/mol
Ea2 = 220000; %J/mol
Hna  = -0.550;
Hca  = -0.303;
Hmg  = -0.314;
Hh   = -0.774;
Hcl  =  0.844;
Hco3 =  0.485;
Hoh  =  0.941;
kla0 = 38; %l h-1
Ko2  = 8.86; %mg/l
Kp   = 0.139; %g/l
Kp1  = 0.070; %g/l
Ks   = 1.030; %g/l
Ks1  = 1.680; %g/l
Kt   = 100*3600;%3.6e5; %J h-1 m-2 K-1
Rsp  = 0.435;
Rsx  = 0.607;
Vj   = 50; %l
Yo2  = 0.970; %mg/mg
Hr   = 518;% en la tabla pone 518 kJ/mol O2
NUo2 = 0.5; % h-1
NUp  = 1.790; % h-1
Roag = 1000; %g/l
Ror  = 1080; %g/l
R    = 8.31;%J (mol K)^-1
```

5.1.5 yeast_reactor_modelo.m

```
%% CÁLCULO REACTOR
SumaHiTi = 127.3899/V; %Sumatorio de las masas molares

Co2_0 = 14.6 - 0.3943 * Tr + 0.007714 * Tr^2 - 0.0000646 * Tr^3;
Co2A  = Co2_0 * 10^(-SumaHiTi); %Co2 asterisco
kla   = kla0 * 1.024^(Tr-20);
Ro2   = NUo2 * (1/Yo2) * Cx * (Co2/(Ko2 + Co2))*1000; %ratio de consumo de
oxígeno OJO: mg/l/h
NUx   = A1 * exp(-(Ea1/(R*(Tr+273)))) - A2 * exp(-(Ea2/(R*(Tr+273))));

%% Balances masas
dV = Fi - Fe;
dCx = NUx * Cx * Cs/(Ks + Cs)  * exp(-Cp*Kp)  - (Fe/V) * Cx;
dCp = NUp * Cx * Cs/(Ks1 + Cs) * exp(-Cp*Kp1) - (Fe/V) * Cp;
dCs = - (1/Rsx) * NUx * Cx * Cs/(Ks  + Cs) * exp(-Cp*Kp) ...
      - (1/Rsp) * NUp * Cx * Cs/(Ks1 + Cs) * exp(-Cp*Kp1) ...
      + (Fi/V)  * cS_in - (Fe/V) * Cs;
dCo2 = kla * (Co2A - Co2 ) - Ro2 -(Fe/V)*Co2;

%% Balances energia
dTr = (Fi/V) * (Tin + 273) - (Fe/V) * (Tr + 273) +
((Ro2*Hr)/(32*Ror*Cheat_r)) ...
      - ((Kt*At*(Tr - Tag))/(V*Ror*Cheat_r));
dTag = (Fag/Vj) * (Tin_ag - Tag) + ((Kt*At*(Tr - Tag))/(Vj*Roag*Cheat_ag));


%% Cálculo salidas
V      = dV  * Ts + V;
Cx     = dCx * Ts + Cx;
Cp     = dCp * Ts + Cp;
Cs     = dCs * Ts + Cs;
Co2    = dCo2 * Ts + Co2;
Tr     = dTr * Ts + Tr;
Tag    = dTag * Ts + Tag;

%% Saturación
if V<10
    V=10;
end
if V>1500 %depósito de 1500l
    V=1500;
end
if Cx<0
    Cx=0;
end
if Cp<0
    Cp=0;
end
if Cs<0
    Cs=0;
end
if Co2<0
    Co2=0;
end
Tiempo=Tiempo+Ts;
```

5.1.6 yeast_reactor_escalado.m

```
%%  ESCALADO A DIGITAL
% Concentraciones [0 100] -> [0 30000]
% Temperaturas [-55 200] -> [0 30000]
% Caudales [0 200] -> [0 30000]
% Volumen [0 3000] -> [0 30000]

V_dig = round( 30000/3000*V);
Cx_dig = round( 30000/100*Cx);
Cs_dig = round( 30000/100*Cs);
Cp_dig = round( 30000/100*Cp);
Co2_dig = round( 30000/100*Co2);
Tr_dig = round(( 30000/255)*(Tr+55));
Tag_dig = round(( 30000/255)*(Tag+55));

%% ESCRITURA DE VALORES DIGITALES EN MODBUS
write(mm,'holdingregs',1,[V_dig Cx_dig Cs_dig Cp_dig Co2_dig Tr_dig
Tag_dig],2,'uint16');

%% LECTURA DE VALORES DIGITALES DE MODBUS
vector_read = read(mm,'inputregs',1,6,2,'int16');
Fi_dig = vector_read(1);
Fag_dig = vector_read(2);
Tin_dig = vector_read(3);
cS_in_dig = vector_read(4);
Tin_ag_dig = vector_read(5);
Fe_dig = vector_read(6);

%% ESCALADO A ANALÓGICO
Fi = Fi_dig/( 30000/200);
Fag = Fag_dig/( 30000/200);
Tin = ((255/ 30000)*Tin_dig-55);
cS_in = cS_in_dig/( 30000/100);
Tin_ag = ((255/ 30000)*Tin_ag_dig-55);
Fe = Fe_dig/( 30000/200);
```

5.1.7 yeast_reactor_MODBUS.m

```
%% APERTURA DE CONEXIÓN MODBUS A CODESYS
mm = modbus('tcpip','192.168.1.58',502);
%% TIMER EJECUCIÓN MODBUS
tt2=timer;
tt2.StartDelay=0;
tt2.ExecutionMode='fixedRate';
tt2.Period=0.5;
tt2.TimerFcn="yeast_reactor_escalado";
start(tt2)
%stop(tt2)
```

5.1.8 yeast_reactor_PERT

```
%% BUCLE PERTURBACIONES
tt3=timer;
tt3.StartDelay=0;
tt3.Period=1;
tt3.TimerFcn="yeast_reactor_perturbaciones";
```

```
tt3.ExecutionMode='fixedRate';
start(tt3)
```

5.1.9 yeast_reactor_perturbaciones

```
%% PERTURBACIONES
% cálculo de número rándom para perturbaciones
Pert_V = rand() * 0.1 - 0.05;
Pert_Fi = rand() * 0.1 - 0.05;
Pert_cS_in = rand() * 0.05 - 0.025;
Pert_Tin = rand() * 0.5 - 0.25;
Pert_Tin_ag = rand() * 0.5 - 0.25;

% sumar la perturbación a la variable
V = V + Pert_V;
Fi = Fi + Pert_Fi;
cS_in = cS_in + Pert_cS_in;
Tin = Tin + Pert_Tin;
Tin_ag = Tin_ag * Pert_Tin_ag;
```

5.1.10 pid2isa.m

```
function [Kc,Ti,Td]=pid2isa(g)
% [Kc,Ti,Td]=pid2isa(g)

[n,d]=tfdata(g,'v');

if length(n)==1
    Kc=n/d;
    Ti=0;
    Td=0;
end
if length(n)==2
    if (d(1)==1) && (d(2)==0)
    % PI
    Kc=n(1)/d(1);
    Ti=n(1)/n(2);
    Td=0;
    elseif (d(1)==0) && (d(2)==1)
    % PD
    Kc=n(2)/d(2);
    Ti=0;
    Td=n(1)/n(2);
    else
    % NO es pid
        error('No es tipo PID')
    end
end

if length(n)==3
    if (d(1)||d(3))~=0
        % No es pid
        error('No es tipo PID')
    else
        %PID
        Kc=n(2)/d(2);
        Ti=n(2)/n(3);
        Td=n(1)/n(2);
```

```
    end
end
```

5.2 Simulink

5.2.1 Matlab Function

```
function y = fcn(u,Ts)

%DATOS INICIALES DEL SISTEMA
% entradas u(1) ... u(6)
Fi     = u(1);
Fag    = u(2);
Tin    = u(3);
cS_in  = u(4);
Tin_ag = u(5);
Fe     = u(6);
% estados u(79 ... u(13)
V      = u(7);
Cx     = u(8);
Cp     = u(9);
Cs     = u(10);
Co2    = u(11);
Tr     = u(12);
Tag    = u(13);

% sales inorgánicas en el medio de reacción
mNaCl  = 500; %g
mCaCO3 = 100; %g
mMgCl2 = 100; %g

% masas molares de los compuestos químicos en g/mol
MNaCl  =  58.5;%58.44;
MNa    =  23;%22.99;
MCaCO3 =  90;%100.09;
MCa    =  40;%40.08;
MMgCl2 =  95;%95.21;
MMg    =  24;%24.30;
MCl    =  35.5;%35.45;
MCO3   =  60;%60.01;

pH = 6; % pH de la fase líquida


%Parámetros del proceso
A1   = 9.5e8;
A2   = 2.55e33;
At   = 1; %m2
Cheat_ag = 4.18; %J g-1 K-1
Cheat_r  = 4.18; %J g-1 K-1
Ea1 = 55000; %J/mol
Ea2 = 220000; %J/mol
Hna  = -0.550;
Hca  = -0.303;
Hmg  = -0.314;
Hh   = -0.774;
Hcl  =  0.844;
Hco3 =  0.485;
```

```
Hoh  =  0.941;
kla0 = 38; %l h-1
Ko2  = 8.86; %mg/l
Kp   = 0.139; %g/l
Kp1  = 0.070; %g/l
Ks   = 1.030; %g/l
Ks1  = 1.680; %g/l
Kt   = 100*3600;%3.6e5; %J h-1 m-2 K-1
Rsp  = 0.435;
Rsx  = 0.607;
Vj   = 50; %l
Yo2  = 0.970; %mg/mg
Hr   = 518;%
NUo2 = 0.5; % h-1
NUp  = 1.790; % h-1
Roag = 1000; %g/l
Ror  = 1080; %g/l
R    = 8.31;%J (mol K)^-1


%---------------------%

%Concentraciones molares
cNa  = (mNaCl/MNaCl)   * (MNa/V);
cCa  = (mCaCO3/MCaCO3) * (MCa/V);
cMg  = (mMgCl2/MMgCl2) * (MMg/V);
cCl  = ((mNaCl/MNaCl) + 2 * (mMgCl2/MMgCl2)) * (MCl/V);
cCO3 = (mCaCO3/MCaCO3) * (MCO3/V);
cH   = 10^(-pH);
cOH  = 10^(-(14-pH));

%Fuerza iónica
INa  = 0.5 * cNa * (1)^2;
ICa  = 0.5 * cCa * (2)^2;
IMg  = 0.5 * cMg * (2)^2;
ICl  = 0.5 * cCl * (-1)^2;
ICO3 = 0.5 * cCO3 * (-2)^2;
IH   = 0.5 * cH * (1)^2;
IOH  = 0.5 * cOH * (-1)^2;

SumaHiIi = Hna * INa + Hca * ICa + Hmg * IMg + Hcl * ICl + Hco3 * ICO3 + Hh *
IH + Hoh * IOH;
%SumaHiIi = 0.1274;
%Co2_0 = 14.6 - 0.3943 * Tr + 0.007714 * Tr^2 - 0.0000646 * Tr^3;
% formula Truesdale & Downing, Nature, 173, 1236 (1954).
Co2_0 = 14.16 - 0.3943 * Tr + 0.007714 * Tr^2 - 0.0000646 * Tr^3; % ppm =
mg/l
Co2A  = Co2_0 * 10^(-SumaHiIi); %Co2 asterisco
kla   = kla0 * 1.024^(Tr-20);
Ro2   = NUo2 * (1/Yo2) * Cx * (Co2/(Ko2 + Co2))*1000; %ratio de consumo de
oxígeno OJO: mg/l/h
NUx   = A1 * exp(-(Ea1/(R*(Tr+273)))) - A2 * exp(-(Ea2/(R*(Tr+273))));

% balances masas
dV = Fi - Fe;
dCx = NUx * Cx * Cs/(Ks + Cs)  * exp(-Cp*Kp)  - (Fe/V) * Cx;
dCp = NUp * Cx * Cs/(Ks1 + Cs) * exp(-Cp*Kp1) - (Fe/V) * Cp;
dCs = - (1/Rsx) * NUx * Cx * Cs/(Ks  + Cs) * exp(-Cp*Kp) ...
      - (1/Rsp) * NUp * Cx * Cs/(Ks1 + Cs) * exp(-Cp*Kp1) ...
```

```
      + (Fi/V)  * cS_in - (Fe/V) * Cs;
dCo2 = kla * (Co2A - Co2 ) - Ro2 -(Fe/V)*Co2;
% balances energia
dTr = (Fi/V) * (Tin + 273) - (Fe/V) * (Tr + 273) +
((Ro2*Hr)/(32*Ror*Cheat_r)) ...
      - ((Kt*At*(Tr - Tag))/(V*Ror*Cheat_r));
dTag = (Fag/Vj) * (Tin_ag - Tag) + ((Kt*At*(Tr - Tag))/(Vj*Roag*Cheat_ag));
%% Cálculo salidas // POR MÉTODO DE INTEGRACIÓN DE EULER
V      = dV  *Ts+V;
Cx     = dCx *Ts+Cx;
Cp     = dCp *Ts+Cp;
Cs     = dCs *Ts+Cs;
Co2    = dCo2*Ts+Co2;
Tr     = dTr *Ts+Tr;
Tag    = dTag*Ts+Tag;

%% satura
if V<10
    V=10;
end
if V>1500 %depósito de 1500l
    V=1500;
end
if Cx<0
    Cx=0;
end
if Cp<0
    Cp=0;
end
if Cs<0
    Cs=0;
end
if Co2<0
    Co2=0;
end

y = [V Cx Cp Cs Co2 Tr Tag]';
```

5.2.2 PID_cpoh

```
function [uk,val_ant] = PID_cpoh(in,param,val_ant)
% Control con PID DIGITAL

% Lee setpoint, variable controlada y offset  ........................
rk=in(1);
yk=in(2);
if length(in)==2
    offset = 0;
else
    offset = in(3); % offset, incluye el efecto de los puntos de
                    % funcionamiento y/o prealimentaciones
end
% Carga parametros   ..............................................
% Periodo de muestreo...
Ts = param(1);

%Límites de la acción de control (saturación).
umax = param(2);
umin = param(3);
```

```
% Parámetros PID (Estándar ISA)...
Kc = param(4); % Constante proporcional
Ti = param(5); % Constante integral  (Ti=0 desconecta acción integral).
Td = param(6); % Constante derivada  (Td=0 desconecta acción derivada).
N  = param(7); % Filtro derivada (N=0 desconecta filtrdo de la derivada).
b  = param(8); % Ponderación de la ref en la acción Proporcional
c  = param(9); % Ponderación de la ref en la acción derivada
tauf = param(10); % Constante de tiempo del filtro de la var manipulada
                  % (tauf=0 desconecta el filtro).

% Valores anteriores ..............................................
uik1 = val_ant(1);
ekf1 = val_ant(2);
ykf1 = val_ant(3);

% Variables auxiliares  ki, kd, kf, kn ...............................
if Ts>0
    if Ti~=0
        ki = (Kc*Ts)/Ti;
    else
        ki = 0; % Ti=0 desconecta accion integral
    end
    kd = (Kc*Td)/Ts; % Td=0 desconecta accion derivada
    kf = tauf/Ts; % tauf=0 desconecta filtro de la variable controlada
    if N~=0
        kn = Td/(N*Ts);
    else
        kn = 0; % N=0 desconecta el filtro de la accion derivada
    end
else % si el periodo de muestreo no es superior a cero no hace nada
    ki = 0;
    kd = 0;
    kf = 0;
    kn = 0;
end

% Controlador PID digital  ........................................
% Filtrado de la medida
ykf=(kf*ykf1+yk)/(kf+1);

% Término proporcional con ponderacion de la referencia...
upk=Kc*(b*rk-ykf);

% Término derivada con ponderacion de la referencia y filtrado...
ek=c*rk-ykf;
ekf=(kn*ekf1+ek)/(kn+1);
udk=kd*(ekf-ekf1);

% Término integral...
uik=uik1+ki*(rk-ykf);

% Acción PID + offset
uk = upk+uik+udk+offset;

% Antiwindup...
if (uk < umin)
    uik=uik1;
    uk=umin;
```

```
end
if (uk > umax)
    uik=uik1;
    uk=umax;
end

% Actualiza estados
val_ant(1) = uik;
val_ant(2) = ekf;
val_ant(3) = ykf;

% Fin Controlador PID digital ......................................
```

5.2.3 Bloques

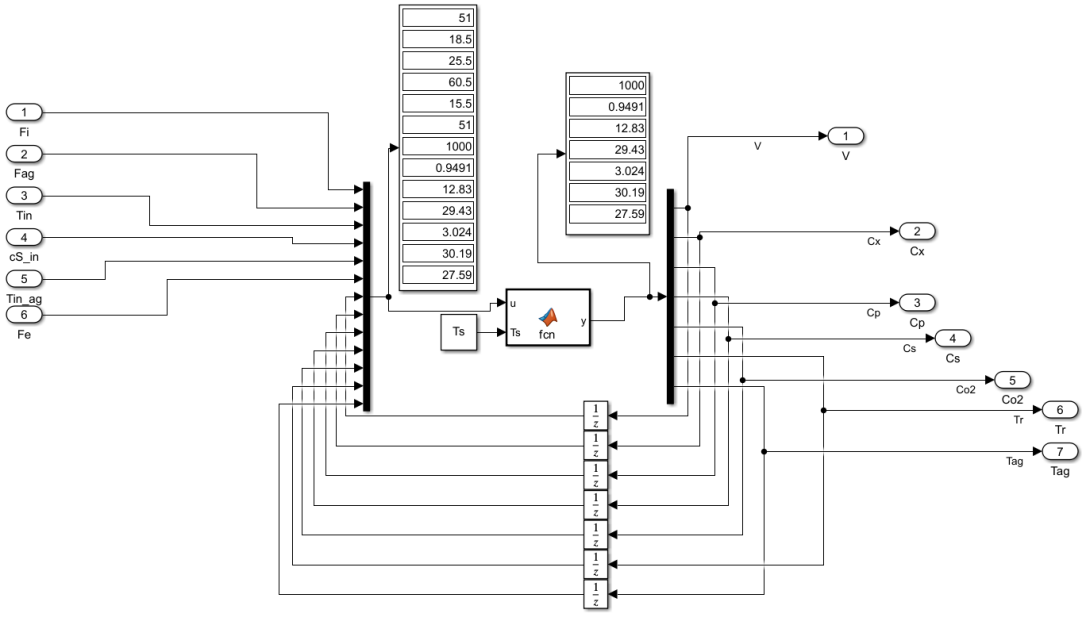

Figura 66. Bloque modelo no lineal

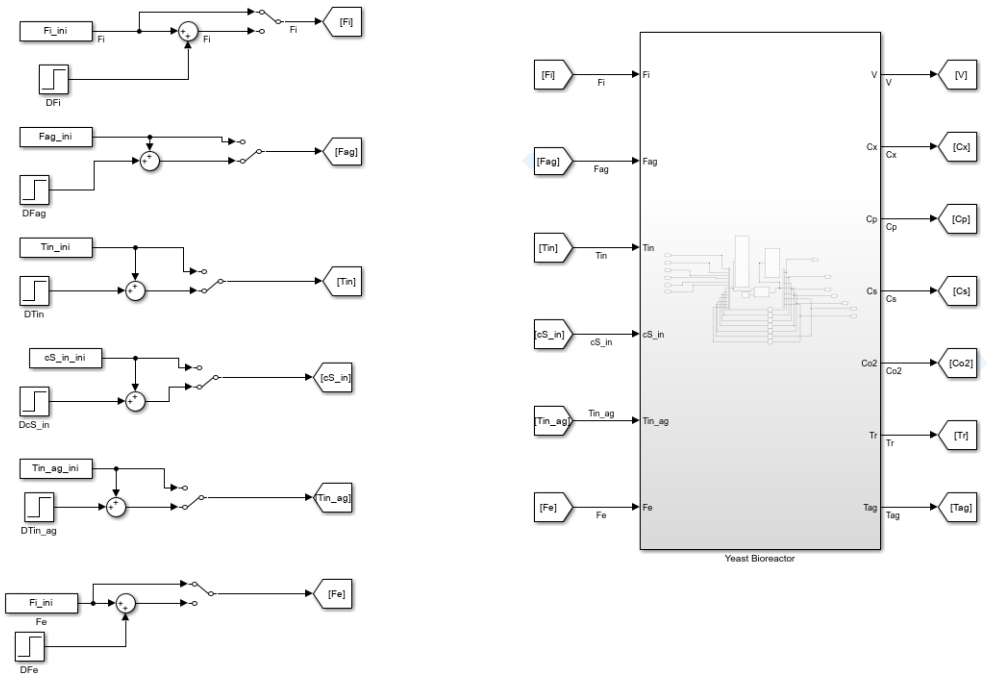

Figura 67. Modelo no lineal

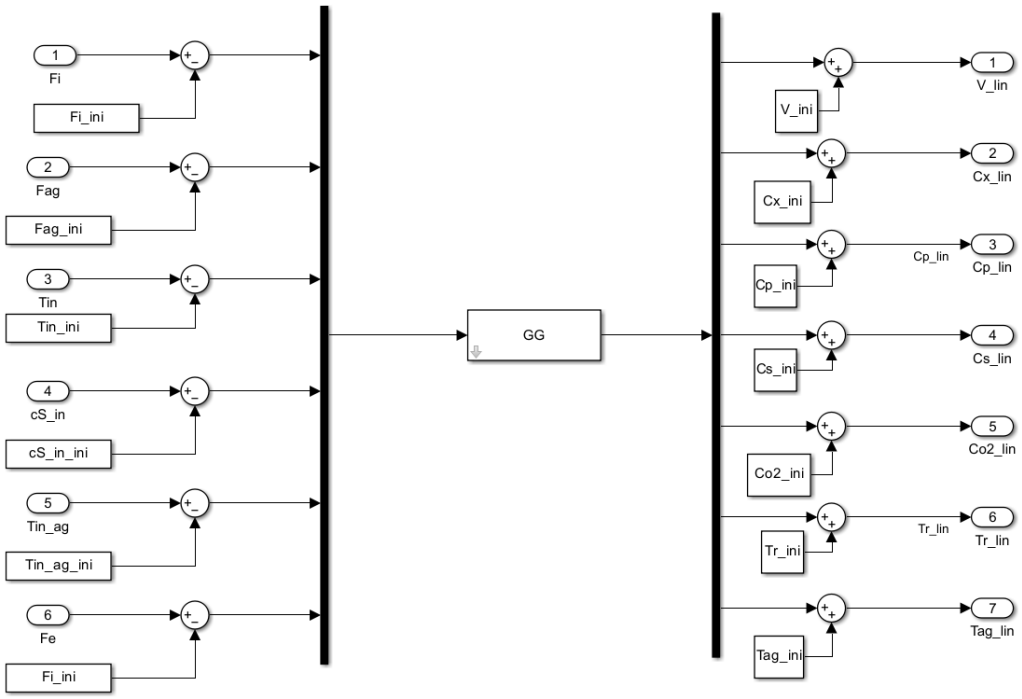

Figura 68. Bloque modelo linealizado

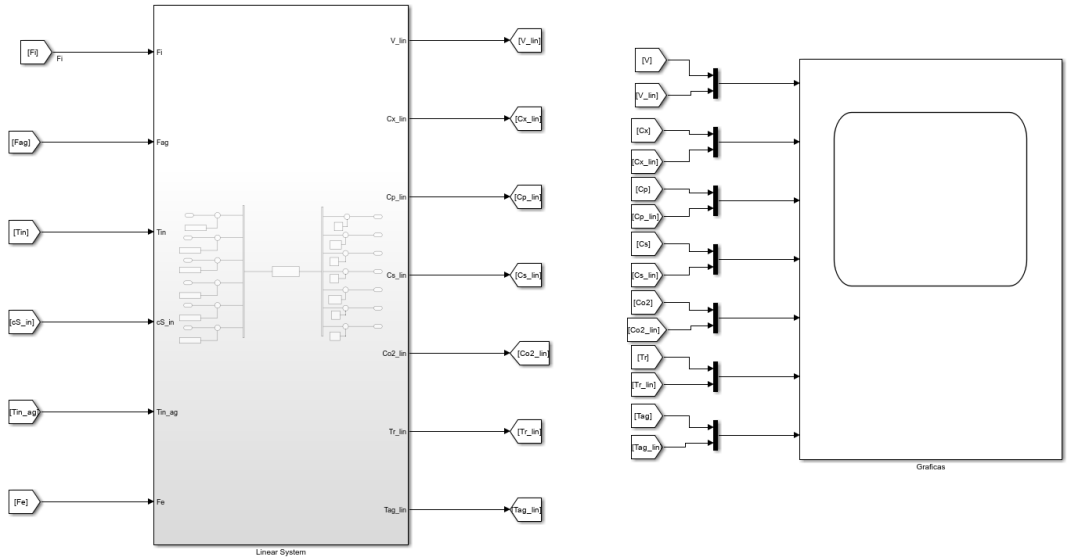

Figura 69. Modelo linealizado y gráficas comparativas

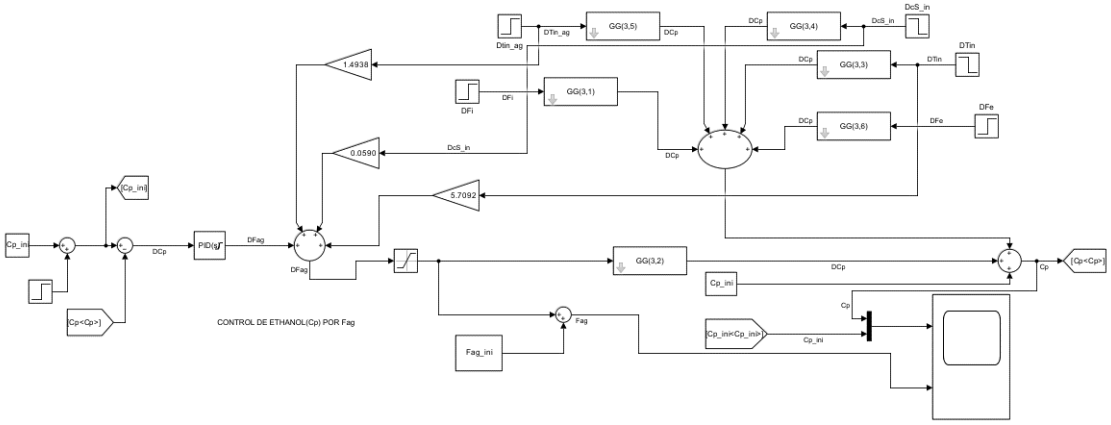

Figura 70. Control PID Cp - Fag

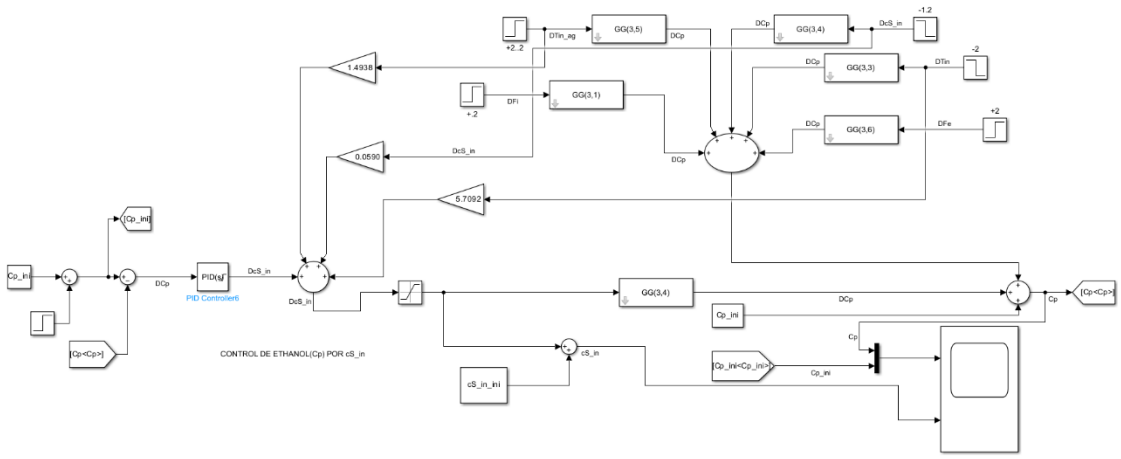

Figura 71. Control PID Cp - cS_in

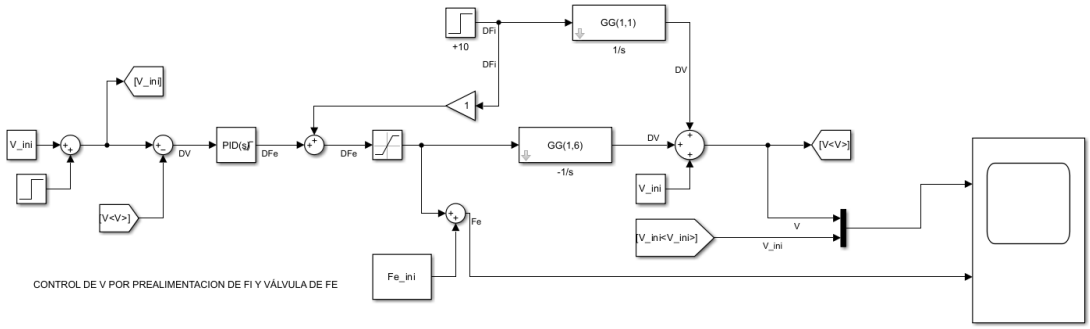

Figura 72. Control PID V – Fe

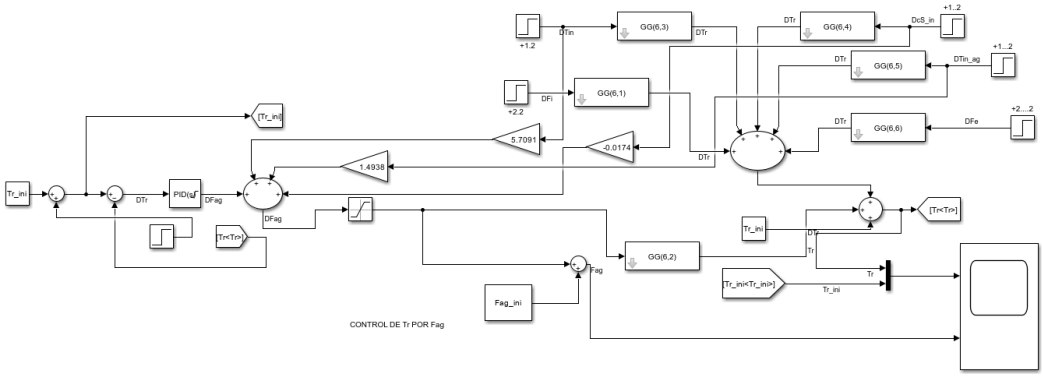

Figura 73. Control PID Tr – Fag

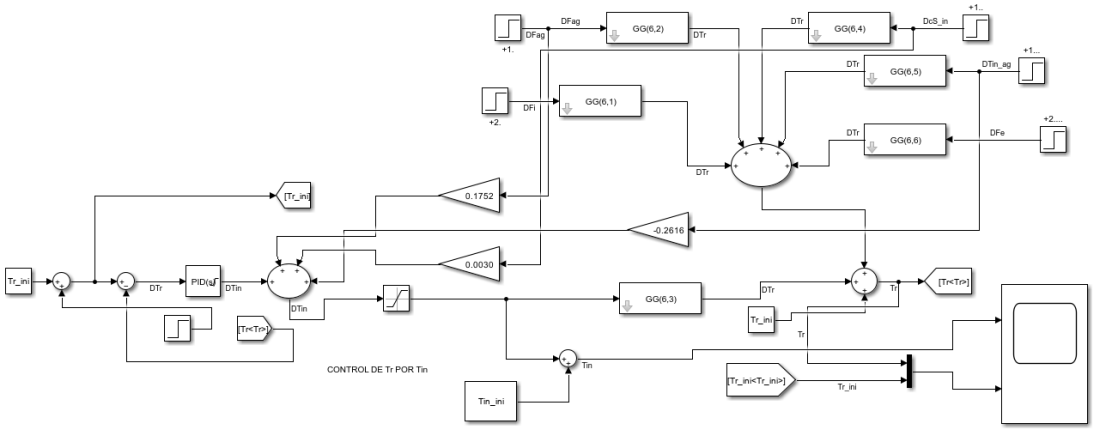

Figura 74. Control PID Tr - Tin

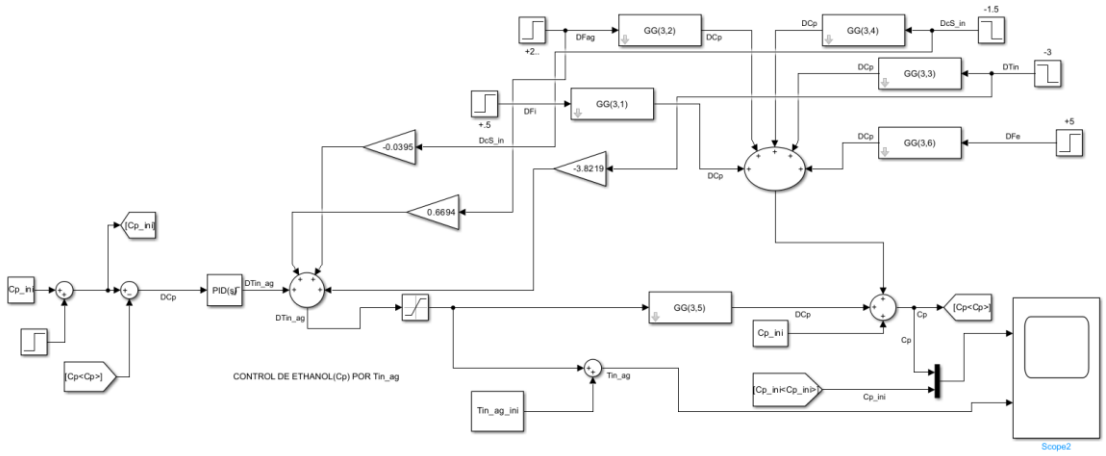

Figura 75. Control PID Cp - Tin_ag

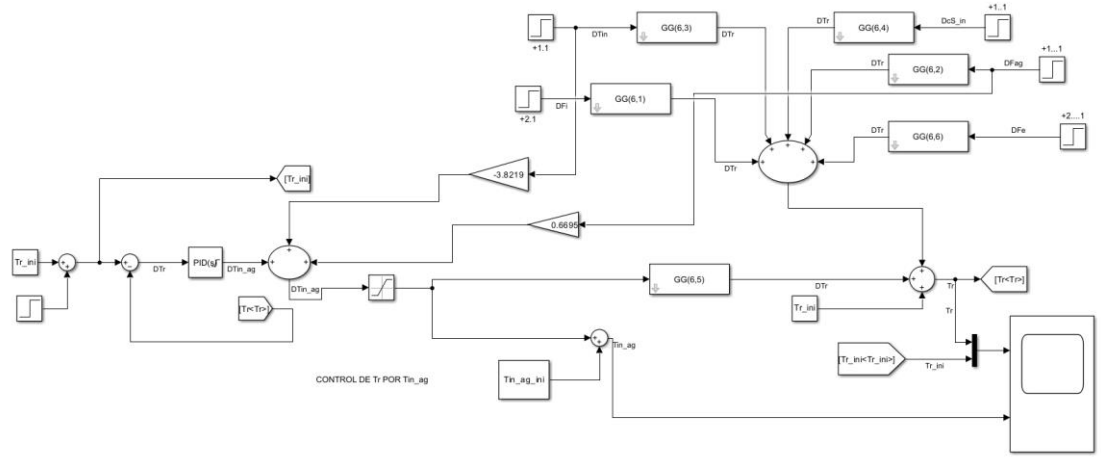

Figura 76. Control PID Tr - Tin_ag

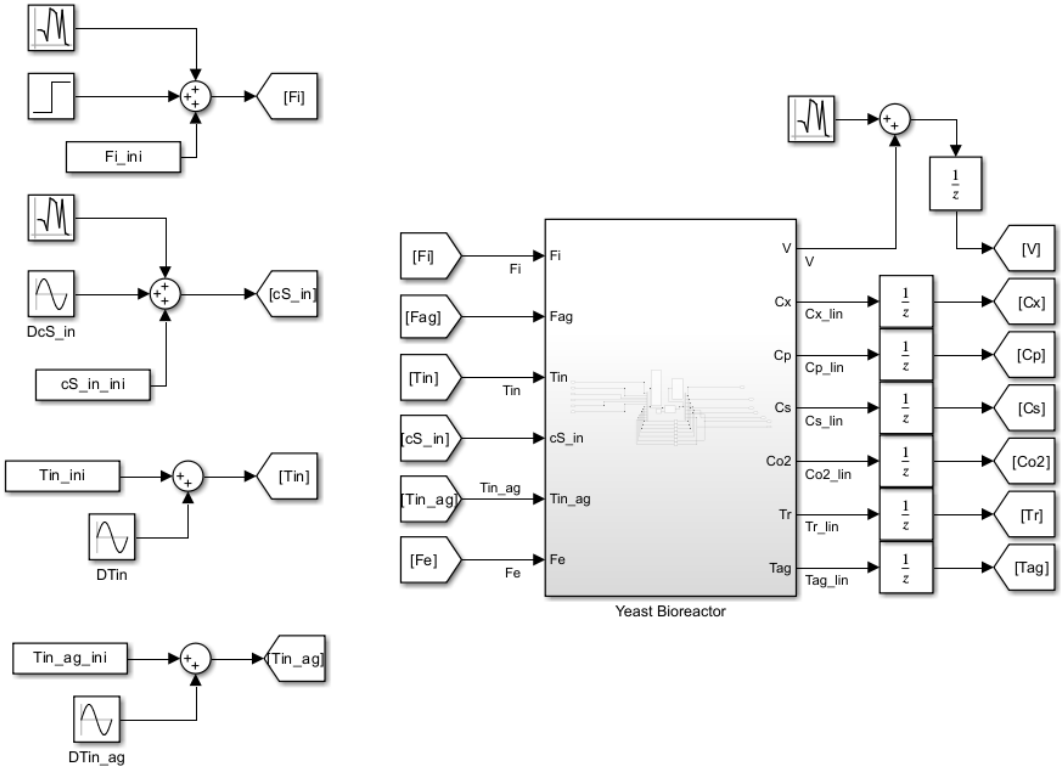

Figura 77. Modelo no lineal con perturbaciones

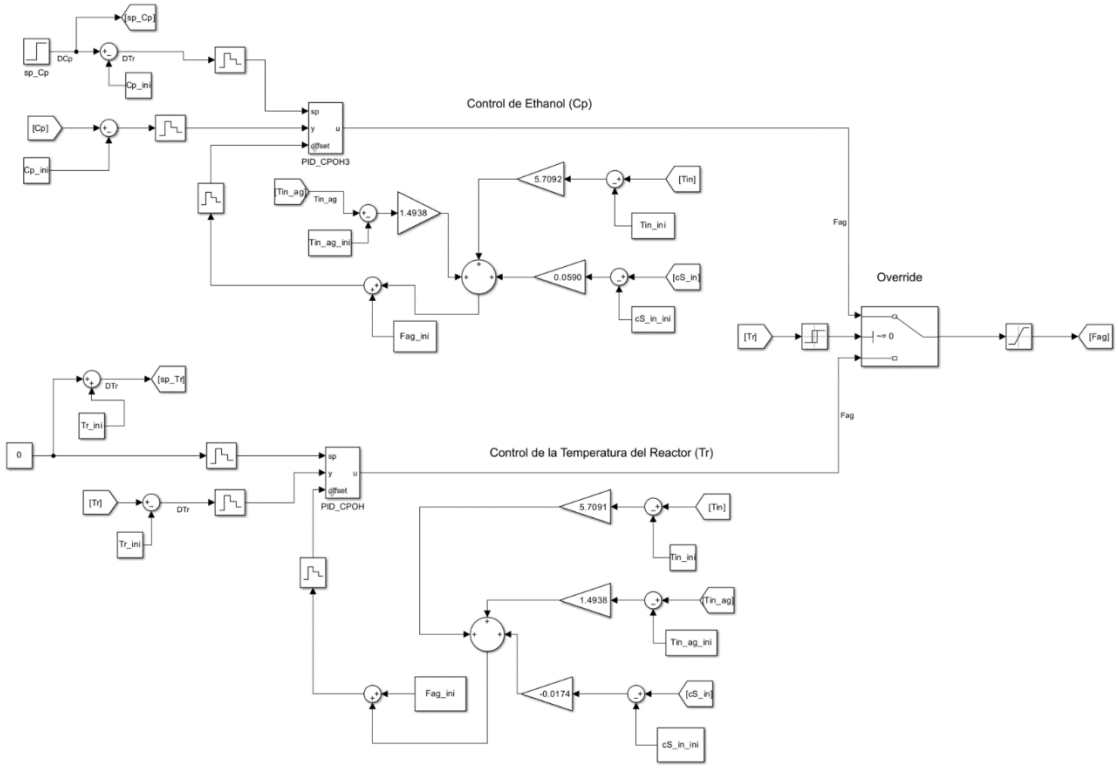

Figura 78. Bucle de control Cp y Tr sobre modelo no lineal

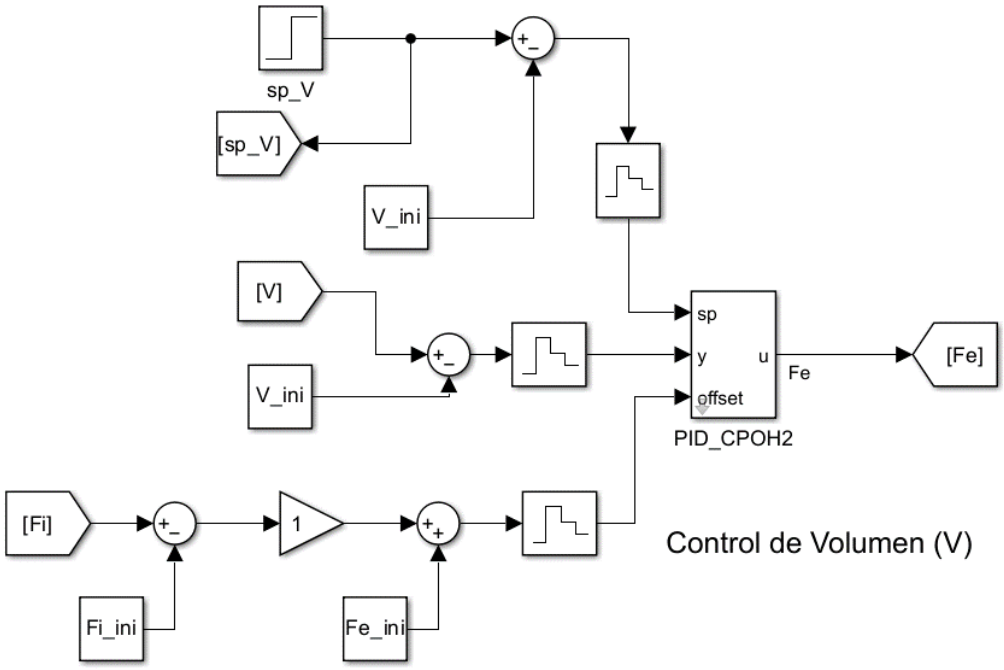

Figura 79. Bucle de control V sobre modelo no lineal

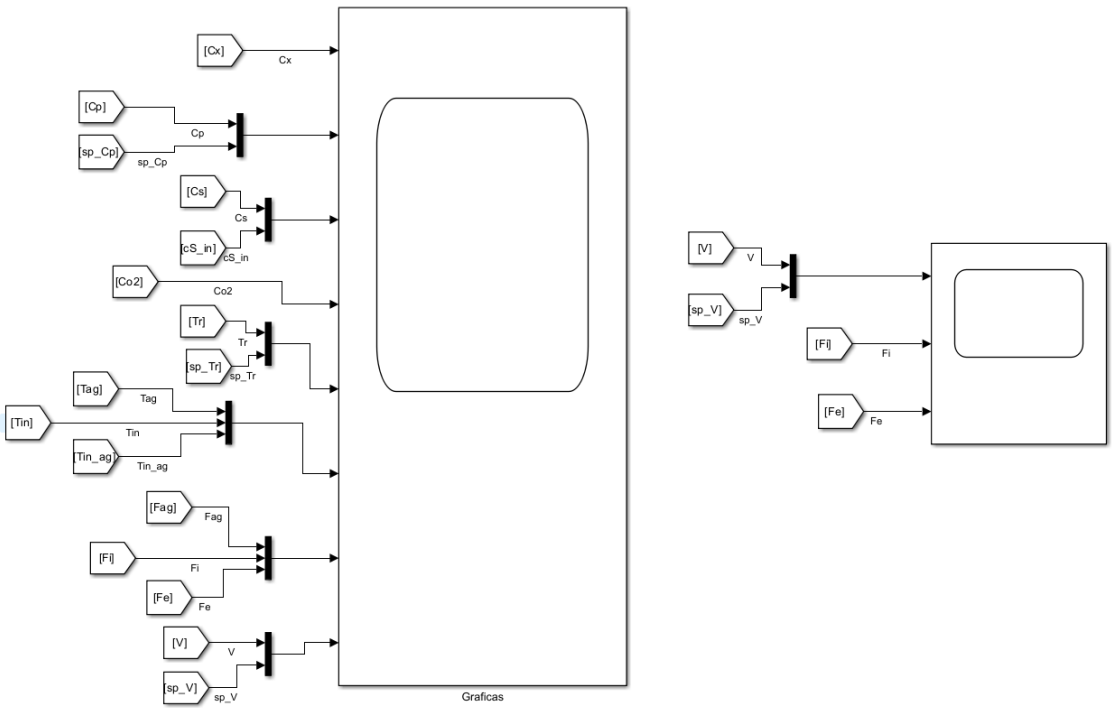

Figura 80. Gráficas modelo no lineal

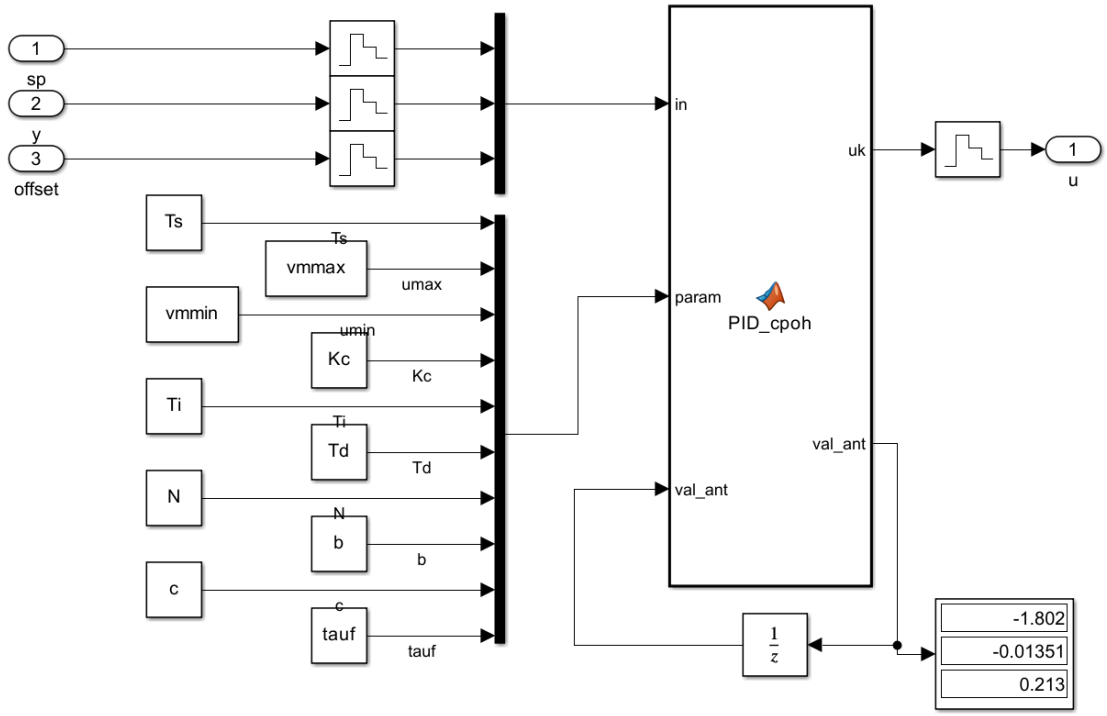

Figura 81. PID_cpoh

5.3 CODESYS

5.3.1 PLC_PRG

```
PROGRAM PLC_PRG


VAR_INPUT
	//ENTRADAS DEL SISTEMA
	Fi : LREAL := 51.0;
	Fi_ini : LREAL := 51.0;
	Fag_ini : LREAL := 18.0;
	Fag : LREAL := 18.0;
	Tin : LREAL := 25.0;
	Tin_ini : LREAL := 25.0;
	cS_in : LREAL := 60.0;
	cS_in_ini : LREAL := 60.0;
	Tin_ag : LREAL := 15;
```

```
	Tin_ag_ini : LREAL := 15.0;
	Fe : LREAL := 51.0;
	Fe_ini : LREAL := 51.0;

END_VAR

VAR
	//VARIABLES DE ESTADO
	V : LREAL := 1000.0;
	Cx : LREAL := 0.9047;
	Cs : LREAL := 29.7389;
	Cp : LREAL := 12.5152;
	Co2 : LREAL := 3.1069;
	Tr : LREAL := 29.57;
	Tag : LREAL := 27.05;

	//PARAMETROS DE CONTROL
	//VOLUMEN
	sp_V : LREAL := 1000.0;
	offsetV : LREAL;
	Gff_Fi_V : LREAL := 1;

	//CONCENTRACIÓN DE ETHANOL
	sp_Cp : LREAL := 12.5152;
	offsetCp : LREAL;
	Gff_Tin_ag_Cp : LREAL := 1.4938;
	Gff_Tin_Cp : LREAL := 5.7092;
	Gff_cS_in_Cp : LREAL := 0.0590;


	// TEMPERATURA DEL REACTOR
	sp_Tr : LREAL := 29.57;
	offsetTr : LREAL;
	Gff_Tin_ag_Tr : LREAL := 1.4938;
```

```
      Gff_Tin_Tr : LREAL := 5.7091;
      Gff_cS_in_Tr : LREAL := -0.0174;

      //OVERRIDE
      override : BOOL := FALSE;
      referencia : LREAL := 29.57; //Tr_ini como punto de referencia
      hysteresis1 : INT := 3;
      hysteresis2 : INT := 0;
      manual_override : BOOL := FALSE;

      //CONTROL MANUAL
      Manual_Fag : BOOL := FALSE;
      vmanual_Fag : LREAL := 18.0;
      Manual_Fe : BOOL := FALSE;
      vmanual_Fe : LREAL := 51;

END_VAR
```

```
//ESCALADO DE LAS SEÑALES

      //ENTRADAS DEL SISTEMA
            FIO.Fi_dig :=  LREAL_TO_WORD(Fi*(30000.0/200.0));
            FIO.Fag_dig :=  LREAL_TO_WORD(Fag*(30000.0/200.0));
            FIO.Tin_dig :=  LREAL_TO_WORD((30000.0/255.0)*(Tin+55.0));
            FIO.cS_in_dig :=  LREAL_TO_WORD(cS_in*(30000.0/100.0));
            FIO.Tin_ag_dig                                          :=
LREAL_TO_WORD((30000.0/255.0)*(Tin_ag+55.0));
            FIO.Fe_dig :=  LREAL_TO_WORD(Fe*(30000.0/200.0));

      //VARIABLES DE ESTADO
      IF FIO.V_dig > 0 THEN //Condición para solo realizar el cálculo
en caso de estar recibiendo valores en el MODBUS
            V := WORD_TO_LREAL(FIO.V_dig)/(30000.0/3000.0);
            Cx := WORD_TO_LREAL(FIO.Cx_dig)/(30000.0/100.0);
            Cs := WORD_TO_LREAL(FIO.Cs_dig)/(30000.0/100.0);
```

```
            Cp := WORD_TO_LREAL(FIO.Cp_dig)/(30000.0/100.0);
            Co2 := WORD_TO_LREAL(FIO.Co2_dig)/(30000.0/100.0);
            Tr := (255.0/30000.0)*WORD_TO_LREAL(FIO.Tr_dig)-55.0;
            Tag := (255.0/30000.0)*WORD_TO_LREAL(FIO.Tag_dig)-55.0;
      END_IF


//CÁLCULO OFFSET V
      offsetV := (Fi - Fi_ini)*Gff_Fi_V + Fe_ini;
      offsetCp := (Tin_ag - Tin_ag_ini)*Gff_Tin_ag_Cp + (Tin -
Tin_ini)*Gff_Tin_Cp + (cS_in - cS_in_ini)*Gff_cS_in_Cp + Fag_ini;
      offsetTr := (Tin_ag - Tin_ag_ini)*Gff_Tin_ag_Tr + (Tin -
Tin_ini)*Gff_Tin_Tr + (cS_in - cS_in_ini)*Gff_cS_in_Tr + Fag_ini;


// Implementación de la histeresis
IF Tr > (referencia + hysteresis1) THEN
    override := TRUE;
ELSIF Tr < (referencia + hysteresis2) THEN
    override := FALSE;
END_IF


manual_override := manual_override;
```

5.3.2 POU

```
PROGRAM POU
VAR
      PID_V: POU_2;
      PID_Cp: POU_2;
      PID_Tr: POU_2;
END_VAR
```

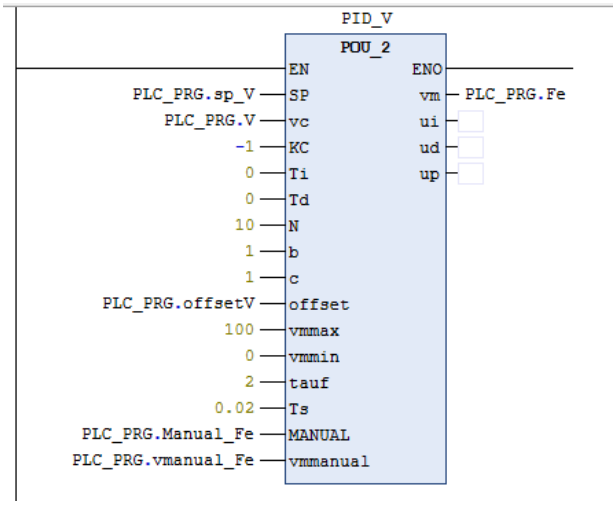

Figura 82. PID control de Volumen

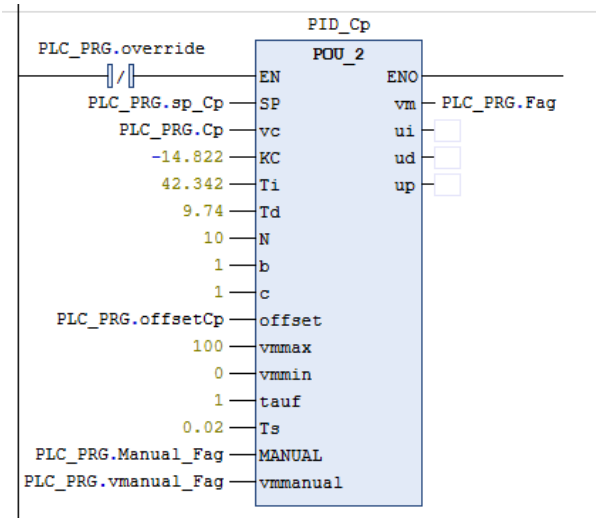

Figura 83. PID control de Etanol

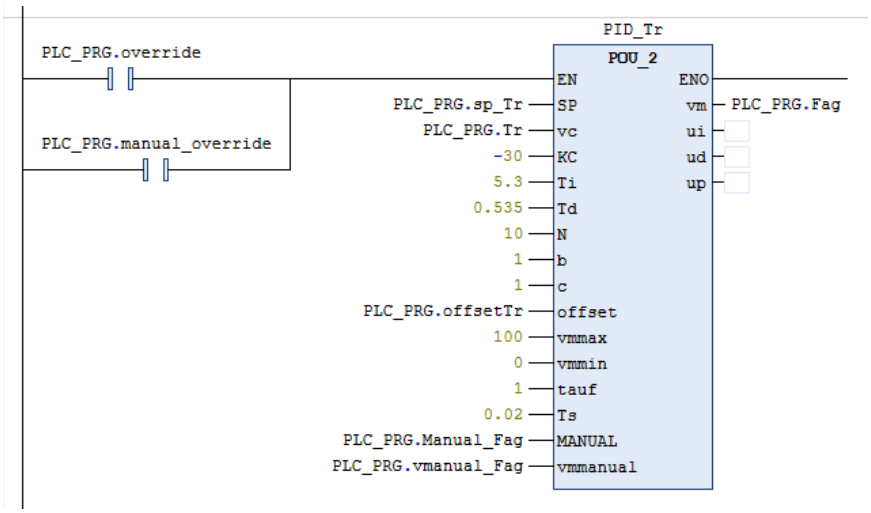

Figura 84. PID control Temperatura del reactor

5.3.3 POU_2

```
FUNCTION_BLOCK POU_2
VAR_INPUT
	SP: REAL; (* Set point *)
	vc: REAL; (* Variable controlada *)
	KC: REAL; (* Constante Proporcional *)
	Ti: REAL; (* Tiempo integral, si 0 no hay acción integral [s] *)
	Td: REAL; (* Tiempo derivada, si 0 no hay acción derivativa [s]
*)
	N: REAL;  (* Parámetro de Filtro de la derivada, si 0 no hay *)
	b: REAL;    (* Ponderación de la referencia en la acción
proporcional *)
	c: REAL;    (* Ponderación de la referencia en la acción
derivativa *)
	offset: REAL;  (* Offset, incluye puntos de funcionamiento y/o
prealimentaciones *)
	vmmax: REAL;  (* Límite superior de la variable manipulada *)
	vmmin: REAL;  (* Límite inferior de la variable manipulada *)
	tauf: REAL;   (* Constante de tiempo del filtro digital, si 0 no
hay *)
	Ts: REAL;     (* Tiempo de muestreo [s] > 0 *)
	MANUAL: BOOL; (* Pasa de manual a automatico *)
	vmmanual: REAL; (* Valor de la var manipulada en modo manual*)
END_VAR

VAR_OUTPUT
	vm: REAL;
	ui: REAL;
	ud: REAL;
	up: REAL;
END_VAR

VAR
	tiempoActual: UDINT; (*tiempo de reloj en mseg*)
	ui1: REAL := 0;
```

```
      ud1: REAL := 0;
      ek1: REAL := 0;
      ek: REAL;
      ykf: REAL;
      ykf1: REAL := 0;
      ki: REAL;
      kd: REAL;
      kf: REAL;
      kn: REAL;
      tiempoInicial: UDINT;
      Tsreal: REAL :=0;
      TsOK: BOOL;

END_VAR
```

```
(*verifica si se ha cumplido el periodo de muestreo TsOK=true*)
tiempoActual := SysTimeGetMs();
IF tiempoActual > tiempoInicial THEN
      Tsreal:= UDINT_TO_REAL(tiempoActual-tiempoInicial)/1000.0;
END_IF

IF Tsreal>=Ts THEN
      TsOK := TRUE;
      tiempoInicial := tiempoActual;
      (* Variables auxiliares: ki, kd, kf, kn *)
      IF Ti > 0.0 THEN
            ki := (KC*Tsreal)/Ti;
      ELSE
            ki := 0.0;  (* Ti = 0 desconecta integral*)
            ui1 :=0.0;
      END_IF
      IF Td > 0.0 THEN
            kd := (Kc*Td)/Tsreal;
      ELSE
```

```
            kd := 0.0; (* Td = 0 desconecta derivada*)
            ud1 := 0.0;
      END_IF
      kf:= tauf/Tsreal; (* tauf = 0 desconecta filtro var controlada*)
      IF N>0.0 THEN
            kn:= Td/(N*Tsreal);
      ELSE
            kn:= 0.0; (* N = 0 desconecta filtro derivada*)
      END_IF
ELSE
      TsOK:=FALSE;
END_IF

(* si se cumple el periodo de muestreo ejecuta las operaciones del PID
*)
IF TsOK THEN
      (* FILTRADO DE LA MEDIDA *)
      ykf:= (kf*ykf1+vc)/(kf+1);

      (* Manual o modo Automático *)
      IF MANUAL THEN
            vm:= vmmanual;
      ELSE
            (* CÁLCULO DEL TÉRMINO PROPORCIONAL *)
            up:= Kc*(b*sp-ykf);

            (* CÁLCULO DEL TÉRMINO DERIVADA *)
            ek:= c*sp-ykf;
            ud:= (kn*ud1+kd*(ek-ek1))/(kn+1);

            (* CÁLCULO DEL TÉRMINO INTEGRAL *)
            ui:= ui1+ki*(sp-ykf);

            (* CÁLCULO PID *)
```

```
            vm:= up+ui+ud+offset;

            (* Anti windup *)
            IF vm < vmmin THEN
                  ui:= ui1;
                  vm:= vmmin;
            END_IF
            IF vm > vmmax THEN
                  ui:=ui1;
                  vm:=vmmax;
            END_IF
      END_IF

      (* Actualización de variables de memoria... *)
      ui1  := ui;
      ek1  := ek;
      ud1  := ud;
      ykf1 := ykf;
END_IF
```

ÍNDICE DE FIGURAS

ÍNDICE TABLAS

Escuela Técnica Superior de Ingeniería Industrial

ETSII

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Máster Universitario en Ingeniería Industrial (Acceso desde Grado I. Electrónica Industrial y Automática)

Desarrollo e implementación de un sistema de control industrial de un biorreactor para la producción de etanol

Trabajo Fin de Máster

6. BIBLIOGRAFÍA

Autor: Daniel Pérez Gutiérrez

Tutor académico: Francesc Xavier Blasco Ferragud

Julio 2023

BIBLIOGRAFÍA

- Symbol Configuration (helpme-codesys.com)
- High Performance OPC UA Server SDK: ua_nodeid (unified-automation.com)
- IEC Symbol Set Configuration (helpme-codesys.com)
- CODESYS Forge - I/O Drivers / Documentation / Generic
- (151) how to start with igss - YouTube
- OPC UA Server (helpme-codesys.com)
- System Configuration Module | IGSS (schneider-electric.com)
- Dialog: Device Security Settings (OPC UA Server) (helpme-codesys.com)
- Configuring OPC UA (helpme-codesys.com)
- Tab: Communication, via OPC UA Server (helpme-codesys.com)
- FAQ | IGSS (schneider-electric.com)
- Uso de modelos de información OPC UA (helpme-codesys.com)
- Training | IGSS (schneider-electric.com)
- PLC Programming - Learn the basics with CoDeSys | Udemy
- Setting up CODESYS Modbus TCP (SP16 or higher) - FACTORY I/O (factoryio.com)
- CODESYS Modbus TCP/RTU integrated in the IEC 61131-3 tool
- Perform a write operation to the connected Modbus server - MATLAB write - MathWorks España
- Modbus Communication - MATLAB & Simulink - MathWorks España
- Modelo de función de transferencia - MATLAB - MathWorks España
- (159) Using Visio to Draw Process Drawings and P&IDs - Part 2 - Changing Symbol Types and Removing ID Tags - YouTube
- NiederlinskiIntro.pdf (upv.es)
- (217) Comunicaciones Industriales: Conceptos básicos - YouTube
- ¿Qué es Modbus? Una introducción al protocolo más conocido (opiron.com)
- OPC - Wikipedia, la enciclopedia libre
- ¿Que es OPC UA? Aprende en menos de 3 minutos (opiron.com)
- Espacio de estados - Wikipedia, la enciclopedia libre